(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 731 424**

(51) Int. CI.:

G10L 19/02 (2013.01)
G10L 19/022 (2013.01)
H03M 7/30 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(96) Fecha de presentación y número de la solicitud europea: **18.05.2009 E 17190130 (9)**

(97) Fecha y número de publicación de la concesión europea: **24.04.2019 EP 3300076**

(54) Título: **Codificador de audio y decodificador de audio**

(30) Prioridad:

11.07.2008 US 79842
08.10.2008 US 103820

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
15.11.2019

(73) Titular/es:

FRAUNHOFER-GESELLSCHAFT ZUR FÖRDERUNG DER ANGEWANDTEN FORSCHUNG E.V. (100.0%)
Hansastraße 27c
80686 München, DE

(72) Inventor/es:

MULTRUS, MARKUS;
GRILL, BERNHARD;
FUCHS, GUILLAUME;
GEYERSBERGER, STEFAN;
RETTELBACH, NIKOLAUS y
BACIGALUPO, VIRGILIO

(74) Agente/Representante:

SALVÀ FERRER, Joan

DESCRIPCIÓN

Codificador de audio y decodificador de audio

[0001] La presente invención se refiere al campo de la codificación de audio, especialmente al campo de la codificación de entropía basada en el contexto. Los conceptos tradicionales de codificación de audio incluyen un esquema de codificación de entropía para reducción de la redundancia. Normalmente, la codificación de entropía se aplica a coeficientes espectrales cuantificados para esquemas de codificación basados en el dominio de la frecuencia o muestras de dominio del tiempo cuantificadas para esquemas de codificación basados en el dominio del tiempo. Estos esquemas de codificación de entropía hacen uso normalmente de la transmisión de una palabra de código en combinación con el índice de un libro de códigos consiguiente, lo que permite que un decodificador busque una determinada página del libro de códigos para decodificar una palabra de información codificada correspondiente a la palabra de código transmitida en dicha página. Sin embargo, en algunos conceptos de codificación, la transmisión del índice del libro de códigos puede no ser imprescindible, por ejemplo, en casos en que el índice del libro de códigos se puede determinar a partir del contexto de un símbolo, que por ejemplo se somete a codificación de entropía tal como se describe en Meine, Edler, “Improved Quantization and Lossless Coding for Subband Audio Coding” y en Meine, “Vektorquantisierung und kontextabhängige arithmetische Codierung für MPEG-4 AAC”, Dissertation, Gottfried Wilhelm Leibnitz Universität Hannover, Hanover 2007. Meine Nikolaus y col.: “Improved quantization and lossless coding for subband audio coding” describe una codificación de entropía de audio que usa un contexto de codificación.

[0002] En el caso de la codificación de audio basada en el dominio de la frecuencia o espectral, un contexto puede describir símbolos o propiedades estadísticas, por ejemplo, coeficientes espectrales cuantificados, que son anteriores en el tiempo y/o la frecuencia. En algunos de los conceptos convencionales, estos símbolos pueden estar disponibles tanto en el lado del codificador como en el del decodificador y, basándose en estos símbolos, se puede determinar un libro de códigos o contexto de forma síncrona tanto en el lado del codificador como en el del decodificador.

[0003] La fig. 9 ilustrará un ejemplo de contexto y sus dependencias. La fig. 9 muestra un plano de frecuencia-tiempo, en el que se ha indicado una serie de símbolos. El símbolo $S_{n,m}$ denota un símbolo en el tiempo n y la frecuencia m. La fig. 9 demuestra que, para codificar un determinado símbolo, se usa su contexto para determinar el libro de códigos asociado. Por ejemplo, en el caso del símbolo $S_{n0,m0}$ esto correspondería a todos los símbolos con
n < n0 y cualquier m, o con n = n0 y m < m0.

[0004] En implementaciones prácticas, un contexto no es infinito sino limitado. En el ejemplo ilustrado en la fig. 9, el contexto del símbolo $S_{0,3}$ podría ser, por ejemplo,
$S_{0,2}$, $S_{0,1}$, $S_{-1,5}$, $S_{-1,4}$, $S_{-1,3}$, $S_{-1,2}$, $S_{-1,1}$, $S_{-2,5}$, $S_{-2,4}$, $S_{-2,3}$, $S_{-2,2}$, $S_{-2,1}$.

[0005] En el caso de la codificación de audio basada en la frecuencia, se pueden usar, por ejemplo, variantes temporales, bancos de filtros de señales adaptativos o las llamadas transformaciones en bloque tal como se describe, por ejemplo, en Edler, B., “Codierung von Audiosignalen mit überlappender Transformation und adaptiven Fensterfunktionen”, Frequenz, Ausgabe 43, septiembre de 1989.

[0006] Dicho de otro modo, dentro de estos conceptos de codificación de audio se pueden producir, con el tiempo, cambios de resolución por frecuencia/tiempo. Un concepto de codificación de audio popular es el denominado AAC (Advanced Audio Coding, codificación de audio avanzada), en cuyo caso se usan dos longitudes de bloques, para lo cual se codifican, por ejemplo, 128 o 1.024 coeficientes transformados que representan los componentes de frecuencia de 256 o 2.048 muestras de dominio del tiempo en ventanas, respectivamente.

[0007] Estos conceptos permiten la conmutación entre diferentes resoluciones, dependiendo de ciertas características de las señales, como por ejemplo la aparición de transitorios o la tonalidad o si la señal es del tipo musical o de voz, etc. En el caso de la conmutación entre diferentes resoluciones de tiempo/frecuencia, como por ejemplo entre diferentes tipos de bloques AAC, el contexto no es estable. Los conceptos convencionales o las implementaciones del estado actual de la técnica pueden usar el reinicio del contexto, es decir, básicamente se conmuta a un estado en el que no se dispone de ningún contexto, en el que el contexto se construye comenzando de cero. Este enfoque puede dar resultados suficientemente satisfactorios, por ejemplo, en AAC, ya que garantiza al menos dos bloques largos u ocho bloques cortos continuos en una fila, en el que se puede suponer que la conmutación sólo tiene lugar esporádicamente.

[0008] Sin embargo, los conceptos convencionales que reinician el contexto en general no son óptimos en términos de eficiencia de codificación, ya que cada vez que se reinicia el contexto, la selección del libro de códigos subsiguiente se basa en valores que están destinados a ser soluciones como último recurso para contextos desconocidos. En general, en esos casos se seleccionan libros de códigos subóptimos. La desventaja en la eficiencia de codificación puede ser ínfima en casos en los cuales la conmutación sólo tiene lugar ocasionalmente.

Sin embargo, en una situación en la que se produce una conmutación más frecuente, esto lleva a una pérdida significativa de eficiencia de codificación. Por un lado, una conmutación más frecuente es muy conveniente en el caso de velocidades más bajas de datos/muestreo, ya que, especialmente en este caso, es conveniente una adaptación de la longitud de transformada para la señal característica. Por otro lado, la eficiencia de codificación se reduce significativamente con la conmutación frecuente.

[0009] El objetivo de la presente invención es proporcionar un concepto para conmutar entre diferentes longitudes de transformada en la codificación de audio, para proporcionar una eficiencia de codificación mejorada.

[0010] El objetivo se alcanza mediante el objeto de las reivindicaciones independientes.

[0011] La presente invención se basa en el hallazgo de que en la codificación basada en el contexto, como por ejemplo la codificación de entropía basada en el contexto, que se puede aplicar a diferentes resoluciones de tiempo/frecuencia, se puede usar un mecanismo de correspondencia de contexto, en el caso de que la resolución de tiempo/frecuencia cambie con el tiempo, para obtener de esa manera una eficiencia de codificación mejorada. Un hallazgo de la presente invención es que, cuando se conmuta entre diferentes resoluciones de tiempo o frecuencia, se pueden deducir contextos para los coeficientes que tienen las nuevas resoluciones a partir de los coeficientes que tienen las resoluciones antiguas. Un hallazgo de la presente invención es que, por ejemplo, se puede usar interpolación, extrapolación, muestreo descendente, submuestreo, sobremuestreo, etc., para la adaptación y/o deducción de contextos cuando se conmutan las resoluciones de tiempo/frecuencia en la codificación de audio.

[0012] Las realizaciones de la presente invención proporcionan un procedimiento de correspondencia, que establece la correspondencia entre la frecuencia o los coeficientes espectrales de un contexto almacenado, que se refiere a una resolución antigua, y la resolución de frecuencia de un contexto o una trama actuales. Dicho de otro modo, se puede usar la información de contexto anterior para la determinación del libro de códigos, es decir, para deducir nueva información de contexto. Las realizaciones pueden permitir, por esta razón, una conmutación más frecuente de longitudes de bloques y, por lo tanto, una mejor adaptación a las características de la señal sin perder eficiencia de codificación.

[0013] Las realizaciones de la presente invención se detallarán usando las figuras adjuntas, en las que

la fig. 1 muestra una realización de un codificador de audio;
la fig. 2 muestra una realización de un decodificador de audio;
la fig. 3 muestra una realización correspondiente a un sobremuestreo de contexto;
la fig. 4 muestra una realización correspondiente a un submuestreo de contexto;
la fig. 5 ilustra un audio con conmutación de resoluciones de tiempo y frecuencia;
la fig. 6 Ilustra una implementación de una realización;
la fig. 7a muestra un diagrama de flujo de una realización de un procedimiento de codificación;
la fig. 7b ilustra el procedimiento general de actualización de contextos según una realización;
la fig. 7c ilustra el procedimiento de actualización de contextos según una realización correspondiente a cambios de resolución;
la fig. 8 muestra un diagrama de flujo de una realización de un procedimiento de decodificación; y
la fig. 9 muestra un esquema de codificación de frecuencia y tiempo del estado actual de la técnica.

[0014] La fig. 1 ilustra una realización de un codificador de audio 100 para codificar segmentos de coeficientes, representando los segmentos de coeficientes diferentes resoluciones de tiempo o frecuencia de una señal de audio muestreada. El codificador de audio 100 comprende un procesador 110 para deducir un contexto de codificación para un coeficiente que se está codificando actualmente de un segmento basándose en un coeficiente codificado previamente de un segmento anterior, en el que el coeficiente codificado previamente representa una resolución diferente de tiempo o frecuencia que el coeficiente que se está codificando actualmente. La realización del codificador de audio comprende además un codificador de entropía 120 para codificación de entropía del coeficiente actual basándose en el contexto de codificación con el fin de obtener un tren de audio codificado.

[0015] En ciertas realizaciones, los coeficientes pueden corresponder a muestras de audio, muestras de audio cuantificadas, coeficientes espectrales o de frecuencia, coeficientes en escala, coeficientes transformados o filtrados, etc., o cualquier combinación de los mismos.

[0016] En algunas realizaciones el codificador de audio 100 puede comprender además un medio para proporcionar los segmentos de coeficientes de un tren de audio, en el que los coeficientes forman una representación espectral de una señal de audio en una resolución espectral que varía entre los coeficientes. El medio para proporcionar los segmentos puede estar adaptado para determinar los segmentos basándose en diferentes longitudes de ventana del dominio de tiempo o diferentes tramas de audio, es decir, tramas de audio que tienen diferentes longitudes o diferentes números de coeficientes por anchura de banda, es decir, una resolución espectral o de frecuencia diferente. El medio de obtención puede estar adaptado para determinar segmentos de 1.024 y 128 coeficientes de tiempo, frecuencia o espectrales.

[0017] En algunas realizaciones, el procesador 110 puede estar adaptado para deducir el contexto de codificación basándose en las representaciones de dominio de frecuencia o espectrales de los coeficientes o segmentos actuales y anteriores. Dicho de otro modo, en algunas realizaciones, los segmentos sucesivos pueden estar representados en diferentes dominios de tiempo y/o frecuencia o espectrales. El procesador 110 puede estar adaptado para deducir el contexto de codificación por banda de frecuencia o espectral del segmento actual, por ejemplo, basándose en los coeficientes espectrales adyacentes de segmentos anteriores y/o del segmento actual. En algunas realizaciones, los segmentos se pueden determinar en un principio en el dominio de tiempo, visualizando en ventana un tren de audio de entrada. Basándose en estos segmentos o coeficientes de dominio de tiempo, se pueden determinar segmentos o coeficientes de dominio de la frecuencia o espectral mediante transformación. Los segmentos se pueden representar en el dominio de frecuencia o espectral en términos de energía, amplitud y fase, amplitud y signo, etc., por cada banda de frecuencia o espectral, es decir, los segmentos se pueden subdividir en diferentes bandas de frecuencia o espectrales. El procesador 110 puede deducir entonces, en algunas realizaciones, contextos de codificación por cada banda de frecuencia o espectral.

[0018] El procesador 110 y el codificador de entropía 120 pueden estar configurados de manera que funcionen basándose en un submuestreo de los coeficientes de frecuencia o espectrales de un segmento anterior cuando un segmento anterior que pertenece al contexto comprende una resolución espectral o de frecuencia más fina que el segmento actual. En algunas realizaciones el procesador 110 y el codificador de entropía 120 pueden estar configurados de manera que funcionen basándose en un sobremuestreo de secuencias de los coeficientes de frecuencia o espectrales de un segmento anterior, cuando un segmento anterior que pertenece al contexto comprende una resolución espectral o de frecuencia menos fina que el segmento actual.

[0019] Las realizaciones pueden ofrecer un procedimiento para codificar segmentos de coeficientes que representan diferentes resoluciones de tiempo o frecuencia de una señal de audio muestreada. El procedimiento puede comprender una etapa de deducción de un contexto de codificación para un coeficiente actual o que se está codificando actualmente de un segmento actual basándose en un coeficiente anterior o codificado previamente de un segmento anterior y basándose también, de manera opcional, en el coeficiente actual o que se está codificando, en el que el coeficiente anterior o codificado previamente representa una resolución de tiempo o de frecuencia diferente a la del coeficiente actual o que se está codificando actualmente. El procedimiento puede comprender además una etapa de codificación de entropía del coeficiente actual basándose en el contexto de codificación para obtener un tren de audio codificado.

[0020] En consecuencia, las realizaciones pueden comprender un decodificador de audio 200 del cual se ilustra una realización en la fig. 2. El decodificador de audio 200 está adaptado para decodificar un tren de audio codificado para obtener segmentos de coeficientes que representan resoluciones de tiempo o frecuencia de una señal de audio muestreada; el decodificador de audio 200 comprende un procesador 210 para deducir un contexto de codificación para un coeficiente actual que se está decodificando actualmente basándose en un coeficiente anterior o decodificado previamente, en el que el coeficiente anterior o decodificado previamente representa una resolución de tiempo o frecuencia diferente de la del coeficiente que se está decodificando actualmente. Además, el decodificador de audio 200 comprende un decodificador de entropía 220 para decodificación de entropía del coeficiente actual basándose en el contexto de codificación y el tren de audio codificado.

[0021] En ciertas realizaciones, el decodificador de audio 200 puede comprender un decodificador de entropía 220, que está adaptado para determinar los segmentos de coeficientes decodificados basándose en diferentes longitudes de ventana de dominio de tiempo o diferentes longitudes de tramas de audio. El decodificador de entropía 220 puede estar adaptado para determinar segmentos, por ejemplo, de 1.024 y 128 muestras de dominio de tiempo o coeficientes de frecuencia o espectrales. En consecuencia, el procesador 210 puede estar adaptado para deducir el contexto de codificación basándose en la representación de dominio de frecuencia o espectral de los coeficientes de segmentos anteriores y/o del segmento actual.

