(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 388 942**

(51) Int. CI.:

G10L 19/02	(2006.01)
H03M 7/40	(2006.01)
H03M 7/46	(2006.01)
H03M 7/48	(2006.01)
G06F 17/20	(2006.01)
G10L 19/14	(2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(96) Número de solicitud europea: **08017491 .5**

(96) Fecha de presentación: **03.09.2003**

(97) Número de publicación de la solicitud: **2006840**

(97) Fecha de publicación de la solicitud: **24.12.2008**

(54) Título: **Codificación entrópica adaptando la codifidación entre modos de nivel y de longitud de serie/nivel**

(30) Prioridad:
04.09.2002 US 408538 P
25.08.2003 US 647923

(45) Fecha de publicación de la mención BOPI:
19.10.2012

(45) Fecha de la publicación del folleto de la patente:
19.10.2012

(73) Titular/es:
MICROSOFT CORPORATION
ONE MICROSOFT WAY
REDMOND, WA 98052-6399, US

(72) Inventor/es:
Mehrohtra, Sanjeev y
Chen, Wei-ge

(74) Agente/Representante:
Carpintero López, Mario

DESCRIPCIÓN

Codificación entrópica adaptando la codificación entre modos de nivel y de longitud de serie/nivel

Campo

La presente invención se refiere a codificación entrópica adaptativa de datos de audio. Por ejemplo, un codificador de audio conmuta entre codificación de Huffman de niveles directos de datos de audio cuantificados y codificación aritmética de longitudes y niveles de serie de datos de audio cuantificados.

Antecedentes

Con la introducción de los discos compactos, las redes de telefonía inalámbrica digital y el suministro de audio por Internet, el audio digital se ha convertido en algo común. Los ingenieros hacen uso de una diversidad de técnicas para procesar eficazmente el audio digital mientras que a la vez mantienen la calidad del audio digital. El hecho de entender estas técnicas, ayuda a entender cómo se representa y se procesa la información de audio en un ordenador.

I. Representación de Información de Audio en un Ordenador

Un ordenador procesa la información de audio como una serie de números que representan la información de audio. Por ejemplo, un número simple puede representar una muestra de audio, que tenga un valor de amplitud (es decir, una intensidad de sonido) en un instante particular. Diversos factores afectan a la calidad de la información de audio, incluyendo profundidad de muestra, frecuencia de muestreo, y modo canal.

La profundidad de muestra (o precisión) indica la gama de números utilizados para representar una muestra. A mayor número de valores posibles para la muestra, más alta es la calidad debido a que el número puede capturar más variaciones leves de amplitud. Por ejemplo, una muestra de 8 bits tiene 256 valores posibles, mientras que una muestra de 16 bits tiene 65.536 valores posibles.

La frecuencia de muestreo (medida normalmente como el número de muestras por segundo), también afecta a la calidad. Cuanto más elevada sea la frecuencia de muestreo, más alta es la calidad debido a que se pueden representar más frecuencias de sonido. Algunas frecuencias de muestreo habituales son 8.000, 11.025, 22.050, 32.000, 44.100, 48.000 y 96.000 muestras/ segundo.

La Tabla 1 muestra varios formatos de audio con diferentes niveles de calidad, junto con los costes correspondientes de velocidad de bit bruta.

Tabla 1: Velocidades de bit para información de audio de calidad diferente

Calidad	**Profundidad de Muestra (bits/muestra)**	**Frecuencia de Muestreo (muestras/segundo)**	**Modo**	**Velocidad de Bit Bruta (bits/segundo)**
Telefonía de Internet	8	8.000	mono	64.000
Teléfono	8	11.025	mono	88.200
Audio de CD	16	44.100	estéreo	1.411.200
Audio de alta calidad	16	48.000	estéreo	1.536.000

Según muestra la Tabla 1, el coste de la información de audio de alta calidad tal como el audio de CD, es una alta velocidad de bit. La información de audio de alta calidad consume grandes cantidades de capacidad de almacenamiento en ordenador y de transmisión. Las compañías y los consumidores dependen sin embargo crecientemente de los ordenadores, para crear, distribuir y reproducir contenidos de audio de alta calidad.

II. Compresión y Descompresión de Audio

Muchos ordenadores y redes de ordenadores carecen de recursos para procesar el audio digital en bruto. La compresión (conocida también como codificación), reduce el coste del almacenaje y la transmisión de información de audio al convertir la información a una forma de velocidad de bit más baja. La compresión puede ser sin pérdidas (en la que no se ve perjudicada la calidad) o con pérdidas (en la que se ve perjudicada la calidad, pero la reducción de velocidad de bit mediante la compresión sin pérdidas es más drástica). La descompresión (también llamada descodificación) extrae una versión reconstruida de la información original a partir de la forma comprimida.

En general, el objetivo de la compresión de audio consiste en representar digitalmente las señales de audio para proporcionar una máxima calidad de señal con la menor cantidad de bits posible. Un sistema convencional de

codificador/ descodificador (“códec”) de audio utiliza codificación de sub-banda/ transformación, cuantificación, control de velocidad, y codificación de longitud variable, para conseguir su compresión. La cuantificación y otras técnicas de compresión con pérdidas introducen ruido potencialmente audible en la señal de audio. La audibilidad del ruido depende de cuánto ruido exista y de cuánto de ese ruido perciba el oyente. El primer factor se refiere principalmente a la calidad objetiva, mientras que el segundo factor depende de la percepción humana del sonido. La codificación de audio convencional comprime entonces sin pérdidas los datos cuantificados utilizando codificación de longitud variable para reducir aún más la velocidad de bit.

A. Compresión y Descompresión de Datos de Audio con Pérdidas

Convencionalmente, un codificador de audio utiliza una diversidad de técnicas diferentes de compresión con pérdidas. Estas técnicas de compresión con pérdidas incluyen típicamente transformaciones de frecuencia, modelación/ ponderación perceptual, y cuantificación. La descompresión correspondiente incluye cuantificación inversa, ponderación inversa, y transformaciones inversas de frecuencia.

Las técnicas de transformación de frecuencia convierten datos en una forma que hace que sea más fácil separar la información perceptualmente importante de la información perceptualmente no importante. La información menos importante puede ser sometida entonces a compresión con más pérdidas, mientras que la información más importante se reserva, con el fin de proporcionar la calidad mejor percibida para una velocidad de bit dada. Un transformador de frecuencia recibe típicamente las muestras de audio y las convierte en datos en el dominio de la frecuencia, denominados a veces coeficientes de frecuencia o coeficientes espectrales.

La mayor parte de la energía de los sonidos naturales tales como la palabra y la música, se concentra en la gama de baja frecuencia. Esto significa que, estadísticamente, las gamas de frecuencias más altas tendrán más coeficientes de frecuencia que sean cero o próximos a cero, reflejando la falta de energía en las gamas de frecuencias más altas.

La modelación perceptual implica procesar datos de audio de acuerdo con un modelo del sistema auditivo humano para mejorar la calidad percibida de la señal de audio reconstruida para una velocidad de bit dada. Por ejemplo, un modelo auditivo considera típicamente la gama de bandas críticas y de audición humanas. Utilizando los resultados de la modelación perceptual, un codificador configura el ruido (por ejemplo, el ruido de cuantificación) de los datos de audio con el objetivo de minimizar la audibilidad del ruido para una velocidad de bit dada. Mientras que el codificador debe introducir ruido a veces (por ejemplo, ruido de cuantificación) para reducir la velocidad de bit, la ponderación permite que el codificador ponga más ruido en las bandas en las que es menos audible, y viceversa.

La cuantificación cartografía las gamas de valores de entrada en los valores simples, introduciendo pérdidas irreversibles de información o de ruido de cuantificación, así como permitiendo también que un codificador regule la calidad y la velocidad de bit de la salida. A veces, el codificador realiza la cuantificación junto con un controlador de velocidad que ajusta la cuantificación para regular la velocidad de bit y/o la calidad. Existen varias clases de cuantificación, incluyendo la adaptativa y la no adaptativa, la escalar y la vectorial, la uniforme y la no uniforme. La ponderación perceptual puede ser considerada como una forma de cuantificación no uniforme.

La cuantificación inversa y la ponderación inversa reconstruyen los datos de coeficientes de frecuencia cuantificados, ponderados, en una aproximación de los datos de coeficientes de frecuencia originales. El transformador de frecuencia inversa convierte a continuación los datos de coeficientes de frecuencia en muestras de audio reconstruidas en el dominio del tiempo.

B. Compresión y Descompresión de Datos de Audio sin Pérdidas

Convencionalmente, el codificador de audio utiliza una o más de una diversidad de técnicas diferentes de compresión sin pérdidas. En general, las técnicas de compresión sin pérdidas incluyen codificación de longitud de serie, codificación de Huffman, y codificación aritmética. Las técnicas de descompresión correspondientes incluyen descodificación de longitud de serie, descodificación de Huffman, y descodificación aritmética.

La codificación de longitud de serie es una técnica de compresión simple, bien conocida, utilizada para videocámaras, texto, y otros tipos de contenidos. En general, la codificación de longitud de serie sustituye una serie (es decir, una secuencia) de símbolos consecutivos que tienen el mismo valor, por el valor y la longitud de la serie. En la descodificación de longitud de serie, la secuencia de símbolos consecutivos es reconstruida a partir del valor de serie y de la longitud de serie. Se han desarrollado numerosas variaciones de codificación/ descodificación de longitud de serie. Para una información adicional en torno a la codificación/ descodificación de longitud de serie y algunas de sus variaciones, véase, por ejemplo, Bell et al., Compresión de Texto, Prentice Hall PTR, páginas 105-107, 1990; Gibson et al., Compresión Digital para Multimedia, Morgan Kaufmann, páginas 17.62, 1998; Patente U.S. núm. 6.304.928 de Mairs, et al.; Patente U.S. núm. 5.883.633 de Gill et al., y Patente U.S. núm. 6.233.017 de Chaddha.

La codificación de nivel de serie es similar a la codificación de longitud de serie en el sentido de qué secuencias de símbolos consecutivos que tienen el mismo valor, son sustituidos por longitudes de la series. El valor de las secuencias es el valor predominante (por ejemplo, 0) en los datos, y las secuencias están separadas por uno o más niveles que tienen un valor diferente (por ejemplo, un valor que no es cero).

Los resultados de la codificación de longitud de serie (por ejemplo, los valores de serie y las longitudes de serie) o de la codificación de nivel de serie, pueden ser codificados en Huffman para reducir aún más la velocidad de bit. Si es así, los datos codificados de Huffman son descodificados en Huffman con anterioridad a la descodificación de longitud de serie.

La codificación de Huffman es otra técnica de compresión bien conocida que se utiliza para videocámaras, texto y otros tipos de contenidos. En general, una tabla de códigos de Huffman asocia códigos de Huffman de longitud variable con valores únicos de símbolo (o combinaciones únicas de valores). Los códigos más cortos son asignados a valores de símbolo más probables, y los códigos más largos son asignados a valores de símbolo menos probables. Las probabilidades son calculadas para ejemplos típicos de alguna clase de contenido. O, las probabilidades son calculadas para datos recién codificados o datos que van a ser codificados, en cuyo caso los códigos de Huffman se adaptan a las probabilidades de cambio para los valores únicos de símbolo. En comparación con la codificación de Huffman, la codificación de Huffman adaptativa reduce normalmente la velocidad de bit de los datos comprimidos al incorporar probabilidades más precisas para los datos, pero también puede que se necesite transmitir información extra que especifique los códigos de Huffman.

Para codificar símbolos, el codificador de Huffman sustituye valores de símbolo por códigos de Huffman de longitud variable asociados a los valores de símbolo en la tabla de códigos de Huffman. Para descodificar, el descodificador de Huffman sustituye los códigos de Huffman por los valores de código asociados a los códigos de Huffman.

En la codificación escalar de Huffman, una tabla de códigos de Huffman asocia un código de Huffman único a un valor, por ejemplo, un nivel directo de un valor de dato cuantificado. En la codificación vectorial de Huffman, una tabla de códigos de Huffman asocia un código de Huffman único a una combinación de valores, por ejemplo, un grupo de niveles directos de valores de datos cuantificados en un orden particular. La codificación vectorial de Huffman puede conducir a una mejor reducción de la velocidad de bit que la codificación escalar de Huffman (por ejemplo, permitiendo que el codificador aproveche fraccionalmente las probabilidades de los códigos binarios de Huffman). Por otra parte, el código de cifrado y descifrado para la codificación vectorial de Huffman puede ser extremadamente grande cuando los códigos simples representan grandes grupos de símbolos o los símbolos tienen grandes gamas de valores potenciales (debido al gran número de combinaciones potenciales). Por ejemplo, si el tamaño del alfabeto es de 256 (para valores de 0 a 255 por símbolo), y el número de símbolos por vector es de 4, el número de combinaciones potenciales es de 256^4 = 4.294.967.296. Esto consume memoria y recursos de procesamiento al calcular el código de cifrado y descifrado y hallar los códigos de Huffman, y consume recursos de transmisión al transmitir el código de cifrado y descifrado.

Se han desarrollado numerosas variaciones de codificación/ descodificación de Huffman. Para una información adicional acerca de la codificación/ descodificación de Huffman y de algunas de sus variaciones, véase, por ejemplo, Bell et al., Compresión de Texto, Prentice Hall PTR, páginas 105-107, 1990; Gibson et al., Compresión Digital para Multimedia, Morgan Kaufmann, páginas 17-62, 1998.

La Patente U.S. núm. 6.223.162 de Chen et al., describe codificación multi-nivel de longitud de serie de datos de audio. Una transformación de frecuencia produce una serie de valores de coeficientes de frecuencia. Para porciones de un espectro de frecuencia en el que el valor predominante sea cero, un codificador multi-nivel de longitud de serie correlaciona las secuencias de valores cero con valores distintos de cero, y asigna palabras de código de longitud variable. Un codificador utiliza un código de cifrado y descifrado especializado, con respecto a la probabilidad de recibir una secuencia de entrada de coeficientes espectrales de valor cero, seguido de un coeficiente de valor distinto de cero. Un descodificador correspondiente asocia una palabra de código de longitud variable con una secuencia de coeficientes de valor cero y de coeficientes adyacentes de valor distinto de cero.

La Patente U.S. núm. 6.377.930 de Chen et al., describe codificación variable respecto a longitud variable de datos de audio. Un codificador asigna un código de longitud variable a un grupo de tamaño variable de valores de coeficientes de frecuencia.

La Patente U.S. núm. 6.300.888 de Chen et al., describe conmutación de modo de codificación entrópica para codificar audio en el dominio de la frecuencia. Un codificador de audio en el dominio de la frecuencia, realiza una selección entre diferentes modos de codificación entrópica de acuerdo con las características de una corriente de entrada. En particular, la corriente de entrada se divide en gamas de frecuencia de acuerdo con criterios estadísticos derivados del análisis estadístico de una entrada típica o real que va a ser codificada. A cada gama se asigna un codificador de entropía optimizado para codificar el tipo de datos de la gama. Durante la codificación y la descodificación, un selector de modo aplica el procedimiento correcto a las diferentes gamas de frecuencia. Los contornos de la partición pueden estar decididos por anticipado, permitiendo que el descodificador conozca implícitamente qué procedimiento de descodificación debe aplicar a los datos codificados. O, se pueden utilizar disposiciones adaptativas, en las que los contornos estén señalizados en la corriente de salida para indicar un cambio de modo de codificación para los datos posteriores. Por ejemplo, un contorno de partición separa principalmente coeficientes de frecuencia cuantificados en cero de principalmente coeficientes cuantificados como no cero, y a continuación aplica codificadores optimizados para tales datos.

Para detalles adicionales acerca de las patentes de Chen, véanse las propias patentes.

La codificación aritmética es otra técnica de compresión bien conocida utilizada para videocámaras y otros tipos de

contenidos. La codificación aritmética se utiliza a veces en aplicaciones en las que el número óptimo de bits para codificar un símbolo de entrada dado, es un número fraccionario de bits, y en casos en los que existe una correlación estadística entre ciertos símbolos de entrada individuales. La codificación aritmética incluye generalmente representar una secuencia de entrada como un número único dentro de una gama dada. Típicamente, el número es un número fraccionario entre 0 y 1. Los símbolos de la secuencia de entrada están asociados a gamas que ocupan porciones del espacio entre el 0 y el 1. Las gamas se calculan en base a la probabilidad de que ocurra el símbolo particular en la secuencia de entrada. El número fraccionario utilizado para representar la secuencia de entrada se construye con referencia a las gamas. Por lo tanto, las distribuciones de probabilidad para los símbolos de entrada son importantes en los esquemas de codificación aritmética.

En codificación aritmética basada en contexto, las diferentes distribuciones de probabilidad para los símbolos de entrada están asociadas a diferentes contextos. La distribución de probabilidad utilizada para codificar la secuencia de entrada, cambia cuando cambia el contexto. El contexto puede ser calculado midiendo diferentes factores que se espera que afecten a la probabilidad de que aparezca un símbolo de entrada particular en una secuencia de entrada. Para información adicional acerca de la codificación/ descodificación aritmética y de algunas de sus variaciones, véase Nelson, El Libro de Compresión de Datos, “Lo Mejor de Huffman: Codificación Aritmética”, Capítulo 5, pp. 123-65 (1992).

Diversos estándares y sistemas códec utilizan compresión y descompresión sin pérdidas, incluyendo versiones del codificador y descodificador de Windows Media Audio [“WMA”] de Microsoft Corporation. Otros sistemas códec han sido proporcionados o especificados por el estándar Audio Layer 3 [“MP3”], de Motion Picture Experts Group, el estándar Advanced Audio Coding [“AAC”], de Motion Picture Experts Group 2, y Dolby AC3. Para información adicional, véanse los estándares respectivos o las publicaciones técnicas.

La codificación de longitud de serie multinivel para codificar audio en el dominio de la frecuencia es conocido a partir del documento WO00/36753.

En cualquier caso, las ventajas de las técnicas y sistemas anteriores para la compresión sin pérdidas de datos de audio, no tienen las ventajas de la presente invención.

Sumario

La invención proporciona un procedimiento según las reivindicaciones 1 y 5.

En resumen, la descripción detallada va dirigida a diversas técnicas y herramientas para la codificación y descodificación entrópica adaptativa de datos de audio. Las diversas técnicas y herramientas pueden ser utilizadas de forma combinada o independiente.

En un aspecto, un codificador selecciona una primera tabla de códigos a partir de un conjunto de tablas de códigos plurales en base al número de símbolos de un primer vector, y representa el primer vector con un código a partir de la primera tabla de códigos. La primera tabla de códigos puede incluir códigos para representar vectores probables que tengan un número de símbolos, y un código de escape para los vectores menos probables. El codificador también codifica un segundo vector que tiene un número diferente de símbolos. Por ejemplo, el primer vector tiene un número de símbolos mayor que el segundo vector, y tiene una probabilidad de ocurrencia más alta que el segundo vector. Para codificar el segundo vector, el codificador puede elegir una segunda tabla de códigos, diferente, basada en el número de símbolos del segundo vector. Si el segundo vector tiene un símbolo, el codificador puede representar el segundo vector utilizando una técnica de codificación sin-tabla.

En otro aspecto, un descodificador descodifica un primer vector al recibir un primer código, y busca el primer código en una primera tabla de códigos. Si el primer código es un código de escape, el descodificador recibe y descodifica un segundo código que no está en la primera tabla. Si el primer código no es un código de escape, el descodificador busca símbolos para el primer vector en la primera tabla de códigos, y los incluye en una corriente de datos descodificados. El número de símbolos del primer vector constituye la base de si el primer código es un código de escape. El descodificador puede descodificar el segundo código buscándolo en una segunda tabla. Si el segundo código es un código de escape, el descodificador recibe y descodifica un tercer código que representa el primer vector que no esté en la segunda tabla. Si el segundo código no es un código de escape, el descodificador busca símbolos para el primer vector en la segunda tabla e incluye los símbolos en la corriente de datos descodificados.

Las características y ventajas de las técnicas de codificación y descodificación entrópicas adaptativas, se pondrán de manifiesto a partir de la descripción detallada que sigue de varias realizaciones, realizada con referencia a los dibujos que se acompañan.

Breve descripción de los dibujos

La Figura 1 es un diagrama de bloques de un entorno de computación adecuado en el que pueden ser implementadas las realizaciones descritas;
la Figura 2 es un diagrama de bloques de un codificador de audio en el que pueden ser implementadas las realizaciones descritas;
la Figura 3 es un diagrama de bloques de un descodificador de audio en el que pueden ser implementadas las

realizaciones descritas;
la Figura 4 es un diagrama de flujo que muestra una técnica de codificación de audio generalizada multi-modo;
la Figura 5 es un diagrama de flujo que muestra una técnica de codificación de audio multi-modo con cálculo de punto de conmutación adaptativo;
la Figura 6 es un diagrama de flujo que muestra una técnica de descodificación de audio generalizada multi-modo;
la Figura 7 es un diagrama de flujo que muestra una técnica de codificación vectorial de Huffman generalizada de dimensión variable;
la Figura 8 es un diagrama de flujo que muestra una técnica detallada para la codificación de datos de audio utilizando codificación vectorial de Huffman de dimensión variable;
la Figura 9 es un diagrama de flujo que muestra una técnica para codificación vectorial de Huffman de dimensión variable de niveles de señal directa, donde el codificador determina adaptativamente un punto de conmutación para cambiar a codificación de longitudes de serie y de niveles de señal;
la Figura 10 es un diagrama de flujo que muestra una técnica de descodificación vectorial de Huffman generalizada de dimensión variable;
la Figura 11 es un diagrama de flujo que muestra una técnica detallada para descodificar vectores codificados utilizando codificación vectorial de Huffman de dimensión variable;
la Figura 12 es un diagrama de flujo que muestra una técnica para la descodificación vectorial de Huffman, de dimensión variable, de niveles de señal directa donde el descodificador determina adaptativamente un punto de conmutación para cambiar a descodificación de longitudes de serie y de niveles de señal;
las Figuras 13A – 13D son distribuciones de probabilidad para niveles de longitud de no serie, en un esquema de codificación aritmética basada en contexto;
las Figuras 14A – 14H son distribuciones de probabilidad para diferentes longitudes de serie en un esquema de codificación aritmética basada en contexto;
las Figuras 15A – 15H son distribuciones de probabilidad para niveles codificados de longitud de serie en un esquema de codificación aritmética basada en contexto;
la Figura 16 es un diagrama de flujo que muestra una técnica para la codificación aritmética directa, basada en contexto, de coeficientes, donde el codificador determina un punto de conmutación para cambiar a codificación de longitudes y niveles de serie,
la Figura 17 es un diagrama de flujo que muestra una técnica de descodificación aritmética basada en contexto, donde el descodificador determina adaptativamente un punto de conmutación para cambiar a descodificación de longitudes de serie y de niveles de señal.

