(19) **Bundesrepublik Deutschland**
Deutsches Patent- und Markenamt

(10) **DE 195 34 943 B4** 2006.01.12

(12) **Patentschrift**

(21) Aktenzeichen: **195 34 943.1**
(22) Anmeldetag: **20.09.1995**
(43) Offenlegungstag: **28.03.1996**
(45) Veröffentlichungstag der Patenterteilung: **12.01.2006**

(51) Int Cl.[8]: ***H03M 7/30*** (2006.01)
H03H 17/02 (2006.01)
G06F 17/14 (2006.01)

Innerhalb von drei Monaten nach Veröffentlichung der Patenterteilung kann nach § 59 Patentgesetz gegen das Patent Einspruch erhoben werden. Der Einspruch ist schriftlich zu erklären und zu begründen. Innerhalb der Einspruchsfrist ist eine Einspruchsgebühr in Höhe von 200 Euro zu entrichten(§ 6 Patentkostengesetz in Verbindung mit der Anlage zu § 2 Abs. 2 Patentkostengesetz).

(30) Unionspriorität:
08/310,146 21.09.1994 US

(73) Patentinhaber:
Ricoh Co., Ltd., Tokio/Tokyo, JP

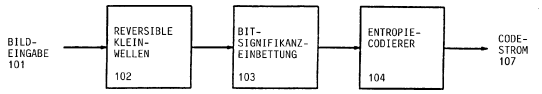
(74) Vertreter:
Schwabe, Sandmair, Marx, 81677 München

(72) Erfinder:
Zandi, Ahmad, Menlo Park, Calif., US; Allen, James D., Menlo Park, Calif., US; Schwartz, Edward L., Menlo Park, Calif., US; Boliek, Martin, Menlo Park, Calif., US

(56) Für die Beurteilung der Patentfähigkeit in Betracht gezogene Druckschriften:
US 52 72 478 A
US 53 81 145
US 50 14 134
Shapiro,J.: An Embedded Hirarchical Image. Coder
Using Zerotrees of Wavelet Coefficients". In: Proc. IEEE Data Compression Conference, S.214-223,
1993;
PENNEBAKER,W.B., et al.: An overview of the basic
principles of the Q-loder adaptive binary arithmetic coder. In: IBM J.Res. Developement, Vol.32, No.6, Nov 1988;

(54) Bezeichnung: **Vorrichtung zur Komprimierung unter Verwendung von eingebetteten Kleinwellen**

(57) Hauptanspruch: Codierer zum Codieren von Eingabedaten zu einem komprimierten Datenstrom, wobei der Codierer aufweist:
einen reversiblen bzw. umkehrbaren Kleinwellenfilter zum Transformieren der Eingabedaten in eine Mehrzahl von Koeffizienten;
einen Einbettungscodierer, der an die reversiblen Kleinwellenfilter bzw. Filter angeschlossen ist, um Einbettungscodieren an der Mehrzahl der Koeffizienten durchzuführen, so daß ein Bitstrom erzeugt wird; wobei das Einbettungscodieren auf die Anordnung der Koeffizienten in einer Reihenfolge bzw. Ordnung gerichtet ist, und
einen Entropiecodierer, der an dem Einbettungscodierer angeschlossen ist bzw. angekoppelt ist, um Entropiecodierung für bzw. an dem Bitstrom durchzuführen, um codierte Daten zu erzeugen.



Beschreibung

[0001] Die vorliegende Erfindung bezieht sich auf den technischen Bereich der Datenkompressions- und -dekompressionssysteme; insbesondere bezieht sich die vorliegende Erfindung auf ein Verfahren und eine Vorrichtung zur verlustfreien und verlustbehafteten Codierung und Decodierung von Daten in Kompressions-/Dekempressionssystemen, wie sie durch die unabhängigen Patentansprüche 1, 2, 4, 9, 12 und 16 umschrieben sind.

[0002] Die Datenkompression ist ein äußerst nützliches Werkzeug zum Speichern und Übertragen großer Datenmengen. Zum Beispiel wird die zur Übertragung eines Bildes, wie etwa eine Faksimileübertragung eines Dokuments, erforderliche Zeit stark reduziert, wenn eine Kompression verwendet wird, um die Anzahl der Bits, die erforderlich ist, um das Bild wiederherzustellen, verringert wird.

Stand der Technik

[0003] Viele verschiedene Datenkompressionstechniken existieren im Stand der Technik. Die Kompressionstechniken können in zwei größere Kategorien unterteilt werden, verlustbehaftete Codierung und verlustlose Codierung. Die verlustbehaftete Codierung bezieht eine Codierung ein, die einen Verlust an Informationen ergibt, so daß es keine Garantie für eine perfekte Wiederherstellung der Originaldaten gibt. Das Ziel der verlustbehafteten Kompression ist, daß Änderungen der Originaldaten in einer solchen Weise vorgenommen werden, daß sie nicht zu beanstanden oder bemerkbar bzw. erfaßbar sind. Bei der verlustlosen Kompression werden sämtliche Informationen zurückbehalten und die Daten werden in einer Weise komprimiert, die die perfekte Wiederherstellung ermöglicht.

[0004] Bei der verlustlosen Kompression werden Eingabesymbole oder Intensitätsdaten in Ausgabecodewörter umgewandelt. Die Eingabe kann Bilddaten, Audiodaten, eindimensionale Daten (z.B. sich räumlich oder zeitlich ändernde Daten), zweidimensionale Daten (z.B. Daten, die sich in zwei räumlichen Richtungen ändern (oder in eine räumliche und eine zeitliche Dimension ändern)) oder vieldimensionale/multispektrale Daten enthalten. Wenn die Kompression erfolgreich ist, werden die Codewörter in weniger Bits dargestellt als es die Anzahl von Bits für die uncodierten Eingabesymbole (oder Intensitätsdaten) erfordert. Verlustlose Codierungsverfahren enthalten Wörterbuchverfahren zur Codierung (z.B. Lempel-Ziv), Lauflängencodieren, aufzählendes Codieren und Entropiecodieren. Bei der verlustlosen Bildkompression wird die Kompression auf Vorhersagen oder Kontexte plus Codieren begründet. Der JBIG-Standard für Faksimilekompression und der DPCM (Differentialpuls-Codemodulation – eine Möglichkeit, bei dem JPEG-Standard) für Bilder mit kontinuierlichen Tönen sind Beispiele für die verlustlose Kompression von Bildern. Bei der verlustbehafteten Kompression werden Eingangssymboldaten oder -intensitätsdaten vor der Umsetzung zu Ausgangscodewörtern quantisiert. Die Quantisierung beabsichtigt, die relevanten Charakteristiken der Daten zu erhalten, während unwichtige Charakteristiken eliminiert werden. Vor der Quantisierung verwenden die verlustbehafteten Systeme häufig eine Transformation, um eine Energiekompaktierung zur Verfügung zu stellen. JPEG ist ein Beispiel für ein verlustbehaftetes Codierungsverfahren für Bilddaten.

[0005] Neuere Entwicklungen bei der Prozessierung von Bilddaten fahren darin fort, die Aufmerksamkeit auf eine Notwendigkeit für effiziente und genaue Arten von Datenkompressionscodierung zu richten. Verschiedene Arten von Transformationssignalprozessierung und/oder pyramidenartiger Signalprozessierung sind vorgeschlagen worden, die eine Mehrlösungs-Pyramidal-Prozessierung und eine Kleinwellen-Pyramidal-Prozessierung umfassen. Auf diese Art wird auch als Unterbandprozessierung und als hierarchische Prozessierung Bezug genommen. Die Kleinwellen-Pyramidal-Prozessierung von Bilddaten ist eine spezifische Art der Mehrlösungs-Pyramidal-Prozessierung, die Quadratur-Spiegelfilter (QMFs) verwenden kann, um eine Unterband-Zergliederung eines Originalbildes zu erzeugen. Es sei zur Kenntnis genommen, daß andere Arten von Nicht-QMF-Kleinwellen existieren. Zum Erhalt von mehr Informationen zur Kleinwellen-Prozessierung siehe Antonini, M., et al. "Image Coding Using Wavelet Transform", IEEE Transactions on Image Processing, Band I, Nr. 2, April 1992, Shapiro, J. "An Embedded Hierachical Image Coder Using Zerotrees of Wavelet Coefficients", Proc. IEEE Data Compression Conference, Seiten 214–223, 1993.

[0006] Ein mit vielen Kleinwellen(wavelet)-Prozessierungen nach dem Stand der Technik verbundenes Problem ist, daß ein großer Speicher erforderlich ist, um sämtliche Daten zu speichern, während sie prozessiert bzw. bearbeitet werden. Mit anderen Worten, bei der Durchführung der Kleinwellen-Prozessierung müssen sämtliche Daten überprüft werden, bevor eine Codierung der Daten durchgeführt wird. In einem solchen Fall gibt es keine Datenausgabe, bis mindestens ein voller Durchlauf durch sämtliche Daten vorgenommen worden ist. Tatsächlich bezieht die Kleinwellen-Prozessierung typischerweise mehrere Durchläufe durch die Daten ein.

Deshalb wird häufig ein großer Speicher benötigt. Es ist wünschenswert, die Kleinwellen-Prozessierung zu verwenden, während das Erfordernis nach einem großen Speicher vermieden werden sollte. Darüber hinaus ist es wünschenswert, die Kleinwellen-Prozessierung nur unter Verwendung eines einzigen Durchlaufs durch die Daten durchzuführen.

[0007] Viele Kleinwellen- oder Unterband-Transformationsumsetzungen erfordern Filter in einer bestimmten kanonischen Form. Zum Beispiel müssen Tief- und Hochpaßfilter die gleiche Länge haben, die Summe der Quadrate der Koeffizienten müssen Eins sein, der Hochpaßfilter muß die zeitliche und frequenzmäßige Umkehrung des Tiefpaßfilters sein usw. (siehe US-Patent Nr. 5,014,134, erteilt im Mai 1991 an Lawton et al.). Es ist wünschenswert, eine breitere Klasse von Filtern zu ermöglichen. Das heißt, es ist wünschenswert, Kleinwellen- oder Subband-Transformationsumsetzungen zur Verfügung zu stellen, die Tief- und Hochpaßfilter verwenden, die nicht die gleiche Länge haben, bei denen die Summe der Quadrate der Koeffizienten nicht Eins sein müssen, bei denen der Hochpaßfilter nicht die zeitliche und frequenzmäßige Umkehrung des Tiefpaßfilters sein muß, usw.

Aufgabenstellung

[0008] Aufgabe der vorliegenden Erfindung ist es eine verbesserte verlustlose und eine verbesserte verlustbehaftete Kompression zur Verfügung zu stellen.

[0009] Die obige Aufgabe wird durch eine Vorrichtung bzw. ein Verfahren gemäß der vorliegenden Erfindung, insbesondere gemäß den Patentansprüchen 1, 2, 4, 9, 12 und 16 gelöst. Zweckmäßige Ausführungsformen der Vorrichtung bzw. Vorrichtungen gemäß der vorliegenden Erfindung bzw. Verfahrensvarianten des Verfahrens gemäß der vorliegenden Erfindung werden durch die in den Unteransprüchen aufgeführten Merkmale zur Verfügung gestellt.

[0010] Vorteilhaft wird eine Transformation verwendet, die eine gute Energiekompaktierung bzw. -verdichtung zur Verfügung stellt. Weiter vorteilhaft wird auch ein Modellieren von räumlich/frequenzmäßig verbundenen Bereichsdaten (Kleinwellen-Transformationsbereich) zur Verfügung gestellt, um eine effiziente Kompression zu ermöglichen. Auch wird eine progressive Übertragung mit durch den Benutzer nach der Codierung auswählbarer Rate oder Verzerrung zur Verfügung gestellt.

[0011] Vorteilhaft wird ein Verfahren und eine Vorrichtung zur Codierung und Decodierung von Daten beschrieben. Vorteilhaft werden ein Verfahren und eine Vorrichtung zum Erzeugen übertragener Signale in Reaktion bzw. Antwort auf Eingabedaten bereitgestellt. Vorteilhaft werden transformierte Signale unter Verwendung einer umkehrbaren bzw. reversiblen Kleinwellen-Transformierung erzeugt. Weiter vorteilhaft werden auch ein Verfahren und eine Vorrichtung zum Komprimieren der transformierten Signale in Daten, die eine verlustlos komprimierte Version der Eingabedaten darstellen, bereitgesgtellt. Vorteilhaft werden die Eingabedaten unter Verwendung eines umkehrbaren bzw. reversiblen Filters mit nicht minimierter Länge zerlegt. Die Zerlegung kann unter Verwendung mehrerer bzw. multipler eindimensionaler Filter durchgeführt werden.

[0012] Weiter vorteilhaft werden auch ein Verfahren und eine Vorrichtung zur Durchführung einer eingegrabenen bzw. vergrabenen Codierung der transformierten Signale bereitgestellt. Die vergrabene bzw. eingebettete Codierung umfaßt vorteilhaft das Ordnen der Reihen von Koeffizienten und die Durchführung eines bit-signifikanten Einbettens bzw. Vergrabens für die transformierten Signale.

[0013] Weiter vorteilhaft werden auch ein Verfahren und eine Vorrichtung zur Dekomprimierung der verlustfrei komprimierten Version der Eingabedaten in transformierte Signale bereitgestellt. Vorteilhaft wird auch die verlustbehaftete Kompression von Eingabesignalen durch Abschneiden von verlustlos komprimierten Daten zur Verfügung gestellt. Weiter vorteilhaft werden auch ein Verfahren und eine Vorrichtung zum Erzeugen der Eingangsdaten aus den transformierten Signalen in eine rekonstruierte Version der Eingangsdaten unter Verwendung einer inversen reversiblen Kleinwellen-Transformation bereitgestellt.

Ausführungsbeispiel

[0014] Nachfolgend wird die vorliegende Erfindung unter Bezugnahme auf Beispiele und ohne Einschränkungen in den Figuren der beigefügten Darstellung wiedergegeben, in denen sich gleiche Bezugsziffern auf gleiche bzw. entsprechende Elemente beziehen, und in denen:

[0015] Fig. 1A ein Blockdiagramm einer Ausführungsform eines Codierungsabschnitts des Codierungssys-

tems nach der vorliegenden Erfindung ist.

[0016] Fig. 1B ein Blockdiagramm einer Ausführungsform des bit-signifikanten Einbettens bzw. Vergrabens nach der vorliegenden Erfindung ist.

[0017] Fig. 2A ein Blockdiagramm eines Kleinwellen-Analyse-/-Synthesesystems ist.

[0018] Fig. 2B vorwärts gerichtete und rückwärts gerichtete Darstellungen von Transformationssystemen zum Filtern mit nicht überlappten reversiblen Filtern minimaler Länge darstellt.

[0019] Fig. 3A–D Ergebnisse der Durchführung einer Zerlegung auf vier Niveaus bzw. in vier Stufen darstellt.

[0020] Fig. 4A ein Blockdiagramm einer Pyramidal-Transformation in vier Niveaus bzw. vier Stufen ist.

[0021] Fig. 4B ein Blockdiagramm einer zweidimensionalen Transformation in zwei Stufen bzw. auf zwei Niveaus ist.

[0022] Fig. 4C ein Blockdiagramm ist, das eindimensionale Filter darstellt, die eine Dekompression mit Mehrfachauflösung durchführen.

[0023] Fig. 4D ein Blockdiagramm eines Systems ist, das reversible Kleinwellen gemäß der Erfindung verwendet.

[0024] Fig. 4E Blockdiagramme von Verstärkungs- und Analysesystemen, die die reversiblen Kleinwellen gemäß der vorliegenden Erfindung verwenden sind.

[0025] Fig. 5 eine Baumstruktur auf bzw. für Kleinwellen-Koeffizienten darstellt.

[0026] Fig. 6A und Fig. 6A (fortgesetzt) ein Flußdiagramm einer Ausführungsform der Einzellisten-Nullbaum-Modellierung zur Codierung gemäß der vorliegenden Erfindung ist.

[0027] Fig. 6B und Fig. 6B (fortgesetzt) ein Flußdiagramm einer Ausführungsform der Einzellisten-Nullbaum-Modellierung zur Codierung gemäß der vorliegenden Erfindung unter Verwendung eines verringerten Flag-Speichers ist.

[0028] Fig. 6C ein Flußdiagramm einer Ausführungsform einer Einzellisten-Nullbaum-Modellierung zum Decodieren gemäß der vorliegenden Erfindung ist.

[0029] Fig. 6D ein Flußdiagramm einer Ausführungsform der Einzellisten-Nullbaum-Modellierung zum Decodieren gemäß der vorliegenden Erfindung unter Verwendung eines verringerten Flag-Speichers ist.

[0030] Fig. 7A ein Flußdiagramm einer Ausführungsform des horizontalen bzw. Horizontmodellierens zum Decodieren gemäß der vorliegenden Erfindung unter Verwendung eines verkleinerten bzw. verringerten Flag-Speichers ist.

[0031] Fig. 8A ein Flußdiagramm einer Ausführungsform des B-Durchlaufs zur Codierung gemäß der vorliegenden Erfindung ist.

[0032] Fig. 8B ein Flußdiagramm einer Ausführungsform des B-Durchlaufs zur Codierung gemäß der vorliegenden Erfindung unter Verwendung eines verringerten bzw. reduzierten Flag-Speichers ist.

[0033] Fig. 9A ein Flußdiagramm einer Ausführungsform des B-Durchlaufs zum Decodieren gemäß der vorliegenden Erfindung ist.

[0034] Fig. 9B ein Flußdiagramm einer Ausführungsform des B-Durchlaufs zum Decodieren gemäß der vorliegenden Erfindung unter Verwendung eines verringerten bzw. reduzierten Flag-Speichers ist.

[0035] Fig. 10 eine Ausführungsform des Vorwärts-Kleinwellenfilters gemäß der vorliegenden Erfindung ist.

[0036] Fig. 11 ein Blockdiagramm einer Ausführungsform eines umgekehrten Kleinwellenfilters gemäß der

vorliegenden Erfindung ist.

[0037] Fig. 12 ein Bild und Koeffizienten in einem Zeilenpuffer für eine Pyramidal-Zerlegung auf vier Niveaus bzw. in vier Stufen darstellt.

[0038] Fig. 13 ein Blockdiagramm einer Ausführungsform des Kleinwellenfilterns unter Verwendung einer Filtersteuereinheit ist.

[0039] Fig. 14 ein Blockdiagramm einer anderen Ausführungsform einer Kleinwellenfilterung unter Verwendung einer Filtersteuereinheit ist.

[0040] Fig. 15 die Zuordnung von Speichergruppen bzw. Speicherbänken darstellt, um einen horizontalen und einen vertikalen Zugriff zu unterstützen.

[0041] Fig. 16 den Filterbetrieb für eine Zerlegung in zwei Niveaus bzw. zwei Stufen darstellt.

[0042] Fig. 17 ein Blockdiagramm einer Ausführungsform des Kontextmodells gemäß der vorliegenden Erfindung ist.

[0043] Fig. 18 ein Blockdiagramm einer Ausführungsform der Vorzeichen- bzw. Zeichen-/Größenordnungseinheit gemäß der vorliegenden Erfindung ist.

[0044] Fig. 19 ein Blockdiagramm einer Ausführungsform der Größenordnungs-Speichereinheit gemäß der vorliegenden Erfindung ist.

[0045] Fig. 20 ein Blockdiagramm einer Ausführungsform der Wertigkeits- bzw. Wichtigkeitseinheit gemäß der vorliegenden Erfindung ist.

[0046] Fig. 21 ein Blockdiagramm einer Ausführungsform der Baum-Speichereinheit gemäß der vorliegenden Erfindung ist.

[0047] Fig. 22 ein Blockdiagramm einer Ausführungsform der Koeffizientenverschiebung gemäß der vorliegenden Erfindung ist.

[0048] Fig. 23 ein Blockdiagramm einer alternativen Ausführungsform der Wertigkeits- bzw. Wichtigkeitseinheit gemäß der vorliegenden Erfindung ist, unter Verwendung einer Ausrichtung bzw. einem Abgleich um bzw. auf 1,5.

[0049] Fig. 24 die dynamische Zuweisung vom Speicher mit codierten Daten für einen Durchlaufbetrieb darstellt.

[0050] Fig. 25A und B ein Flußdiagramm einer Ausführungsform des Codierungsverfahrens nach der vorliegenden Erfindung ist.

[0051] Fig. 26A und B ein Flußdiagramm der Decodierung nach einer Ausführungsform des Decodierungsverfahrens gemäß der vorliegenden Erfindung ist.

[0052] Fig. 27A und B ein Flußdiagramm einer Ausführungsform des Verfahrens zur Modellierung jedes Koeffizienten sowohl für das Codierungs- als auch das Decodierungsverfahren gemäß der vorliegenden Erfindung ist.

[0053] Fig. 28A und B Flußdiagramme einer alternativen Ausführungsform des Codierungsverfahrens gemäß der vorliegenden Erfindung ist.

[0054] Fig. 29A und B ein Flußdiagramm einer alternativen Ausführungsform des Decodierungsverfahrens gemäß der vorliegenden Erfindung ist.

[0055] Fig. 30A und B ein Flußdiagramm einer alternativen Ausführungsform des Verfahrens zur Modellierung jedes Koeffizienten in den Codierungs- und Decodierungsverfahren gemäß der vorliegenden Erfindung ist.

[0056] Fig. 31 eine Ausführungsform des Multiplizierers bzw. Vervielfachers für das Frequenzband ist, das für die Koeffizientenzuordnung bzw. -ausrichtung gemäß der vorliegenden Erfindung verwendet wird.

[0057] Ein Verfahren und eine Vorrichtung zur Kompression und Dekompression wird hier nachfolgend beschrieben. In der nachfolgenden detaillierten Beschreibung der vorliegenden Erfindung werden zahlreiche spezifische Einzelheiten zum Ausdruck gebracht, wie etwa Arten von Codierereinrichtungen, Anzahlen von Bits, Namen von Signalen usw., um ein tiefgehendes Verständnis der vorliegenden Erfindung zu ermöglichen. Jedoch wird es für den Fachmann klar werden, daß die vorliegende Erfindung ohne diese spezifischen Einzelheiten in die Tat umgesetzt werden kann. In anderen Beispielen werden wohlbekannte Konstruktionen und Vorrichtungen eher in Blockdiagrammform als in Einzelheiten gezeigt, um eine Verundeutlichung der vorliegenden Erfindung zu vermeiden.

[0058] Einige Abschnitte der detaillierten Beschreibung werden in Ausdrücken von Algorithmen und von symbolischen Darstellungen von Operationen auf Datenbits innerhalb eines Computerspeichers präsentiert. Diese algorithmischen Beschreibungen und Darstellungen sind die Mittel, die von den Fachleuten auf dem Gebiet der Datenverarbeitung verwendet werden, um den Gegenstand ihrer Tätigkeit anderen Fachleuten am effektivsten zuzutragen. Ein Algorithmus wird hier und im allgemeinen als eine selbstkonsistente Folge von Schritten zu verstehen sein, die zu einem gewünschten Ergebnis führt. Die Schritte sind derartig, daß sie physikalische Manipulationen und physikalische Größen erfordern. Üblicherweise haben diese Größen die Form von elektrischen oder magnetischen Signalen, die geeignet sind, gespeichert, übertragen, kombiniert, verglichen und auf andere Weise manipuliert zu werden, obwohl dies nicht notwendig ist. Es hat sich zur Zeit prinzipiell aus Gründen der üblichen Verwendungen als günstig erwiesen, auf diese Signale als Bits, Werte, Elemente, Symbole, Schriftzeichen, Ausdrücke, Nummern bzw. Zahlen oder dergleichen Bezug zu nehmen.

[0059] Es sollte jedoch beachtet werden, daß all diese und ähnliche Ausdrücke mit den angemessenen physikalischen Größen in Verbindung zu sehen sind und nichts weiter als geeignete Oberbegriffe sind, die diesen Größen zugeordnet sind. Wenn es nicht besonders anders ausgesagt wird, wie aus der nachfolgenden Erörterung klar wird, ist es bevorzugt, daß für die gesamte vorliegende Erfindung Erörterungen, die Ausdrücke, wie etwa "Prozessieren" oder "Berechnen" bzw. "Veranschlagen" oder "Kalkulieren" oder "Bestimmen" oder "Anzeigen" oder dergleichen, verwenden, sich auf Tätigkeiten oder Prozesse eines Computersystems oder ähnliche bzw. gleiche elektronische Computereinrichtungen beziehen, die Daten, die als physikalische (elektronische) Größen innerhalb der Register und Speicher des Computersystems dargestellt werden, in andere ähnliche bzw. vergleichbar dargestellte Daten als physikalische Größen innerhalb des Speichers oder Registers des Computersystems oder anderer solcher Informationsspeicher, Übertragungs- oder Wiedergabeeinrichtungen manipulieren und übertragen.

[0060] Die vorliegende Erfindung bezieht sich auch auf eine Vorrichtung zur Durchführung der Operationen hierin. Die Vorrichtung kann für die erforderlichen Zwecke besonders aufgebaut sein oder sie kann einen Computer für allgemeine Zwecke umfassen, der selektiv aktiviert oder durch ein Computerprogramm, das in dem Computer gespeichert ist, rekonfiguriert werden. Die Algorithmen und Anzeigen, die hierin dargestellt werden, sind nicht inhärent auf bestimmte Computer oder andere Vorrichtungen bezogen. Verschiedene Vorrichtungen für allgemeine Zwecke können mit diesen Programmen in Übereinstimmung mit den Lehren hierin verwendet werden, oder es kann sich als bevorzugt erweisen, spezialisiertere Vorrichtungen aufzubauen, um die erforderlichen Verfahrensschritte durchzuführen. Die erforderliche Konstruktion für verschiedene dieser Vorrichtungen wird aus der nachfolgenden Beschreibung erscheinen. Zusätzlich wird die vorliegende Erfindung nicht unter Bezugnahme auf irgendeine bestimmte Programmiersprache beschrieben. Es wird zu bevorzugen sein, daß viele verschiedene Programmiersprachen verwendet werden können, um die Lehren der Erfindung in die Tat umzusetzen, wie sie hierin beschrieben wird.

[0061] Nachfolgend wird ein Überblick über die Erfindung gegeben. Die vorliegende Erfindung stellt ein Kompressions-/Dekompressionssystem zur Verfügung, das einen Codierungsabschnitt und einen Decodierungsabschnitt aufweist. Der Codierungs- bzw. Verschlüsselungsabschnitt ist für die Codierung von Eingabe- bzw. Eingangsdaten verantwortlich, um komprimierte Daten zu erzeugen, während der Decodierungsabschnitt für die Decodierung zuvor codierter Daten zuständig ist, um eine rekonstruierte bzw. wieder aufgebaute Version der Originaleingabedaten zu erzeugen. Die Eingabedaten können verschiedene Datentypen aufweisen, wie etwa Bilddaten (stehende oder Videodaten), Audiodaten usw. In einer Ausführungsform sind die Daten digitale Signaldaten; jedoch sind analog digitalisierte Daten, Textformatdaten und andere Formate möglich. Die Quelle der Daten kann ein Speicher oder Kanal für den Codierungsabschnitt und/oder den Decodierungsabschnitt sein.

[0062] In der vorliegenden Erfindung können Elemente des Codierungsabschnittes und/oder des Decodierungsabschnittes in Hardware oder Software umgesetzt werden, so wie sie für ein Computersystem verwendet werden. Die vorliegende Erfindung stellt ein verlustfreies Kompressions-/Dekompressionssystem zur Verfügung. Die vorliegende Erfindung kann auch konfiguriert werden, um eine verlustbehaftete Kompression/Dekompression durchzuführen.

[0063] Die Fig. 1A ist ein Blockdiagramm einer Ausführungsform des Codierungsabschnitts des Systems. Man beachte, daß der Decodierungsabschnitt des Systems in der umgekehrten Ordnung gleichermaßen mit dem Datenfluß operiert. Bezugnehmend auf **Fig.** 1 werden Eingabebilddaten **101** durch den Kleinwellen-Übertragungsblock **102** empfangen. Der Ausgang des Kleinwellen-Übertragungsblocks **102** wird zu einem Bit-Wichtungs-Einbettungsblock **103** gekoppelt. In Antwort auf die Ausgabe von dem Kleinwellen-Übertragungsblock **102** erzeugt der Bit-Wichtungs-Einbettungsblock **103** mindestens einen Bitstrom, der durch einen Entropiecodierer **104** empfangen wird. In Antwort auf die Eingabe von der Bit-Wichtungs-Einbettung **103** erzeugt der Entropiecodierer **104** einen Codestrom **107**.