[0022] En ciertas realizaciones, el procesador 210 puede estar adaptado para deducir el contexto de codificación por cada banda de frecuencia o espectral del segmento actual, por ejemplo, basándose en coeficientes espectrales adyacentes del segmento o segmentos anteriores y, opcionalmente, del segmento actual. Dicho de otro modo, se pueden procesar los segmentos en el dominio de la frecuencia o espectral, lo que se puede realizar por cada banda de frecuencia o espectral. Por consiguiente, el procesador 210 puede estar adaptado entonces para deducir un contexto que corresponde a una banda de frecuencia o espectral específica.

[0023] El decodificador de entropía 200 puede estar adaptado para codificación de entropía del coeficiente actual basándose en una regla de codificación de entropía o de longitud variable.

[0024] El procesador 210 puede estar adaptado para deducir el contexto de codificación basándose en el submuestreo de los coeficientes de frecuencia o espectrales de un segmento anterior cuando el segmento anterior comprende más coeficientes por anchura de banda (es decir, una resolución espectral o de frecuencia más fina) que el segmento actual. En realizaciones adicionales, el procesador 210 y el codificador de entropía 220 pueden estar

configurados para operar basándose en un sobremuestreo de secuencia de los coeficientes espectrales de un segmento anterior cuando el segmento anterior comprende menos coeficientes por anchura de banda (es decir, una resolución espectral o de frecuencia menos fina) que el segmento actual.

[0025] En consecuencia, las realizaciones pueden proporcionar un procedimiento para decodificar un tren de audio codificado con el fin de obtener segmentos de coeficientes que representan muestras de audio decodificadas. El procedimiento de decodificación puede comprender una etapa de deducción de un contexto de codificación correspondiente a un coeficiente actual o que se está decodificando actualmente de un segmento actual basándose en un coeficiente anterior o codificado previamente de un segmento anterior, en el que el coeficiente anterior o codificado previamente representa una resolución de tiempo o frecuencia diferente que en el coeficiente decodificado actualmente. Además, el procedimiento puede comprender una etapa de decodificación de entropía del coeficiente actual basándose en el contexto de codificación y el tren de audio codificado. Opcionalmente, el procedimiento puede comprender una etapa de determinación de los segmentos de coeficiente de audio codificados de un tren de audio codificado, en el que los segmentos representan diferentes números de coeficientes de audio.

[0026] La fig. 3 ilustra la manera en que un procesador 110; 210 puede deducir un contexto de codificación correspondiente a un segmento actual de coeficientes $M_{c,nuevos}$ basándose en un segmento anterior de coeficientes $M_{c,anteriores}$, en el que el segmento anterior comprende un número diferente de coeficientes de audio que el segmento actual. En la realización ilustrada en la fig. 3, el número de coeficientes del segmento M determina la resolución de frecuencia o espectral del segmento. La realización puede comprender un procedimiento de correspondencia, que establece la correspondencia entre los coeficientes $M_{c,anteriores}$ de un segmento anterior y los coeficientes $M_{c,nuevos}$ que tienen la misma resolución de frecuencia o espectral del contexto que el segmento actual. La fig. 3 muestra dos conjuntos de coeficientes dentro de dos segmentos, es decir, el segmento original anterior 310 que representa los coeficientes $M_{c,anteriores}$ $S_{n,0}$, $S_{n,1}$, $S_{n,2}$, etc., y, por consiguiente, el segmento anterior puesto en correspondencia 320, que tiene una resolución más alta, es decir, $M_{c,nuevo}$ es mayor que $M_{c,anterior}$, y representa los coeficientes $M_{c,nuevos}$ $S_{n,0}$, $S_{n,1}$, $S_{n,2}$, $S_{n,3}$, etc.

[0027] En general, se pueden distinguir dos realizaciones, dependiendo de si la resolución del contexto del segmento actual es más alta o baja que la resolución del contexto del segmento anterior. La fig. 3 ilustra una realización en la que la resolución del segmento anterior de coeficientes $M_{c,anteriores}$ es menor que la resolución del segmento actual de coeficientes $M_{c,nuevos}$. La fig. 3 muestra los coeficientes del segmento anterior 310 y los símbolos del segmento anterior puesto en correspondencia 320. De la fig. 3 se puede deducir que la resolución del segmento actual de coeficientes $M_{c,nuevos}$ es más alta que la resolución del segmento anterior 310 que sólo tiene coeficientes $M_{c,anteriores}$. En una realización el segmento anterior 310 se sobremuestrea en un segmento 320 de $M_{c,nuevos}$ coeficientes para que coincida con la resolución de frecuencia o espectral del segmento actual. Esto puede incluir el sobremuestreo puro con mecanismos de duplicación de símbolos y diezmado como, por ejemplo, la repetición de cada valor $M_{c,nuevo}$ veces antes de diezmar el segmento sobremuestreado manteniendo sólo 1 coeficiente por cada $M_{c,anterior}$. También se pueden usar otros mecanismos de interpolación o extrapolación.

[0028] En ciertas realizaciones, se puede llevar a cabo la correspondencia de todos los segmentos anteriores 310 que sean necesarios para determinar los contextos para el segmento actual, por ejemplo, en el instante n; dicho de otro modo, se pueden tener en cuenta múltiples segmentos anteriores, es decir, segmentos anteriores en los instantes n–1, n–2, etc. En general, las realizaciones pueden tener en cuenta múltiples intervalos de tiempo o segmentos anteriores, y el número de intervalos de tiempo necesarios para definir un contexto completo puede ser diferente en otras implementaciones o realizaciones.

[0029] La fig. 4 ilustra otra realización, en la que los coeficientes de un segmento anterior 410 se submuestrean hasta obtener un segmento 420 usado para calcular los contextos del segmento actual, es decir, en la que el número de coeficientes $M_{c,anteriores}$ del segmento anterior 410 es mayor que el número de coeficientes $M_{c,nuevos}$ del segmento actual. La fig. 4 usa una ilustración similar a la de la fig. 3, por consiguiente, se muestran múltiples coeficientes en cada segmento 410 y 420. Como se ilustra en la fig. 4, $M_{c,anterior}$ es mayor que $M_{c,nuevo}$. Por lo tanto, los coeficientes $M_{c,anteriores}$ se submuestrean para que coincidan con la resolución de frecuencia o espectral del segmento actual de $M_{c,nuevos}$ coeficientes, es decir, en ciertas realizaciones se pueden submuestrear los segmentos anteriores con una resolución mayor para que se correspondan con la resolución del segmento actual que tiene una resolución más baja. En algunas realizaciones esto puede incluir submuestreo puro con mecanismos de duplicación y diezmado como, por ejemplo, la repetición de cada valor $M_{c,nuevo}$ veces antes de diezmar el segmento sobremuestreado así obtenido manteniendo sólo 1 coeficiente por cada $M_{c,anterior}$. En otras realizaciones, se pueden tener en cuenta operaciones de filtro, como por ejemplo el promediado de dos o múltiples valores adyacentes.

[0030] La fig. 5 ilustra otra realización, en la que se lleva a cabo la conmutación entre diferentes resoluciones. La fig. 5 muestra un plano de tiempo/frecuencia, en el que se ilustran tres segmentos subsiguientes de coeficientes de audio, es decir, 510, 520 y 530. Cada uno de los segmentos 510, 520 y 530 corresponde a un único conjunto de coeficientes. En la realización ilustrada en la fig. 5, se supone que el segundo segmento 520 tiene una longitud doble que los segmentos primero y tercero 510 y 530. Esto se puede lograr usando diferentes ventanas durante la segmentación en el dominio del tiempo, como se hace, por ejemplo, en AAC. En la realización ilustrada en la fig. 5,

se supone que la velocidad de muestreo se mantiene constante; dicho de otro modo, el segundo segmento más largo 520 comprende el doble de coeficientes de audio por anchura de banda que el primer o el tercer segmento 510 o 530.

[0031] La fig. 5 muestra que en este caso la resolución se somete a escala con la extensión del segmento en el dominio del tiempo. Dicho de otro modo, cuanto más corta es la ventana en el dominio del tiempo, más baja es la resolución en el dominio de frecuencias o espectral. Cuando se evalúan los contextos para codificar las muestras en el dominio de frecuencias o espectral, la fig. 5 muestra que la codificación ha de tener una versión de resolución más elevada del segmento 510 cuando se codifica el segundo segmento 520 como en el ejemplo considerado, se debe deducir una resolución doble del segmento 510. En otras realizaciones, especialmente cuando se usan otras transformaciones o bancos de filtro de tiempo–frecuencia, se pueden producir otras relaciones entre las resoluciones del dominio del tiempo y de la frecuencia.

[0032] Según una realización, los coeficientes codificados durante el primer segmento 510 proporcionan una base para determinar el contexto para el segundo segmento 520, por ejemplo, por medio de un sobremuestreo intermedio. Dicho de otro modo, el contenido del contexto procedente del primer segmento 510 se puede obtener mediante el sobremuestreo del primer segmento 510, por ejemplo, en términos de interpolación o extrapolación, para deducir el contexto del segundo segmento 520, que tiene una resolución más alta.

[0033] Como se ilustra en la fig. 5, al conmutar del segundo segmento 520 al tercer segmento 530, el elemento constituyente del contexto tiene también que cambiar, ya que la resolución se ha reducido. Según una realización, se pueden usar los coeficientes codificados durante el segundo segmento 520 para deducir el contexto para el tercer segmento, por medio de un submuestreo intermedio. Esto se puede llevar a cabo, por ejemplo, en términos de promediado o simplemente usando un único valor por segundo u otras medidas para el submuestreo.

[0034] Las realizaciones otorgan la ventaja de una mayor eficiencia de codificación, teniendo en cuenta el contexto pasado deducido de segmentos anteriores cuando haya cambios de resolución o de longitud de ventana. Los elementos constituyentes del contexto se pueden adaptar a nuevas resoluciones, en términos de sobremuestreo o submuestreo, por ejemplo, con interpolación o extrapolación, filtrado o promediado, etc.

[0035] A continuación se presenta una realización más específica en términos de codificación espectral sin ruido. La codificación espectral sin ruido se puede usar para reducir aún más la redundancia de un espectro cuantificado en la codificación de audio. La codificación espectral sin ruido se puede basar en una codificación aritmética juntamente con una adaptación dinámica del contexto.

[0036] La codificación sin ruido se puede basar en valores espectrales cuantificados y puede usar tablas de frecuencia acumulada dependiente del contexto deducidas, por ejemplo, a partir de cuatro tuplas adyacentes decodificadas previamente. La fig. 6 ilustra otra realización. La fig. 6 muestra un plano de tiempo-frecuencia, en el que a lo largo del eje tiempo hay tres intervalos de tiempo indexados n, n–1 y n–2. Además, la fig. 6 ilustra cuatro bandas de frecuencia o espectrales que llevan la designación m–2, m–1, m y m+1. La fig. 6 muestra el interior de cada recuadro de intervalo de tiempo–frecuencia, que representa tuplas de muestras para codificar o decodificar. En la fig. 6 se ilustran tres tipos de tuplas diferentes, y los recuadros con borde sombreado o en línea discontinua indican las tuplas restantes que se han de codificar o decodificar, los recuadros blancos con un borde en línea continua indican las tuplas ya codificadas o decodificadas y los recuadros grises con borde en línea continua indican las tuplas codificadas/decodificadas previamente que se usan para determinar el contexto para la tupla actual que se va a codificar o a decodificar.

[0037] Debe observarse que los segmentos anteriores y actuales a los que se hace referencia en las realizaciones descritas previamente pueden corresponder a una tupla en la presente realización, dicho de otro modo, los segmentos pueden ser procesados en bandas en el dominio de la frecuencia o espectral. Como se ilustra en la fig. 6, se pueden tener en cuenta las tuplas o segmentos en las zonas adyacentes a una tupla actual (es decir, en el dominio de tiempo y en el dominio de frecuencias o espectral) para deducir un contexto. El codificador aritmético puede usar entonces tablas de frecuencias acumuladas para generar un código binario de longitud variable. El codificador aritmético puede producir un código binario para un conjunto de símbolos dado y sus probabilidades respectivas. El código binario se puede generar estableciendo la correspondencia entre un intervalo de probabilidades, en el que se encuentra el conjunto de símbolos, y una palabra de código. El codificador aritmético puede corresponder al codificador de entropía 120, o respectivamente al decodificador de entropía 220, de las realizaciones descritas previamente.

[0038] En la presente realización, la codificación aritmética basada en el contexto se puede llevar a cabo basándose en 4 tuplas (es decir, cuatro índices de coeficientes espectrales) a las que también se designa como q(n,m), que representan los coeficientes espectrales después de la cuantificación, que son adyacentes en el dominio de la frecuencia o espectral y que se someten a codificación de entropía en una etapa. Según la descripción anterior, la codificación se puede llevar a cabo basándose en el contexto de codificación. Como se indica en la fig. 6, además de la 4–tupla, que se codifica (es decir, el segmento actual), se tienen en cuenta cuatro 4–tuplas codificadas

previamente para deducir el contexto. Estas cuatro 4–tuplas determinan el contexto y son anteriores en el dominio de la frecuencia y/o anteriores en el dominio del tiempo.

[0039] La fig. 7a muestra un diagrama de flujo de un codificador USAC (Universal Speech y Audio Coder, codificador universal de voz y audio) dependiente del contexto para codificar el esquema de coeficientes espectrales. El procedimiento de codificación depende de la 4–tupla actual más el contexto, en el que el contexto se usa para seleccionar la distribución de probabilidades del codificador aritmético y para predecir la amplitud de los coeficientes espectrales. En la fig. 7a el recuadro 705 representa la determinación del contexto, que se basa en t0, t1, t2 y t3 que corresponden a q(n–1, m), q(n,m–1), q (n–1,m–1) y q (n–1,m+1), es decir, los recuadros grises con bordes en línea continua de la fig. 6.

[0040] En general, en algunas realizaciones el codificador de entropía puede estar adaptado para codificar el segmento actual en unidades de una 4–tupla de coeficientes espectrales y para predecir un intervalo de amplitudes de la 4–tupla basándose en el contexto de codificación.

[0041] En la presente realización el esquema de codificación comprende varias fases. En primer lugar, se codifica una palabra de código literal usando un codificador aritmético y una distribución de probabilidades específica. La palabra de código representa cuatro coeficientes espectrales adyacentes (a,b,c,d), sin embargo, cada uno de a, b, c, d está limitado en el intervalo:
$-5 < a,b,c,d < 4$.

[0042] En general, en ciertas realizaciones el codificador de entropía 120 puede estar adaptado para dividir la 4–tupla por un factor predeterminado el número de veces necesario para ajustar un resultado de la división en el intervalo previsto o en un intervalo predeterminado y para codificar un número de divisiones necesario, un resto de la división y el resultado de la división cuando la 4–tupla no se encuentra dentro del intervalo estimado y para codificar un resto de la división y el resultado de la división de otro modo.