Descripción detallada

En las realizaciones descritas, un codificador de audio realiza varias técnicas de codificación entrópica adaptativa. Las técnicas de codificación entrópica adaptativa mejoran el comportamiento del codificador, reduciendo la velocidad de bit y/o mejorando la calidad. Un descodificador ejecuta las técnicas de descodificación entrópica correspondientes. Mientras que aquí se han descrito las técnicas como parte de un único sistema integrado, las técnicas pueden ser aplicadas por separado, potencialmente en combinación con otras técnicas.

El codificador y el descodificador de audio, procesan señales de audio discretas. En las realizaciones descritas, las señales de audio son coeficientes cuantificados a partir de señales de audio transformadas en frecuencia. Alternativamente, el codificador y el descodificador procesan otra clase de señal de audio discreta o de señal discreta que representa video u otra clase de información.

En las realizaciones descritas, el codificador y el descodificador de audio ejecutan varias técnicas. Aunque las operaciones relativas a estas técnicas se describen típicamente en un orden secuencial, particular, por motivos de presentación, se comprenderá que esta manera de descripción abarca reordenamientos menos importantes en cuanto al orden de operaciones. Además, por motivos de simplicidad, los diagramas de flujo no muestran típicamente las diversas formas en que pueden ser usadas las técnicas particulares conjuntamente con otras técnicas.

I. Entorno de Computación

La Figura 1 ilustra un ejemplo generalizado de un entorno (100) de computación adecuado en el que pueden ser implementadas las realizaciones descritas. No se pretende que el entorno (100) de computación sugiera ninguna limitación en cuanto al uso o la funcionalidad de la invención, puesto que la presente invención puede ser implementada en entornos muy diversos de propósito general o de propósito especial.

Con referencia a la Figura 1, el entorno (100) de computación incluye al menos una unidad (110) de procesamiento, y memoria (120). En la Figura 1, la configuración (130) más básica ha sido incluida dentro de una línea de puntos. La unidad (110) de procesamiento ejecuta instrucciones ejecutables con ordenador, y puede ser un procesador real o virtual. En un sistema multi-procesamiento, múltiples unidades de procesamiento ejecutan instrucciones ejecutables con ordenador para incrementar la potencia de procesamiento. La memoria (120) puede ser memoria volátil (por ejemplo, registros, caché, RAM), memoria no volátil (por ejemplo, ROM, EEPROM, memoria flash, etc.), o cualquier combinación de las dos. La memoria (120) almacena software (180) que implementa un codificador/ descodificador de audio que realiza codificación/ descodificación entrópica adaptativa de datos de audio.

Un entorno de computación puede tener características adicionales. Por ejemplo, el entorno (100) de computación incluye un dispositivo de almacenaje (140), uno o más dispositivos de entrada (150), uno o más dispositivos de salida (160), y una o más conexiones (170) de comunicación. Un mecanismo de interconexión (no representado), tal como un bus, un controlador, o una red, interconecta los componentes del entorno (100) de computación. Típicamente, el software del sistema operativo (no representado) proporciona un entorno operativo para otra ejecución de software en el entorno (100) de computación, y coordina las actividades de los componentes del entorno (100) de computación.

El almacenaje (140) puede ser extraíble o no extraíble, e incluye discos los magnéticos, cintas o casetes magnéticos, CD-ROM, CD-RW, DVD o cualquier otro medio que pueda ser utilizado para almacenar información y al que se pueda acceder dentro del entorno (100) computacional. El almacenaje (140) almacena instrucciones para el software (180) que implementa el codificador/ descodificador de audio que realiza la codificación/ descodificación entrópica de datos de audio.

El (los) dispositivo(s) de entrada(150) puede(n) ser un dispositivo táctil de entrada tal como un teclado, un ratón, un lápiz o una bola de seguimiento, un dispositivo de entrada por voz, un dispositivo de escaneo, un adaptador de red, u otro dispositivo que proporcione entrada al entorno (100) de computación. Para el audio, el (los) dispositivo(s) (150) de entrada puede(n) ser una tarjeta de sonido o un dispositivo similar que acepte una entrada de audio en forma analógica o digital, o un lector de CD-ROM que proporcione muestras de audio al entorno de computación. El (los) dispositivo(s) (160) de salida puede(n) ser una pantalla de visualización, una impresora, un altavoz, un transcriptor de CD/DVD, un adaptador de red, u otro dispositivo que proporcione una salida desde el entorno (100) de computación.

La(s) conexión(es) (170) de comunicación permite(n) la comunicación mediante un medio de comunicación con otra entidad de computación. El medio de comunicación transporta información tal como instrucciones ejecutables con ordenador, información de audio comprimido, u otros datos en una señal de datos modulada. Una señal de datos modulada es una señal que tiene una o más de sus características dispuestas o cambiadas para codificar información en la señal. A título de ejemplo, y sin limitación alguna, los medios de comunicación incluyen técnicas alámbricas o inalámbricas implementadas con un portador eléctrico, óptico, de RF, de infrarrojos, acústico, o de otro tipo.

La invención puede ser descrita en el contexto general de los medios susceptibles de lectura con ordenador. Los medios legibles con ordenador consisten en cualquier medio disponible al que se pueda acceder con un entorno de computación. A título de ejemplo, y sin limitación, dentro del entorno (100) de computación, los medios legibles con ordenador incluyen memoria (120), medios de almacenaje (140), medios de comunicación, y combinaciones de cualquiera de los anteriores.

La invención puede ser descrita en el contexto general de las instrucciones ejecutables con ordenador, tales como las incluidas en módulos de programa, que son ejecutadas en un entorno de computación sobre un procesador objetivo real o virtual. En general, los módulos de programa incluyen rutinas, programas, librerías, objetos, clases, componentes, estructuras de datos, etc., que realizan tareas particulares o que implementan tipos particulares de datos abstractos. La funcionalidad de los módulos de programa puede ser combinada o dividida en módulos de programa según se desee, en diversas realizaciones. Las instrucciones ejecutables con ordenador para módulos de programa pueden ser ejecutadas dentro de un entorno de computación local o distribuido.

Por motivos de presentación, la descripción detallada utiliza términos tales como “analizar”, “enviar”, “comparar” y “comprobar” para describir operaciones de ordenador en un entorno de computación. Estos términos son abstracciones de alto nivel para operaciones realizadas por un ordenador, y no deben ser confundidos con los actos llevados a cabo por un ser humano. Las operaciones reales de ordenador que corresponden a esos términos, varían dependiendo de la implementación.

II. Codificador y Descodificador de Audio Generalizado

La Figura 2 es un diagrama de bloques de un codificador (200) de audio generalizado en el que pueden ser implementadas las realizaciones descritas. El codificador (200) realiza codificación entrópica adaptativa de datos de audio. La Figura 3 es un diagrama de bloques de un descodificador (300) de audio generalizado en el que pueden ser implementadas las realizaciones descritas. El descodificador (300) descodifica datos de audio codificados.

Las relaciones mostradas entre módulos del interior del codificador y del descodificador indican un flujo de información en un ejemplo de codificador y de descodificador; no se muestran otras relaciones por motivos de simplicidad. Dependiendo de la implementación y del tipo de compresión que se desee, módulos de codificador o de descodificador pueden ser añadidos, omitidos, divididos en múltiples módulos, combinados con otros módulos, y/o sustituidos por módulos similares. En realizaciones alternativas, codificadores o descodificadores con diferentes módulos y/o con otras configuraciones realizan codificación y descodificación entrópica adaptativa de datos de audio.

A. Codificador de Audio Generalizado

El codificador (200) de audio generalizado incluye un selector (208), un pre-procesador (210) multi-canal, un divisor/ configurador (220) de mosaico, un transformador (230) de frecuencia, un modelador (240) de percepción, un ponderador (242), un transformador (250) multi-canal, un cuantificador (260), un codificador (270) de entropía, un controlador (280), un codificador (272) sin pérdidas mezclado/ puro y un codificador (274) de entropía asociado, y un multiplexor (290) de

corriente de bits ["MUX"]. A continuación se realiza la descripción de algunos de los módulos del codificador (200). Para la descripción de otros módulos del codificador (200) de algunas realizaciones, véanse las aplicaciones referenciadas en la sección de Datos de Aplicación Relacionados.

El codificador (200) recibe una serie en el tiempo de muestras (205) de audio de entrada a una profundidad y frecuencia de muestreo en formato modulado de código de pulso ["PCM"]. Las muestras (205) de audio de entrada pueden ser audio multi-canal (por ejemplo, modo estéreo, envolvente) o mono. El codificador (200) comprime las muestras (205) de audio y multiplexa la información producida por los diversos módulos del codificador (200) para presentar a la salida una corriente de bits (295) en un formato tal como un formato Windows Media Audio ["WMA"] o Advanced Streaming Format ["ASF"]. Alternativamente, el codificador (200) trabaja con otros formatos de entrada y/o de salida.

Inicialmente, el selector (208) elige entre los múltiples modos de codificación para las muestras (205) de audio. En la Figura 2, el selector (208) conmuta entre dos modos: un modo de codificación sin pérdidas mezclada/ pura y un modo de codificación con pérdidas. El modo de codificación sin pérdidas incluye el codificador (272) sin pérdidas mezclado/ puro, y se utiliza típicamente para compresión de alta calidad (y de alta velocidad de bit). El modo de codificación con pérdidas incluye componentes tales como el ponderador (242) y el cuantificador (260), y se utiliza típicamente para compresión de calidad ajustable (y velocidad de bit controlada). La decisión de selección en el selector (208) depende de la entrada de usuario (por ejemplo, un usuario que selecciona codificación sin pérdidas para realizar copias de audio de alta calidad) o de otros criterios. En otras circunstancias (por ejemplo, cuando la compresión con pérdidas falla en cuanto a la entrega de la calidad adecuada o sobre-produce bits), el codificador (200) puede conmutar desde codificación con pérdidas a codificación sin pérdidas mezclada/ pura para una trama o un conjunto de tramas.

El transformador (230) de frecuencia recibe las muestras (205) de audio y las convierte en datos en el dominio de la frecuencia. El transformador (230) de frecuencia presenta a la salida bloques de datos de coeficientes de frecuencia para el ponderador (242), y presenta a la salida información colateral tal como tamaños de bloque para el MUX (290). El transformador (230) de frecuencia presenta a la salida los coeficientes de frecuencia y la información colateral para el modelador (240) de percepción.

El modelador (240) de percepción modela propiedades del sistema auditivo humano para mejorar la calidad percibida de la señal de audio reconstruida para una cierta velocidad de bit. En general, el modelador (240) de percepción procesa los datos de audio de acuerdo con un modelo auditivo, y a continuación proporciona información al ponderador (242) que puede ser utilizada para generar factores de ponderación para los datos de audio. El modelador (240) de percepción utiliza uno cualquiera de diversos modelos auditivos y pasa información patrón de excitación u otra información al ponderador (242).

Como ponderador de banda de cuantificación, el ponderador (242) genera factores de ponderación para una matriz de cuantificación basada en la información recibida desde el modelador (240) de percepción, y aplica los factores de ponderación a los datos recibidos desde el transformador (230) de frecuencia. El ponderador (242) presenta a la salida información colateral tal como el conjunto de factores de ponderación para el MUX (290). Como ponderador de canal, el ponderador (242) genera a continuación factores de ponderación específicos del canal, en base a la información recibida desde el modelador (240) de percepción, y también en base a la calidad de la señal localmente reconstruida. Estas ponderaciones escalares permiten que los canales reconstruidos tengan una calidad aproximadamente uniforme. El ponderador (242) presenta a la salida bloques ponderados de datos de coeficiente para el transformador (250) multi-canal, y presenta a la salida información colateral tal como el conjunto de factores de ponderación de canal para el MUX (290). Alternativamente, el codificador (200) utiliza otra forma de ponderación o de ponderación no continua.

Para datos de audio multi-canal, los múltiples canales de datos de coeficientes de frecuencia conformados con ruido producidos por el ponderador (242), están muchas veces correlacionados. Para aprovechar esta correlación, el transformador (250) multi-canal puede aplicar una transformación multi-canal a los datos de audio. El transformador (250) multi-canal produce información colateral para el MUX (290) que indican, por ejemplo, las transformaciones multi-canal utilizadas y las partes de las tramas transformadas multi-canal.

El cuantificador (260) cuantifica la salida del transformador (250) multi-canal, produciendo datos de coeficiente cuantificados para el codificador (270) de entropía e información colateral que incluye tamaños de escalón de cuantificación para el MUX (290). La cuantificación introduce pérdidas de información irreversibles, pero también permite que el codificador (200) regule la calidad y la velocidad de bit de la corriente de bits (295) de salida junto con el controlador (280). En algunas realizaciones, el cuantificador (260) es un cuantificador escalar, uniforme, adaptativo. En realizaciones alternativas, el cuantificador es un cuantificador no uniforme, un cuantificador vectorial, y/o un cuantificador no adaptativo, o utiliza una forma diferente de cuantificación escalar, uniforme, adaptativa.

El codificador (270) de entropía comprime de manera sin pérdidas, datos de coeficiente cuantificados recibidos desde el cuantificador (260). En algunas realizaciones, el codificador (270) de entropía utiliza codificación entrópica adaptativa según se describe en la sección que sigue. El codificador (270) de entropía puede calcular el número de bits gastados en la codificación de la información de audio y pasar esta información al controlador (280) de velocidad/ calidad.

El controlador (280) trabaja con el cuantificador (260) para regular la velocidad de bit y/o la calidad de la salida del codificador (200). El controlador (280) recibe información desde otros módulos del codificador (200) y procesa la

información recibida para determinar condiciones actuales dadas de los factores de cuantificación deseados. El controlador (280) presenta a la salida los factores de cuantificación para el cuantificador (260) con el objetivo de satisfacer exigencias de calidad y/o de velocidad de bit.

El codificador (272) mezclado sin pérdidas/ puro sin pérdidas, y el codificador (274) de entropía asociado, comprimen datos de audio para el modo de codificación sin pérdidas mezclado/ puro. El codificador (200) utiliza el modo de codificación sin pérdidas mezclado/ puro para una secuencia completa, o conmuta entre modos de codificación sobre una base de trama-a-trama o sobre otra base.

El MUX (290) multiplexa la información colateral recibida desde los otros módulos del codificador (200) de audio junto con los datos codificados de entropía recibidos desde el codificador (270) de entropía. El MUX (290) presenta a la salida la información en un formato WMA o en otro formato que sea reconocido por un descodificador de audio. El MUX (290) incluye una memoria intermedia virtual que almacena la corriente de bits (295) que va a ser presentada a la salida por el codificador (200). La cantidad de llenado actual de la memoria intermedia, la velocidad de cambio de la cantidad de llenado de la memoria intermedia, y otras características de la memoria intermedia, pueden ser utilizadas por el controlador (280) para regular la calidad y/o la velocidad de bit para las diferentes aplicaciones (por ejemplo, a calidad constante/ velocidad de bit variable, o a baja velocidad de bit constante/ calidad variable).

B. Descodificador de Audio Generalizado

Con referencia a la Figura 3, el descodificador (300) de audio generalizado incluye un desmultiplexor (310) de corriente de bit ["DEMUX"], uno o más descodificadores (320) de entropía, un descodificador (322) sin pérdidas mezclado/ puro, un descodificador (330) de configuración, un transformador (340) multi-canal inverso, un cuantificador/ ponderador (350) inverso, un transformador (360) de frecuencia inverso, un solapador/ sumador (370), y un post-procesador (380) multi-canal. El descodificador (300) es a veces más simple que el codificador (300) debido a que el descodificador (300) no incluye módulos para control de velocidad/ calidad o para modelación de percepción. A continuación se realiza la descripción de algunos de los módulos del descodificador (300). Para una descripción acerca de otros módulos del descodificador (300) de alguna de las realizaciones, véanse las aplicaciones referenciadas en la sección Datos de Aplicación Relacionada.

El descodificador (300) recibe una corriente de bits (305) de información de audio comprimida en un formato WMA o en otro formato. La corriente de bits (305) incluye datos codificados de entropía, así como también información colateral a partir de la cual el descodificador (300) reconstruye las muestras (395) de audio.

El DEMUX (310) analiza la información de la corriente de bits (305) y envía información a los módulos del descodificador (300). El DEMUX (310) incluye una o más memorias intermedias para compensar las variaciones a corto plazo de la velocidad de bits debidas a las fluctuaciones en la complejidad del audio, a las fluctuaciones de la red, y/o a otros factores.

El, o los, descodificador(es) (320) de entropía descomprime(n) sin pérdidas códigos de entropía recibidos desde el DEMUX (310). Por motivos de simplicidad, un módulo descodificador de entropía ha sido representado en la Figura 3, aunque se pueden utilizar diferentes descodificadores de entropía para los modos de codificación con pérdidas y sin pérdidas, o incluso dentro de los modos. También, por motivos de simplicidad, la Figura 3 no muestra la lógica de selección de modo. El descodificador (320) de entropía aplica típicamente la inversa de la técnica de codificación de entropía utilizada en el codificador (200). Cuando se descodifican datos comprimidos en modo de codificación con pérdidas, el descodificador (320) de entropía produce datos de coeficiente de frecuencia cuantificados.

El descodificador (322) sin pérdidas mezclado/ puro y el (los) descodificador(es) (320) de entropía asociado(s), descomprimen datos de audio codificados sin pérdidas para el modo de codificación sin pérdidas mezclado/ puro. El descodificador (300) utiliza un modo de descodificación particular para una secuencia completa, o conmuta modos de descodificación sobre una base de trama-a-trama o sobre otra base.

El transformador (340) multi-canal inverso recibe los datos de coeficiente de frecuencia cuantificados descodificados de entropía desde el (los) descodificador(es) (320) de entropía, así como también información colateral desde el DEMUX (310) que indica, por ejemplo, la transformación multi-canal utilizada y las partes o tramas transformadas.

El cuantificador/ ponderador (350) inverso recibe factores de cuantificación, así como también matrices de cuantificación desde el DEMUX (310), y recibe datos de coeficiente de frecuencia cuantificados desde el transformador (340) multi-canal. El cuantificador/ ponderador (350) inverso descomprime el factor de cuantificación/ información de matriz que se ha recibido según sea necesario, y a continuación realiza la cuantificación y ponderación inversas.

El transformador (360) de frecuencia inverso recibe los datos de coeficiente de frecuencia presentados a la salida por el cuantificador/ponderador (350) inverso, así como también toda la información colateral procedente del DEMUX (310). El transformador (360) de frecuencia inverso, aplica la inversa de la transformación de frecuencia utilizada en el codificador, y presenta a la salida bloques para el solapador/ sumador (370).

El solapador/ sumador (370) recibe información descodificada desde el transformador (360) de frecuencia inverso y/o desde el descodificador (322) sin pérdidas mezclado/ puro. El solapador/ sumador (370) solapa y suma datos de audio

según sea necesario, e intercala tramas u otras secuencias de datos de audio codificados en diferentes modos.

III. Conmutación de Modo de Codificación/ Descodificación de Entropía Adaptativa

Los métodos de codificación de nivel de serie son frecuentemente más eficaces que la codificación directa de niveles cuando una secuencia de entrada contiene muchas ocurrencias de un valor simple (por ejemplo, 0). Sin embargo, puesto que los coeficientes de transformación cuantificados no-cero son comunes en secuencias de entrada de datos de audio, especialmente a las frecuencias más bajas, la codificación de nivel de serie no es eficaz a través de la gama de frecuencias completa. Además, en audio de calidad más alta, los coeficientes de transformación cuantificados no-cero resultan más comunes incluso a frecuencias más altas. (En audio de calidad más alta, los niveles de cuantificación son típicamente más pequeños). Por lo tanto, en algunas realizaciones, un codificador tal como el codificador (200) de la Figura 2, realiza una técnica de codificación multi-modo que puede hacer uso de codificación de nivel de serie para una porción de la secuencia de entrada de datos de audio, y codificación directa de niveles para otra porción de la secuencia. Un descodificador, tal como el descodificador (300) de la Figura 3, ejecuta una técnica de descodificación multi-modo correspondiente.

A. Conmutación de Modo de Codificación de Entropía Adaptativa

Con referencia a la Figura 4, en una técnica 400 de codificación multi-modo, el codificador realiza en primer lugar la codificación de niveles de señal en una corriente de entrada (410) directamente. Por ejemplo, el codificador realiza codificación de Huffman de dimensión variable, codificación aritmética basada en contexto, u otra técnica de codificación entrópica directamente sobre los niveles de señal.

En un punto de conmutación durante la codificación, el codificador cambia el esquema (420) de codificación. El codificador puede cambiar el esquema de codificación en un punto de conmutación predeterminado, o el codificador puede analizar los datos de entrada para determinar un punto apropiado para cambiar los esquemas de codificación. Por ejemplo, el codificador puede analizar una secuencia de entrada para hallar el mejor punto en el que conmutar a codificación de nivel de serie, enviando el punto de conmutación al descodificador en la corriente de bits de salida. O, el codificador puede calcular el punto de conmutación adaptativamente contando ceros consecutivos (o alternativamente, otro valor predominante) en los datos de entrada, y conmutar a codificación de nivel de serie cuando se ha contado un número de umbral particular de ceros consecutivos. El descodificador puede calcular el punto de conmutación de la misma manera, de modo que el punto de conmutación no necesita ser incluido en la corriente de bits. O, el codificador y el descodificador utilizan algún otro criterio para determinar el punto de conmutación.

Tras el punto de conmutación, el codificador codifica los niveles de señal restantes utilizando codificación (430) de nivel de serie. Por ejemplo, el codificador realiza codificación de Huffman, codificación aritmética basada en contexto, u otra técnica de codificación de entropía sobre longitudes de serie y niveles de señal. El codificador puede usar la misma técnica (por ejemplo, codificación aritmética basada en contexto) antes y después del punto de conmutación, o el codificador puede usar técnicas diferentes.

Además, aunque la Figura 4 y otras varias Figuras de la aplicación muestran un sólo punto de conmutación, se pueden utilizar puntos de conmutación adicionales para dividir los datos de entrada en más de dos porciones. Por ejemplo, se pueden establecer puntos adicionales de conmutación adaptativa para umbrales incrementados de ceros consecutivos. A continuación se pueden aplicar diferentes esquemas de codificación a las diferentes porciones. O, el codificador puede experimentar con diferentes puntos de segmentación en la secuencia, ponderando las eficiencias de codificación para las diferentes configuraciones de segmentación junto con los costes de señalizar las diferentes configuraciones para el descodificador.

La Figura 5 muestra una técnica (500) de codificación multi-modo con cálculo de punto de conmutación adaptativa de acuerdo con una implementación. El punto de conmutación adaptativa depende de un conteo de coeficientes de valor cero consecutivos. Los datos de entrada son niveles de señal para coeficientes de transformación cuantificados, que se incrementan a partir del coeficiente de frecuencia más baja hasta el coeficiente de frecuencia más alta. En la práctica, la posición del punto de conmutación depende de la señal que se está comprimiendo y de la velocidad de bits/ calidad de la codificación. Alternativamente, los datos de entrada son otra forma y/u otra organización de datos de audio.

