(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 716 560**

(51) Int. CI.:

G06T 9/00 (2006.01)
H04N 19/105 (2014.01)
H04N 19/176 (2014.01)
H04N 19/186 (2014.01)
H04N 19/184 (2014.01)
H04N 19/98 (2014.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(96) Fecha de presentación y número de la solicitud europea: **08.07.2004 E 13172504 (6)**

(97) Fecha y número de publicación de la concesión europea: **09.01.2019 EP 2688042**

(54) Título: **Procesamiento de imágenes**

(30) Prioridad:

19.12.2003 SE 0303497

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
13.06.2019

(73) Titular/es:

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

(72) Inventor/es:

STRÖM, JACOB y
AKENINE-MÖLLER, TOMAS

(74) Agente/Representante:

LINAGE GONZÁLEZ, Rafael

DESCRIPCIÓN

Procesamiento de imágenes

Campo técnico

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

Antecedentes

La presentación y renderización de imágenes y gráficos en sistemas de procesamiento de datos y terminales de usuario, 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 aplicaciones atractivas en dichos terminales, incluidos juegos, mapas y mensajes 3D, salvapantallas e interfaces hombre-máquina.

Un proceso de procesamiento de gráficos 3D típicamente comprende tres subetapas. Brevemente, una primera etapa, la etapa de aplicación, crea varios triángulos. Las esquinas de estos triángulos se transforman, proyectan e iluminan en una segunda etapa, la etapa de geometría. En una tercera etapa, la etapa de rasterización, las imágenes, a menudo denominadas texturas, se pueden "pegar" a los triángulos, lo que aumenta el realismo de la imagen procesada. La tercera etapa también realiza la clasificación usando un búfer z.

Sin embargo, la renderización de imágenes y texturas, y en particular las imágenes y gráficos 3D, es una tarea computacionalmente costosa en términos de ancho de banda de memoria y potencia de procesamiento requerida para los sistemas gráficos. Por ejemplo, las texturas son costosas tanto en términos de memoria, las texturas deben colocarse en una memoria rápida en chip, y en términos de ancho de banda de memoria, se puede acceder a una textura varias veces para dibujar un solo píxel.

Para reducir el ancho de banda y los requisitos de potencia de procesamiento, se emplea típicamente un método o sistema de codificación de imágenes (textura). Un sistema de codificación de este tipo debería dar como resultado un uso más eficiente de la costosa memoria en chip y un menor ancho de banda de la memoria durante el procesamiento y, por lo tanto, un menor consumo de energía y/o una renderización más rápida. Esta reducción en el ancho de banda y los requisitos de potencia de procesamiento es particularmente importante para clientes livianos, como unidades móviles y teléfonos, con una pequeña cantidad de memoria, poco ancho de banda de memoria y potencia limitada (alimentado por baterías).

Delp y Mitchell [1] describen un esquema simple, llamado codificación de truncamiento de bloques (BTC), para la codificación de imágenes. Su esquema codifica imágenes en escala de grises al considerar un bloque de 4 píxeles x 4 píxeles a la vez. Para cada bloque de este tipo, se generan dos valores de escala de grises de 8 bits y cada píxel del bloque usa un solo bit para indexar una de estas escalas de grises. Esto resulta en una tasa de compresión de 2 bits por píxel (bpp). Sin embargo, la BTC sufre la producción de artefactos, sobre todo en regiones alrededor de los bordes y en áreas de bajo contraste que contienen un nivel de pendiente gris. Además, los bordes en una imagen de escala de grises procesada por BTC tienen una tendencia a ser irregulares.

Campbell et al. [2] presentaron una extensión simple de BTC, llamada compresión de células en color (CCC). En lugar de usar dos valores de escala de grises de 8 bits para cada bloque de imágenes, dos valores de 8 bits se emplean como índices en una paleta de colores. Esta paleta comprende 256 colores representados por 8 bits para cada uno de los componentes R, G y B. Un índice de 8 bits luego apunta a un color (24 bits) en la paleta. Esto permite la compresión de imágenes a 2 bpp. Sin embargo, esto requiere una búsqueda de memoria en la paleta. Además, la paleta está restringida en tamaño. El esquema CCC también introduce "dientes de sierra" de gran color y codifica escasamente el caso donde existen más de dos colores en un bloque de imágenes.

En una descripción de patente [3], Iourcha et al. divulgan un esquema de compresión de textura llamado S3TC (compresión de textura S3) o DXTC (compresión de textura DirectX), que se puede ver como una extensión de CCC. Una imagen se descompone en una serie de bloques de imágenes de 4 píxeles x 4 píxeles. Cada uno de estos bloques de imágenes se codifica en una secuencia de bits de 64 bits, lo que resulta en una tasa de compresión de 4 bpp. La secuencia de 64 bits comprende dos colores básicos o códigos de color (16 bits cada una) y una secuencia de 32 bits de índices de 2 bits, un índice para cada píxel en el bloque de imágenes. Durante la decodificación, se genera una paleta de colores de cuatro colores. Los dos primeros colores RGB (rojo, verde y azul) de la paleta corresponden a los dos colores básicos (códigos). Los dos colores adicionales, situados entre los colores básicos en el espacio RGB, se interpolan a continuación. Cada índice de 2 bits identifica, para cada píxel, uno de los cuatro colores de la paleta que se usará para ese píxel.

Aunque, el esquema S3TC funciona bastante bien para terminales de ordenador, no está bien adaptado para unidades móviles y otros clientes livianos. Tales unidades móviles típicamente solo tienen buses de memoria de 16 o 32 bits en el mejor de los casos. Por lo tanto, se requieren al menos dos, y posiblemente hasta cuatro, accesos de

memoria para leer la versión comprimida de 64 bits de un bloque de imágenes, si S3TC se implementa en una unidad móvil. Además, durante la interpolación de los dos colores adicionales de la paleta de colores, se realiza la multiplicación por 1/3 y 2/3, lo que no es ideal en hardware. La compresión que usa S3TC también consume bastante tiempo, al menos en un terminal móvil.

Akenine-Möller y Ström [4] han desarrollado una variante de S3TC, llamada POOMA, que está dirigida específicamente a teléfonos móviles. En POOMA, un bloque de imágenes de 3 píxeles x 2 píxeles se codifica en 32 bits, dando 5,33 bpp. La representación codificada de 32 bits de un bloque de imágenes se adapta a los buses de memoria de los teléfonos móviles, que típicamente son de 32 bits como máximo. Por lo tanto, un píxel se puede renderizar usando solo un acceso a la memoria en comparación con dos accesos para S3TC. POOMA usa dos colores base, pero solo un color adicional se interpola entre los colores base, lo que da como resultado una paleta de colores de tres colores.

Una desventaja importante de POOMA es el tamaño de bloque de 3 × 2 píxeles. Como resultado, el cálculo de la dirección de bloque para un píxel o téxel particular (elemento de textura) requiere una división por 3, lo que no es ideal para el diseño de hardware. Además, los anchos y las alturas de las texturas (imágenes) en gráficos son siempre típicamente una potencia de dos, lo que significa que un ancho de bloque de 3 es inapropiado. En cuanto a S3TC, la codificación que usa POOMA consume bastante tiempo, en particular cuando se implementa en un terminal móvil. Fenny [5] divulga un esquema de codificación de imágenes usado en la plataforma de hardware de gráficos MBX para teléfonos móviles. Este esquema usa dos imágenes de baja resolución, donde una imagen suele ser una versión filtrada de paso bajo de la imagen original. Durante la decodificación, se crea un aumento bilineal (mejora de escala) de estas dos imágenes. Cada píxel también almacena un factor de mezcla entre estas dos imágenes en escala superior. Se usan 64 bits para codificar cada bloque de imágenes y se describe una tasa de compresión de 2 bpp y 4 bpp. Se necesita información de los bloques de imágenes vecinos, lo que complica la decodificación.

Sumario

La presente invención supera estos y otros inconvenientes de las disposiciones de la técnica anterior.

Es un objeto general de la presente invención proporcionar un procesamiento de imágenes eficiente.

Otro objeto de la invención es proporcionar una codificación de imágenes y decodificación de imágenes eficientes.

Otro objeto más de la invención es proporcionar codificación y decodificación de imágenes adaptadas para uso en clientes livianos con poca memoria y capacidad de ancho de banda de memoria.

Un objeto adicional de la invención es proporcionar codificación y decodificación de imágenes adaptadas para gráficos e imágenes tridimensionales (3D).

La invención cumple con estos y otros objetos tal como se definen en las reivindicaciones de patente adjuntas.

Brevemente, la presente invención implica el procesamiento de imágenes en forma de codificación (compresión) de una imagen y decodificación (descompresión) de una imagen codificada (comprimida).

De acuerdo con la invención, una imagen a codificar se descompone en una serie de bloques de imágenes que comprenden múltiples elementos de imagen (píxeles o elementos de textura, téxeles). Un bloque de imágenes comprende preferiblemente ocho elementos de imagen y tiene un tamaño de $2^m \times 2^n$ elementos de imagen, donde m = 3-n y n = 0, 1, 2, 3. Cada elemento de imagen en un bloque se caracteriza por un color, por ejemplo, un color RGB (rojo, verde, azul) de 12 bits. Los bloques de imágenes son codificados después.

En esta codificación de bloques perdida, se determina un código de color para el bloque de imágenes. El código de color es una representación de los colores de los elementos de imagen del bloque de imágenes. Una representación preferida es un valor promedio de los colores de los elementos de imagen en bloque, cuantificados en 12 bits (4 bits para cada uno de los tres componentes de color para un color RGB). Por lo tanto, se genera un mismo código de color (es decir, representación de color) para un bloque de imágenes, es decir, para todos los elementos de imagen del bloque de imágenes. A partir de entonces, se proporciona un código de intensidad para el bloque de imágenes. Este código de intensidad es una representación de un conjunto de múltiples modificadores de intensidad que se usan (durante la decodificación) para modificar o modular la intensidad de los elementos de imagen en el bloque de imágenes. Una vez que se proporciona el código de intensidad, se seleccionan las representaciones de intensidad para los elementos de imagen en el bloque de imágenes. Cada representación de intensidad de este tipo está asociada con un modificador de intensidad del conjunto de modificador de intensidad. En otras palabras, la representación de intensidad permite identificar qué modificador de intensidad del conjunto se debe usar para un elemento de imagen específico del bloque.

El bloque de imágenes codificada resultante luego comprende el código de color, preferiblemente 12 bits, el código de intensidad, preferiblemente 4 bits, y una secuencia de las representaciones de intensidad, preferiblemente 8 × 2 = 16 bits. El tamaño resultante de un bloque de imágenes codificadas es, por lo tanto, solo 32 bits y se obtiene una tasa de compresión de 4 bits por píxel (elemento de imagen). Este pequeño tamaño de 32 bits está bien adaptado para clientes livianos, como unidades móviles y teléfonos, que típicamente tienen buses de memoria de 16 o 32 bits. Como resultado, solo se necesitan uno o, en el peor de los casos, dos accesos de memoria para leer el bloque de imágenes codificadas desde una ubicación de memoria.

En una realización preferida de la invención, el código de intensidad es un índice de intensidad que permite la identificación de un conjunto de modificador de intensidad. Este índice podría identificar o apuntar al conjunto en una tabla o libro de códigos que comprende varios conjuntos de modificador de intensidad diferentes. Cada conjunto comprende preferiblemente cuatro valores de modificador complementarios (matemáticamente). En tal caso, los conjuntos de modificador almacenados en la tabla solo tienen que comprender dos valores diferentes de modificador de intensidad cada uno y luego los otros dos valores (complementarios) del conjunto podrían calcularse a partir de ellos. Además, la tabla de intensidad comprende preferiblemente ambos conjuntos que incluyen valores de modificador de pequeña intensidad, que están adaptados para permitir la representación de superficies que cambian suavemente, y conjuntos que incluyen valores de modificador de gran intensidad, que están adaptados para permitir la representación de bordes afilados.

Durante la decodificación, el bloque o bloques de imágenes codificadas que deben decodificarse se identifica y se obtiene, por ejemplo, de una ubicación de memoria. Una vez que se identifica el bloque de imágenes codificadas correcto, se proporciona un conjunto de modificador de intensidad. Este conjunto de modificador se proporciona basándose en el código de intensidad en el bloque de imágenes codificadas. Esta disposición de conjunto se realiza preferiblemente identificando, por medio del código de intensidad, un conjunto de modificador de intensidad de una tabla de intensidad que comprende múltiples conjuntos de modificador.

A partir de entonces, se genera una representación de color para al menos uno de los elementos de imagen del bloque de imágenes. Esta generación de color se realiza basándose en el código de color en la representación de bloques codificada. Luego se selecciona el modificador de intensidad a usar para el elemento de imagen que debe decodificarse. El valor de modificador se selecciona del conjunto de modificador proporcionado basándose en la representación de intensidad asociada con el elemento de imagen y se encuentra en la secuencia de representación del bloque de imágenes codificadas. Una vez que se selecciona el valor de modificador de intensidad correcto, la intensidad del elemento de imagen se modifica con este valor.

La selección de modificador de intensidad y la modificación de la intensidad se realizan preferiblemente para todos los elementos de imagen que deben decodificarse en el bloque de imágenes codificadas actual. La decodificación de bloques se repite preferiblemente para todos los bloques de imágenes que comprenden elementos de imagen que deben decodificarse. A partir de entonces, se puede generar una representación decodificada de una imagen original, o una porción de la misma, basándose en los elementos y bloques de imágenes decodificadas.

La representación de color se genera preferiblemente expandiendo los tres componentes de color de 4 bits del código de color en tres componentes de 8 bits. El color de 24 bits resultante se asigna al elemento o elementos de imagen del bloque de imágenes que se van a decodificar. La intensidad de los elementos de imagen se modifica preferiblemente añadiendo o multiplicando el modificador de intensidad a cada componente de color, o cada componente de color se modula de otra manera con el modificador de intensidad. A partir de entonces, los componentes de color modificados de intensidad resultantes se fijan entre un valor de umbral mínimo y máximo.

La codificación y decodificación de imágenes de la invención se puede aplicar a varios tipos diferentes de imágenes, incluyendo imágenes "sintéticas" 1D, 2D y 3D, fotos, texto, juegos, mapas y escenas 3D, mensajes 3D, por ejemplo, mensajes animados, salvapantallas, interfaces hombre-máquina (MMI), etc.

Debido al pequeño tamaño (32 bits) de un bloque de imágenes codificadas, la invención se adapta bien para clientes livianos con capacidad de memoria y ancho de banda limitados. Además, la codificación es muy rápida, por lo que también se puede realizar en terminales con bajas frecuencias de reloj. Además, la decodificación se puede implementar de manera extremadamente simple, por ejemplo, hardware usando solo unos pocos componentes estándar.

La invención ofrece las siguientes ventajas:

- bien adaptada para coincidir con el sistema visual humano, ya que el componente de luminancia se conserva mejor que los componentes de crominancia;

- proporciona alta calidad (relación señal/ruido) para diferentes tipos de imágenes;

- la implementación de hardware de decodificación es extremadamente simple;

- la codificación es muy rápida, lo que permite implementaciones también en terminales con bajas frecuencias de reloj;

- la codificación exhaustiva es posible a una velocidad factible en un ordenador; y

- los datos de imágenes codificadas tienen un tamaño adecuado para clientes livianos con capacidad de memoria y ancho de banda limitados.

Otras ventajas ofrecidas por la presente invención se apreciarán al leer la siguiente descripción de las realizaciones de la invención.

Breve descripción de los dibujos

La invención, junto con otros objetos y ventajas de la misma, puede entenderse mejor haciendo referencia a la siguiente descripción tomada junto con los dibujos adjuntos, en los que:

la figura 1 es un diagrama de flujo que ilustra un método de codificación de imágenes de acuerdo con la presente invención;

la figura 2 ilustra una realización de un bloque de imágenes de acuerdo con la presente invención;

la figura 3 ilustra otra realización de un bloque de imágenes de acuerdo con la presente invención;

la figura 4 ilustra una realización de una representación codificada de un bloque de imágenes de acuerdo con la presente invención;

la figura 5 es un diagrama de flujo que ilustra el paso de determinación de códigos de color de la figura 1 con más detalle;

la figura 6 es un diagrama de flujo que ilustra el paso para proporcionar el código de intensidad y el paso de selección de representación de intensidad de la figura 1 con más detalle;

la figura 7 es un diagrama de flujo que ilustra un método de decodificación de imágenes de acuerdo con la presente invención;

la figura 8 es un diagrama de flujo que ilustra los pasos para proporcionar un conjunto de modificador de intensidad y generar una representación de color de la figura 7 con más detalle;

la figura 9 es un diagrama de flujo que ilustra el paso de modificación de intensidad de la figura 7 con más detalle;

la figura 10 ilustra esquemáticamente un ejemplo de un terminal de usuario con un codificador y decodificador de imágenes de acuerdo con la presente invención;

la figura 11 es un diagrama de bloques que ilustra esquemáticamente una realización de un codificador de imágenes de acuerdo con la presente invención;

la figura 12 es un diagrama de bloques que ilustra esquemáticamente otra realización de un codificador de imágenes de acuerdo con la presente invención;

la figura 13 es un diagrama de bloques que ilustra esquemáticamente una realización de un codificador de bloques de acuerdo con la presente invención;

la figura 14 es un diagrama de bloques que ilustra esquemáticamente otra realización de un codificador de bloques de acuerdo con la presente invención;

la figura 15 es un diagrama de bloques que ilustra esquemáticamente el cuantificador de color del codificador de bloques de las figuras 13 y 14 con más detalle;

la figura 16 es un diagrama de bloques que ilustra esquemáticamente una realización de un decodificador de imágenes de acuerdo con la presente invención;

la figura 17 es un diagrama de bloques que ilustra esquemáticamente otra realización de un decodificador de imágenes de acuerdo con la presente invención;

la figura 18 es un diagrama de bloques que ilustra esquemáticamente una realización de un decodificador de bloques de acuerdo con la presente invención;

la figura 19 es un diagrama de bloques de hardware que ilustra esquemáticamente una realización de un decodificador de bloques de acuerdo con la presente invención;

la figura 20 es un diagrama de bloques de hardware que ilustra una realización de los extensores de bits de la figura 19 con más detalle;

la figura 21 es un diagrama de bloques de hardware que ilustra una realización de la tabla de consulta de la figura 19 con más detalle; y

la figura 22 es un diagrama de bloques de hardware que ilustra una realización de los fijadores de la figura 19 con más detalle.