[0043] En lo sucesivo, si el término (a,b,c,d), es decir, cualquier coeficiente a, b, c, d, supera el intervalo dado en la presente realización, esto se puede considerar en general dividiendo (a,b,c,d) por un factor (por ejemplo 2 o 4) el número de veces que sea necesario, para ajustar la palabra de código así obtenida al intervalo dado. La división por un factor de 2 corresponde a un desplazamiento binario hacia la derecha, es decir, (a,b,c,d)>> 1. Esta disminución se realiza en una representación por números enteros, es decir, se puede perder información. Los bits menos significativos, que se pueden perder por el desplazamiento a la derecha, son almacenados y más adelante se codifican usando el codificador aritmético y una distribución de probabilidades uniforme. El procedimiento de desplazamiento a la derecha se lleva a cabo para los cuatro coeficientes espectrales (a,b,c,d).

[0044] En las realizaciones generales, el codificador de entropía 120 puede estar adaptado para codificar el resultado de la división de la 4–tupla usando un índice de grupo ng, en el que el índice de grupo ng se refiere a un grupo de una o más palabras de código para las cuales la distribución de probabilidades se basa en el contexto de codificación, y un índice de elemento ne en el caso en que el grupo comprenda más de una palabra de código, en el que el índice de elemento ne se refiere a una palabra de código dentro del grupo y se puede suponer que el índice de elemento se distribuye de manera uniforme, y para codificar el número de divisiones por un número de símbolos de escape, en el que un símbolo de escape es un índice de grupo específico ng que sólo se usa para indicar una división y para codificar los restos de las divisiones basándose en una distribución uniforme usando una regla de codificación aritmética. El codificador de entropía 120 puede estar adaptado para codificar una secuencia de símbolos en el tren de audio codificado usando un alfabeto de símbolos que comprende el símbolo de escape y símbolos de grupo que corresponden a un conjunto de índices de grupo disponibles, un alfabeto de símbolos que comprende índices de elementos y un alfabeto de símbolos que comprende los diferentes valores de los restos.

[0045] En la realización de la fig. 7a, se puede deducir del contexto la distribución de probabilidades para la codificación de la palabra de código literal y también una estimación del número de etapas de reducción de intervalo. Por ejemplo, todas las palabras de código, en un total de $8^4 = 4.096$, se extienden a un total de 544 grupos, que consisten en uno o más elementos. La palabra de código puede estar representada en el tren de bits en forma del índice de grupo ng y el elemento de grupo ne. Los dos valores pueden ser codificados usando el codificador aritmético, usando ciertas distribuciones de probabilidades. En una realización la distribución de probabilidades correspondiente a ng puede deducirse del contexto, en tanto que se puede suponer que la distribución de probabilidades correspondiente a ne es uniforme. Una combinación de ng y ne puede identificar una palabra de código de manera inequívoca. Se puede suponer que el resto de la división, es decir, los planos de bits desplazados, se distribuyen también de manera uniforme.

[0046] En la fig. 7a, en la etapa 710, se presenta la 4–tupla q(n,m), que consiste en (a,b,c,d) o el segmento actual y se inicia un parámetro lev ajustándolo a 0.

[0047] En la etapa 715 se estima el intervalo de (a,b,c,d) a partir del contexto. Según esta estimación, (a,b,c,d) se puede reducir en lev0 niveles, es decir, dividirse por un factor de 2^{lev0}. Los planos de bits lev0 menos

significativos se almacenan para su uso posterior en la etapa 750.

[0048] En la etapa 720 se verifica si (a,b,c,d) supera el intervalo dado y, en tal caso, se reduce el intervalo de (a,b,c,d) en un factor 4 en la etapa 725. Dicho de otro modo, en la etapa 725 (a,b,c,d) se desplazan 2 a la derecha y los planos de bits suprimidos se almacenan para su uso posterior en la etapa 750.

[0049] Para indicar esta etapa de reducción, se ajusta ng a 544 en la etapa 730, es decir, ng = 544 sirve como palabra de código de escape. A continuación, se escribe esta palabra de código en el tren de bits de la etapa 755, en el que para deducir la palabra de código en la etapa 730 se usa un codificador aritmético con una distribución de probabilidades deducida del contexto. Cuando se aplica esta etapa de reducción la primera vez, es decir, si lev==lev0, se adapta ligeramente el contexto. Cuando la etapa de reducción se aplica más de una vez, se descarta el contexto y se usa en lo sucesivo una distribución por defecto. El procedimiento continúa seguidamente con la etapa 720.

[0050] Si en la etapa 720 se detecta una correspondencia de intervalos, más específicamente si (a,b,c,d) se corresponde con la condición del intervalo, se establece la correspondencia (a,b,c,d) con respecto a un grupo ng, y, si pudiera aplicarse, el índice de elemento de grupo ne. Esta correspondencia es inequívoca, es decir, (a,b,c,d) se puede deducir de ng y ne. A continuación, el índice de grupo ng es codificado por un codificador aritmético usando una distribución de probabilidades obtenida para el contexto adaptado/descartado en la etapa 735. Seguidamente se inserta el índice de grupo ng en el tren de bits en la etapa 755. En una etapa posterior 740 se verifica si el número de elementos de un grupo es superior a 1. Si es necesario, es decir, si el grupo indexado por ng consiste en más de un elemento, el índice de elemento de grupo ne es codificado por el codificador aritmético en la etapa 745, suponiendo una distribución de probabilidades uniforme en la presente realización.

[0051] Después de la etapa 745, se inserta el índice de elementos de grupo ne en el tren de bits en la etapa 755. Por último, en la etapa 750, se codifican todos los planos de bits almacenados usando el codificador aritmético suponiendo una distribución de probabilidades uniforme en la etapa 755.

[0052] En ciertas realizaciones el decodificador de entropía 220 puede estar adaptado para decodificar un índice de grupo ng del tren de audio codificado basándose en una distribución de probabilidades deducida del contexto de codificación, en el que el índice de grupo ng representa un grupo de una o más palabras de código y para decodificar, basándose en una distribución de probabilidades uniforme, un índice de elementos ne del tren de audio codificado si el índice de grupo ng indica un grupo que comprende más de una palabra de código y para deducir una 4–tupla de coeficientes espectrales del segmento actual basándose en el índice de grupo ng y el índice de elemento ne, para obtener así la representación de dominios espectrales en tuplas de coeficientes espectrales.

[0053] En ciertas realizaciones el decodificador de entropía 220 puede estar adaptado para decodificar una secuencia de símbolos correspondientes al tren de audio codificado basándose en la distribución de probabilidades deducida del contexto de codificación usando un alfabeto de símbolos que comprende un símbolo de escape y símbolos de grupo correspondientes a una serie de índices de grupo disponibles ng, para deducir una 4–tupla preliminar de coeficientes espectrales basándose en un índice de grupo ng disponible al cual corresponde un símbolo de grupo de la secuencia de símbolos y basándose en el índice de elemento ne y para multiplicar la 4–tupla preliminar por un factor que depende del número de símbolos de escape en la secuencia de símbolos con el fin de obtener la tupla de coeficientes espectrales.

[0054] El decodificador de entropía 220 puede estar adaptado asimismo para decodificar un resto del tren de audio codificado basándose en una distribución uniforme de probabilidades usando una regla de codificación aritmética y para añadir el resto a la 4–tupla preliminar multiplicada para obtener la 4–tupla de coeficientes espectrales.

[0055] El decodificador de entropía 220 puede estar adaptado para multiplicar la 4–tupla por un factor predeterminado siempre que un símbolo de escape sea decodificado a partir del tren de audio codificado, en el que un símbolo de escape es un índice de grupo específico ng que sólo se usa para indicar una multiplicación y para decodificar un resto de un tren de audio codificado basándose en una distribución uniforme de probabilidades usando una regla de codificación aritmética, el decodificador de entropía 220 puede estar adaptado además para añadir el resto a la 4–tupla multiplicada para obtener el segmento actual.

[0056] A continuación, se describe una realización de un esquema de decodificación por codificador aritmético dependiente del contexto USAC. Como corresponde a la realización anterior del esquema de codificación, se consideran las 4–tuplas que corresponden a coeficientes espectrales cuantificados, que son codificados sin ruido. Además, se supone que las 4–tuplas se transmiten a partir del coeficiente de frecuencia o espectral más bajo y en progresión hasta el coeficiente de frecuencia o espectral más elevado. Los coeficientes pueden corresponder, por ejemplo, a coeficientes AAC, que se almacenan en una matriz y se supone que el orden de transmisión de las palabras de código sin ruido es tal que se decodifican en el orden recibido y se almacenen en la matriz, bin es el índice de incremento más rápido y g es el índice con incremento más lento. Dentro de una palabra de código, el

orden de decodificación es a,b,c,d.

[0057] La fig. 7b ilustra el procedimiento de actualización del contexto general según una realización. En la presente realización se consideran detalles relativos a la adaptación al contexto según un mecanismo de predicción de profundidad de bits. La fig. 7b ilustra un plano 760 que muestra el posible intervalo de una 4–tupla (a,b,c,d) en términos de planos de bits. Se puede predecir la profundidad de bits, es decir, el número de planos de bits necesario para representar una 4–tupla, por el contexto de la 4–tupla actual por medio del cálculo de la variable denominada lev0, que también está indicada en la fig. 7b. A continuación, se divide la 4–tupla por 2^{lev0}, es decir, se suprimen lev=lev0 planos de bits y se almacenan para su uso posterior según la etapa 715 descrita previamente.

[0058] Si la 4–tupla está en el intervalo –5<a,b,c,d<4, la profundidad de bits predicha lev0 ha sido correctamente predicha o sobreestimada. La 4–tupla puede ser codificada entonces por el índice de grupo ng, el índice de elementos ne y los lev planos de bits restantes, en línea con la descripción anterior. Finaliza entonces la codificación de la 4–tupla actual. La codificación del índice de elemento ne está indicada en la fig. 7b por la distribución uniforme de probabilidades 762, que en lo sucesivo se usa siempre para codificar índices de elementos, de manera que, en la fig. 7b, el parámetro r representa el resto de la 4–tupla después de la división y p(r) representa la función de densidad de probabilidad uniforme correspondiente.

[0059] Si la 4–tupla no está en el intervalo –5<a,b,c,d<4 la predicción basada en el contexto de codificación 764 es demasiado baja, se codifica un símbolo de escape (ng=544) 766 y la 4–tupla se divide por 4 y reincrementa lev en 2, según la etapa 730 en la fig. 7a. El contexto se adapta de la siguiente manera: si lev==lev0+2 el contexto se adapta levemente, lo que corresponde a 768 en la fig. 7b. Es posible establecer un indicador en la representación del contexto t, y a continuación se usa un nuevo modelo de distribución de probabilidades para codificar los símbolos ng futuros.

[0060] Si lev>lev0+2 se codifica otro símbolo de escape según la etapa 770 de la fig. 7b y el contexto se reinicia por completo, véase 772, o se descarta como en la etapa 730 de la fig. 7a, respectivamente. No se usa ninguna adaptación más del contexto porque se considera no relevante para la codificación de la 4–tupla actual. Se usa entonces el modelo de probabilidades por defecto, el usado cuando no se disponía de ningún contexto, para los futuros símbolos ng, lo que está indicado por las etapas 774 y 776 en la fig. 7b. A continuación, se repite el procedimiento para otras tuplas.

[0061] Para resumir, la adaptación del contexto es un mecanismo que persigue reducir la significación del contexto en la codificación adaptable al contexto. La adaptación al contexto se puede activar cuando el lev0 predicho y el lev real no coinciden. Este hecho se detecta fácilmente por el número de símbolos de escape codificados (ng=544), en comparación con 766 y 770 en la fig. 7b, y por lo tanto también se puede llevar a cabo en el decodificador de manera similar.

[0062] La adaptación del contexto se puede realizar activando un indicador en la representación de estado del contexto t. El valor t se calcula por la función get state() (adquirir estado), en forma de lev0, usando el contexto deducido de la trama o segmento anterior y/o actual de la 4–tupla actual, que se almacena en una tabla q[][]. El estado del contexto puede estar representado, por ejemplo, por 24 bits. En una realización hay 1.905.800 estados posibles. Estos estados pueden representarse con sólo 21 bits. Los bits 23º y 24º de t están reservados para adaptar el estado del contexto. Según los valores de los bits 23º y 24º, get_pk() produce diferentes modelos de distribución de probabilidades. En una realización, el bit 23º de t se puede ajustar a uno cuando la 4–tupla se divide por 4 después de haber sido dividida previamente por lev0, es decir, lev==lev0+2.

[0063] En consecuencia, la correspondencia entre el estado del contexto t y el modelo de distribución de probabilidades pki es diferente para lev==lev0+2 que para lev==lev0. La correspondencia entre el estado del contexto t y el modelo pki se predefine durante una fase de entrenamiento realizando optimizaciones en las estadísticas generales de la secuencia de entrenamiento. Cuando lev>lev0+2, se deben ajustar a cero el contexto y t. Get_pk() produce a continuación el modelo pki de distribución de probabilidades por defecto, que corresponde a t=0.

[0064] A continuación, se describen los detalles de una correspondencia de contexto según una realización. La correspondencia de contexto es la primera operación que se realiza en la codificación adaptable al contexto después del reinicio eventual del contexto según la presente realización. Esto se realiza en dos etapas.

[0065] En primer lugar, antes de la codificación, se establece la correspondencia entre la tabla de contextos qs[] del tamaño previous_lg/4, almacenada en la trama anterior, en una tabla de contextos q[0][] del tamaño lg/4 correspondiente al tamaño de la trama actual. La correspondencia se lleva a cabo en la función arith_map_context (correspondencia aritmética contexto), que se ilustra en el siguiente seudocódigo:

```
/* variable de entrada */
lg/4 /* número of 4–tuplas */
```

```
arith_map_context()
{
        v=w=0
if (core_mode==1) {
        q[0][v++]=qs[w++];
}

ratio= ((float)previous_lg)/((float)lg);
for (j=0; j<lg/4; j++) {
        k = (int) ((float)) ((j)*ratio);
        q[0][v++] = qs[w+k];
}

if(core_mode==0) {
        q[0][lg/4]=qs[previous_lg/4];
}
q[0][lg/4+1]=qs[previous_lg/4+1];

previous_lg=lg; }
```

[0066] Como se puede observar en el seudocódigo, la correspondencia puede no ser exactamente igual para todas las estrategias de codificación. En la presente realización, la correspondencia difiere cuando se usa AAC (Advanced Audio Coding, codificación avanzada de audio) (core_mode==0) para un coeficiente obtenido cuando se usa TCX (Transform based Coding, codificación basada en transformadas) (core_mode==1). Una de las diferencias proviene de la manera en que se tratan los límites de las tablas. En AAC, la correspondencia puede iniciarse a partir del índice 0 (primer valor de la tabla), en tanto que en el caso de TCX se puede iniciar desde el índice 1 (segundo valor de la tabla) sabiendo que el primer valor siempre se ajusta como “desconocido” (estado específico empleado para reiniciar el contexto). La ratio de previous_lg con respecto a lg determina el orden del sobremuestreo (ratio < 1) o el submuestreo (ratio > 1), que se realiza en la presente realización. La fig. 7c ilustra el caso de TCX cuando se convierte a partir de una tabla de contextos almacenada de un tamaño 1.024/4, como en el lado izquierdo 780 de la fig. 7c, a un tamaño de 512/4, como en el lado derecho 782 de la fig. 7c. Se puede observar que, si bien en la tabla de contextos actual 782 se usan incrementos en pasos de 1, para la tabla de contextos almacenada 780 se usan incrementos por pasos según la ratio descrita previamente.