Para empezar, el codificador inicializa varias variables. Específicamente, el codificador establece una variable de conteo de serie en 0 (510) y establece una variable de estado de configuración en “directa” (512).

El codificador recibe el siguiente coeficiente QC como entrada (520). El codificador comprueba (530) a continuación si el coeficiente QC es cero. Si el coeficiente QC no es cero, el codificador reinicia el conteo de serie (538). En otro caso (es decir, si el coeficiente QC es cero), el codificador incrementa la variable de conteo de serie (532), y comprueba si el conteo actual de serie supera el umbral de conteo de serie (534). El umbral de conteo de serie puede ser estático o puede depender de un factor tal como el tamaño de un bloque de coeficientes (por ejemplo, un umbral de conteo de serie de 4 para una secuencia de 256 coeficientes, 8 para una secuencia de 512 coeficientes, etc.), o puede ser adaptativo de alguna otra manera. Si el conteo de serie excede el umbral, el codificador cambia el estado de codificación a codificación de nivel de serie [“RLE”] (536).

El codificador codifica a continuación el coeficiente QC si es apropiado (540). (En algunos casos, se codifican

conjuntamente grupos de coeficientes utilizando una técnica tal como codificación vectorial de Huffman. En esos casos, el codificador puede retrasar la codificación del coeficiente QC).

El codificador comprueba a continuación (550) si el codificador debe conmutar los modos de codificación. En particular, el codificador comprueba el estado de codificación. Si el estado de codificación ya no es directo (por ejemplo, si el codificador ha cambiado el estado de codificación a RLE como resultado de haber alcanzado un número de umbral de coeficientes cero), el codificador empieza la codificación de nivel de serie de los coeficientes (560). (De nuevo, en casos en los que se codifican conjuntamente grupos de coeficientes, el codificador puede retrasar la decisión de conmutación hasta alcanzar un punto de interrupción conveniente para un grupo de coeficientes).

Si el codificador no conmuta los modos de codificación, el codificador comprueba si ha acabado la codificación de los coeficientes (570). Si es así, el codificador sale. En otro caso, el codificador introduce el siguiente coeficiente (520) para que continúe el proceso de codificación.

B. Conmutación de Modo de Descodificación de Entropía Adaptativa

Con referencia a la Figura 6, en una técnica (600) de descodificación multi-modo, el descodificador descodifica directamente niveles (610) de señal codificados. Por ejemplo, el descodificador ejecuta descodificación de Huffman de dimensión variable, descodificación aritmética basada en contexto, u otra técnica de descodificación de entropía sobre niveles de señal codificados directamente.

En un punto de conmutación durante la descodificación, el descodificador cambia el esquema de descodificación (620). Si el punto de conmutación está predeterminado, el descodificador puede recibir, en forma de banderola o de otro mecanismo de notificación, datos que indican explícitamente al descodificador cuándo cambiar los esquemas de descodificación. O, el descodificador puede calcular adaptativamente cuándo cambiar los esquemas de descodificación en base a los datos de entrada que reciba. Si el descodificador calcula el punto de conmutación, el descodificador utiliza la misma técnica de cálculo utilizada por el codificador para asegurar que el esquema de descodificación cambia en un punto correcto. Por ejemplo, el descodificador cuenta ceros consecutivos (o alternativamente, otro valor predominante) para determinar el punto de conmutación adaptativamente. En una implementación, el descodificador utiliza una técnica correspondiente a la técnica de codificación mostrada en la Figura 5. O, el descodificador utiliza algún otro criterio para determinar el punto de conmutación.

Después del punto de conmutación, el descodificador descodifica los restantes niveles de señal codificados en nivel de serie (630). Por ejemplo, el decodificador realiza descodificación de Huffman, descodificación aritmética basada en contexto, u otra técnica de descodificación de entropía sobre longitudes de serie y niveles de señal codificados. El descodificador puede usar la misma técnica (por ejemplo, descodificación aritmética basada en contexto), antes y después del punto de conmutación, o el descodificador puede usar técnicas diferentes.

IV. Codificación y Descodificación de Huffman de Dimensión Variable

Mientras que símbolos tales como niveles de señal directa, pueden ser codificados utilizando codificación escalar de Huffman, esa alternativa está limitada cuando el número óptimo de bits para codificar un símbolo es un número fraccionario. La codificación escalar de Huffman está también limitada por la incapacidad de los códigos escalares de Huffman para responder de la correlación estadística entre símbolos. La codificación vectorial de Huffman produce una mejor reducción de velocidad de bit que la codificación escalar de Huffman (por ejemplo, permitiendo que el codificador aproveche probabilidades fraccionalmente en códigos binarios de Huffman). Y, en general, los vectores de dimensión más alta producen mejor reducción de velocidad de bit que los vectores de dimensión más pequeña. Sin embargo, si se asigna un código a cada combinación de símbolo posible, el tamaño del código de cifrado y descifrado se incrementa exponencialmente según se incrementa la dimensión del vector. Por ejemplo, en un sistema de 32 bits, el número de combinaciones posibles para un vector de 4 dimensiones es de $(2^{32})^4$. El tiempo de búsqueda para emparejar un vector y hallar un código de Huffman, también se incrementa drásticamente según se incrementa el tamaño del código de cifrado y descifrado.

En algunas realizaciones, para reducir el tamaño del código de cifrado y descifrado, un codificador tal como el codificador (200) de la Figura 2, utiliza una técnica de codificación vectorial de Huffman de dimensión variable. En vez de asignar un código de cifrado y descifrado a cada combinación n-dimensional posible, se asignan códigos a un número limitado de vectores n-dimensionales más probables. Si no se asigna un código a un vector n-dimensional particular, el vector n-dimensional es en ese caso codificado en forma de vectores de dimensión más pequeña (por ejemplo, dos vectores de dimensión n/2), como escalares con códigos de Huffman, o como escalares utilizando la técnica de sin-tabla para representar valores discretos. Un descodificador tal como el descodificador (300) de la Figura 3, reconstruye un vector hallando el (los) código(s) para el vector y hallando los valores asociados.

Por ejemplo, en el caso de vectores de 4 dimensiones con 256 valores posibles por símbolo, el codificador codifica los 500 vectores de 4 dimensiones más probables con códigos de Huffman, y utiliza un código de escape para indicar otros vectores. El codificador codifica los 500 vectores de 2 dimensiones más probables con códigos de Huffman y utiliza un código de escape para indicar otros vectores, que son divididos y codificados con códigos escalares de Huffman. De ese modo, el codificador utiliza 501 + 501 + 256 códigos.

En términos de determinar qué vectores o escalares se representan con códigos de Huffman en una tabla, y en términos de asignar los códigos de Huffman propios para la tabla, la construcción de código de cifrado y descifrado puede ser estática, adaptativa para los datos previamente codificados, o adaptativa para los datos que van a ser codificados.

A. Codificación Vectorial de Huffman de Dimensión Variable

Con referencia a la Figura 7, un codificador utiliza una técnica (700) de codificación vectorial de Huffman de dimensión variable ["VDVH"]. Por ejemplo, el codificador utiliza la técnica (700) para codificar directamente niveles de señal para coeficientes de frecuencia de datos de audio. Alternativamente, el codificador utiliza la técnica (700) para codificar otra forma de datos de audio. Por motivos de simplicidad, la Figura 7 no muestra la construcción del código de cifrado y descifrado. La construcción del código de cifrado y descifrado puede ser estática, adaptativa para los datos previamente codificados, o adaptativa para los datos que van a ser codificados.

El codificador obtiene (710) el siguiente vector de n símbolos. Por ejemplo, el codificador obtiene los siguientes 4 símbolos en serie.

El codificador comprueba (720) si el código de cifrado y descifrado incluye un código para el vector. Si es así, el codificador utiliza (730) un código de Huffman simple para codificar el vector. Por ejemplo, para determinar cómo codificar un vector de n dimensiones, el codificador comprueba una tabla de códigos de vector n-dimensional para un código asociado al vector. Puesto que los vectores de mayor dimensión producen normalmente ahorros de velocidad de bits, el codificador utiliza códigos de Huffman para los vectores n-dimensionales más probables. Pero, para limitar el tamaño de la tabla, solamente algunos de los vectores n-dimensionales tienen códigos asociados.

Si el código de cifrado y descifrado no incluye un código para el vector, el codificador divide (740) el vector en vectores y/o escalares más pequeños, y codifica los vectores y/o escalares más pequeños. Por ejemplo, el codificador divide un vector de n símbolos en x vectores de n/x símbolos. Para cada vector de n/x símbolos, el codificador repite recursivamente la técnica de codificación, saliendo cuando el vector de n/x símbolos o sus vectores/ escalares componentes han sido codificados con códigos de Huffman o (para los escalares) utilizando una técnica de sin-tabla para representar valores discretos.

El codificador comprueba (750) a continuación si existen algunos vectores adicionales para codificar. Si no es así, el codificador sale. En otro caso, el codificador obtiene (710) el siguiente vector de n símbolos.

I. Ejemplo de Implementación

La Figura 8 muestra una técnica (800) detallada para codificar vectores utilizando codificación VDVH en una implementación. En la técnica (800), el codificador suma los valores enteros de los símbolos en un vector de símbolos para determinar si debe codificar el vector utilizando un código simple de Huffman, o dividir el vector en vectores/ escalares más pequeños. Esto limita efectivamente el tamaño de código de cifrado y descifrado, y agiliza la búsqueda de códigos.

Una tabla de código de cifrado y descifrado para vectores n-dimensionales ["n-dim"] incluye un código de escape. Los L_1 códigos son para cada vector para el que la suma de los componentes del vector (que son números enteros) está por debajo de un umbral T_1 particular. Por ejemplo, supóngase que n es 4 y el umbral T_1 para los vectores 4-dim es 6. La tabla de código de cifrado y descifrado para los vectores 4-dim incluye el código de escape y 126 códigos, uno para cada vector posible cuyos componentes (por ejemplo, los valores absolutos de los componentes) sumen menos de 6 – (0, 0, 0, 0), (0, 0, 0, 1), etc. Limitar el tamaño de la tabla en base a la suma de componentes de los vectores, resulta efectivo debido, en general, a que los vectores más probables son aquellos cuyas sumas de componentes son más pequeñas.

Si la tabla de código de cifrado y descifrado para los vectores n-dim no tiene un código de Huffman para un vector n-dim particular, el codificador añade un código de escape a la corriente de bits de salida y codifica el vector n-dim como vectores o escalares de dimensión más pequeña, buscando esos vectores o escalares de dimensión más pequeña en otras tablas de código de cifrado y descifrado. Por ejemplo, la dimensión más pequeña es n/2 a menos que n/2 sea 1, en cuyo caso el vector n-dim se divide en escalares. Alternativamente, el vector n-dim se divide de alguna otra manera.

La tabla de código de cifrado y descifrado para vectores de dimensión más pequeña incluye códigos de Huffman para L_2 vectores de dimensión más pequeña, así como también un código de escape. Los L_2 códigos son para cada vector para el que la suma de componentes de vector esté por debajo de un umbral T_2 particular para la tabla de dimensión más pequeña. Por ejemplo, supóngase que la dimensión más pequeña es 2 y que el umbral T_2 para vectores 2-dim es 16. La tabla de código de cifrado y descifrado para vectores 2-dim incluye el código de escape y 136 códigos, uno para cada vector posible cuyos componentes (por ejemplo, los valores absolutos de los componentes) sumen menos de 16 – (0, 0), (0, 1), etc.

Si la tabla de código de cifrado y descifrado para vectores de dimensión más pequeña no tiene un código de Huffman para un vector particular de dimensión más pequeña, el codificador añade un código de escape a la corriente de bits de salida y codifica el vector como vectores o escales incluso más pequeños, utilizando otras tablas de código de cifrado y descifrado. Este proceso se repite decreciendo a nivel escalar. Por ejemplo, la división es mediante una potencia de 2, descendiendo al nivel escalar. Alternativamente, el vector se divide de alguna otra manera.

A nivel escalar, la tabla de código de cifrado y descifrado incluye códigos de Huffman para L_3 escalares así como también un código de escape. Los L_3 códigos son para cada escalar que esté por debajo de un umbral T_3 (lo que supone que los valores pequeños son los más probables). Por ejemplo, supóngase que el umbral T_3 para escalares es 100. La tabla de código de cifrado y descifrado para escalares incluye 100 códigos y un código de escape. Si un escalar no tiene un código asociado en la tabla de código escalar, el escalar es codificado con el código de escape, y con un valor (por ejemplo, literal) de acuerdo con una técnica de sin-tabla. Utilizando todos los ejemplos numéricos dados en esta sección, las tablas tendrían que incluir un total de 126 + 1 + 136 + 1 + 100 + 1 = 365 códigos.

Los tamaños dimensionales para las tablas, los factores de división de vector, y los umbrales para las sumas de los componentes de vector, dependen de la implementación. Otras implementaciones utilizan diferentes tamaños de vector, diferentes factores de división, y/o diferentes umbrales. Alternativamente, un codificador utiliza criterios distintos de las sumas de componentes de vector para conmutar tamaños de vector/ tablas de código de cifrado y descifrado, en codificación VDVH.

Con referencia a la Figura 8, el codificador obtiene en primer lugar un vector (810) n-dim. El vector n-dim comprende n símbolos, teniendo cada símbolo, por ejemplo, un valor que representa el nivel cuantificado para un coeficiente de frecuencia de datos de audio.

El codificador suma los componentes de vector (812) y compara la suma con un umbral (820) para vectores n-dim. Si la suma es menor, o igual, que el umbral, el codificador codifica el vector n-dim con un código de Huffman procedente de una tabla (822) de código, y continúa hasta que la codificación está completa (824). Si la suma es mayor, o igual, que el umbral, el codificador envía un código de escape (826) y divide el vector n-dim en dos vectores más pequeños con dimensiones de n/2 (830).

El codificador obtiene el siguiente vector n/2-dim (840) y suma los componentes del vector n/2-dim (842). El codificador comprueba la suma respecto a un umbral asociado a vectores n/2-dim (850). Si la suma es menor o igual que el umbral, el codificador codifica el vector n/2-dim con un código de Huffman procedente de una tabla de código (852) para vectores n/2-dim, y obtiene el siguiente vector n/2-dim (840) si el codificador no ha acabado de codificar los vectores n/2-dim (854). Si la suma es mayor que el umbral para vectores n/2-dim, el codificador envía otro código (856) de escape.

El codificador sigue en general este patrón en el procesamiento de los vectores, tanto para codificar cada vector como para dividir el vector en vectores de dimensión más pequeña. En casos en los que el codificador divide un vector en dos componentes (860) escalares (de 1 dimensión), el codificador obtiene el siguiente escalar (870) y compara el valor del escalar con un umbral asociado a valores escalares (880). Si el valor escalar es menor, o igual, que el umbral (880), el codificador codifica el escalar utilizando un código de Huffman procedente de una tabla de código (882) para escalares. Si el valor escalar es mayor que el umbral, el codificador codifica el escalar utilizando una técnica de sin-tabla (884). El codificador obtiene a continuación el siguiente escalar (870) si no ha acabado de procesar los escalares (886).

Alternativamente, el codificador utiliza tablas con diferentes tamaños dimensionales, divide vectores de alguna manera distinta a una potencia de 2, y/o utiliza criterios distintos a la suma de componentes de vector para conmutar tamaños de vector/ tablas de código de cifrado y descifrado, en codificación VDVH.

2. Conmutación Adaptativa

La Figura 9 muestra una técnica (900) para codificación VDVH directa de coeficientes de niveles de señal, donde el codificador determina adaptativamente un punto de conmutación para cambiar a codificación de longitudes de serie y de niveles de señal, de acuerdo con una implementación. El punto de conmutación adaptativa depende de un conteo de coeficientes consecutivos de valor cero. Los datos de entrada son niveles de señal para coeficientes de transformación cuantificados, que progresan desde el coeficiente de frecuencia más baja hasta el coeficiente de frecuencia más alta. Alternativamente, los datos de entrada son de otra forma y/u organización que los datos de audio.

Para empezar, el codificador inicializa varias variables. Específicamente, el codificador establece una variable de conteo de serie en 0 (910), establece una variable de vector actual en vacío (912), y establece una variable de estado de codificación en vector Huffman directo de dimensión variable [“DVDVH”] (914).

El codificador recibe el siguiente coeficiente QC como entrada (920). El codificador comprueba a continuación (930) si el coeficiente es cero. Si el coeficiente QC no es cero, el codificador reinicia el conteo de serie (938) y suma el coeficiente QC del vector actual (940). En otro caso (es decir, si el coeficiente QC es cero), el codificador incrementa la variable de conteo de serie (932), y comprueba si el conteo de serie actual supera el umbral de conteo de serie (934). El umbral de conteo de serie puede ser estático o puede depender de un factor tal como el tamaño de un bloque de coeficientes (por ejemplo, cuatro ceros en una secuencia de entrada de 256 coeficientes), o puede ser adaptativo de alguna otra forma. Por ejemplo, el umbral puede ser incrementado o rebajado, con o sin relación con el número de coeficientes de una secuencia de entrada. Si el conteo de serie excede el umbral, el codificador cambia el estado de codificación a codificación de nivel de serie [“RLE”] (936), y el coeficiente QC es añadido como componente al vector actual (940).

Añadir el coeficiente QC al vector actual incrementa la dimensión del vector. El codificador determina (950) si el vector actual está listo para codificar, comparando el número de componentes del vector actual con la máxima dimensión para el vector actual. Si es así, el codificador codifica el vector actual utilizando codificación DVDVH (960). Si el vector actual

es más pequeño que la dimensión máxima, pero el coeficiente QC es el último de una secuencia, el codificador puede rellenar el vector actual y codificarlo utilizando codificación DVDVH (960). La dimensión máxima depende de la implementación. En una implementación, es 8. Sin embargo, la dimensión máxima puede ser incrementada o disminuida dependiendo, por ejemplo, de la cantidad de recursos disponibles para crear, almacenar o transmitir un código de cifrado y descifrado.

Tras la codificación del vector, el codificador comprueba el estado de codificación (970). Si el estado de codificación ya no es DVDVH (por ejemplo, si el codificador ha cambiado el estado de codificación a RLE como resultado de exceder el número de umbral de coeficientes cero), el codificador empieza la codificación de los coeficientes como longitudes y niveles de serie (980). La codificación de nivel de serie puede ser llevada a cabo de varias formas, incluyendo, por ejemplo, codificación de Huffman, codificación vectorial de Huffman, o codificación aritmética basada en contexto. En algunas realizaciones, la codificación de nivel de serie se lleva a cabo utilizando codificación de Huffman con dos tablas de códigos de Huffman, donde una tabla se utiliza para codificar datos en los que las secuencias más cortas son las más probables, y una tabla se utiliza para codificar datos en los que las secuencias más largas son las más probables. El codificador trata cada tabla y elige códigos de una de las tablas, indicando con un bit de señal cuál de las tablas es la que ha utilizado el codificador.

Si el estado de codificación no ha cambiado o el vector actual no está listo para codificar, el codificador determina (990) si existen más coeficientes que deban ser codificados. Si es así, el codificador introduce el siguiente coeficiente (920) y continúa el proceso de codificación.

B. Descodificación Vectorial de Huffman de Dimensión Variable

La Figura 10 muestra una técnica (1000) de descodificación VDVH correspondiente con la técnica (700) de codificación VDVH mostrada en la Figura 7. Por ejemplo, un descodificador utiliza la técnica (1000) para descodificar niveles de señal codificados directamente para coeficientes de frecuencia de datos de audio. Alternativamente, el descodificador utiliza la técnica para descodificar otra forma de datos de audio.

El descodificador obtiene (1010) el siguiente código de Huffman para una tabla de codificación de Huffman de vector n-dimensional. Por ejemplo, el descodificador obtiene el siguiente código de Huffman para 4 símbolos en secuencia.

El descodificador comprueba (1020) si el código de Huffman es el código de escape para la tabla de codificación vectorial de Huffman n-dimensional. Si no lo es, el descodificador obtiene (1030) los n símbolos representados por el código de Huffman. Por ejemplo, el descodificador obtiene los 4 símbolos asociados al código de Huffman en un código de cifrado y descifrado de Huffman de vector de 4 dimensiones.

Si el código es el código de escape, el código de cifrado y descifrado n-dimensional no incluye un código para el vector, y el descodificador obtiene (1040) códigos de Huffman para vectores y/o escalares más pequeños. Por ejemplo, el descodificador obtiene códigos para x vectores de n/x símbolos. Para cada vector de n/x símbolos, el descodificador repite recursivamente la técnica de descodificación, saliendo cuando el vector o sus vectores/ escalares componentes están codificados.

El descodificador comprueba a continuación (1050) si existen códigos adicionales para la tabla de codificación vectorial de Huffman n-dimensional que deban ser descodificados. Si no los hay, el descodificador sale. En otro caso, el descodificador obtiene (1010) el siguiente de tales códigos de Huffman.

I. Ejemplo de Implementación

La Figura 11 muestra una técnica (1100) detallada para descodificar vectores codificados utilizando codificación VDVH en una implementación. La técnica de descodificación (1100) corresponde a la técnica (800) de codificación mostrada en la Figura 8.

Con referencia a la Figura 11, el descodificador obtiene el siguiente código para una tabla de código de Huffman de vector n-dim (1110). El descodificador comprueba si el código es el código de escape para la tabla de código de Huffman de vector n-dim (1120). Si no lo es, el descodificador obtiene los n símbolos representados por el código en la tabla de vector n-dim (1122). El descodificador continúa hasta que el descodificador ha acabado de procesar los datos codificados (1124).

Si el código es el código de escape para la tabla de código de Huffman de vector n-dim, el descodificador descodifica el vector n-dim como dos vectores n/2-dim utilizando una tabla de código de Huffman de vector n/2–dim. Específicamente, el descodificador obtiene el siguiente código para la tabla de código de Huffman de vector n/2-dim (1130). El descodificador comprueba si el código es el código de escape para la tabla de código de Huffman de vector n/2-dim (1140). Si no lo es, el descodificador obtiene los n/2 símbolos representados por el código en la tabla de código de Huffman de vector n/2-dim (1142). El descodificador continúa procesando los códigos para la tabla de código de Huffman de vector n/2-dim hasta que el procesamiento de tales códigos se ha completado (1144).

Si el código es el código de escape para la tabla de código de Huffman de vector n/2-dim, el descodificador descodifica el vector n/2-dim como dos vectores n/4-dim, que pueden ser escalares, etc.

El descodificador sigue generalmente este patrón de descodificación de vectores de mayor dimensión como dos vectores de dimensiones más pequeñas cuando se detectan códigos de escape, hasta que los vectores que van ser descodificados son escalares (vectores 1-dim). En ese punto, el descodificador obtiene el siguiente código para una tabla escalar de Huffman (1150). El descodificador comprueba si el código es el código de escape para la tabla de código escalar de Huffman (1160). Si no lo es, el descodificador obtiene el escalar representado por el código en la tabla de código escalar de Huffman (1162). El descodificador sigue procesando los códigos para los escalares hasta que el procesamiento de tales códigos se ha completado (1164). Si el código es el código de escape para la tabla de código escalar de Huffman, el escalar se codifica utilizando una técnica de sin-tabla, y el descodificador obtiene el valor (1170).