[0064] In einer Ausführungsform weist eine Bit-Signifikanzeinbettung **103** ein frequenzbasierendes Kontextmodell **105** und ein verbundenes Raum-/Frequenz-Kontextmodell **106**, wie in Fig. 1B gezeigt, auf. In einer Ausführungsform weist das verbundene Raum-/Frequenz-Kontextmodell **106** ein Horizontal-Kontextmodell auf. In einigen Ausführungsformen weist ein frequenzbasierender Kontektmodellblock **105** ein Nullbaum-Modell auf. In einer anderen Ausführungsform weist das frequenzbasierende Kontextmodell **105** ein Wichtungsbaummodell auf. Die Zeichen-Größenordnungseinheit **109**, das frequenzbasierende Kontextmodell **105** und das verbundene Raum-/Frequenz-(JSF)-Kontextmodell **106** führen in der vorliegenden Erfindung eine Bit-Wichtungs-Einbettung durch. Die Eingabe der Zeichen-Größenordnungseinheit **109** wird zu dem Ausgang des Kleinwellen-Transformations-Codierungsblocks **102** gekoppelt. Der Ausgang der Zeichen-Größenordnungseinheit **109** wird zu einem Schalter **108** gekoppelt bzw. übertragen. Der Schalter **108** wird gekoppelt, um den Ausgang der Zeichen-Größenordnungseinheit **109** zu einem Eingang entweder eines frequenzbasierenden Modellierungsblocks **105** oder eines verbundenen Raum-/Frequenz-Modellierungsblocks **106** zur Verfügung zu stellen. Der Ausgang des frequenzbasierenden Codierungsblocks **105** und des Horizontal-Ordnungs-Codierungsblocks **106** werden zu dem Eingang des Entropiecodierers **104** gekoppelt. Der Entropiecodierer **104** erzeugt den Ausgangscodestrom **107**.

[0065] Zurückgehend auf Fig. 1A werden gemäß der vorliegenden Erfindung die Bilddaten **101** empfangen und unter Verwendung reversibler Kleinwellen im Kleinwellen-Transformationsblock **102**, wie später unten bestimmt wird, codiert transformiert, um eine Reihe von Koeffizienten zu erzeugen, die eine Mehrfachauflösungs-Zerlegung des Bildes darstellen. Diese Koeffizienten werden durch die Bit-Wichtungs-Einbettung **103** empfangen.

[0066] Die Wichtungs-Einbettung **103** fordert an und konvertiert die Koeffizienten in ein Zeichen-Größenordnungsformat und basierend auf ihrer Wichtung (wie unten später beschrieben wird) werden die formatierten Koeffizienten einer Kombination von verschiedenen Einbettungs-Modellierungsverfahren ausgesetzt. Gemäß der vorliegenden Erfindung werden die formatierten Koeffizienten einem von zwei Einbettungs-Modellierungsverfahren ausgesetzt (z.B. frequenzbasierendem Modellieren und JSF-Modellieren).

[0067] In einer Ausführungsform werden die formatierten Koeffizienten entweder einem frequenzbasierenden Modellieren oder einer verbundenen Raum-/Frequenz-Modellierung ausgesetzt. Wenn die eingegebenen Daten Bilddaten aufweisen, die mehrere Bitebenen haben, werden gemäß der vorliegenden Erfindung eine Anzahl von Bitebenen mit frequenzbasierender Modellierung codiert, während die verbleibenden Bitebenen mit JSF-Modellierung codiert werden. Die Entscheidung, welches Verfahren für welche Bitebenen verwendet wird, kann ein vom Benutzer bestimmter Parameter bzw. ein Benutzerparameter sein. In einer Ausführungsform werden die Bitebenen höherer Ordnung von Koeffizienten angefordert und codiert mit der frequenzbasierenden Modellierung nach der vorliegenden Erfindung. In dem frequenzbasierenden Kontext-Modellverfahren nach der vorliegenden Erfindung bezieht sich die Vorhersage der Wichtung bzw. Wertigkeit der Koeffizientenbits auf die Pyramidal-Struktur der Klein- bzw. Unterwellen. Die Koeffizienten-Bitebenen niederer Ordnung werden geordnet und mit dem verbundenen Raum-/Frequenz-Kontextmodell nach der vorliegenden Erfindung codiert. Das JSF-Modellieren, z.B. das Horizontal-Modellieren, stellt gegenüber dem frequenzbasierenden Codieren von Bitebenen, die weniger im Hinblick auf die Frequenzbereich-Koeffizientverhältnisse korreliert sind, Vorteile zur Verfügung.

[0068] Das Ergebnis von Bit-Wichtungs-Einbettung sind Entscheidungen (oder Symbole), die durch den Entropiecodierer zu codieren sind. In einer Ausführungsform werden sämtliche Entscheidungen zu einem einzi-

gen Codierer geleitet. In einer anderen Ausführungsform werden Entscheidungen durch Wichtung beschriftet bzw. bewertet, und Entscheidungen für jeden Wichtungspegel werden durch verschiedene (physikalische oder virtuelle) Mehrfachcodierer verarbeitet.

[0069] Die Bitströme, die aus dem frequenzbasierenden Kontext-Modellblock **105** und dem JSF-Kontext-Modellblock **106** resultieren, werden in der Ordnung ihrer Wichtung unter Verwendung des Entropiecodierers **104** codiert. In einer Ausführungsform weist der Entropiecodierer **104** einen Binär-Entropiecodierer auf. In einer Ausführungsform weist der Entropiecodierer **104** einen Q-Codierer, einen B-Codierer, der in dem US-Patent Nr. 5,272,478 A definiert ist, oder einen Codierer, wie etwa in der US-Patentanmeldung mit der Serial-Nr. 08/016,035 beschrieben ist, betitelt mit "Method and Apparatus for Parallel Decoding and Encoding of Data", angemeldet am 10. Februar 1993, auf. Für mehr Informationen zu dem Q-Codierer, siehe Pennebaker, W.B., et al. "An Overview of the Basic Principles of the Q-coder Adaptive Binary Arithmetic", IBM Journal of Research and Development, Vol. 32, Seiten 717–26, 1988. In einer Ausführungsform erzeugt ein einzelner Codierer einen einzelnen Ausgangscodestrom. In einer anderen Ausführungsform erzeugen mehrere (physikalische oder virtuelle) Codierer mehrere (physikalische oder virtuelle) Datenströme.

Kleinwellenzerlegung

[0070] Die vorliegende Erfindung führt anfänglich eine Zerlegung eines Bildes (in der Form von Bilddaten) oder anderer Datensignale unter Verwendung reversibler Klein- bzw. Unterwellen durch. Gemäß der vorliegenden Erfindung weist eine reversible Klein- bzw. Nebenwellen-Transformation eine Umsetzung eines genauen Wiederherstellungssystems in ganzzahliger Arithmetik auf, so daß ein Signal mit ganzzahligen Koeffizienten verlustfrei zurückgewonnen werden kann. Unter Verwendung reversibler Kleinwellen ist die vorliegende Erfindung auch dazu in der Lage, eine verlustfreie Kompression mit endlicher Präzisionsarithmetik zur Verfügung zu stellen. Die Ergebnisse, die durch Anwenden der reversiblen Kleinwellen-Transformation auf die Bilddaten erzeugt werden, sind eine Reihe von Koeffizienten. In einer Ausführungsform der vorliegenden Erfindung wird die reversible Kleinwellen-Transformation unter Verwendung einer Einstellung von Filtern in die Tat umgesetzt. Gemäß einer Ausführungsform sind die Filter ein Tiefpaßfilter mit zwei Gängen bzw. Abgriffen oder Transformatorstufen ("two-tap") und ein Hochpaßfilter mit sechs Abgriffen bzw. Abzweigen oder Transformierungsstufen. In einer Ausführungsform werden diese Filter nur unter Verwendung von Additions- und Subtraktionsoperationen (und verdrahteter Bitverschiebung) in die Tat umgesetzt. Auch erzeugt der Hochpaßfilter gemäß der vorliegenden Erfindung seinen Ausgang unter Verwendung der Ergebnisse des Hochpaßfilters. Die sich ergebenden Hochpaßkoeffizienten sind nur ein paar Bits größer als die Bildpunktauflösung und die Tiefpaßkoeffizienten sind die gleichen wie die Bildpunktauflösung. Weil nur die Tiefpaßkoeffizienten wiederholt in einer pyramidalen Zerlegung gefiltert werden, wird die Auflösung nicht in mehrstufige bzw. mehrpegelige Zerlegungen erweitert.

[0071] Ein Kleinwellen-Transformationssystem wird durch ein Paar von FIR-Analysefiltern $h_0(n)$, $h_1(n)$ bestimmt und ein Paar von FIR-Synthesefiltern $g_0(n)$, $g_1(n)$ definiert. Gemäß der vorliegenden Erfindung sind h_0 und g_0 die Tiefpaßfilter und h_1 und g_1 sind die Hochpaßfilter. Ein Blockdiagramm des Kleinwellensystems ist in Fig. 2A gezeigt. Bezugnehmend auf Fig. 2A werden für ein Eingangssignal x(n) die Analysefilter h_0 und h_1 angelegt und die Ausgänge werden um 2 verringert (kritisches Subabtasten), um die transformierten Signale $y_0(n)$ und $y_1(n)$ zu erzeugen, auf die hierin als Tiefpaß- bzw. Hochpaßkoeffizienten Bezug genommen wird. Die Analysefilter und ihre entsprechend verringerten oder unterabgetasteten Blöcke bilden den Analyseabschnitt des Kleinwellen-Transformationssystems. Der Codierer/Decodierer enthält sämtliche Verarbeitungslogik und -routinen bzw. -prozeduren, die in dem transformierten Bereich durchgeführt werden (z.B. Vorhersage, Quantisierung, Codierung etc.). Das in Fig. 2A gezeigte Kleinwellensystem enthält auch einen Syntheseabschnitt, indem die Transformationssignale um 2 aufwärts abgetastet sind (z.B. wird nach jedem zweiten Ausdruck eine Null eingefügt) und anschließend durch die Synthesefilter $g_0(n)$ und $g_1(n)$ hindurchgeführt. Die Tiefpaßkoeffizienten $y_0(n)$ werden durch den Tiefpaßsynthesefilter g_0 und das hochpaßgefilterte $y_1(n)$ wird durch Hochpaßfilter g_1 hindurchgeführt. Der Ausgang der Filter $g_0(n)$ und $g_1(n)$ wird zur Erzeugung von x(n) kombiniert.

[0072] Während in einigen Ausführungsformen Ab- und Aufwärtsabtastung durchgeführt werden, werden in anderen Ausführungsformen Filter verwendet, so daß Berechnungen, die wegen des Ab- und Aufwärtsabtastens unnötig sind, nicht durchgeführt werden.

[0073] Das Kleinwellensystem kann in Ausdrücken von Z-Transformationen beschrieben werden, wobei X(Z), $\hat{X}(Z)$ die Eingangs- bzw. Ausgangssignale sind, wobei $Y_0(Z)$, $Y_1(Z)$ die Tiefpaß- und Hochpaß-Transformationssignale sind, $H_0(Z)$, $H_1(Z)$ die Hochpaß- und die Tiefpaßanalysefilter und schließlich $G_0(Z)$, $G_1(Z)$ die Tiefpaß- und die Hochpaßsynthesefilter sind. Falls keine Änderung oder Quantisierung in dem Transformationsbereich

vorkommt, ist der Ausgang $\hat{X}(Z)$ gemäß **Fig.** 2 gegeben durch

$$X(Z) = \frac{1}{2}[H_0(Z)G_0(Z)+H_1(Z)G_1(Z)]X(Z)+\frac{1}{2}[H_0(-Z)G_0(Z)+H_1(-Z)G_1(Z)]X(-Z).$$

[0074] Gemäß der vorliegenden Erfindung wird der zweite Ausdruck von $\hat{X}(Z)$, auf den als "Faltungs"(aliasing)-Ausdruck Bezug genommen wird, gestrichen bzw. abgeschafft, weil die Synthesefilter als der Quadraturspiegel bzw. Quertransformationsspiegel der Analysefilter definiert sind, nämlich

$$\begin{cases} G_0(Z)=H_1(-Z) \\ G_1(Z)=-H_0(-Z) \end{cases}$$

[0075] In den Ausdrücken der Filterkoeffizienten heißt dies

$$\begin{cases} g_0(n)=(-1)^n h_1(n) \\ g_1(n)=-(-1)^n h_0(n) \end{cases}$$

[0076] Deshalb lautet der Ausgang eines Quadraturspiegel- bzw. Quertransformationsfilterpaares nach der Substitution bzw. Ersetzung:

$$\hat{X}(Z) = \frac{1}{2}[H_0(Z)H_1(-Z)-H_1(Z)H_0(-Z)]X(Z).$$

[0077] Folglich wird bei einem Querspiegel- bzw. Quertransformationssystem nach der vorliegenden Erfindung der Ausgang nur in Ausdrücken der Analysefilter bestimmt. Die Kleinwellen-Transformation wird rekursiv auf das transformierte Signal angewendet, indem die durch die Filter erzeugten Ausgänge direkt oder indirekt als Eingänge in die Filter verwendet werden. Bei der beschriebenen Ausführungsform wird nur die tiefpaßtransformierte Komponente $y_0(n)$ rekursiv bzw. wiederholbar transformiert, so daß das System pyramidal bzw. in Form einer Pyramide ist. Ein Beispiel eines solchen pyramidalen Systems ist in **Fig. 4A** gezeigt.

[0078] Die Z-Transformation ist eine angemessene Notation, um die Operation in Form von Daten auf Hardware und/oder auf Software auszudrücken. Die Multiplikation mit Z^{-m} modelliert bzw. stellt dar einen m Verzögerungszeitzyklus bzw. -zeitgeberzyklus in der Hardware und einen Zugriff auf die Anordnung des m-ten davorliegenden Elements in der Software. Derartige Umsetzungen von Hardware enthalten Speicher, Rohr- bzw. Leitungsstufen, Verschieber, Register usw.

[0079] Gemäß der vorliegenden Erfindung sind die Signale x(n) und $\hat{x}(n)$ identisch bis zu einer multiplikativen Konstante und einem Verzögerungsausdruck, nämlich in Ausdrücken der Z-Transformation

$\hat{X}(Z) = cZ^{-m}X(Z)$.

[0080] Dies wird als ein genaues Wiederherstellungssystem bezeichnet. Folglich ist in einer Ausführungsform gemäß der vorliegenden Erfindung die Kleinwellen-Transformation, die eingangs auf die Eingabedaten angewendet wird, genau wiederherstellbar.

[0081] Eine Ausführungsform gemäß der vorliegenden Erfindung, die die Hadamard-Transformation verwendet, ein genaues Wiederherstellungssystem, welches in der normalisierten Form die nachfolgende Darstellung in dem Z-Bereich aufweist:

$$\begin{cases} H_0(Z) = \dfrac{1}{\sqrt{2}}(1+Z^{-1}) \\ H_1(Z) = \dfrac{1}{\sqrt{2}}(1-Z^{-1}) \end{cases}$$

[0082] Nach der Ersetzung lautet der Ausgang
$\hat{X}(Z)=Z^{-1}X(Z)$,

welche deutlicherweise eine genaue Wiederherstellung ist. Um mehr Informationen zu der Hadamard-Transformation zu erhalten, siehe Anil K. Jain, Fundamentals of Image Processing, Seite 155.

[0083] Auf eine reversible bzw. umkehrbare Version der Hadamard-Transformation wird hierin als die S-Transformation Bezug genommen. Für mehr Informationen zu der S-Transformation siehe Said, A. und Pearlman, W. "Reversible Image Compression via Multiresolution Representation and Predictive Coding", Dept. of Electrical, Computer and Systems Engineering, Renssealaer Polytechnic Institute, Troy, NY 1993. Da die Hadamard-Transformation eine genaue Wiederherstellungstransformation ist, ist die nachfolgende unnormierte Version (die sich von der Hadamard-Transformation um konstante Faktoren unterscheidet) ebenfalls eine genaue Wiederherstellungstransformation:

$$\begin{cases} h_0(Z) = \dfrac{1}{2}(1+Z^{-1}) \\ h_1(Z) = 1-Z^{-1} \end{cases}$$

[0084] Es seien die Proben der Eingangssignale als x_0, x_1 gegeben, wobei die S-Transformation eine reversible bzw. umkehrbare Implementation dieses Systems ist, als

$$\begin{cases} y_0(0) = \lfloor (x(0)+x(1))/2 \rfloor \\ y_1(0) = x(0)-x(1) \end{cases}$$

[0085] Die Notation $\lfloor . \rfloor$ bedeutet abzurunden oder zu kürzen bzw. abzuschneiden, und darauf wird manchmal als "Boden- bzw. Grundfunktion" (floor function) Bezug genommen. Ähnlich bedeutet die Dach- bzw. Deckenfunktion (ceiling function) $\lceil . \rceil$, daß auf die nächste ganze Zahl aufgrundet wird.

[0086] Der Beweis, daß diese Implementation bzw. Umsetzung reversibel ist, folgt aus der Tatsache, daß die einzige Information, die bei der Annäherung verlorengeht, das letzte signifikante Bit von x(0) + x(1) ist. Aber da die letzten signifikanten Bits von x(0) + x(1) und x(0) – x(1) identisch sind, kann dies von dem Hochpaßausgang $y_1(0)$ zurückgewonnen werden. Mit anderen Worten

$$\begin{cases} x(0) = y_0(0) + \lfloor (y_1(0)+1)/2 \rfloor \\ x(1) = y_0(0) - \lceil (y_1(0)-1)/2 \rceil \end{cases}$$

[0087] Die S-Transformation ist eine nicht-überlappende Transformation, die reversible Filter mit minimaler Länge verwendet. Filter mit minimaler Länge weisen ein Paar von Filtern auf, wobei sämtliche Filter zwei Abgriffe bzw. Stufen aufweisen. Transformationen mit minimaler Länge stellen keine gute Energie-Kompaktierung zur Verfügung. Filter mit minimaler Länge setzen eine nicht-überlappende Transformation in die Tat um, weil die Länge der Filter gleich der Anzahl der Filter ist. Überlappende Transformationen verwenden zumindest einen Filter, der eine Länge aufweist, die größer ist als die Anzahl der Filter. Überlappende Transformationen, die längere Filter verwenden (nicht-minimaler Länge), können eine bessere Energie-Kompaktierung zur Verfügung stellen. Die vorliegende Erfindung stellt reversible Filter mit nicht-minimaler Länge zur Verfügung, die eine überlappende Transformation ermöglichen.

[0088] Ein anderes Beispiel eines genauen Wiederherstellungssystems weist die 2/6-(TS)-Transformation auf, die die Z-Bereich-Definition hat

$$H_0(Z) = \frac{1}{\sqrt{2}}(1+Z^{-1})$$

$$H_1(Z) = \frac{1}{8\sqrt{2}}(-1-Z^{-1}+8Z^{-2}-8Z^{-3}+Z^{-4}+Z^{-5})$$

[0089] Nach dieser Substitution lautet der Ausgang

$\hat{X}(Z) = 2Z^{-3}X(Z)$,

die eine genaue bzw. exakte Wiederherstellungstransformation ist.

[0090] Die rationale, unormierte Version der TS-Transformation weist auf:

$$\begin{cases} h_0(Z) = \frac{1}{2}(1+Z^{-1}) \\ h_1(Z) = \frac{1}{8}(-1-Z^{-1}+8Z^{-2}-8Z^{-3}+Z^{-4}+Z^{-5}) \end{cases}$$

[0091] Wenn x(0), x(1) ... x(5) sechs Proben der Signale sind, dann sind die ersten drei Tiefpaßkoeffizienten $y_0(0)$, $y_0(1)$, $y_0(2)$ und der erste Hochpaßkoeffizient $y_1(0)$ gegeben durch:

$$\begin{cases} y_0(0) = \lfloor (x(0)+x(1))/2 \rfloor \\ y_0(1) = \lfloor (x(2)+x(3))/2 \rfloor \\ y_0(2) = \lfloor (x(4)+x(5))/2 \rfloor \end{cases}$$

$$y_1(0) = \lfloor (-(x(0)+x(1)))+8(x(2)-x(3))+(x(4)+x(5))/8 \rfloor .$$

[0092] Jedoch ist die direkte Vorwärts-Umsetzung bzw. -Implementation der rationalen unnormierten Version der TS-Transformation nicht umgekehrbar. Das folgende Beispiel zeigt, daß die Implementation lokal nicht umkehrbar ist. Eine längere Folge kann wie ein Beispiel für den allgemeinen Fall aufgebaut werden. Da $-(x(0) + x(1)) + (x(4) + x(5)) \neq -y_0(0) + y_0(2)$ ist, weil zum Berechnen von $y_0(0)$ und $y_0(2)$ gerundet wird, ist diese Transformation unter Verwendung von lokalen Informationen nicht umkehrbar bzw. reversibel.

[0093] Zum Beispiel, falls x(0) = 1,x(1) = 1,x(2) = 3,x(3) = 1,x(4) = 1,x(5) = 1, dann

$y_0(0) = \lfloor (1 + 1)/2 \rfloor = 1$

$y_0(1) = \lfloor (3 + 1)/2 \rfloor = 2$

$y_0(2) = \lfloor (1 + 1)/2 \rfloor = 1$

$y_1(0) = \lfloor [-(1 + 1) + 8(3 – 1) + (1 + 1)]/8 \rfloor \lfloor (–2 + 16 + 2)/8 \rfloor = 2$

und wenn x(0) = 1,x(1) = 2,x(2) = 4,x(3) = 1,x(4) = 1,x(5) = 1, dann

$y_0(0) = \lfloor (1 + 2)/2 \rfloor = 1$

$y_0(1) = \lfloor (4 + 1)/2 \rfloor = 2$

$y_0(2) = \lfloor (1 + 1)/2 \rfloor = 1$

$y_1(0) = \lfloor [-(1 + 2) + 8(4 – 1) + (1 + 1)] \rfloor /8 = \lfloor (–3 + 24 + 2)/8 \rfloor = \lfloor 23/8 \rfloor = 2$

[0094] Da $y_0(0)$, $y_0(1)$, $y_0(2)$ und $y_0(0)$ die gleichen für die zwei verschiedenen Sätze von Eingängen x(0) ... x(5) sind, ist die Transformation nicht reversibel, da bei gegebenen $y_0(0)$, ... $y_1(0)$ nicht aus dieser lokalen In-

formation bestimmt werden kann, welcher der zwei Sätze eingegeben worden ist. (Man registriere, daß bewiesen werden kann, daß die Transformation unter Verwendung allgemeiner Informationen von sämtlichen Koeffizienten nicht reversibel ist.)

[0095] Man ziehe nun ein reversible TS-Transformation in Betracht, auf die hierin als eine RTS-Transformation Bezug genommen wird, die einen verschiedenen Hochpaßfilterbetrieb zur Verfügung stellt.

[0096] Falls x(0), x(1), x(2), x(3), x(4), x(5) sechs Proben des Signals sind, dann werden die ersten drei Tiefpaßkoeffizienten $y_0(0)$, $y_0(1)$, $y_0(2)$ und der erste Hochpaßkoeffizient $y_1(0)$ gegeben durch

$$y_0(0) = \lfloor (x(0) + x(1))/2 \rfloor$$

$$y_0(1) = \lfloor (x(2) + x(3))/2 \rfloor$$

$$y_0(2) = \lfloor (x(4) + x(5))/2 \rfloor$$

$$y_1(0) = \lfloor (- \lfloor (x(0) + x(1))/2 \rfloor + 4(x(2) - x(3)) + \lfloor (x(4) + x(5))/2 \rfloor)/4$$
$$= \lfloor (-y_0) + 4(x(2) - x(3)) + y_0(2))/4 \rfloor .$$

[0097] Da

$$x(2) - x(3) = y_1(0) - \lfloor - (y_0(0) + y_0(2))/4 \rfloor$$

ist dann x(2) – x(3) vollständig bekannt. Mit $y_0(1) = \lfloor (x(2) + x(3))/2 \rfloor$ und x(2) – x(3) und x(2) – x(3), die oben bestimmt sind, können x(2) und x(3) zurückgewonnen werden, weil die letzten signifikanten Bits von x(0) + x(1) und x(0) – x(1) identisch sind.

[0098] Insbesondere sei

$$d(0) = x(2) - x(3) = y_1(0) - \lfloor (-y_0(0) + y_0(2)/4 \rfloor$$

$$x(2) = y_0(1) + \lfloor (d(0) + 1)/2 \rfloor$$

$$x(3) = y_0(1) + \lceil (d(0) - 1)/2 \rceil$$

[0099] Eine Ausführungsform des Vorwärtsfilters für die RTS-Transformation wird in dem Anhang A als durch die Programmiersprache "C" umgesetzt gezeigt. Man nehme zur Kenntnis, daß mathematisch die Gleichung

$$\frac{1}{8}(-1-Z^{-1}+8Z^{-2}-8Z^{-3}+Z^{-4}+Z^{-5})$$

und die Gleichung:

$$\frac{1}{4}\left(\frac{1}{2}\left(-1-Z^{-1}\right)+4\left(Z^{-2}-Z^{-3}\right)+\frac{1}{2}\left(Z^{-4}+Z^{-5}\right)\right)$$

gleich sind, wenn sie mit einer unendlichen Präzisions-Arithmetik durchgeführt werden. Der Grund, daß zwei Gleichungen einen reversiblen Filter darstellen, wird klar, wenn sie physikalisch mit ganzzahliger Arithmetik umgesetzt werden. Beispielhafte Hardware-Umsetzungen des Tiefpaßfilters und des Hochpaßfilters werden in Verbindung mit den **Fig. 10** und **Fig. 11** beschrieben.

[0100] Man beachte, daß sowohl bei der S-Transformation als auch der RTS-Transformation der Tiefpaßfilter so umgesetzt ist, daß der Bereich des Eingangssignals x(n) der gleiche ist, wie das Ausgangssignal $y_0(n)$. Wenn zum Beispiel das Signal ein 8-Bit-Bild ist, beträgt der Ausgang des Tiefpaßfilters ebenfalls 8 Bits. Für ein pyramidales System ist dies eine wichtige Eigenschaft, wo der Tiefpaßfilter sukzessive eingesetzt wird, weil in Systemen nach dem Stand der Technik der Bereich des Ausgangssignals größer ist als der des Eingangssignals, wodurch sukzessive Anwendungen des Filters schwierig gemacht werden. Zusätzlich hat der Tiefpaßfilter nur zwei Abgriffe bzw. Stufen, die aus ihm einen nicht-überlappenden Filter machen. Diese Eigenschaft ist für die Umsetzung in Hardware wichtig, wie später unten beschrieben wird.

[0101] Gemäß einer Ausführungsform wird der Tiefpaßfilter und der Hochpaßfilter unter Bezugnahme auf die RTS-Transformation wie folgt definiert:

$$\begin{cases} h_0(Z) = \frac{1}{2}(1+Z^{-1}) \\ h_1(Z) = \frac{1}{4}\left(-\frac{1}{2}\left(1+Z^{-1}\right)+4\left(Z^{-2}-Z^{-3}\right)+\frac{1}{2}\left(Z^{-4}+Z^{-5}\right)\right) \end{cases}$$

[0102] Folglich können die Ergebnisse von den Tiefpaßfiltern zweimal in dem Hochpaßfilter (in dem ersten und dem dritten Ausdruck bzw. Term) verwendet werden. Folglich ist es nur erforderlich, zwei andere Additionen durchzuführen, um das Ergebnis des Hochpaßfilters zu erlangen.

[0103] Viele überlappende reversible Filter mit nicht-minimaler Länge können gemäß der vorliegenden Erfindung verwendet werden. Derartige Vorwärts- und Invers-Darstellungen von dem Transformationssystem zur Filterung mit nicht-überlappenden reversiblen Filtern minimaler Länge sind in Fig. 2B gezeigt. Zum Beispiel kann die folgende Klasse von Filtern gemäß der vorliegenden Erfindung verwendet werden. Für eine ganze Zahl L ≥ z gilt

$d(0) = x(2(\lfloor L/2 \rfloor + 1)) - x(2(\lfloor L/2 \rfloor + 1) + 1)$

und

$y_0(0) = \lfloor (x(0) + x(1))/2 \rfloor$

$y_0(1) = \lfloor (x(2) + x(3))/2 \rfloor$

$y_0(L-1) = \lfloor (x(2\lfloor (L-1)/2 \rfloor) + x(2\lfloor (L-1)/2 \rfloor + 1))/2 \rfloor$

und

$$y_1(0) = \frac{\sum_{i=0}^{\lfloor L/2 \rfloor} a_i y_0(i) + bd(0) + \sum_{j=\lfloor L/2 \rfloor + 2}^{L-1} c_i y_0(j)}{k}$$

[0104] Die Länge des Hochpaßfilters beträgt 2L. Wenn L ungerade ist, kann der Filter näher bei einem symmetrischen Filter sein. Wenn a_i, b, c_i und k ganze Zahlen bzw. Integer sind und k ≤ b it, dann ist der Filter reversibel bzw. umkehrbar. Wenn a_i, b, c_i und k Potenzen von zwei sind (oder das negative oder das Komplement einer Potenz von zwei), dann kann die Implementation des Filters vereinfacht werden. Wenn k = b ist (ungeachtet des Wertes von a_i und c_i) dann ist der Bereich des Ausgangs des Hochpaßfilters y_1 minimiert. Für jedes a_i, wenn es nur genau ein c_j mit $a_i = -c_j$ gibt, dann wird der Hochpaßfilter keine Reaktion bzw. Antwort auf einen konstanten Eingangswert bzw. Eingang haben. Wenn $a_i - c_j$ wenn j – (L – 1) = i ist, dann kann der Filter dichter bei einem symmetrischen Filter sein.