[0067] La fig. 7c ilustra el procedimiento de actualización de contexto de una realización para los cambios de resolución. Una vez realizada la correspondencia, se ejecuta la codificación adaptable al contexto. Al final de la codificación, se almacenan los elementos de la trama actual en la tabla qs[] para la siguiente trama. Esto se puede hacer en arith_update_context(), que se ilustra mediante el seudocódigo:

```
/* variables de entrada */
a,b,c,d /* valor de la 4–tupla decodificada */
i /* el índice de la 4–tupla para decodificación en el vector */
lg/4 /* número de 4–tuplas */
arith_update_context()
{
        q[1][1+i].a=a;
        q[1][1+i].b=b;
        q[1][1+i].c=c;
        q[1][1+i].d=d;
        if ( (a<–4) || (a>=4) || (b<–4) || (b>=4) || (c<–4) || (c>=4) || (d<–4) || (d>=4) ) {
        q[1][1+i].v =1024;
}
else q[1][1+i].v=egroups[4+a][4+b][4+c][4+d];

if(i==lg/4 && core_mode==1) {
        qs[0]=q[1][0];
        ratio= ((float) lg)/((float)1024);
        for (j=0; j<256; j++) {
                k = (int) ((float) j*ratio);
                qs[1+k] = q[1][1+j];
        }
        qs[previous_lg/4+1] = q[1][lg/4+1];
        previous_lg = 1024;
}
if(i==lg/4 && core_mode==0) {
```

```
                for (j=0; j<258; j++) {
                qs[j] = q[1][k];
                }
                previous_lg = min(1024,lg);
                }
}
```

[0068] En la presente realización, el almacenamiento se realiza de una manera diferente según el codificador de núcleo (AAC o TCX). En TCX el contexto se almacena siempre en la tabla qs[] de 1.024/4 valores. Esta correspondencia adicional se puede realizar en virtud de la decisión de bucle cerrado del AMR–WB+ (Adaptive Multirate WideBand Codec, códec adaptativo multivelocidad en banda ancha). En la decisión de bucle cerrado se necesitan varios procedimientos de copia de los estados del codificador para analizar cada combinación posible de TCX y ACELP (Arithmetic Coded Excited Linear Prediction, predicción lineal de salida por codificación aritmética). La copia del estado es más fácil de implementar cuando todos los modos TCX comparten el mismo tamaño de la tabla qs[]. En tal caso se usa una correspondencia para convertir sistemáticamente de lg/4 a 1.024/4. Por otra parte, AAC almacena sólo el contexto y no realiza correspondencias durante esta fase.

[0069] La fig. 8 ilustra un diagrama de flujo de la realización correspondiente al esquema de decodificación. En la etapa 805, que corresponde a la etapa 705, el contexto se deduce basándose en t0, t1, t2 y t3. En la etapa 810, se estima el primer nivel de reducción lev0 del contexto y la variable lev se ajusta a lev0. En la etapa siguiente 815, el ng del grupo se lee a partir del tren de bits y la distribución de probabilidades del contexto para la decodificación de ng se deduce del contexto. A continuación, en la etapa 815, se puede decodificar el ng del grupo del tren de bits.

[0070] En la etapa 820 se determina si el ng es igual a 544, que corresponde al valor escape. Si es así, se puede incrementar la variable lev en 2 antes de volver a la etapa 815. Si se usa esta ramificación por primera vez, es decir, si lev==lev0, la distribución de probabilidades, y respectivamente el contexto, se puede adaptar de manera correspondiente, o descartarse respectivamente si la ramificación no se usa por primera vez en línea con el mecanismo de adaptación del contexto descrito previamente como ocurre en las fig. 7b y 7c. En el caso en que el índice de grupo ng no es igual a 544 en la etapa 820, en una etapa posterior 825 se determina si el número de elementos de un grupo es mayor que 1, y si es así, en la etapa 830 se lee el elemento del grupo ne y se decodifica a partir de la secuencia de datos suponiendo una distribución uniforme de las probabilidades. El índice de elemento ne se deduce del tren de bits empleando codificación aritmética y una distribución de probabilidades uniforme.

[0071] En la etapa 835 se deduce la palabra de código (a,b,c,d) a partir de ng y ne, por ejemplo, por medio de un procedimiento de búsqueda en las tablas, por ejemplo, con referencia a dgroups[ng] y acod_ne[ne].

[0072] En la etapa 840 correspondiente a los planos de bit de todos los lev, los planos se leen a partir del tren de bits usando codificación aritmética y suponiendo una distribución de probabilidades uniforme. Seguidamente se pueden adjuntar los planos de bits a (a,b,c,d) desplazando (a,b,c,d) a la izquierda y añadiendo el plano de bits bp: ((a,b,c,d)<<=1)|=bp. Este procedimiento se puede repetir lev veces.

[0073] Por último, en la etapa 845 se puede producir la 4–tupla q(n,m), es decir, (a,b,c,d).

[0074] A continuación, se presentan detalles de seudocódigos e implementación según una realización. Se usan las siguientes definiciones.
(a,b,c,d) 4–tupla para decodificar

ng índice de grupo del plano de 2 bits más significativo de la 4–tupla, en el que 0 <= ng <= 544. Este último valor 544 corresponde al símbolo de escape, ARITH_ESCAPE.

ne Índice de elemento dentro de un grupo. ne se encuentra entre 0 y el cardinal de cada grupo mm. El número máximo de elementos dentro de un grupo es 73.

lev Nivel de los planos de bits restantes. Corresponde al número de planos de bits menos significativos que el plano de 2 bits más significativo.

Egroups [a][b][c][d] Tabla de índice de grupo. Permite establecer la correspondencia del plano de 2 bits más significativo de la 4–tupla (a,b,c,d) para los 544 grupos.

mm Cardinal del grupo

og Desplazamiento del grupo

dgroups[ ] Establece la correspondencia entre el índice del grupo ng con el cardinal de cada grupo mm (primeros 8

bits) y el desplazamiento del grupo og en dgvectors[] (últimos 8 bits).

dgvectors[ ] Establecen la correspondencia entre el desplazamiento del grupo og y el índice del elemento ne con el plano de 2 bits más significativos de la 4–tupla (a,b,c,d).

arith_cf_ng_hash[ ] Tabla sombreada que establece la correspondencia entre el estado del contexto y un índice de tablas de frecuencias acumuladas pki.

arith_cf_ng[pki][545] Modelos de las frecuencias acumuladas correspondientes al símbolo índice de grupo ng.

arith_cf_ne [ ] Frecuencias acumuladas correspondientes al símbolo índice ne.

r Plano de bits de la 4–tupla menos significativa que el plano de 2 bits menos significativos.

arith_cf_r [ ] Frecuencias acumuladas correspondientes al símbolo de planos de bis menos significativos r

[0075] A continuación, se considera, en primer lugar, el procedimiento de decodificación. Se codifican sin ruido coeficientes espectrales cuantificados por 4–tuplas y se transmiten a partir del coeficiente de frecuencia o espectral más bajo y progresando hasta el coeficiente de frecuencia o espectral más alto. Los coeficientes de AAC se almacenan en la matriz x_ac_quant[g][win][sfb][bin], y el orden de transmisión de las palabras de código de la codificación sin ruido es tal que cuando se decodifican en el orden recibido y se almacenan en la matriz, bin es el índice que se incrementa más rápidamente y g es el índice con incremento más lento. Dentro de una palabra de código, el orden de decodificación es a, b, c, d. El coeficiente del TCX se almacena directamente en la matriz x_tcx_invquant[win][bin], y el orden de la transmisión de las palabras de código de la codificación sin ruido es tal que cuando se decodifican en el orden recibido y se almacenan en la matriz, bin es el índice que se incrementa más rápidamente y win es el índice que se incrementa más lentamente. Dentro de una palabra de código el orden de decodificación es a, b, c, d. En primer lugar, el indicador arith_reset_flag determina si se debe reiniciar el contexto. Si el indicador es TRUE (verdadero), se usa la siguiente función:

```
*variables globales*/
q[2][290] /* contexto actual */
qs[258] /* contexto anterior */
previous_lg /* número de 4–tuplas del contexto anterior */
arith_reset_context()
{
        for (i=0;i<258;i++) {
                qs[i].a=0; qs[i].b=0; qs[i].c=0; qs[i].d=0
                qs[i].v=–1;
        }
        for (i=0;i<290;i++) {
                q[0][i].a=0; q[0][i].b=0; q[0][i].c=0;
                q[0][i].d=0
                q[0][i].v=–1;
                q[1][i].a=0; q[1][i].b=0; q[1][i].c=0;
                q[1][i].d=0
                q[1][i].v=–1;
}
previous_lg=256;
}
```

En caso contrario, cuando el indicador arith_reset_flag es FALSE (falso), se ejecuta una correspondencia entre el contexto anterior y el contexto actual:

```
/* variable de entrada */
lg /* número de 4–tuplas */
arith_map_context(lg)
{
v=w=0
if(core_mode==1) {
        q[0] [v++]=qs[w++];
}
ratio= ((float)previous_lg)/((float)lg);
for (j=0; j<lg; j++) {
        k = (int) ((float)) ((j)*ratio);
        q[0][v++] = qs[w+k];
```

```
}
if(core_mode==0) {
        q[0][lg]=qs[previous_lg];
}
q[0][lg+1]=qs[previous_lg+1];

previous_lg=lg;
}
```

[0076] El decodificador sin ruido produce una salida de 4–tuplas de coeficientes espectrales cuantificados identificados. Al principio se calcula el estado del contexto basándose en los cuatro grupos decodificados previamente que rodean a la 4–tupla que se ha de decodificar. El estado del contexto viene dado por la función arith_get_context():

```
/* variables de entrada */
i /* el índice de la 4–tupla que se ha de decodificar en el vector */
arith_get_context(i,)
{
        t0=q[0][1+i].v+1;
        t1=q[1][1+i–1].v+1;
        t2=q[0][1+i–1].v+1;
        T3=q[0][1+i+1].v+1;

        if ( (t0<10) && (t1<10) && (t2<10) && (t3<10) ) {
            if ( t2>1 ) t2=2;
            if ( t3>1 ) t3=2;
            return 3*(3*(3*(3*(3*(10*(10*t0+t1))+t2)+t3)));
        }
        if ( (t0<34) && (t1<34) && (t2<34) && (t3<34) ) {
            if ( (t2>1) && (t2<10) ) t2=2; else if ( t2>=10 ) t2=3;
            if ( (t3>1) && (t3<10) ) t3=2; else if ( t3>=10 ) t3=3;
            return 252000+4*(4*(34*(34*t0+t1))+t2)+t3;
        }
        if ( (t0<90) && (t1<90) ) return 880864+90*(90*t0+t1);

        if ( (t0<544) && (t1<544) ) return 1609864 + 544*t0+t1;

        if ( t0>1 )
        {
                a0=q[0][i].a;
                b0=q[0][i].b;
                c0=q[0][i].c;
                d0=q[0][i].d;
        }
        else a0=b0=c0=d0=0;

if ( t1>1 )
{ a1=q[1][i–1].a;
        b1=q[1][i–1].b;
        c1=q[1][i–1].c;
        d1=q[1][i–1].d;
}
else a1=b1=c1=d1=0;
l=0;
do
{
        a0>>=1;
        b0>>=1;
        c0>>=1;
        d0>>=1;

        a1>>=1;
        b1>>=1;
        c1>>=1;
        d1>>=1;
```

```
                1++;
}
while ( (a0<–4) || (a0>=4) || (b0<–4) || (b0>=4) || (c0<–4) || (c0>=4) || (d0<–4) || (d0>=4) ||
(a1<–4) || (a1>=4) || (b1<–4) || (b1>=4) || (c1<–4) || (c1>=4) || (d1<–4) || (d1>=4) );
if ( t0>1 ) t0=1+(egroups[4+a0][4+b0][4+c0][4+d0] >> 16);
if ( t1>1 ) t1=1+(egroups[4+a1][4+b1][4+c1][4+d1] >> 16);
return 1609864 + ((l<<24)|(544*t0+t1)); }
```

[0077] Una vez conocido el estado, se decodifica el grupo al cual pertenece el plano de 2 bits más significativo de la 4–tupla usando la arith_decode() suministrada con la tabla de frecuencias acumuladas apropiadas correspondiente al estado del contexto. La correspondencia se realiza mediante la función arith_get_pk():

```
/* variable de entrada */
s /* estado del contexto */
arith_get_pk(s)
{
psci[28] = {
                247,248,249,250,251,252,253,254,254,0,254,254,254,255,250,215,
                215,70,70,123,123,123,123,3,67,78,82,152
};
register unsigned char *p;
register unsigned long i,j;
i=123*s;
for (;;)
{
                j=arith_cf_nq_hash[i&32767];
                if ( j==0xFFFFFFFFul ) break;
                if ( (j>>8)==s ) return j&255;

                                i++;
                } p=psci+7*(s>>22);
                j= s & 4194303;
                if ( j<436961 )
                {
                if ( j<252001 ) return p[(j<243001)?0:1]; else return p[(j<288993)?2:3];
                }
else
{
                if ( j<1609865 ) return p[(j<880865)?4:5]; else return p[6];
}
                }
```

[0078] A continuación, se invoca a la función arith_decode() con la tabla de frecuencias acumuladas que corresponde al retorno del índice por la arith_get_pk(). El codificador aritmético es una implementación de números enteros que genera identificadores con cambio de escala. El siguiente seudocódigo en C describe el algoritmo usado.

```
/* funciones auxiliares */
bool arith_first_symbol(void);
    /* Devolver TRUE (verdadero) si es el primer símbolo de la secuencia, FALSE (falso) si no lo es */
    Ushort arith_get_next_bit(void);
    /* Adquirir el siguiente bit del tren de bits */
    /* variables globales */
    low
    high
    value

    /* variables de entrada */
    cum_freq[];
    cfl;

    arith_decode()
    if(arith_first_symbol())
    {
```

```
            value = 0;
            para (i=1; i<=20; i++)
            {
            value = (val<<1) | arith_get_next_bit();
            }
            low=0;
            high=1048575;

        }
        range = high–low+1;
        cum =((((int64) (value–low+1))<<16)–((int64) 1))/((int64) range);
    p = cum_freq–1;

        do
        {
        q=p+(cfl>>1);
        if ( *q > cum ) { p=q; cfl++; }
    cfl>>=1;
    }
    while ( cfl>1 );

    symbol = p–cum_freq+1;
    if(symbol)
    high = low + (((int64) range)*((int64)cum_freq[symbol–1]))>>16 – 1;
    low += (((int64) range)* ((int64) cum_freq[symbol]))>>16;

    for (;;)
    {
    if ( high<524286) { }
        else if ( low>=524286)
            {
            value –=524286;
            low –=524286;
            high –=524286;
            }
            else if (low>=262143 && high<786429)
            {
            value –= 262143;
            low –= 262143;
            high –= 262143;
        }
    else break;

    low += low;
    high += high+1;
            value = (value<<1) | arith_get_next_bit();
        }
    return symbol;
    }
```