Descripción detallada

A lo largo de los dibujos, se usarán los mismos caracteres de referencia para elementos correspondientes o similares.

La presente invención se refiere al procesamiento de imágenes y gráficos, y en particular a la codificación o compresión de imágenes y a la decodificación o descompresión de imágenes codificadas (comprimidas).

En general, de acuerdo con la invención, durante la codificación de imágenes, una imagen se descompone o se divide en varios bloques de imágenes. Cada uno de estos bloques de imágenes comprende múltiples elementos de imagen que tienen, entre otros, un cierto color. Los bloques de imágenes se codifican después para generar una representación codificada de la imagen.

Cuando posteriormente se va a renderizar una imagen codificada o un primitivo gráfico, por ejemplo, mostrada en una pantalla, los elementos de imagen relevantes de los bloques de imágenes codificadas se identifican y decodifican. Estos elementos de imagen decodificada se usan luego para generar una representación decodificada de la imagen original o primitivos gráficos.

La presente invención está bien adaptada para su uso con gráficos tridimensionales (3D), tales como juegos, mapas y escenas 3D, mensajes 3D, por ejemplo, mensajes animados, salvapantallas, interfaces hombre-máquina (MMI), etc., pero no se limitan a estos. Por lo tanto, la invención también podría emplearse para codificar otros tipos de imágenes o gráficos, por ejemplo, imágenes unidimensionales (1D) o bidimensionales (2D).

En el procesamiento de gráficos 3D, típicamente se crean varios triángulos y se determinan las coordenadas de pantalla correspondientes de las esquinas de estos triángulos. En cada triángulo, una imagen (o porción de una imagen), o una llamada textura, es mapeada ("pegada"). 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 la memoria, cuando las texturas se recuperan de la memoria. Este es un problema particularmente para clientes livianos, como unidades móviles y teléfonos, con capacidad de memoria y ancho de banda limitados. Como consecuencia, a menudo se emplea un esquema de codificación de textura o imagen. En tal esquema, una textura típicamente se descompone o se divide en una serie de bloques de imágenes que comprenden múltiples téxeles. Los bloques de imágenes se codifican y almacenan en una memoria. Téngase en cuenta que el tamaño de un bloque de imágenes codificadas (o una versión de este) es más pequeño que el tamaño correspondiente de la versión no codificada del bloque de imágenes.

En la presente invención, la expresión "elemento de imagen" se refiere a un elemento en un bloque de imágenes o representación codificada de un bloque de imágenes. Este bloque de imágenes, a su vez, corresponde a una porción de una imagen o textura. Así, de acuerdo con la invención, un elemento de imagen podría ser un téxel (elemento de textura) de una textura (1D, 2D o 3D) o un píxel de una imagen (1D, 2D o 3D). En general, un elemento de imagen se caracteriza por ciertas propiedades de elemento de imagen, como un valor de color. Además, a continuación, el término "imagen" se usa para indicar cualquier imagen o textura 1D, 2D o 3D que pueda codificarse y decodificarse por medio de la presente invención, incluidas, entre otras, fotos, texturas de tipo juego, texto, dibujos, etc.

La figura 1 ilustra un método (con pérdida) para codificar una imagen de acuerdo con la presente invención. En un primer paso S1, la imagen se descompone o se divide en varios bloques de imágenes. Cada uno de estos bloques de imágenes comprende múltiples elementos de imagen. En una realización preferida de la invención, un bloque de imágenes comprende ocho elementos de imagen (píxeles o téxeles) y tiene un tamaño de $2^m \times 2^n$ elementos de imagen, donde m = 3-n y n = 0, 1, 2, 3. Más preferiblemente, n es 1 o 2. Las figuras 2 y 3 ilustran esquemáticamente dos ejemplos de un bloque 600 de imágenes con ocho elementos 610 de imagen de acuerdo con la presente invención. En la figura 2, la altura es dos elementos 610 de imagen y el ancho es cuatro elementos 610 de imagen, es decir, m = 1 y n = 2, mientras que para el bloque 600 de imágenes en la figura 3 m = 2 y n = 1. Volviendo a la figura 1, el bloque de imágenes completo se descompone preferiblemente en bloques de imágenes (no

superpuestos) en el paso S1. Sin embargo, en algunas aplicaciones, solo una porción de una imagen está codificada y, por lo tanto, solo esta porción se descompone en bloques de imágenes.

Los siguientes pasos S2 a S4 realizan una codificación o compresión de los bloques de imágenes. En primer lugar, en el paso S2, se determina un código de color para un bloque de imágenes. Este código de color es una representación de los colores de los elementos de imagen en el bloque de imágenes. En una realización preferida, el código de color es una representación de un color promedio de los elementos de imagen del bloque. El color puede ser un color RGB (rojo, verde, azul), un color en el espacio YUV o espacio YCrCb, o cualquier otro espacio de color propietario usado en el procesamiento y la gestión de imágenes y gráficos. El código de color está preferiblemente en el mismo formato de color (espacio) que la imagen. Sin embargo, en algunos casos, puede ser útil convertir la imagen a un formato de color diferente, es decir, tener el código de color en un primer espacio de color y la imagen original en un segundo espacio de color diferente. El código de color es preferiblemente una secuencia de representación de color de 12 bits. Por ejemplo, un código de color RGB podría comprender 4 bits para el componente de color rojo, 4 bits para el componente verde y 4 bits para el componente azul. De manera correspondiente, un código de color YUV podría incluir 6 bits, 3 bits y 3 bits, respectivamente, para los tres componentes diferentes.

Téngase en cuenta que se genera un código del mismo color (es decir, representación de color) para un bloque de imágenes, es decir, para todos los elementos de imagen del bloque de imágenes.

A partir de entonces, se proporciona un código de intensidad en el paso S3. Este código de intensidad es una representación de un conjunto de múltiples modificadores de intensidad que se usan (durante la decodificación) para modificar la intensidad de los elementos de imagen en el bloque de imágenes.

En una realización preferida de la invención, el código de intensidad es un índice de intensidad que permite la identificación de un conjunto de modificador de intensidad. Este índice podría identificar o apuntar al conjunto en una tabla o libro de códigos que comprende varios conjuntos de modificador de intensidad diferentes. Cada conjunto comprende dos o más valores de modificador de intensidad, preferiblemente al menos cuatro valores de modificador. Además, los valores de modificador de un conjunto son, preferiblemente, valores matemáticamente complementarios, es decir, cada conjunto es preferiblemente simétrico. Por ejemplo, un posible conjunto de modificador de intensidad podría ser [-a, -b, b, a], donde a y b son enteros positivos y a> b.

La tabla de intensidad comprende preferiblemente conjuntos que incluyen valores de modificador de intensidad pequeña, que están adaptados para permitir la representación de superficies que cambian suavemente. Además, la tabla preferiblemente también comprende conjuntos que incluyen valores de modificador de gran intensidad, que están adaptados para permitir la representación de bordes afilados.

Los valores de modificador de intensidad reales de los conjuntos en la tabla se pueden encontrar comenzando con valores aleatorios y luego optimizando estos valores usando una serie de diferentes esquemas de optimización y algoritmos, como las versiones del algoritmo LBG (Linde, Buzo y Gray) [6], recocido simulado y búsqueda de coordenadas, que son conocidos por un experto en la técnica. Un puñado de imágenes de diferentes tipos, por ejemplo, se pueden usar fotos, texturas de tipo juego, texto, etc., como datos de entrenamiento.

Para hacer que una implementación de hardware de la tabla de intensidad sea menos costosa, se puede obligar a los modificadores de intensidad de un conjunto a que sean simétricos, como se explicó anteriormente, y/o los modificadores de intensidad de un conjunto dado podrían ser una copia de los modificadores de intensidad de otro conjunto modificado por un factor, por ejemplo, dos.

La tabla 1 ilustra un ejemplo actualmente preferido de una tabla de intensidad que comprende 16 conjuntos de modificador de intensidad, con cuatro valores de modificador en cada conjunto.

Tabla 1

Conjunto	Código	Valor de modificador de intensidad			
1	0000_{bin}	-8	-2	2	8
2	0001_{bin}	-12	-4	4	12
3	0010_{bin}	-31	-6	6	31
4	0011_{bin}	-34	-12	12	34
5	0100_{bin}	-50	-8	8	50
6	0101_{bin}	-47	-19	19	47
7	0110_{bin}	-80	-28	28	80
8	0111_{bin}	-127	-42	42	127
9	1000_{bin}	-16	-4	4	16
10	1001_{bin}	-24	-8	8	24
11	1010_{bin}	-62	-12	12	62
12	1011_{bin}	-68	-24	24	68
13	1100_{bin}	-100	-16	16	100
14	1101_{bin}	-94	-38	38	94
15	1110_{bin}	-160	-56	56	160
16	1111_{bin}	-254	-84	84	254

En la tabla 1, los conjuntos 9-16 de modificador de intensidad son una copia de los conjuntos 1-8 multiplicados por un factor de dos.

Si la tabla de intensidad comprende como máximo 16 conjuntos de modificador de intensidad diferentes, el código de intensidad es preferiblemente un índice de 4 bits (0000_{bin}-1111_{bin}) que identifica uno de los (16) conjuntos, por ejemplo, [-8, -2, 2, 8] para el código 0000_{bin} (0000 base 2), de la tabla. Debido a la elección cuidadosa de los valores de modificador en los conjuntos (los conjuntos simétricos y la mitad de los conjuntos son un factor dos de la mitad restante), la tabla 1 completa se puede reconstruir usando solo 16 valores de modificador, y los 48 valores restantes se podrían calcular a partir de estos.

Sin embargo, la presente invención no está limitada al uso de la tabla 1, sino que podría usar otras tablas con otros conjuntos y valores de modificador de intensidad. Además, para más o menos de 16 conjuntos en una tabla, el tamaño del código de intensidad podría tener que cambiarse. Por ejemplo, si la tabla comprende dos (3-4, 5-8 o más de 16) conjuntos de modificador de intensidad, el tamaño del código podría limitarse a un bit (dos bits, tres bits o más de cuatro bits). Además, el número de valores de modificador de intensidad por conjunto podría diferir de cuatro, por ejemplo, se podrían usar cinco valores por conjunto, dando un ejemplo de [-8, -2, 0, 2, 8]. Los valores de intensidad de los conjuntos en la tabla podrían determinarse usando varios tipos diferentes de imágenes como datos de entrenamiento, como se explicó anteriormente. Sin embargo, si solo se va a codificar un tipo de imagen específico, los valores de modificador podrían determinarse usando los datos de entrenamiento correspondientes a ese tipo de imagen, es decir, dando una tabla de intensidad dedicada para un tipo de imagen específico. También podría ser posible tener una tabla de intensidad con valores de modificador de intensidad adaptados para una imagen específica. En estos casos, es decir, la tabla dedicada a la imagen o el tipo de imagen, podría ser necesario incluir los valores de modificador de intensidad de la tabla en el archivo comprimido de los bloques de imágenes codificadas o asociarlos de otro modo.

Además, el código de intensidad no tiene que ser un índice o puntero a un conjunto de modificador de intensidad en una tabla, pero en realidad podría ser un conjunto de modificador de intensidad en sí, por ejemplo, comprende dos valores de modificador, como 2 y 8, y donde los otros valores de modificador, como -2 y -8, se pueden determinar a partir de estos dos valores.

Téngase en cuenta que se usa el mismo conjunto de modificador de intensidad para el bloque de imágenes.

Una vez que se proporciona el código de intensidad en el paso S3, el siguiente paso S4 selecciona las representaciones de intensidad para los elementos de imagen en el bloque de imágenes. Cada representación de intensidad de este tipo está asociada con un valor de modificador de intensidad del conjunto de modificador de intensidad proporcionado en el paso S3. En otras palabras, la representación de intensidad permite identificar qué modificador de intensidad del conjunto se debe usar para un elemento de imagen específico del bloque.

En el caso de un conjunto de modificador de intensidad que comprende cuatro valores de modificador, como -8, -2, 2, 8, la representación de intensidad podría ser una secuencia de 2 bits que identifica uno de estos cuatro valores,

por ejemplo, 11_{bin} corresponde a -8, 10_{bin} corresponde a -2, 00_{bin} corresponde a 2 y 01_{bin} corresponde a 8. Si se usan más de cuatro valores de modificador de intensidad por conjunto, entonces se requieren más de dos bits para que cada elemento de intensidad identifique el modificador correcto.

El paso S4 se repite preferiblemente para todos los elementos de imagen en el bloque de imágenes (ilustrado esquemáticamente por la línea 1). El resultado de la codificación de los pasos S2 a S4 es un bloque de imágenes codificadas o, más precisamente, una representación codificada (comprimida) del bloque de imágenes. Tal representación 700 de bloques codificada se ilustra en la figura 4. La representación 700 (bloque de imágenes codificadas) comprende el código 710 de color, el código 720 de intensidad y una secuencia 730 o mapa de bits de representaciones de intensidad (preferiblemente una representación de intensidad para cada elemento de imagen en el bloque). Téngase en cuenta que el orden mutuo del código 710 de color, el código 720 de intensidad y la secuencia 730 de representación de intensidad del bloque 700 de imágenes codificadas pueden diferir de lo que se ilustra en la figura.

Si el bloque de imágenes comprende ocho elementos de imagen (véanse, por ejemplo, las figuras 2 y 3) y cada representación de intensidad es de 2 bits, el tamaño de la secuencia 730 es de 16 bits. Además, supóngase que los tamaños correspondientes de los códigos de color e intensidad son 12 y 4 bits, respectivamente. El tamaño total de la representación codificada 700 del bloque de imágenes es entonces de 32 bits y se obtiene una tasa de compresión de 4 bits por píxel (elemento de imagen) (bpp). Este pequeño tamaño (32 bits) de la representación 700 está bien adaptado para clientes livianos, como las unidades móviles, que típicamente tienen buses de memoria de 16 o 32 bits. Como resultado, solo se necesitan uno o, en el peor de los casos, dos accesos de memoria para leer la representación codificada 700.

Volviendo a la figura 1, los pasos S2 a S4 se repiten preferiblemente para todos los bloques de imágenes proporcionados durante la descomposición del paso S1 (ilustrado esquemáticamente por la línea 2). El resultado es entonces una secuencia o archivo de bloques de imágenes codificadas. Los bloques de imágenes codificadas resultantes (representaciones codificadas de los bloques de imágenes) se pueden ordenar en un archivo de izquierda a derecha y de arriba a abajo en el mismo orden en que se desglosaron en el bloque de descomposición del paso S1. El método entonces termina.

La imagen codificada podría luego proporcionarse a una memoria para almacenarla en ella hasta una renderización posterior, por ejemplo, la visualización de la imagen. Además, la imagen codificada podría proporcionarse como una señal de representaciones de bloques codificadas a un transmisor para la transmisión (inalámbrica o por cable) a otra unidad.

La figura 5 ilustra una realización del paso S2 de la figura 1 con más detalle. En el paso S10, se determina un color promedio de los elementos de imagen en el bloque de imágenes. A continuación, se supone que el color de un píxel o téxel (elemento de imagen) de una imagen está representado por 24 bits de color RGB, es decir, 8 bits del componente rojo, 8 bits del componente verde y 8 bits del componente azul Sin embargo, la invención no se limita a este ejemplo particular, sino que puede aplicarse a cualquier representación de color de píxeles y téxeles. El color promedio $(\overline{R}, \overline{G}, \overline{B})$ se determina entonces como:

$$\overline{R} = \frac{1}{N}\sum_{i=1}^{N} R_i$$

$$\overline{G} = \frac{1}{N}\sum_{i=1}^{N} G_i \qquad (1)$$

$$\overline{B} = \frac{1}{N}\sum_{i=1}^{N} B_i ,$$

donde R_i, G_i, B_i son el componente R, G, B del elemento de imagen i y N es el número total de elementos de imagen en el bloque de imágenes.

Una vez que se determina el color promedio $(\overline{R}, \overline{G}, \overline{B})$ en el paso S10, el siguiente paso S11 cuantifica el color promedio. El color promedio (de 24 bits) se cuantifica preferiblemente en una secuencia de 12 bits (código de color). En otras palabras, cada componente promedio de 8 bits se cuantifica en un componente promedio de 4 bits. Por ejemplo, si el color promedio $(\overline{R}, \overline{G}, \overline{B})$ se calcula para:

$$\begin{bmatrix} 178 \\ 88 \\ 21 \end{bmatrix} = \begin{bmatrix} B2 \\ 58 \\ 15 \end{bmatrix}_{hex} = \begin{bmatrix} 10110010 \\ 01011000 \\ 00010101 \end{bmatrix}_{bin},$$

una versión cuantificada de 4 bits $(\hat{\bar{R}}, \hat{\bar{G}}, \hat{\bar{B}})$ podría generarse a partir de:

$$\begin{bmatrix} 170 \\ 85 \\ 17 \end{bmatrix} = \begin{bmatrix} AA \\ 55 \\ 11 \end{bmatrix}_{hex} = \begin{bmatrix} 10101010 \\ 01010101 \\ 00010001 \end{bmatrix}_{bin},$$

es decir $[A, 5, 1]_{hex} = [1010, 0101, 0001]_{bin}$ podría usarse como un código de color (12 bits). El método luego continúa con el paso S3 de la figura 1.