[0105] Eine andere zweckmäßige Eigenschaft ist

$$\sum_{i=0}^{\lfloor L/2 \rfloor}\left[(a_i)(2i)^m + (a_i)(2i+1)^m\right] + (b)\left(2\left(\lfloor 1/2 \rfloor + 1\right)\right)^m$$
$$-(b)\left(2\left(\lfloor 1/2 \rfloor + 1\right)+1\right)^m + \sum_{j=\lfloor L/2 \rfloor + 2}^{L-1}\left[c_j\left(2_j\right)^m + c_j\left(2_j+1\right)^m\right] = 0$$

[0106] Dies bewirkt, daß der Hochpaßfilter keine Reaktion bzw. Antwort auf einen sich linear ändernden Eingang hat, wenn m = 1 und sich der Eingang quadratisch ändert, wenn m = 2 usw., wobei m die momentane Bedingung bzw. der momentane Zustand ist. Diese Eigenschaft ist der prinzipielle Grund, weshalb die RTS-Transformation eine bessere Energiekompaktierung hat als die S-Transformation.

[0107] Während die Filter die minimalen Zwangsbedingungen bzw. Nebenbedingungen für die Umkehrbarkeit

für verschiedene Anwendungen erfüllen müssen, können Filter verwendet werden, die keine, einige oder sämtliche der anderen Eigenschaften erfüllen. Bei einigen Ausführungsformen wird einer der folgenden beispielhaften Hochpaßfilter verwendet. Die Filter sind in einer Notation aufgeführt, die nur die ganzzahligen Koeffizieten der rationalen Version des Filters aufführt, um eine Verundeutlichung der Erfindung zu vermeiden.

1 1 -4 -4 16 -16 4 4 -1 -1

1 1 -3 -3 8 -8 3 3 -1 -1

-1 -1 0 0 16 -16 0 0 1 1

-1 -1 4 4 -16 -16 256 -256 16 16 -4 -4 1 1

3 3 -22 -22 128 -128 22 22 -3 -3

[0108] Auf den letzten Filter wird als den (zwei/zehn) TT-Filter Bezug genommen und er weist die Eigenschaft auf, daß er keine Reaktion bzw. Antwort auf eine kubisch anwachsende Funktion hat. Man bemerke, daß dieser Filter mit insgesamt sieben Additionen und Subtraktionen in die Tat umgesetzt werden kann, da 22 = 16 + 2 × 3 und 3 = 2 + 1 ist.

[0109] Die strikten Umkehrbarkeitserfordernisse für Filter können unter Bemerkung des nachfolgenden entspannt werden. Hochpaßkoeffizienten werden in der gleichen Reihenfolge codiert bzw. decodiert. Bildpunktwerte, die zuvor decodierten Hochpaßkoeffizienten entsprechen, sind genau bekannt, so daß sie in gegenwärtigen Hochpaßfilterungen verwendet werden können. Zum Beispiel kann der folgende Filter verwendet werden, wenn eine Rasterreihe verwendet wird.

$$H_1(Z)=\left\lfloor\frac{1}{4}\left(\left\lfloor-\frac{1}{2}\left(1+Z^{-1}\right)\right\rfloor+\left\lfloor\frac{1}{2}\left(8\left(Z^{-2}-Z^{-3}\right)+\left(Z^{-4}+Z^{-5}\right)\right)\right\rfloor\right)\right\rfloor$$

[0110] Die Verwendung eines einzelnen festen Hochpaßfilters ist nicht erforderlich. Adaptive Filter können verwendet werden oder multiple Filter können verwendet werden. Die Daten, die verwendet werden, um multiple Filter zu adaptieren oder auszuwählen, müssen auf Daten eingeschränkt werden, die in dem Decodierer vor einer bestimmten inversen Filterbetätigung verfügbar sind.

[0111] Eine Art, um multiple Filter zu verwenden, ist es, die Hochpaßkoeffizienten fortschreitend zu verarbeiten. Abwechselnde Hochpaßfilterbetätigungen ($y_1(0)$, $y_1(2)$, $y_1(4)$, ...) können zuerst mit einem reversiblen Filter, wie etwa dem RTS-Hochpaßfilter, prozessiert werden. Die verbleibende Prozessierung ($y_1(1)$, $y_1(3)$, $y_1(5)$, ...) kann einen nicht-reversiblen Filter von bis zu sechs Abgriffen bzw. Stufen verwenden, weil die genauen Werte der Eingänge zu diesem überlappenden Abschnitt des Filters bekannt sind. Zum Beispiel kann ein beliebiger der folgenden Filter verwendet werden.

-1 3 -3 1

-1 4 -4 1

-3 8 -8 3

1 -5 10 -10 5 -1

1 -4 8 -8 4 -1

[0112] In einigen Ausführungsformen kann der Hochpaßfilter durch eine Vorhersage-/Interpolations-Operation ersetzt werden. Ein Vorhersager/Interpolator kann die Differenz zwischen einem Paar von Ausgängen vorhersagen, indem beliebige Daten, die in dem Decodierer vor einer besonderen Vorhersage-/Interpolations-Operation verfügbar sind, verwendet werden. Die vorhergesagte Differenz wird von der gegenwärtigen Differenz der Eingänge abgezogen und wird ausgegeben. Gemäß einer Ausführungsform werden Vorhersageverfahren nach dem Stand der Technik, die in DPCM bei progressivem Codieren oder Raumbereich-Codieren verwendet werden, benutzt.

[0113] Unter Verwendung der Hochpaß- und Tiefpaßfilter nach der vorliegenden Erfindung wird eine Zerlegung mit mehreren Lösungen durchgeführt. Die Anzahl der Pegel bzw. Niveaus der Zusammensetzung ist va-

riabel und kann irgendeine Anzahl sein; jedoch entspricht gegenwärtig die Anzahl der Zerlegungspegel zwei bis fünf Pegeln.

[0114] Zum Beispiel wird der erste Pegel bzw. das erste Niveau der Zerlegung auf die feinsten Einzelheiten oder Auflösungen angesetzt, wenn die reversible Kleinwellen-Transformation rekursiv bzw. wiederholbar auf ein Bild angesetzt wird. Bei einem ersten Zerlegungsniveau wird das Bild in zwei Subbilder (z.B. Subbänder) zerlegt. Jedes Subband stellt ein Band räumlicher Frequenzen dar. Die ersten Niveau-Subbänder sind LL_0, LH_0, HL_0 und HH_0 benannt. Das Verfahren zur Zerlegung des Originalbildes bezieht Unterabtastung um bzw. durch zwei sowohl in horizontalen als auch in vertikalen Dimensionen ein, so daß die ersten Pegel bzw. Niveau-Unterbänder LL_0, LH_0, HL_0 und HH_0 jeweils ein Viertel so viele Koeffizienten haben, wie der Eingang Bildpunkte (oder Koeffizienten) des Bildes hat, so wie in Fig. 3A gezeigt wird.

[0115] Das Unterband LL_0 enthält gleichzeitig horizontale Niederfrequenz- und vertikale Niederfrequenzinformationen. Üblicherweise wird in diesem Unterband ein großer Abschnitt der Bildenergie konzentriert. Das Unterband LH_0 enthält horizontale Niederfrequenz- und vertikale Hochfrequenzinformationen (beispielsweise horizontale Kanteninformationen). Das Unterband HL_0 enthält horizontale Hochfrequenzinformationen und vertikale Niederfrequenzinformationen (beispielsweise Informationen von vertikalen Kanten). Das Unterband HH_0 enthält horizontale Hochfrequenzinformationen und vertikale Hochfrequenzinformationen (z.B. Informationen über Texturen oder diagonale Kanten).

[0116] Jede der nachfolgenden zweiten, dritten und vierten niedrigeren Zerlegungspegel bzw. -niveaus wird durch Zerlegen der niederfrequenten LL-Unterbänder des vorherigen Pegels bzw. Niveaus erzeugt. Dieses Unterband LL_0 des ersten Pegels bzw. Niveaus wird zerlegt, um die Unterbänder LL_1, LH_1, HL_1 und HH_1 des gemäßigten zweiten Detailpegels bzw. -niveaus zu erzeugen. Ähnlich wird das Unterband LL_1 zerlegt, um Grobdetailunterbänder LL_2, LH_2, HL_2 und HH_2 des zweiten Niveaus bzw. Pegels zu erzeugen. Auch wird das Unterband LL_2 zerlegt, um die gröberen Detailunterbände LL_3, LH_3, HL_3 und HH_3 des dritten Niveaus, wie in Fig. 3D gezeigt, zu erzeugen. Aufgrund des Unterabtastens um zwei ist jedes zweite Niveau- bzw. Pegelunterband ein Sechzehntel der Größe des Originalbildes. Jede Probe bzw. Abtastung (z.B. "pel") stellt bei diesem Pegel bzw. Niveau eine mittlere Einzelheit bzw. ein mittleres oder gemäßigtes Detail in dem Originalbild an dem gleichen Ort dar. Ähnlich bzw. gleichermaßen ist jedes dritte Unterbandniveau bzw. Unterbandpegel 1/64 der Größe des Originalbildes. Jedes "pel" entspricht bei diesem Niveau bzw. Pegel einem relativ großen Detail in dem Originalbild an dem gleichen Ort. Auch ist jeder vierte Unterbandpegel bzw. jedes vierte Unterbandniveau bzw. -pegel 1/256 der Größe des Originalbildes.

[0117] Da die zerlegten Bilder wegen der Unterabtastung physikalisch kleiner sind als das Originalbild, kann der gleiche Speicher, der zur Speicherung der Originaldaten verwendet wird, verwendet werden, um sämtliche der zerlegten Unterbänder zu speichern. Mit anderen Worten, das Originalbild und die zerlegten Unterbänder LL_0 und LL_1 werden ausrangiert und werden nicht in einer Zerlegung in drei Niveaus bzw. Pegel gespeichert.

[0118] Ein Eltern-Kind- bzw. Vorgänger-Nachkomme-Verhältnis existiert zwischen einer Unterbandkomponente, die ein Grobdetail darstellt, im Verhältnis zu einer entsprechenden Unterbandkomponente bei dem nächstfeineren Detailniveau bzw. Detailpegel.

[0119] Auch wenn nur vier Unterband-Zerlegungspegel gezeigt werden, können in Entsprechung mit den Erfordernissen eines bestimmten Systems zusätzliche Pegel entwickelt werden. Auch mit anderen Transformationen, wie etwa DCT oder linear beabstandeten Unterbändern, können unterschiedliche Eltern-Kind-Verhältnisse bestimmt werden.

[0120] Das Verfahren einer Zerlegung mit mehreren Auflösungen kann unter Verwendung eines Filtersystems durchgeführt werden, wie es etwa in Fig. 4A angedeutet ist. Ein Eingabesignal, das ein eindimensionales Signal mit der Länge L darstellt, wird durch Filtereinheiten **401** und **402** tiefpaß- und hochpaßgefiltert, bevor sie durch zwei Durchgangseinheiten **403** und **405** unterabgetastet werden. Ein unterabgetastetes Ausgangssignal von der Einheit **403** wird durch die Einheiten **405** und **406** tiefpaß- und hochpaßgefiltert, bevor sie durch die Durchgangseinheiten **407** bzw. **408** unterabgetastet werden. Die Unterbandkomponenten L und H erscheinen an jeweiligen Ausgängen von Einheiten **407** und **408**. Gleichermaßen werden die Ausgangssignale von der Einheit **405** durch die Einheiten **409** und **410** tiefpaß- und hochpaßgefiltert, bevor sie durch die Einheiten **411** bzw. **412** unterabgetastet werden. Die Unterbandkomponenten L und H erscheinen an jeweiligen Ausgängen von Einheiten **411** und **412**. Wie oben beschrieben, sind in einer Ausführungsform nach der vorliegenden Erfindung, die bei Unterband-Zerlegung verwendet wird, digitale Quadraturspiegelfilter bzw. Quertransformationsspiegelfilter, um die horizontalen und vertikalen Frequenzbänder in Niederfrequenz- und Hochfrequenz-

bänder aufzuteilen.

[0121] Die Fig. 4B stellt eine zweidimensionale zweipegelige Transformation dar. Die Fig. 4C stellt auch eine zweidimensionale, zweipegelige Transformation dar, die unter Verwendung von eindimensionalen Filtern in die Tat umgesetzt ist, wie etwa den in den Fig. 10 und Fig. 11 gezeigten. Die eindimensionalen Filter werden an jeder anderen Stelle eingesetzt, um Berechnungen zu vermeiden, die durch Unterabtastung unnötig geworden sind. In einer Ausführungsform unterteilen eindimensionale Filter die Berechnung zwischen Tiefpaß- und Hochpaßberechnung.

[0122] Deshalb stellt die vorliegende Erfindung ein System zur Kompression und Dekompression zur Verfügung, indem überlappende reversible Filter mit nicht-minimaler Länge verwendet werden. Die Fig. 4D ist ein Blockdiagramm einer Ausführungsform eines solchen Systems. Bezugnehmend auf Fig. 4D wird eingangs eine hierarchische Dekompression durchgeführt. Die Ergebnisse der hierarchischen Dekompression werden zu einem Komprimierer zur Kompression geleitet. Die durchgeführte Kompression kann eine Vektorquantisierung, eine Skalarquantisierung, eine Nullauf-Längenzählung, Huffman-Codierung usw. enthalten. Der Ausgang des Komprimierers komprimiert Daten, die eine komprimierte Version der Originaleingangsdaten darstellt. Ein Dekomprimierer kann die Daten irgendwann in der Zukunft empfangen und die Daten dekomprimieren. Die vorliegende Erfindung führt dann eine umgekehrte Zerlegung unter Verwendung überlappender reversibler Filter mit nicht-minimaler Länge durch, um eine wiederhergestellte Version der Originaldaten zu erzeugen.

[0123] Die reversiblen Kleinwellen- bzw. Unterwellenfilter nach der vorliegenden Erfindung können auch bei exemplarischen Analyse- und Verstärkungssystemen verwendet werden, wie etwa in Fig. 4E gezeigt. Bezugnehmend auf Fig. 4E wird hierarchische Zerlegung an Eingangsdaten unter Verwendung überlappender reversibler Kleinwellenfilter mit nicht-minimaler Länge durchgeführt. Die Analyseeinheit empfängt die durch die Filter erzeugten Koeffizienten und klassifiziert sie zu Entscheidungen, beispielsweise werden eher als die Koeffizienten vollständig zu codieren, nur relevante Informationen extrahiert. Zum Beispiel können in einem Dokumentenarchivierungssystem Leerseiten nur unter Verwendung des gröbsten Tiefpaß-Unterbandes erkannt werden. Ein anderes Beispiel wäre es, nur Hochpaßinformationen von einem bestimmten Unterband zu verwenden, um zwischen Bild und Text und Bildern von natürlichen Landschaften, Motiven oder dergleichen zu unterscheiden. Die hierarchische Zerlegung kann zur Registrierung verschiedenster Bilder verwendet werden, so daß zuerst eine Grobregistrierung mit Grobunterbändern vorgenommen wird. Bei anderen Ausführungsformen unterliegen die Koeffizienten, gefolgt durch eine inverse Zerlegung, einer Verstärkung oder Filterung. Verschärfung, Kantenverstärkung, Rausch- bzw. Störsteuerung usw. können unter Verwendung einer hierarchischen Zerlegung durchgeführt werden. Folglich stellt die vorliegende Erfindung eine Kleinwellen-Transformation zur Verwendung in verbundenen Zeit-/Raum- und Frequenzbereichsanalyse- und Filterungs-Verstärkungssystemen dar.

Die Bit-Wichtungs- bzw. Wertigkeits-Einbettungscodierung

[0124] Gemäß der vorliegenden Erfindung werden die als ein Ergebnis der Kleinwellenzerlegung erzeugten Koeffizienten entropie-codiert. Gemäß der vorliegenden Erfindung werden die Koeffizienten anfänglich einer Einbettungscodierung unterzogen, in der die Koeffizienten in einer visuellen Wichtungsordnung geordnet werden oder, allgemeiner ausgedrückt, im Hinblick auf einige Fehlermaße bzw. Maßstäbe geordnet werden (z.B. Verzerrungsmetrik). Fehler- oder Verzerrungsmetriken enthalten Spitzenfehler und mittlere quadratische Fehler (MSE). Zusätzlich kann ein Ordnen durchgeführt werden, um der Bitsignifikanz bzw. Bitwichtungsraumanordnung, Relevanz für auf Daten basierendes Fragen und richtungsorientiert durchzuführen (vertikal, horizontal, diagonal usw.). Die vorliegende Erfindung verwendet Mehrfacheinbettungs-Codierungstechniken, in denen ein Abschnitt der Koeffizienten bei einem Wichtungspegel mit einer Codierungstechnik codiert wird, während die verbleibenden Koeffizienten mit anderen Techniken codiert werden. Gemäß der vorliegenden Erfindung sind auf Frequenzen basierendes Modellieren und verbundenes Raum-/Frequenz-Modellieren zwei verschiedene Einbettungscodierungssysteme, die verwendet werden, um die durch die Kleinwellen-Transformation nach der vorliegenden Erfindung erzeugten Koeffizienten zu codieren. Das frequenzbasierende Modellieren bezieht ein, daß eine Anzahl von Koeffizienten bei höheren Frequenzen, als wenn ein Koeffizient bei einer niedrigeren Frequenz codiert wird, vorhergesagt werden. Das verbundene Raum-/Frequenz-Modellieren nimmt einen Vorteil sowohl von den bekannten Frequenzbändern als auch den benachbarten Bildpunkten (oder Daten). Auf eine Ausführungsform der verbundenen Raum-/Frequenz-Modellierung wird hierin als Horizontal- bzw. Horizontmodellierung Bezug genommen.

[0125] Die Daten werden eingangs in Zeichengrößenordnungsformate formatiert, was von dem Sortieren der

Daten auf der Grundlage der Wichtung gefolgt ist. Nachdem die Daten im Hinblick auf die gegebene Wichtungsmetrik sortiert sind, werden die Daten codiert. Sowohl das auf Frequenzen basierende Codieren als auch das Horizontalcodieren können auf Bit-Signifikanzordnung basieren, aber verwenden unterschiedliche Verfahren zum Codieren der Ereignisse.

[0126] Ein digitales Signal x(n) sei angenommen, wobei jedes x(n) mit R-Bits für die Genauigkeit dargestellt wird, wobei dann die Einbettungscodierung nach der vorliegenden Erfindung das signifikanteste Bit (oder Bits) für jedes x(n) von dem Signal codiert, dann das nächstsignifikante Bit (oder Bits) codiert usw. Zum Beispiel könnte in dem Fall des visuell definierten Ordnens ein Bild, das in dem Zentrum eine bessere Qualität als entlang der Ecken und nahe der Kanten (wie beispielsweise einige medizinische Bilder) erfordert, einer solchen Codierung unterzogen werden, daß Bits niederer Ordnung der zentralen Bildpunkte vor den Bits höherer Ordnung der Umgebungsbildpunkte codiert werden.

[0127] Für ein eingebettetes System, das auf einer Bit-Signifikanzverzerrungsmessung basiert, werden binäre Werte der Daten nach den Größenordnungen geordnet. In dem Fall, wo die Werte nicht negative ganze Zahlen (integer) sind, so wie sie im Hinblick auf die Intensität von Bildpunkten auftreten, ist die Ordnung, die verwendet werden kann, die Bitebenen-Ordnung (z.B. von der signifikantesten zu der am wenigsten signifikanten Bitebene). In den Ausführungsformen, in denen komplementäre negative ganze Zahlen bzw. Integers von zweien ebenfalls erlaubt sind, ist die eingebettete Ordnung des Zeichenbits die gleiche wie das erste Bit, das nicht Null ist, des absoluten Wertes der ganzen Zahl. Deshalb wird das Zeichenbit nicht beachtet, bis ein Bit codiert wird, das nicht Null ist. Als ein Ergebnis werden die möglichen Werte für ein Ereignis in dem Bit-Signifikanz-Einbettungssystem nach der vorliegenden Erfindung ternär, bevor das Zeichenbit codiert wird. Die ternären Ereignisse sind "nicht signifikant", "positiv signifikant" und "negativ signifikant". Zum Beispiel lautet die 16-Bit-Zahl –7 unter Verwendung der Zeichengrößenordnungsnotation:
1000000000000111

[0128] Auf einer Bit-Ebenen-Grundlage werden die ersten zwölf Entscheidungen "nicht signifikant" lauten. Das erste 1-Bit tritt bei der dreizehnten Entscheidung auf. Die dreizehnte Entscheidung wird "negativ signifikant" lauten. Nachdem das Zeichenbit codiert ist, werden die möglichen Ereignisse zu binären verringert, d.h. 0, 1. Die vierzehnte und die fünfzehnte Entscheidung lauten beide "1".

[0129] Bei einer Ausführungsform nach der vorliegenden Erfindung wird eine Liste verwendet, um sich über die Koeffizienten auf dem laufenden zu halten. In einer Ausführungsform unterscheidet ein Bit-Flag, auf das hierin als Gruppenflag Bezug genommen wird, das mit jedem Koeffizienten verbunden ist, Koeffizienten voneinander, deren Zeichenbit bis jetzt noch nicht mit den Koeffizienten mit den bereits codierten Zeichenbits codiert worden ist. In einer anderen Ausführungsform können zwei oder mehr Listen anstelle eines Flag-Bits verwendet werden. In einer anderen Ausführungsform wird eine einzige Liste ohne ein Flag verwendet.

[0130] In einer anderen Ausführungsform werden keine Listen verwendet. Sämtliche Entscheidungen für einen Koeffizienten werden erzeugt und durch die Signifikanz bzw. Wertigkeit benannt bzw. beschriftet, bevor irgendwelche Entscheidungen für den nächsten Koeffizienten erzeugt werden. Dies eliminiert das Erfordernis zum Speichern sämtlicher Koeffizienten in der Liste.

Das Codierungs- und Decodierungsverfahren nach der vorliegenden Erfindung

[0131] Die folgenden Flußdiagramm, die **Fig.** 25 bis 30, stellen Ausführungsformen des Codierungs- und Decodierungsverfahrens nach der vorliegenden Erfindung dar.

[0132] Die **Fig.** 25 zeigt ein Flußdiagramm, das eine Codierungstransformation und ein Modellierverfahren nach der vorliegenden Erfindung darstellt. Bezugnehmend auf **Fig.** 25 beginnt die Codierungstransformation und das Modellierungsverfahren durch den Erhalt von Eingangsdaten (Verfahrensblock **2501**). Nachdem Eingangsdaten erhalten worden sind, wendet die vorliegende Erfindung einen reversiblen Kleinwellenfilter an (Verfahrensblock **2502**).

[0133] Als nächstes bestimmt ein Test, wenn ein anderer Pegel der Zerlegung erwünscht ist (Verfahrensblock **2503**). Wenn ein anderer Zerlegungspegel gewünscht wird, wird das Verfahren beim Verfahrensschritt **2504** fortgesetzt, wo der reversible Filter auf die LL-Koeffizienten angewendet wird, die aus der unmittelbar davor befindlichen Zerlegung resultieren und das Verfahren wird hinten beim Verfahrensschritt **2503** fortgesetzt. Auf diese Weise ermöglicht die vorliegende Erfindung, jede Anzahl von Zerlegungspegeln bzw. -niveaus durchzuführen.

[0134] Wenn ein anderer Zerlegungspegel gewünscht wird, wird das Verfahren bei dem Verfahrensblock **2506** fortgesetzt, wo das Gruppenflag von jedem Koeffizienten für die A-Gruppe vorbereitet wird. Nach der Initialisierung bzw. Vorbereitung des Gruppenflags wird die Bitebene für den A-Durchgang, S_A, auf die signifikanteste Bitebene (max) eingestellt bzw. gesetzt (Verfahrensblock **2507**). Als nächstes wird die Bitebene für den B-Durchgang, S_B, auf die nächstsignifikanteste Bitebene (max-1) eingestellt bzw. gesetzt (Verfahrens- bzw. Verarbeitungsblock **2508**).

[0135] Anschließend bestimmt ein Test, ob die Bitebene für den A-Durchgang S_A mittels einem frequenzbasierenden Modell codiert werden muß (Verfahrensblock **2509**). Wenn die Bitebene S_A mit dem frequenzbasierenden Modell zu codieren ist, wird das Verfahren bei dem Verfahrensblock **2510** fortgesetzt, wo jeder Koeffizient mit dem frequenzbasierenden Modell und dem Entropiecode modelliert wird. Andererseits wird das Verfahren bei dem Verfahrensblock **2511** fortgesetzt, wenn jeder Koeffizient mit einem verbundenen Raum-/Frequenz-Modell und einem Entropiecode modelliert wird, wenn die Bitebene S_A nicht mit dem frequenzbasierenden Modell zu codieren ist.

[0136] In jedem Fall wird das Verfahren danach bei dem Verfahrensblock **2512** fortgesetzt, wo ein Test bestimmt, ob die Bitebene S_A größer ist als oder gleich Null ist, wodurch angezeigt wird, ob es die letzte Bitebene ist. Wenn die Bitebene S_A größer als oder gleich Null ist, zieht das Verfahren eine Schleife zurück zu dem Verfahrensblock **2509**. Andererseits, wenn die Bitebene S_A größer als oder gleich Null ist, wird das Verfahren bei dem Verfahrensblock **2513** fortgesetzt, wo ein Test bestimmt, ob die Bitebene S_B größer als oder gleich Null ist, so daß das Verfahren bestimmt, ob die Bitebene die letzte Bitebene ist, die einen B-Durchgang zu durchlaufen hat. Wenn die Bitebene S_B größer als oder gleich Null ist, wird das Verfahren bei dem Verfahrensblock **2509** fortgesetzt. Jedoch wird das Verfahren beim Verfahrensblock **2514** fortgesetzt, wo die codierten Daten entweder auf einen Kanal übertragen oder in einem Speicher gespeichert werden, wenn die Bitebene S_B nicht größer als oder gleich Null ist. Nach dem Speichern oder Übertragen der codierten Daten endet das Codiertransformations- und Modellierungsverfahren nach der vorliegenden Erfindung.

[0137] Die **Fig.** 26 stellt ein Decodiertransformations- und Modellierungsverfahren nach der vorliegenden Erfindung dar. Bezugnehmend auf **Fig.** 26 beginnt das Decodiertransformations- und Modellierungsverfahren nach der vorliegenden Erfindung mit dem Wiederauffinden codierter Daten (Verfahrensblock **2601**). Die codierten Daten können von einem Kanal oder Speicher oder einem anderen Übertragungssystem empfangen werden. Nach dem Wiederauffinden der codierten Daten wird ein Gruppenflag für jeden Koeffizienten zu der A-Gruppe initialisiert (Verfahrensblock **2602**). Nachfolgend zu dieser Initialisierung wird die Bitebene für den A-Durchgang S_A auf die signifikanteste Bitebene (max) (Verfahrensblock **2603**) gesetzt und die Bitebene für den B-Durchgang S_B wird auf die nächstsignifikante Bitebene (max-1) (Verfahrensblock **2604**) gesetzt. Dann wird der Wert jedes Koeffizienten auf einen Anfangswert von Null eingestellt (Verfahrensblock **2605**).

[0138] Nach dem Initialisieren des Wertes von jedem Koeffizienten auf Null, bestimmt ein Test, ob die Bitebene S_A mit einem frequenzbasierenden Modell zu decodieren ist oder nicht (Verfahren **2606**). Wenn die Bitebene S_A mit einem frequenzbasierenden Modell zu decodieren ist, setzt das Verfahren beim Verfahrensblock **2607** fort, wo jeder Koeffizient mit einem frequenzbasierenden Modell und einer Entropiedecodierung modelliert wird. Wenn die Bitebene S_A nicht mit einem frequenzbasierenden Modell decodiert wird, wird das Verfahren beim Verfahrensschritt **2608** fortgesetzt, wo jeder Koeffizient mit einem verbundenen Raum-/Frequenz-Modell und Entropiedecodierung modelliert wird.

[0139] Nachdem jeder Koeffizient modelliert ist, setzt das Verfahren beim Verfahrensblock **2609** fort, wo die Bitebene S_A bestimmt, ob es die letzte Bitebene ist, indem geprüft wird, ob sie größer als oder gleich Null ist. Wenn die Bitebene S_A größer als oder gleich Null ist, wird das Verfahren beim Verfahrensblock **2606** fortgesetzt. Andererseits bestimmt ein Test, ob die B-Durchgangs-Bitebene S_B größer als oder gleich Null ist, wenn die Bitebene S_A nicht größer als oder gleich Null ist (Verfahrensblock **2610**), wodurch angezeigt wird, daß es sich für einen B-Durchgang um die letzte Bitebene handelt. Falls dem so ist, wird das Verfahren bei dem Verfahrensblock **2606** für die weitere Decodierung fortgesetzt. Andererseits wird, falls die Bitebene für den B-Durchgang S_B nicht größer als oder gleich Null ist, ein inverser reversibler Filter auf die Koeffizienten von dem gröbsten Pegel der Zerlegung angewendet (Verfahrensblock **2611**). Ein Test bestimmt dann, ob sämtliche der Pegel invers gefiltert worden sind (Verfahrensblock **2612**). Falls nicht, wird abermals der inverse reversible Filter auf die Koeffizienten auf dem gröbsten verbleibenden Pegel bzw. Niveau der Zusammensetzung angewendet (Verfahrensblock **2613**). Danach wird das Verfahren hinten beim Verfahrensschritt **2612** fortgesetzt, um noch einmal zu prüfen, ob sämtliche der Pegel invers gefiltert worden sind.