[0079] Si bien el índice del grupo decodificado ng es el símbolo de escape, ARITH_ESCAPE, se decodifica un índice de grupo adicional ng y se incrementa en dos la variable lev. Una vez que el índice de grupo ya no es el símbolo de escape, ARITH_ESCAPE, se deduce el número de elementos, mm, dentro del grupo y el desplazamiento del grupo og, buscando en la tabla dgroups[]:

```
mm=dgroups[nq]&255
og = dgroups[nq]>>8
```

[0080] A continuación, se decodifica el índice del elemento ne invocando arith_decode() con la tabla de frecuencias acumuladas (arith_cf_ne+((mm*(mm–1))>>1)[]. Una vez decodificado el índice de elemento se puede deducir el plano de 2 bits más significativo de la 4–tupla con la tabla dgvector[]:

```
a=dgvectors[4*(og+ne)]

b=dgvectors[4*(og+ne)+1]
```

c=dgvectors[4*(og+ne)+2]

d=dgvectors[4*(og+ne)+3]

[0081] A continuación, se decodifica el resto de los planos de bits desde el nivel más significativo hasta el nivel menos significativo invocando lev veces arith_decode() con la tabla de frecuencias acumuladas arith_cf_r []. El plano de bits codificado r permite afinar la 4–tupla decodificada de la siguiente manera:

a = (a<<1) | (r&1)

b = (b<<1) | ((r>>1)&1)

c = (c<<1) | ((r>>2)&1)

d = (d<<1) | (r>>3)

[0082] Una vez que la 4–tupla (a,b,c,d) ha sido completamente decodificada se actualizan las tablas de contexto q y qs invocando la función arith_update_context().

```
arith_update_context(a,b,c,d,i,lg)
{
    q[1][1+i].a=a;
    q[1][1+i].b=b;
    q[1][1+i].c=c;
    q[1][1+i].d=d;

    if ( (a<–4) || (a>=4) || (b<–4) || (b>=4) || (c<–4) || (c>=4)
    || (d<–4) || (d>=4) )
    q[1][1+i].v =1024;
    else q[1][1+i].v=egroups[4+a][4+b][4+c][4+d];
    if(i==lg && core_mode==1) {
        qs[0]=q[1] [0];

    ratio= ((float) lg)/((float)256);
    for (j=0; j<256; j++) {
                k = (int) ((float)) ((j)*ratio);
                qs[1+k] = q[1][1+j];
    }
    qs[previous_lg+1]=q[1][lg+1];
    previous_lg=256;
    }
if(i==lg && core_mode==0) {
    for (j=0; j<258; j++) {
                qs[j] = q[1][k];
    }
    previous_lg=min(1024,lg);
    } }
```

[0083] A continuación, se describirán realizaciones y aspectos adicionales de la invención que pueden usarse de forma individual o en combinación con las características y funcionalidades descritas en la presente memoria.

[0084] Según un primer aspecto, un codificador de audio (100) para codificar segmentos de coeficientes, representando los segmentos de coeficientes diferentes resoluciones de tiempo o de frecuencia de una señal de audio muestreada, puede comprender un procesador (110) para deducir un contexto de codificación para un coeficiente que se está codificando actualmente de un segmento actual basándose en un coeficiente codificado previamente de un segmento anterior, representando el coeficiente codificado previamente una resolución de tiempo o de frecuencia diferente que el coeficiente que se está codificando actualmente; y un codificador de entropía (120) para codificación de entropía del coeficiente actual basándose en el contexto de codificación con el fin de obtener un tren de audio codificado.

[0085] Según un segundo aspecto que hace referencia al primer aspecto, el codificador de audio (100) puede comprender además un medio para proporcionar los segmentos de coeficientes de un tren de audio, formando los coeficientes una representación espectral de una señal de audio representada por el tren de audio a una espectral que varía entre los segmentos.

[0086] Según un tercer aspecto que hace referencia al segundo aspecto, en el codificador de audio (100), el codificador de entropía (120) puede estar adaptado de manera que codifique el coeficiente actual en unidades de una tupla de coeficientes espectrales y prediga un intervalo de la tupla basándose en el contexto de codificación.

[0087] Según un cuarto aspecto que hace referencia al tercer aspecto, en el codificador de audio (100), el codificador de entropía (120) puede estar adaptado para dividir la tupla en un factor predeterminado el número de veces que sea necesario para ajustar un resultado de la división en un intervalo predeterminado y para codificar una serie de divisiones necesaria, un resto de división y el resultado de la división cuando la tupla no está situada dentro del intervalo predicho, y para codificar de otro modo un resto de división y el resultado la división.

[0088] Según un quinto aspecto que hace referencia al cuarto aspecto, en el codificador de audio (100), el codificador de entropía (120) puede estar adaptado de manera que codifique el resultado de la división o la tupla usando un índice de grupo, de manera que el índice de grupo se refiere a un grupo de una o más palabras de código para las cuales la distribución de probabilidades se basa en el contexto de codificación, y, basándose en una distribución de probabilidades uniforme, un índice de elemento en el caso en que el grupo comprenda más de una palabra de código, haciendo referencia el índice de elemento a una palabra de código dentro del grupo, y codifique la serie de divisiones por una serie de símbolos de escape, siendo un símbolo de escape un índice de grupo específico usado solamente para indicar una división, y codifique los restos de las divisiones basándose en una distribución de probabilidades uniforme mediante el uso de una regla de codificación aritmética.

[0089] Según un sexto aspecto que hace referencia al quinto aspecto, en el codificador de audio (100), el codificador de entropía (120) puede adaptarse de manera que codifique una secuencia de símbolos en el tren de audio codificado usando un alfabeto de símbolos que comprende el símbolo de escape, y símbolos de grupo que corresponden a un conjunto de índices de grupo disponibles, comprendiendo un alfabeto de símbolos los índices de elemento correspondientes, y comprendiendo un alfabeto de símbolos los diferentes valores de los restos.

[0090] Según un séptimo aspecto que hace referencia a cualquiera de los aspectos primero a sexto, en el codificador de audio (100), el procesador (110) y el codificador de entropía (120) pueden configurarse de manera que funcionen basándose en un submuestreo de coeficientes espectrales del segmento anterior, cuando el segmento anterior muestra una resolución espectral más fina que el segmento actual y/o el procesador (110) y el codificador de entropía (120) pueden configurarse de manera que funcionen basándose en un sobremuestreo de coeficientes espectrales del segmento anterior, cuando el segmento anterior muestra una resolución espectral menos fina que el segmento actual.

[0091] Según un octavo aspecto, un procedimiento para codificar segmentos de coeficientes, representando los segmentos resoluciones de tiempo o de frecuencia diferentes de una señal de audio muestreada, puede comprender las etapas consistentes en: deducción de un contexto de codificación para un coeficiente que se está codificando actualmente a partir de un segmento actual basándose en un coeficiente codificado previamente de un segmento anterior, de manera que el coeficiente codificado previamente representa una resolución de tiempo o de frecuencia diferente que el coeficiente que se está codificando actualmente; y codificación de entropía del coeficiente actual basándose en el contexto de codificación con el fin de obtener un tren de audio codificado.

[0092] Según un noveno aspecto, un decodificador de audio (200) para decodificar un tren de audio codificado con el fin de obtener segmentos de coeficientes que representan resoluciones de tiempo o de frecuencia diferentes de una señal de audio muestreada puede comprender: un procesador (210) para deducir un contexto de codificación para un coeficiente que se está decodificando actualmente de un segmento actual basándose en un coeficiente decodificado previamente de un segmento anterior, representando el coeficiente decodificado previamente una resolución de tiempo o de frecuencia diferente que el coeficiente que se está decodificando actualmente; y un decodificador de entropía (220) para decodificación de entropía del coeficiente actual basándose en el contexto de codificación y el tren de audio codificado.

[0093] Según un décimo aspecto que hace referencia al noveno aspecto, en el decodificador de audio (200), el procesador (210) puede estar adaptado de manera que deduzca el contexto de codificación basándose en el coeficiente anterior, formando los coeficientes una representación espectral de una señal de audio representada por el tren de audio a una resolución espectral que varía entre los segmentos.

[0094] Según un decimoprimer aspecto que hace referencia a cualquiera de los aspectos noveno a décimo, en el decodificador de audio (200), el procesador puede estar adaptado de manera que deduzca el contexto de codificación por banda espectral para el coeficiente actual, basándose en coeficientes espectrales adyacentes decodificados previamente en el segmento anterior y opcionalmente en el presente segmento.

[0095] Según un decimosegundo aspecto que hace referencia al decimoprimer aspecto, en el decodificador de audio (200), el decodificador de entropía (220) puede adaptarse de manera que decodifique un índice de grupo a partir del tren de audio codificado basándose en una distribución de probabilidades deducida del contexto de

codificación, en el que el índice de grupo representa un grupo de una o más palabras de código, y que, basándose en una distribución de probabilidades uniforme, decodifique un índice de elemento a partir del tren de audio codificado si el índice de grupo indica un grupo que comprende más de una palabra de código, y que deduzca una tupla de coeficientes espectrales del segmento actual basándose en el índice de grupo y el índice de elemento, obteniendo de este modo la representación en el dominio espectral en tuplas de coeficientes espectrales.

[0096] Según un decimotercer aspecto que hace referencia al decimosegundo aspecto, en el decodificador de audio (200), el decodificador de entropía (220) puede estar adaptado de manera que decodifique una secuencia de símbolos a partir del tren de audio codificado basándose en la distribución de probabilidades deducida a partir del contexto de codificación usando un alfabeto de símbolos que comprende un símbolo de escape y símbolos de grupo que corresponden a un conjunto de índices de grupo disponibles, para deducir una tupla preliminar de coeficientes espectrales basándose en un índice de grupo disponible al cual corresponde un símbolo de grupo de la secuencia de símbolos, y basándose en el índice de elemento, y que multiplique la tupla preliminar por un factor que depende de un número de símbolos de escape en la secuencia de símbolos con el fin de obtener la tupla de coeficientes espectrales.

[0097] Según un decimocuarto aspecto que hace referencia al decimotercer aspecto, en el decodificador de audio (200), el decodificador de entropía (220) puede estar adaptado de manera que decodifique un resto de división a partir del tren de audio codificado basándose en una distribución de probabilidades uniforme que usa una regla de codificación aritmética y añada el resto a la tupla preliminar multiplicada con el fin de obtener la tupla de coeficientes espectrales.

[0098] Según un decimoquinto aspecto que hace referencia a cualquiera de los aspectos noveno a decimocuarto, en el decodificador de audio (200), el procesador (210) y el codificador de entropía (220) pueden configurarse de manera que funcionen basándose en un submuestreo de coeficientes espectrales del segmento anterior, cuando el segmento anterior muestra una resolución espectral más fina que el segmento actual y/o el procesador (210) y el codificador de entropía (220) pueden configurarse de manera que funcionen basándose en un sobremuestreo de coeficientes espectrales del segmento anterior, cuando el segmento anterior muestra una resolución espectral menos fina que el segmento actual.

[0099] Según un decimosexto aspecto, un procedimiento para decodificar un tren de audio codificado con el fin de obtener segmentos de coeficientes que representan muestras de audio decodificadas puede comprender las etapas consistentes en: deducción de un contexto de codificación para un coeficiente que se está decodificando actualmente a partir de un segmento actual basándose en un coeficiente decodificado previamente de un segmento anterior, de manera que el coeficiente decodificado previamente representa una resolución de tiempo o de frecuencia diferente que el coeficiente que se está decodificando actualmente; y decodificación de entropía del coeficiente actual basándose en el contexto de codificación y el tren de audio codificado. Según un decimoséptimo aspecto, un programa informático puede tener un código de programa para realizar uno de los procedimientos según el aspecto octavo o decimosexto, cuando el código de programa se ejecuta en un ordenador o un procesador.

[0100] Dependiendo de ciertos requisitos de implementación de los procedimientos de la presente invención, los procedimientos de la invención se pueden implementar en hardware o software. La implementación se puede formar usando un medio de almacenamiento digital, en especial un disco, un DVD o un CD, que tiene una señal de control legible de manera electrónica almacenada en el mismo, que coopera con el ordenador programable de tal manera que se puedan ejecutar los procedimientos de la presente invención. En general, por lo tanto, la presente invención es un producto de programa informático con un código de programa para un soporte legible por la máquina, en el que el código de programa es operativo para ejecutar los procedimientos de la presente invención cuando el programa informático se ejecuta en un ordenador. Dicho de otro modo, los procedimientos de la presente invención consisten por lo tanto en un programa informático que tiene un código de programa para ejecutar al menos uno de los procedimientos de la presente invención cuando el programa informático se ejecuta en un ordenador.

REIVINDICACIONES

1. Un codificador de audio (100) para codificar segmentos de coeficientes, en el que los segmentos se siguen unos a otros en el tiempo, comprendiendo el codificador de audio (100)

un medio para proporcionar la secuencia de segmentos de coeficientes a partir de un tren de audio que representa una señal de audio muestreada usando diferentes longitudes de transformada de manera que los segmentos de coeficientes para los cuales se usan diferentes longitudes de transformada representan espectralmente la señal de audio muestreada a diferentes resoluciones de frecuencia;

un procesador (110) para deducir un contexto de codificación correspondiente a un coeficiente codificado actualmente de un segmento actual basándose en un coeficiente codificado previamente de un segmento anterior, en el que los segmentos anterior y actual corresponden a diferentes resoluciones de frecuencia y diferentes longitudes de transformada, respectivamente; y

un codificador de entropía (120) para codificación de entropía del coeficiente actual basándose en el contexto de codificación para obtener un tren de audio codificado,

en el que el procesador (110) y el codificador de entropía (120) están configurados para funcionar basándose en un submuestreo de coeficientes espectrales del segmento anterior, cuando el segmento anterior muestra una resolución espectral más fina que el segmento actual y/o en el que el procesador (110) y el codificador de entropía (120) están configurados para funcionar basándose en un sobremuestreo de coeficientes espectrales del segmento anterior, cuando el segmento anterior muestra una resolución espectral menos fina que el segmento actual.

2. El codificador de audio (100) según la reivindicación 1, en el que el codificador de entropía (120) está adaptado para codificar el segmento actual en unidades de una tupla de coeficientes espectrales y para predecir un intervalo de la tupla basándose en el contexto de codificación.

3. El codificador de audio (100) según la reivindicación 2, en el que el codificador de entropía (120) está adaptado para dividir la tupla por un factor predeterminado tantas veces como sea necesario para ajustar un resultado de la división a un intervalo predeterminado y para codificar una serie de divisiones necesaria, un resto de la división y el resultado de la división cuando la tupla no se sitúa dentro del intervalo predicho.

4. El codificador de audio (100) según la reivindicación 3, en el que el codificador de entropía (120) está adaptado para codificar el resultado de la división o la tupla usando un índice de grupo, refiriéndose el índice de grupo a un grupo de una o más palabras de código con respecto a las cuales la distribución de probabilidades se basa en el contexto de codificación y, basándose en una distribución uniforme de las probabilidades, un índice de elemento en un caso en que el grupo comprende más de una palabra de código, refiriéndose el índice de elemento a una palabra de código dentro del grupo, y para codificar la serie de divisiones mediante una serie de símbolos de escape, siendo un símbolo de escape un índice de grupo específico usado sólo para indicar una división, y para codificar los restos de las divisiones basándose en una distribución uniforme de probabilidades usando una regla de codificación aritmética.