La figura 6 ilustra una realización de los pasos S3 y S4 de la figura 1 con más detalle. El paso S20 investiga los diferentes conjuntos de modificador de intensidad de la tabla y los diferentes valores de modificador de los conjuntos y calcula un valor de error para cada uno de dichos conjuntos de modificador y cada prueba de valor de modificador. Basándose en estos valores de error, en el paso S21 se seleccionan un conjunto de modificador y los valores de modificador de intensidad del conjunto que resultan en un valor de error más pequeño. Esto se describe en más detalle a continuación. El método entonces termina.

A continuación, la codificación de los bloques de imágenes se explica con más detalle en relación con tres ejemplos diferentes. En estos ejemplos, se usa una tabla de intensidad correspondiente a la tabla 1.

Codificación simple

Para codificar un bloque de imágenes de acuerdo con esta realización de la invención, se seleccionan básicamente un código de color y un conjunto de modificador de intensidad correctos. Una vez hecho esto, la codificación de cada elemento de imagen en el bloque de imágenes se realiza probando los cuatro modificadores de intensidad del conjunto y calculando el error. Supóngase que el color original (24 bits) de un elemento de imagen es (R, G, B) y que el código de color (color promedio cuantificado, 12 bits) es $(\hat{\bar{R}}, \hat{\bar{G}}, \hat{\bar{B}})$ y que el conjunto de modificador elegido es [-a, -b, b, a]. Un valor de error podría entonces determinarse como:

$$\varepsilon^2 = \left(\hat{\bar{R}} + \alpha - R\right)^2 + \left(\hat{\bar{G}} + \alpha - G\right)^2 + \left(\hat{\bar{B}} + \alpha - B\right)^2, \qquad (2)$$

donde $\alpha \in$ [-a, -b, b, a]. Para cada uno de los elementos de imagen en el bloque de imágenes, se selecciona el α que minimiza el error ε^2. Esto podría implementarse calculando el valor de error para una primera selección de valores de modificador para los elementos de imagen y almacenar este primer valor de error (y la selección elegida de valores de modificador). A partir de entonces, se calcula un valor de error para una selección diferente de valores de modificador. Este valor de error se compara con el valor almacenado. Si es menor que el valor almacenado, el valor almacenado se reemplaza por este nuevo valor de error y los valores de modificador usados en el cálculo de este valor de error también se almacenan. Esto se repite para todas las combinaciones de modificadores y conjuntos.

La ecuación (2) dará el mejor rendimiento en términos de relación señal/ruido (PSNR), ya que minimizará el error cuadrático medio en la imagen. Perceptualmente, sin embargo, puede que no produzca el mejor resultado. Para algunos píxeles (elementos de imagen) p_1 y p_2, donde p_1 parece más brillante que p_2 en la imagen original, p_2 puede parecer más brillante que p_1 en una versión decodificada de la imagen. La razón de esto es que los componentes rojo, verde y azul no aportan contribuciones iguales a lo que el sistema visual humano percibe como intensidad. Dado que el componente verde proporciona una contribución desproporcionada a la percepción de la intensidad, su aproximación debe representarse con mayor precisión (más exacta) que las del rojo y el azul. Como resultado, se podría emplear un valor de error ponderado, como:

$$\varepsilon^2 = w_R\left(\hat{\bar{R}} + \alpha - R\right)^2 + w_G\left(\hat{\bar{G}} + \alpha - G\right)^2 + w_B\left(\hat{\bar{B}} + \alpha - B\right)^2, \qquad (3)$$

donde w_R, w_G, w_B son diferentes ponderaciones para los componentes de color. Además, w_G es preferiblemente más grande que w_R y w_B. Por ejemplo, $w_R = \frac{5}{16}$, $w_G = \frac{9}{16}$ y $w_B = \frac{2}{16}$, o w_R = 0,299, w_G = 0,587 y w_B = 0,114.

En esta codificación simple, un color promedio de ocho elementos de imagen en el bloque, cuantificado a 4 bits por componente de color, se usa como código de color. El conjunto de modificador de intensidad correcto se elige mediante una búsqueda exhaustiva, es decir, se prueban todos los 16 conjuntos de la tabla y se selecciona el conjunto que minimiza el valor de error. Esto requiere 16 x 4 = 64 evaluaciones por elemento de imagen. Si las ponderaciones son $w_R = \frac{5}{16}$, $w_G = \frac{9}{16}$ y $w_B = \frac{2}{16}$, se puede usar aritmética entera y la codificación se vuelve rápida. Para esta selección de ponderaciones, la codificación de una imagen de 128 x 128 píxeles (elementos de imagen) usando la codificación simple lleva alrededor de 60 ms en un portátil PC a 1,2 GHz.

Codificación exhaustiva

En la codificación simple descrita anteriormente, el color promedio cuantificado se usó simplemente como una representación (código de color) de los colores de los elementos de imagen en el bloque de imágenes. En esta realización de la codificación exhaustiva de acuerdo con la invención, se eligen los colores y los conjuntos de modificador de intensidad (incluidos los valores de modificador), es decir, se prueban todas las combinaciones posibles. Para un elemento de imagen dado, se añade una iteración adicional a través de los 12 bits de color además de la iteración anterior de los 4 bits del conjunto de modificador de intensidad y los 2 bits de la representación de intensidad, que juntos dan 2^{18} pasos. La codificación de una imagen de 128 × 128 píxeles lleva aproximadamente 5 minutos usando el mismo portátil PC que para la compresión simple. Aunque esto puede ser demasiado largo para las aplicaciones en tiempo de ejecución, no es prohibitivo para la codificación fuera de línea. La comparación de los resultados de la codificación exhaustiva con los de la codificación simple anterior muestra una diferencia en PSNR de aproximadamente 1,5 dB. Visualmente, las imágenes difieren en que algunas áreas de las imágenes codificadas con el esquema de codificación simple muestran cambios de color. Los bloques de una sola imagen en áreas de lo contrario grises de repente obtienen un ligero tono verde. La razón de esto es que, en el esquema simple, los componentes R, G y B del código de color se cuantifican individualmente, lo cual es insuficiente.

Cuantificación combinada

En cuanto al esquema de codificación simple, esta realización de la presente invención comienza con un color promedio (24 bits) $(\overline{R}, \overline{G}, \overline{B})$, pero los componentes de color de este color promedio se cuantifican junto con los componentes de intensidad, es decir, la selección de conjuntos y valores de modificador de intensidad.

Si R_{low} and R_{high} indican los niveles de cuantificación de 4 bits o valores que están directamente por debajo y por encima de $\overline{R}$, respectivamente, de modo que $R_{low} \leq R \leq R_{high}$. La tarea es entonces elegir $\hat{\overline{R}}$ como R_{low} o R_{high}. Lo mismo es cierto para los componentes verde y azul.

En primer lugar, el valor de error se calcula con $(\hat{\overline{R}}, \hat{\overline{G}}, \hat{\overline{B}}) = (R_{low}, G_{low}, B_{low})$:

$$\varepsilon^2 = (R_{low} + \alpha - \overline{R})^2 + (G_{low} + \alpha - \overline{G})^2 + (B_{low} + \alpha - \overline{B})^2 . \qquad (4)$$

Esto se puede simplificar en:

$$\varepsilon^2 = (\delta_R + \alpha)^2 + (\delta_G + \alpha)^2 + (\delta_B + \alpha)^2 , \qquad (5)$$

donde $\delta_R = R_{low} - \overline{R}$, $\delta_G = G_{low} - \overline{G}$ and $\delta_B = B_{low} - \overline{B}$. Además, supóngase que α (el modificador de intensidad) se puede elegir libremente, es decir, es igual a la $\alpha = -\frac{\delta_R + \delta_G + \delta_B}{3}$ óptima. Al insertar esta α óptima en la ecuación (5) se obtiene después de la simplificación:

$$\varepsilon^2 = \frac{2}{3}(\delta_R^2 + \delta_G^2 + \delta_B^2 - \delta_R\delta_G - \delta_R\delta_B - \delta_G\delta_B) = \frac{2}{3}\xi , \qquad (6)$$

donde ξ es la expresión entre paréntesis.

Sin embargo, si se elige el valor más alto para el componente rojo, es decir, $(\hat{\bar{R}}, \hat{\bar{G}}, \hat{\bar{B}})=(R_{high}, G_{low}, B_{low})$, y el hecho de que se use $R_{high} - \bar{R} = 17 + \delta_R$ usado, la ecuación (5) se puede reescribir como:

$$\varepsilon^2 = ((\delta_R + 17) + \alpha)^2 + (\delta_G + \alpha)^2 + (\delta_B + \alpha)^2. \qquad (7)$$

Esta expresión puede simplificarse aún más, insertando la $\alpha = -\frac{\delta_R + 17 + \delta_G + \delta_B}{3}$ óptima para este caso, en:

$$\begin{aligned} \varepsilon^2 &= \frac{2}{3}(\delta_R^2 + \delta_G^2 + \delta_B^2 - \delta_R\delta_G - \delta_R\delta_B - \delta_G\delta_B + 17^2 + 17 \times 2\delta_R - \delta_G - \delta_B) \\ &= \frac{2}{3}(\xi + 17[17 + 2\delta_R - \delta_G - \delta_B]). \end{aligned} \qquad (8)$$

Para determinar cuál de estos dos colores cuantificados (códigos de color) (R_{low}, G_{low}, B_{low}) o (R_{high}, G_{low}, B_{low}) es el mejor, es decir, da el valor de error más pequeño, la expresión adicional entre corchetes de la ecuación (8) se investiga. En otras palabras, si se elige $17 + 2\delta_R - \delta_G - \delta_B < 0$, ($R_{high}$, G_{low}, B_{low}), de lo contrario se elige (R_{low}, G_{low}, B_{low}) (en el caso $17 + 2\delta_R - \delta_G - \delta_B = 0$, cualquiera los códigos podría ser elegido). Este procedimiento se repite luego para todas las combinaciones posibles de cuantificaciones bajas y altas para los tres componentes de color, es decir, para todos los colores cuantificados vecinos del color promedio. El resultado se presenta en la tabla 2 a continuación.

Tabla 2

Código de color	Valor de error ε^2
R_{low}, G_{low}, B_{low}	$\frac{2}{3}\xi$
R_{high}, G_{low}, B_{low}	$\frac{2}{3}(\xi + 17[17 + 2\delta_R - \delta_G - \delta_B])$
R_{low}, G_{high}, B_{low}	$\frac{2}{3}(\xi + 17[17 + 2\delta_G - \delta_R - \delta_B])$
R_{low}, G_{low}, B_{high}	$\frac{2}{3}(\xi + 17[17 + 2\delta_B - \delta_R - \delta_G])$
R_{low}, G_{high}, B_{high}	$\frac{2}{3}(\xi + 17[17 - 2\delta_R + \delta_G + \delta_B])$
R_{high}, G_{low}, B_{high}	$\frac{2}{3}(\xi + 17[17 - 2\delta_G + \delta_R + \delta_B])$
R_{high}, G_{high}, B_{low}	$\frac{2}{3}(\xi + 17[17 - 2\delta_B + \delta_R + \delta_G])$
R_{high}, G_{high}, B_{high}	$\frac{2}{3}\xi$

Téngase en cuenta que no se requiere que ξ se calcule explícitamente, solo las expresiones (representaciones de error) en los corchetes de la tabla 2 deben calcularse para seleccionar los niveles de cuantificación (código de color) a usar. Además, téngase en cuenta que el código de color (R_{low}, G_{low}, B_{low}) y (R_{high}, G_{high}, B_{high}) dan el mismo valor de error. Esto es bajo el supuesto de que se puede alcanzar cualquier α (valor de modificador de intensidad). Sin embargo, en realidad, α se limita a los valores de modificador de intensidad de los conjuntos de modificador usados, por ejemplo, los valores de modificador de la tabla 1. De acuerdo con la tabla 1, los valores de modificador más pequeños (α) se pueden especificar con mayor precisión que los valores más grandes, lo que significa que es mejor

elegir (R_{high}, G_{high}, B_{high}) en lugar de (R_{low}, G_{low}, B_{low}) si $(\overline{R}, \overline{G}, \overline{B})$ está más cerca de (R_{high}, G_{high}, B_{high}) que de (R_{low}, G_{low}, B_{low}) y viceversa. La cuantificación combinada de esta realización de la invención aumenta la PSNR con aproximadamente 1 dB en comparación con la codificación simple. Por lo tanto, la PSNR es solo unos 0,5 dB más bajo que el resultado del esquema de codificación exhaustivo (óptimo). El tiempo total de codificación no se modificó de forma mensurable en comparación con la codificación simple, es decir, una imagen de 128 × 128 píxeles aún se comprime en aproximadamente 60 ms.

También es posible crear una tabla correspondiente a la tabla 2 usando la medida de error ponderada perceptualmente de la ecuación (3). La α óptima es entonces $\alpha = -\frac{w_R^2\delta_R + w_G^2\delta_G + w_B^2\delta_B}{w_R^2 + w_G^2 + w_B^2}$ para $(\hat{\overline{R}}, \hat{\overline{G}}, \hat{\overline{B}})=(R_{low}, G_{low}, B_{low})$. Si los valores de error para R_{low}, G_{low}, B_{low} y R_{high}, G_{high}, B_{high} son "normalizados" a 0 (básicamente corresponde a restar $\frac{2}{3}\xi$ del valor de error ε^2 para todas las diferentes variantes de R, G y B en la tabla 2) se obtiene la siguiente tabla de valores de error normalizados. Téngase en cuenta que los valores de error normalizados para los códigos de color que no sean R_{low}, G_{low}, B_{low} y R_{high}, G_{high}, B_{high} de la tabla 3 corresponden a las expresiones entre corchetes de la tabla 2.

Tabla 3

Código de color	Valor de error normalizado ε^2
R_{low}, G_{low}, B_{low}	0
R_{high}, G_{low}, B_{low}	$w_R^2[17(w_G^2 + w_B^2) - 2w_G^2(\delta_G - \delta_R) - 2w_B^2(\delta_B - \delta_R)]$
R_{low}, G_{high}, B_{low}	$w_G^2[17(w_R^2 + w_B^2) - 2w_R^2(\delta_R - \delta_G) - 2w_B^2(\delta_B - \delta_G)]$
R_{low}, G_{low}, B_{high}	$w_B^2[17(w_R^2 + w_G^2) - 2w_R^2(\delta_R - \delta_B) - 2w_G^2(\delta_G - \delta_B)]$
R_{low}, G_{high}, B_{high}	$w_R^2[17(w_G^2 + w_B^2) + 2w_G^2(\delta_G - \delta_R) + 2w_B^2(\delta_B - \delta_R)]$
R_{high}, G_{low}, B_{high}	$w_G^2[17(w_R^2 + w_B^2) + 2w_R^2(\delta_R - \delta_G) + 2w_B^2(\delta_B - \delta_G)]$
R_{high}, G_{high}, B_{low}	$w_B^2[17(w_R^2 + w_G^2) + 2w_R^2(\delta_R - \delta_B) + 2w_G^2(\delta_G - \delta_B)]$
R_{high}, G_{high}, B_{high}	0

Qué colores cuantificados vecinos ($R_XG_YB_Z$, donde X, Y, Z representan independientemente bajo o alto) para usar como código de color se puede determinar comparando los valores de error "normalizados" o las representaciones y seleccionando como código de color el color cuantificado vecino que resulta en un valor de error más pequeño.

Otra forma de elegir el código de color entre los colores cuantificados vecinos es comprimir el bloque de imágenes con cada uno de esos colores cuantificados y seleccionar el asociado con el error más pequeño. Sin embargo, tal enfoque es algo más lento que simplemente usando las representaciones de error de la tabla 2 o 3 para determinar un código de color adecuado, ya que el bloque debe comprimirse ocho veces en comparación con una vez.

La figura 7 ilustra un diagrama de flujo de un método para decodificar una imagen codificada o una versión codificada de una imagen original de acuerdo con la presente invención. La imagen codificada comprende básicamente varias representaciones codificadas de bloques de imágenes, como las representaciones 700 de la figura 4. Estas representaciones de bloques codificadas se generan preferiblemente mediante el método de codificación de imágenes explicado anteriormente en relación con la figura 1.

El método en general comienza identificando bloques de imágenes codificadas para decodificar. Podría ser posible que todos los bloques de imágenes codificadas de una imagen codificada se decodifiquen para generar una representación decodificada de la imagen original. Alternativamente, solo se puede acceder a una porción de la imagen original. Como consecuencia, solo debe decodificarse un número seleccionado de bloques de imágenes (o, más precisamente, debe decodificarse una cantidad seleccionada de elementos de imagen de ciertos bloques de imágenes).

Una vez que se identifica el bloque o bloques de imágenes codificadas correctos o su representación o representaciones, el paso S30 proporciona un conjunto de modificador de intensidad. Este conjunto de modificador se proporciona basándose en el código de intensidad en la representación codificada. Esta disposición de conjunto se realiza preferiblemente identificando, por medio del código de intensidad, un conjunto de modificador de

intensidad de una tabla, por ejemplo, la tabla 1 anterior, que comprende múltiples conjuntos de modificador. Sin embargo, en algunas aplicaciones podría ser posible que el propio código de intensidad comprenda el conjunto de modificador y que no se requiera una búsqueda en la tabla.

En el siguiente paso S31, se genera una representación de color para al menos uno de los elementos de imagen del bloque de imágenes (es decir, para el elemento o elementos de imagen que debe decodificarse). Esta generación de color se realiza basándose en el código de color en la representación de bloques codificada. En el paso S32, se selecciona el modificador de intensidad a usar para el elemento de imagen que debe decodificarse. El valor de modificador se selecciona del conjunto de modificador proporcionado en el paso S30 basándose en la representación de intensidad asociada con el elemento de imagen y se encuentra en la secuencia de representación de la representación de bloques codificada. Una vez que se selecciona el valor correcto de modificador de intensidad en el paso S32, la intensidad del elemento de imagen se modifica o modula con este valor en el paso S33. La modificación de intensidad de acuerdo con la invención se refiere a la modificación, por ejemplo, añadiendo o multiplicando, todos los componentes de color de la representación de color por el valor de modificador de intensidad (posiblemente ponderado).