[0140] Wenn sämtliche der Pegel invers gefiltert worden sind, wird das Verfahren beim Verfahrensblock **2612**

fortgesetzt, wo eine Speicherung oder eine Übertragung von wiederhergestellten Daten auftritt.

[0141] Die **Fig.** 27 stellt eine Ausführungsform des Verfahrens zum Modellieren jedes Koeffizienten dar. Das wiedergegebene Verfahren stellt das Modellierungsverfahren für entweder das frequenzbasierende bzw. JSF-Modellieren oder Codieren bzw. Decodieren dar. Das heißt, jeder der vier Blöcke (**2507**, **2508**, **2607**, **2608**) kann mit dem Modellierungsverfahren nach **Fig.** 27 in die Tat umgesetzt werden. Bezugnehmend auf **Fig.** 27 beginnt ein anfängliches Verfahren durch anfängliches Prüfen, ob die Modellierung in einem Durchgang durchgeführt wird (Verfahrensblock **2701**). Falls die Modellierung nicht in einem Durchgang geschehen kann, bestimmt ein Test, ob die Bitebene S_A größer ist als die Bitebene S_B (Verfahrensblock **2702**). Wenn dies nicht der Fall ist, dann geht das Verfahren zu dem Verfahrensblock **2703** über, wo ein Flag (do_A_flag) freigemacht wird, um anzuzeigen, daß ein A-Durchgang nicht durchgeführt wird. Wenn die Bitebene S_A größer als die Bitebene S_B ist, dann wird das Verfahren bei dem Verfahrensblock **2704** fortgesetzt, wo das Flag (do_A_flag) gesetzt wird, um anzuzeigen, daß ein A-Durchgang durchzuführen ist.

[0142] Nach einem der Verfahrensblöcke **2703** oder **2704** wird das Verfahren bei dem Verfahrensblock **2705** fortgesetzt, wo ein Test bestimmt, ob die Bitebene S_B gleich der Bitebene S_A ist. Wenn die Bitebenen gleich sind, macht die vorliegende Erfindung ein Flag (do_B_flag) frei, um das Auftreten eines B-Durchlaufs zu verhindern (Verfahrensblock **2705**) und das Verfahren wird danach bei dem Verfahrensblock **2707** fortgesetzt. Wenn die Bitebene S_B gleich der Bitebene S_A ist, ist das "do_B_flag"-Flag gesetzt, um anzuzeigen, daß ein B-Durchgang durchzuführen ist (Verfahrensblock **2706**) und das Verfahren wird danach auch bei dem Verfahrensblock **2707** fortgesetzt.

[0143] Bei dem Verfahrensblock **2707** bestimmt ein Test, ob das Flag des A-Durchgangs gesetzt ist und die Nullbaum-Modellierung durchzuführen ist. Falls das Flag anzeigt, daß ein A-Durchgang aufzutreten hat und eine Nullbaum-Modellierung durchzuführen ist, wird ein "bestimmt/unbestimmt"-Flag auf den Zustand "unbestimmt" für jeden Koeffizienten initialisiert (Verfahrensblock **2708**), und das Verfahren wird bei dem Verfahrensblock **2709** fortgesetzt. Wenn andererseits entweder das A-Durchgangsanzeigeflag oder die Nullbaum-Modellierungsanzeige nicht gesetzt sind, wird unmittelbar mit dem Verfahrensblock **2709** fortgesetzt. Bei dem Verfahrensblock **2709** ist der erste Koeffizient auf die Variable C gesetzt.

[0144] Wenn der erste Koeffizient der Variablen C zugeordnet worden ist, bestimmt ein Test ob der B-Durchgangsanzeigeflag gesetzt ist (Verfahrensblock **2719**). Wenn der B-Durchgangsanzeigeflag (do_B_flag) gesetzt ist, führt die vorliegende Erfindung einen B-Durchgang mit dem Koeffizienten C (Verfahrensblock **2710**) durch und das Verfahren wird bei dem Verfahrensblock **2711** fortgesetzt. Andererseits wird, falls der B-Durchgangsflag nicht gesetzt ist, dann ein B-Durchgang nicht für C durchgeführt und das Verfahren wird unmittelbar beim Verfahrensblock **2711** fortgesetzt.

[0145] Ein Test bestimmt dann, ob der A-Durchgangsanzeigeflag (do_A_flag) gesetzt ist (Verfahrensblock **2711**). Wenn der A-Durchgangsanzeigeflag gesetzt ist, dann wird mit dem Koeffizienten C ein A-Durchgang durchgeführt (Verfahrensblock **2717**). Danach wird das Verfahren beim Verfahrensblock **2713** fortgesetzt. Falls der A-Durchgangsanzeigeflag nicht gesetzt ist, wird das Verfahren beim Verfahrensblock **2713** fortgesetzt, ohne einen A-Durchgang mit Koeffizient C durchzuführen.

[0146] Beim Verfahrensblock **2713** bestimmt ein Test, ob der Koeffizient C der letzte Koeffizient ist. Wenn der Koeffizient C nicht der letzte Koeffizient ist, dann wird das Verfahren beim Verfahrensblock **2714** fortgesetzt, wo der nächste Koeffizient der Variablen C zugeordnet wird und das Verfahren beim Verfahrensblock **2719** fortgesetzt wird. Jedoch wird das Verfahren beim Verfahrensblock **2715** fortgesetzt, wo ein Test bestimmt, wenn der B-Durchgangsflag (do_B_flag) gesetzt ist, falls der Koeffizient C der letzte Koeffizient ist. Wenn der B-Durchgangsflag gesetzt ist, wird die Bitebene S_B gleich der Bitebene S_B-1 gesetzt (Verfahrensblock **2716**) und das Verfahren wird bei dem Verfahrensblock **2717** fortgesetzt. Falls der B-Durchgangsanzeigeflag nicht gesetzt ist, wird das Verfahren beim Verfahrensblock **2717** fortgesetzt. Bei dem Verfahrensblock **2717** bestimmt ein Test, ob der A-Durchgangsflag gesetzt ist. Falls er gesetzt ist, dann ist die Bitebene S_A gleich zu der Bitebene S_A-1 gesetzt (Verfahrensblock **2718**) und das Verfahren endet. Auch falls der A-Durchgangsflag nicht gesetzt ist, endet das Verfahren unmittelbar.

[0147] Bei einigen Ausführungsformen kann ohne die Verwendung eines Flag-Bits bestimmt werden, ob ein Koeffizient bei einer bestimmten Bitebene in der A-Gruppe oder der B-Gruppe ist. Dies spart ein Speicherbit pro Koeffizient, was für große Bilder wesentlich sein kann. Stattdessen wird eine Maske unter Verwendung von UND-Logik für einen Koeffizienten verglichen. Wenn das Ergebnis des UND Null ist, ist das Bit in der A-Gruppe; ansonsten ist es in der B-Gruppe. Ein Beispiel für diese Masken wird in der Tabelle 7 für acht Bitebenen ge-

zeigt. Man beachte, daß diese Masken die Zweier-Komplemente von $2^{(\text{Bitebene}+1)}$ sind (ohne das Zeichenbit).

Tabelle 7 – Masken

Bitebene	Maske (binär)
7	00000000
6	10000000
5	11000000
4	11100000
3	11110000
2	11111000
1	11111100
0	11111110

[0148] Da den A-Durchgang bzw. B-Durchgang unabhängige Masken zugeordnet werden können (hierin M_A und M_B genannt), können so viele A-Durchgänge wie gewünscht, vor dem entsprechenden B-Durchgang durchgeführt werden. Gemäß einer Ausführungsform mit 17 Bitebenen werden drei A-Durchgänge durchgeführt, dann werden 14 gleichzeitige A-Durchgänge und B-Durchgänge durchgeführt und schließlich werden zwei B-Durchgänge durchgeführt. Da typischerweise A-Durchgangsentscheidungen effizienter codiert werden können als B-Durchgangsentscheidungen, können anfangs mehrfach durchgeführt A-Durchgänge die Qualität für eine verlustbehaftete Kompression verbessern.

[0149] Die **Fig.** 28 stellt eine Ausführungsform eines Codierers nach der vorliegenden Erfindung dar, der einen verkleinerten bzw. verringerten Flagspeicher verwendet (wie später in der detaillierten Beschreibung erörtert wird). Bezugnehmend auf **Fig.** 28 beginnt das Codier-Transformations- und Modelierungsverfahren mit dem Erhalt der Eingangsdaten (Verfahrensblock **2801**). nach dem Erhalten der Eingangsdaten legt die vorliegende Erfindung einen umkehrbaren Kleinwellenfilter (Verfahrensblock **2802**) an.

[0150] Als nächstes bestimmt ein Test, ob ein anderer Pegel der Zerlegung gewünscht ist (Verfahrensblock **2803**). Falls ein anderer Zerlegungspegel gewünscht ist, wird das Verfahren beim Verfahrensblock **2804** fortgesetzt, wo der reversible bzw. umkehrbare Filter an die LL-Koeffizienten angelegt wird, die aus der unmittelbar vorherigen Zerlegung resultieren, und das Verfahren wird hinten beim Verfahrensschritt **2803** fortgesetzt. In dieser Weise ermöglicht es die vorliegende Erfindung, jede Anzahl von Zerlegungspegeln durchzuführen.

[0151] Wenn ein anderer Zerlegungspegel nicht gewünscht ist, wird das Verfahren beim Verfahrensblock **2805** fortgesetzt, wo die Bitebene für den A-Durchgang S_A auf die signifikanteste Bitebene (max) gesetzt wird. Als nächstes wird die Bitebene für den B-Durchgang S_B auf die nächstsignifikanteste Bitebene (max-1) gesetzt (Verfahrensblock **2806**).

[0152] Als nächstes wird die Maske M_A auf $-2^{(S_A+1)}$ (Verfahrensblock **2807**) gesetzt und die Maske M_B wird auf $-2^{(S_B+1)}$ gesetzt (Verfahrensblock **2808**). Dann bestimmt ein Test, ob die Bitebene für den A-Durchgang S_A mit einem frequenzbasierenden Modell (Verfahrensblock **2808**) codiert werden soll. Falls die Bitebene S_A mit dem frequenzbasierenden Modell zu codieren ist, wird das Verfahren beim Verfahrensblock **2809** fortgesetzt, wo ein Bit von jedem Koeffizienten mit dem frequenzbasierenden Modell und dem Entropiecode modelliert wird. Andererseits wird das Verfahren beim Verfahrensblock **2810** fortgesetzt, wo ein Bit von jedem Koeffizienten mit einem verbundenen Raum-/Frequenz-Modell und Entropiecode modelliert wird, falls die Bitebene S_A nicht mit dem frequenzbasierenden Modell zu codieren ist.

[0153] In jedem Fall wird das Verfahren danach beim Verfahrensblock **2811** fortgesetzt, wo ein Test bestimmt, ob die Bitebene S_A größer als oder gleich Null ist, wodurch angezeigt wird, ob es die letzte Bitebene ist. Falls die Bitebene S_A größer als oder gleich Null ist, macht das Verfahren eine Schleife zurück zu dem Verfahrensblock **2808**. Andererseits wird das Verfahren beim Verfahrensblock **2812** fortgesetzt, wo ein Test bestimmt, ob die Bitebene S_B größer als oder gleich Null ist, falls die Bitebene S_A nicht größer als oder gleich Null ist, so daß das Verfahren bestimmt, ob die Bitebene die letzte Bitebene ist, die einen B-Durchgang durchläuft. Falls die Bitebene S_B größer als oder gleich Null ist, setzt das Verfahren beim Verfahrensblock **2808** fort. Falls jedoch die Bitebene S_B nicht größer als oder gleich Null ist, wird das Verfahren beim Verfahrensblock **2813** fortgesetzt, wo codierte Daten entweder auf einen Kanal übertragen oder in einem Speicher gespeichert werden. Nach dem Speichern oder Übertragen der codierten Daten endet das Codiertransformations- und Modellierungsverfahren nach der vorliegenden Erfindung.

[0154] Die **Fig.** 29 stellt eine alternative Ausführungsform des Decodertransformations- und Modellierungsverfahrens nach der vorliegenden Erfindung dar, wenn ein reduzierter Flagspeicher verwendet wird. Bezugnehmend auf **Fig.** 29 beginnt das Decodiertransformations- und Modellierungsverfahren nach der vorliegenden Erfindung durch Wiederherstellen bzw. Wiederauffinden codierter Daten (Verfahrensblock **2901**). Die codierten Daten können von einem Kanal oder Speicher oder anderen Übertragungssystemen empfangen wreden. Sobald die codierten Daten empfangen werden, wird die Bitebene für den A-Durchgang S_A auf die signifikanteste Bitebene (max) gesetzt (Verfahrensblock **2903**) und die Bitebene für den B-Durchgang S_B wird auf die nächstsignifikanteste Bitebene (max-1) gesetzt (Verfahrensblock **2904**). Nach dem Initialisieren des Wertes von jedem Koeffizienten auf Null, wird der Wert von jedem Koeffizienten auf einen anfänglichen Wert von Null gesetzt (Verfahrensblock **2905**). Dann wird die Maske M_B auf $-2^{(S_B+1)}$ gesetzt (Verfahrensblock **2902**) und die Maske M_A wird auf $-2^{(S_A+1)}$ gesetzt (Verfahrensblock **2915**).

[0155] Dann bestimmt ein Test, ob die Bitebene S_A mit einem frequenzbasierenden Modell zu decodieren ist oder nicht (Verfahrensblock **2906**). Falls die Bitebene S_A mit einem frequenzbasierenden Modell zu decodieren ist, wird das Verfahren beim Verfahrensblock **2907** fortgesetzt, wo ein Bit von jedem Koeffizienten mit einem frequenzbasierenden Modell und einer Entropiedecodierung modelliert wird. Falls die Bitebene S_A nicht mit einem frequenzbasierenden Modell zu decodieren ist, wird das Verfahren beim Verfahrensblock **2908** fortgesetzt, wo ein Bit von jedem Koeffizienten mit einem verbundenen Raum-/Frequenz-Modell und einer Entropiedecodierung modelliert wird.

[0156] Nachdem jeder Koeffizient modelliert ist, wird das Verfahren beim Verfahrensblock **2909** fortgesetzt, wo die Bitebene S_A bestimmt, ob es die letzte Bitebene ist, indem geprüft wird, ob sie größer als oder gleich Null ist. Ist die Bitebene S_A größer als oder gleich Null, wird das Verfahren beim Verfahrensblock **2906** fortgesetzt. Andererseits, falls die Bitebene S_A nicht größer als oder gleich Null ist, dann bestimmt ein Test, ob die B-Durchgangsbitebene S_B größer als oder gleich Null ist (Verfahrensblock **2910**), wobei angezeigt wird, daß es die letzte Bitebene für einen B-Durchgang ist. Falls dem so ist, wird das Verfahren beim Verfahrensblock **2902** für die weitere Decodierung fortgesetzt. Falls andererseits die Bitebene für den B-Durchgang, S_B, nicht größer als oder gleich Null ist, wird ein reverser reversibler Filter an die Koeffizienten von dem gröbsten Zerlegungspegel angesetzt (Verfahrensblock **2911**). Ein Test bestimmt dann, ob sämtliche der Pegel umgekehrt bzw. invers gefiltert worden sind (Verfahrensblock **2912**). Falls nicht, wird der inverse reversible Filter wieder auf die Koeffizienten auf dem verbleibenden gröbsten Verbindungsniveau bzw. Zerlegungsniveau angelegt (Verfahrens- bzw. Verarbeitungsblock **2913**). Danach wird das Verfahren hinten beim Verfahrensblock **2912** fortgesetzt, wo ein Test wiederum bestimmt, ob sämtliche der Pegel bzw. Niveaus invers gefiltert worden sind.

[0157] Sobald sämtliche der Niveaus bzw. Pegel invers gefiltert worden sind, wird das Verfahren beim Verfahrensblock **2912** fortgesetzt, wo ein Speichern oder Übertragen von wiederhergestellten Daten auftritt.

[0158] **Fig.** 30 stellt eine Ausführungsform des Verfahrens zum Modellieren von jedem Koeffizienten dar. Man bemerke, daß, wie bei **Fig.** 27, das Verfahren nach **Fig.** 30 verwendet werden kann, um die Modellierungsschritte nach den **Fig.** 28 und 29 in die Tat umzusetzen. Bezugnehmend auf **Fig.** 30 beginnt ein anfängliches Verfahren durch anfängliches Prüfen, ob ein A-Durchgang gewünscht ist und ob S_A größer als oder gleich Null ist (Verfahrensblock **3001**). Falls dem so ist, dann wird das Flag (do_A_flag) gesetzt (Verfahrensblock **3004**), das anzeigt, daß ein A-Durchgang durchzuführen ist, und das Verfahren wird beim Verfahrensblock **3002** fortgesetzt. Ansonsten wird das do_A_flag-Flag freigemacht bzw. gelöscht (verfahrensblock **3003**).

[0159] Falls die Bitebene S_A größer ist als die Bitebene S_B, dann wird das Verfahren beim Verfahrensblock **3004** fortgesetzt, wo ein Flag gesetzt wird, um anzudeuten, daß ein A-Durchgang aufzutreten hat. Falls die Bitebene S_A nicht größer als die Bitebene S_B ist, dann wird das Verfahren beim Verfahrensblock **3003** fortge-

setzt, wo das Flag, das einen A-Durchgang anzeigt, als klar auftretend angenommen wird.

[0160] Nach entweder dem Verfahrensblock **3003** oder **3004** wird das Verfahren beim Verfahrensblock **3002** fortgesetzt, wo ein Test bestimmt, ob die Bitebene S_B größer als oder gleich der Bitebene S_A ist und ob ein B-Durchgang gewünscht wird. Falls die Bitebenen nicht gleich sind, löscht bzw. macht die vorliegende Erfindung ein Flag (do_B_flag) frei, um zu verhindern, daß ein B-Durchgang auftritt (Verfahrensblock **3005**) und das Verfahren wird danach beim Verfahrensblock **3007** fortgesetzt. Falls die Bitebene S_B gleich der Bitebene S_A ist, wird das do_B_flag-Flag gesetzt, um anzuzeigen, daß ein B-Durchgang durchzuführen ist (Verfahrensblock **3006**) und das Verfahren wird auch danach beim Verfahrensblock **3007** fortgesetzt.

[0161] Beim Verfahrensblock **3007** bestimmt ein Test, ob der A-Durchgangsflag gesetzt ist und die Nullbaum-Modellierung durchzuführen ist. Falls das Flag andeutet, daß ein A-Durchgang aufzutreten hat und eine Nullbaum-Modellierung durchzuführen ist, wird ein "bestimmt/unbestimmt"-Flag mit dem "unbestimmt"-Zustand für jeden Koeffizienten initialisiert, der Kinder bzw. Nachkommen (Verfahrensblock **3008**) hat und das Verfahren wird beim Verfahrensblock **3009** fortgesetzt. Andererseits wird, falls entweder der A-Durchgangsanzeigeflag oder die Nullbaum-Modellierungsanzeige nicht gesetzt sind, das Verfahren unmittelbar beim Verfahrensblock **3009** fortgesetzt. Beim Verfahrensblock **3009** wird der erste Koeffizient auf die Variable C gesetzt.

[0162] Sobald der erste Koeffizient der Variablen C zugeordnet worden ist, bestimmt ein Test, ob der B-Durchgangsanzeigeflag gesetzt ist (Verfahrensblock **3019**). Falls der B-Durchgangsanzeigeflag (do_B_flag) gesetzt ist, führt die vorliegende Erfindung einen B-Durchgang an dem Koeffizienten C (Verfahrensblock **3010**) durch und das Verfahren wird beim Verfahrensblock **3011** fortgesetzt. Andererseits wird ein B-Durchgang nicht an C durchgeführt, falls der B-Durchgangsflag nicht gesetzt ist, und das Verfahren wird unmittelbar beim Verfahrensblock **3011** fortgesetzt.

[0163] Der Test bestimmt dann, ob der A-Durchgangsanzeigeflag gesetzt worden ist (Verfahrensblock **3011**). Falls der A-Durchgangsanzeigeflag gesetzt worden ist, dann wird auf den Koeffizienten C ein A-Durchgang durchgeführt (Verfahrensblock **3017**). Danach wird das Verfahren beim Verfahrensblock **3013** fortgesetzt. Falls der A-Durchgangsanzeigeflag gesetzt ist, wird das Verfahren beim Verfahrensblock **3013** fortgesetzt, ohne einen A-Durchgang an dem Koeffizienten C durchzuführen.

[0164] Beim Verfahrensblock **3013** bestimmt ein Test, ob der Koeffizient C der letzte Koeffizient ist. Fall der Koeffizient C nicht der letzte Koeffizient ist, dann wird das Verfahren beim Verfahrensblock **3013** fortgesetzt, wo der nächste Koeffizient der Variablen C zugeordnet wird und das Verfahren wird beim Verfahrensblock **3019** fortgesetzt. Jedoch wird das Verfahren beim Verfahrensblock **3015** fortgesetzt, wo ein Test bestimmt, ob das B-Durchgangsflag (do_B_flag) gesetzt ist, falls der Koeffizient C der letzte Koeffizient ist. Wenn der B-Durchgangsflag gesetzt ist, ist die Bitebene S_B gleich der Bitebene S_B-1 (Verfahrensblock **3016**) und das Verfahren wird beim Verfahrensblock **3017** fortgesetzt. Falls der B-Durchgangsanzeigeflag nicht gesetzt ist, wird das Verfahren beim Verfahrensblock **3017** fortgesetzt. Beim Verfahrensblock **3017** bestimmt ein Test, ob der A-Durchgangsflag gesetzt ist. Falls er gesetzt ist, dann wird die Bitebene S_A gleich der Bitebene S_A-1 gesetzt (Verfahrensblock **3018**) und das Verfahren endet. Auch falls der A-Durchgangsflag nicht gesetzt wird, endet dann das Verfahren unmittelbar.

Koeffizientenbäume

[0165] In einem Pyramidalsystem können Koeffizienten unter Verwendung einer Baumstruktur in Sätze gruppiert werden. Die Wurzel von jedem Baum ist ein reiner Tiefpaßkoeffizient. Die **Fig. 5** stellt die Baumstruktur eines reinen Tiefpaßkoeffizienten des transformierten Bildes dar. Für ein zweidimensionales Signal, wie etwa ein Bild, hat die Wurzel des Baumes drei "Kinder" und der Rest der Knoten weist jeweils vier Kinder bzw. Nachkommen auf. Die Baumhierarchie ist nicht auf zweidimensionale Signale begrenzt. Zum Beispiel weist eine Wurzel für ein eindimensionales Signal einen Nachkommen und Nicht-Wurzel-Knoten auf, die jeweils zwei Nachkommen haben. Höhere Dimensionen folgen aus den eindimensionalen und zweidimensionalen Fällen.

[0166] Die Baumstruktur wird auch aus den Operationen der in den **Fig. 4A** bis **Fig. 4C** gezeigten Filter deutlich. Die Operation der Filterpaare mit Unterabtastung bewirkt die Inbezugsetzung der zuvor beschriebenen Koeffizienten.

[0167] Bei der vorliegenden Erfindung bestimmt ein Kontextmodell, welches der mehreren Codierungsverfahren zu verwenden ist, um die Koeffizienten weiter zu codieren, nachdem die Koeffizienten im Zeichen-Größenordnungsformat angeordnet worden sind. Ein frequenzbasierendes Codierungsschema, wie etwa Null-

baum-Codierung, codiert die Signifikanzdaten, die mit einer gegebenen Unterbandzerlegung für eine spezifizierte Schwelle verbunden sind, effizient. Zusätzlich zur Verwendung von Symbolen, die die Signifikanz oder Insignifikanz von einzelnen isolierten Koeffizienten in der verbundenen Unterbandzerlegung anzeigen, werden die Eingänge von insignifikanten Eltern mit allen insignifikanten Nachkommen (jene mit Größenordnungen, die geringer als oder gleich der gegebenen Schwelle sind) zusammengruppiert und gemeinsam codiert. Auf diese Bäume wird manchmal als Nullbäume Bezug genommen. Dies insignifikanten Bäume werden mit einzelnen zugeordneten bzw. zweckmäßigen Symbolen codiert, die manchmal als Nullbaumwurzel bezeichnet werden. Wenn es da jedoch einen signifikanten Herabsteigenden gibt, wird der Eingang eines insignifikanten Koeffizienten unter Verwendung des Symbols für eine "isolierte bzw. getrennte Null" codiert. Folglich wird ein Baum mit vier Symbolen (positive Signifikanz, negative Signifikanz, isolierte Null oder Nullbaumwurzel) für Entscheidungen codiert, wobei das Zeichen für den Koeffizienten bis dahin nicht codiert worden ist.

[0168] Das frequenzbasierende Codieren ist besonders zweckmäßig bei Kompressionssystemen, weil es das verbundene Codieren von insignifikanten Bäumen einer kleinen Anzahl von Eltern- bzw. Stammkoeffizienten ermöglicht, die Insignifikanz einer großen Anzahl von absteigenden bzw. abfallenden Koeffizienten vorherzusagen. Da die Eingänge in den Baum, der mit den absteigenden Koeffizienten verbunden ist, von der Wurzel vorhergesagt werden kann, sind keine zusätzlichen Symbole nötig, um ihre Insignifikanz zu codieren. Die Insignifikanz des gesamten Baumes kann bei sehr geringen Kosten codiert werden. Folglich bestehen die Bitebenen höherer Ordnung zumeist aus insignifikanten Koeffizienten, von denen viele weder Nullbaumwurzeln noch isolierte Nullen sind (d.h. sie sind Kinder bzw. Nachkommen in insignifikanten Bäumen, welche nicht codiert werden müssen).

[0169] Shapiro offenbart ein frequenzbasierendes Modell, das Nullbaum genannt wird, in dem US-Patent Nr. 5,321,776. Bei dem Verfahren von Shapiro werden zwei Listen, eine dominierende Liste und eine untergeordnete Liste, verwendet, um sämtliche der Koeffizienten zu speichern. Für jeden Signifikanz- bzw. Wertigkeitspegel bzw. jedes Signifikanzniveau werden zwei Durchgänge vorgenommen, ein dominanter Durchgang und ein untergeordneter Durchgang. Bei einer Ausführungsform ist das frequenzbasierende Modell nach der vorliegenden Erfindung ein Nullbaum.

[0170] In einer anderen Ausführungsform wird ein frequenzbasierendes Modell, ähnlich dem Nullbaum (wie durch Shapiro beschrieben), verwendet. Anstelle der Verwendung mehrerer Listen bzw. multipler Listen wird nur eine einzige Liste verwendet, wobei jedes der Listenelemente als ein Mitglied einer von zwei Gruppen markiert worden ist. Die Trennung von Koeffizienten in eine A-Gruppe und eine B-Gruppe ist äquivalent zu der Trennung, die Shapiro mit dominanten und untergeordneten Listen (jeweils) erzielt. Shapiro's Verwendung von multiplen Listen ermöglicht eine größere Flexibilität bei der Ordnung von Koeffizienten in untergeordnete Listen zu Lasten von größerer bzw. umfangreicher Software-/Hardware-Komplexität. Das Einzellisten-Nullbaum-Verfahren verwendet zwei Durchgänge, den A-Durchgang und den B-Durchgang, die äquivalent zu Shapiro's dominantem Durchgang bzw. untergeordnetem Durchgang sind. Das Einzellisten-Nullbaum-Modell wird unten beschrieben.

[0171] Das Codierungssystem nach der vorliegenden Erfindung hält eine Liste der Koeffizienten in einer Zeichen-Größenordnungsform im Speicher aufrecht. Jedes Element in der Liste hat eine Einbit-Benennung bzw. -Beschriftung, die andeutet, ob das Element ein Mitglied der "A-Gruppe" oder der "B-Gruppe" ist. Bei dem Beginn einer Stufe werden jene Koeffizienten, die nicht bis jetzt als signifikant herausgefunden worden sind, als in die A-Gruppe gehörend benannt. Jene Koeffizienten, die zuvor als signifikant im Hinblick auf vorherige, größere Schwellen herausgefunden worden sind, werden als in die B-Gruppe gehörig benannt. Die Liste enthält die Koeffizienten in der Ordnung, in der sie für die Codierung verarbeitet werden. Bei dem Beginn der allerersten Stufe werden sämtliche Koeffizienten als Mitglieder der A-Gruppe benannt, da keine Koeffizienten als signifikant eingerichtet worden sind. Wenn Koeffizienten als signifikant oder insignifikant bestimmt worden sind, werden die Benennungen für deren Eingänge von der ursprünglichen A-Gruppenbenennung zu der B-Gruppenbenennung gewechselt. Die Liste wird nachfolgend bei zunehmend feineren Schwellen verfeinert. Das heißt, mehrfache Durchgänge durch die Liste treten auf.

[0172] In einer Ausführungsform werden die binären Ereignisse, die den Koeffizienten der B-Gruppe entsprechen, binär arithmetisch unter einem Markov-Kontextmodell nullter Ordnung codiert. Die Vierer-Ereignisse (4-ary events), die den Koeffizienten der A-Gruppe entsprechen, werden ebenfalls unter einem Markov-Kontextmodell nullter Ordnung codiert.