Alternativamente, el descodificador utiliza tablas con diferentes tamaños dimensionales y/o utiliza tablas que dividen los vectores de alguna forma distinta a la de potencia de 2 en descodificación VDVH.

2. Conmutación Adaptativa

La Figura 12 muestra una técnica (1200) para descodificar vectores que han sido codificados utilizando codificación VDVH de acuerdo con una implementación, en la que el descodificador determina adaptativamente un punto de conmutación para cambiar a descodificación de longitudes de serie y de niveles de señal. El punto de conmutación adaptativa depende de un conteo de coeficientes de valor cero consecutivos en los datos, que son niveles de señal para coeficientes de transformación cuantificados, que progresan desde el coeficiente de frecuencia más baja hasta el coeficiente de frecuencia más alta. Alternativamente, los datos son otra forma y/u organización de datos de audio.

Para empezar, el descodificador inicializa varias variables. Específicamente, el descodificador establece un conteo de serie en 0 (1210) y establece un estado de descodificación en DVDVH (1212).

El descodificador descodifica el siguiente vector buscando el código para ese vector en una tabla (1220) de codificación de Huffman. Por ejemplo, el descodificador lleva a cabo la técnica (1100) de descodificación mostrada en la Figura 11. El descodificador actualiza a continuación el conteo de serie en base al vector descodificado (1230) (específicamente, utilizando el número de valores cero en el vector descodificado para reiniciar, incrementar, o ajustar de otro modo el conteo de serie).

El descodificador comprueba si el conteo de serie excede un umbral (1240). El umbral de conteo de serie puede ser estático o puede depender de un factor tal como el tamaño de un bloque de coeficientes (por ejemplo, cuatro ceros en una secuencia de entrada de 256 coeficientes), o puede ser adaptativo de alguna otra manera. Si el conteo de serie excede el umbral, el descodificador empieza a descodificar los coeficientes codificados utilizando descodificación de nivel de serie (1250). La descodificación de nivel de serie puede ser llevada a cabo de varias formas, incluyendo, por ejemplo, descodificación de Huffman, descodificación vectorial de Huffman, o descodificación aritmética basada en contexto.

En algunas realizaciones, la descodificación de nivel de serie se lleva a cabo utilizando descodificación de Huffman con dos tablas potenciales de códigos de Huffman, donde una tabla se utiliza para descodificar datos en los que son más probables secuencias más cortas, y una tabla se utiliza para descodificar datos en los que son más probables secuencias más largas. Cuando el descodificador recibe un código, un bit de señal presente en el código indica qué tabla ha usado el codificador, y el descodificador busca el código en la tabla apropiada.

Si el conteo de serie no excede del umbral, el descodificador sigue procesando vectores hasta que la descodificación ha terminado (1260).

V. Codificación y Descodificación Aritméticas Basadas en Contexto

En algunas realizaciones que no forman parte de la invención, un codificador tal como el codificador (200) de la Figura 2 utiliza codificación aritmética basada en contexto [“CBA”] para codificar secuencias de datos de audio. En codificación CBA, se asocian diferentes distribuciones de probabilidad respecto a los símbolos de entrada, con diferentes contextos. La distribución de probabilidad utilizada para codificar la secuencia de entrada cambia cuando cambia el contexto. El contexto puede ser calculado midiendo los diferentes factores que se espera que afecten a la probabilidad de que aparezca un símbolo de entrada particular en una secuencia de entrada. Un descodificador tal como el descodificador (300) de la Figura 3, realiza la correspondiente descodificación aritmética.

Cuando se codifican coeficientes directamente (es decir, como niveles directos), el codificador utiliza factores que incluyen los valores de los coeficientes previos de la secuencia para calcular el contexto. Cuando se codifican coeficientes utilizando codificación de nivel de serie, el codificador utiliza factores que incluyen las longitudes de la secuencia actual y de las secuencias previas, además de los valores de los coeficientes previos, para calcular el contexto. El codificador utiliza una distribución de probabilidad asociada al contexto calculado para determinar el código aritmético apropiado para los datos. Así, utilizando los diversos factores en el cálculo de los contextos, el codificador determina los contextos adaptativamente con respecto a los datos y con respecto al modo (es decir, directo, nivel de serie) de representación de los datos.

En realizaciones alternativas, el codificador puede utilizar factores adicionales, puede omitir algunos factores, o puede usar los factores mencionados anteriormente en otras combinaciones.

A. Ejemplo de Implementación de Contextos

Las Tablas 2-5 y las Figuras 13A – 13D, 14A – 14H, y 15A – 15H, muestran contextos y distribuciones de probabilidad, respectivamente, utilizados en codificación y descodificación CBA, según un ejemplo de implementación. Alternativamente, la codificación y la descodificación CBA utilizan diferentes contextos y/o diferentes distribuciones de probabilidad.

Aunque la discusión que sigue está enfocada al cálculo de contexto en el codificador según el ejemplo de implementación, el descodificador realiza el correspondiente cálculo de contexto durante la descodificación utilizando los datos de audio previamente codificados.

Según se ha indicado anteriormente, el codificador puede codificar coeficientes utilizando codificación CBA si el codificador está codificando niveles directos solamente o longitudes de serie y niveles directos. En una implementación, sin embargo, las técnicas para calcular contextos varían dependiendo de si el codificador está codificando niveles directos solamente o longitudes de serie y niveles directos. Adicionalmente, cuando se codifican longitudes de serie y niveles directos, el codificador utiliza diferentes contextos dependiendo de si el codificador está codificando una longitud de serie o un nivel directo.

El codificador utiliza un cuarto sistema de contexto para calcular contextos durante la codificación aritmética de niveles directos utilizando contexto causal. El codificador calcula el contexto para un nivel L[n] actual en base al valor del nivel (L[n-1]) previo y al nivel justamente anterior al nivel previo (L[n-2]). Este cálculo de contexto se basa en la suposición de que: 1) si los niveles previos son bajos, el nivel actual es probable que sea bajo, y 2) los dos niveles medios es probable que sean mejores pronosticadores del nivel actual que otros niveles. La Tabla 2 muestra los contextos asociados a los valores de los dos niveles previos en el cuarto sistema de contexto. Las Figuras 13A – 13D muestran la distribución de probabilidad para niveles actuales para estos contextos.

Tabla 2: Contextos para Codificación/ Descodificación CBA de niveles directos

L[n-1]	**L[n-2]**	**Contexto**
= 0	= 0	0
= 0	$\geq$ 1	1
= 1	Cualquiera	2
$\geq$ 2	Cualquiera	3

La distribución de probabilidad de las Figuras 13A – 13D supone que cuando los dos niveles previos son cero o casi cero, es más probable que el nivel actual sea cero o casi cero.

El codificador también puede utilizar codificación CBA cuando realiza codificación de longitud de serie de niveles. Cuando se codifica una longitud de serie, los factores utilizados por el codificador para calcular el contexto incluyen el porcentaje de ceros en la secuencia de entrada (ejecución total sobre parte de, o sobre toda, la secuencia) y la longitud de la serie previa de ceros (R[n-1]). El codificador calcula un índice de porcentaje de ceros en base al porcentaje de ceros de la secuencia de entrada, como se muestra a continuación en la Tabla 3:

Tabla 3: Índices de porcentaje de ceros para codificación/ descodificación CBA de longitudes de serie

% de Ceros	**Índice de % de Ceros**
$\geq$ 90	0
$\geq$ 80	1
$\geq$ 60	2
< 60	3

El codificador utiliza el índice de porcentaje de ceros junto con la longitud de la serie previa para calcular el contexto para codificar la longitud de serie actual, como se muestra a continuación en la Tabla 4. Las Figuras 14A – 14H muestran distribuciones de probabilidad para diferentes valores de longitud de serie asociados a estos contextos.

Tabla 4: Contextos para codificación/ descodificación CBA de longitudes de serie

Índice de % de Ceros	R[n-1]	Contexto
0	= 0	0
0	> 0	4
1	= 0	1
1	> 0	5
2	= 0	2
2	> 0	6
3	= 0	3
3	> 0	7

Por ejemplo, en una secuencia de entrada en la que el 91% de los niveles son cero (lo que da como resultado un índice de porcentaje de ceros de 0), y en la que la longitud de la serie previa de ceros era 15, el contexto es 14. Las distribuciones de probabilidad en las Figuras 14A – 14H muestran que cuando el porcentaje de ceros en una secuencia de entrada es más alto, son más probables las longitudes de serie más largas. Las distribuciones de probabilidad suponen también que dentro de un índice de porcentaje de ceros dado, las longitudes de serie que siguen a una longitud de serie cero, es más probable que sean más cortas que las longitudes de serie que siguen a una longitud de serie mayor de cero.

Cuando se codifica un nivel en datos de nivel de serie, los factores utilizados por el codificador para calcular el contexto incluyen la longitud de la serie actual (R[n]), la longitud de la serie previa (R[n-1]), y los valores de los dos niveles previos (L[n-1]) y (L[n-2]). Este cálculo de contexto está basado en la observación de que el nivel actual depende de los dos niveles previos siempre que la separación (es decir, las longitudes de serie) entre los niveles no sea demasiado grande. También, si los niveles previos son más bajos, y si las secuencias previas son más cortas, es probable que el nivel actual sea bajo. Cuando las secuencias previas son más largas, el nivel previo tiene menos efecto sobre el nivel actual.

El contexto asociado a los valores de la longitud de serie actual, a la longitud de serie previa, y a los dos niveles previos, se muestra a continuación en la Tabla 5. Las Figuras 15A – 15H muestran distribuciones de probabilidad para niveles asociados a estos contextos.

Tabla 5: Contextos para codificación / descodificación CBA de niveles en codificación de nivel de serie

R[n]	R[n-1]	L[n-1]	L[n-2]	Contexto
$\geq$ 2	Cualquiera	Cualquiera	Cualquiera	0
< 2	$\geq$ 2	= 1	Cualquiera	1
< 2	$\geq$ 2	= 2	Cualquiera	2
< 2	$\geq$ 2	> 2	Cualquiera	3
< 2	< 2	= 1	= 1	4
< 2	< 2	= 1	> 1	5
< 2	< 2	= 1	Cualquiera	6
< 2	< 2	> 2	Cualquiera	7

Por ejemplo, en una secuencia de entrada en la que la longitud de la serie actual de ceros es 1, la longitud de la serie de ceros previa es 2, y el nivel previo es 1, el contexto es 1. Las distribuciones de probabilidad de las Figuras 15A – 15H muestran que cuando los niveles previos son más bajos, y cuando las longitudes de serie actual y previa son más cortas, es más probable que el nivel actual sea cero o casi cero.

B. Conmutación Adaptativa

La Figura 16 muestra una técnica (1600) para la codificación CBA directa de coeficientes de niveles de señal, en la que el codificador determina adaptativamente un punto de conmutación para cambiar a codificación de longitudes de serie y de niveles de señal de acuerdo con una implementación. El punto de conmutación adaptativa depende del conteo de coeficientes consecutivos de valor cero. Los datos de entrada son niveles de señal para coeficientes de transformación cuantificados, que progresan desde el coeficiente de frecuencia más baja hasta el coeficiente de frecuencia más alta. Alternativamente, los datos de entrada constituyen otra forma y/u organización de datos de audio

Para empezar, el codificador inicializa diversas variables. Específicamente, el codificador establece una variable de conteo de serie en 0 (1610) y establece una variable de estado de codificación en aritmética directa basada en contexto (DCBA) (1612).

El codificador recibe el siguiente coeficiente QC como entrada (1620). El codificador comprueba a continuación (1630) si el coeficiente es cero. Si el coeficiente no es cero, el codificador reinicia el conteo de serie (1638) y codifica el coeficiente utilizando codificación DCBA (1640).

En otro caso (es decir, si el coeficiente QC es cero), el codificador incrementa la variable de conteo de serie (1632), y comprueba si el conteo de serie actual excede el umbral de conteo de serie (1634). El umbral de conteo de serie puede ser estático o puede depender de un factor tal como el tamaño de un bloque de coeficientes (por ejemplo, cuatro ceros en una secuencia de entrada de 256 coeficientes), o puede ser adaptativo de alguna otra manera. Por ejemplo, el umbral puede ser incrementado o decrementado, con o sin relación con el número de coeficientes de una secuencia de entrada. Si el conteo de serie excede el umbral, el codificador cambia el estado de codificación a codificación de nivel de serie ["RLE"] (1636). El codificador codifica entonces el coeficiente utilizando codificación DCBA (1640).

Tras la codificación del coeficiente, el codificador comprueba el estado de codificación (1650). Si el estado de codificación ya no es DCBA (por ejemplo, si el codificador ha cambiado el estado de codificación a RLE como resultado de exceder un número de umbral de coeficientes cero), el codificador empieza la codificación de los coeficientes como longitudes y niveles de serie (1660). La codificación de nivel de serie puede ser realizada de varias formas incluyendo, por ejemplo, codificación de Huffman, codificación vectorial de Huffman, o codificación CBA (potencialmente con contextos diferentes a la codificación CBA, como se ha descrito anteriormente). En algunas realizaciones, la codificación de nivel de serie se realiza utilizando codificación de Huffman con dos tablas de código de Huffman, donde una tabla se utiliza para codificar datos en los que las secuencias más cortas son más probables, y una tabla se utiliza para codificar datos en los que las secuencias más largas son más probables. El codificador trata cada tabla, y elige códigos de una de las tablas, con un bit de señal que indica qué tabla ha usado el codificador.

Si el estado de codificación no ha cambiado, el codificador determina (1670) si existen más coeficientes que han de ser codificados. Si es así, el codificador introduce el siguiente coeficiente (1620) y continúa el proceso de codificación.

C. Descodificación Aritmética basada en Contexto

La Figura 17 muestra una técnica (1700) para descodificar coeficientes que han sido codificados con la utilización de codificación CBA de acuerdo con una implementación, en la que el descodificador determina adaptativamente un punto de conmutación para cambiar a descodificación de longitudes de serie y de niveles de señal. El punto de conmutación adaptativa depende de un conteo de coeficientes consecutivos de valor cero en los datos, que son niveles de señal para coeficientes de transformación cuantificados, que progresan desde el coeficiente de frecuencia más baja hasta el coeficiente de frecuencia más alta. Alternativamente, los datos son otra forma y/u organización de datos de audio.

Para empezar, el descodificador inicializa varias variables. Específicamente, el descodificador establece un conteo de serie en 0 (1710) y establece un estado de descodificación en aritmética directa basada en contexto (DCBA) (1712).

El descodificador descodifica el siguiente coeficiente cuantificado utilizando DCBA (1720) buscando el número que el codificador ha utilizado para representar el coeficiente en codificación aritmética, y extrayendo el valor del coeficiente a partir de ese número. El descodificador actualiza a continuación el conteo de serie en base al coeficiente descodificado (1730) (específicamente, en base a si el coeficiente descodificado es un valor cero para reinicia o incrementar el conteo de serie).

El descodificador comprueba si el conteo de serie excede un umbral (1740). El umbral de conteo de serie puede ser estático o puede depender de un factor tal como el tamaño de un bloque de coeficientes (por ejemplo, cuatro ceros en una secuencia de entrada de 256 coeficientes), o puede ser adaptativo de alguna otra manera. Si el conteo de serie excede el umbral, el descodificador empieza a descodificar los coeficientes codificados utilizando descodificación de nivel de serie (1750). La descodificación de nivel de serie puede ser llevada a cabo de varias formas, incluyendo, por ejemplo, descodificación de Huffman, descodificación vectorial de Huffman, o descodificación CBA (potencialmente con contextos diferentes a la descodificación CBA anterior, como se ha descrito en lo que antecede). En algunas realizaciones, la descodificación de nivel de serie se realiza utilizando descodificación de Huffman con dos tablas potenciales de código de Huffman, donde una tabla se utiliza para descodificar datos en los que las secuencias más cortas son más probables,

y una tabla se utiliza para descodificar datos en los que las secuencias más largas son más probables. Cuando el descodificador recibe un código, un bit de señal presente en el código indica qué tabla ha sido utilizada por el codificador, y el descodificador busca el código en una tabla apropiada.

Si el conteo de serie no supera el umbral, el descodificador sigue procesando coeficientes hasta que la descodificación haya terminado (1760).

VI. Codificación Sin-Tabla

En algunas realizaciones que utilizan codificación de Huffman, un codificador tal como el codificador (200) de la Figura 2, utiliza un código de escape para que una tabla de código de Huffman indique que un símbolo (o una combinación de símbolos) particular no tiene un código asociado en la tabla. A veces, se utiliza un código de escape para indicar que un símbolo particular (por ejemplo, un valor escalar para un nivel que no está representado en una tabla escalar de código de Huffman para niveles, una longitud de serie que no está representada en una tabla escalar de código de Huffman para longitudes de serie, etc.), debe ser codificado sin utilizar ninguna tabla de Huffman. En otras palabras, el símbolo debe ser codificado utilizando una técnica de codificación “sin-tabla”.

En algunas realizaciones que utilizan codificación aritmética, se utiliza a veces un código de escape para indicar que un símbolo particular no ha sido codificado aritméticamente. El símbolo podría ser codificado utilizando un código de una tabla de Huffman, o podría ser codificado utilizando una técnica de codificación “sin-tabla”.

Algunas técnicas de codificación sin-tabla utilizan códigos de longitud fija para representar símbolos. Sin embargo, la utilización de códigos de longitud fija puede conducir a códigos innecesariamente largos.

En algunas realizaciones, por lo tanto, los símbolos tales como los coeficientes de transformación cuantificados, están representados por códigos de longitud variable en una técnica de codificación sin-tabla cuando los símbolos no están codificados de otra manera. Un descodificador tal como el descodificador (300) de la Figura 3 ejecuta una técnica correspondiente de descodificación sin-tabla.

Por ejemplo, la Tabla 6 muestra un seudo-código para una implementación de dicha técnica de codificación sin-tabla.

```
Si (valor <2^8) {
        Enviar “0”;
        Enviar valor usando 8 bits;
}
        o bien si (valor <2^16) {
                Enviar “10”;
                Enviar valor usando 16 bits
        }
                o bien si (valor <2^24) {
                        Enviar “110”;
                        Enviar valor usando 24 bits;
                }
                        o bien si (valor <2^31) {
                                Enviar “111”;
                                Enviar valor usando 31 bits;
                        }
```

Tabla 6: Seudo-código para técnica de codificación sin-tabla en una implementación

El número de bits que utiliza el codificador para codificar el coeficiente depende del valor del coeficiente. El codificador envía un valor de uno, dos o tres bits para indicar el número de bits utilizados para codificar el valor, y a continuación envía el propio valor codificado utilizando 8, 16, 24 ó 31 bits. El número total bits que el codificador utiliza para codificar el coeficiente, está comprendido en la gama de 9 bits para un valor menor de 2^8, a 34 bits para un valor mayor o igual a 2^{24}, pero menor que 2^{31}.

Para una serie de coeficientes, el promedio de bits enviados será igual a:

$$P(0 \leq C < 2^8)*9 + P(2^8 \leq C < 2^{16})*18 + P(2^{16} \leq C < 2^{24})*27 + P(2^{24} \leq C < 2^{31})*34.$$

donde $P(m \leq C < n)$ es la probabilidad de ocurrencia, en una secuencia de entrada, de un coeficiente C dentro del intervalo indicado. Por lo tanto, son posibles ahorros significativos de bits cuando un gran porcentaje de los coeficientes son pequeños (por ejemplo, menos de 2^{16}).

Alternativamente, el codificador y el descodificador utilizan otra técnica de codificación/ descodificación sin-tabla.

Habiendo descrito e ilustrado los principios de nuestra invención con referencia a las diversas realizaciones descritas, se comprenderá que las realizaciones descritas pueden ser modificadas en cuanto a estructura y detalle sin apartarse de

tales principios. Se comprenderá que los programas, procesos, o procedimientos aquí descritos no están relacionados o limitados en cuanto a algún tipo de entorno de computación, a menos que se indique otra cosa. Se pueden usar diversos tipos de entornos de computación especializados o de propósito general con, o llevando a cabo, operaciones de acuerdo con las enseñanzas aquí descritas. Los elementos de las realizaciones descritas mostrados en software pueden ser implementados en hardware, y viceversa.

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

REIVINDICACIONES

1.- Un procedimiento de codificación de datos de audio que comprende varios símbolos en un sistema informático, **caracterizándose** el procedimiento **por**

codificar un primer vector que comprender un primer número de símbolos que representan la dimensión del primer vector, en el cual la codificación del primer vector comprende:

verificar si un código está incluido para el primer vector en una primera tabla de código procedente de un conjunto de varias tablas de códigos para el primer número de símbolos, en el cual la primera tabla de códigos comprende códigos para representan vectores que tienen el primer número de símbolos y un código de escape;
si un código para el primer vector está incluido en la primera tabla de códigos, la representación del primer vector con el código procedente de la primera tabla de códigos; y
en caso contrario, dividir el primer vector en segundos vectores , comprendiendo cada uno un segundo número de símbolos que representa la dimensión del segundo vector, en el cual el segundo número de símbolos difiere del primer número de símbolos, y
para cada uno de los segundos vectores, codificar el segundo vector,

en el cual la codificación del segundo vector comprende:

seleccionar una segunda tabla basada en el segundo número de símbolo; y
representar el segundo vector con un código procedente de la segunda tabla de códigos, en el cual la segunda tabla de códigos difiere de la primera tabla de códigos.

2.- El procedimiento de la reivindicación 1, en el cual el primer número de símbolos es superior al segunda número de símbolos, y en el cual el primer vector tiene una probabilidad de ocurrencia más alta que el segundo vector.

3.- El procedimiento de la reivindicación 1 en el cual la primera tabla de códigos comprende:

códigos para representar vectores probables de un conjunto de vectores posibles que tienen el primer número de símbolos; y
un código de escape para vectores menos probables.

4.- El procedimiento según la reivindicación 1, en el cual el primer número de símbolos difiere del segundo número de símbolo por un factor de 2.

5.- Un procedimiento de descodificación de datos de audio que comprende varios vectores codificados en un sistema informático, **caracterizándose** el procedimiento **por**:

descodificar un primer vector, que tiene un primer número de símbolos en el cual la descodificación del primer vector comprende:

recibir un primer código;
buscar el primer código en una primera tabla de códigos en un grupo de barias tablas de códigos;
determinar si el primer código es un código de escape;
si el primer código es un código de escape:

recibir un segundo código que representa una parte del primer vector, en el cual el segundo código no está incluido en la primera tabla de códigos; y
descodificar el segundo código;

si el primer código no es un código de escape:

buscar símbolos para el primer vector en la primera tabla de códigos; e
incluir los símbolos en una corriente de datos descodificada.

en el cual el primer número de símbolos es una base para saber si el primer código es un código de escape o no es un código de escape.