Los pasos S32 y S33 podrían realizarse para varios elementos de imagen en el bloque de imágenes (ilustrado esquemáticamente por la línea 3). La invención anticipa que en algunas aplicaciones, solo se decodifica un elemento de imagen de un bloque de imágenes específico, se decodifican varios elementos de imagen de un bloque de imágenes específico y/o se decodifican todos los elementos de imagen de un bloque específico.

Los pasos S30 a S33 se repiten preferiblemente para todos los bloques de imágenes que comprenden elementos de imagen que deben decodificarse (ilustrados esquemáticamente por la línea 4). Esto significa que el bucle de los pasos S30 a S33 se podría realizar una vez, pero con mayor frecuencia varias veces para diferentes bloques de imágenes codificadas y/o varias veces para un bloque de imágenes codificadas específicas.

En el paso opcional S34, se genera una representación decodificada de la imagen original, o una porción de la misma, basándose en los elementos y bloques de imágenes decodificadas. Téngase en cuenta que en algunas aplicaciones, se deben decodificar varios elementos de imagen para representar un solo píxel de la representación decodificada. Por ejemplo, durante la interpolación trilineal, se decodifican ocho elementos de imagen vecinos y para la interpolación bilineal, el número correspondiente es de cuatro elementos de imagen, que es bien conocido por los expertos en la técnica. El método entonces termina.

La figura 8 ilustra una realización de los pasos S30 y S31 de la figura 7 con más detalle. En el paso S40, se identifica un conjunto de modificador de intensidad correcto y se selecciona de la tabla de intensidad por medio del código de intensidad. Si el conjunto de modificador de intensidad almacenado en la tabla de intensidad comprende un primer subconjunto de valores de modificador, por ejemplo, [a, b], un segundo subconjunto de valores de modificador de intensidad se puede determinar a partir de los valores del primer subconjunto, por ejemplo, [-a, -b]. En un siguiente paso S41, el color cuantificado del código de color, preferiblemente de 12 bits, se expande o se extiende, preferiblemente, a 24 bits. Para el caso con un color RGB, cada componente de color cuantificado de 4 bits del código de color se expande a un componente de color de 8 bits. Esta expansión de color puede realizarse multiplicando los componentes de color cuantificados de 4 bits por 17 para una implementación con 256 colores diferentes $(\frac{256-1}{16-1}=17).$ Esto es lo mismo que replicar el patrón de 4 bits en los primeros 4 bits (superior) y últimos (inferior) del código de color expandido de 8 bits. En otras palabras, un componente de color de 4 bits de 1010_{bin} se expandió a 1010 1010_{bin}. Si el código de color comprende 15 bits (5 bits para cada componente R, G, B), la expansión daría como resultado un color de 30 bits. Esto se puede realizar multiplicando el componente de color (5 bits) por $33\ (\frac{1024-1}{32-1}=33),$ o replicando el patrón de 5 bits a los 5 bits superiores e inferiores del código de color expandido de 10 bits. En el paso S42, el color expandido se asigna entonces a los elementos de imagen del bloque de imágenes, que se van a decodificar. El método continúa luego en el paso S32 de la figura 7.

La figura 9 ilustra una realización del paso S33 en la figura 7 con más detalle. En el paso S50, el valor de modificador de intensidad identificado y seleccionado se añade al color expandido. Por lo tanto, este valor de modificador se añade a todos los componentes (los tres para un color RGB) del color para el elemento de imagen. Esto podría implementarse como una simple adición del valor de modificador a todos los componentes de color. Sin embargo, en algunas aplicaciones podría preferirse ponderar el valor de modificador antes de añadirlo a los componentes. En tal caso, se pueden emplear diferentes ponderaciones para los diferentes componentes de color. En una realización alternativa, podría emplearse otro tipo de modificación que una simple adición, por ejemplo, multiplicación, XOR u otra modificación. En tal caso, se realiza la misma modulación a todos los componentes del color expandido usando uno y el mismo valor de modificador de intensidad, aunque este valor puede tener una ponderación diferente para los componentes. En el siguiente paso S51, los valores de los componentes de color modificados por intensidad resultantes se fijan entre un umbral de color mínimo y un umbral de color máximo. Por ejemplo, si después de añadir el valor de modificador de intensidad (posiblemente ponderado) a un componente de

color, el valor resultante es más pequeño que el umbral mínimo, el valor se fija al valor de este umbral. En consecuencia, si el valor resultante es mayor que el umbral máximo, el valor del umbral debe usarse para ese componente. Un ejemplo no limitativo de un umbral mínimo y máximo es 0 y 255, respectivamente, para el caso con 256 colores diferentes. Los umbrales correspondientes para 1024 colores podrían ser 0 y 1023. El método continúa luego en el paso S34 de la figura 7.

La decodificación de un bloque de imágenes codificada se ilustrará más adelante con un ejemplo a continuación. En este ejemplo, se supone una representación de bloques codificada como se ilustra en la figura 4 y un bloque de imágenes como se ilustra en la figura 2.

La representación codificada del bloque de imágenes es de acuerdo con 1010 0101 0001 0111 11 01 10 00 10 01 00 00_{bin} ($a517D890_{hex}$), donde el bit 0-3 es el componente rojo del código de color, el bit 4-7 es el componente verde del código de color, el bit 8-11 es el componente azul del código de color, el bit 12-15 es el código de intensidad y el bit 16-31 es la secuencia de representaciones de intensidad para los elementos de imagen del bloque.

El código de color se decodifica (expande) para generar la representación de color del bloque de imágenes. Cada componente de color en el código de color está en 4 bits, pero se expande a 8 bits multiplicando por 17 ($17 = 11_{hex}$), que es lo mismo que replicar el patrón de 4 bits en los 4 bits superior e inferior del código de 8 bits:

$$\text{Rojo: } a_{hex} \times 11_{hex} = aa_{hex} \Leftrightarrow 1010\ 1010_{bin} \Leftrightarrow 170$$

$$\text{Verde: } 5_{hex} \times 11_{hex} = 55_{hex} \Leftrightarrow 0101\ 0101_{bin} \Leftrightarrow 85$$

$$\text{Azul: } 1_{hex} \times 11_{hex} = 11_{hex} \Leftrightarrow 0001\ 0001_{bin} \Leftrightarrow 17$$

Este color expandido luego se asigna a los elementos de imagen de los bloques de imágenes dando:

(170, 85, 17)	(170, 85, 17)	(170, 85, 17)	(170, 85, 17)
(170, 85, 17)	(170, 85, 17)	(170, 85, 17)	(170, 85, 17)

El conjunto de modificador de intensidad correcto a usar se selecciona de la tabla 1 basándose en el código de intensidad. Como se ve en la tabla 1, un código de intensidad de 0111_{bin} corresponde a modificadores de intensidad [-127, -42, 42, 127].

La secuencia de representaciones de intensidad permite identificar cuál de estos cuatro valores de modificador se debe usar para el elemento de imagen diferente de acuerdo con:

$$\begin{bmatrix} 11_{bin} \\ 10_{bin} \\ 00_{bin} \\ 01_{bin} \end{bmatrix} = \begin{bmatrix} -127 \\ -42 \\ 42 \\ 127 \end{bmatrix}$$

La primera representación de intensidad es 11_{bin}, lo que significa que el primer valor de modificador de intensidad, -127, debe añadirse a los tres componentes del primer elemento de imagen:

$$\begin{bmatrix} 170 \\ 85 \\ 17 \end{bmatrix} + \begin{bmatrix} -127 \\ -127 \\ -127 \end{bmatrix} = \begin{bmatrix} 43 \\ -42 \\ -110 \end{bmatrix}$$

Los componentes resultantes se fijan entre 0 y 255, dando así (43, 0, 0). El bloque de imágenes parcialmente decodificado ahora está de acuerdo con:

(43, 0, 0)	(170, 85, 17)	(170, 85, 17)	(170, 85, 17)
(170, 85, 17)	(170, 85, 17)	(170, 85, 17)	(170, 85, 17)

Para el siguiente elemento de imagen, la representación de intensidad es 01_{bin}, es decir, el modificador de intensidad 127 debe añadirse a los tres componentes de color. El resultado después de la sujeción es (255, 212, 144). Repetir este procedimiento para todos los elementos de imagen en el bloque crearía el bloque de imágenes decodificado final que se muestra a continuación:

(43, 0, 0)	(255, 212, 144)	(128, 43, 0)	(212, 127, 59)
(128, 43, 0)	(255, 212, 144)	(212, 127, 59)	(212, 127, 59)

El esquema de codificación de imágenes (codificación de bloques de imágenes) y decodificación de imágenes (decodificación o procesamiento de bloques de imágenes) de acuerdo con la presente invención podría proporcionarse en un sistema general de procesamiento de datos, por ejemplo, en un terminal de usuario u otra unidad configurada para procesar y/o representar imágenes. Tal terminal podría ser un ordenador. Sin embargo, la invención está bien adaptada para clientes livianos, como asistencia digital personal (PDA), unidades móviles y teléfonos. Tales terminales se caracterizan típicamente por una capacidad de memoria y un ancho de banda de memoria limitados, y son alimentados por baterías, es decir, también por una fuente de alimentación limitada. Dado que tanto la codificación como la decodificación de acuerdo con la presente invención pueden implementarse de manera muy simple en hardware, software o una combinación de hardware y software, y un bloque de imágenes codificadas preferiblemente solo tiene un tamaño máximo de 32 bits, la invención podría aplicarse con ventaja a un cliente liviano.

La figura 10 ilustra un terminal 100 de usuario representado por una unidad móvil. Sin embargo, la invención no está limitada a unidades móviles, ya que podría implementarse en otros terminales y unidades de procesamiento de datos. En la figura solo se ilustran los medios y elementos de la unidad móvil 100 directamente involucrados en la presente invención.

La unidad móvil 100 comprende una unidad 200 de procesamiento (central) (CPU) para procesar datos, incluidos datos de imágenes, dentro de la unidad móvil 100. Se proporciona un sistema gráfico 130 en la unidad móvil 100 para gestionar datos de imágenes y gráficos. En particular, el sistema gráfico 130 está adaptado para procesar o mostrar 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 ella. En esta memoria 140 pueden almacenarse datos de imágenes, en particular datos de imágenes codificadas (bloques de imágenes codificadas) de acuerdo con la presente invención. Debido al pequeño tamaño total de los bloques de imágenes (32 bits) y la alta tasa de compresión (4 bpp), los datos de imágenes se pueden almacenar de manera eficiente en la memoria 140 también en los casos con una unidad móvil 100 con capacidad de memoria limitada.

Un codificador 210 de imágenes de acuerdo con la presente invención se proporciona en la unidad móvil 100. Este codificador 210 está configurado para codificar una imagen o textura en una representación codificada de la imagen (o textura). Como se explicó anteriormente, tal representación codificada comprende una secuencia o archivo de múltiples bloques de imágenes decodificadas. Este codificador 210 de imágenes se puede proporcionar como software que se ejecuta en la CPU 200, como se ilustra en la figura. Alternativamente, o además, el codificador 210 podría estar dispuesto en el sistema gráfico 130 o en cualquier otro lugar en la unidad móvil 100.

Se puede proporcionar una representación codificada de una imagen desde el codificador 210 de bloques a la memoria 140 a través de un bus (memoria) 150, para almacenarla en ella hasta una representación posterior de la imagen. Alternativamente, o además, los datos de la imagen codificada pueden reenviarse a una unidad 110 de entrada y salida (I/O) para la transmisión (inalámbrica o por cable) a otros terminales o unidades externas. Esta unidad 110 de I/O también puede adaptarse para recibir datos de imágenes desde una unidad externa. Estos datos de imágenes podrían ser una imagen que debería estar codificada por el codificador 210 de imágenes o los datos de imágenes codificadas que deberían decodificarse. También podría ser posible almacenar la representación de la imagen codificada en una memoria de textura dedicada provista, por ejemplo, en el sistema gráfico 130. Además, porciones de la imagen codificada podrían también, o alternativamente, almacenarse (temporalmente) en una memoria caché de texturas, por ejemplo, en el sistema gráfico 130.

Si el bus (memoria) 150 tiene un ancho de banda máximo de 32 bits, se requiere un solo acceso a la memoria para obtener o leer una representación de imágenes codificadas de la invención desde la memoria 140. Sin embargo, si el bus 150 tiene mayor capacidad de ancho de banda, por ejemplo, 64 bits o incluso 128 bits, se pueden obtener múltiples representaciones de imágenes codificadas en un solo acceso de memoria. Por ejemplo, supongamos un bus 150 de 64 bits y un tamaño de bloque de imágenes de acuerdo con la figura 2. Si los bloques de imágenes se apilan "uno encima del otro", un bloque de imágenes junto con el bloque de imágenes posterior en la memoria 140 formará un cuadrado de 4x4 de elementos de imagen. Sin embargo, si los bloques están posicionados "uno al lado del otro", el bloque de imágenes junto con el siguiente bloque formará un cuadro de 2x8. Se prefiere un cuadrado 4x4, ya que la probabilidad de encontrar un elemento de imagen deseado en el cuadrado 4x4 es mayor que para el cuadro 2x8, si se emplea alguna forma de sistema de almacenamiento en caché de textura, que es bien conocido por los expertos en la técnica.

Se proporciona un decodificador 220 de imágenes de acuerdo con la presente invención en la unidad móvil 100 para decodificar una imagen codificada con el fin de generar una representación de imagen decodificada. Esta representación decodificada podría corresponder a la imagen original completa o una porción de la misma. El decodificador 220 de imágenes proporciona datos de imágenes decodificadas al sistema gráfico 130, que a su vez típicamente procesa los datos antes de que se representen o presenten en la pantalla 120. El decodificador 220 de

imágenes puede disponerse en el sistema gráfico 130, como se ilustra en la figura. Alternativamente, o además, el decodificador 200 se puede proporcionar como software que se ejecuta en la CPU 200 o en cualquier otro lugar de la unidad móvil 100.

La unidad móvil 100 podría estar equipada tanto con un codificador 210 de imágenes como con un decodificador 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, los datos de la imagen codificada podrían transmitirse a otro terminal que realiza la decodificación y, posiblemente, la representación de la imagen. En consecuencia, un terminal 100 solo podría incluir un decodificador 220 de imágenes, es decir, sin codificador. Tal terminal 100 recibe una señal que comprende datos de imágenes codificadas de otro terminal y la decodifica para generar una representación de imagen decodificada. Por lo tanto, la señal de la imagen codificada podría transmitirse de forma inalámbrica entre los terminales usando un transmisor y receptor de radio. Alternativamente, podrían emplearse otras técnicas para distribuir imágenes y representaciones de imágenes codificadas entre terminales de acuerdo con la invención, tales como técnicas de IR que usan puertos IR y transferencia por cable de datos de imágenes entre terminales. También se podrían usar tarjetas de memoria o chips que se pueden conectar e intercambiar entre terminales para esta distribución entre terminales de datos de imágenes.

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

La figura 11 ilustra un diagrama de bloques de una realización de un codificador 210 de imágenes de acuerdo con la presente invención. El codificador 210 comprende típicamente un descomponedor 215 de imágenes para descomponer o dividir una imagen de entrada en varios bloques de imágenes. El descomponedor 215 está configurado preferiblemente para descomponer la imagen en bloques de imágenes que comprenden ocho elementos de imagen (píxeles o téxeles), es decir, que tienen un tamaño general de elementos de imagen 8x1 u 8x1, más preferiblemente elementos de imagen 4x2 o 2x4. Este descomponedor 215 podría adaptarse para descomponer diferentes imágenes de entrada en bloques de imágenes con diferentes tamaños. Por ejemplo, para un primer tipo de imagen se usa un tamaño de bloque de imágenes de elementos de imagen 4x2, mientras que para un segundo tipo se puede usar un tamaño de bloque de 8x1. En tal caso, el descomponedor 215 recibe preferiblemente información de entrada, lo que permite la identificación del formato de bloque de imágenes que se usará para una imagen dada.

Esta realización del codificador 210 de imágenes comprende un codificador 300 de bloques único. Este codificador 300 de bloques codifica el bloque o bloques de imágenes recibidos del descomponedor de imágenes para generar una representación o representaciones de bloques codificadas. Tal representación de bloques de imágenes comprende un código de color, un código de intensidad y una secuencia de representaciones de intensidad. El tamaño total de la representación de bloques es mucho más pequeño que el tamaño correspondiente del bloque de imágenes sin codificar. El codificador 300 de bloques está configurado preferiblemente para procesar (codificar) cada bloque de imágenes desde el descomponedor 215 de forma secuencial.

El codificador 300 de bloques preferiblemente comprende, o tiene acceso a, una tabla 500 de intensidad que comprende múltiples conjuntos de modificador de intensidad. Los conjuntos de modificador de la tabla 500 se usan durante la codificación para la generación de la intensidad, y posiblemente el código de color. La tabla 500 de intensidad podría estar dispuesta en el codificador 300 de bloques o en cualquier otra parte del codificador 210 de imágenes.

El codificador 210 de imágenes podría comprender una tabla 500 de intensidad única. Alternativamente, se podrían disponer varias tablas diferentes en el codificador 210, donde los modificadores de intensidad de las tablas se adaptan para diferentes tipos de imágenes o una tabla se podría adaptar para una imagen específica. Por ejemplo, una primera tabla de intensidad podría usarse durante la codificación de un primer tipo de imagen, por ejemplo, foto, mientras que una segunda tabla se usa para codificar una imagen de un segundo tipo, por ejemplo, texto. Sin embargo, para ahorrar memoria, una tabla 500 de intensidad única generada con datos de entrenamiento de varios tipos de imágenes diferentes se emplea preferiblemente en el codificador 210.