[0173] Die Reihenfolge der Koeffizienten in der Liste gemäß der vorliegenden Erfindung bewahrt die Baumstruktur derart, daß kein Kind bzw. Nachkomme vor seinen Eltern modelliert werden kann. Folglich ist eine Ord-

nung, die die Baumstruktur bewahrt, fest eingerichtet und wird einheitlich bzw. konsistent verwendet. In einer Ausführungsform werden die Koeffizienten im Speicher in der Reihenfolge von dem ersten Speicherort verwendet. In einer anderen Ausführungsform kann eine verbundene Liste verwendet werden.

[0174] In einer Ausführungsform werden die Koeffizienten in einem Bit-Signifikanz- oder einem Bitebenen-Einbettungssystem codiert. Da die Koeffizienten von der signifikantesten Bitebene zu der am wenigsten signifikanten Bitebene codiert werden, muß die Anzahl der Bitebenen in den Daten bestimmt werden. Gemäß der vorliegenden Erfindung wird dies ausgeführt, indem eine obere Grenze bzw. Schranke der Größenordnungen der Koeffizientenwerte aus den Daten berechnet wird oder aus der Tiefe des Bildes und der Filterkoeffizienten erhalten wird. Zum Beispiel gibt es acht signifikante Bits oder acht Bitebenen, wenn die obere Schranke **149** beträgt.

[0175] Die Fig. 6A stellt eine Ausführungsform des Einzellisten-Nullbaum-Codierungsverfahrens nach der vorliegenden Erfindung dar. In einer Ausführungsform kann das Verfahren gemäß Fig. 6A auch bei dem Modellierungsverfahren nach **Fig.** 27 verwendet werden. Bezugnehmend auf Fig. 6A beginnt das Verfahren mit dem Testen, ob der Gruppenflag für den Koeffizienten C auf die "A-Gruppe" (Verfahrensschritt **3221**) gesetzt ist. Falls nicht, endet das Verfahren. Andererseits wird dann das Verfahren bei dem Verfahrensblock **3222** fortgesetzt, wo ein Test bestimmt, ob der "bestimmt/unbestimmt"-Flag für den Koeffizienten C auf "unbestimmt" gesetzt ist, falls der Gruppenflag für den Koeffizienten C auf die "A-Gruppe" gesetzt ist. Falls das "bestimmt/unbestimmt"-Flag für den Koeffizienten nicht auf "unbestimmt" gesetzt ist, endet das Verfahren. Falls jedoch der "bestimmt/unbestimmt"-Flag für den Koeffizienten C auf "unbestimmt" gesetzt ist, wird das Verfahren bei dem Verfahrensblock **3203** fortgesetzt, wo ein Test bestimmt, ob das Bit S_A des Koeffizienten C Eins ist.

[0176] Wenn das Bit S_A des Koeffizienten C nicht Eins ist, wird das Verfahren bei dem Verfahrensblock **3207** fortgesetzt. Andererseits wird das Verfahren bzw. die Verarbeitung beim Verfahrensblock **3204** fortgesetzt, wo ein Test bestimmt, ob das Zeichen des Koeffizienten C positiv ist, falls das Bit S_A des Koeffizienten C Eins ist. Wenn das Zeichen des Koeffizienten C nicht positiv ist, wird die Entscheidung bei "negativ signifikant" in dem/den "A-Gruppen"-Kontext(en) codiert (Verfahrensblock **3205**), und das Verfahren wird beim Verfahrensblock bzw. Verarbeitungsblock **3229** fortgesetzt. Wenn das Zeichen des Koeffizienten C positiv ist, wird die Entscheidung als "positiv signifikant" in dem/den "A-Gruppen"-Kontext(en) codiert (Verarbeitungsblock **3206**) und das Verfahren wird bei dem Verfahrens- bzw. Verarbeitungsblock **3229** fortgesetzt. Bei dem Verarbeitungs- bzw. Verfahrensblock **3229** wird der Gruppenflag für C auf die "B-Gruppe" gesetzt.

[0177] Bei dem Verarbeitungsblock **3207** bestimmt ein Test, ob das Bit S_A für sämtliche absteigenden (Kinder bzw. Nachkommen) des Koeffizienten C Null ist. Wenn das Bit S_A nicht Null ist, wird die Entscheidung als "insignifikant mit signifikanten Nachkommen" (01) in dem/den "A-Gruppen"-Kontext(en) codiert (Verarbeitungsblock **3208**) und das Verfahren endet. Andererseits wird, falls das Bit S_A für sämtliche absteigenden (Kinder bzw. Nachkommen) des Koeffizienten C Null ist, die Entscheidung als "Nullbaumwurzel" (00) mit bzw. bei "A-Gruppen"-Kontext(en) codiert (Verarbeitungsblock **3209**). Danach wird der "bestimmt/unbestimmt"-Flag für sämtliche der Absteigenden des Koeffizienten C auf "bestimmt" gesetzt (Verarbeitungsblock **3221**) und das Verfahren endet.

[0178] In einer anderen Ausführungsform kann der Terminierungstest für das Verfahren sein, ob ein bestimmtes gewünschtes Kompressionsverhältnis erreicht wird.

[0179] Gemäß einer Ausführungsform werden die binären Ereignisse, die aus dem B-Durchlauf resultieren, unter dem Markov-Quellenkontextmodell nullter Ordnung entropiecodiert. Das 2-Bit-Alphabet (Größe 4), das aus dem A-Durchlauf resultiert, wird ebenfalls unter der Markov-Quelle nullter Ordnung durch einen 4er (Alphabet der Größe 4) Arithmetikcodierer decodiert.

[0180] Die Fig. 6B und Fig. 6B (fortgesetzt) stellen eine alternative Ausführungsform des Einzellisten-Nullbaum-Codierungsverfahrens nach der vorliegenden Erfindung dar, das einen reduzierten Flagspeicher verwendet. Gemäß einer Ausführungsform kann das Verfahren nach Fig. 6B als der A-Durchgang in dem Verfahren nach **Fig.** 30 verwendet werden. Bezugnehmend auf die Fig. 6B und Fig. 6B (fortgesetzt), beginnt das Verfahren, indem getestet wird, ob das Ergebnis einer UND-Operation des Koeffizienten C mit der Maske M_A Null ergibt (Verfahrensschritt **3201**). Falls nicht, endet das Verfahren. Andererseits, falls das Ergebnis der UND-Operation des Koeffizienten C mit der Maske M_A Null ist, wird die Verarbeitung bzw. das Verfahren bei dem Verfahrens- bzw. Verarbeitungsblock **3202** fortgesetzt, wo ein Test bestimmt, ob der "bestimmt/unbestimmt"-Flag für die Eltern bzw. den Stamm des Koeffizienten C auf "unbestimmt" gesetzt ist, falls der Flag für die Eltern bzw. den Stamm der Koeffizienten auf "unbestimmt" gesetzt ist, endet das Verfahren. Falls jedoch

der "bestimmt/unbestimmt"-Flag für den Stamm des Koeffizienten C auf "unbestimmt" gesetzt ist, wird die Verarbeitung beim Verfahrensblock **3203** fortgesetzt, wo ein Test bestimmt, ob das Bit S_A des Koeffizienten C Eins ist.

[0181] Falls das Bit S_A des Koeffizienten C nicht Eins ist, wird die Verarbeitung beim Verfahrensblock **3207** fortgesetzt. Andererseits wird die Verarbeitung beim Verarbeitungsblock **3204** fortgesetzt, wo ein Test bestimmt, ob das Zeichen des Koeffizienten C positiv ist, falls das Bit S_A des Koeffizienten C Eins ist. Wenn das Zeichen des Koeffizienten C nicht positiv ist, wird die Entscheidung bei "negativ signifikant" in dem/den "A-Gruppen"-Kontext(en) codiert (Verarbeitungsblock **3205**) und das Verfahren endet. Falls das Zeichen des Koeffizienten C positiv ist, wird die Entscheidung als "positiv signifikant" in dem/den "A-Gruppen"-Kontext(en) codiert (Verarbeitungsblock **3206**) und das Verfahren endet. In einer Ausführungsform wird ein 4er-Codierer verwendet und 4er-Entscheidungen werden in einem Kontext codiert. In einer anderen Ausführungsform wird ein binärer Codierer verwendet und drei Kontexte werden verwendet (z.B. sind die drei Kontexte das erste Bit der Entscheidung, das zweite Bit, wenn das erste Bit Null ist und das zweite Bit, wenn das erste Bit Eins ist).

[0182] Beim Verarbeitungsblock **3207** bestimmt ein Test, ob das Bit S_A für sämtliche Absteigenden (Nachkommen) des Koeffizienten C Null ist. Falls das Bit S_A nicht Null ist, wird die Entscheidung als "insignifikant mit signifikanten Nachkommen", "isolierte bzw. getrennte Null" (01) in dem/den "A-Gruppen"-Kontext(en) (Verarbeitungsblock **3208**) codiert, und das Verfahren endet. Andererseits wird, wenn das Bit S_A für sämtliche der Absteigenden (Nachkommen) des Koeffizienten C Null ist, die Entscheidung als "Nullbaumwurzel" (00) in dem/den "A-Gruppen"-Kontext(en) (Verarbeitungsblock **3209**) codiert. Dann wird der "bestimmt/unbestimmt"-Flag für den Koeffizienten C auf "bestimmt" (Verarbeitungsblock **3210**) gesetzt. Danach wird der "bestimmt/unbestimmt"-Flag für sämtliche der Absteigenden des Koeffizienten, welche umgekehrt bzw. folglich Absteigende aufweisen, auf "bestimmt" (Verarbeitungsblock **3211**) gesetzt, und das Verfahren endet.

Decodierungsschritte

[0183] Gemäß der vorliegenden Erfindung wird das Decodieren eng verknüpft bzw. im Schulterschluß mit der Codierung durchgeführt.

[0184] Die Fig. 6C stellt eine Ausführungsform des A-Durchgangsverfahrens für einen Nullbaum-Horizontal-Decodierungsprozeß dar und kann in Verbindung mit dem Verfahren nach **Fig.** 27 verwendet werden. Bezugnehmend auf Fig. 6C beginnt das Verfahren, indem getestet wird, ob der Gruppenflag für den Koeffizienten C auf die "A-Gruppe" gesetzt ist (Verarbeitungsblock **3521**). Falls nicht, endet das Verfahren. Jedoch wird, falls dem so ist, die Verarbeitung bei dem Verarbeitungsblock **3528** fortgesetzt, wo ein Test bestimmt, ob der "bestimmt/unbestimmt"-Flag für den Koeffizienten C auf "unbestimmt" gesetzt ist. Falls nicht endet das Verfahren. Falls dem so ist, wird das Verfahren beim Verarbeitungsblock **3502** fortgesetzt, wo die ternäre Entscheidung in einem A-Gruppen-Kontext(en) decodiert wird.

[0185] Dann bestimmt ein Test, ob die Entscheidung "positiv signifikant" ist (Verarbeitungsblock **3503**). Falls die Entscheidung "positiv signifikant" ist, wird das Zeichen des Koeffizienten C auf positiv gesetzt (Verarbeitungsblock **3505**), die Größenordnung des Koeffizienten wird auf 2^{S_A} gesetzt (Verarbeitungsblock **3507**), der Gruppenflag für den Koeffizienten C wird auf die "B-Gruppe" (Verarbeitungsblock **3541**) gesetzt und das Verfahren endet.

[0186] Falls die Entscheidung nicht "positiv signifikant" (Verfahrensblock **3503**) ist, bestimmt ein Test, ob die Entscheidung "negativ signifikant" ist (Verarbeitungsblock **3504**). Falls die Entscheidung nicht "negativ signifikant" ist, wird das Verfahren beim Verarbeitungsblock **3509** fortgesetzt, wo ein Test bestimmt, ob die Entscheidung eine Nullbaumwurzel ist. Falls die Entscheidung keine Nullbaumwurzel ist, endet das Verfahren. Falls die Entscheidung eine Nullbaumwurzel ist, wird das "bestimmt/unbestimmt"-Flag für sämtliche Absteigenden des Koeffizienten C auf "unbestimmt" (Verarbeitungsblock **3531**) gesetzt und das Verfahren endet.

[0187] Jedoch wird, falls der Test bzw. die Prüfung des Verarbeitungsblocks **3504** bestimmt, daß die Entscheidung "negativ signifikant" ist, anschließend das Zeichen des Koeffizienten C auf negativ gesetzt (Verarbeitungsblock **3506**), die Größenordnung des Koeffiziente auf 2^{S_A} gesetzt (Verarbeitungsblock **3507**), der Gruppenflag für den Koeffizienten C auf die B-Gruppe (Verarbeitungsblock **3541**) gesetzt und das Verfahren endet.

[0188] Die Fig. 6D stellt eine alternative Ausführungsform des A-Durchgangsverfahrens für einen Nullbaum-Horizontal-Decodierungsprozeß unter Verwendung eines reduzierten Flagspeichers dar und kann in dem in **Fig.** 30 beschriebenen Verfahren verwendet werden. Bezugnehmend auf Fig. 6D beginnt das Verfah-

ren durch Testen, ob das Ergebnis einer UND-Operation des Koeffizienten C mit der Maske M_A Null ist (Verarbeitungsblock **3501**). Falls nicht, endet dann der Prozeß. Falls jedoch das Ergebnis der UND-Operation des Koeffizienten C mit der Maske M_A Null ist, wird die Verarbeitung beim Verarbeitungsblock **3508** fortgesetzt, wo ein Test bestimmt, ob der "bestimmt/unbestimmt"-Flag für den Stamm von C "unbestimmt" lautet. Falls nicht, endet das Verfahren. Falls dem so ist, wird dann das Verfahren beim Verarbeitungsblock **3502** fortgesetzt, wo die ternäre Entscheidung in A-Gruppen-Kontext(en) decodiert wird.

[0189] Anschließend bestimmt ein Test, ob die Entscheidung "positiv signifikant" ist (Verarbeitungsblock **3503**). Falls die Entscheidung "positiv signifikant" ist, wird das Zeichen des Koeffizienten auf positiv gesetzt (Verarbeitungsblock **3505**), die Größenordnung des Koeffizienten auf 2^{S}_{A} gesetzt (Verarbeitungsblock **3507**) und das Verfahren endet.

[0190] Falls die Entscheidung nicht "positiv signifikant" ist, bestimmt ein Test, ob sie "negativ signifikant" ist (Verarbeitungsblock **3504**). Falls die Entscheidung nicht "negativ signifikant" ist, wird das Verfahren beim Verarbeitungsblock **3509** fortgesetzt, wo ein Test bestimmt, ob die Entscheidung eine Nullbaumwurzel ist. Falls die Entscheidung keine Nullbaumwurzel ist, endet das Verfahren. Falls die Entscheidung eine Nullbaumwurzel ist, wird der "bestimmt/unbestimmt"-Flag für den Koeffizienten C auf "bestimmt" gesetzt (Verarbeitungsblock **3510**), die "bestimmt/unbestimmt"-Flags für sämtliche Absteigenden des Koeffizienten C, welche umgekehrt bzw. in Folge Absteigende aufweisen, auf "unbestimmt" gesetzt (Verarbeitungsblock **3511**), und das Verfahren endet.

[0191] Jedoch wird dann, wenn der Test vom Verarbeitungsblock **3504** bestimmt, daß die Entscheidung "negativ signifikant" lautet, das Zeichen des Koeffizienten C auf negativ gesetzt (Verarbeitungsblock **3506**), die Größenordnung des Koeffizienten C wird auf 2^{S}_{A} (Verarbeitungsblock **3507**) gesetzt und das Verfahren endet.

[0192] Alternativen existieren gemäß der durch Shapiro vorgenommenen Auswahl, um 4er-Entscheidungen zu verwenden, um Bäume zu beschreiben. Größere Alphabete können verwendet werden, um die Charakteristiken eines vollkommenen Baumes ferner zu spezifizieren, wenn die Wurzel des Baumes codiert wird. In einer Ausführungsform werden die folgenden von 6er-Entscheidungen verwendet.

- insignifikant mit insignifikanten Nachkommen (Nullbaumwurzel)
- insignifikant mit zumindest einem signifikanten Nachkommen
- signifikant, positiv und sämtliche Nachkommen nicht negativ
- signifikant, positiv und zumindest ein Nachkomme ist negativ
- signifikant, negativ und sämtliche Nachkommen sind nicht positiv
- signifikant, negativ und zumindest ein Nachkomme ist positiv

[0193] In dieser Ausführungsform wird Zeicheninformation zusätzlich zur Insignifikanz für einen gesamten Baum vorhergesagt. In anderen Ausführungsformen können Bäume mit anderen Zeichen Zwangs- bzw. Nebenbedingungen oder mit Größenordnungs-Zwangs- bzw. Nebenbedingungen vorhergesagt werden. Alternativ könnten Vorhersager insbesondere bei der Darstellung von Texturen bzw. Strukturen oder bei der Darstellung von Mehr-Auflösungsmerkmalen zweckmäßig sein. Bei größeren Alphabeten kann die Verwendung von Markov-Kontexten höherer Ordnung (wie später beschrieben wird) zweckmäßig sein.

Listenbasierende mehrdurchgangs-verbundene Raum-/Frequenz-Einbettungsmodellierung

[0194] Gemäß der vorliegenden Erfindung wird fequenz-eingebettetes Codieren, wie etwa hierin offenbartes Horizont- bzw. Horizontal-Ordnungsmodellieren, zum Codieren von ternären Ereignissen, die den Koeffizienten in der A-Gruppe entsprechen, verwendet.

[0195] Beim Horizont- bzw. Horizontalcodieren sind sämtliche den Codierungsschritten vorausgehende Initialisierungen identisch zu dem frequenzbasierenden System. In einer Ausführungsform wird binäres Entropiecodieren mit drei Kontexten durchgeführt, der "A-Gruppen-Größenordnung", dem "A-Gruppenzeichen" und der "B-Gruppe".

[0196] Die Fig. 7A ist ein Flußdiagramm einer Ausführungsform des A-Durchgangs für ein Einzellisten-Horizontal-Codierverfahren nach der vorliegenden Erfindung. Dieses Verfahren kann in dem Verfahren nach **Fig.** 27 verwendet werden. Bezugnehmend auf Fig. 7A beginnt das A-Durchgangsverfahren, indem getestet wird, ob der Gruppenflag für den Koeffizienten C auf die "A-Gruppe" (Verfahrensblock **3111**) gesetzt ist. Falls nicht, endet das Verfahren. Falls der Gruppenflag für den Koeffizienten C auf die "A-Gruppe" gesetzt ist, wird die Verarbeitung bei dem Verarbeitungsblock **3102** fortgesetzt, wo ein Test bestimmt, ob das Bit S_A des Koef-

fizienten C Eins ist. Falls das Bit S_A des Koeffizienten C nicht Eins ist, wird die Entscheidung als insignifikant (0) in dem "A-Gruppen"-Kontext (Verarbeitungsblock **3103**) codiert und das Verfahren endet. Falls das Bit S_A des Koeffizienten C Eins ist, dann wird die Verarbeitung beim Verarbeitungsblock **3104** fortgesetzt, wo ein Test bestimmt, ob das Zeichen des Koeffizienten C positiv ist. Falls das Zeichen bzw. Vorzeichen des Koeffizienten C positiv ist, wird die Entscheidung als "positiv signifikant" (10) in dem/den "A-Gruppen"-Kontext(en) kodiert (Verarbeitungsblock **3106**) und das Verfahren wird beim Verarbeitungsblock **3117** fortgesetzt. Andererseits wird, falls das Zeichen bzw. Vorzeichen des Koeffizienten C nicht positiv ist, die Entscheidung als "negativ signifikant" (11) in dem/den "A-Gruppen"-Kontext(en) (Verarbeitungsblock **3105**) codiert und das Verfahren wird beim Verarbeitungsblock **3117** fortgesetzt. Bei dem Verarbeitungsblock **3117** wird der Gruppenflag für den Koeffizienten C auf die "B-Gruppe" gesetzt.

[0197] Die Fig. 7B ist ein Flußdiagramm einer alternativen Ausführungsform des A-Durchgangs für ein Einzellisten-Horizontal-Codierungsverfahren, das einen reduzierten Flagspeicher verwendet. Dieses Verfahren kann in dem Verfahren nach **Fig.** 30 verwendet werden. Bezugnehmend auf Fig. 7B beginnt der A-Durchlauf, indem getestet wird, ob das Ergebnis der UND-Operation des Koeffizienten C mit der Maske M_A Null ergibt (Verarbeitungsblock **3101**). Falls nicht, endet dann das Verfahren. Falls das Ergebnis der UND-Operation des Koeffizienten C mit der Maske M_A Null ist, wird die Verarbeitung beim Verarbeitungsblock **3102** fortgesetzt, wo ein Test bestimmt, ob das Bit S_A des Koeffizienten C Eins ist. Falls das Bit S_A des Koeffizienten C nicht Eins ist, wird die Entscheidung als insignifikant (0) in dem "A-Gruppen"-Kontext (Verarbeitungsblock **3103**) codiert und das Verfahren endet. Falls das Bit S_A des Koeffizienten C Eins ist, dann wird die Verarbeitung beim Verarbeitungsblock **3104** fortgesetzt, wo ein Test bestimmt, ob das Zeichen bzw. Vorzeichen des Koeffizienten C positiv ist. Falls das Vorzeichen des Koeffizienten C positiv ist, wird die Entscheidung als "positiv signifikant" (10) in dem/den "A-Gruppen"-Kontext(en) (Verarbeitungsblock **3106**) codiert und das Verfahren endet. Andererseits wird, falls das Vorzeichen des Koeffizienten C nicht positiv ist, die Entscheidung als "negativ signifikant" (11) in dem/den "A-Gruppen"-Kontext(en) (Verarbeitungsblock **3105**) codiert und das Verfahren endet.

Decodierungsschritte

[0198] Die Fig. 7C stellt eine Ausführungsform des A-Durchgangsverfahrens für ein Einzellisten-Horizontal-Decodierungsverfahren nach der vorliegenden Erfindung dar und kann in dem Verfahren nach **Fig.** 27 verwendet werden. Bezugnehmend auf Fig. 7C beginnt das Verfahren, indem getestet wird, ob der Gruppenflag für den Koeffizienten C auf die "A-Gruppe" gesetzt ist (Verarbeitungsblock **3411**). Falls nicht, endet das Verfahren. Jedoch wird, falls der Gruppenflag für den Koeffizienten C auf die "A-Gruppe" gesetzt ist, die Verarbeitung bei dem Verarbeitungsblock **3402** fortgesetzt, wo die ternäre Entscheidung in dem/den "A-Gruppen"-Kontext(en) codiert wird.

[0199] Anschließend bestimmt ein Test, ob die Entscheidung "positiv signifikant" ist (Verarbeitungsblock **3403**). Falls die Entscheidung "positiv signifikant" ist, wird das Vorzeichen des Koeffizienten C auf positiv gesetzt (Verarbeitungsblock **3405**), die Größenordnung des Koeffizienten auf 2^{S_A} gesetzt (Verarbeitungs- bzw. Verfahrensblock **3407**), der Gruppenflag für den Koeffizienten C wird auf die "B-Gruppe" gesetzt (Verarbeitungsblock **3418**), und das Verfahren endet.

[0200] Falls die Entscheidung nicht "positiv signifikant" lautet, bestimmt ein Test, ob sie "negativ signifikant" (Verarbeitungsblock **3404**) ist. Falls die Entscheidung nicht "negativ signifikant" ist, endet das Verfahren. Jedoch wird dann, wenn die Entscheidung "negativ signifikant" ist, der Koeffizient C auf negativ (Verarbeitungsblock **3406**) gesetzt, die Größenordnung von C auf 2^{S_A} gesetzt (Verarbeitungsblock **3407**), der Gruppenflag für den Koeffizienten C wird auf die "B-Gruppe" gesetzt (Verarbeitungsblock **3418**) und das Verfahren endet.

[0201] Die Fig. 7D stellt eine alternative Ausführungsform des A-Durchlaufverfahrens für ein Einzellisten-Horizontal-Decodierungsverfahren unter Verwendung eines reduzierten Flagspeichers dar und kann in dem Verfahren nach **Fig.** 30 verwendet werden. Bezugnehmend auf Fig. 7D beginnt das Verfahren, indem getestet wird, ob das Ergebnis der UND-Operation des Koeffizienten C mit der Maske M_A Null ist (Verarbeitungsblock **3401**). Falls nicht, endet das Verfahren. Jedoch wird, falls das Ergebnis der UND-Operation des Koeffizienten C mit der Maske M_A Null ist, die Verarbeitung beim Verarbeitungsblock **3402** fortgesetzt, wo die ternäre Entscheidung in dem/den A-Gruppen-Kontexten) decodiert wird.

[0202] Anschließend bestimmt ein Test, ob die Entscheidung "positiv signifikant" ist (Verarbeitungsblock **3403**). Falls die Entscheidung "positiv signifikant" ist, wird das Zeichen bzw. Vorzeichen des Koeffizienten C auf positiv gesetzt (Verarbeitungsblock **3405**), die Größenordnung des Koeffizienten wird auf 2^{S_A} gesetzt (Verarbeitungsblock **3407**) und das Verfahren endet.

[0203] Falls die Entscheidung nicht "positiv signifikant" ist, bestimmt ein Test, ob sie "negativ signifikant" (Verarbeitungsblock **3404**) ist. Falls die Entscheidung nicht "negativ signifikant" ist, endet das Verfahren. Jedoch wird, falls die Entscheidung "negativ signifikant" lautet, dann das Vorzeichen des Koeffizienten C auf negativ gesetzt (Verarbeitungsblock **3406**), die Größenordnung von C wird auf 2^{S}_{A} gesetzt (Verarbeitungsblock **3407**) und das Verfahren endet.

B-Durchlauf sowohl für Nullbaum als auch für Horizont

[0204] In einer Ausführungsform ist das B-Durchlaufverfahren sowohl für den Nullbaum als auch für den Horizont nach der vorliegenden Erfindung das gleiche. Ausführungsformen für den B-Durchlaufalgorithmus für das Codierverfahren und das Decodierverfahren sind in den Fig. 8A und Fig. 8B bzw. Fig. 9A und Fig. 9B gezeigt.

[0205] Die Fig. 8A stellt eine Ausführungsform eines B-Durchlaufverfahrens dar, das teilweise für Nullbaum und Einzellisten-Horizontal-Codierungsverfahren verwendet wird und in dem Verfahren nach **Fig.** 27 verwendet werden kann. Bezugnehmend auf die Fig. 8A prüft das Verfahren eingangs, ob der Gruppenflag für den Koeffizienten C gesetzt ist (Verfahrensblock **3311**), falls nicht, endet das Verfahren. Andererseits wird, falls der Gruppenflag gesetzt ist, die Verarbeitung beim Verarbeitungsblock **3302** fortgesetzt, wo ein Test bestimmt, ob das Bit S_B des Koeffizienten C "1" ist. Falls das Bit von S_B des Koeffizienten C nicht "1" ist, dann wird die Entscheidung als "0" in dem/den "B-Gruppen"-Kontext(en) (Verarbeitungsblock **3303**) codiert und das Verfahren endet. Falls das Bit S_B des Koeffizienten C "1" ist, dann wird die Entscheidung als "1" in dem/den "B-Gruppen"-Kontext(en) (Verarbeitungsblock **3304**) codiert und das Verfahren endet.

[0206] Die Fig. 8B stellt eine alternative Ausführungsform des B-Durchgangsverfahrens dar, das teilweise für Nullbaum- und Einzellisten-Horizontal-Codierungsverfahren verwendet wird und einen reduzierten Flagspeicher verwendet und in dem Verfahren nach **Fig.** 30 verwendet werden kann. Bezugnehmend auf die Fig. 8B prüft das Verfahren anfangs, ob das Ergebnis einer UND-Operation des Koeffizienten C mit der Maske M_B nicht Null ist (Verarbeitungsblock **3301**). Falls nicht, endet das Verfahren. Andererseits wird, wenn das Ergebnis der UND-Operation des Koeffizienten C mit der Maske M_B nicht Null ist, die Verarbeitung beim Verarbeitungsblock **3302** fortgesetzt, wo ein Test bestimmt, ob das Bit S_B des Koeffizienten C "1" ist. Falls das Bit von S_B des Koeffizienten C nicht "1" ist, dann wird die Entscheidung als "0" in dem/den "B-Gruppen"-Kontext(en) (Verarbeitungsblock **3303**) codiert und das Verfahren endet. Falls das Bit S_B des Koeffizienten C "1" ist, dann wird die Entscheidung als "1" in dem/den "B-Gruppen"-Kontext(en) (Verarbeitungsblock **3304**) codiert und das Verfahren endet.

[0207] Die Fig. 9A stellt eine Ausführungsform der B-Durchlaufdecodierung nach der vorliegenden Erfindung dar und kann in dem Verfahren nach **Fig.** 27 verwendet werden. Bezugnehmend auf die Fig. 9A bestimmt anfangs ein Test, ob der Gruppenflag für den Koeffizienten C auf die "B-Gruppe" gesetzt ist (Verarbeitungsblock **3611**). Falls nicht, endet das Verfahren. Jedoch werden, falls der Gruppenflag für den Koeffizienten C auf die "B-Gruppe" gesetzt ist, dann die Entscheidungen in dem/den "B-Gruppen"-Kontext(en) (Verarbeitungsblock **3602**) decodiert. Ein Test entscheidet dann, ob die Entscheidung eine "1" ist (Verarbeitungsblock **3603**). Falls die Entscheidung keine "1" ist, endet das Verfahren. Falls die Entscheidung eine "1" ist, wird das Bit S_B des Koeffizienten C gesetzt (Verarbeitungsblock **3604**) und das Verfahren endet.