6.- El procedimiento de la reivindicación 5 en el cual la descodificación del segundo código comprende:

buscar el segundo código en una segunda tabla de códigos en el grupo de varias tablas de códigos;
determinar si el segundo código es un código de escape;
si el segundo código e s un código de escape:

recibir un tercer código que representa el primer vector, en el cual el tercer código no está incluido en la segunda tabla de códigos; y

descodificar el tercer código; y
si el segundo código no es un código de escape:

buscar símbolos para el primer vector en la segunda tabla de códigos; e
incluir los símbolos en la corriente de datos descodificada;

en el cual la segunda tabla de códigos difiere de la primera tabla de códigos.

7.- El procedimiento de la reivindicación 5 que comprende, además:

descodificar un segundo vector, en el cual el segundo vector tiene un segundo número de símbolos, y en el cual el primer número difiere del segundo número por un factor de 2.

8.- El procedimiento de la reivindicación 1 en el cual la codificación del segundo vector comprende:

determinar si un código está disponible para el segundo vector en la segunda tabla de códigos, en el cual la segunda tabla de códigos comprende un código de escape que indica un cambio respecto de la segunda tabla de códigos para una técnica de codificación sin tabla;
si el código está disponible para el segundo vector en la segunda tabla de códigos, representar el segundo vector con el código procedente de la segunda tabla de códigos; y
en caso contrario, dividir el segundo vector en varios símbolos simples, y para cada uno de los varios símbolos simples, representar el símbolo simple con un código obtenido por la técnica de codificación sin tabla.

9.- El procedimiento de la reivindicación 1 en el cual la codificación del segundo vector comprende:

determinar si un código está disponible para el segundo vector en la segunda tabla de códigos, en el cual la segunda tabla de códigos comprende un código de escape que indica un cambio respecto de la segunda tabla de códigos para una tercera tabla de códigos de las varias tablas de códigos;
si el código está disponible para el segundo vector en la segunda tabla de códigos, representar el segundo vector con el código procedente de la segunda tabla de códigos; y
en caso contrario, dividir el segundo vector en varios terceros vectores, comprendiendo cada uno un tercer número de símbolos y para cada uno de los varios terceros vectores, codificar el tercer vector utilizando la tercera tabla de códigos, en el cual el tercer número de símbolos difiere del primer número de símbolos y difiere del segundo número de símbolos.

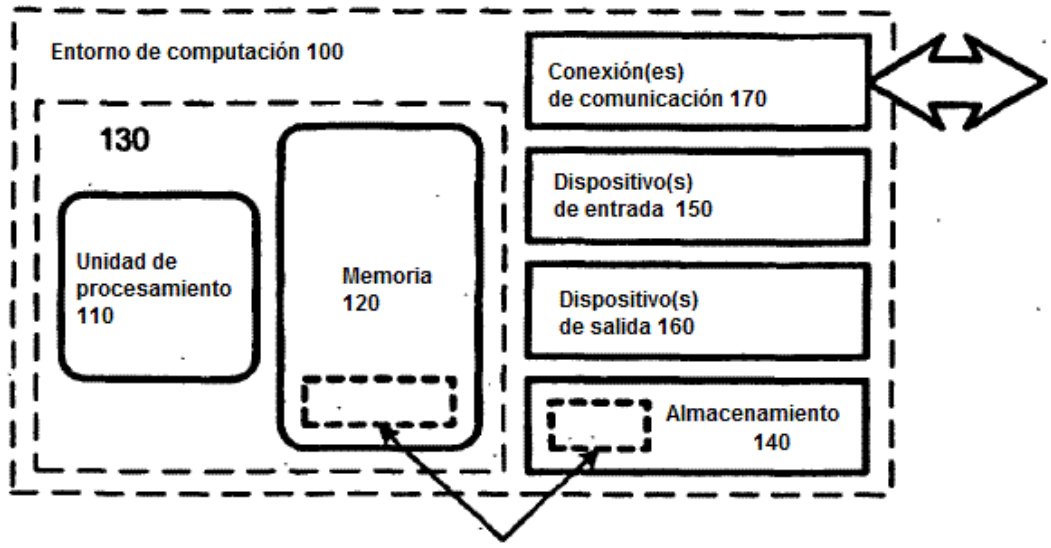
10.- El procedimiento de la reivindicación 5 en el cual la descodificación del segundo código comprende:

buscar el segundo código en una segunda tabla de códigos en el grupo de varias tablas de códigos;
determinar si el segundo código es un código de escape procedente de la segunda tabla de códigos;
si el segundo código es el código de escape procedente de la segunda tabla de códigos, para cada uno de los varios símbolos simples en el primer vector, descodificar un código que representa el símbolo simple utilizando una técnica de descodificación sin tabla e incluir el símbolo simple en la corriente de datos descodificada;
si el segundo código no es el código de escape procedente de la segunda tabla de códigos:

buscar un segundo vector, que tiene un segundo número de símbolos, en la segunda tabla de códigos, en el cual el segundo vector está asociado al segundo código en la segunda tabla de códigos; e
incluir el segundo número de símbolos del segundo vector en la corriente de datos descodificada.

11.- Soporte legible por ordenador que almacena instrucciones que pueden ser ejecutadas por ordenador para hacer que un ordenador efectúe el procedimiento de una cualquiera de las reivindicaciones 1 a 10.

Figura 1


Entorno de computación 100
130
Unidad de procesamiento 110
Memoria 120
Conexión(es) de comunicación 170
Dispositivo(s) de entrada 150
Dispositivo(s) de salida 160
Almacenamiento 140
Software 180 que implementa técnicas de codificación/descodificación entrópica adaptativa de datos de audio

5

Figura 2

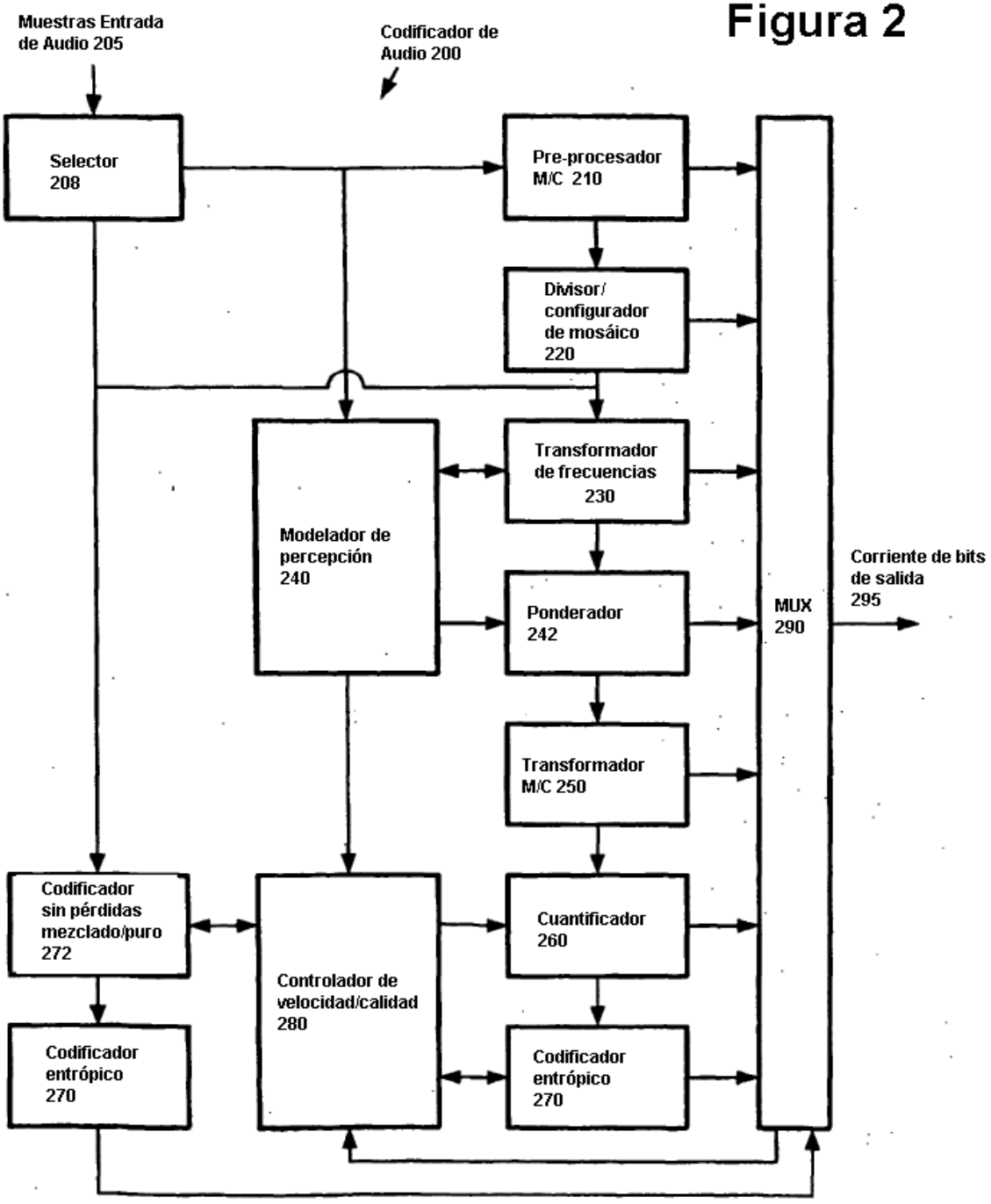

Muestras Entrada de Audio 205
Codificador de Audio 200
Selector 208
Pre-procesador M/C 210
Divisor/ configurador de mosáico 220
Transformador de frecuencias 230
Modelador de percepción 240
Ponderador 242
Transformador M/C 250
Cuantificador 260
Controlador de velocidad/calidad 280
Codificador entrópico 270
Codificador sin pérdidas mezclado/puro 272
Codificador entrópico 270
MUX 290
Corriente de bits de salida 295

Figura 3

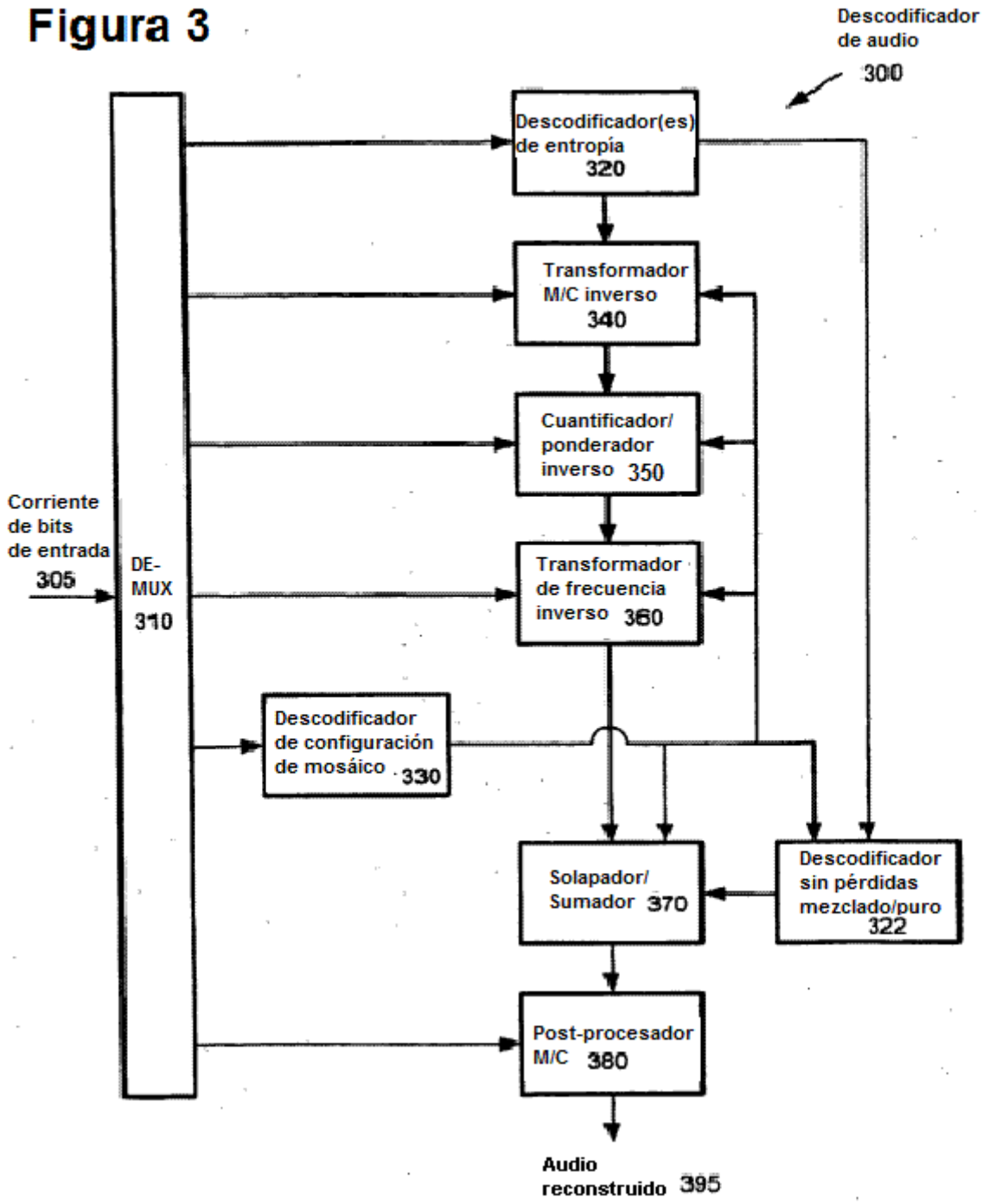

Descodificador de audio
300
Descodificador(es) de entropía
320
Transformador M/C inverso
340
Cuantificador/ ponderador inverso
350
Transformador de frecuencia inverso
360
Corriente de bits de entrada
305
DE-MUX
310
Descodificador de configuración de mosáico
330
Solapador/ Sumador
370
Descodificador sin pérdidas mezclado/puro
322
Post-procesador M/C
380
Audio reconstruido
395

Figura 4

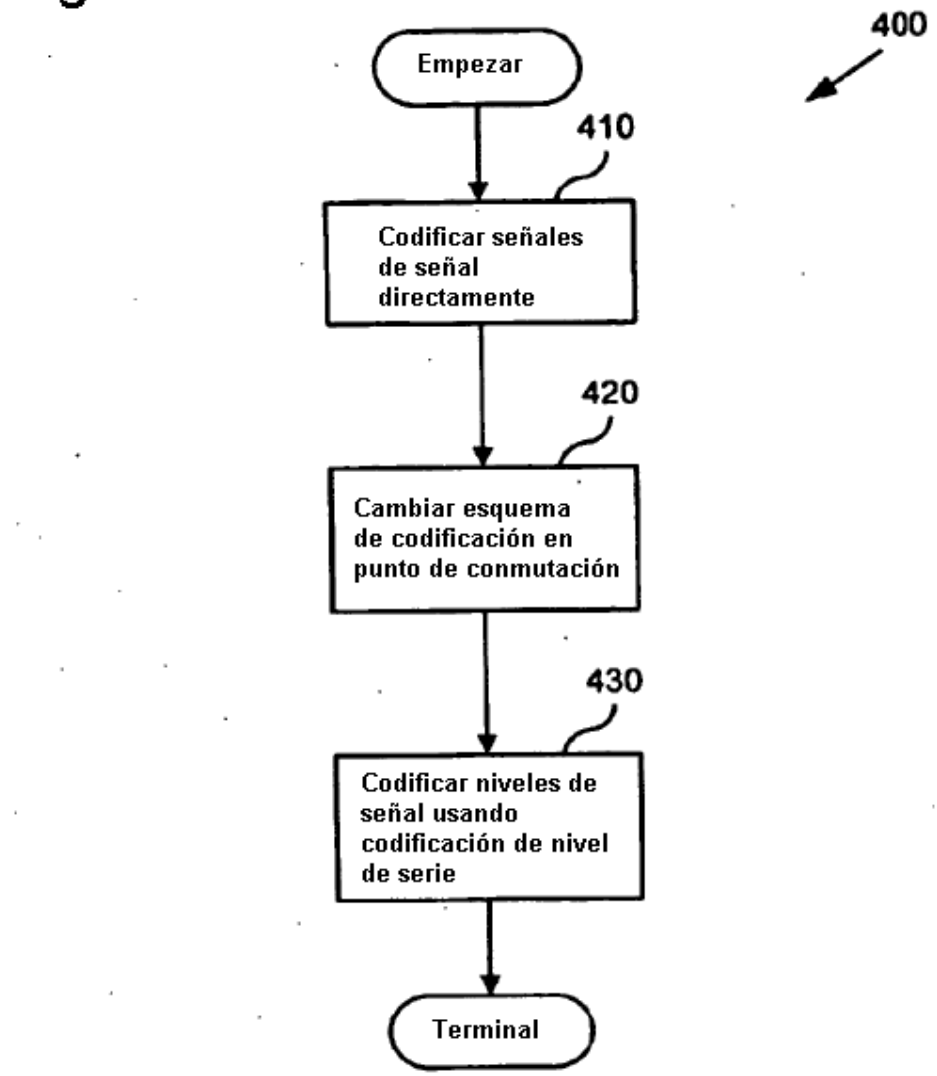

400
Empezar
410
Codificar señales de señal directamente
420
Cambiar esquema de codificación en punto de conmutación
430
Codificar niveles de señal usando codificación de nivel de serie
Terminal

Figura 4

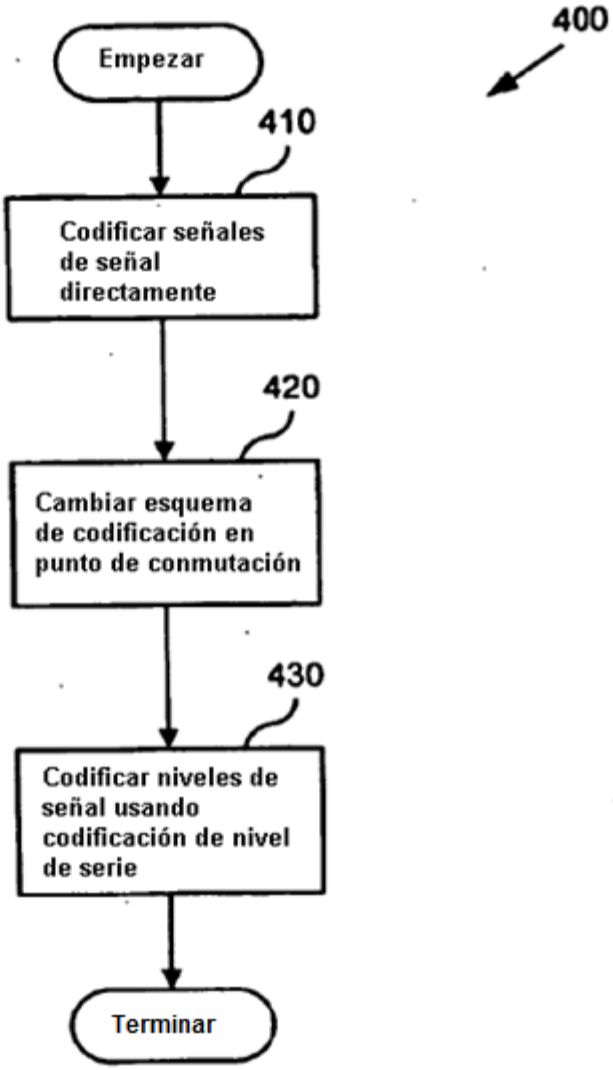

400
Empezar
410
Codificar señales de señal directamente
420
Cambiar esquema de codificación en punto de conmutación
430
Codificar niveles de señal usando codificación de nivel de serie
Terminar

Figura 5

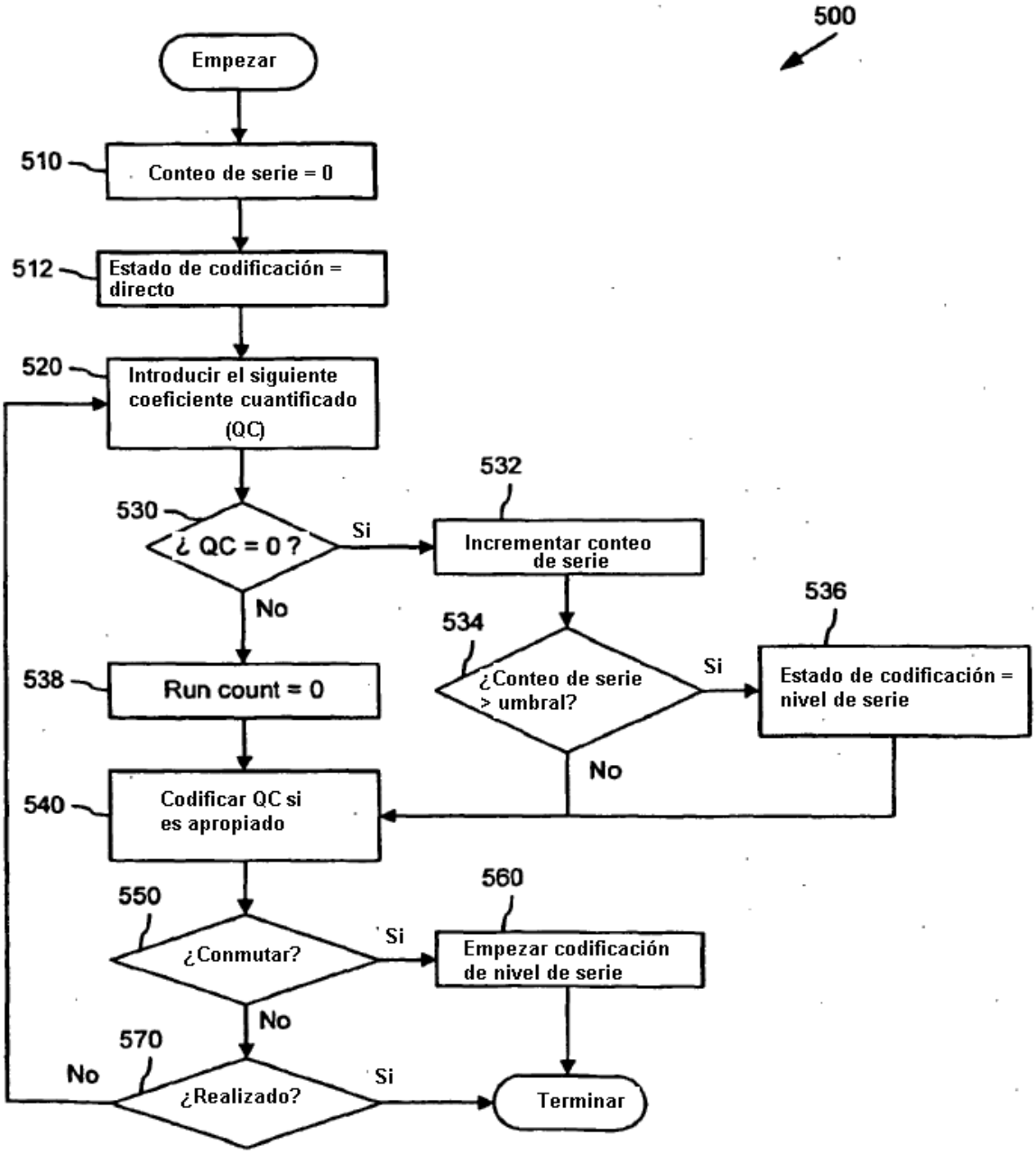

500
Empezar
510
Conteo de serie = 0
512
Estado de codificación = directo
520
Introducir el siguiente coeficiente cuantificado (QC)
530
¿ QC = 0 ?
Si
No
532
Incrementar conteo de serie
534
¿Conteo de serie > umbral?
Si
No
536
Estado de codificación = nivel de serie
538
Run count = 0
540
Codificar QC si es apropiado
550
¿Conmutar?
Si
No
560
Empezar codificación de nivel de serie
570
¿Realizado?
No
Si
Terminar