Las unidades 215 y 300 del codificador 210 de imágenes pueden proporcionarse como software, hardware o una combinación de los mismos. Las unidades 215, 300 y 500 pueden implementarse juntas en el codificador 210 de imágenes. Alternativamente, una implementación distribuida también es posible con algunas de las unidades proporcionadas en otro lugar de la unidad móvil.

La figura 12 ilustra un diagrama de bloques de otra realización del codificador 210 de imágenes de acuerdo con la presente invención. Este codificador 210 de imágenes comprende un descomponedor 215 de imágenes como la realización de la figura 11, que no se explica más. Sin embargo, el codificador 210 incluye múltiples codificadores 300-1 a 300-M de bloques (M, donde M es un número entero positivo mayor que uno). Cada uno de estos codificadores 300-1 a 300-M de bloques corresponde básicamente al codificador de bloques del codificador de imágenes de la figura 11. Al proporcionar múltiples codificadores 300-1 a 300-M de bloques en el codificador 210 de

imágenes, se pueden procesar (codificar) múltiples bloques de imágenes del descomponedor 215 en paralelo, lo que reduce el tiempo total de codificación de imágenes.

Cada codificador 300-1 a 300-M de bloques podría comprender una tabla 500 de intensidad. Las tablas 500 de intensidad en los diferentes codificadores 300-1 a 300-M podrían incluir valores de modificador de intensidad idénticos. Alternativamente, diferentes codificadores de bloques podrían incluir diferentes tablas. En tal caso, uno o varios codificadores de bloques podrían adaptarse para un determinado tipo de imagen, mientras que otros codificadores de bloques están adaptados para otro tipo o tipos de imagen. En una implementación alternativa, una tabla 500 de intensidad única está dispuesta en el codificador 210 de imágenes y conectada a todos los codificadores 300-1 a 300-M de bloques.

Las unidades 215 y 300-1 a 300-M del codificador 210 de imágenes pueden proporcionarse como software, hardware o una combinación de los mismos. Las unidades 215, 300-1 a 300-M y 500 pueden implementarse juntas en el codificador 210 de imágenes. Alternativamente, una implementación distribuida también es posible con algunas de las unidades proporcionadas en otro lugar de la unidad móvil.

La figura 13 ilustra un diagrama de bloques de una realización de un codificador 300 de bloques de acuerdo con la presente invención, tal como el codificador de bloques del codificador de imágenes en la figura 11 o uno de los codificadores de bloques en el codificador de imágenes de la figura 12. El codificador 300 comprende un cuantificador 310 de color que determina una representación de color de los colores de los elementos de imagen en el bloque de imágenes y cuantifica esta representación de color. La representación de color es preferiblemente un color promedio de 24 bits del elemento de imagen y posteriormente es cuantificada en una representación de color de 12 bits, es decir, el código de color, por el cuantificador 310.

Se proporciona un cuantificador 320 de intensidad en el codificador 300 de bloques para identificar un conjunto de modificador de intensidad para usar para un bloque de imágenes actual. El cuantificador 320 está configurado preferiblemente para seleccionar este conjunto de modificador de una tabla 500 de intensidad asociada. El cuantificador 320 genera entonces un código de intensidad asociada con el conjunto de modificador seleccionado. El codificador 300 incluye además un selector 330 de intensidad que selecciona, para los elementos de imagen en el bloque de imágenes, un modificador de intensidad del conjunto de modificador de intensidad identificado. El cuantificador 310 de color, el cuantificador 320 de intensidad y el selector 330 de intensidad están configurados preferiblemente para la cuantificación combinada del código de color e intensidad, como se explicó en detalle anteriormente.

Las unidades 310, 320 y 330 del codificador 300 de bloques pueden proporcionarse como software, hardware o una combinación de los mismos. Las unidades 310, 320, 330 y 500 pueden implementarse juntas en el codificador 300 de bloques. Alternativamente, una implementación distribuida también es posible con algunas de las unidades proporcionadas en otro lugar en el codificador de imágenes.

La figura 14 ilustra un diagrama de bloques de otra realización de un codificador 300 de bloques de acuerdo con la presente invención. Este codificador 300 de bloques comprende un estimador 340 de error para estimar los valores de error con el fin de seleccionar el conjunto y los valores de modificador de intensidad, y posiblemente el valor de color cuantificado, para usar para un bloque de imágenes. Este estimador 340 está configurado preferiblemente para calcular un valor de error para una primera selección de un conjunto y los valores de modificador de intensidad (y color en el caso de codificación exhaustiva) para los elementos de imagen en el bloque de imágenes. Este primer valor de error se almacena. El cálculo de error se repite para todas las selecciones posibles de conjunto y los valores de modificador (y color), y después de cada cálculo, el valor de error estimado se compara con el error almacenado. Si es más pequeño que el valor almacenado, reemplaza el error almacenado anteriormente. Además, también se almacena la selección del conjunto y los valores de modificador (y el color) asociados con el valor de error. Una vez que se han probado todas las combinaciones, la selección que resulta en el error más pequeño se usa para la generación de códigos de intensidad (y color) y representaciones de intensidad. Un selector 322 de conjunto de modificador de intensidad y un selector 330 de intensidad seleccionan el conjunto y los valores de modificador de modificador asociados con el error más pequeño. Las unidades restantes del codificador 300 de bloques tienen correspondencias en la figura 13 y no se explican más a fondo. Alternativamente, esta realización del codificador 300 de bloques podría funcionar de acuerdo con la cuantificación combinada como se describe anteriormente. En tal caso, el cuantificador 310 de color proporciona los colores cuantificados vecinos del color promedio calculado del bloque de imágenes. El estimador 340 de error determina las representaciones de error para cada uno de dichos colores cuantificados vecinos, por ejemplo, como se ilustra en la tabla 2 o 3, y el color cuantificado vecino que resulta en una representación de error más pequeña se selecciona como el código de color para el bloque de imágenes.

Las unidades 310, 320, 322, 330 y 340 del codificador 300 de bloques pueden proporcionarse como software, hardware o una combinación de los mismos. Las unidades 310, 320, 322, 330, 340 y 500 pueden implementarse juntas en el codificador 300 de bloques. Alternativamente, una implementación distribuida también es posible con algunas de las unidades proporcionadas en otro lugar en el codificador de imágenes.

Una implementación preferida de un cuantificador 310 de color de acuerdo con la presente invención se ilustra en el diagrama de bloques de la figura 15. El cuantificador 310 comprende medios 312 configurados para determinar un promedio de los colores de los elementos de imagen en el bloque de imágenes. Este color promedio es preferiblemente un color RGB, pero podría ser cualquier otro formato de color usado en el procesamiento de imágenes. Este color promedio determinado se proporciona luego a los 314 medios de cuantificación, que cuantifican el color promedio. El cuantificador 314 está configurado preferiblemente para cuantificar un color RGB promedio de 24 bits del promediador 312 de color en un color RGB de 12 bits.

Las unidades 312 y 314 del cuantificador 310 de color pueden proporcionarse como software, hardware o una combinación de los mismos. Las unidades 310 y 314 pueden implementarse juntas en el cuantificador 310 de color. Alternativamente, una implementación distribuida también es posible con algunas de las unidades proporcionadas en otro lugar del codificador de bloques.

La figura 16 ilustra un diagrama de bloques de una realización de un decodificador 220 de bloques de acuerdo con la presente invención. El decodificador 220 de bloques comprende preferiblemente un selector 222 de bloques que se adapta para seleccionar, por ejemplo, desde una memoria, que codifica el bloque o bloques de imágenes que deben proporcionarse a un decodificador 400 de bloques para decodificar. El selector 222 de bloques recibe preferiblemente información de entrada asociada con los datos de la imagen codificada, por ejemplo, desde un encabezado o un motor de renderización. Una dirección de un bloque de imágenes codificadas que tiene el elemento o elementos de imagen deseados se calcula basándose en la información de entrada. Esta dirección calculada depende preferiblemente de las coordenadas del elemento de imagen (píxel o téxel) dentro de una imagen. Usando la dirección, el selector 222 de bloques identifica el bloque de imágenes codificadas de la memoria. Este bloque de imágenes codificadas identificado se recupera del almacenamiento y se proporciona al decodificador 400 de bloques.

El acceso (aleatorio) a los elementos de imagen de un bloque de imágenes permite ventajosamente la decodificación selectiva de solo aquellas porciones de una imagen que son necesarias. Además, la imagen se puede decodificar en cualquier orden en que se requieran los datos. Por ejemplo, en el mapeo de texturas solo pueden requerirse porciones de la textura y estas porciones en general se requerirán en un orden no secuencial. Por lo tanto, la decodificación de imágenes de la presente invención puede aplicarse ventajosamente para procesar solo una porción o sección de una imagen.

El bloque de imágenes codificadas seleccionado se reenvía al decodificador 400 de bloques. Además del bloque de imágenes, el decodificador 400 preferiblemente recibe información que especifica qué elementos de imagen del bloque deben decodificarse. La información podría especificar que todo el bloque de imágenes, es decir, todos los elementos de imagen en él, debería decodificarse. Sin embargo, la información recibida podría identificar solo uno o algunos de los elementos de imagen que deberían decodificarse. El decodificador 400 de bloques genera una representación decodificada del elemento o elementos de imagen en el bloque. Esta representación decodificada es preferiblemente un color de 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. El decodificador 400 de bloques comprende preferiblemente una tabla 500 de intensidad que se usa durante el procedimiento de decodificación. Alternativamente, esta tabla 500 de intensidad podría proporcionarse en otro lugar en el decodificador 220 de imágenes. El uso de diferentes tablas de intensidad para diferentes tipos de imágenes, explicado anteriormente en relación con la figura 11, también se aplica al decodificador 220 de imágenes.

Se podría proporcionar un compositor 224 de imágenes opcional en el decodificador 220. Este compositor recibe los elementos de imagen decodificados del decodificador 400 de bloques y los compone para generar un píxel que se puede renderizar o visualizar en una pantalla. El compositor 224 podría requerir varios elementos de imagen de entrada para generar un solo píxel. Este compositor 224 de imágenes podría proporcionarse alternativamente en el sistema gráfico.

Las unidades 222, 224 y 400 del decodificador 220 de bloques pueden proporcionarse como software, hardware o una combinación de los mismos. Las unidades 222, 224, 400 y 500 pueden implementarse juntas en el decodificador 220 de bloques. Alternativamente, una implementación distribuida también es posible con algunas de las unidades proporcionadas en otro lugar de la unidad móvil.

La figura 17 ilustra un diagrama de bloques de otra realización de un decodificador 220 de imágenes de acuerdo con la presente invención. El selector 222 de bloques y el compositor 224 de imágenes son similares a las unidades correspondientes en la figura 16 y no se explican más.

El decodificador 220 de imágenes comprende múltiples decodificadores 400-1 a 400-Q de bloques (Q es un entero positivo mayor que uno). Al tener acceso a múltiples decodificadores 400-1 a 400-Q de bloques, el decodificador 220 de imágenes puede procesar (decodificar) múltiples bloques de imágenes codificadas en paralelo. Estos múltiples decodificadores 400-1 a 400-Q de bloques permiten un procesamiento paralelo que aumenta el rendimiento y la eficiencia de procesamiento del decodificador 220 de imágenes. Por ejemplo, un elemento de imagen decodificado es en general suficiente para la interpolación del vecino más cercano, mientras que cuatro (ocho) elementos de

imagen son necesarios para la interpolación bilineal (trilineal). Cada decodificador 400-1 a 400-Q de bloques podría comprender una tabla 500 de intensidad usada para la decodificación. Alternativamente, una única tabla 500 está dispuesta en el decodificador 220 de imágenes y conectada a todos los decodificadores 400-1 a 400-Q de bloques. La explicación adicional sobre el uso de diferentes tipos de tablas de intensidad, véase más arriba en relación con la figura 12, también se aplica al decodificador 220 de imágenes.

Las unidades 222, 224 y 400-1 a 400-Q del decodificador 220 de imágenes pueden proporcionarse como software, hardware o una combinación de los mismos. Las unidades 222, 224, 400-1 a 400-Q y 500 pueden implementarse juntas en el decodificador 220 de imágenes. Alternativamente, una implementación distribuida también es posible con algunas de las unidades proporcionadas en otro lugar de la unidad móvil.

La figura 18 es una ilustración de una realización de un decodificador 400 de bloques de acuerdo con la presente invención. El decodificador 400 de bloques comprende medios 410 para proporcionar un conjunto de modificador de intensidad a partir de una tabla 500 de intensidad asociada basada en el código de intensidad. Este proveedor 410 podría configurarse para obtener un primer subconjunto de valores de modificador de la tabla 500 de intensidad y determinar un segundo subconjunto de modificador basado en el primer subconjunto. Un generador 420 de color genera una única representación de color para todos los elementos de imagen en el bloque de imágenes basándose en el código de color. Este generador 420 preferiblemente expande el color de 12 bits del código en un color (RGB) de 24 bits.

Un selector 430 de modificador de intensidad está dispuesto para seleccionar uno de los valores de modificador de intensidad del conjunto de modificador proporcionado por los medios 410. El selector 430 de modificador está configurado para seleccionar los valores correctos de modificador para los elementos de imagen en el bloque de imágenes codificadas basándose en la secuencia de representaciones de intensidad. El color expandido del generador 420 de color y el valor de modificador del selector 430 de modificador se reenvían a un modulador de intensidad o modificador 440 que modifica la intensidad de los componentes de color del color expandido con el valor de modificador. El modificador 440 podría usar un valor de modificador de intensidad ponderado, con diferentes ponderaciones para los diferentes componentes de color. Además, una vez que se modificó la intensidad de los componentes de color, el modificador 440 preferiblemente sujeta los componentes entre un umbral máximo y mínimo, por ejemplo, entre 0 y 255.

Las unidades 410, 420, 430 y 440 del decodificador 400 de bloques pueden proporcionarse como software, hardware o una combinación de los mismos. Las unidades 410, 420, 430, 440 y 500 pueden implementarse juntas en el decodificador 400 de bloques. Alternativamente, una implementación distribuida también es posible con algunas de las unidades proporcionadas en otro lugar en el decodificador de imágenes.

La figura 19 ilustra esquemáticamente una posible implementación de hardware de un decodificador 400 de bloques de acuerdo con la presente invención. La entrada al decodificador 400 de bloques es una representación 700 de bloques codificada que comprende un código 710 de color de 12 bits (4 bits para cada uno de los componentes rojo, verde y azul), un código 720 de intensidad de 4 bits y una secuencia de 16 bits de intensidad 730.

El código de color se proporciona al generador 420 de color, que se realiza mediante tres extensores 422 a 426 de bits. Un primer extensor 422 de bits recibe el componente rojo de 4 bits, un segundo extensor 424 y tercero 426 reciben el componente verde y azul de 4 bits, respectivamente. La salida del extensor respectivo 422 a 426 es un componente de color de 8 bits. Este componente de 8 bits se obtiene simplemente multiplicando el componente de entrada por 17, o multiplicando el componente por 16 y luego añadiendo el componente. Alternativamente, los extensores 422 a 426 podrían implementarse como desplazadores de bits y puertas OR, por ejemplo, $(1011_{bin} << 4)$ OR $1011_{bin} = 1011\ 0000_{bin}$ OR $1011_{bin} = 1011\ 1011_{bin}$, donde << 4 corresponde a desplazar un código de cuatro bits a la izquierda.

Un selector 430 de modificador se implementa como un multiplexor 435. Se introduce un índice de direcciones de 3 bits en este multiplexor 435. Basándose en el índice de direcciones, el multiplexor 435 selecciona cuál de los ocho elementos de imagen se decodifica. La representación de intensidad de 2 bits asociada con el elemento de imagen seleccionado se reenvía a una tabla 415 de consulta. Esta tabla de consulta corresponde al proveedor 410 de conjuntos de modificador y la tabla 500 de intensidad de la figura 18. Usando el código de intensidad de entrada y la representación de intensidad, la consulta 415 obtiene el valor correcto de modificador de intensidad de uno de los conjuntos de modificador en la tabla. Este valor de modificador con signo de 9 bits (positivo o negativo) se proporciona a un modificador 440 de intensidad. En esta implementación de hardware, el modificador 440 comprende tres sumadores 441 a 443 y tres fijadores 444 a 446. El valor de modificador se introduce en el sumador respectivo 441 a 443. Un primer sumador 441 añade el valor de modificador de intensidad al componente rojo de 8 bits del extensor 422 de bits. En consecuencia, el sumador 442 y el sumador 443 añaden el valor de modificador al componente verde y azul de 8 bits del extensor 424 y 426 de bits, respectivamente. En una implementación alternativa, los sumadores 441 a 443 se pueden reemplazar con otros elementos de modificación, por ejemplo, multiplicadores o puertas XOR. Las salidas de los sumadores 441 a 443 se reenvían a los fijadores 444 a 446, que fijan la intensidad de los componentes de color modificados entre 0 y 255. La salida de los fijadores 444 a 446 es el color de 24 bits descomprimido o decodificado del elemento de imagen.

La figura 20 ilustra esquemáticamente una posible implementación de hardware de los extensores 422; 424; 426 de bits de la figura 19. Estos extensores reciben un componente de color de 4 bits (rojo, verde o azul) y generan un componente de color de 8 bits correspondiente extendido. En el componente de color de 8 bits de salida, los cuatro bits más significativos (MSB) constituyen el componente de color de 4 bits de entrada, el "quinto MSB" corresponde al MSB del componente de entrada, el "sexto MSB" corresponde al "segundo MSB "del componente de entrada y los dos bits menos significativos restantes (LSB) corresponden a los dos LSB del componente de entrada.