[0208] Die Fig. 9B stellt eine alternative Ausführungsform der B-Durchlaufdecodierung nach der vorliegenden Erfindung unter Verwendung eines reduzierten Flagspeichers dar und kann in dem Verfahren nach **Fig.** 30 verwendet werden. Bezugnehmend auf Fig. 9B bestimmt ein Testanfang, ob das Ergebnis einer UND-Operation des Koeffizienten C mit der Maske M_B nicht Null ist (Verarbeitungsblock **3601**). Falls das Ergebnis der UND-Operation des Koeffizienten C mit der Maske M_B Null ist, endet das Verfahren. Jedoch werden, falls das Ergebnis der UND-Operation des Koeffizienten C mit der Maske M_B nicht Null ist, die Entscheidungen in dem/den "B-Gruppen"-Kontext(en) (Verfahrensblock **3602**) decodiert. Ein Test entscheidet dann, ob die Entscheidung eine "1" ist (Verarbeitungsblock **3603**). Falls die Entscheidung keine "1" ist, endet das Verfahren. Falls die Entscheidung eine "1" ist, wird das Bit S_B des Koeffizienten C (Verarbeitungsblock **3604**) gesetzt und das Verfahren endet.

[0209] Unter Verwendung der Kombination des Nullbaum-Ordnungscodierens und des Horizontal-Ordnungscodierens stellt die vorliegende Erfindung eine bitsignifikante Codierung der Koeffizienten, die durch reversible Kleinwellen erzeugt werden, zur Verfügung. Man beachte, daß die Verwendung sowohl der A-Gruppe als auch der B-Gruppe und der ternären und der binären Ereignisse, die den "A"- und "B"-Durchläufen jeweils entsprechen, insbesondere im Hinblick auf die Tatsache wichtig ist, daß ein Schalter aus der Verwendung des Null-

baum-Ordnens zu dem Horizontal-Ordnen an dem Ende von jedem A-Durchlauf gemacht wird. Dies kompensiert die Ineffizienz bei der Vorhersage, die das Nullbaumordnen bei den Bits niederer Ordnung begleitet. Deshalb beginnt gemäß der vorliegenden Erfindung das System, indem die Bitdaten höherer Ordnung nullbaum-codiert werden und nach einer Anzahl von Durchläufen durch die Listen, d.h. nachdem eine Anzahl von Bitebenen codiert sind, schaltet der Codierer nach der vorliegenden Erfindung um, um den Rest der Daten unter Verwendung von Horizontal-Codieren zu codieren. Die Anzahl der Durchläufe kann statistisch ausgewählt werden oder kann angepaßt ausgewählt werden, indem die Durchführung bzw. Funktion des Nullbaum-Ordnungs-Codierungsblocks überwacht wird.

Alternativen zum Kontextmodell

[0210] In einer Ausführungsform werden fünf binäre Kontext-Überrahmen bzw. -Rahmen verwendet. Dies ist gering, falls dies mit anderen Systemen, wie etwa JBIG, verglichen wird, welches etwas mehr als 1024 Kontexte verwendet. Die Kompression kann unter Verwendung von mehr Kontext-Überrahmen bzw. -Rahmen verbessert werden. Entscheidungen können anhand von räumlichen Anordnungen, Pegeln und/oder Bitstellungen verarbeitet bzw. aufbereitet werden. Im allgemeinen können die vorher beschriebenen Markov-Kontexte nullter Ordnung durch Markov-Kontexte höherer Ordnung ersetzt werden.

[0211] Einige Beispiele lauten wie folgt. Das signifikanteste (und deshalb am leichtesten vorhersagbare) Bit von jeder Mantisse (B-Gruppendaten in einigen Ausführungsformen) könnte einen von dem Rest der Bits verschiedenen Kontext verwenden. Die Signifikanz-/Nicht-Signifikanz-Entscheidung könnte auf der Grundlage der gleichen Entscheidung aufbereitet bzw. verarbeitet werden, wie sie für räumlich eng beieinanderliegende vorherige Koeffizienten bei dem gleichen Transformationspegel gemacht worden ist. Gleichermaßen können die Zeichen- bzw. Vorzeichenbits für die signifikanten Koeffizienten auf den Zeichen von räumlich dicht davorliegenden Koeffizienten bei dem gleichen Pegel oder den Zeichen bzw. Vorzeichen des Koeffizienten des Stamms bzw. der Eltern verarbeitet bzw. aufbereitet werden.

[0212] Verbesserungen des Kontextmodells können insbesondere wichtig sein, wenn Bilder komprimiert werden, die eine räumliche oder eine Multi-Auflösungsstruktur aufweisen. Graustufenbilder von Linienzeichnungen oder Texte sind ein Beispiel für Bilder mit beiden dieser Arten von Strukturen. Verbesserungen sind auch für die Kompression von Daten wichtig, die bereits mit einem spezifizierten Spitzenfehler hätten komprimiert bzw. dekomprimiert werden sollen.

Alternative Ausführungsformen der vorliegenden Erfindung

[0213] Die vorliegende Erfindung kann in Hardware und/oder Software in die Tat umgesetzt werden. Eine Hardware-Umsetzung der vorliegenden Erfindung erfordert, Umsetzung der Kleinwellenfilter, des Speicher-/Datenflußmanagements, um Daten für die Filter zur Verfügung zu stellen, ein Kontextmodell, um die Einbettungscodierung nach der vorliegenden Erfindung zu steuern, das Speicher-/Datenflußmanagement, um die Daten für das Kontextmodell zur Verfügung zu stellen, und einen binären Entropiecodierer.

Klein- bzw. Unterwellenfilter

[0214] Eine Ausführungsform des Vorwärts-Kleinwellenfilters nach der vorliegenden Erfindung ist in Fig. 10 dargestellt. Der in Fig. 10 gezeigte Kleinwellenfilter bezieht ein bzw. beherbergt vier 16-Bit-Zweier-Komplement-Eingabebildelemente, die als x(2)–x(5) gezeigt sind.

[0215] Bezugnehmend auf Fig. 10 verwenden die Tiefpaßfilter mit zwei Abgriffen bzw. Stufen "1 1" einen 16-Bit-Addierer **1001**. Die Ausgänge werden als S bzw. D bezeichnet. Der Ausgang des Addierers (S) ist auf 16 Bits unter Verwendung eines 1-Verschiebungsblocks **1003** auf 16 Bits verringert. Der 1-Verschiebungsblock **1003** führt eine Divisiondurch-2-Funktion durch Verschieben seines 17-Bit-Einganges zu dem rechten einen Bit bzw. dem richtigen Bit durch.

[0216] Der Hochpaßfilter mit sechs Abgriffen bzw. Stufen "–1 –1 8 –8 1 1" erfordert die Berechnung von $-S_0 + 4D_1 + S_2$. Die Funktion $S_2 - S_0$ wird mit einem 16-Bit-Subtrahierer **1005** berechnet, der den Ausgang des 1-Verschiebungsblocks **1003** und das $Y_0(0)$ empfängt. Der Ausdruck $4D_1$ wird unter Verwendung des Subtrahierers **1002** und des 2-Verschiebungsblocks **1004** berechnet. Der Ausgang, der durch den 16-Bit-Subtrahierer **1002** erzeugt wird, wird auf die linken zwei Plätze verschoben, wodurch sein Ausgang effektiv mit vier multipliziert wird. Das Addieren des Ausgangs $4D_1$ von dem 2-Verschiebungsblock **1004** zu dem Ausgang des Subtrahierers **1005** wird durch den 20-Bit-Addierer **1006** durchgeführt. Der Ausgang des Schlußaddierers ist unter

Verwendung des 2-Verschiebungsblocks **1007** auf 18 Bits verringert. Der 2-Verschiebungsblock **1007** führt eine Division-durch-4-Funktion durch, indem sein 20-Bit-Eingang zu den rechten zwei Bits verschoben wird.

[0217] Folglich ist die gesamte erforderliche Berechnungshardware (Register für die zeitweise Speicherung von Ergebnissen nicht eingerechnet):
- @ 16-Bit-Addierer,
- @ 16-Bit-Subtrahierer,
- @ 19-Bit-Addierer.

[0218] Man beachte, daß das Verschieben durch die Verdrahtung vorgenommen wird, so daß keine Logik erforderlich ist.

[0219] In anderen Ausführungsformen können für Eingänge der Größe N, ein N-Bit-Addierer, zwei N-Bit-Subtrahierer und ein (N + 3)-Bit-Addierer verwendet werden.

[0220] Wegen der extrem niedrigen Hardwarekosten für diese Addierer/Subtrahierer können parallele Umsetzungen für die Filter verwendet werden, falls dies gewünscht ist.

[0221] Man bemerke, daß alternativ anstelle des Subtrahierens von X(3) und X(2) X(4)–X(5) berechnet werden kann und aufrechterhalten werden kann, bis es später als X(2)–X(3) für die nächste Verschiebung oder Anwendung des Filters benötigt wird. Sowohl der Vorwärtsfilter (und der oben beschriebene inverse bzw. Rückwärtsfilter) können in einer Leitung angeordnet werden, um einen höheren Durchsatz zu erzielen.

[0222] Der inverse Wellen- bzw. Kleinwellenfilter wird in Fig. 11 dargestellt. Die Eingänge von $Y_0(0)$ und $Y_0(2)$ werden durch den Subtrahierer **1101** subtrahiert. Das Ergebnis der Subtraktion wird durch den Um-2-Verschiebungsblock **1102** um zwei Bits nach rechts verschoben. Dies dividiert den Ausgang des Subtrahierers effektiv durch vier. Eine Subtraktion wird zwischen dem Ausgang des Um-2-Verschiebungsblocks **1104** und des $Y_1(0)$-Eingang durchgeführt. Der Eingang $Y_0(1)$ wird um ein Bit nach links durch den Um-1-Verschiebungsblock **1103** um ein Bit nach links verschoben, wodurch der Eingang mit 2 multipliziert wird. Nachdem $Y_0(1)$ um eins verschoben ist (multipliziert mit 2), ist das LSB des verschobenen Wertes das LSB, das von dem Ausgang des Subtrahierers **1104** genommen ist, und wird mit dem 16-Bit-Ausgang von dem Um-1-Verschiebungsblock **1103** kombiniert, um einen Eingang für den Addierer **1105** und den Subtrahierer **1106** zu bilden. Der andere Eingang für den Addierer **1105** und den Subtrahierer **1106** ist der Ausgang des Subtrahierers **1104**. Die Ausgänge des Addierers **1105** und des Subtrahierers **1106** können nachfolgend abgeschnitten werden bzw. einer Abschneidung unterzogen werden.

[0223] Eine Auswahl von zwei Abschneide-Operationen kann verwendet werden. In beiden Fällen wird der 20-Bit-Wert um eins verschoben (durch 2 geteilt) zu einem 19-Bit-Wert. Für ein System, das nur eine verlustlose Kompression durchführt, können die 16 am wenigsten signifikanten Bits ausgegeben werden (die verbleibenden drei Bits können außer acht gelassen werden). In einem verlustbehafteten System (oder einem verlustbehafteten/verlustlosen System) wird der 19-Bit-Wert auf Null gesetzt, falls er negativ ist, oder auf 2^{16}-1 gesetzt, falls er größer als 2^{16}-1 ist; ansonsten können die 16 am wenigsten signifikanten Bits ausgegeben werden.

[0224] Für Eingänge von N-Bits können ein N-Bit-Subtrahierer, ein (N + 2)-Bit-Subtrahierer, ein (N + 3)-Bit-Addierer und ein (N + 3)-Bit-Subtrahierer verwendet werden und die Abschneide- bzw. Kürzungseinrichtung gibt N Bits aus.

Speicherverwendung

[0225] Im Hinblick auf das Speicher- und Datenflußmanagement für die Wellen- bzw. Kleinwellenfilter nach der vorliegenden Erfindung für Bilder, wo ein vollständiger Rahmen in einen Speicher passen kann, ist das Speicher-/Datenflußmanagement keine schwierige Angelegenheit. Auch für viele Anwendungen ist es vernünftig, was für 1024 × 1024 medizinische 16-Bit-Bilder (z.B. mit einer Größe von 2 Megabytes), die einen vollständigen Pufferrahmen erfordern, ist. Für größere Bilder (z.B. A4, 400 DPI 4-Farbenbilder weisen eine Größe von etwa 50 Megabyte auf) ist die Durchführung der Kleinwellentransformation mit einer begrenzten Menge an Linien-Pufferspeicher wünschenswert.

[0226] Man bemerke, daß ein vollständiger Rahmenpuffer nicht erforderlich ist, um die vorliegende Erfindung in ein System mit einem Durchlauf umzusetzen. Deshalb kann der erforderliche Speicher um einen Faktor 100

verringert werden (verglichen mit einem vollständen Rahmenpuffer für große Bilder). Das System mit einem Durchlauf gemäß der vorliegenden Erfindung wird später beschrieben.

[0227] Die in dem Filterspeicher gespeicherten Daten sind eine Reihe von Koeffizienten, die dem Einbettungscodieren und dem binären Entropiecodieren auszusetzen sind. Das Einbettungscodieren verwendet ein Kontextmodell, um die Verwendung von frequenzbasierender Codierung oder Horizontalcodierung zu koordinieren, und um Daten in einer zweckmäßigen bzw. angemessenen Ordnung zur Verfügung zu stellen. Das Kontextmodell arbeitet in Verbindung mit einem Speicher-Managementschema. Für Systeme mit einem vollständigen Rahmenpuffer ist es nicht schwierig, Daten in der angemessenen Ordnung zur Verfügung zu stellen. Für Systeme ohne einen vollständigen Rahmenpuffer stellt das Datentransformations-Managementschema der Ausführungsform mit einem Durchlauf gemäß der vorliegenden Erfindung (unten beschrieben) dem Kontextmodell Koeffizienten zur Verfügung, so daß es das Kontextmodell nur erfordert, Koeffizienten für einen Baum zu puffern. Ein frequenzbasierendes Kontextmodell mit einem Durchlauf und ein verbundenes Raum-/Frequenz-Kontextmodell mit einem Durchlauf werden zu einer Zeit bzw. einem Zeitpunkt an einem Baum betrieben.

[0228] Die Ergebnisse der Einbettungsoperation gemäß der vorliegenden Erfindung ist zum Erzeugen von Bit-Strömen von dem frequenzbasierenden Modellierungsmechanismus gemäß der vorliegenden Erfindung und dem verbundenen Raum-/Frequenz-Modellierungsmechanismus gemäß der vorliegenden Erfindung. Diese Bit-Ströme werden anschließend unter Verwendung eines binären Entropiecodierers codiert.

[0229] Für Systeme mit einem vollständigen Rahmenpuffer können beliebige binäre Entropiecodierer oder andere angemessene Codierer verwendet werden. Für Systeme ohne einen vollständigen Rahmenpuffer müssen entweder mehrfache unabhängige Codierer verwendet werden oder der Codierer muß dazu in der Lage sein, mehrfache unabhängige Codierer zu simulieren. Auch wird ein Speicher oder ein Kanal-Management benötigt, um die Ausgänge von den unabhängigen Codierern zu verfolgen bzw. auf dem laufenden zu halten. Der Vorteil der vorliegenden Erfindung ist, daß die Daten, die zu handhaben bzw. zu managen sind, prioritiert sind (eingebettet). Falls während der Kompression oder Übertragung nicht ausreichend Raum bzw. Platz oder Bandbreite zur Verfügung steht, können weniger wichtige Daten auf dem Wege außer acht gelassen werden, um eine vernünftige verlustbehaftete Kompression zur Verfügung zu stellen.

System mit einem Durchlauf gemäß der vorliegenden Erfindung

[0230] Die vorliegende Erfindung stellt eine Transformation mit einem Durchlauf zur Verfügung, die es ermöglicht, die Eingabedaten in das System, wie sie empfangen werden, vollständig zu prozessieren. In einem solchen System ist die Verarbeitung der Daten nicht von den Daten abhängig, die folgen. Der erforderliche Speicher, um ein Bild zu komprimieren, ist abhängig von der Länge des Bildes. Durch die Entfernung der Abhängigkeit stellt die vorliegende Erfindung ein System zur Verfügung, das komprimierte Daten ausgeben kann, bevor sämtliche Daten verarbeitet worden sind.

A. Datenmanagement für die Transformation mit einem Durchlauf

[0231] Die Fig. 12 stellt einen Bereich eines Bildes dar, das in einer Rasterordnung durch eine Band- bzw. Frequenzbandmode bzw. -methode unter Verwendung der Lehren gemäß der vorliegenden Erfindung komprimiert. Man ziehe eine Zerlegung mit vier Pegeln in Betracht. Jeder Baum weist $2^4 \times 2^4 = 16 \times 16 = 256$ Koeffizienten auf. Jedoch hängt jeder Baum von mehr als 256 Eingabebildelementen ab, da der Hochpaßfilter der Kleinwellentransformation gemäß der vorliegenden Erfindung überlappt. Der "1 1"-Tiefpaßfilter mit zwei Abgriffen bzw. Stufen (L) bewirkt keine Überlappung und die gesamte Überlappung stammt von dem Hochpaßfilter (H) mit sechs Abgriffen bzw. Stufen "–1 –1 8 –8 1 1". Die größte Überlappung tritt für die Kaskade von drei Anwendungen des Tiefpaßfilter, gefolgt durch eine Anwendung des Hochpaßfilters (LLLH), auf. Drei Anwendungen des Tiefpaßfilters (LLL) erfordern eine Unterstützung von $2^3 = 8$ Eingabebildelementen. Stütz- bzw. Unterstützungsbereiche mit einer Größe von 8 × 8 Bildelementen sind in Fig. 12 gezeigt. Wenn der Hochpaßfilter in diese Kaskade einbezogen wird, betragen die Unterstützungsbereiche $(6 \times 2^3) \times (6 \times 2^3) = 48 \times 48$ Bildelemente. Ein Unterstützungsbereich mit 48 × 48 Bildelementen, der aus sechsunddreißig 8 × 8 Blöcken besteht, ist in Fig. 12 gezeigt.

[0232] Man nehme an, daß die Koeffizienten des in Fig. 12 gezeigten Unterstützungsbereichs mit 48 × 48 Bildelementen gegenwärtig gerade verarbeitet werden. Der hell schattierte Abschnitt des Unterstützungsbereichs stellt Bildelemente dar, die bereits in vorangehenden Unterstützungsbereichen verwendet worden sind. Der hell schattierte Teil, der außerhalb des Unterstützungsbereiches ist, stellt Bildelemente dar, die bereits in

vorangehenden Unterstützungsbereichen verwendet worden sind und in zukünftigen Unterstützungsbereichen benötigt werden. Der schwarze 16 × 16 Bereich ist der Teil des Unterstützungsbereichs, der Bildelemente enthält, die zuvor nicht verwendet worden sind. In ähnlicher Weise enthält der dunkel schattierte 16 × 16 Bereich Bildelemente, die zuvor nicht verwendet worden sind, welche jedoch in dem nächsten 48 × 48 Unterstützungsbereich verwendet werden. Eine 16 × 16 Transformation eines Baumpegels wird berechnet, wobei die vorherigen Ergebnisse von acht anderen 16 × 16 Transformationen mit drei Pegeln aus einem Puffer zurückgerufen werden und der vierte Pegel der Transformation wird auf die neun 16 × 16 Transformationen mit drei Pegeln angewendet. Die erforderliche Pufferung, um dies vorzunehmen, reicht aus, um die dreipegeligen Transformationskoeffizienten für (2 × Breite des Bildes + 32) × 16 Bildelemente zu speichern und reicht aus, um einen 16-Linien-Puffer (ein Band bzw. Frequenzband) von Bildelementen zu speichern.

[0233] Die Fig. 13 ist ein Blockdiagramm einer Ausführungsform einer Kleinwellenfilterungseinheit mit einem Durchlauf, die eine Filtersteuereinheit **1301**, einen Speicher **1302** und einen Filter **1303** enthält. Der Filter **1303** weist den in Verbindung mit Fig. 10 beschriebenen Filter auf. Der Speicher **1302** bezieht sich auf den Speicher, der oben in Verbindung mit Fig. 12 beschrieben worden ist, und speichert entweder Bildelemente oder Koeffizienten. Die Filtersteuereinheit **1301** bestimmt den Datenfluß zwischen dem Speicher **1302** und dem Filter **1303**. Die Operation der Filtersteuereinheit **1301** wird unten beschrieben.

[0234] Die Fig. 14 stellt eine alternative Kleinwellenfiltereinheit dar. Um einen Hochgeschwindigkeitsbetrieb zu erzielen, können Filter mehrfach verwendet werden. In einer Ausführungsform könnte die Speicherbandbreite wesentlich sein, da der Filter **1303** vier oder fünf Eingänge erfordert (beispielsweise inverser Filter, Vorwärtsfilter). Der Speicher könnte mehrfache Bildelemente/Koeffizienten pro Stelle bzw. Ort, mehrfache Bänke und/oder mehrfache Anschlüsse bzw. Ports aufweisen. Eine Speicherschnittstelleneinheit **1401** verringert die Bandbreite des Speichers, die erforderlich ist, indem kleine Puffer für lokale Daten, die während der Verarbeitung benötigt werden, vorgesehen werden. Die Speicherschnittstelleneinheit **1401** stellt auch Multiplexen/Demultiplexen zwischen dem Eingang/Ausgang (I/O) des Speichers **1302** und den I/O des Filters **1303** zur Verfügung.

[0235] Zusätzlich zu der Speicherbandbreite, die für das Filtern erforderlich ist, kann zusätzliche Bandbreite für die Eingabe der Bildelemente in den Speicher **1302** und die Ausgabe der Koeffizienten zu dem Kontextmodell erforderlich sein. Wenn Bildelemente bzw. Bildpunkte in der Rasterordnung eingegeben werden, kann zusätzlicher Speicher für den Band- bzw. Frequenzbandpuffer erforderlich werden.

[0236] Falls ein Speicher mehrere bzw. Mehrfachelemente (Bildelemente oder Koeffizienten) pro Stelle bzw. Ort speichert, statt horizontale oder vertikale benachbarte Elemente in einer Reihe oder Spalte zu speichern, kann es die Menge an Speicherzugriffen und Pufferungen reduzieren, die erforderlich ist, wenn Elemente in einem N × N-Block die gleiche Stelle bzw. den gleichen Ort teilen, wenn N eine Potenz von 2 ist. Dies ermöglicht die gleiche Angemessenheit für vertikale und für horizontale Zugriffe.

[0237] Mehrfache Bänke bzw. Speicherbänke können auch in die Tat umgesetzt werden, so daß sowohl horizontale als auch vertikale Zugriffe die gleichen Vorteile aus mehrfachen Bänken ziehen können, wie in Fig. 15 gezeigt wird. Für den Fall mit zwei Bänken kann ein Bank-Auswahl-Bit ausgebildet werden, das zur Verfügung gestellt wird, um eine der Bänke auszuwählen, und zwar in einer Ausführungsform durch eine Exklusiv-Oder-Operation der LSBs der horizontalen und der vertikalen Koordinaten. Für den Fall mit vier Bänken können die zwei Bank-Auswähl-Bits ausgebildet werden, indem (Modul 4 mit einem 2-Bit-Addierer) die zwei LSBs der horizontalen und vertikalen Koordinaten addiert werden.

[0238] Die Fig. 16 stellt die Filteroperation mit einem Durchlauf für eine Zerlegungsimplementation mit zwei Pegeln bzw. zwei Niveaus durch die Filtersteuereinheit **1301** (Fig. 13) dar. Man bemerke, daß zu Darstellungszwecken eine Beschreibung mit zwei Pegeln zuerst erörtert wird, um die allgemeine Technik gemäß der vorliegenden Erfindung darzustellen. In anderen Ausführungsformen werden dreipegelige, vierpegelige oder Zerlegungen mit höheren Pegeln verwendet. Eine zweipegelige Zerlegung weist 16 Koeffizienten pro Baum auf und erfordert eine Berechnung mit 16 Eingabebildelementen, die zuvor nicht verwendet worden sind. Das Filtern für einen Baum mit Koeffizienten wird in 16 oder weniger Zeiteinheiten durchgeführt, um der Eingabe- und Ausgaberate zu entsprechen. Für dieses Beispiel werden zwei Filter in Parallelanordnung bzw. parallel betrieben verwendet, um den gewünschten Durchsatz von zwei Filteroperationen pro Zeiteinheit zu erzielen. Für jede räumliche Stelle bzw. jeden räumlichen Ort, wo die führende Kante bzw. Flanke eines Filters angelegt wird, zeigt Fig. 16 eine Zahl, die die Zeit anzeigt, in der jede Filterbetätigung durchgeführt wird.

[0239] Da die Ordnung des Filterns durch die führende Kante des Filters bestimmt wird, erzeugt das Filtern

nicht sämtliche der Koeffizienten eines Baumes, bevor einige bzw. beliebige der Koeffizienten des nächsten Baumes erzeugt sind. Die Filterung der Nachkommen des Baumes tritt vor dem Filtern der Stämme bzw. Eltern auf und das Tiefpaßfiltern wird vor dem entsprechenden Hochpaßfiltern vorgenommen. Das Filtern betreibt eine A-Gruppe von Koeffizienten, die die gleiche Anzahl von Koeffizienten eines Typs aufweist, wie ein Baum.

[0240] Die Horizontal-Filterung mit dem Pegel 1 wird während einer Zeit 0 bis 7 durchgeführt und die Ergebnisse werden in einem temporären Puffer gespeichert. (Jede räumliche Stelle bzw. jeder räumliche Ort ergibt zwei Koeffizienten.) Während einer Zeit 2 bis 9 wird vertikales Filtern an den Daten in dem Puffer (unter Verwendung des zweiten Filters) und den Daten der vorherigen horizontalen Filter vom Speicher (zweifach pro räumlichem Ort bzw. Stelle) durchgeführt. Vertikales Filtern kann beginnen, sobald die zweite horizontale Filterungsoperation abgeschlossen ist. Die Koeffizienten HH, HL und LH sind fertig für die Ausgabe zu dem Kontextmodell (zu der angemessenen Zeit). Die Koeffizienten LL werden in dem nächsten Pegel bzw. auf dem nächsten Niveau verwendet.

[0241] Mit nur zwei Filtern kann die Null-Horizontalfilterung mit Pegel 0 bis zur Zeit 8 nicht beginnen, wenn die Horizontalfilterung mit Pegel 1 nicht vollständig ist, was einen Filter verfügbar werden läßt. Die Horizontalfilterung mit Pegel 0 kann bis zur Zeit 10 nicht beendet werden, einem Zyklus nach dem die Vertikalfilterung mit Pegel 0 vollständig ist, wobei sämtliche erhaltenen Daten bestätigt bzw. bewiesen werden. Während der Zeit 11 und 12 kann als nächstes die vertikale Filterung mit Pegel 1 auftreten.

[0242] Die Tabelle 1 unten faßt den Betrieb jeden Filters während jeder Zeiteinheit zusammen. Das Format der Eingänge lautet Pegelzahl, horizontal oder vertikal ("H" oder "V") und die räumliche Stelle bzw. Anordnung der führenden Kante. Die Eingänge der vertikalen Filteroperationen sind auch als Tiefpaß zu Hochpaß erkannt, mit einem Index "$_L$" oder "$_H$". Man bemerke, daß es erforderlich ist, einen Filter zuzuordnen, um horizontale Filterung durchzuführen, und den anderen, um vertikale Filterung durchzuführen, da beide Filter identisch sind.

Tabelle 1

Zeit	Filter 1	Filter 2
0	1H(0,0)	(außer Betrieb)
1	1H(0,1)	
2	1H(2,0)	$1V_L(0,0)$
3	1H(2,1)	$1V_H(0,0)$
4	1H(0,2)	$1V_L(0,2)$
5	1H(0,3)	$1V_H(2,0)$
6	1H(2,2)	$1V_L(0,2)$
7	1H(2,3)	$1V_H(0,2)$
8	0H(0,0)	$1V_L(2,2)$
9	(außer Betrieb)	$1V_H(2,2)$
10	0H(0,1)	(außer Betrieb)
11	(außer Betrieb)	$0V_L(0,0)$
12		$0V_H(0,0)$
13		(außer Betrieb)
14		
15		

[0243] Während die horizontale Filterung mit Pegel 1 wieder für die nächste Gruppe von Eingabebildelementen bei einer Zeit 11 beginnen kann, würde dies den Filter dazu veranlassen, schneller betätigt zu werden als dies Eingabe- und Ausgaberate. Stattdessen werden gemäß der vorliegenden Erfindung die Filter außer Betrieb sein und die nächste Gruppe wird bei der Zeit 16 gestartet. Zyklen mit außer Betrieb gesetzten Filtern können für Speicherübertragungen verwendet werden. Anstelle an dem Ende des Filterns für jede Gruppe aufzutreten, können die Außer-Betrieb-Zyklen, falls gewünscht, unter den Filterzyklen verteilt werden.

[0244] Im Hinblick auf die Erläuterung zu dem Fall mit zwei Pegel wird der Fall mit drei Pegeln in Tabelle 2 gezeigt. Kettenschaltungen bzw. Kaskadierungen von zwei oder vier Zeiteinheiten werden verwendet, um die Informationen auf eine Seite zu übertragen, wodurch es leichter gemacht wird, zu lesen.