5. El codificador de audio (100) según la reivindicación 4, en el que el codificador de entropía (120) está adaptado para codificar una secuencia de símbolos en el tren de audio codificado usando un alfabeto de símbolos que comprende el símbolo de escape y símbolos de grupo que corresponden a un conjunto de índices de grupos disponibles, comprendiendo un alfabeto de símbolos los índices de elementos correspondientes, y comprendiendo un alfabeto de símbolos los diferentes valores de los restos.

6. Un procedimiento para codificar una secuencia de segmentos de coeficientes, en el que los segmentos se siguen unos a otros en el tiempo, comprendiendo el procedimiento las etapas siguientes

suministro de la secuencia de segmentos de coeficientes a partir de un tren de audio que representa una señal de audio muestreada usando diferentes longitudes de transformada de manera que los segmentos de coeficientes para los que se usan diferentes longitudes de transformada representan espectralmente la señal de audio muestreada a diferentes resoluciones de frecuencia;

deducción de un contexto de codificación correspondiente a un coeficiente codificado actualmente de un segmento actual basándose en un coeficiente codificado previamente de un segmento anterior, en el que el coeficiente codificado previamente corresponde a diferentes resoluciones de frecuencia y diferentes longitudes de transformada, respectivamente; y

codificación de entropía del coeficiente actual basándose en el contexto de codificación para obtener un tren de audio codificado, en el que el suministro y la deducción se realizan basándose en un submuestreo de coeficientes espectrales del segmento anterior, cuando el segmento anterior muestra una resolución espectral más fina que el

segmento actual y/o en el que el suministro y la deducción se realizan basándose en un sobremuestreo de coeficientes espectrales del segmento anterior, cuando el segmento anterior muestra una resolución espectral menos fina que el segmento actual.

7. Un decodificador de audio (200) para decodificar un tren de audio codificado que representa una señal de audio muestreada para obtener una secuencia de segmentos de coeficientes que se siguen unos a otros en el tiempo y que representan la señal de audio muestreada usando diferentes longitudes de transformada de manera que los segmentos de coeficientes para los que se usan diferentes longitudes de transformada representan espectralmente la señal de audio muestreada a diferentes resoluciones de frecuencia, que comprende

un procesador (210) para deducir un contexto de codificación para un coeficiente decodificado actualmente de un segmento actual basándose en un coeficiente decodificado previamente de un segmento anterior, en el que los segmentos anterior y actual corresponden a diferentes resoluciones de frecuencia y diferentes longitudes de transformada, respectivamente; y

un decodificador de entropía (220) para decodificación de entropía del coeficiente actual basándose en el contexto de codificación y el tren de audio codificado,

en el que el procesador (210) y el decodificador de entropía (220) están configurados para funcionar basándose en un submuestreo de coeficientes espectrales del segmento anterior, cuando el segmento anterior muestra una resolución espectral más fina que el segmento actual y/o en el que el procesador (210) y el decodificador de entropía (220) están configurados para funcionar basándose en un sobremuestreo de coeficientes espectrales del segmento anterior, cuando el segmento anterior muestra una resolución espectral menos fina que el segmento actual.

8. El decodificador de audio (200) según la reivindicación 7, en el que el procesador está adaptado para deducir el contexto de codificación por banda espectral correspondiente al coeficiente actual, basándose en coeficientes espectrales adyacentes decodificados previamente en el segmento anterior y opcionalmente en el segmento actual.

9. Un procedimiento para decodificar un tren de audio codificado que representa una señal de audio muestreada para obtener una secuencia de segmentos de coeficientes que se siguen unos a otros en el tiempo y que representan las señales de audio muestreadas usando diferentes longitudes de transformada de manera que los segmentos de coeficientes para los que se usan diferentes longitudes de transformada representan espectralmente la señal de audio muestreada a diferentes resoluciones de frecuencia, que comprende las etapas de

deducción de un contexto de codificación para un coeficiente decodificado actualmente de un segmento actual basándose en un coeficiente decodificado previamente de un segmento anterior, en el que los coeficientes anterior y actual corresponden a diferentes resoluciones de frecuencia y diferentes longitudes de transformada, respectivamente; y

decodificación de entropía del coeficiente actual basándose en el contexto de codificación y el tren de audio codificado,

en el que la deducción y la decodificación de entropía se realizan basándose en un submuestreo de coeficientes espectrales del segmento anterior, cuando el segmento anterior muestra una resolución espectral más fina que el segmento actual y/o en el que la deducción y la decodificación de entropía se realizan basándose en un sobremuestreo de coeficientes espectrales del segmento anterior, cuando el segmento anterior muestra una resolución espectral menos fina que el segmento actual.

10. Un programa informático que tiene un código de programa para ejecutar uno de los procedimientos según las reivindicaciones 6 o 9, cuando el código de programa se ejecuta en un ordenador o un procesador.

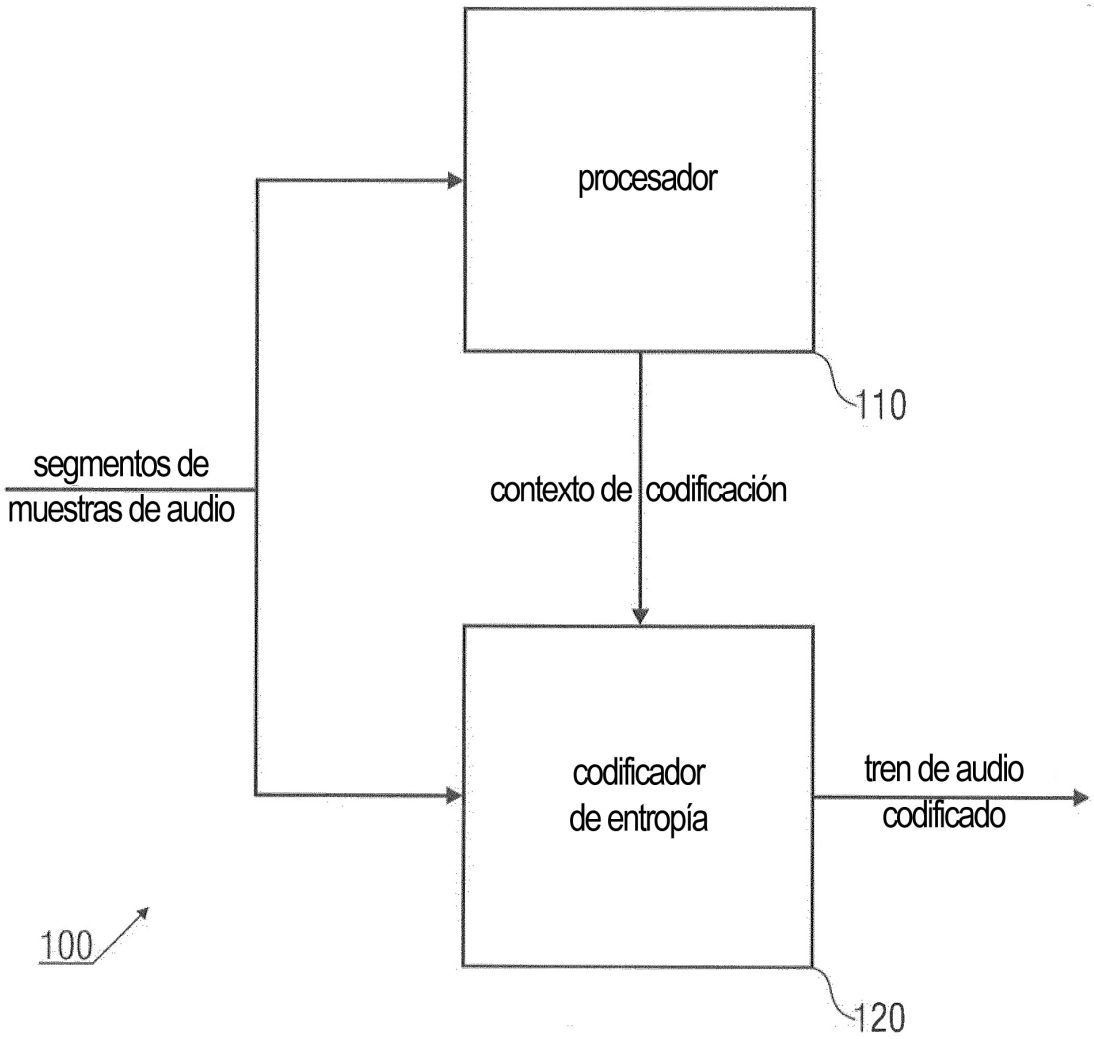
procesador
110
segmentos de
muestras de audio
contexto de codificación
codificador
de entropía
tren de audio
codificado
100
120

FIG 1

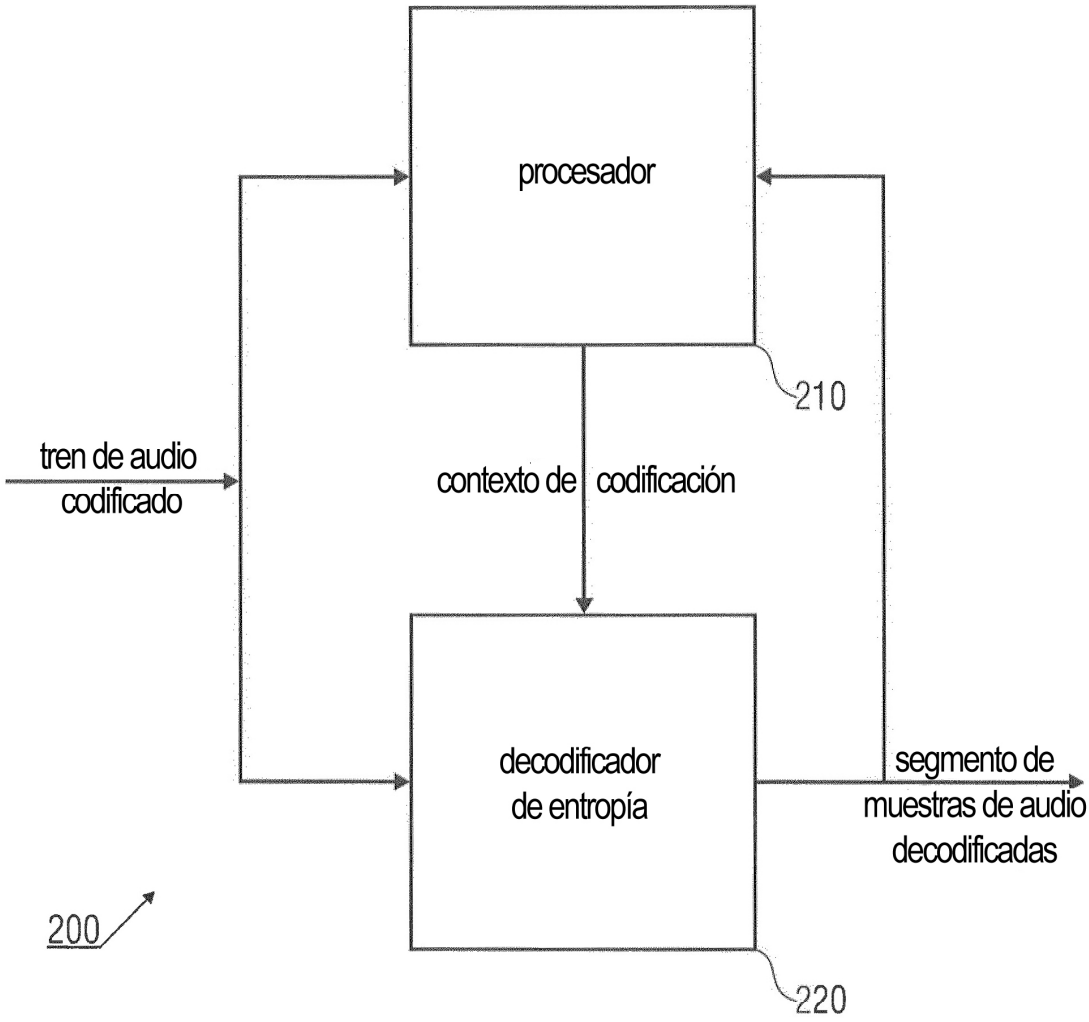
procesador
210
tren de audio codificado
contexto de codificación
decodificador de entropía
segmento de muestras de audio decodificadas
200
220

FIG 2

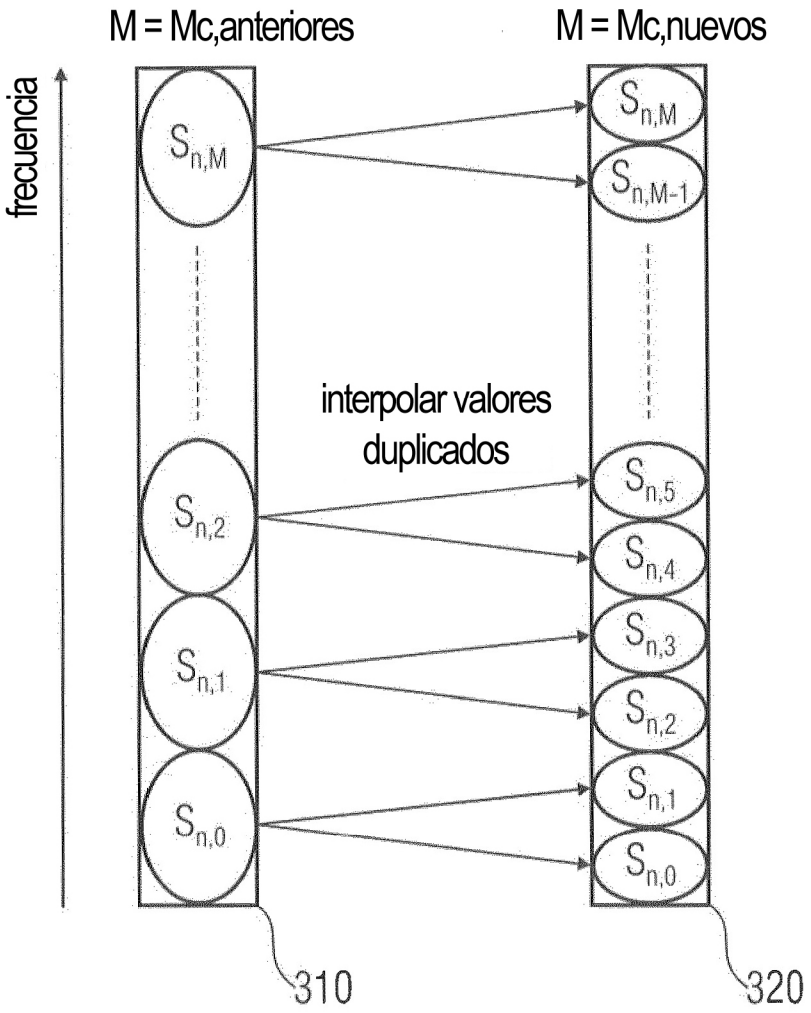
M = Mc,anteriores
M = Mc,nuevos
frecuencia
$S_{n,M}$
$S_{n,2}$
$S_{n,1}$
$S_{n,0}$
interpolar valores duplicados
$S_{n,M}$
$S_{n,M-1}$
$S_{n,5}$
$S_{n,4}$
$S_{n,3}$
$S_{n,2}$
$S_{n,1}$
$S_{n,0}$
310
320