Figura 6

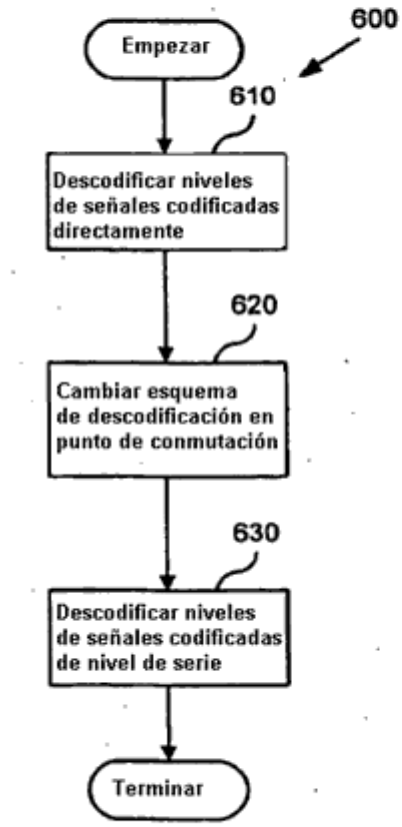


Figura 7

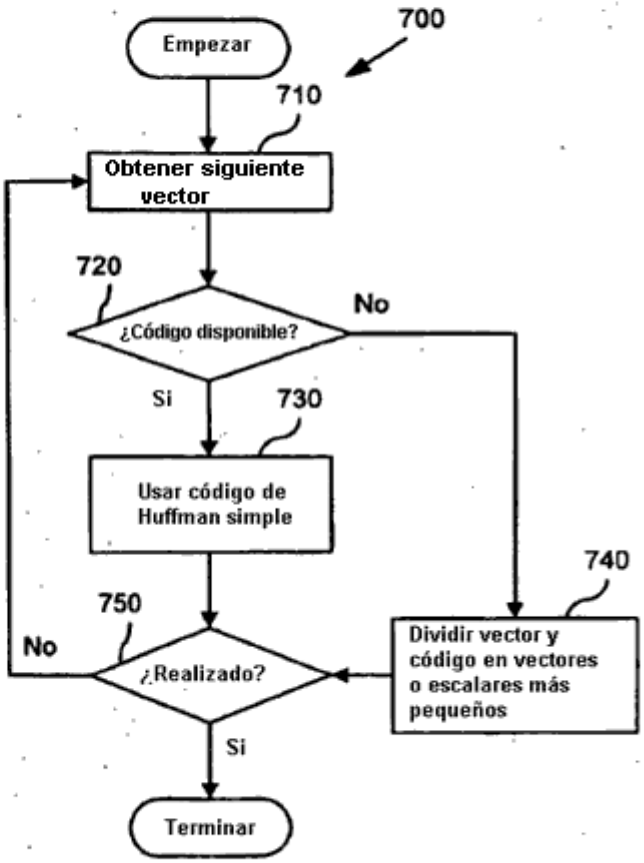


Figura 8

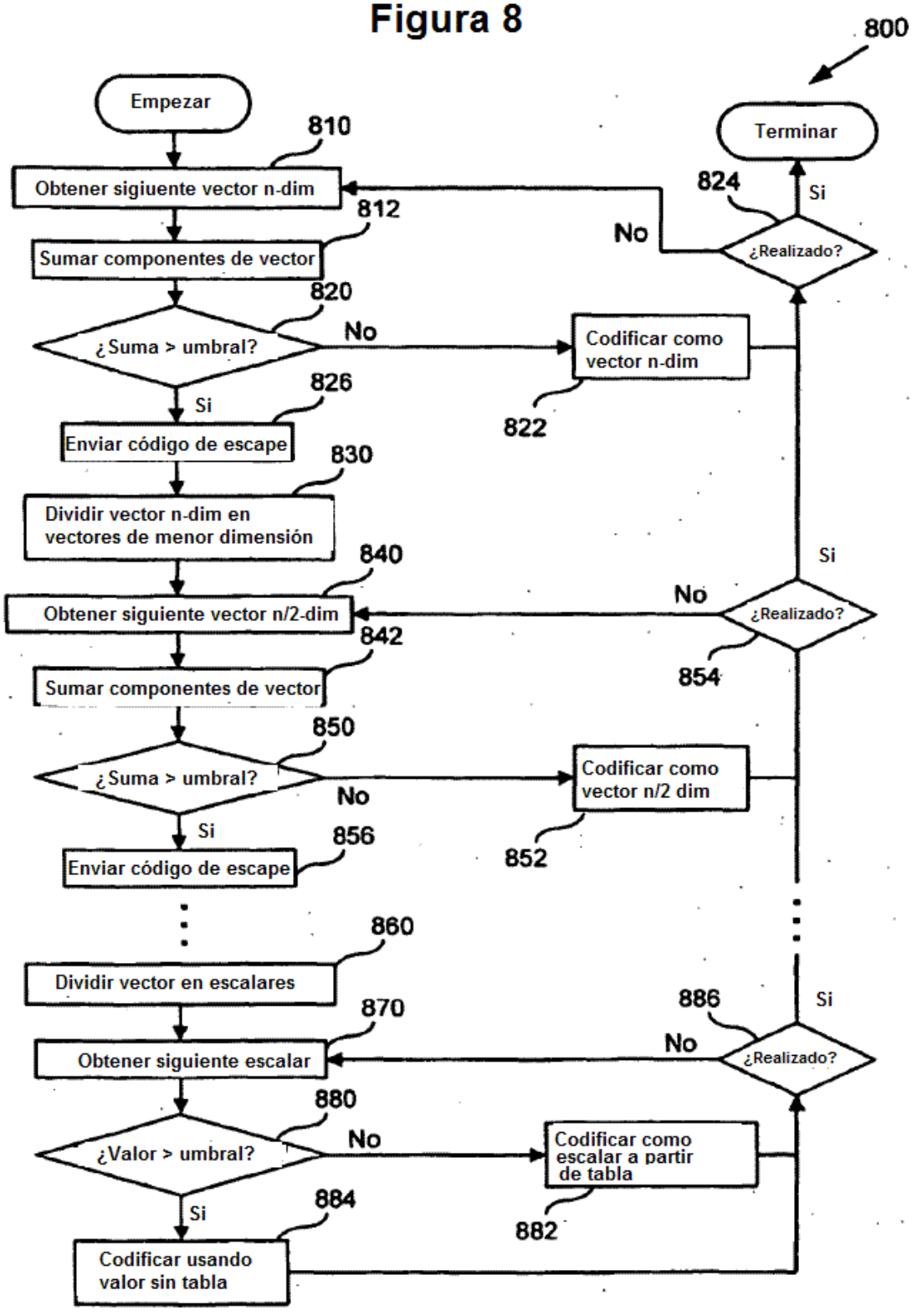

800
Empezar
Terminar
810
Obtener sigiuente vector n-dim
812
Sumar componentes de vector
820
¿Suma > umbral?
No
Codificar como vector n-dim
822
Si
826
Enviar código de escape
830
Dividir vector n-dim en vectores de menor dimensión
840
Obtener siguiente vector n/2-dim
842
Sumar componentes de vector
850
¿Suma > umbral?
No
Codificar como vector n/2 dim
852
Si
856
Enviar código de escape
860
Dividir vector en escalares
870
Obtener siguiente escalar
880
¿Valor > umbral?
No
Codificar como escalar a partir de tabla
882
Si
884
Codificar usando valor sin tabla
824
¿Realizado?
No
Si
854
¿Realizado?
No
Si
886
¿Realizado?
No
Si

Figura 9

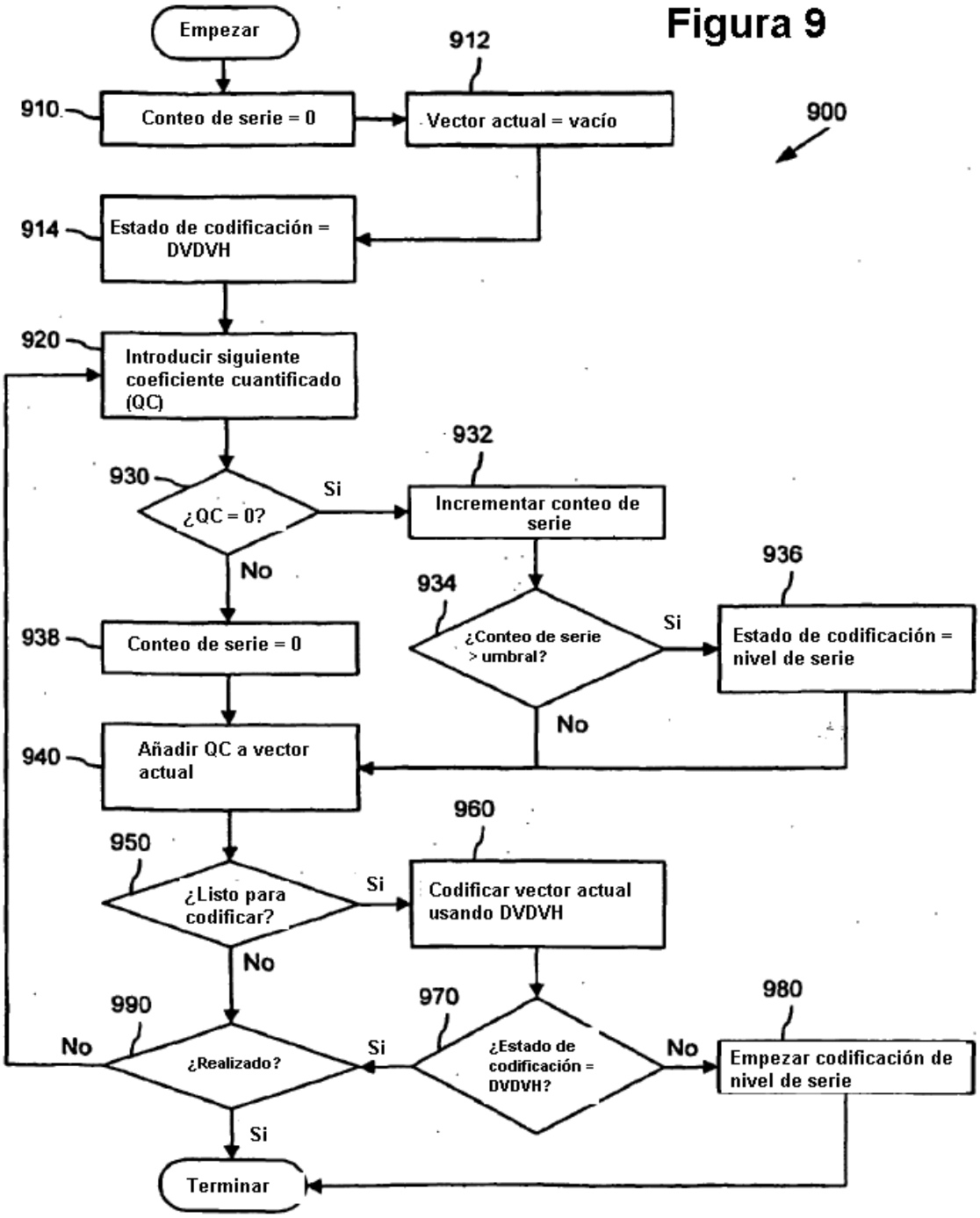

Empezar
912
910
Conteo de serie = 0
Vector actual = vacío
900
914
Estado de codificación = DVDVH
920
Introducir siguiente coeficiente cuantificado (QC)
932
930
¿QC = 0?
Si
Incrementar conteo de serie
No
936
934
¿Conteo de serie > umbral?
Si
938
Conteo de serie = 0
Estado de codificación = nivel de serie
No
940
Añadir QC a vector actual
960
950
¿Listo para codificar?
Si
Codificar vector actual usando DVDVH
No
990
970
980
No
¿Realizado?
Si
¿Estado de codificación = DVDVH?
No
Empezar codificación de nivel de serie
Si
Terminar

Figura 10

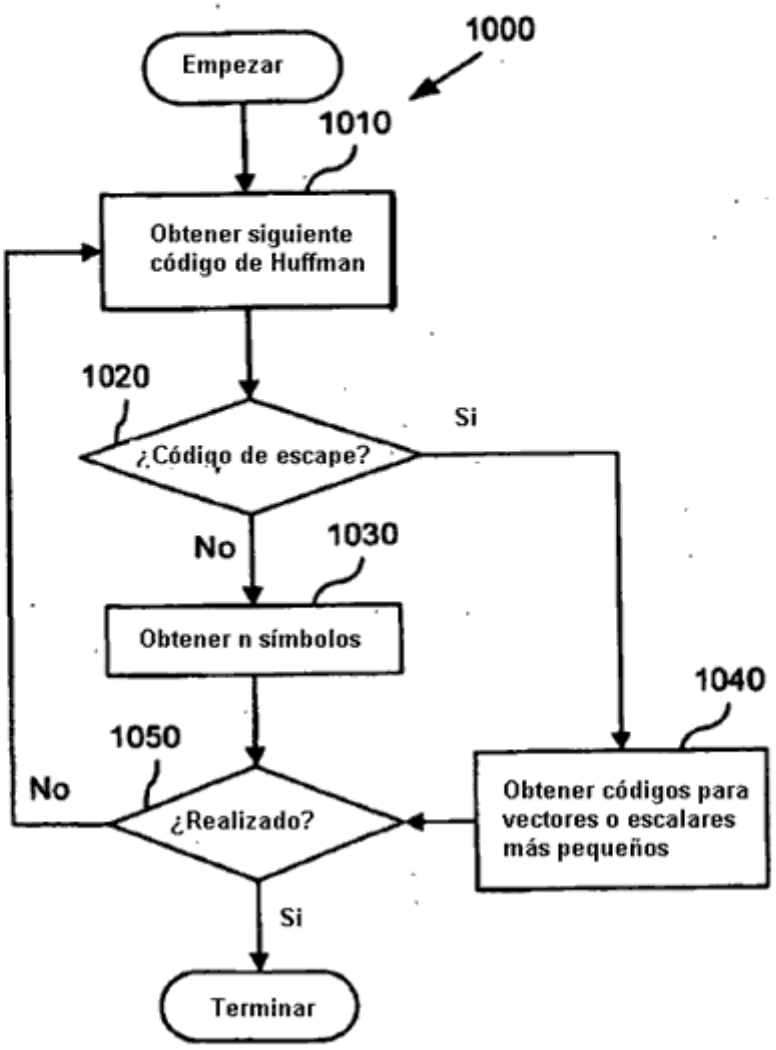

1000
Empezar
1010
Obtener siguiente código de Huffman
1020
¿Código de escape?
Si
No
1030
Obtener n símbolos
1040
Obtener códigos para vectores o escalares más pequeños
1050
No
¿Realizado?
Si
Terminar

Figura 11

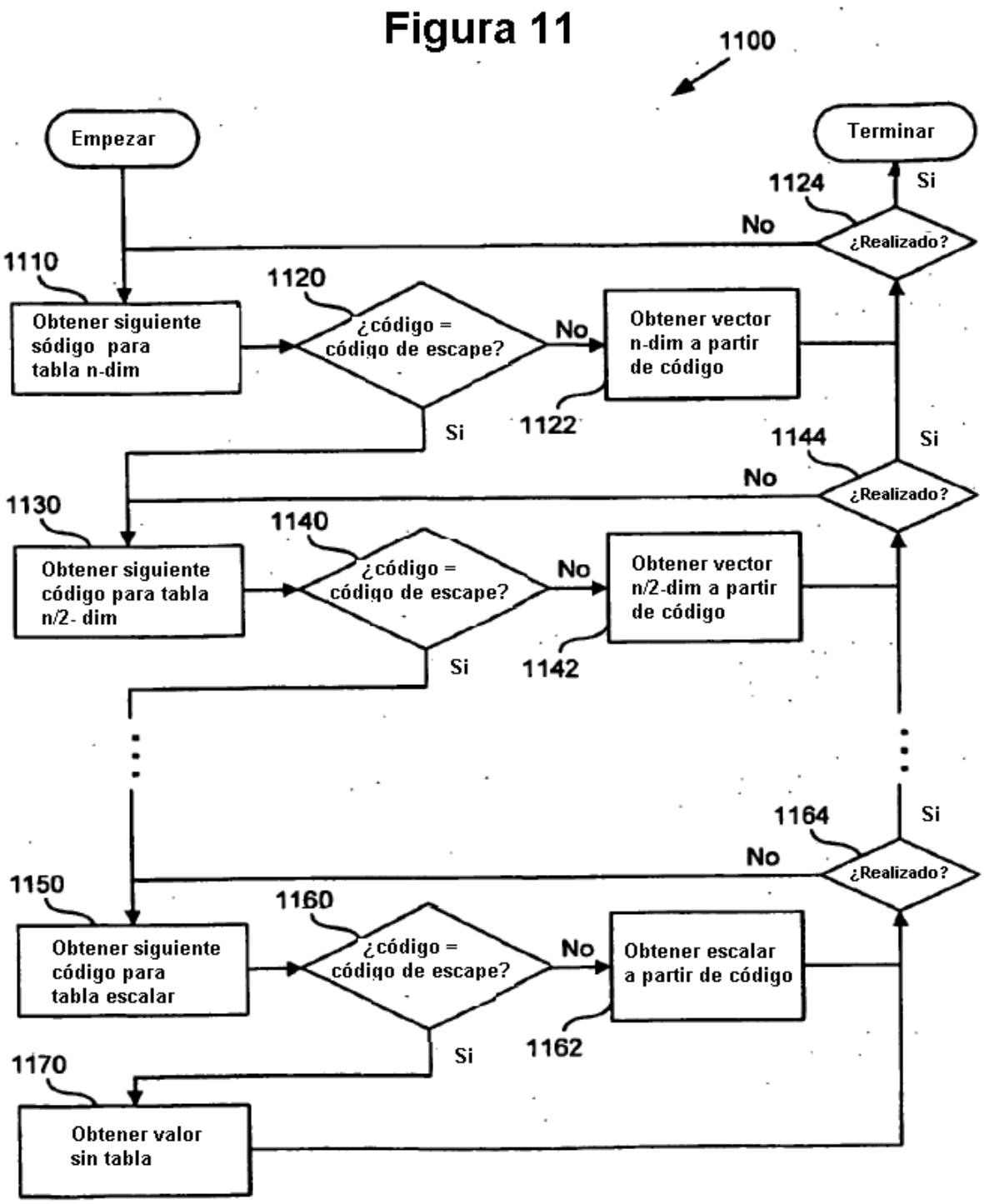

1100
Empezar
Terminar
1124
Si
No
¿Realizado?
1110
Obtener siguiente sódigo para tabla n-dim
1120
¿código = código de escape?
No
Obtener vector n-dim a partir de código
1122
Si
1144
Si
No
¿Realizado?
1130
Obtener siguiente código para tabla n/2- dim
1140
¿código = código de escape?
No
Obtener vector n/2-dim a partir de código
1142
Si
1164
Si
No
¿Realizado?
1150
Obtener siguiente código para tabla escalar
1160
¿código = código de escape?
No
Obtener escalar a partir de código
1162
Si
1170
Obtener valor sin tabla

Figura 12

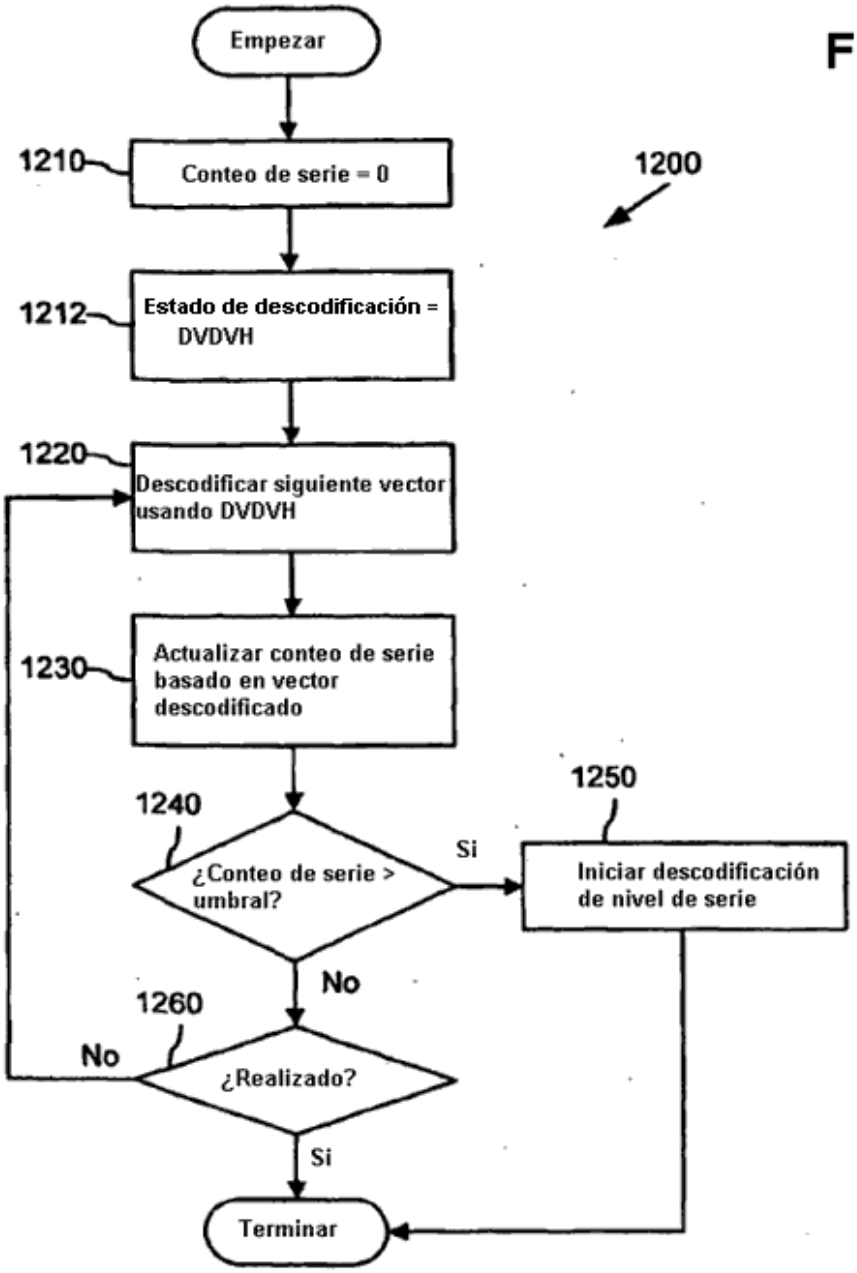

Empezar
1210
Conteo de serie = 0
1200
1212
Estado de descodificación = DVDVH
1220
Descodificar siguiente vector usando DVDVH
1230
Actualizar conteo de serie basado en vector descodificado
1240
¿Conteo de serie > umbral?
Si
1250
Iniciar descodificación de nivel de serie
No
1260
No
¿Realizado?
Si
Terminar