Tabelle 2

Zeit	Filter 1	Filter 2
0-3	2H(0,0),2H(0,1),2H(2,0),2H(2,1)	(außer Betrieb),(außer Betrieb), $2V_L(0,0)$, $2V_H(0,0)$
4-7	2H(4,0),2H(4,1),2H(6,0),2H(6,1)	$2V_L(2,0),2V_H(2,0),2V_L(4,0),2V_H(4,0)$
8-11	2H(0,2),2H(0,3),2H(2,2),2H(2,3)	$2V_L(6,0),2V_H(6,0),2V_L(0,2),2V_H(0,2)$
12-15	2H(4,2),2H(4,3),2H(6,2),2H(6,3)	$2V_L(2,2),2V_H(2,2),2V_L(4,2),2V_H(4,2)$
16-19	2H(0,4),2H(0,5),2H(2,4),2H(2,5)	$2V_L(6,2),2V_H(6,2),2V_L(0,4),2V_H(0,4)$
20-23	2H(4,4),2H(4,5),2H(6,4),2H(6,5)	$2V_L(2,4),2V_H(2,4),2V_L(4,4),2V_H(4,4)$
24-27	2H(0,6),2H(0,7),2H(2,6),2H(2,7)	$2V_L(6,4),2V_H(6,4),2V_L(0,6),2V_H(0,6)$
28-31	2H(4,6),2H(4,7),2H(6,6),2H(6,7)	$2V_L(2,6),2V_H(2,6),2V_L(4,6),2V_H(4,6)$
32-35	1H(0,0),1H(0,1),1H(2,0),1H(2,1)	$2V_L(6,6),2V_H(6,6),1V_L(0,0),1V_H(0,0)$
36-39	1H(0,2),1H(0,3),1H(2,2),1H(2,3)	$1V_L(2,0),1V_H(2,0),1V_L(0,2),1V_H(0,2)$
40-43	0H(0,0), (außer Betrieb), OH(0,1), (außer Betrieb)	$1V_L(2,2)$, $2V_H(2,2)$, (außer Betrieb), $0V_L(0,0)$
44-47	(außer Betrieb)	$2V_H(0,0)$, (außer Betrieb), (außer Betrieb), (außer Betrieb)
48-51		(außer Betrieb)
52-55		
56-59		
60-63		

[0245] Die Tabelle 3 stellt den Fall mit vier Pegeln bzw. Niveaus dar. Da es nun 256 Zeiteinheiten pro Gruppe von Koeffizienten gibt, wird zur Vereinfachung nur der Pegel bzw. das Niveau und die Filterungsrichtung gezeigt.

Tabelle 3

Zeit	Filter 1	Filter 2
0-1	Pegel 3 horizontal	(außer Betrieb)
2-127	Pegel 3 horizontal	Pegel 3 vertikal
128-129	Pegel 2 horizontal	Pegel 3 vertikal
130-159	Pegel 2 horizontal	Pegel 2 vertikal
160-161	Pegel 1 horizontal	Pegel 2 vertikal
162-167	Pegel 1 horizontal	Pegel 1 vertikal
168	Pegel 0 horizontal	Pegel 1 vertikal
169	(außer Betrieb)	Pegel 1 vertikal
170	Pegel 0 horizontal	(außer Betrieb)
171	(außer Betrieb)	Pegel 0 vertikal
172	(außer Betrieb)	Pegel 0 vertikal
173-255	(außer Betrieb)	(außer Betrieb)

[0246] Der Ausgang des Filterungs- und Speicheruntersystems gemäß der vorliegenden Erfindung ist eine Reihe von Koeffizienten, die einer Bitsignifikanz-Einbettungskodierung gemäß der vorliegenden Erfindung unterzogen werden.

B. Das Kontextmodell für das System mit einem Durchlauf

[0247] In einer Ausführungsform der vorliegenden Erfindung wird mit dem Bitwertigkeits- bzw. -signifikanz-Einbettungskontextmodell für das System mit einem Durchgang jeder Baum in vier Teilen verarbeitet.

[0248] Die Wurzel des Baumes, der LL-Koeffizient ist dem höchsten Pegel, wird durch Horizontalordnungscodieren in einem Durchgang codiert. Die drei Unterbäume, die mit jeder der Wurzeln drei Nachkommen beginnen, die HH-, HL- und LH-Koeffizienten mit dem höchsten Pegel, werden sowohl mit der verbundenen Raum-/Frequenzmodelierung mit einem Durchgang als auch mit dem frequenzbasierenden Modelieren mit einem Durchgang verarbeitet. Die Koeffizienten werden so codiert, daß codierte Daten ausgegeben werden können, bevor das Bitsignifikanz-Einbettungskontextmodell mit sämtlichen Daten arbeitet.

Signifikanzbaum mit einem Durchgang

[0249] Das Nullbaum-Kontextmodell kann in dem System mit einem Durchlauf nicht verwendet werden. Der Nullbaum erfordert eine Liste (oder mehrere Listen), die jeden Koeffizienten enthalten, und Nullbäume führen mehrere Durchläufe durch die Liste(n) durch. Ein alternatives frequenzbasierendes Modell, ein Signifikanzbaum mit einem Durchlauf, erfordert keine Listen, die sämtliche Koeffizienten enthalten. Ein anderer Unterschied zwischen einem Signifikanzbaum mit einem Durchlauf und einem Nullbaum ist, daß der Signifikanzbaum sämtliche Nachkommen bearbeitet, bevor ihre Eltern bzw. ihre Stämme verarbeitet werden, wenn Entscheidungen erzeugt werden, im Gegensatz zu dem Nullbaum, der die Eltern bzw. den Stamm zuerst verar-

beitet.

[0250] Das Kontextmodell gemäß der vorliegenden Erfindung ist in der Form eines Blockdiagramms in Fig. 17 gezeigt. Das Kontextmodell **1700** enthält zwei Verarbeitungseinheiten, die Zeichen bzw. Vorzeichen-/Größenordnungseinheit **109** (Fig. 1A) und die Signifikanzeinheit **1702**. Das Kontextmodell **1700** verwendet auch zwei Speicher (mit Speichersteuerlogik), einen Größenordnungs-Speicher **1701** und einen Baumspeicher **1703**. Jede dieser zwei Speichereinheiten kann mit multiplen Speicherbereichen in die Tat umgesetzt werden, um eine abwechselnde Verwendung während des Hochgeschwindigkeitsbetriebes zu ermöglichen (d.h. während Daten in den einen geschrieben werden, wird der andere gelesen oder freigemacht).

[0251] Der Größen- bzw. Größenordnungs-Speicher **1701** ordnet die Koeffizienten in dem Baum neu in einer Ordnung, die auf Signifikanz bzw. Wichtung basiert, sowie eine Ordnung, die auf ihren Größen bzw. Größenordnungen basiert. Dies wird ausgeführt, indem eine Schlange von jeder möglichen Größenordnung aufrechterhalten wird. Die Signifikanzeinheit **1702** empfängt Koeffizienten in der Ordnung der Signifikanz (z.B. der Größe bzw. Größenordnung) und erzeugt Entscheidungen für einen Codierer der den A-Durchlaufalgorithmus verwendet bzw. handhabt. Der Baumspeicher **1703** ist an die Signifikanz- bzw. Wertigkeitseinheit **1702** angeschlossen und beseitigt Nullbäume nach sämtlichen Nullen.

[0252] Die nachfolgende Erörterung nimmt an, daß die Koeffizienten **18** Bits aufweisen und daß die Eingangsdaten einer Zerlegung mit vier Pegeln bzw. Niveaus unterzogen worden sind.

[0253] Eine Ausführungsform der Zeichen- bzw. Vorzeichen-/Größenordnungseinheit **109** ist in Fig. 18 dargestellt, und überträgt die Eingangskoeffizienten in ein Zeichen- bzw. Vorzeichen-/Größenordnungsformat. Die Zeichen- bzw. Vorzeichen-/Größenordnungseinheit **109** ist angeschlossen, um 18 Bits der Koeffizienten zu empfangen und enthält einen Inverter **1801**, einen Multiplexer (MUX) **1802**, einen Prioritätscodierer **1803** und eine Zähleinrichtung **1804**. Die Vorzeichen-/Größenordnungseinheit **109** gibt eine Wertigkeitsanzeige bzw. Signifikanzanzeige (z.B. einen 5-Bit-Wert), die Mantisse des Eingabekoeffizienten (z.B. 17 Bits), das Zeichen bzw. Vorzeichen des Eingangskoeffizienten mit einem Bit und einen Index von der Zähleinrichtung **1804** (z.B. 7 Bits).

[0254] Die MUX **1802** ist angeschlossen, um 17 Bits von den Koeffizienten, die unmittelbar in die Vorzeichen-/Größenordnungseinheit **109** eingegeben sind, und eine invertierte Version der 17 Bits von der Zweier-Komplementärwerteinrichtung **1801** zu empfangen. Basierend auf dem Zeichen- bzw. Vorzeichenbit (Koeffizientenbit 17), das an dem ausgewählten Eingang der MUX **1802** empfangen wird, gibt der positive der zwei seinen Ausgang als die Mantisse ein.

[0255] Die Vorzeichen-/Größenordnungseinheit **109** verwendet einen Prioritätscodierer **1803**, um das erte signifikante Bit bzw. Wertigkeitsbit von jedem Koeffizienten zu bestimmen. Auf der Grundlage des ersten signifkanten Bits von jedem Koeffizienten kann ein Signifikanz- bzw. Wertigkeitspegel mit dem Koeffizienten in Verbindung gebracht werden.

[0256] Die Zähleinrichtung **1804** wird verwendet, um einen Index mit dem gegenwärtigen Baumelement zu verbinden bzw. zu verknüpfen. Für eine Zerlegung mit vier Pegeln variiert der Index von 0 bis 84 (weil 1 + 4 + 16 + 64 = 85 die Anzahl der Elemente in einem Unterbaum ist). Die Eingangskoeffizienten sind in der Baumordnung, welche in diesem Beispiel so angenommen werden, daß die Eltern bzw. Stämme zuerst und die Kinder bzw. Nachkommen zuletzt sind. Die Koeffizienten sind für die geordneten Koeffizienten von unterschiedlichen Zerlegungspegeln bzw. -niveaus, wie in Tabelle 4 gezeigt wird.

Tabelle 4

Pegel	Index der Koeffizienten
0	0
1	1, 22, 43, 64
2	2, 7, 12, 17, 23, 28, 33, 38, 44, 49, 54, 59, 65, 70, 75, 80
3	3…6, 8…11, 13…16, 18…21, 24…27, 29…32, 34…37, 39…42, 45…48, 50…53, 55…58, 60…63, 66…69, 71…74, 76…79, 81…84

[0257] Die Fig. 19 ist eine Ausführungsform des Blockdiagramms des Größen- bzw. Größenordnungsspeichers **1701**. Eine Zähleinrichtung und ein Speicher sind miteinander für jeden möglichen Wertigkeits- bzw. Signifikanzpegel verbunden (ausgenommen nichts ist erforderlich für Null-Koeffizienten, die nicht codiert zu werden brauchen). Zum Beispiel sind die Zähleinrichtung **1916** und der Speicher **1936** mit dem Signifikanzpegel bzw. Wertigkeitspegel **17** verbunden. In einer Ausführungsform gibt es 16 Wertigkeits- bzw. Signifikanzpegel. Deshalb gibt es 17 Zähleinrichtungen und 17 angeschlossene Speicher.

[0258] In einer Ausführungform muß jeder Speicher **85** Stellen bzw. Orte für jeden möglichen Koeffizienten in einem Unterbaum aufweisen (da jeder Unterbaum **85** Koeffizienten enthält), jedoch kann die Speichergröße auf eine Potenz von 2 angemessen aufgerundet werden, wie etwa 128. Jeder Speichereingang kann ein Zeichen- bzw. Vorzeichenbit, einen 7-Bit-Index und N-Größenordnungsbits aufweisen, wobei N der Signifikanz- bzw. Wertigkeitspegel ist. Falls die Verwendung einer fixen Speicherbreite gewünscht wird, können Eingänge für die Signifikanz bzw. Wertigkeit 16 und 0, 15 und 1 usw. kombiniert werden, so daß jedes Wort zwei Eingänge hat, die zusammen 32 Bits ergeben. Natürlich muß bei einer ungeraden Anzahl von Signifikanz- bzw. Wertigkeitspegeln ein Wort nur einen Eingang enthalten, welches in diesem Beispiel der Pegel bzw. das Niveau 7 ist.

[0259] Zeichen- bzw. Vorzeichen-, Index- und Mantissenwerte, die von der Zeichen- bzw. Vorzeichen-/Größenordnungseinheit **109** empfangen worden sind, werden in den passenden bzw. angemessenen Speicher an der Adresse geschrieben, die durch die angeschlossene Zähleinrichtung des Speichers zur Verfügung gestellt wird. Die angeschlossene Zähleinrichtung wird dann inkrementiert bzw. erhöht, so daß der nächste Koeffizient bei diesem Signifikanz- bzw. Wertigkeitspegel an dem nächsten Ort bzw. der nächsten Stelle gespeichert werden kann.

[0260] Der Speicher wird von jedem der Speicher **1920**–**1926** in abnehmender Ordnung der Signifikanz bzw. Wertigkeit ausgelesen. Der Ausgang von jedem Koeffizienten enthält seinen Mantissen-, seinen Zeichen- bzw. Vorzeichen- und seinen Indexausgang. Wenn die Zähleinrichtung für den höchsten Pegel bzw. das höchste Niveau der Signifikanz bzw. der Wertigkeit nicht Null ist (z.B. Pegel 16), wird er verringert, und der Speicher wird bei dieser Adresse gelesen. Dies wird wiederholt, bis der Wert der Zähleinrichtung Null ist. Anschließend wird der nächste Wertigkeits- bzw. Signifikanzpegel (z.B. Pegel 15) in Betracht gezogen. Jeder Signifikanz- bzw. Wertigkeitspegel wird der Reihe nach in Betracht gezogen, bis sämtliche Zähler bzw. Zähleinrichtungen bis auf Null herabgesetzt worden sind und sämtliche Speicher geleert worden sind.

[0261] In einem Realzeitsystem kann es wünschenswert sein, zwei Bänke von Zähleinrichtungen und Speichern zu verwenden, so daß eine Bank für den Eingang verwendet wird, während die andere für den Ausgang verwendet wird.

[0262] Die Zähler adressieren ihre angeschlossenen Speicher so, daß ein LIFO (last in, first out, letzter rein, erster raus) in die Tat umgesetzt wird. Ein LIFO ist die zutreffende Ordnung, wenn Unterbäume in der Ordnung mit Stämmen bzw. Eltern zuerst eingegeben werden. Alternativ kann die Betätigung der Zäheinrichtungen, falls die Unterbäume zuerst mit den Nachkommen bzw. Kindern eingegeben werden, geändert werden, um ein FIFO (first in, first out, erster rein, erster raus) in die Tat umzusetzen.

[0263] Die Fig. 20 ist ein Blockdiagramm einer Ausführungsform einer Wertigkeits- bzw. Signifikanzeinheit **1702**. Bezugnehmend auf Fig. 20 wird die Index-Zähleinrichtung **2001** verwendet, um durch jeden Koeffizienten in einem Unterbau mit Nachkommen zuerst hindurchzuschreiten. In einer Ausführungsform wird die Index-Zähleinrichtung **2001** mit 84 initialisiert und zählt herab bis Null. Der Signifikanz- bzw. Wertigkeitszähler **2004** beginnt bei einem maximalen Wertigkeitspegel (in dem Beispiel beispielsweise 16) und zählt jedesmal abwärts, wenn die Index-Zähleinrichtung **84** einen Zyklus vervollständigt (zu 84 zurückkehrt), so daß die Wertigkeits- bzw. Signifikanzzähleinrichtung **2004** die Bitebene verfolgt bzw. auf dem laufenden hält. Der Pegel eines bestimmten Index wird durch eine Logik bestimmt (Index zu Pegel **2003**), die die in der obigen Tabelle 4 gezeigte Funktion durchführt.

[0264] Die Größenordnungsspeichereinheit **1701** stellt einen Index, eine Größenordnung und ein Zeichen bzw. Vorzeichen des nächsten Koeffizienten in dem Speicher zur Verfügung, der von dem Signifikanzzähler **2004** freigegeben wird. Wenn der von dem Speicher eingegebene Index der gleiche ist, wie der von der Index-Zähleinrichtung **2001** ausgegebene Index, gibt die Äquivalenzlogik-2002-Ausgabeanzeige nicht Null an. Die Ausgangsanzeige nicht Null bedeutet, daß der Größenordnungsspeicher den nächsten Index zur Verfügung stellen sollte usw. für den nächsten Zyklus. Wenn keine Übereinstimmung auftritt, dann wird eine Anzeige keine Übereinstimmung zu einem Diskussionsgenerator **2008** gesendet.

[0265] In einer Ausführungsform werden drei als Flag 0 (**2005**), Flag 1 (**2006**) und Flag 2 (**2007**) gezeigte Flipflops verwendet, um die Nicht-Null-Daten zu verfolgen bzw. auf dem laufenden zu halten und werden Zerlegungspegeln bzw. -niveaus 0, 1 bzw. 2 zugeordnet. Man beachte, daß die Anzahl der erforderlichen Flipflops um eins niedriger ist als die Anzahl der Zerlegungspegel bzw. -niveaus. Die Flipflops **2005** bis **2007** werden anfangs freigemacht bzw. gelöscht. Wenn das Nicht-Null-Signal von der Äquivalenzlogik **202** angegeben wird, dann werden sämtliche der Flipflops unter den Flipflops **2005** bis **2007** gesetzt, die einem Pegel zugeordnet sind, der niedriger ist als der gegenwärtige Pegel. Der Flipflop, der dem gegenwärtigen Pegel zugeordnet ist, wird freigemacht bzw. gelöscht. Der Pegel wird durch die Index-zu-Pegel-Logik **2003** zur Verfügung gestellt, die den Pegel in Antwort zu dem durch die Index-Zieleinrichtung **2001** zur Verfügung gestellten Index zur Verfügung stellt.

[0266] "Codierte" Flags werden gespeichert (in einigen Ausführungsformen ein Registerfeld), ein Bit für jeden Index. Wenn das Nicht-Null-Signal angegeben wird, wird das Bit, das mit dem gegenwärtigen Wert der Index-Zähleinrichtung in dem codierten Flagspeicher verknüpft ist, gesetzt. Andererseits wird, falls der Signifikanz- bzw. Wertigkeits-Zähleinrichtungswert der maximale Wert ist, das verküpfte Bit gelöscht bzw. freigegeben. Ansonsten bleibt der Wert des Bits ungeändert. Das bereits codierte Ausgangssignal von dem codierten Speicher der Flags ist das gleiche, wie der neue Wert des Bits, das mit dem gegenwärtigen Index verknüpft ist. Man bemerke, daß in einer alternativen Ausführungsform die codierten Flags nicht verwendet werden und das bereits codierte Signal niemals benutzt wird.

[0267] In einer Ausführungsform bestimmt der Entscheidungsgenerator **2008**, wann der gegenwärtige Pegel 3 ist und der vorherige Pegel es nicht war bzw. nicht war. In Antwort auf diese Bestimmung gibt der Entscheidungsgenerator **2008** den Start-Ausgang und der Startpegel-Ausgang ist der vorangehende Pegel. Wenn das Nicht-Null-Signal angegeben wird, gibt der Entscheidungsgenerator **2008** eine Entscheidung als "Signifikant" aus und gibt auch das Zeichen (00, 01) und die Mantisse aus. Andererseits wird, falls der bereits codierte Eingang angegeben wird, keine Entscheidung ausgegeben. Andererseits gibt, falls das Flag-Flipflop, das dem gegenwärtigen Pegel bzw. Niveau zugeordnet ist, gesetzt wird, der Entscheidungsgenerator **2008** die Entscheidung als "insignifkant mit signifikanten Nachkommen" (10) aus. Andererseits gibt der Entscheidungsgenerator **2008** die Entscheidung als "insignifikant und insignifikaten Nachkommen" (11) aus und gibt das ingesamt Null-Signal an.

[0268] Man bemerke, daß zur Umsetzung sowohl des frequenzbasierenden Modellierens als auch des horizontalen verbundenen Raum-/Frequenz-Modellierens mit einem Durchlauf die folgende Änderung in der Signifikanz- bzw. Wertigkeitseinheit **2000** gemacht wird. Die Signifikanz-Zähleinrichtung **2004** wird mit einer Schwelle verglichen und der insgesamt Null-Ausgang wird nur angegeben, wenn der Wert der Zählereinrichtung größer ist als die Schwelle.

[0269] In einer Ausführungsform ist die Signifikanz-Kategorie-Eingabe in den Baumspeicher **1703** (gezeigt in Fig. 21 und unten beschrieben) der Ausgang der Signifikanz-Zähl-einrichtung **2004**. In dieser Ausführungsform des Kontextmodells (z.B. Bit-Signifikanz-Einbettungseinheit) liegt die Anzahl der Bitebenen der Signifikanz-Kategorie zugrunde, und es gibt 17 verschiedene Signifikanz-Kategorien. Dies ist eine willkürliche bzw. beliebige Wahl. In einer anderen Ausführungsform können Bitebenen kombiniert werden, um weniger Signifi-

kanz-Kategorien zu erzeugen. Auch kann Pegelinformation zu Bitebeneninformation hinzuaddiert werden, um mehr Signifikanz-Kategorien zu erzeugen. Mehr Signifikanz-Kategorien könnten eine bessere verlustbehaftete Kompression zur Verfügung stellen, während weniger die Hardwarekomplexität verringern könnten.

[0270] Die Fig. 21 ist ein Blockdiagramm einer Ausführungsform der Baumspeichereinheit gemäß der vorliegenden Erfindung. Bezugnehmend auf Fig. 21 weist der Speicher **2101** angemessenen Platz zum Speichern einer Entscheidung und einer Signifikanzanzeige für jede mögliche Entscheidung auf. In einer Ausführungsform für eine Pegel-Zerlegung mit 17 Signifikanzpegeln ist die Anzahl der Orte bzw. Stellen im Speicher **2101** gleich zu 85 × 17 = 1445.

[0271] Um auf den Speicher **2101** zuzugreifen, werden Adressen erzeugt. Die Zählereinrichtung **2102** ist anfangs Null. Wenn der Entscheidungsgenerator **2008** nicht den Insgesamt-Null-Eingang angibt, wird der Wert in der Zählereinrichtung **2102** verwendet, um den Speicher zu adressieren. Wenn der Entscheidungsgenerator **2008** den Starteingang angibt, wird der gegenwärtige Wert der Zählereinrichtung **2102** in einem der Register **2110**–**2112** gemäß dem Startpegel gespeichert, der als Auswählmechanismus tätig ist. Die Zählereinrichtung **2102** wird dann heraufgesetzt.

[0272] Wenn der Entscheidungsgenerator **2008** den Insgesamt-Null-Eingang angibt, ist der Wert in dem Register (z.B. **2110**, **2111**, **2112**), der durch den Pegeleingang ausgewählt ist, verwendet, um den Speicher **2101** zu adressieren, und dieser Wert plus Eins wird in die Zählereinrichtung **2102** geladen. Dies bewirkt, daß die für die insignifikanten Nachkommen eines insignifikanten Elternteils bzw. Stammes verwendeten Speicherstellen bzw. -orte ignoriert werden.

[0273] Während der Speicherausgabe wird die Zählereinrichtung **2102** herabgesetzt, um die Adresse der Stelle, die ausgegeben werden soll, zur Verfügung zu stellen. Die Ausgabe (und Herabsetzung) wird angehalten, wenn die Zählereinrichtung **2102** Null erreicht. Der Ausgang von der Baumspeichereinheit **2100** wird durch einen Entropiecodierer empfangen, der die Entscheidung bei der spezifizierten Signifikanz bzw. Wertigkeit angemessen codiert.

[0274] Für einen Realzeitbetrieb können zwei Baumspeichereinheiten verwendet werden, so daß eine für die Eingabe verwendet wird, während die andere für die Ausgabe verwendet wird.

Ausrichtung bzw. Anordnung der Koeffizienten

[0275] In einer Ausführungsform gemäß der vorliegenden Erfindung verwendet das Nullbaum-Kontextmodell einen unnormierten $1 + Z^{-1}$ Tiefpaßfilter. Jedoch kann das Nullbaum-Kontextmodell mit normierten Filtern verwendet werden, wie etwa

$$\frac{1 + Z^{-1}}{\sqrt{2}}$$

[0276] Um normierte Filter zu verwenden, kann eine Ausrichtungs- bzw. Anordnungseinheit, wie etwa eine Anordnungseinheit **2200** gemäß Fig. 22, zwischen dem Vorwärts-Kleinwellenfilter **1000** und dem Kontextmodell **105** verwendet werden, um die Energie zu kompensieren, die von dem unnormierten Filter gewonnen (oder alternativ verloren) worden ist, was die Kompression verbessert. Da die Ausrichtung bzw. Anordnung eine nicht-einheitliche Quantisierung der Verlustoperation ermöglicht, kann die Ausrichtung bzw. Anordnung die sichtbare bzw. visuelle Qualität von verlustbehafteten Bildwiederherstellungen verstärken. In dem eindimensionalen Fall würden Koeffizienten von jedem Pegel bzw. Niveau des Baumes unterschiedliche Ausrichtungen bzw. Anordnungen aufweisen (Divisoren = $\sqrt{2}$, 2, $2\sqrt{2}$, 4, Multiplikatoren = $2\sqrt{2}$, 2, $\sqrt{2}$, 1). In dem zweidimensionalen Fall würden die Divisoren 2, 4, 8, 16 sein und die Multiplikatoren würden 8, 4, 2, 1 sein.

[0277] Da die Ausrichtung bzw. Anordnung gerade für das Gruppieren gleicher bzw. ähnlicher binärer Entscheidungen für die Codierung ist, ist die Verwendung der genauen Normierungswerte nicht kritisch. Die Ausrichtung bzw. Anordnung muß während der Decodierung umgekehrt bzw. invertiert werden, so daß sowohl Multiplikation als auch Division erforderlich sind. Die Verwendung von Faktoren/Divisoren, die eine Potenz von Zwei sind, würde es ermöglichen, effiziente Hardware-Verschiebungen stattdessen durchzuführen. Wenn Koeffizienten mit einer Potenz von Zwei multipliziert werden, brauchen die weniger signifikanten addierten bzw. hinzugefügten Null-Bits nicht codiert zu werden.

[0278] Jedoch kann anstelle einer Einschränkung der Anordnungs- bzw. Ausrichtungsfaktoren/-Divisoren auf eine Potenz von Zwei eine Annäherung wie $\sqrt{2} \approx 1{,}5$ oder $\sqrt{2} \approx 2 \div 1{,}5$ mit dem folgenden Verfahren verwendet

werden. Anstelle von Multiplikations-/Divisions-Koeffizienten durch die Faktoren/Divisoren würden nur die "Signifikant"-Koeffizienten durch den Faktor/Divisor skaliert werden. Die (Vor-)Zeichen-/Größenordnungseinheit kann, wie in Fig. 23 gezeigt, modifiziert werden, um einen "1,5"-Prioritätscodierer **2301** zu enthalten, der die Position entweder (1) des signifikantesten "1"-Bits, falls das nächst-signifikanteste Bit ebenfalls "1" ist, oder andererseits (2) dasjenige, das geringer ist als die Stelle bzw. Position des signifikantesten "1"-Bits zurückgibt. Eine Wahrheitstabelle für einen "1,5"-Prioritätscodierer für 3 Eingangsbits ist in der Tabelle 5 gezeigt.

Tabelle 5

Eingabe (binär)	Ausgabe
001	0
010	0
011	1
100	1
101	1
110	2
111	2

[0279] Abhängig von dem Niveau des Koeffizienten, der durch den gegenwärtigen Indexwert angezeigt wird, wählt ein Multiplexer **2302** die Signifikanz bzw. Wertigkeit entweder des Standard-Prioritätscodierers oder des "1,5"-Prioritätscodierers. Immer wenn die "1,5"-Ausrichtung bzw. -Anordnung verwendet wird, enthält die Mantisse N + 1 Bits, wobei N der signifikante Wert ist. Ansonsten enthält die Mantisse N Bits.

[0280] Eine Ausrichtungs- bzw. Anordnungseinheit **2200**, die einen Multiplexer mit zwei Eingängen aufweist, der als eine Verschiebungseinrichtung in die Tat umgesetzt werden kann und richtet aus bzw. ordnet an um 1 oder 2. Wird dies mit der 1,5-Anordnung bzw. -ausrichtung, die durch die (Vor-)Zeichen-/Größenordnungseinheit zur Verfügung gestellt wird, kombiniert, so ermöglicht dies die Ausrichtung bzw. Anordnung von 1, 1,5, 2 oder 3, was eine gute Annäherung der gewünschten Multiplizierer für eindimensionale Signale ist, da die Zahlen einfacher sind (z.B. Potenzen von Zwei). (Für zweidimensionale Signale, wie Bilder, sind die Zahlen einfacher.) Während der Decodierung ist das N + 2-te Bit der Mantisse (welches nicht codiert ist, das Komplement des n + 1-ten Bits, wenn der "1,5"-Prioritätscodierer verwendet wird.