FIG 3

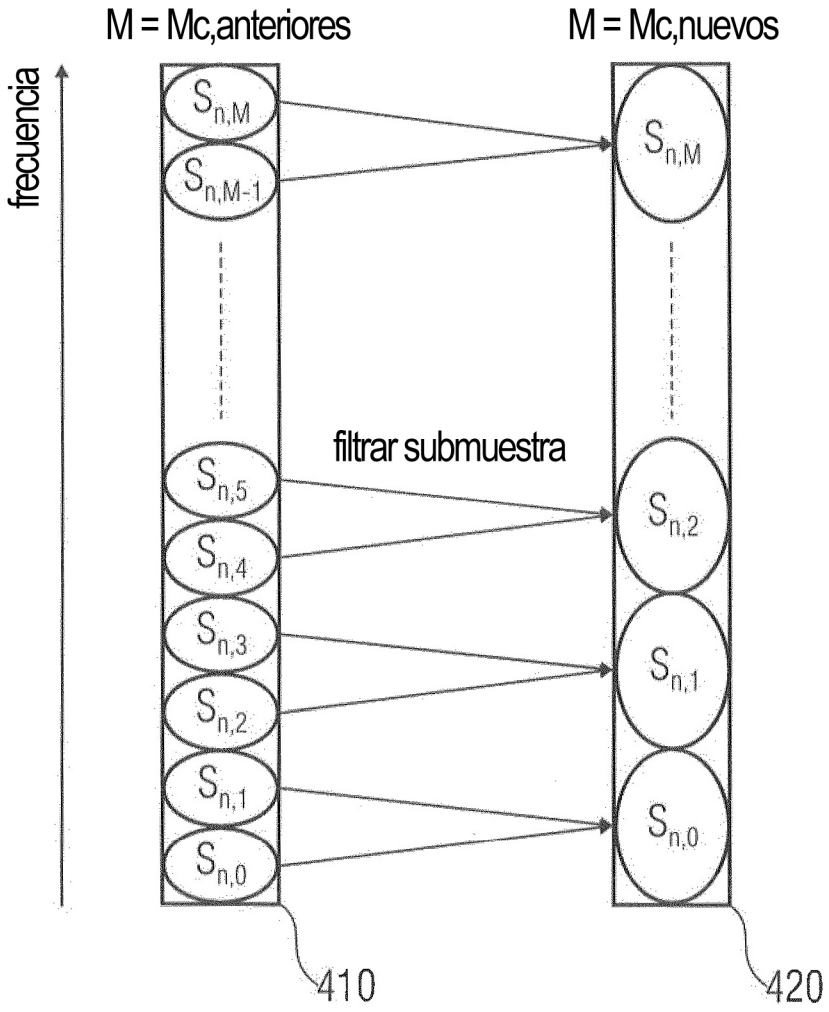
M = Mc,anteriores
M = Mc,nuevos
frecuencia
$S_{n,M}$
$S_{n,M-1}$
filtrar submuestra
$S_{n,5}$
$S_{n,4}$
$S_{n,3}$
$S_{n,2}$
$S_{n,1}$
$S_{n,0}$
$S_{n,M}$
$S_{n,2}$
$S_{n,1}$
$S_{n,0}$
410
420

FIG 4

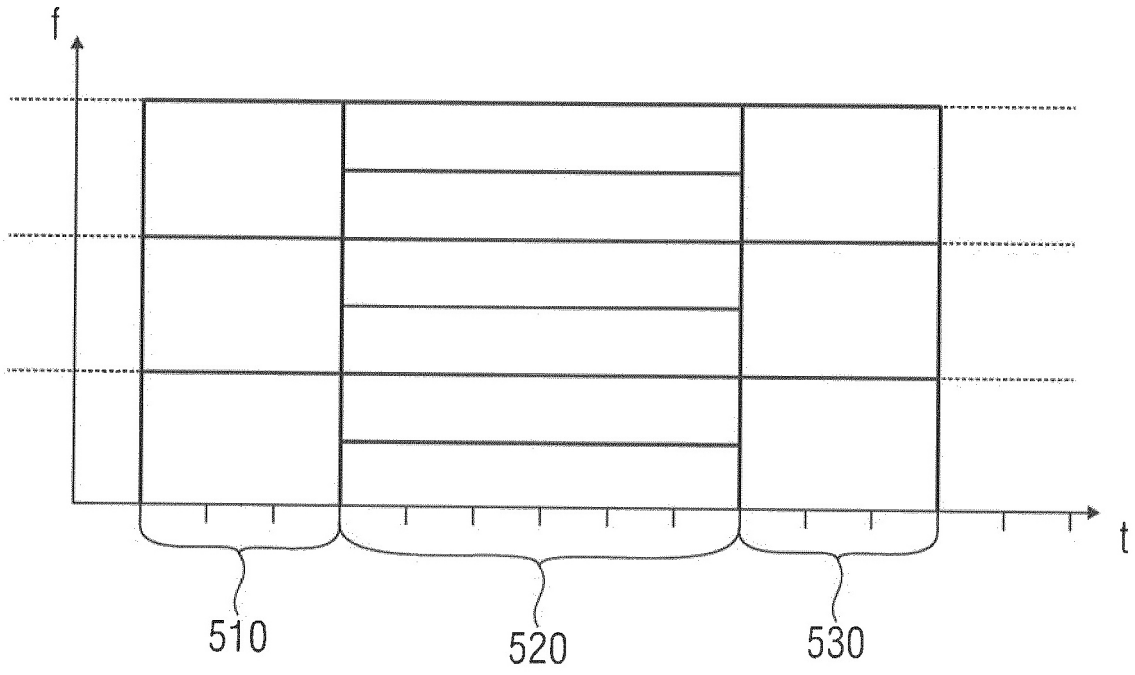

FIG 5

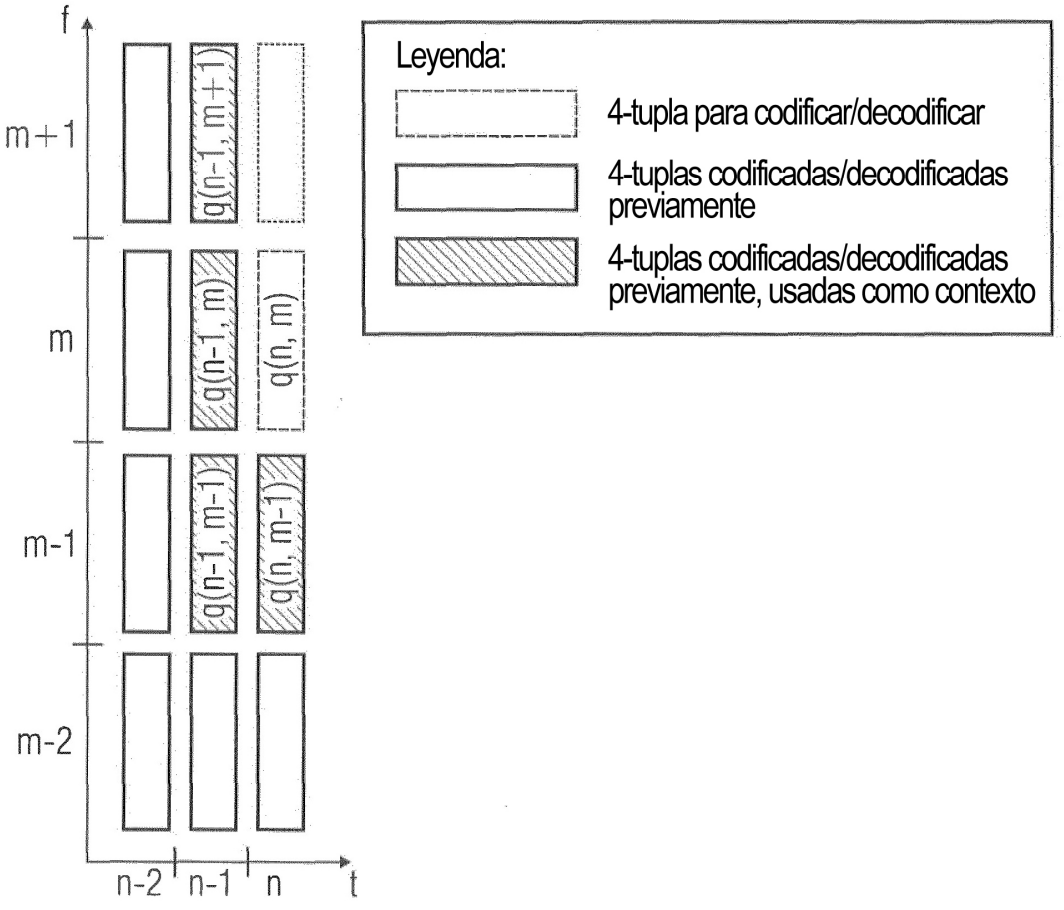
f
m+1
m
m-1
m-2
n-2
n-1
n
t
q(n-1, m+1)
q(n-1, m)
q(n, m)
q(n-1, m-1)
q(n, m-1)
Leyenda:
4-tupla para codificar/decodificar
4-tuplas codificadas/decodificadas previamente
4-tuplas codificadas/decodificadas previamente, usadas como contexto

FIG 6

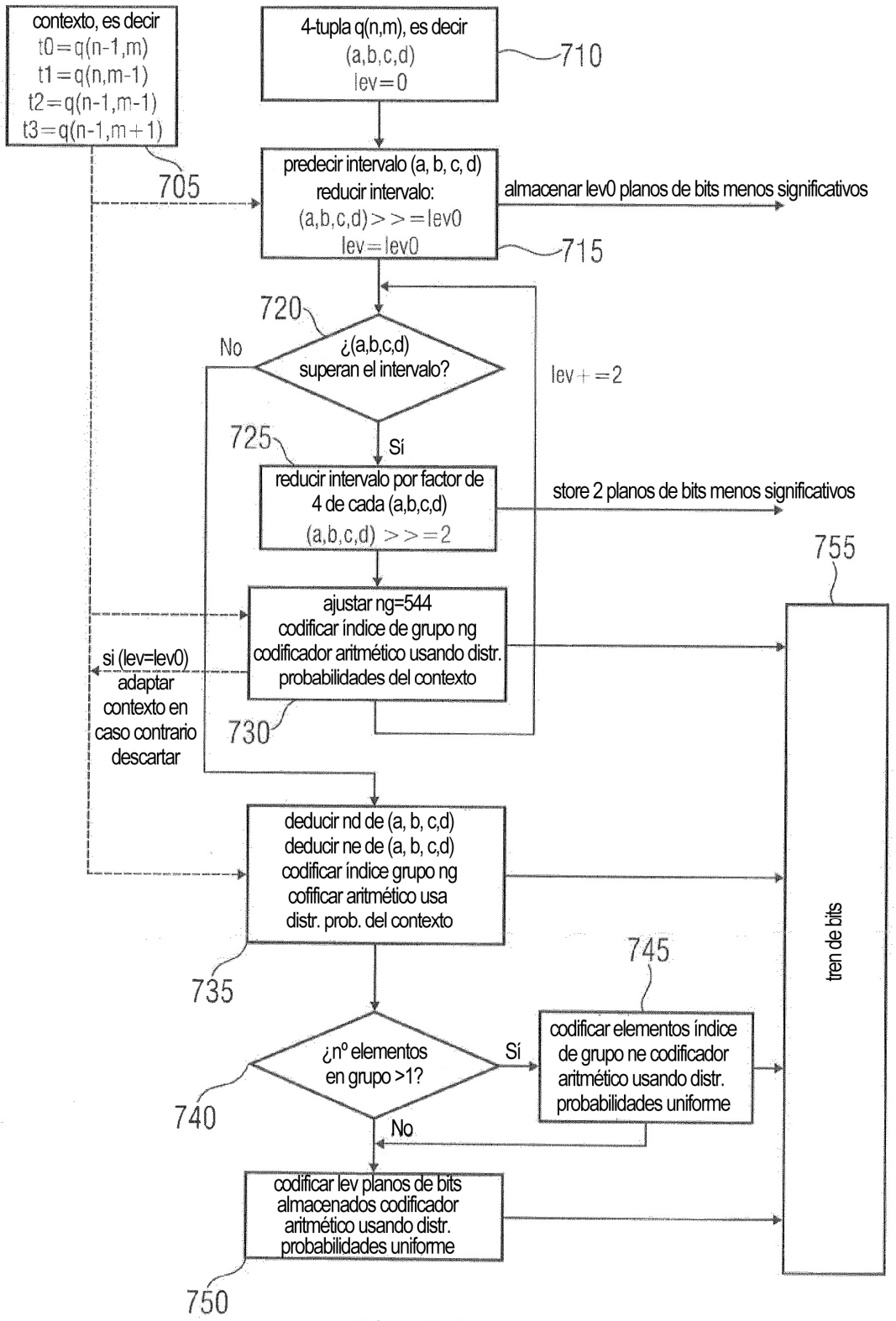
contexto, es decir
t0=q(n-1,m)
t1=q(n,m-1)
t2=q(n-1,m-1)
t3=q(n-1,m+1)
705
4-tupla q(n,m), es decir
(a,b,c,d)
lev=0
710
predecir intervalo (a, b, c, d)
reducir intervalo:
(a,b,c,d)>>=lev0
lev=lev0
715
almacenar lev0 planos de bits menos significativos
720
¿(a,b,c,d)
superan el intervalo?
No
Sí
lev+=2
725
reducir intervalo por factor de
4 de cada (a,b,c,d)
(a,b,c,d) >>=2
store 2 planos de bits menos significativos
755
ajustar ng=544
codificar índice de grupo ng
codificador aritmético usando distr.
probabilidades del contexto
730
si (lev=lev0)
adaptar
contexto en
caso contrario
descartar
deducir nd de (a, b, c,d)
deducir ne de (a, b, c,d)
codificar índice grupo ng
cofificar aritmético usa
distr. prob. del contexto
735
745
¿nº elementos
en grupo >1?
Sí
codificar elementos índice
de grupo ne codificador
aritmético usando distr.
probabilidades uniforme
740
No
codificar lev planos de bits
almacenados codificador
aritmético usando distr.
probabilidades uniforme
750
tren de bits