La figura 21 ilustra esquemáticamente una posible implementación de hardware de la tabla 415 de consulta de la figura 19. Los tres LSB de los códigos de intensidad de entrada de 4 bits se introducen en dos multiplexores 411 y 412 para seleccionar un valor de modificador de intensidad de 7 bits de ocho posibles valores de modificador para cada multiplexor 411 y 412. A partir de estos 16 modificadores de intensidad, los 48 valores restantes podrían calcularse si se emplea una tabla de intensidad de acuerdo con la tabla 1. Los valores de modificador de intensidad seleccionados de los multiplexores 411 y 412 se introducen a otro multiplexor 413 que selecciona uno de estos valores basándose en datos de entrada de 1 bit (1 bit de la representación de intensidad de 2 bits) del multiplexor 435 en la figura 19. El valor de modificador seleccionado luego se reenvía tanto a un multiplexor 416 como al medio 414 de negación que niega el valor de modificador. También este valor negado se reenvía al multiplexor 416. Este multiplexor 416 selecciona el valor positivo de modificador de intensidad de 7 bits o el valor negado basándose en el bit restante de la representación de intensidad del multiplexor 435 en la figura 19. El valor de modificador seleccionado (8 bits) se lleva entonces a un multiplexor 418 y a un desplazador 417 de bits que desplaza el valor de modificador un bit a la izquierda, lo que resulta en un modificador de intensidad de 9 bits (corresponde a una multiplicación del valor, de base diez, por dos). El multiplexor 418 luego selecciona el valor de modificador de 8 bits o el valor de modificador de 9 bits basado en el MSB del código de intensidad. El resultado de la selección es el valor de modificador de intensidad de 9 bits, de los 64 posibles valores de modificador, para usar para un elemento de imagen específico.

La figura 22 ilustra esquemáticamente una posible implementación de hardware de los fijadores 444; 445; 446 de la figura 19. La entrada al fijador 444; 445; 446 es un valor de componente de color modificado en intensidad de 10 bits. Los ocho LSB de este valor de entrada se traen un multiplexor 447. La otra entrada al multiplexor es el valor de umbral máximo (255; 8 bits). El multiplexor 447 selecciona el valor de entrada de 8 bits o el valor de umbral máximo basado en el segundo MSB del componente de color modificado por intensidad. En otras palabras, si este segundo MSB es igual a uno, el multiplexor 447 genera el valor de umbral; de lo contrario (el segundo MSB es igual a cero), el valor de entrada de 8 bits se envía a un segundo multiplexor 448. Este segundo multiplexor 448 compara la salida del primer multiplexor 447 con el valor de umbral mínimo (0; 8 bits) basado en el MSB del componente de color. Si este MSB o bit de signo es igual a uno, la salida del primer multiplexor 447 es negativa y el valor mínimo del umbral debería ser seleccionado por el segundo multiplexor 448. Sin embargo, si el bit de signo es cero, la salida del primer multiplexor 447 también debería ser emitida desde el segundo multiplexor 448.

La solución de hardware para el codificador 400 de bloques en la figura 19 es muy simple, básicamente comprende solo tres adiciones, una negación y 12 multiplexores si los extensores 422; 424; 426 de bits, la tabla 415 de consulta y fijadores 444; 445; 446 se implementan de acuerdo con la figura 20, la figura 21 y la figura 22, respectivamente. Esto debe compararse con la descompresión que usa el esquema S3TC [3], que requiere hasta 42 adiciones y dos multiplexores.

Un experto en la técnica entenderá que pueden realizarse diversas modificaciones y cambios a la presente invención sin apartarse del alcance de la misma, que se define en las reivindicaciones adjuntas.

Referencias

[1] E. J. Delp y O. R. Mitchell, "Compresión de imagen usando codificación de truncamiento de bloque", IEEE Transacciones en comunicaciones, vol. COM-2, nº. 9, págs. 1335-1342, septiembre de 1979

[2] G. Campbell, T. A. DeFanti, J. Frederiksen, S. A. Joyce, L. A. Leske, J. A. Lindberg y D. J. Sandin, "Codificación a todo color de dos bits/píxeles", Procesos de la Conferencia SIGGRAPH '86, vol. 20, nº 4, págs. 215-223, agosto de 1986

[3] Patente de Estados Unidos Nº 5.956.431

[4] T. Akenine-Möller y J. Ström, "Gráficos para las masas: una arquitectura de hardware para teléfonos móviles", ACM Transacciones en gráficos, vol. 22, nº 3, Actas de ACM SIGGRAPH 2003, págs. 801-808, julio de 2003

[5] S. Fenney, "Compresión de textura usando modulación de señal de baja frecuencia", Hardware de gráficos 2003, págs. 84-91, julio de 2003

[6] Y. Linde, A. Buzo y R. Gray, "Un algoritmo para el diseño de cuantificador vectorial", IEEE Transacciones en comunicaciones, vol. 28, págs. 84-94, enero, 1980

REIVINDICACIONES

1.- Un decodificador (400) de bloques que comprende:

un generador (420) de color configurado para generar, basándose en un código (710) de color de una representación codificada (700) de un bloque (600) de imágenes que comprende múltiples píxeles (610), una representación de color que comprende un valor de componente de color rojo, un valor de componente de color verde y un valor de componente de color azul para un píxel (610) de dichos múltiples píxeles (610), caracterizado por:

un proveedor (410) de conjuntos de modificador configurado para seleccionar, de una tabla (500) de intensidad que comprende múltiples conjuntos de modificador de intensidad, un conjunto de modificador de intensidad de múltiples modificadores de intensidad basados en un código (720) de intensidad de dicha representación codificada (700);

un selector (430) de modificador configurado para seleccionar, para dicho píxel (610) y basado en un índice de intensidad de dicha representación codificada (700), un modificador de intensidad de dicho conjunto de modificador de intensidad seleccionado por dicho proveedor (410) de conjunto de modificador; y

un modificador (440) de intensidad configurado para modificar una intensidad de dicho píxel (610) añadiendo dicho modificador de intensidad a cada uno de dicho valor de componente de color rojo, dicho valor de componente de color verde y dicho valor de componente de color azul de dicha representación de color.

2.- El decodificador de bloques de acuerdo con la reivindicación 1, caracterizado porque dicho modificador (440) de intensidad está configurado para fijar las sumas de dicho valor de modificador de intensidad y dicho valor de componente de color rojo, dicho valor de componente de color verde y dicho valor de componente de color azul entre un valor de umbral mínimo y un valor de umbral máximo.

3.- El decodificador de bloques de acuerdo con la reivindicación 2, caracterizado porque dicho modificador (440) de intensidad está configurado para fijar dichas sumas de dicho valor de modificador de intensidad y dicho valor de componente de color rojo, dicho valor de componente de color verde y dicho valor de componente de color azul entre 0 y 255.

4.- El decodificador de bloques de acuerdo con cualquiera de las reivindicaciones 1 a 3, caracterizado por dicha tabla (500) de intensidad que comprende dichos múltiples conjuntos de modificador de intensidad.

5.- El decodificador de bloques de acuerdo con cualquiera de las reivindicaciones 1 a 4, caracterizado porque dicho código de color comprende tres componentes de color cuantificados de 4 bits y dicho generador (420) de color está configurado para replicar, para cada componente de color cuantificado de 4 bits, el patrón de 4 bits de dicho componente de color cuantificado de 4 bits a los primeros y últimos 4 bits de un valor de componente de color de 8 bits.

6.- Un decodificador (220) de imágenes que comprende:

un selector (222) de bloques configurado para seleccionar un bloque (600) de imágenes que comprende múltiples píxeles (610);

un decodificador (400) de bloques de acuerdo con cualquiera de las reivindicaciones 1 a 5 configurado para decodificar una representación codificada (700) de dicho bloque (600) de imágenes; y

un compositor (224) de imágenes configurado para componer múltiples píxeles coloreados y de intensidad modificada (610) a partir de dicho decodificador (400) de bloques para generar una representación decodificada de una imagen.

7.- Un terminal (100) de usuario que comprende:

una memoria (140) configurada para almacenar representaciones codificadas (700) de bloques (600) de imágenes; y

un sistema de gráficos (130) que comprende un decodificador (220) de imágenes de acuerdo con la reivindicación 6.

8.- Un método para procesar una representación comprimida (700) de un bloque (600) de imágenes que comprende múltiples píxeles (610), comprendiendo dicho método:

generar, basándose un código (710) de color de dicha representación comprimida (700), una representación de color que comprende un valor de componente de color rojo, un valor de componente de color verde y un valor de componente de color azul para al menos un píxel (610) de dichos múltiples píxeles (610), caracterizado por:

seleccionar, de una tabla (500) de intensidad que comprende múltiples conjuntos de modificador de intensidad, un conjunto de modificador de intensidad de múltiples modificadores de intensidad basados en un código (720) de intensidad de dicha representación comprimida (700);

seleccionar, para al menos dicho píxel (610) y basándose en un índice de intensidad de dicha representación comprimida (700), un modificador de intensidad de dicho conjunto de modificador de intensidad; y

modificar una intensidad de al menos dicho píxel (610) añadiendo dicho modificador de intensidad a cada uno de dicho valor de componente de color rojo, dicho valor de componente de color verde y dicho valor de componente de color azul de dicha representación de color.

9.- El método de acuerdo con la reivindicación 8, caracterizado por fijar las sumas de dicho valor de modificador de intensidad y dicho valor de componente de color rojo, dicho valor de componente de color verde y dicho valor de componente de color azul entre un valor de umbral mínimo y un valor de umbral máximo.

10.- El método de acuerdo con la reivindicación 9, caracterizado porque la fijación de dichas sumas comprende la fijación de dichas sumas de dicho valor de modificador de intensidad y dicho valor de componente de color rojo, dicho valor de componente de color verde y dicho valor de componente de color azul entre 0 y 255.

11.- El método de acuerdo con cualquiera de las reivindicaciones 8 a 10, caracterizado porque dicho código de color comprende tres componentes de color cuantificados de 4 bits y generar dicha representación de color comprende replicar, para cada componente de color cuantificado de 4 bits, el patrón de 4 bits de dicho componente de color cuantificado de 4 bits a los primeros y últimos 4 bits de un valor de componente de color de 8 bits.

12.- Un codificador (300) de bloques que comprende:

un cuantificador (310) de color configurado para determinar un código (710) de color que es una representación, que comprende un valor de componente de color rojo, un valor de componente de color verde y un valor de componente de color azul, de los colores de múltiples píxeles (610) en un bloque (600) de imágenes, caracterizado por:

un cuantificador (320) de intensidad que comprende un selector (322) configurado para seleccionar un conjunto de modificador de intensidad de una tabla (500) de intensidad que comprende múltiples conjuntos de modificador de intensidad, en el que dicho conjunto de modificador de intensidad comprende múltiples modificadores de intensidad para modificar la intensidad de dichos píxeles (610) en dicho el bloque (600) de imágenes añadiendo un modificador de intensidad a cada uno de dicho valor de componente de color rojo, dicho valor de componente de color verde y dicho valor de componente de color azul de dicha representación de color y dicho código (720) de intensidad permite la identificación de dicho conjunto de modificador de intensidad seleccionado de dicha tabla (500) de intensidad por dicho selector (322); y

un selector (330) de intensidad configurado para seleccionar, para cada píxel (610) en dicho bloque (600) de imágenes, un índice de intensidad asociado con un modificador de intensidad de dicho conjunto de modificador de intensidad.

13.- Un codificador (210) de imágenes que comprende:

un descomponedor (215) de imágenes configurado para descomponer una imagen de entrada en bloques (610) de imágenes, cada bloque (600) de imágenes comprendiendo múltiples píxeles (610); y

un codificador (300) de bloques de acuerdo con la reivindicación 12 para codificar un bloque (600) de imágenes para generar una representación codificada (700) de dicho bloque (600) de imágenes.

14.- Un terminal de usuario (100) que comprende:

una unidad (200) de procesamiento;

un codificador (210) de imágenes de acuerdo con la reivindicación 13, proporcionado como software para ejecutarse en dicha unidad (200) de procesamiento; y

una memoria (140) configurada para almacenar representaciones codificadas (700) de bloques (600) de imágenes.

15.- Un método para comprimir un bloque (600) de imágenes que comprende múltiples píxeles (610), comprendiendo dicho método:

determinar un código (710) de colores que es una representación de color, que comprende un valor de componente de color rojo, un valor de componente de color verde y un valor de componente de color azul, de los colores de dichos píxeles (610) en dicho bloque (600) de imágenes;

seleccionar un conjunto de modificador de intensidad de una tabla (500) de intensidad que comprende múltiples conjuntos de modificador de intensidad, en el que dicho conjunto de modificador de intensidad comprende múltiples modificadores de intensidad para modificar la intensidad de dichos píxeles (610) en dicho bloque (600) de imágenes añadiendo un modificador de intensidad a cada uno de dicho valor de componente de color rojo, dicho valor de componente de color verde y dicho valor de componente de color azul de dicha representación de color;

proporcionar un código (720) de intensidad que permite la identificación de dicho conjunto de modificador de intensidad seleccionado de dicha tabla (500) de intensidad; y

seleccionar, para cada píxel (610) en dicho bloque (600) de imágenes, un índice (730) de intensidad asociado con un modificador de intensidad de dicho conjunto de modificador de intensidad.

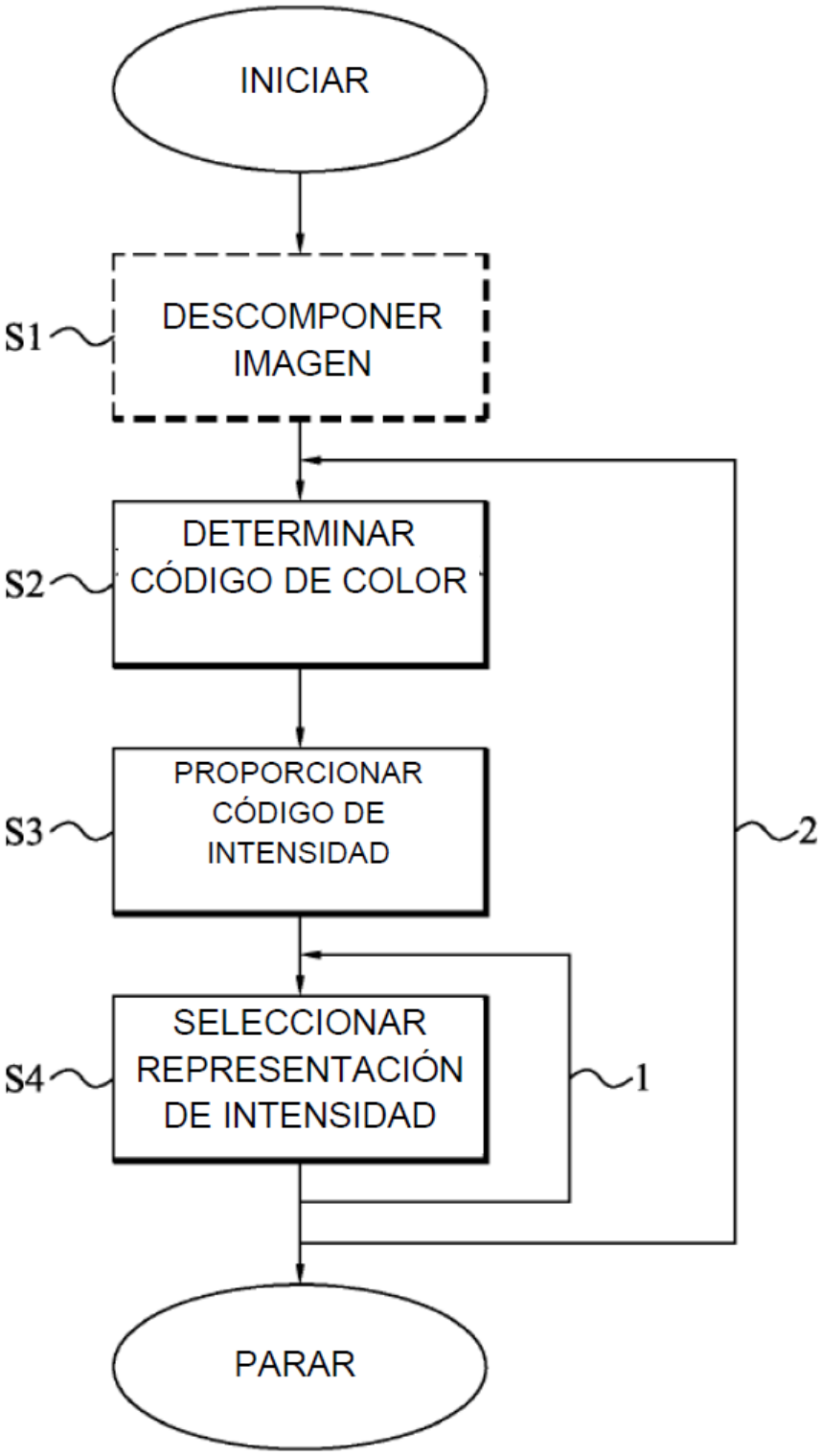
INICIAR
S1
DESCOMPONER IMAGEN
S2
DETERMINAR CÓDIGO DE COLOR
S3
PROPORCIONAR CÓDIGO DE INTENSIDAD
S4
SELECCIONAR REPRESENTACIÓN DE INTENSIDAD
1
2
PARAR