Figura 13 A

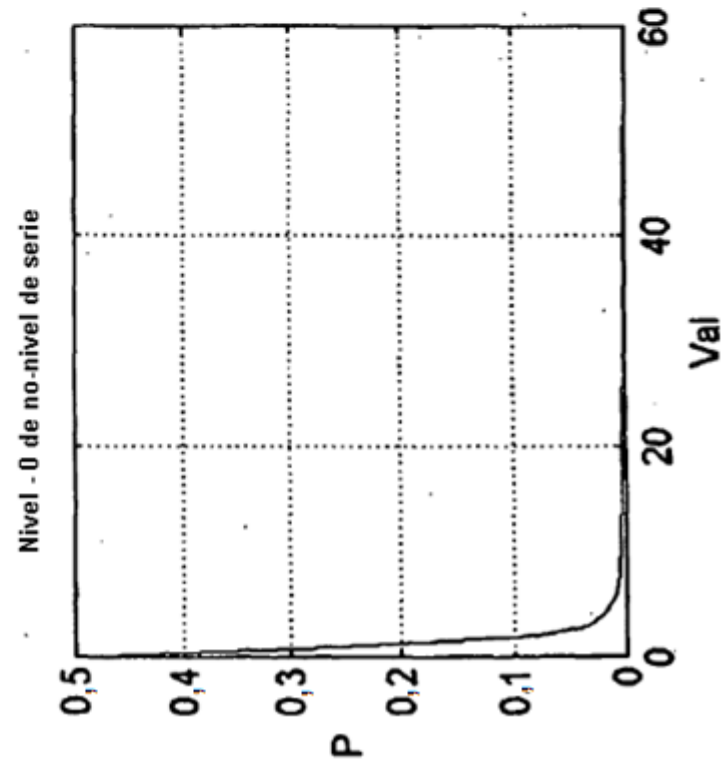


Figura 13 B

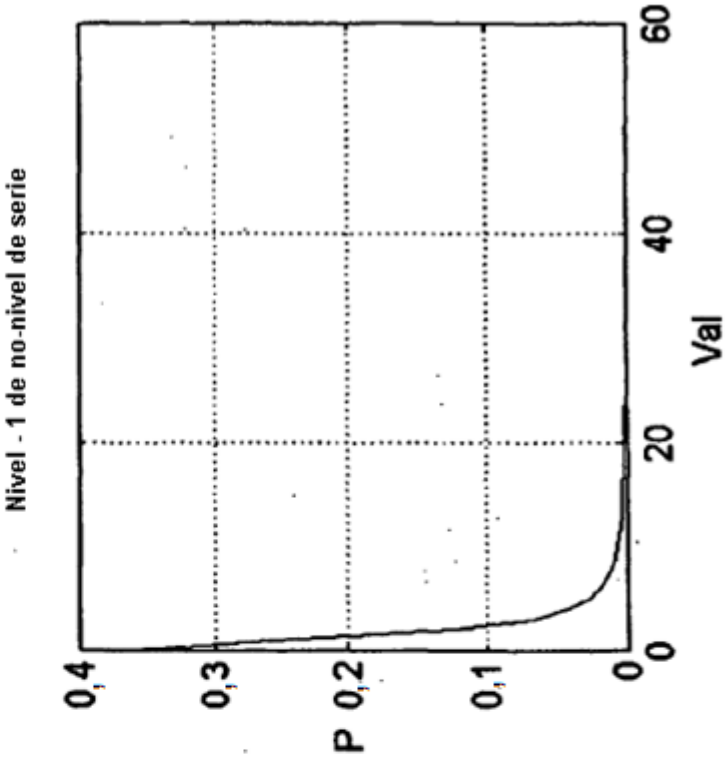


Figura 13D

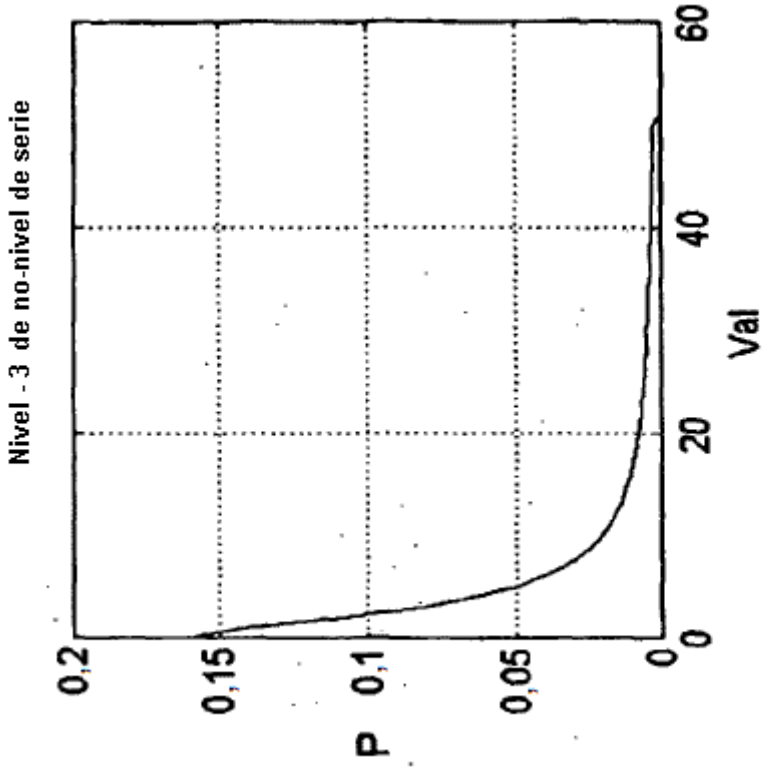


Figura 13C

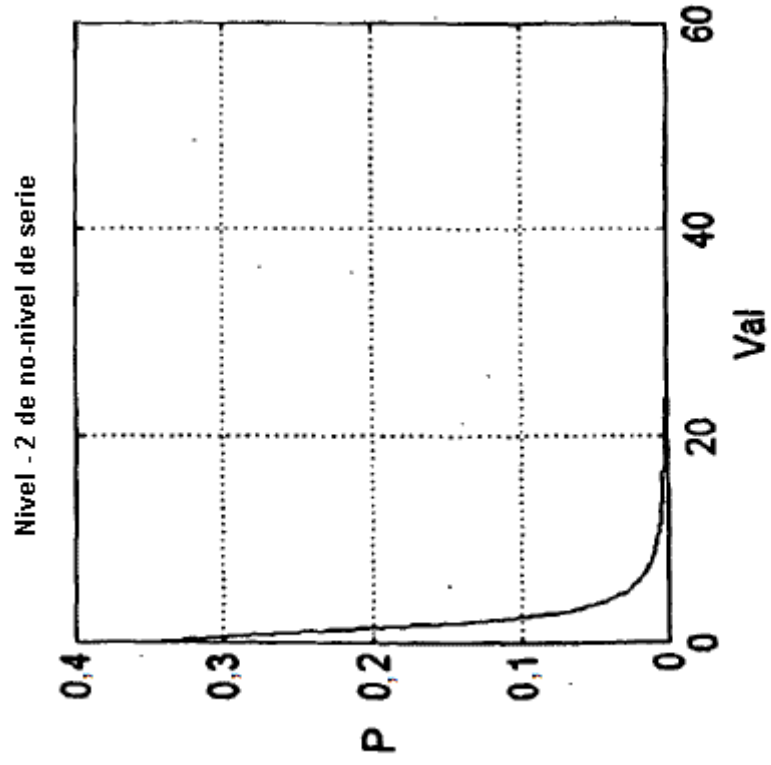


Figura 14A

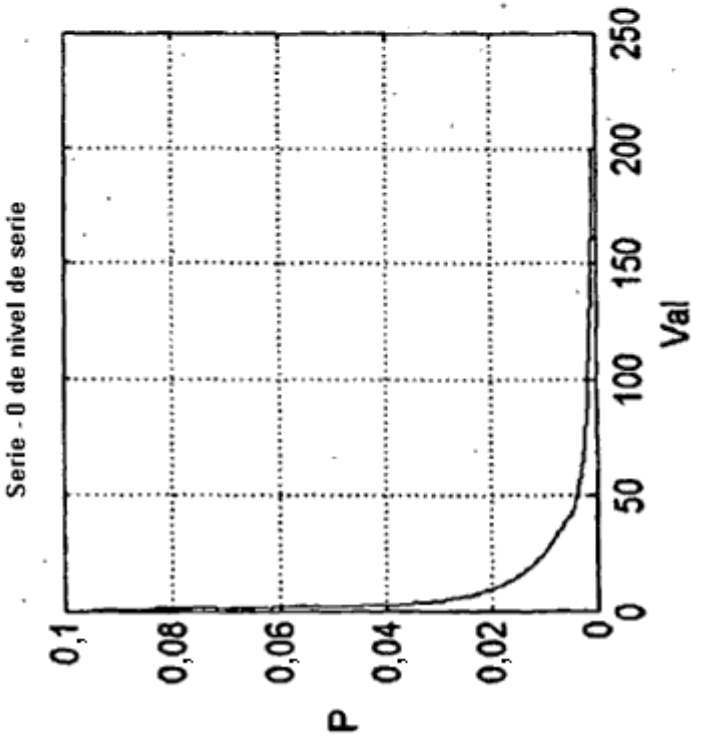


Figura 14B

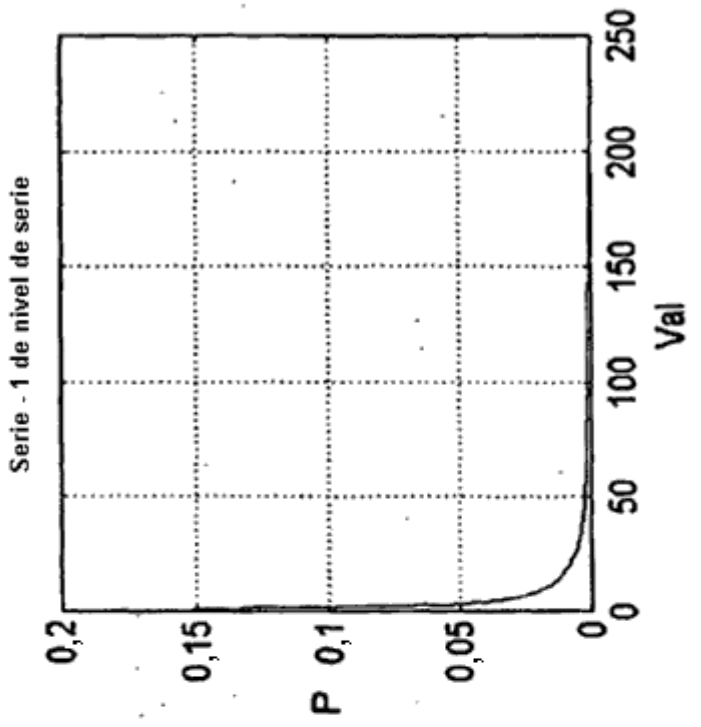


Figura 14D

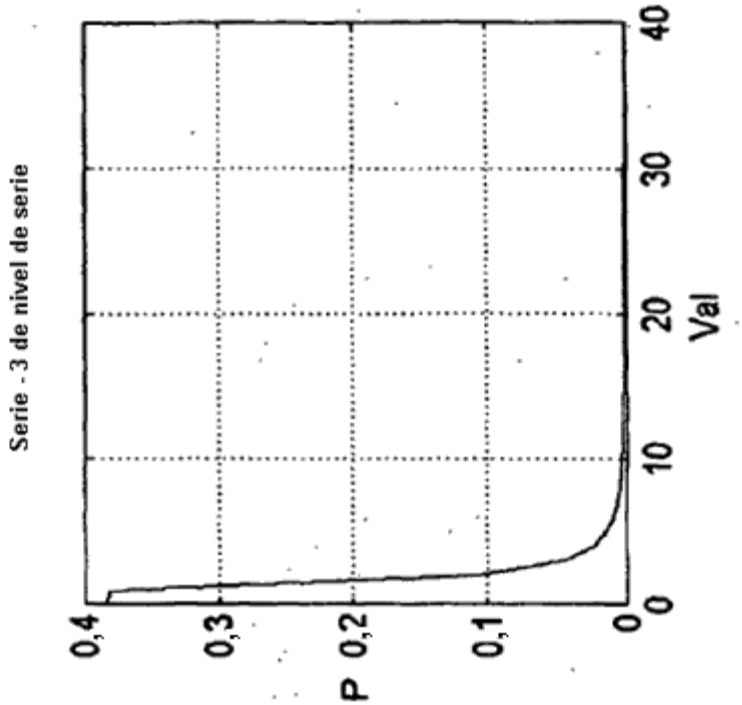


Figura 14C

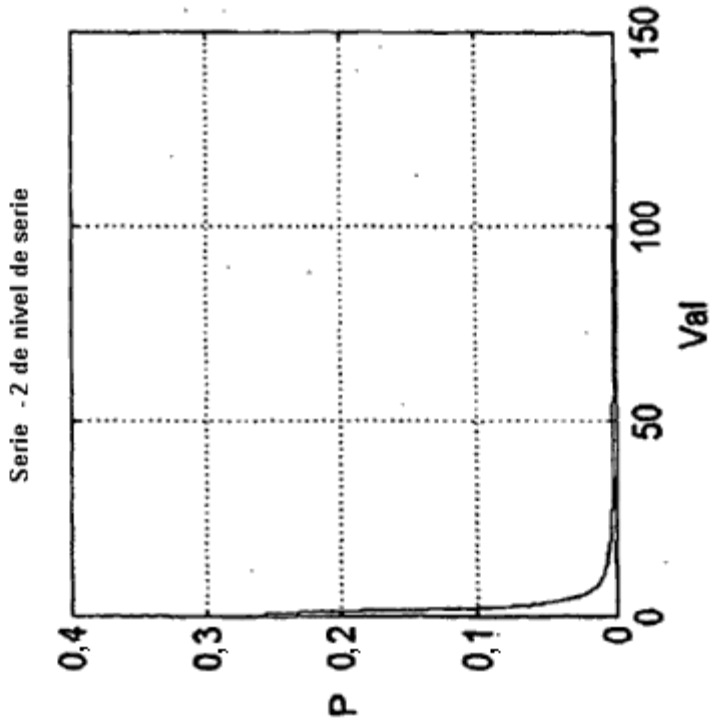


Figura 14F

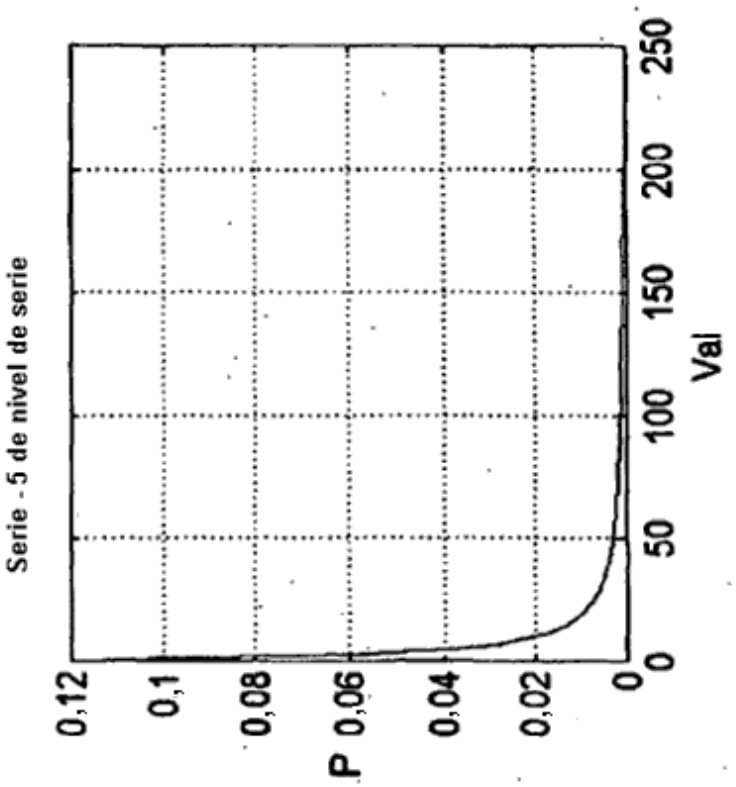


Figura 14E

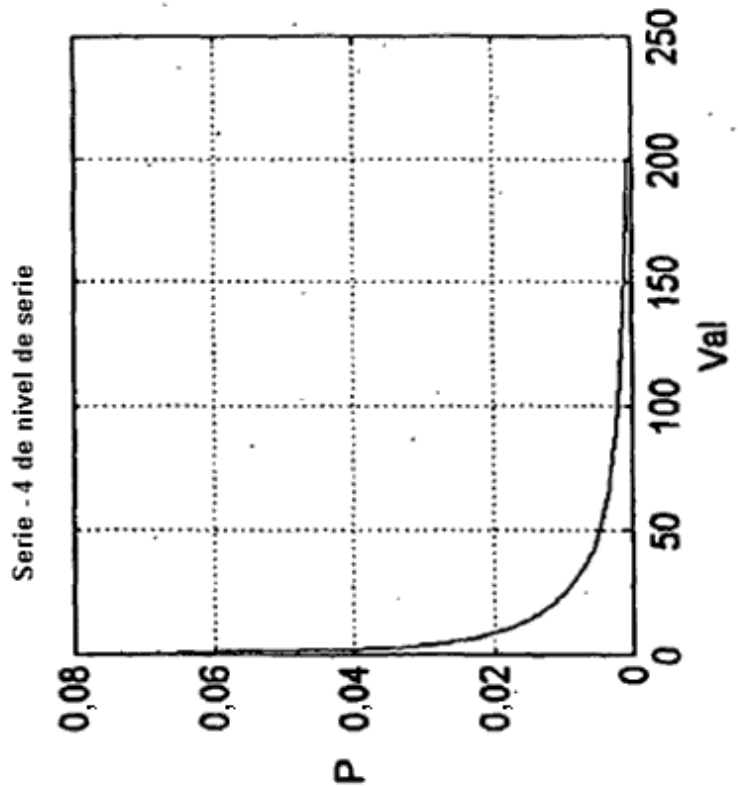


Figura 14H

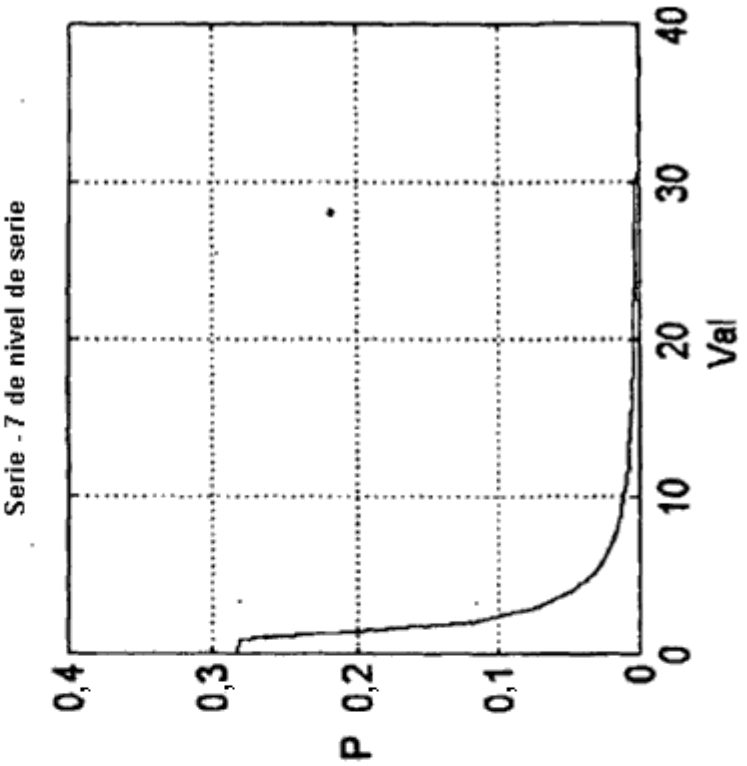


Figura 14G

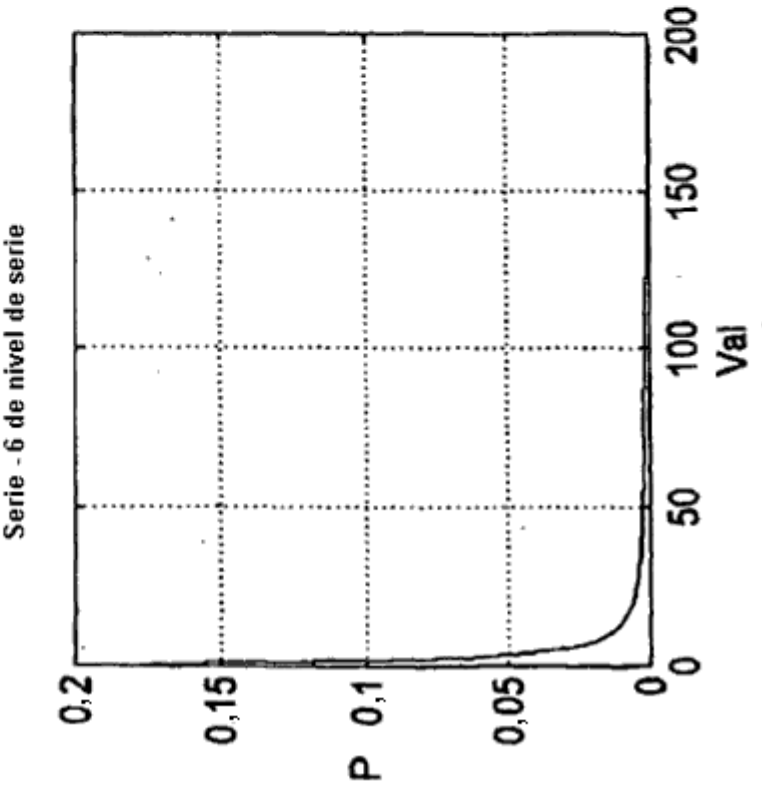


Figura 15B

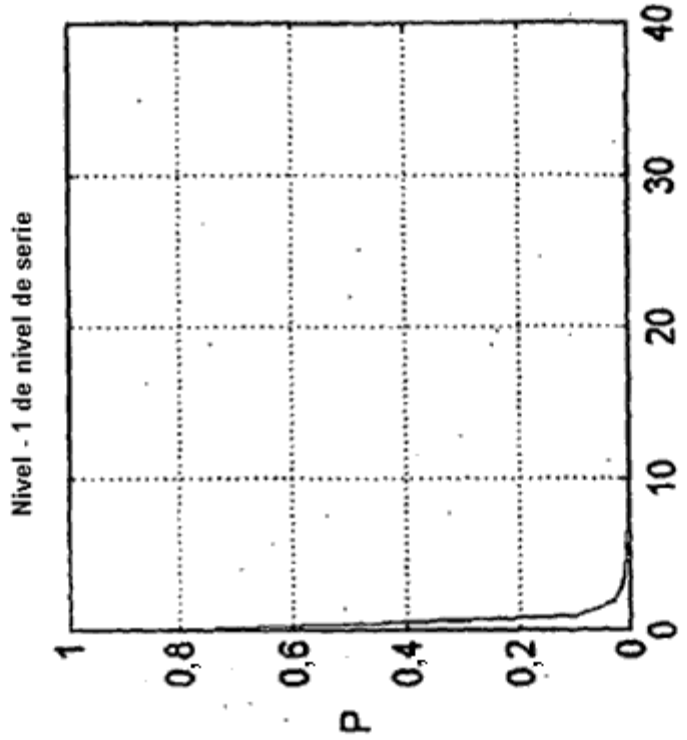


Figura 15A

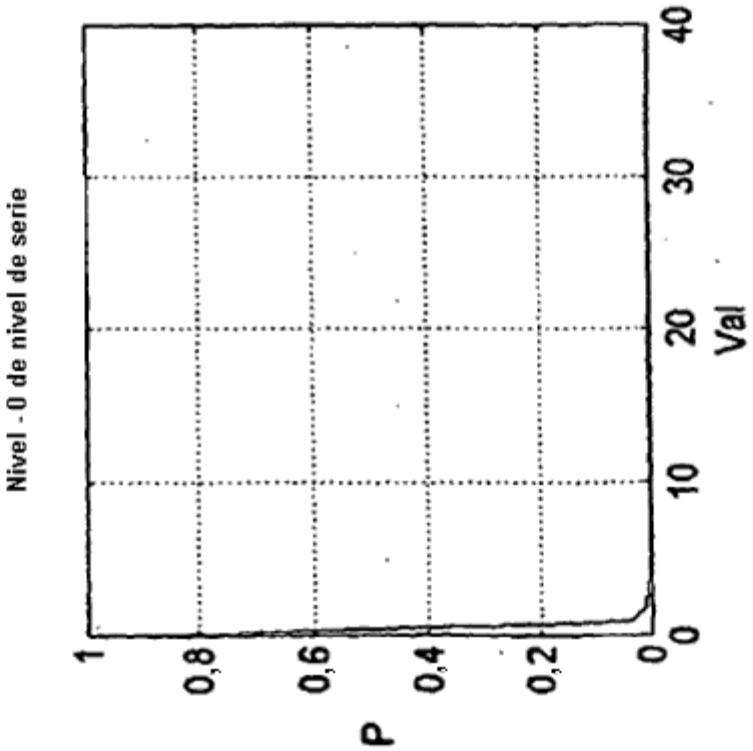


Figura 15C

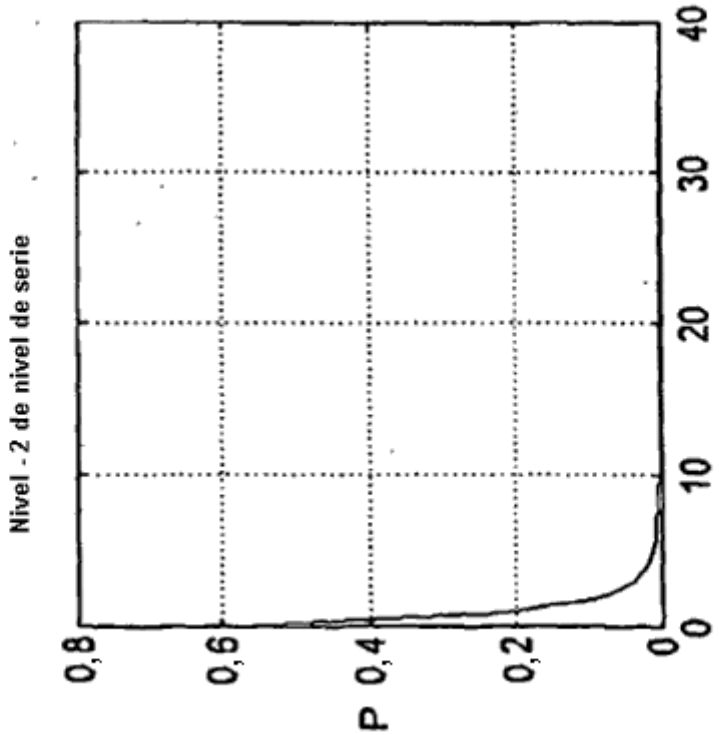


Figura 15D

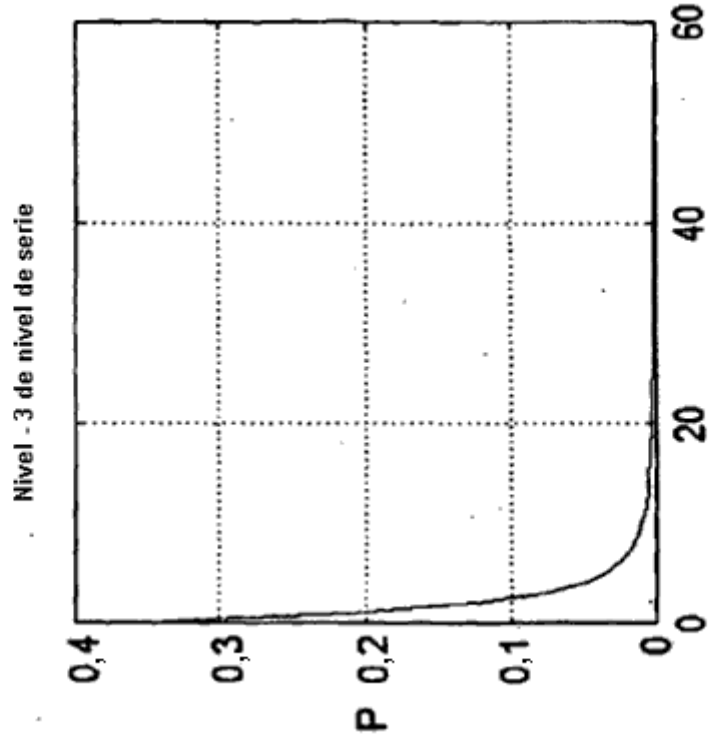


Figura 15E

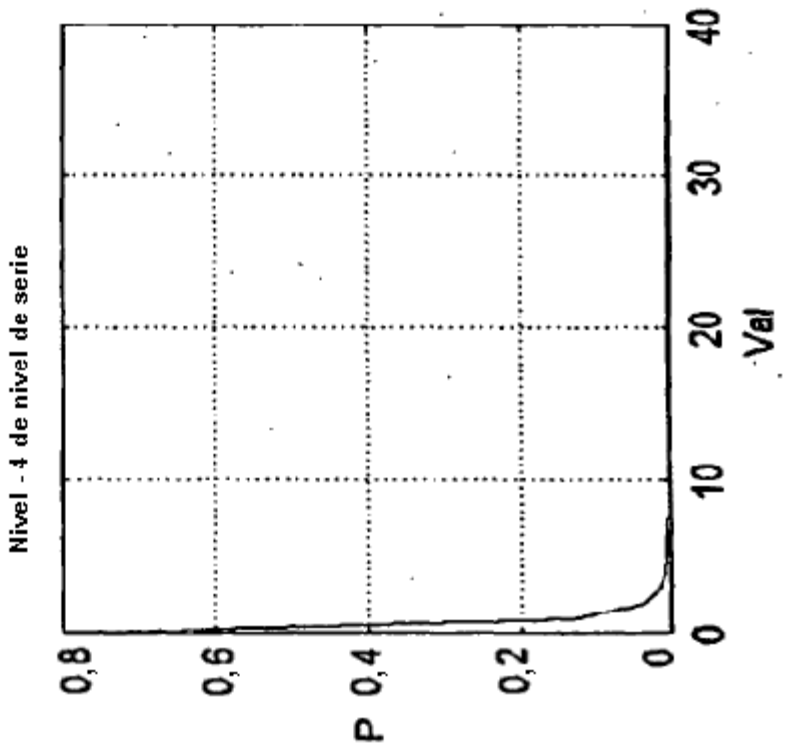


Figura 15F

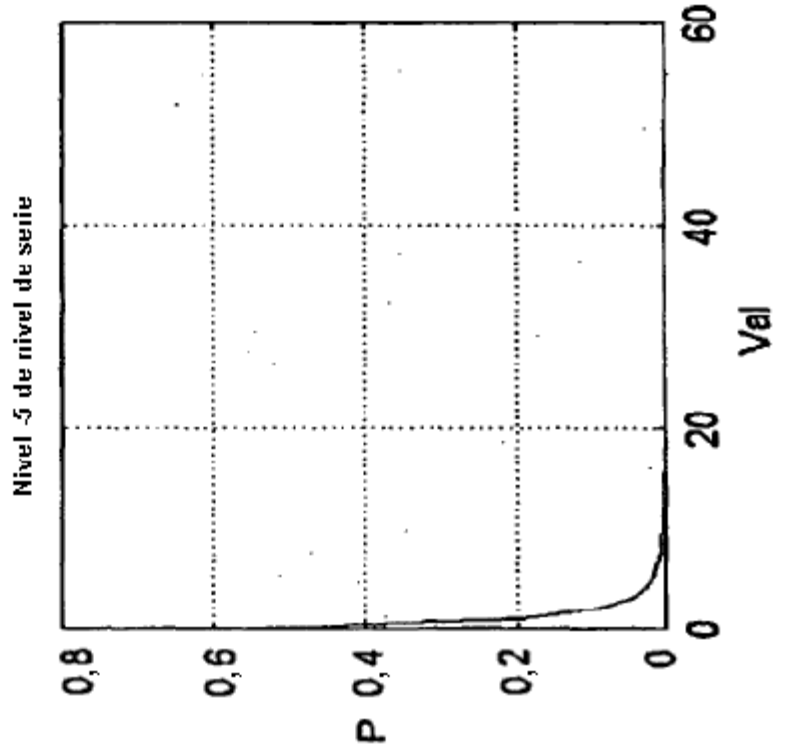