FIG 7A

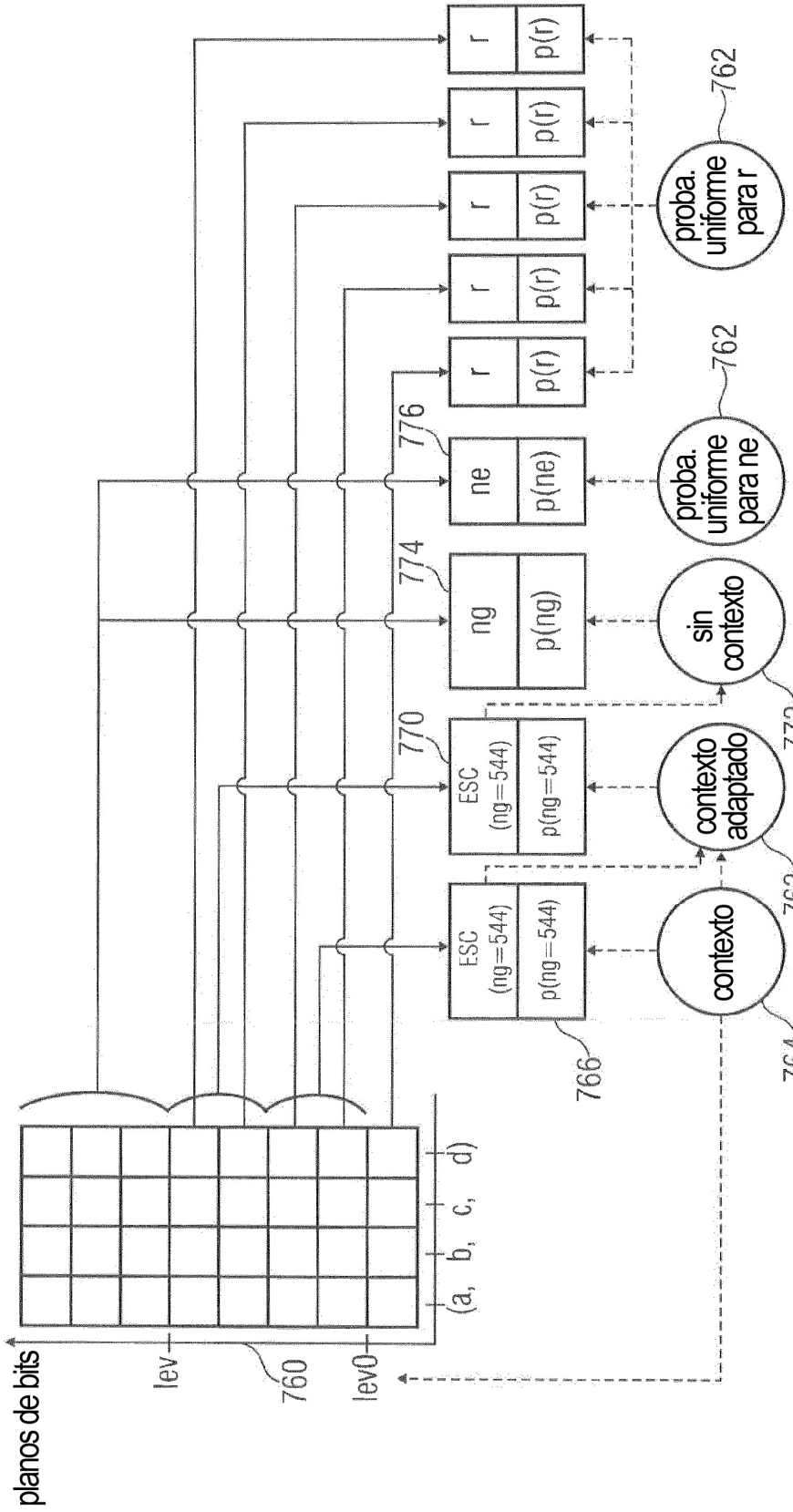
planos de bits
lev
760
lev0
(a, b, c, d)
ESC (ng=544)
p(ng=544)
766
770
ng
p(ng)
774
ne
p(ne)
776
r
p(r)
contexto
764
contexto adaptado
763
sin contexto
772
proba. uniforme para ne
762
proba. uniforme para r
762

FIG 7B

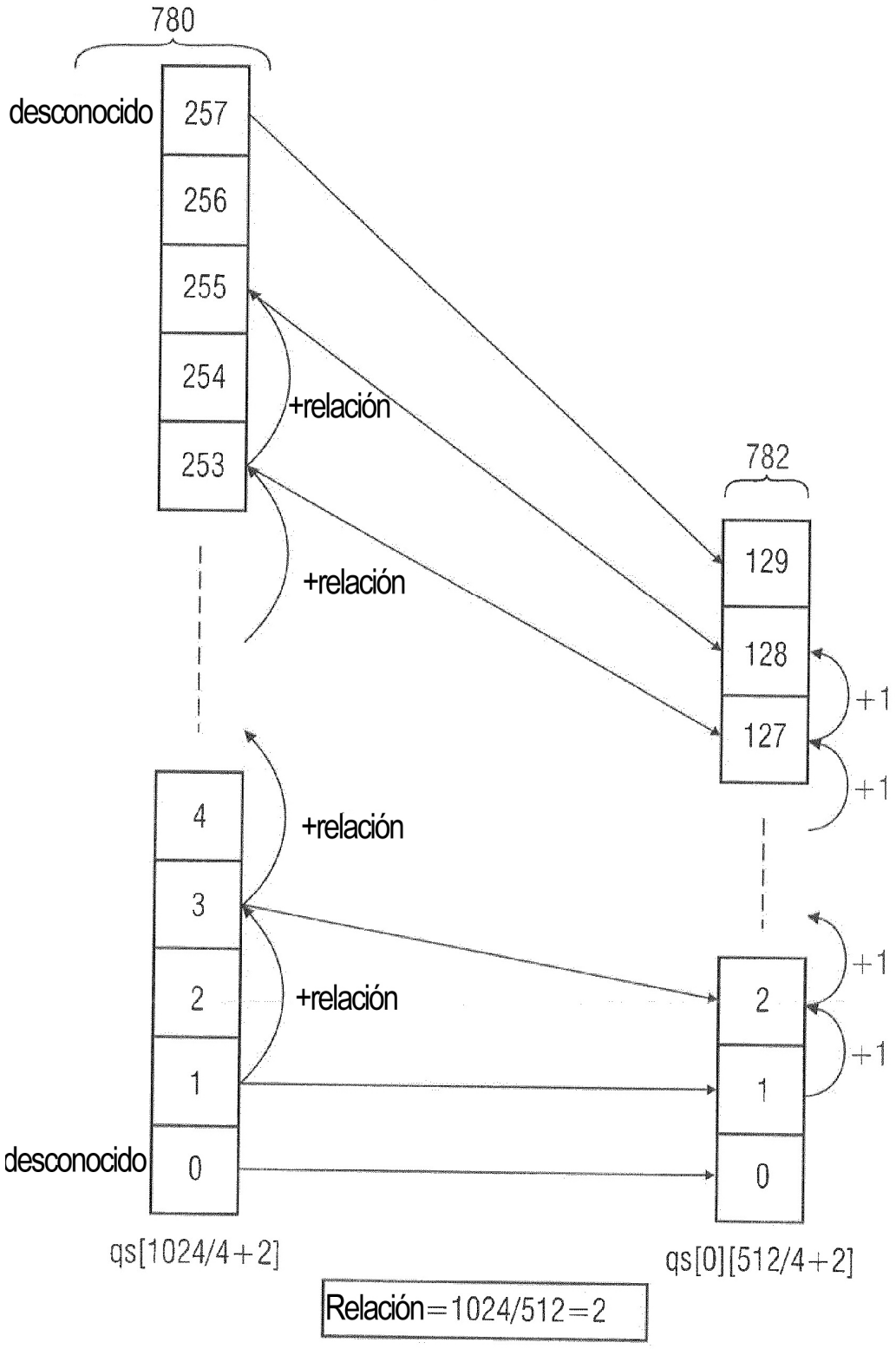
780
desconocido
257
256
255
254
253
+relación
+relación
4
3
2
1
0
+relación
+relación
desconocido
qs[1024/4+2]
782
129
128
127
+1
+1
2
1
0
+1
+1
qs[0][512/4+2]
Relación=1024/512=2

FIG 7C

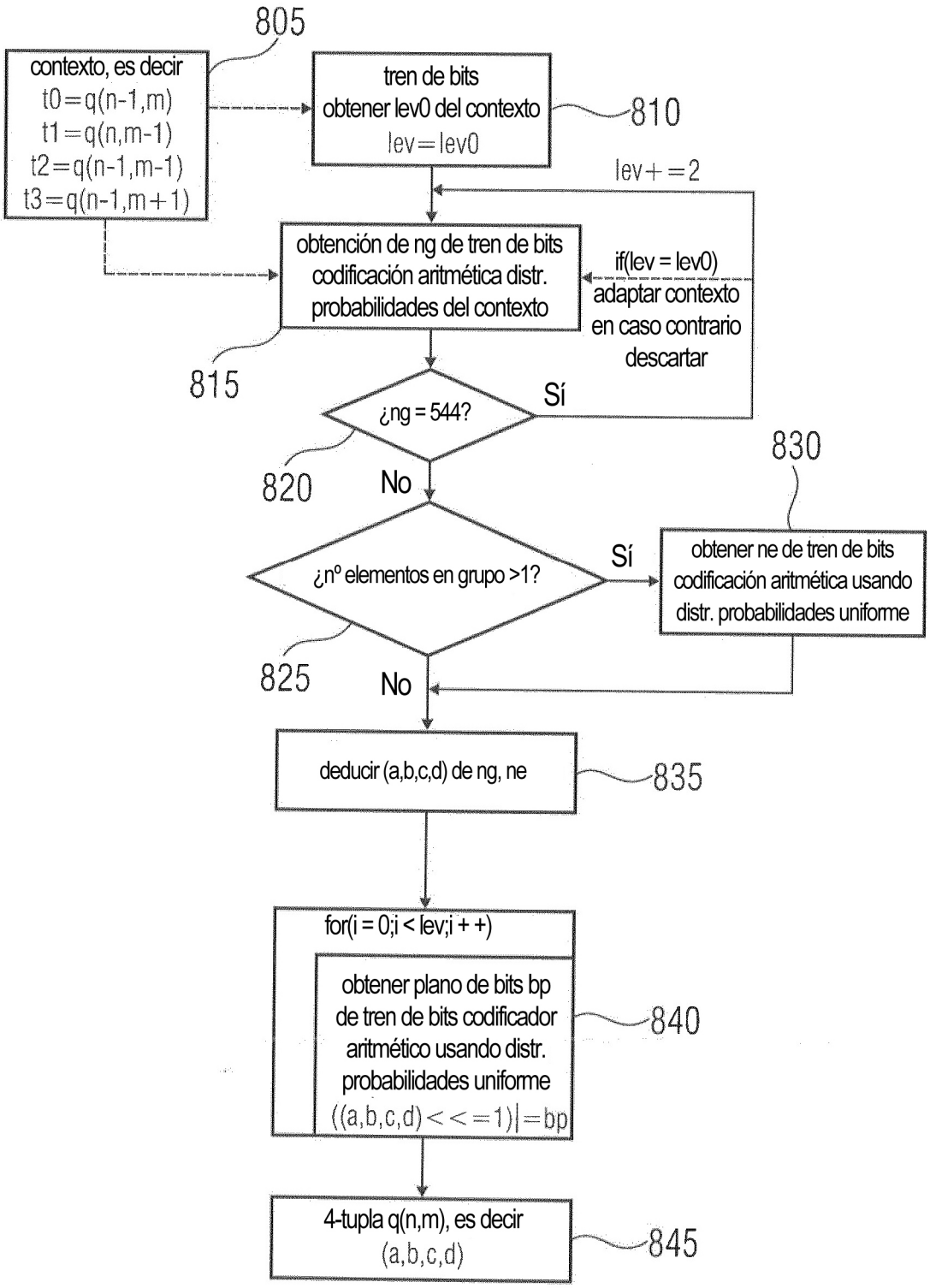
805
contexto, es decir
t0=q(n-1,m)
t1=q(n,m-1)
t2=q(n-1,m-1)
t3=q(n-1,m+1)
tren de bits
obtener lev0 del contexto
lev=lev0
810
lev+=2
obtención de ng de tren de bits
codificación aritmética distr.
probabilidades del contexto
if(lev = lev0)
adaptar contexto
en caso contrario
descartar
815
¿ng = 544?
Sí
820
No
830
¿nº elementos en grupo >1?
Sí
obtener ne de tren de bits
codificación aritmética usando
distr. probabilidades uniforme
825
No
deducir (a,b,c,d) de ng, ne
835
for(i = 0;i < lev;i + +)
obtener plano de bits bp
de tren de bits codificador
aritmético usando distr.
probabilidades uniforme
((a,b,c,d)<<=1)|=bp
840
4-tupla q(n,m), es decir
(a,b,c,d)
845

FIG 8

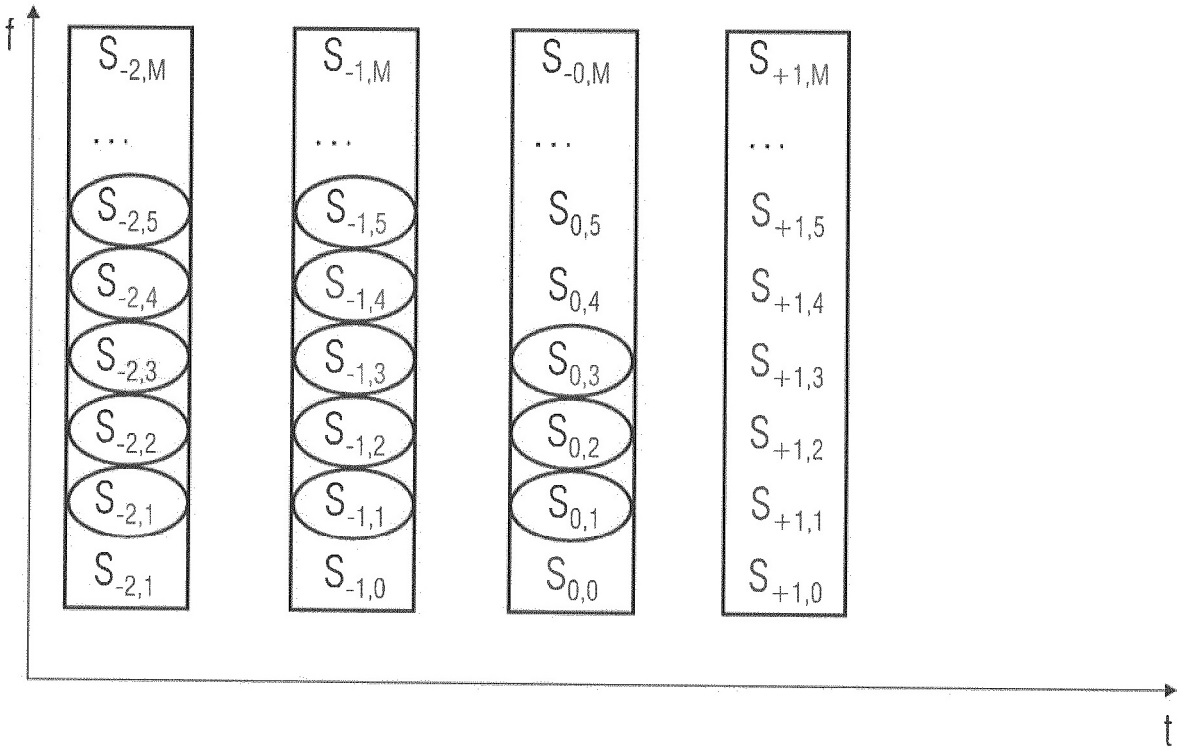

FIG 9
(ESTADO DE LA TÉCNICA)