Fig. 1

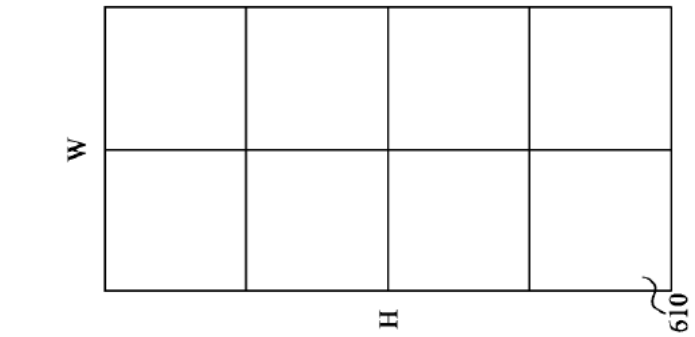

Fig. 3

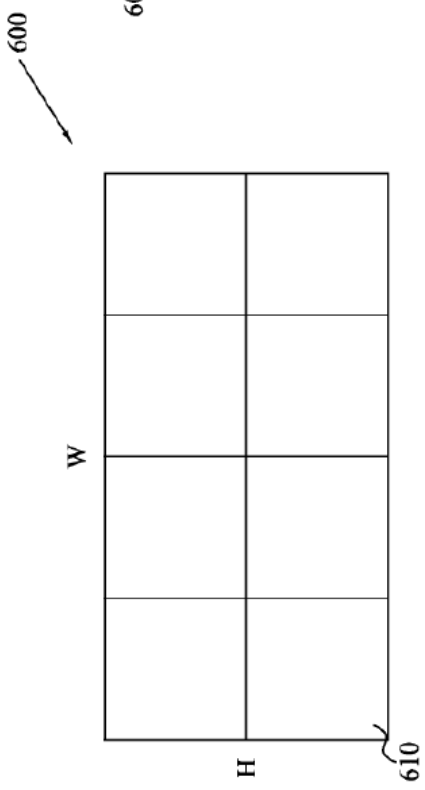

Fig. 2

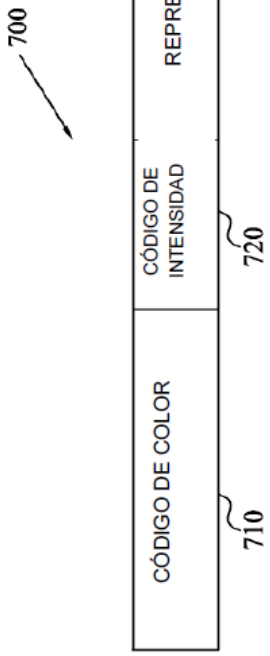

Fig. 4

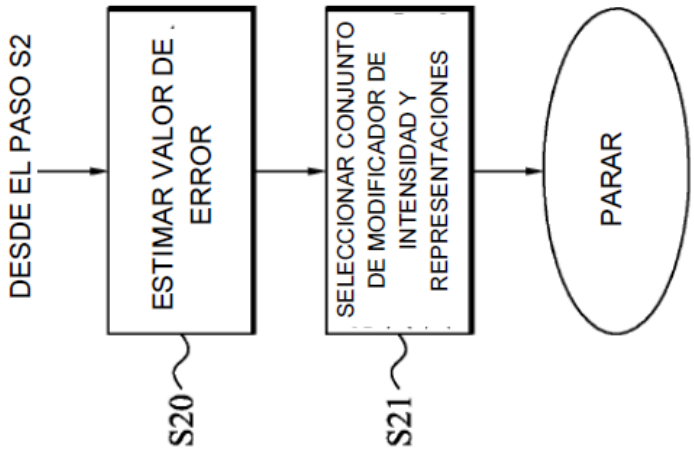
DESDE EL PASO S2
ESTIMAR VALOR DE ERROR
SELECCIONAR CONJUNTO DE MODIFICADOR DE INTENSIDAD Y REPRESENTACIONES
PARAR
S20
S21

Fig. 6

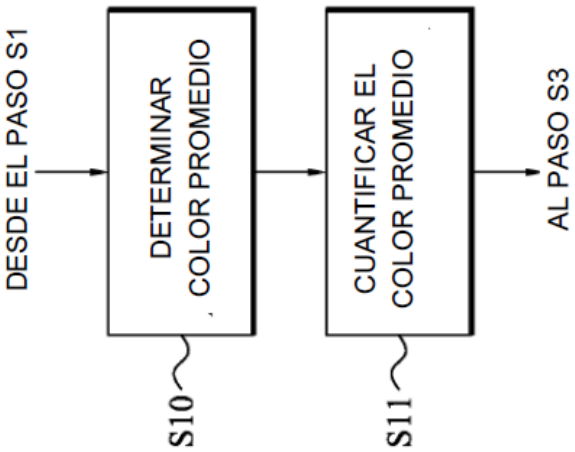
DESDE EL PASO S1
DETERMINAR COLOR PROMEDIO
CUANTIFICAR EL COLOR PROMEDIO
AL PASO S3
S10
S11

Fig. 5

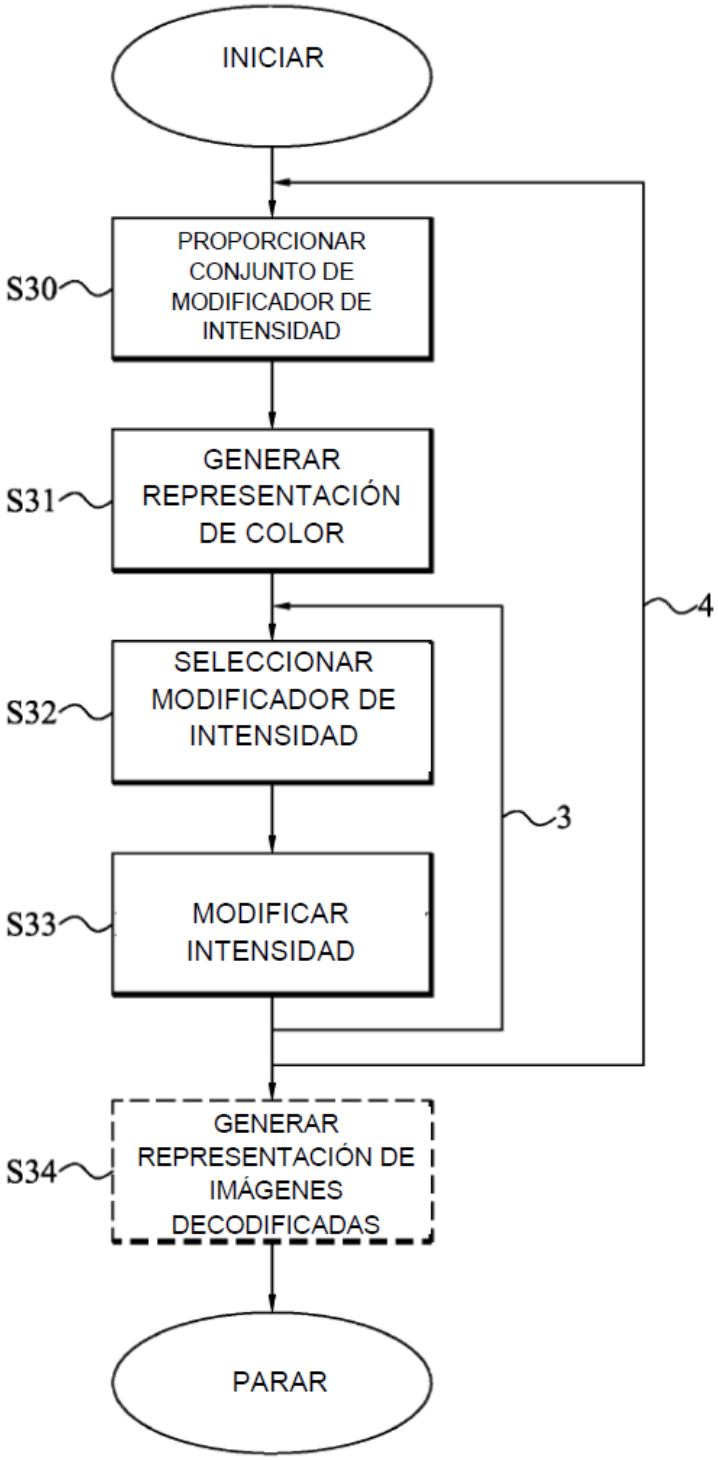
INICIAR
PROPORCIONAR CONJUNTO DE MODIFICADOR DE INTENSIDAD
S30
GENERAR REPRESENTACIÓN DE COLOR
S31
SELECCIONAR MODIFICADOR DE INTENSIDAD
S32
MODIFICAR INTENSIDAD
S33
GENERAR REPRESENTACIÓN DE IMÁGENES DECODIFICADAS
S34
PARAR
3
4

Fig. 7

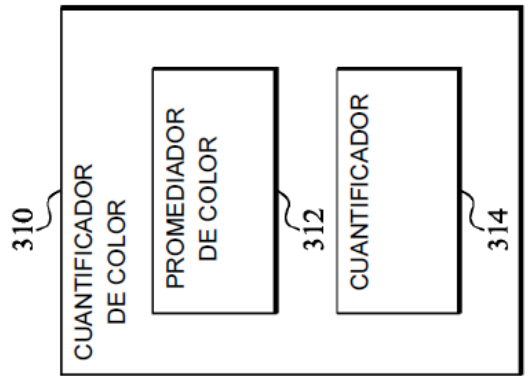

Fig. 15

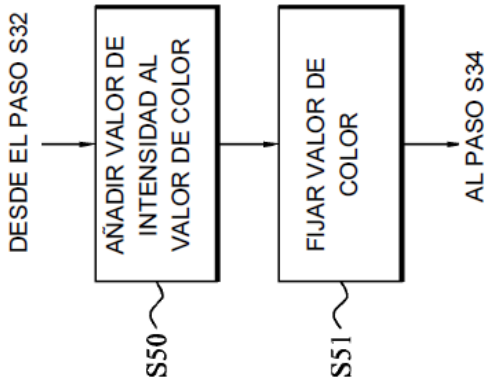

Fig. 9

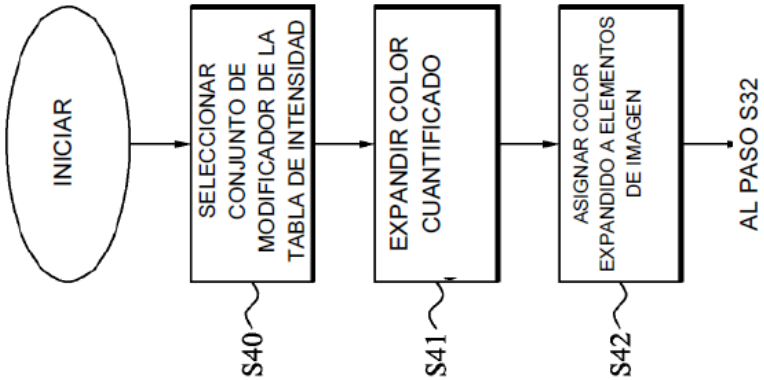

Fig. 8

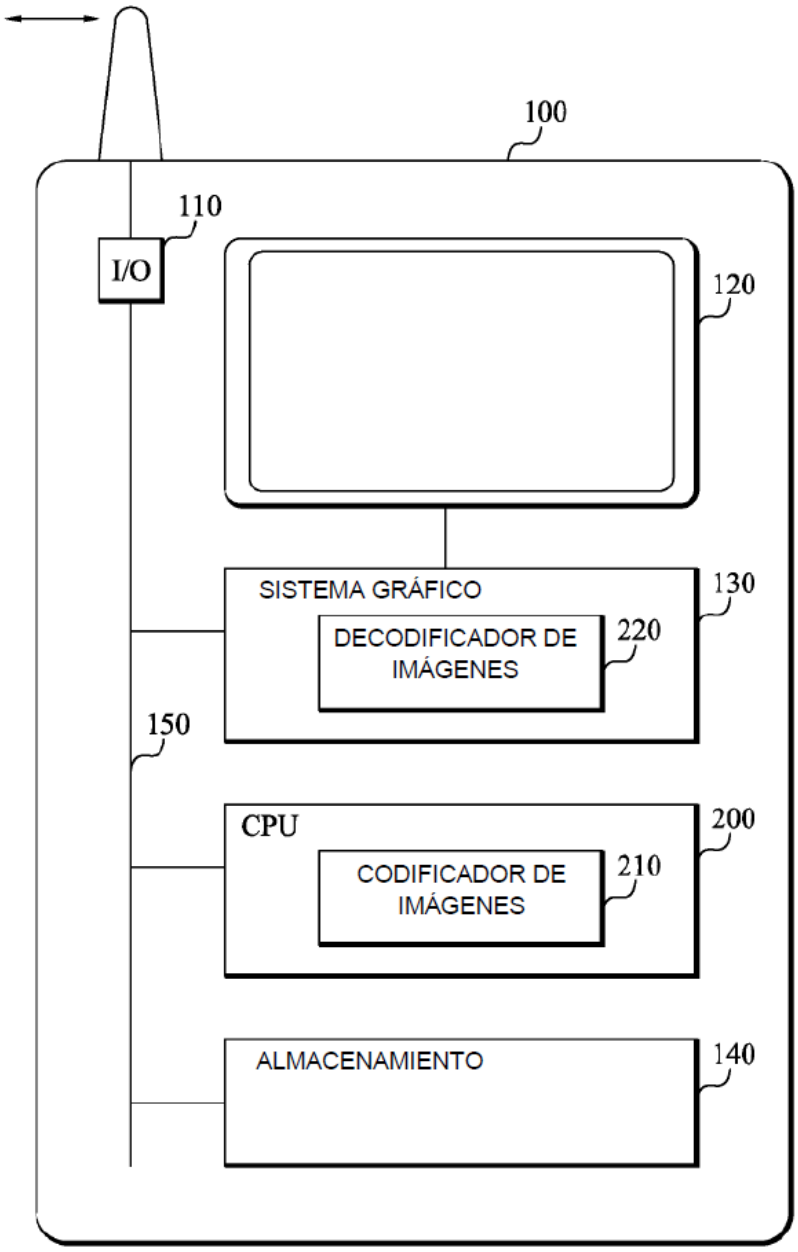
100
110
I/O
120
SISTEMA GRÁFICO
130
DECODIFICADOR DE IMÁGENES
220
150
CPU
200
CODIFICADOR DE IMÁGENES
210
ALMACENAMIENTO
140

Fig. 10

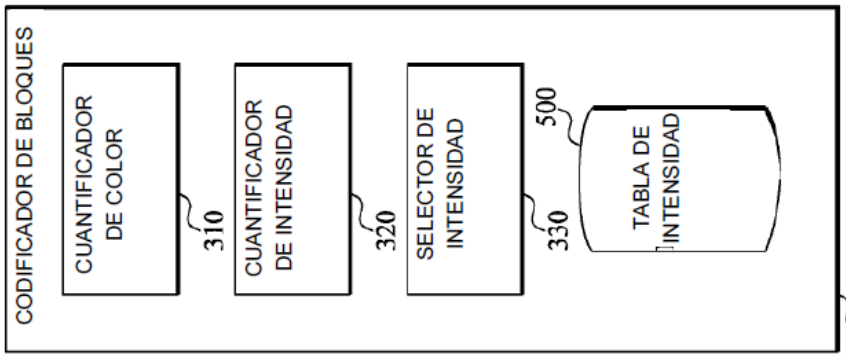

Fig. 13

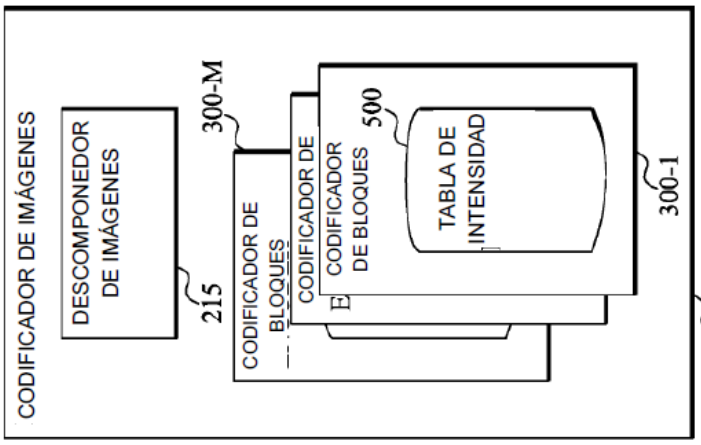

Fig. 12

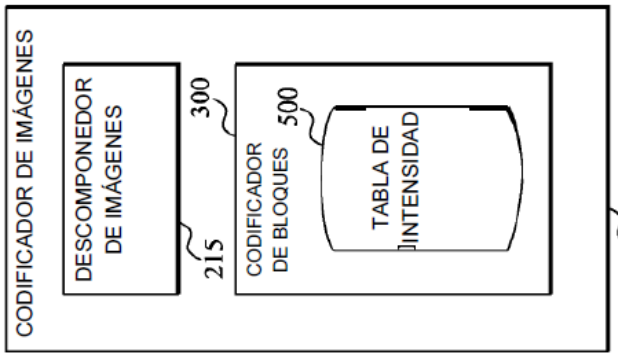

Fig. 11

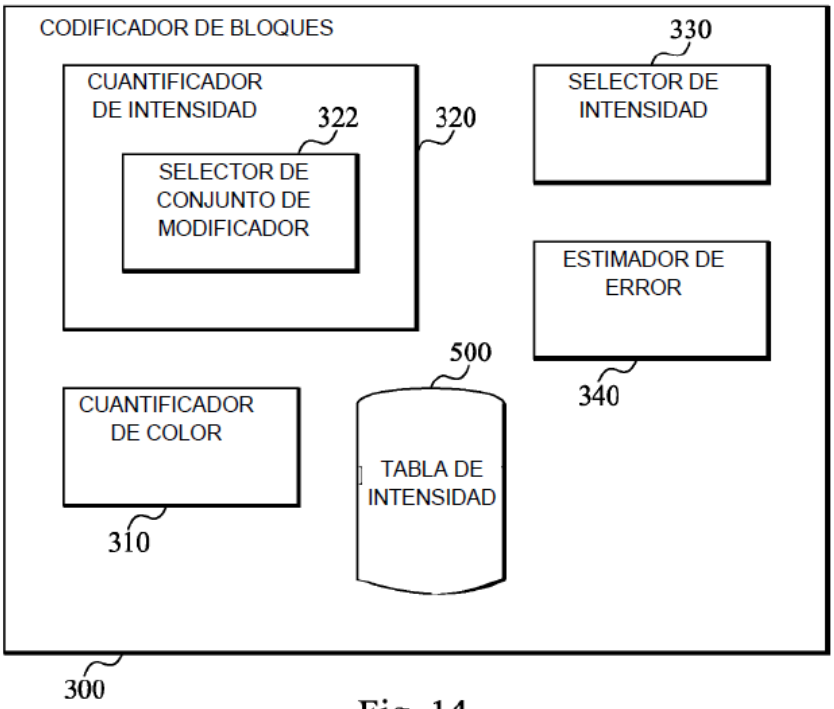
CODIFICADOR DE BLOQUES
330
CUANTIFICADOR DE INTENSIDAD
322
320
SELECTOR DE INTENSIDAD
SELECTOR DE CONJUNTO DE MODIFICADOR
ESTIMADOR DE ERROR
500
340
CUANTIFICADOR DE COLOR
TABLA DE INTENSIDAD
310
300