Figura 15G

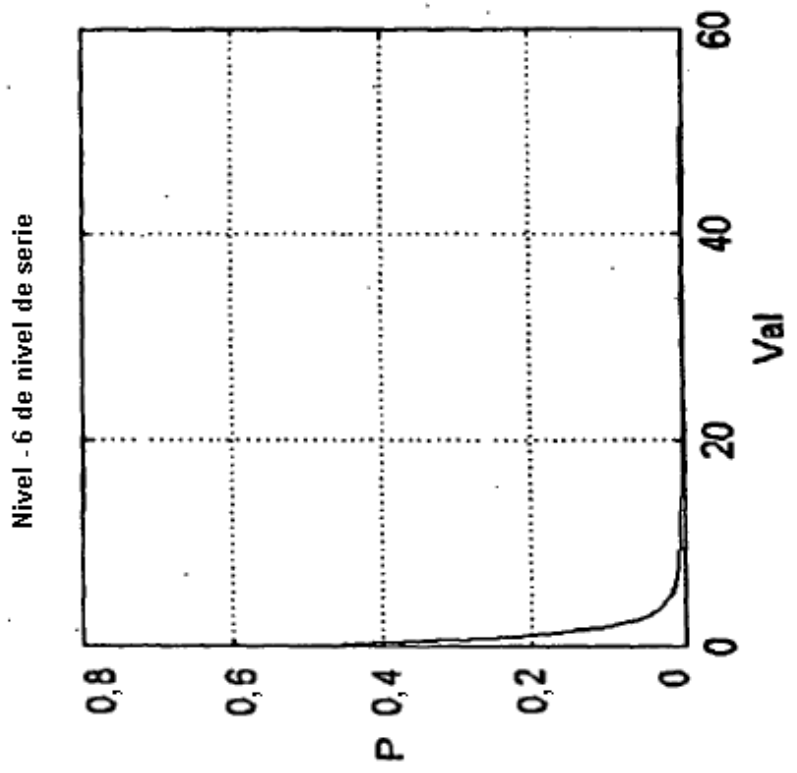


Figura 15H

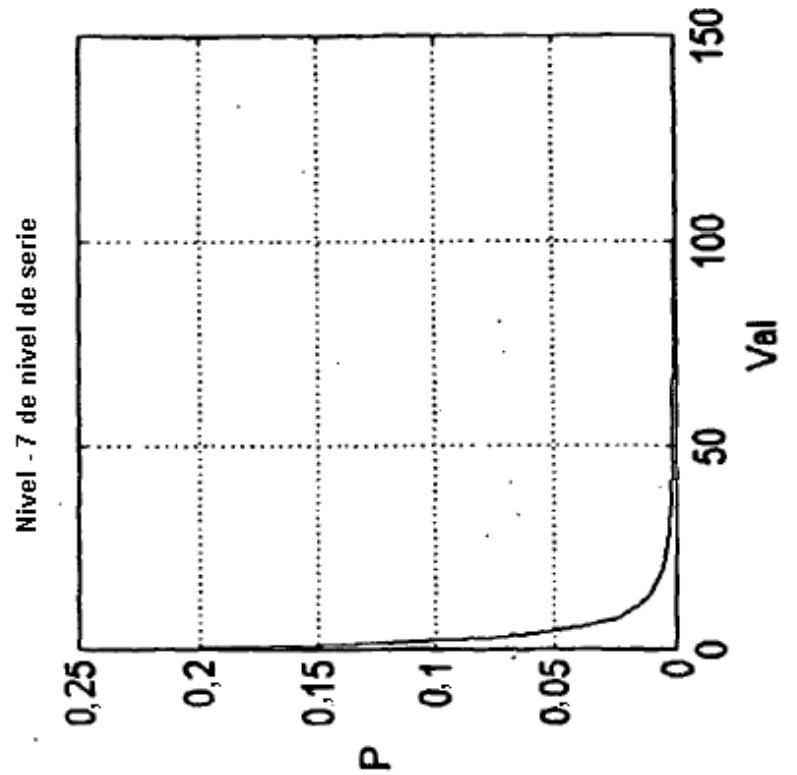


Figura 16

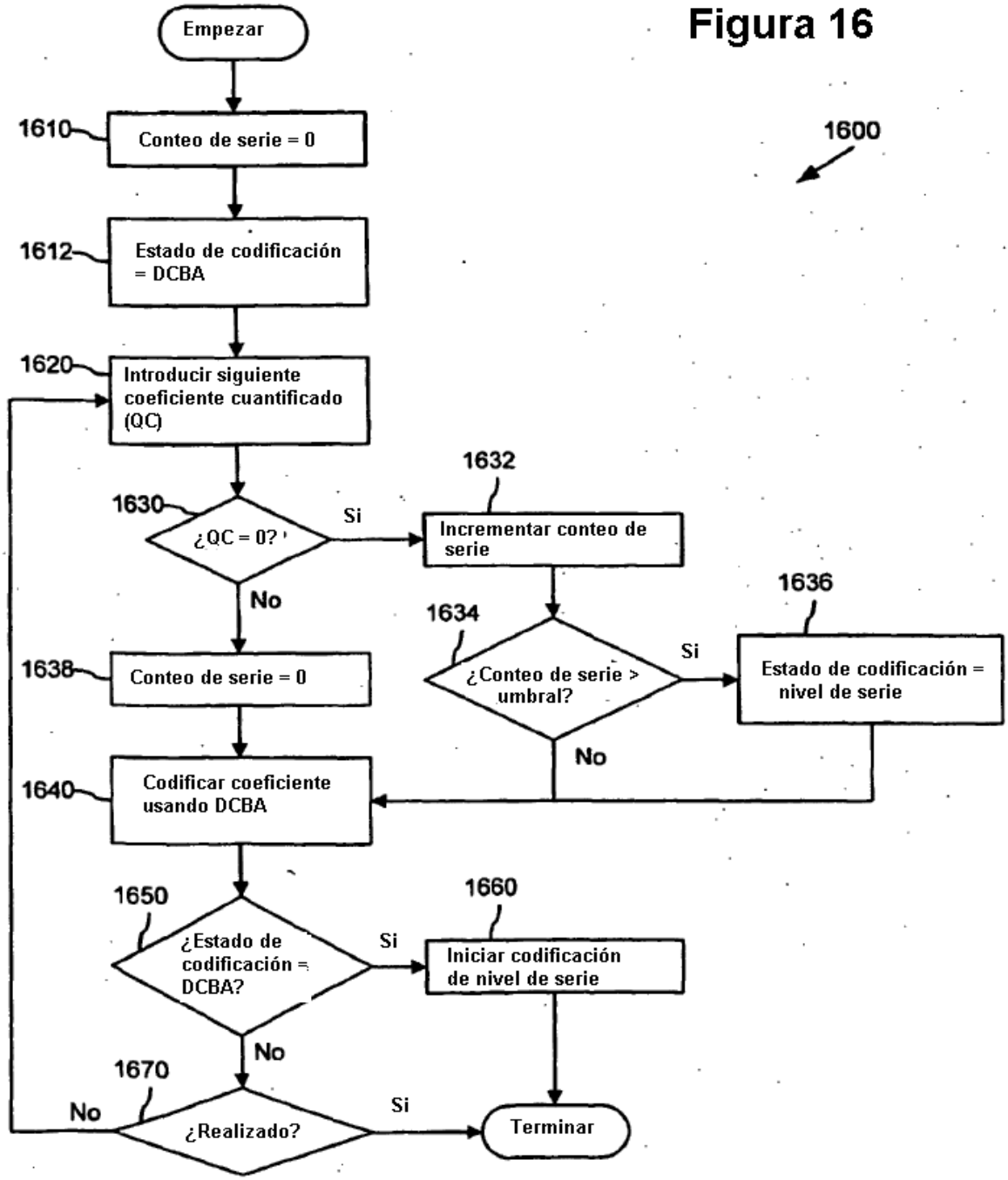
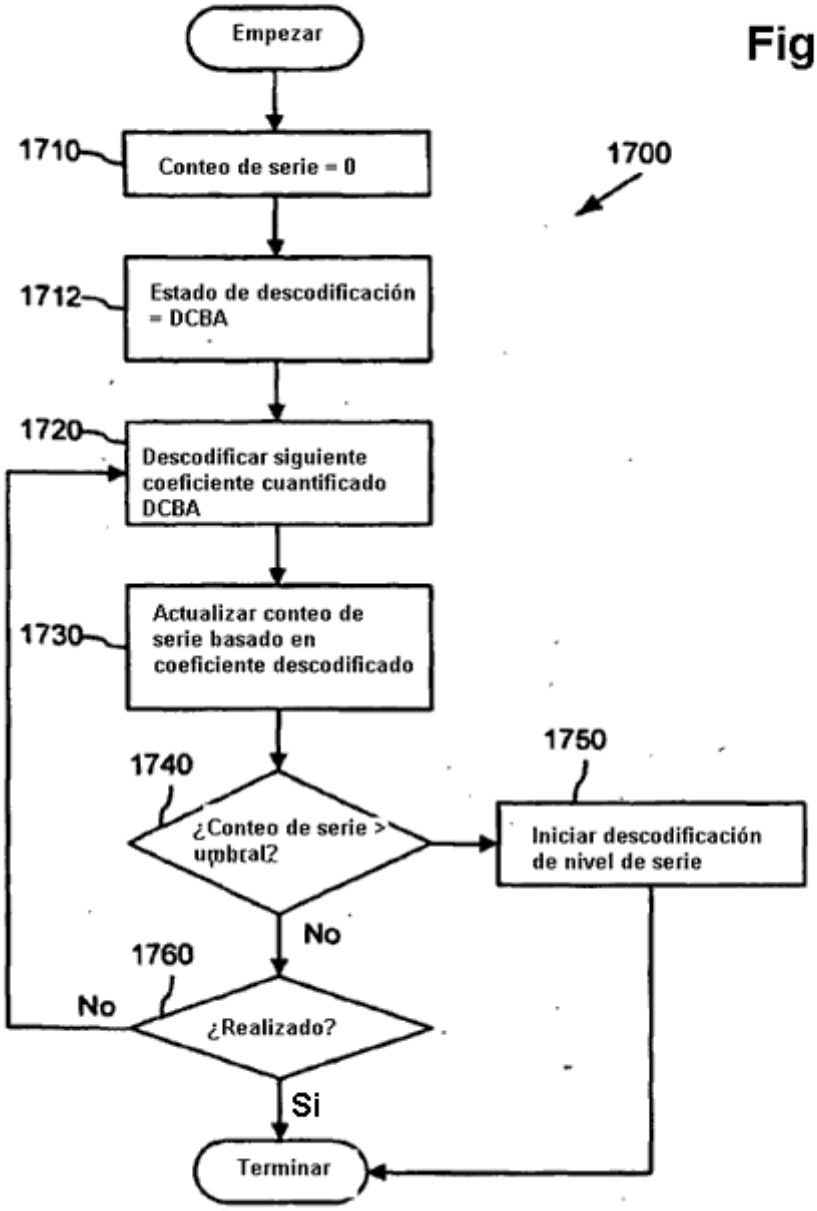

Empezar
1600
1610
Conteo de serie = 0
1612
Estado de codificación = DCBA
1620
Introducir siguiente coeficiente cuantificado (QC)
1630
¿QC = 0?
Si
1632
Incrementar conteo de serie
No
1634
¿Conteo de serie > umbral?
Si
1636
Estado de codificación = nivel de serie
No
1638
Conteo de serie = 0
1640
Codificar coeficiente usando DCBA
1650
¿Estado de codificación = DCBA?
Si
1660
Iniciar codificación de nivel de serie
No
1670
¿Realizado?
No
Si
Terminar

Figura 17


Empezar
1700
1710
Conteo de serie = 0
1712
Estado de descodificación = DCBA
1720
Descodificar siguiente coeficiente cuantificado DCBA
1730
Actualizar conteo de serie basado en coeficiente descodificado
1740
¿Conteo de serie > umbral2
1750
Iniciar descodificación de nivel de serie
No
1760
No
¿Realizado?
Si
Terminar