Fig. 14

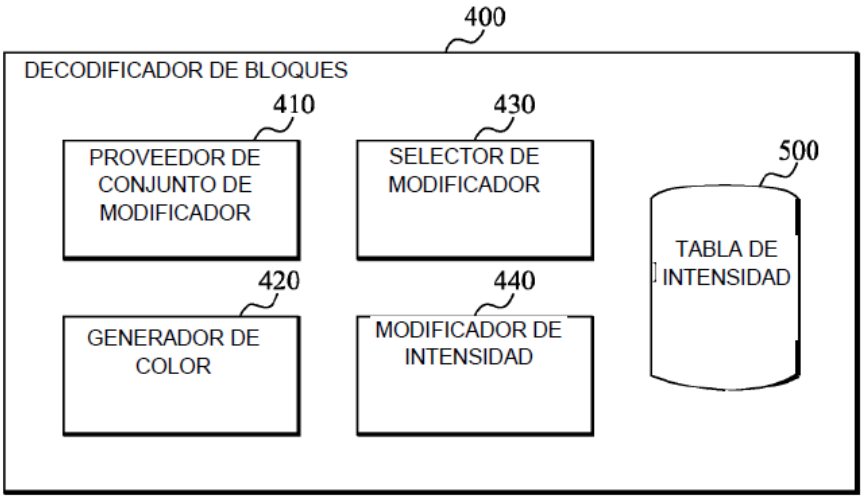
400
DECODIFICADOR DE BLOQUES
410
430
500
PROVEEDOR DE CONJUNTO DE MODIFICADOR
SELECTOR DE MODIFICADOR
TABLA DE INTENSIDAD
420
440
GENERADOR DE COLOR
MODIFICADOR DE INTENSIDAD

Fig. 18

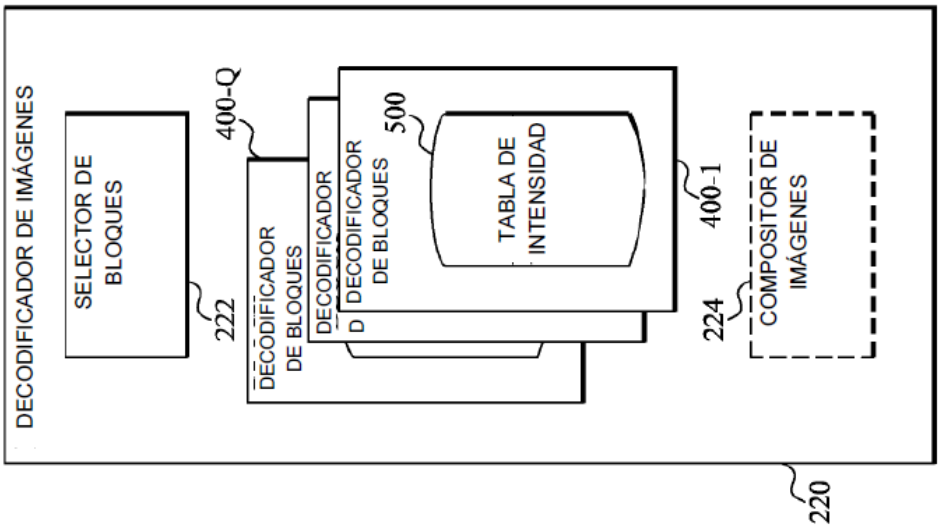
DECODIFICADOR DE IMÁGENES
SELECTOR DE BLOQUES
222
400-Q
DECODIFICADOR DE BLOQUES
DECODIFICADOR D
DECODIFICADOR DE BLOQUES
500
TABLA DE INTENSIDAD
400-1
224
COMPOSITOR DE IMÁGENES
220

Fig. 17

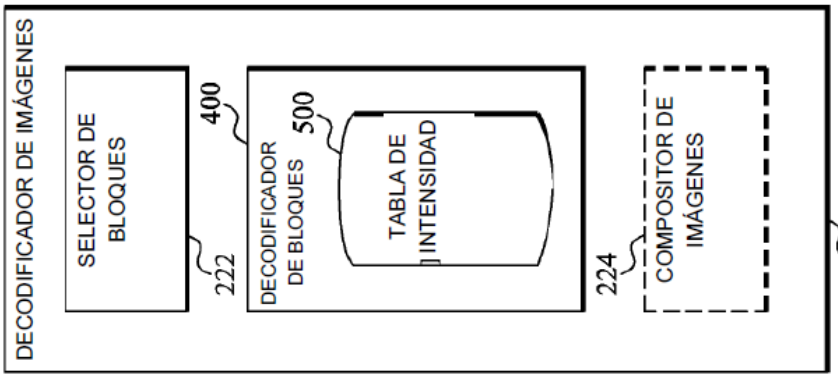
DECODIFICADOR DE IMÁGENES
SELECTOR DE BLOQUES
222
400
DECODIFICADOR DE BLOQUES
500
TABLA DE INTENSIDAD
224
COMPOSITOR DE IMÁGENES
220

Fig. 16

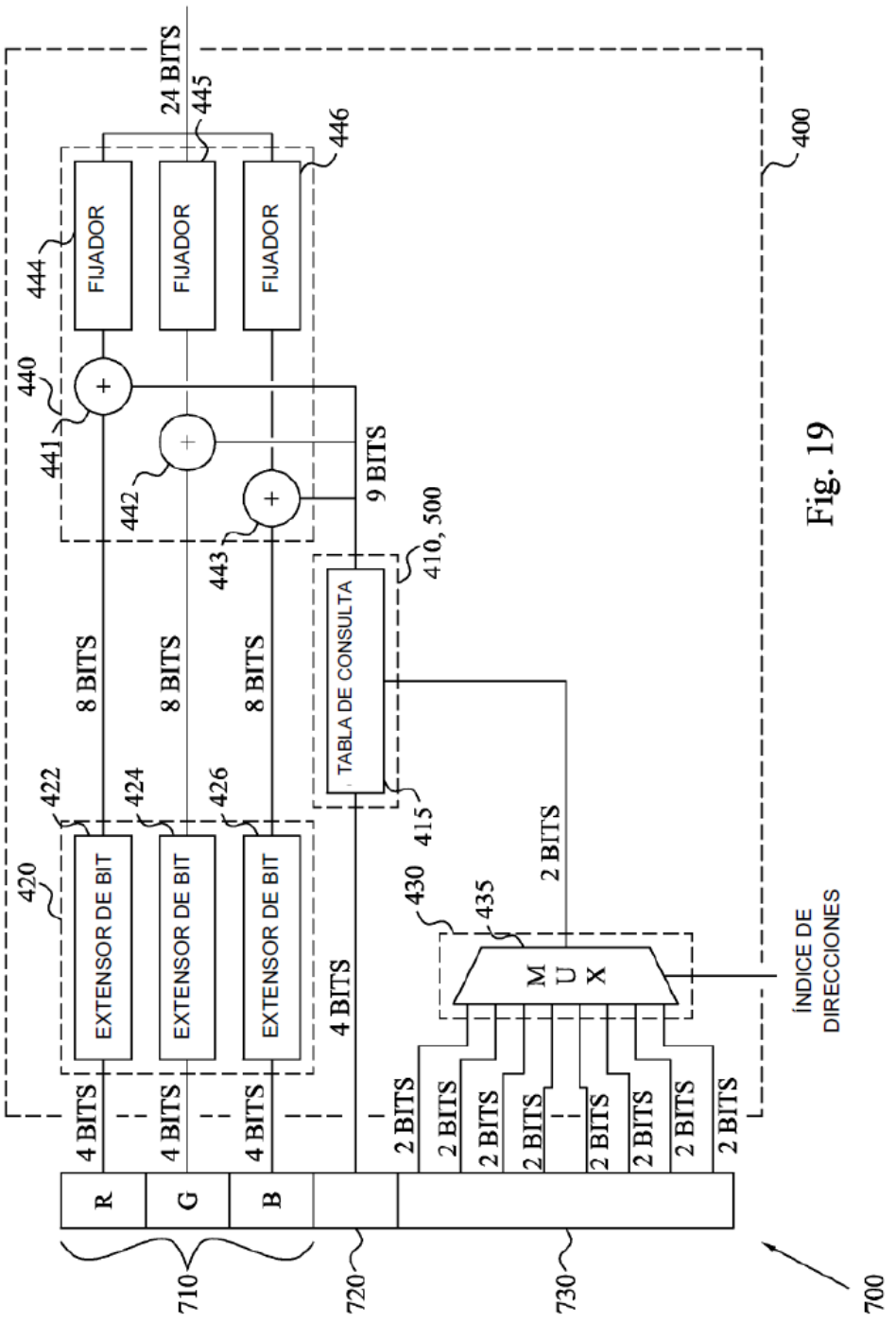
400
24 BITS
445
446
444
FIJADOR
FIJADOR
FIJADOR
440
441
442
443
9 BITS
410, 500
TABLA DE CONSULTA
415
8 BITS
8 BITS
8 BITS
422
424
426
420
EXTENSOR DE BIT
EXTENSOR DE BIT
EXTENSOR DE BIT
4 BITS
430
435
2 BITS
M
U
X
ÍNDICE DE DIRECCIONES
4 BITS
4 BITS
4 BITS
2 BITS
2 BITS
2 BITS
2 BITS
2 BITS
2 BITS
2 BITS
2 BITS
R
G
B
710
720
730
700

Fig. 19

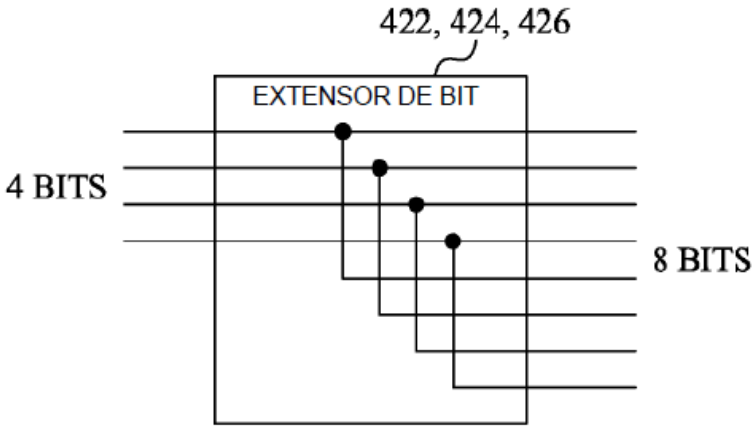
422, 424, 426
EXTENSOR DE BIT
4 BITS
8 BITS

Fig. 20

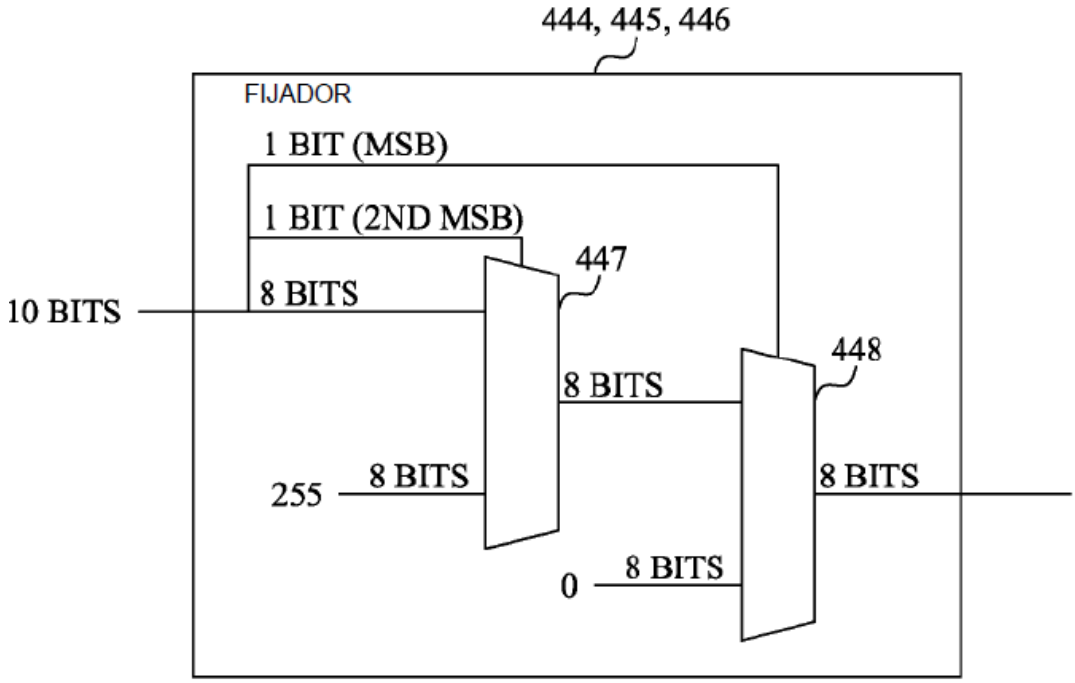
444, 445, 446
FIJADOR
1 BIT (MSB)
1 BIT (2ND MSB)
8 BITS
10 BITS
447
448
8 BITS
8 BITS
255
8 BITS
8 BITS
0
8 BITS

Fig. 22

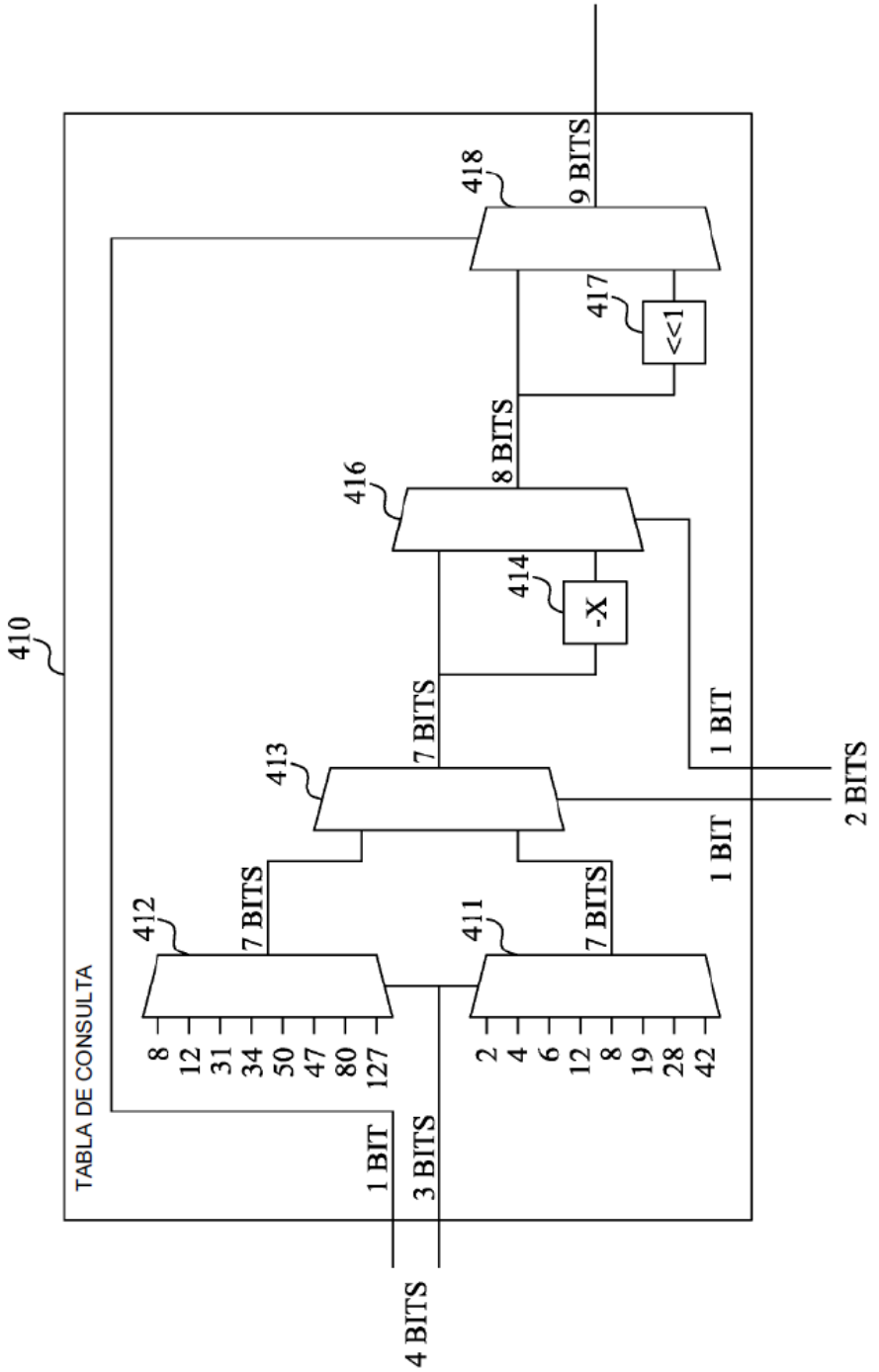
410
TABLA DE CONSULTA
412
7 BITS
8
12
31
34
50
47
80
127
411
7 BITS
2
4
6
12
8
19
28
42
413
7 BITS
1 BIT
1 BIT
2 BITS
414
-X
416
8 BITS
417
<<1
418
9 BITS
1 BIT
3 BITS
4 BITS

Fig. 21