[0281] Die Koeffizientenanordnung bzw. -ausrichtung kann zum Abstimmen des Nullbaums und zur genaueren und nicht gleichmäßigen Quantisierung verwendet werden. Im Fall von Bildern (zweidimensionalen Signalen) richtet eine Ausführungsform der RTS-Transformation die Koeffizienten durch Multiplizieren des Frequenzbandes mit den in Fig. 31 dargestellten Zahlen aus bzw. ordnet sie an. Das Multiplizieren dieser Zahlen resultiert in die RTS-Transformation, die eine sehr gute Annäherung an die genauen wiederhergestellten Kleinwellen der TS-Transformation ist.

[0282] Der Entropiecodierer muß das Anordnungs- bzw. Ausrichtungsverfahren in Betracht ziehen, um effizient zu sein.

Frequenzbasierendes Kontextmodell durch teilweise bzw. bruchstückhafte Bitebenen

[0283] Ein alternatives Verfahren des frequenzbasierenden Modellierens verwendet teilweise bzw. bruchstückhafte Bitebenen oder teilweise bzw. bruchstückhafte Signifikanz- bzw. Wertigkeitsbits. Eine Umsetzung von diesen ist die zweifache Verarbeitung jeder Bitebene, so daß die Durchläufe einen A1-Durchlauf, einen B1-Durchlauf, einen A0-Durchlauf und einen B0-Durchlauf enthalten. Man bemerke, daß die Namen der

Durchläufe ausgewählt wurden, weil der A1-Durchlauf Koeffizienten behandelt bzw. handhabt, die mit "11" beginnen und der A0-Durchlauf jene handhabt bzw. behandelt, die mit "10" beginnen.

[0284] Während des A1-Durchlaufs für eine Bitebene S ist ein Koeffizient in der A-Gruppe nur signifikant, wenn sowohl Bits S als auch S-1 nicht Null sind. Während des A2-Durchlaufs ist ein Koeffizient in der A-Gruppe signifikant, falls das Bit S nicht Null ist. Da die zwei signifikantesten Bits bekannt sind, brauchen der B1-Durchlauf und der B0-Durchlauf nur S-1 Bits zu verarbeiten (unter der Annahme, daß S=0 die am wenigsten signifikante Bitebene ist).

[0285] Da abwechselnde bruchstückhafte bzw. teilweise Bitebenen sich voneinander um einen Faktor 1,5 oder 2/1,5 unterscheiden, kann die Anordnung für unterschiedliche Pegel durch Gruppieren der gewünschten teilweisen Bitebenen für jeden Pegel erzielt werden.

[0286] Die teilweisen bzw. bruchstückhaften Bitebenen verursachen eine genauere Modellierung der Daten durch das Eltern- bzw. Stamm-/Nachkommen-Verhältnis, das durch das frequenzbasierende Kontextmodell verwendet wird. Mehr als zwei Durchläufe, beispielsweise vier oder acht Durchläufe könnten für eine noch genauere Modellierung verwendet werden. Zum Beispiel würde in dem Fall mit vier Durchläufen der A11-Durchlauf Koeffizienten behandeln bzw. handhaben, die mit "111" beginnen. Die anderen Durchläufe würden "110", "101" und "100" handhaben. Eine weniger genaue Modellierung könnte ebenfalls verwendet werden. Zum Beispiel könnte ein Durchlauf nur für jede andere Bitebene vorgenommen werden. In dem Fall der weniger genauen Modellierung werden mehr Bits durch die B-Gruppe codiert.

C. Codierer- und Speicher-/Kanalmanagement für ein System mit einem Durchlauf bzw. Durchgang

[0287] Das Speichermanagement für codierte Daten wird in dem System mit einem Durchlauf für Systeme, die sämtliche der Daten im Speicher abspeichern, und für Systeme vorgestellt, die die Daten in einem Kanal übertragen. In dem System mit einem Durchlauf müssen codierte Daten so gespeichert werden, daß sie in der eingebetteten Kausalart bzw. -mode zugegriffen werden können, so daß weniger signifikante bzw. wertige Daten mißachtet werden können, ohne signifikantere Daten zu verlieren. Da codierte Daten eine variable Länge aufweisen, kann eine Zuweisung dynamischen Speichers verwendet werden.

[0288] Gemäß einer Ausführungsform nach der vorliegenden Erfindung verwendet das eingebettete Codierungsschema 18 Bitebenen und überträgt 18 Signifikanz- bzw. Wertigkeitspegel auf die Daten. Der Codierer muß in einem System mit einem Durchlauf "kausal einbettend" sein. Die Decodierungsereignisse, die einer Bitebene entsprechen, erfordern keine Informationen von Bitebenen geringerer Ordnung. In dem Fall mit einem Durchlauf werden üblicherweise sämtliche der Bits eines Baumes codiert, bevor irgendwelche der Bits des nächsten Baumes codiert werden, so daß Bits unterschiedlicher Signifikanz bzw. Wertigkeit nicht getrennt werden. Für Codierer, die keinen inneren Zustand bzw. internen Zustand verwenden, wie Huffman-Codierer, ist dies kein Problem. Jedoch verwenden viele kompliziertere Komprimierer mit besserer Komprimierung interne bzw. innere Zustände.

[0289] Ein Weg, dieses Problem für diese Codierer zu lösen, ist es, 18 verschiedene Codierer, vielleicht 18 Q-Codiererchips, zu verwenden. Eine Technik, die die Verwendung von 9 Q-Codiererchips ermöglicht, ist in dem US-Patent Nr. 5,097,261 (Langdon, Jr.) beschrieben, betitelt mit "Data Compression for Recording on a Record Medium", erteilt am 17. März 1992. Eine bessere Art verwendet einen in einer Leitung bzw. in einer Reihe angeordneten Codierer, um verschiedene virtuelle Codes mit einem einzigen physikalischen Codierer in die Tat umzusetzen, so wie es in der US-Patentanmeldung mit der Serial No. 08/016,035, betitelt mit "Method and Apparatus for Parallel Decoding and Encoding of Data", eingereicht am 10. Februar 1993, beschrieben wird. In einem solchen Codierer sind die mehrfachen bzw. multiplen Bitgeneratorzustände für jede Wahrscheinlichkeit jeweils einem Teil der Daten zugeordnet. Zum Beispiel könnte jeder von 18 Zuständen einer bestimmten Bitebene für 18 Bitdaten zugeordnet werden. Register in der Verschiebungseinrichtung in dem Codierer sind ebenfalls jedem Teil der Daten zugeordnet. In dem Codierer wird keine Verschachtelung durchgeführt; jede Art von Daten wird einfach bitgestapelt bzw. nach Bits abgelegt.

[0290] In Ausführungsformen entweder mit mehrfachen bzw. multiplen physikalischen Codierern oder virtuellen Codierern wird Speicher jedem Teil der Daten zugeordnet. Wenn die Komprimierung vollständig ist, ist eine verknüpfte Liste, die den zugeordneten Speicher plus den Inhalt des zugeordneten Speichers beschreibt, das Ergebnis.

[0291] Falls der Speicher übergeht, bewirkt die Speicherzuordnungswegbestimmung, daß wichtigere Daten

die weniger wichtigen Daten überschreiben. Zum Beispiel kann das am wenigsten signifikante Bit von numerischen Daten zuerst überschrieben werden. Die Information, die beschreibt, wie Speicher zugeordnet wird, muß zusätzlich zu den codierten Daten gespeichert werden.

[0292] Die Fig. 24 zeigt ein Beispiel einer dynamischen Speicherzuordnungseinheit für drei Kategorien der Signifikanz bzw. Wertigkeit. Nur drei Kategorien werden beschrieben, um eine Verundeutlichung der vorliegenden Erfindung zu vermeiden; typischerweise würde eine große Anzahl von Kategorien, wie etwa 8, 16 oder 18 verwendet werden. Ein Registerfeld (oder anderer Speicher) hält einen Zeiger für jede Signifikanz bzw. Wertigkeitskategorie und einen anderen Zeiger, um den nächsten freien Speicherplatz anzuzeigen. Der Speicher wird in Seiten fester Größe unterteilt.

[0293] Eingangs zeigt jeder Zeiger, der einer Signifikanz-Kategorie zugeordnet ist, auf den Beginn einer Seite des Speichers und der freie Zeiger deutet auf die nächste verfügbare Seite des Speichers. Die codierten Daten, die mit einer Signifikanz-Kategorie identifiziert werden, werden an dem Speicherort gespeichert, der durch den entsprechenden Zeiger adressiert ist. Der Zeiger wird dann auf den nächsten Speicherplatz bzw. -ort erhöht bzw. inkrementiert.

[0294] Wenn der Zeiger das Maximum für die gegenwärtige Seite erreicht, wird die Adresse des Starts der nächsten freien Seite, die in dem freien Zeiger gespeichert ist, mit der gegenwärtigen Seite als Verbindung bzw. Verknüpfung gespeichert. In einer Ausführungsform kann der Teil des codierten Datenspeichers oder eines getrennten Speichers oder Registerfeldes für diesen Zweck verwendet werden. Anschließend wird der gegenwärtige Zeiger auf die nächste freie Seite gesetzt. Der freie Zeiger wird erhöht. Diese Schritte bewirken, daß eine neue Seite des Speichers einer bestimmten Signifikanz- bzw. Wertigkeits-Kategorie zugeordnet wird und stellt Verbindungen bzw. Verknüpfungen zu Seiten des Speichers zur Verfügung, die Daten für eine allgemeine Signifikanz-Kategorie enthalten, so daß die Ordnung der Zuteilung während des Decodierens bestimmt werden kann.

[0295] Wenn sämtliche Seiten in dem Speicher in Benutzung sind und mehr Daten vorhanden sind, die signifikanter sind als die wenig signifikanten Daten im Speicher, kann eine Neuzuordnung durchgeführt werden. Drei derartige Neuzuordnungstechniken werden beschrieben. In sämtlichen drei Fällen wird Speicher, der den weniger bzw. am wenigsten signifkanten Daten zugeordnet ist, signifikanteren Daten neu zugeordnet und keine weniger bzw. am wenigsten signifikanten Daten werden länger gespeichert.

[0296] Zuerst wird die erste gegenwärtig durch die am wenigsten signifikanten Daten verwendete Seite einfach den mehr signifikanten Daten zugeordnet. Da die meisten typischen Entropiecodierer interne Zustandsinformationen verwenden, wird sämtliche der weniger bzw. am wenigsten signifikanten Daten, die zuvor in dieser Seite gespeichert wurden, verloren.

[0297] Zweitens wird die gegenwärtig durch die am wenigsten signifikanten Daten verwendete Seite den signifikanteren Daten zugeordnet. Anders als bei dem vorherigen Fall wird der Zeiger auf das Ende der Seite gesetzt und signifikantere Daten werden in die Seite geschrieben, wobei der entsprechende Zeiger erniedrigt bzw. dekrementiert wird. Dies hat den Vorteil, daß die am wenigsten signifikanten Daten bei dem Beginn von der Seite bewahrt werden, wenn die signifikanteren Daten nicht die gesamte Seite erfordern.

[0298] Drittens kann anstelle der gegenwärtigen Seite der am wenigsten signifikanten Daten, die neu zugeordnet werden, irgendeine Seite mit am wenigsten signifikanten Daten neu zugeordnet werden. Dies erfordert, daß die codierten Daten für sämtliche Seiten unabhängig codiert werden, was die erzielte Komprimierung verringern kann. Es erfordert auch, daß die uncodierten Daten, die dem Beginn sämtlicher Seiten entsprechen, erkannt bzw. identifiziert werden. Da eine beliebige Seite am wenigsten signifikanter Daten außer Acht gelassen werden kann, ist eine größere Flexibilität bei der Quantisierung verfügbar.

[0299] Die dritte Alternative kann besonders vorteilhaft in einem System sein, das eine feste Komprimierungsrate über Bereiche des Bildes erzielt. Eine spezifizierte Anzahl von Speicherseiten kann einem Bereich des Bildes zugewiesen werden. Ob weniger signifikante Daten zurückbehalten werden oder nicht, kann von der in einem bestimmten Bereich erzielten Komprimierung abhängen. Man beachte, daß der einem Bereich zugeordnete Speicher nicht vollständig in Benutzung genommen sein kann, falls verlustlose Komprimierung weniger als die zugeordnete Speichermenge erfordert. Die Erzielung einer festen Kompressionsrate für einen Bereich des Bildes kann einen wahlfreien Zugriff zu den Bildbereichen unterstützen.

[0300] Wenn die Komprimierung vollständig ist, können die Daten übertragen werden, falls gewünscht zu ei-

nem Kanal oder einer Speichereinrichtung, in der Ordnung der Signifikanz bzw. Wertigkeit. Die verschiedenen Verknüpfungen und Zeiger würden dann nicht mehr benötigt werden und eine Decodierung mit mehrfachen Durchläufen könnte durchgeführt werden. Alternativ können die Zeiger zu den Daten für jede Signifikanz für eine Decodierung mit einem Durchlauf aufrechterhalten werden.

[0301] In einigen Anwendungen können einige Signifikanz- bzw. Wertigkeits-Kategorien nicht mehr verwendet werden. Zum Beispiel könnte ein 16-Bit-Komprimierer für ein 12-Bit-Medizinalbild verwendet werden, so daß die Signifikanz-Kategorien, die zu Bitebenen 15...12 korrespondieren, ungenutzt sein würden. Bei Umsetzungen mit großen Seiten und vielen ungenutzten Signifikanz-Kategorien würde dies Speicher verschwenden (falls das System nicht im voraus wüßte, daß einige Kategorien unbenutzt sind), da der Speicher diesen nicht zugeordnet werden muß. Eine andere Lösung für diese Speicherverschwendung wäre es, einen kleinen Speicher (oder Register) zu verwenden, um eine Zahl bzw. Zählung für jede Signifikanz-Kategorie zu halten. Die Zahl bzw. Zählung würde die Anzahl von Entscheidungen "insignifikant, nicht signifikante Nachkommen" verfolgen bzw. auf dem laufenden halten, die auftreten, bevor irgendwelche anderen Entscheidungen vorkommen. Der Speicher, der zum Speichern dieser Zählungen bzw. dieser Zähler erforderlich ist, muß gegenüber dem durch unbenutzte Signifikanz-Kategorien verwendeten Speicher aufgerechnet ("traded off") werden.

[0302] Die Fähigkeit, Daten in jede Seite von beiden Enden einzuschreiben, kann verwendet werden, den insgesamt in dem System verfügbaren Speicherplatz besser auszunutzen. Wenn sämtliche Seiten zugewiesen sind, kann jede Seite, die ausreichend freien Platz an dem Ende zur Verfügung hat, für die Verwendung von dem Ende zugewiesen werden. Die Fähigkeit, beide Enden einer Seite zu verwenden, muß gegenüber den Kosten für die Verfolgung des Platzes, wo die zwei Arten von Daten aufeinandertreffen, abgewogen werden. Man bemerke, daß dies von dem Fall unterschiedlich ist, wo einer der Datentypen nicht signifikant war und einfach überschrieben werden konnte.

Verwendung eines Kanals

[0303] In einem System, in dem Daten in einem Kanal übertragen werden, anstelle in einem Speicher gespeichert zu werden, und Seiten fester Größe des Speichers verwendet werden kann (aber nur eine Seite pro Signifikanz-Kategorie erforderlich ist), wenn eine Seite des Speichers voll ist, diese in dem Kanal übertragen werden und der Speicherplatz kann, sobald die Daten übertragen sind, wieder verwendet werden. Bei einigen Anwendungen kann die Seitengröße des Speichers die Größe des Datenpaketes sein, die in dem Kanal verwendet wird, oder ein Mehrfaches der Paketgröße. (Man bemerke, daß in einer Ausführungsform zwei Seiten pro Signifikanzpegel verwendet werden können, so daß Daten in eine geschrieben werden können, während die andere zur Ausgabe zu dem Kanal gelesen wird.)

[0304] In einigen Kommunikationssystemen, z.B. ATM (asynchroner Übertragungsmodus) (Asynchrounous Transfer Mode) können Prioritäten den Paketen zugeordnet werden. ATM weist zwei Prioritätspegel, eine Primäre und eine Sekundäre, auf. Sekundäre Pakete werden nur übertragen, wenn eine hinreichende Bandbreite zur Verfügung stellt. Eine Schwelle kann verwendet werden, um zu bestimmen, welche Signifikanz-Kategorien primär sind und welche sekundär sind. Ein anderes Verfahren wäre die Verwendung einer Schwelle bei dem Codierer, um keine Signifikanz-Kategorien zu übertragen, die weniger signifikant waren als die Schwelle.

Verlustbehaftete Komprimierung mit begrenztem Spitzenfehler

[0305] In einigen Anwendungen ist keine perfekte (verlustfreie) Wiederherstellung erforderlich. Es kann wünschenswert sein, eine Komprimierung mit einem bestimmten maximalen Spitzenfehler zu erzielen. Der Spitzenfehler sei ±E. Dies kann erzielt werden, indem die komprimierten Daten abgeschnitten werden, so daß sämtliche weniger signifikanten Daten, die nicht benötigt werden, um die gewünschte Auflösung bzw. Genauigkeit zu erzielen, außer Acht gelassen werden.

[0306] Ein anderer Weg zur Erzielung der Komprimierung mit einem bestimmten maximalen Spitzenfehler ist es, durch einen Wert zu teilen (mit ganzzahliger Division), der geringer ist als oder gleich ist zu 2 × E + 1 für jedes Bildelement des zu komprimierenden Bildes. Während der Wiederherstellung wird jedes Bildelement in dem Bild verarbeitet mit:

Ausgabe-Bildelement = (2 × E + 1) × Eingabe-Bildelement + E.

[0307] (Alternativ kann anstelle des Addierens von E während der Dekomprimierung eine Subtraktion während der Komprimierung vor der Division durch 2 × E + 1 auftreten.)

[0308] Ein anderer Weg, um die Komprimierung mit einem spezifizierten maximalen Spitzenfehler zu erzielen, ist es, die Division und die Multiplikation durch Verschiebungen zu ersetzen. Der Verschiebungsbetrag ist $\lfloor \log2(2 \times E + 1) \rfloor$. Weil die Verschiebung angenehm bzw. angemessen ist, kann eine bessere Fehlerspezifikation (die den Spitzenfehler ersetzt) Fehler der Form (2^n < Fehler < -2^n] sein.

[0309] Das Vorhergehende sollte nicht mit der Quanitisierung von Koeffizienten durcheinandergebracht werden, die im Stand der Technik von verlustbehafteten Bildkomprimierungen wohlbekannt ist. In vielen verlustbehafteten Komprimierungssystemen (z.B. JPEG) werden Transformationsbereichskoeffizienten einem maximalen Spitzenfehler zugeordnet, welcher den Spitzenfehler des Bildes nur indirekt steuert. Ein kritischer Unterschied ist, daß die vorliegende Erfindung die Quantisierung auf Bildelemente durchführt und verlustlose Komprimierung von Koeffizienten verwendet.

[0310] Die Transformationsbereichsquantisierung kann ebenfalls verwendet werden. Viele Koeffizienten haben eine Wirkung auf den Spitzenfehler, der sich durch mehrere Pegel bzw. Niveaus der Transformation hindurchzieht. Es ist leichter, den Effekt auf den Spitzenfehler für die Hochpaßkoeffizienten zu bestimmen, die keine Nachkommen haben.

[0311] Man ziehe ein eindimensionales Bild in Betracht, das mit einem maximalen Spitzenfehler von ±E zu codieren ist. Dies kann durch Quantisieren der Hochpaßkoeffizienten der feinsten Einzelheit zu ±2E erzielt werden. Für ein zweidimensionales Signal kann, da es zwei Anwendungen des Hochpaßfilters gibt, die Koeffizienten HH der feinsten Einzelheit zu ±4E quantisiert werden.

[0312] Eine Alternative, die Quantisierung des Eingabebildes zu verwenden, ist, die Entscheidungen zu dem Entropiecodierer zu steuern. Ein Beispiel lautet wie folgt. Für jeden Koeffizienten wird, falls die Einstellung des Koeffizienten auf Null nicht den Fehler in irgendeinem Bildelement durch den Koeffizienten beeinträchtigen würde, um den maximalen Fehler zu überschreiten, der Koeffizient auf Null gesetzt werden. In einigen Umsetzungen werden nur bestimmte Koeffizienten geprüft, vielleicht nur die AC-Koeffizienten, die keine Nachkommen haben. Die Koeffizienten können mit einer gierigen Strategie in Betracht gezogen werden, bei der einer zu einer Zeit bzw. einem Zeitpunkt in Betracht gezogen wird. Andere Schritte dagegen können kleinere Gruppen von Koeffizienten in Betracht ziehen und den größten möglichen Untersatz der Gruppe zu Null wählen.

[0313] Während viele Änderungen und Modifikationen der vorliegenden Erfindung möglich sind, wird bei einem Fachmann aus dem Stand der Technik kein Zweifel darüber auftreten, nachdem er die vorangehende Beschreibung gelesen hat, daß die bestimmte Ausführungsform, die durch Darstellungen gezeigt und beschrieben worden ist, in keiner Weise als einschränkend aufzufassen ist. Deshalb sind Bezugnahmen auf Einzelheiten der bevorzugten Ausführungsform nicht als dem Schutzbereich der Ansprüche einschränkend beabsichtigt, welche nur die Merkmale wiedergeben, die als wesentlich für die Erfindung angesehen werden.

[0314] Ein Verfahren und eine Vorrichtung zum Codieren und Decodieren von Daten werden beschrieben. Die vorliegende Erfindung enthält ein Verfahren und eine Vorrichtung zum Erzeugen transformierter Signale in Antwort auf Eingabedaten. In einer Ausführungsform werden die transformierten Signale unter Verwendung einer reversiblen Wellen- bzw. Kleinwellen-Transformation erzeugt. Die vorliegende Erfindung enthält auch ein Verfahren und eine Vorrichtung zum Komprimieren der transformierten Signale in Daten, die eine verlustlos komprimierte Version der Eingabedaten darstellen. In einer Ausführungsform zerlegt die vorliegende Erfindung die Eingabedaten unter Verwendung eines reversiblen Filters nichtminimaler Länge. Die Zerlegung kann unter Verwendung mehrerer bzw. mehrfacher eindimensionaler Filter durchgeführt werden.

Beispiel-Code für eine RTS-(Vorwärts-)Transformation – einpegelige Dekompression –

```
void RTS(T,w,m,n)
int w,m,n;
int *T;
{
  int i,j,k;
  int *X, *Y;

  for (i=0;i<n;i++)
  {
    for (j=0;j<m;j+=2)
    {
      X[j/2] = (T[i*w+j]+T[i*w+j+1])>>1;
      Y[j/2] = T[i*w+j]-T[i*w+j+1];
    }
    for (j=0;j<m/2;j++)
    {
      T[i*m+j]=X[j];
      T[i*m+m/2+j]= (-X[j]+(Y[(j+1)%(m/2)]<<2)+X[(j+2)%(m/2)])>>2;
    }
  }
```

Beispiel-Code für (inverse) RTS-Transformation – einpegelige Dekompression –

```
void InverseRTS(T,w,m,n)
int w,m,n;
int *T;
{
  int i,j;
  int  *B, *D;

  for (i=0;i<n;i++)
  {
    for (j=0;j<m/2;j++)
    {
      B[j] = (-T[i*m+j]+T[i*m+((j+2)%(m/2))])&0X3;
      D[(j+1)%(m/2)] = ((T[i*m+j]-T[i*m+((j+2)%(m/2))]+
                        (T[i*m+m/2+j]<<2)+B[j])>>2);
    }
    for (j=0;j<m/2;j++)
    {
        T[i*w+2*j] = ((T[i*m+j]<<1)+D[j]+1)>>1;
        T[i*w+2*j+1] = T[i*w+2*j] - D[j];
    }
  }
}
```

Zweipegelige Dekompression-RTS-(Vorwärts-)Transformation

```
void RTS(T,w,m,n)
int w,m,n;
int *T;
{
  int i,j,k;
  int *temp;
  int *X, *Y;

  for (i=0;i<n;i++)
  {
    for (j=0;j<m;j+=2)
    {
      X[j/2] = (T[i*w+j]+T[i*w+j+1])>>1;
      Y[j/2] = T[i*w+j]-T[i*w+j+1];
    }
    for (j=0;j<m/2;j++)
    {
      temp[i*m+j]=X[j];
      temp[i*m+m/2+j]= (-X[j]+(Y[(j+1)%(m/2)]<<2)+X[(j+2)%(m/2)])>>2;
    }
  }

  for (j=0;j<m;j++)
  {
    for (i=0;i<n;i+=2)
    {
      X[i/2] = (temp[i*m+j]+temp[(i+1)*m+j])>>1;
      Y[i/2] = temp[i*m+j]-temp[(i+1)*m+j];
    }
    for (i=0;i<n/2;i++)
    {
      T[i*w+j]=X[i];
      T[(i+n/2)*w+j]= (-X[i]+(Y[(i+1)%(n/2)]<<2)+X[(i+2)%(n/2)])>>2;
    }
  }
}
```

Zweipegelige Dekompression (inverse) RTS-Transformation

```
void InverseRTS(T,w,m,n)
int w,m,n;
int *T;
{
  int i,j;
  int  *B, *D;
  int *temp;

  for (j=0;j<m;j++)
  {
    for (i=0;i<n/2;i++)
    {
      B[i] = (-T[i*w+j]+T[((i+2)%(n/2))*w+j])&0X3;
      D[(i+1)%(n/2)] = ((T[i*w+j]-T[((i+2)%(n/2))*w+j]+(T[(n/2+i)*w+j]<<2)+B[i])>>2);
    }
    for (i=0;i<n/2;i++)
    {
        temp[2*i*m+j] = ((T[i*w+j]<<1)+D[i]+1)>>1;
        temp[(2*i+1)*m+j] = temp[2*i*m+j] - D[i];
    }
  }

  for (i=0;i<n;i++)
  {
    for (j=0;j<m/2;j++)
    {
      B[j] = (-temp[i*m+j]+temp[i*m+((j+2)%(m/2))])&0X3;
      D[(j+1)%(m/2)] = ((temp[i*m+j]-temp[i*m+((j+2)%(m/2))]+
                        (temp[i*m+m/2+j]<<2)+B[j])>>2);


    }
    for (j=0;j<m/2;j++)
    {
        T[i*w+2*j] = ((temp[i*m+j]<<1)+D[j]+1)>>1;
        T[i*w+2*j+1] = T[i*w+2*j] - D[j];
    }
  }
}
```

Patentansprüche

1. Codierer zum Codieren von Eingabedaten zu einem komprimierten Datenstrom, wobei der Codierer aufweist:
einen reversiblen bzw. umkehrbaren Kleinwellenfilter zum Transformieren der Eingabedaten in eine Mehrzahl von Koeffizienten;
einen Einbettungscodierer, der an die reversiblen Kleinwellenfilter bzw. Filter angeschlossen ist, um Einbettungscodieren an der Mehrzahl der Koeffizienten durchzuführen, so daß ein Bitstrom erzeugt wird; wobei das Einbettungscodieren auf die Anordnung der Koeffizienten in einer Reihenfolge bzw. Ordnung gerichtet ist, und
einen Entropiecodierer, der an dem Einbettungscodierer angeschlossen ist bzw. angekoppelt ist, um Entropiecodierung für bzw. an dem Bitstrom durchzuführen, um codierte Daten zu erzeugen.

2. Codierer zum Codieren von Eingabedaten, der aufweist:
einen Transformierungscodierer, der angekoppelt bzw. angeschlossen ist, um die Eingabedaten zu empfangen, und eine Reihe bzw. Serie von Koeffizienten zu erzeugen, wobei die Reihe bzw. Serie von Koeffizienten eine Zerlegung der Eingabedaten darstellt; und
einen Einbettungscodierer, der angekoppelt bzw. angeschlossen ist, um die Serie bzw. Reihe von Koeffizienten zu empfangen und eine Bit-Signifikanz- bzw. -Wertigkeits-Codierung auf die Reihe von Koeffizienten anzuwenden, um codierte Daten zu erzeugen, wobei der Einbettungscodierer codierte Daten vor dem Empfang sämtlicher der Reihe bzw. Serie von Koeffizienten erzeugt, und bei der Bit-Signifikanz-Codierung die Koeffizienten binär basierend auf der Signifikanz bzw. Wertigkeit des Bits ausgedrückt werden.

3. Codierer nach Anspruch 2, in dem der Transformationscodierer und der Einbettungscodierer betrieben werden, um codierte Daten von den Eingangsdaten in einem Durchlauf zu erzeugen.

4. Wavelet-Transformationsfilter zur Transformation von Eingabedatensignalen, das aufweist:
einen ersten Addierer zum Addieren eines ersten Proben- bzw. Signalpaares der Eingabedatensignale, um die

Summe der Addition als ein erstes Ergebnis zu erzeugen und auszugeben;
eine erste Ausgangslogik, in die das erste Ergebnis eingeht und die basierend darauf einen ersten Tiefpaßkoeffizienten ausgibt;
einen ersten Subtrahierer, der die Differenz zweier Eingaben, von denen eine erste der erste Tiefpaßkoeffizient ist, erzeugt und ausgibt und dessen Ausgabe ein zweites Ergebnis darstellt, wobei die Ausgabe als zweite Eingabe zu dem ersten Subtrahierer zurückgeführt wird und mittels dem ersten Subtrahierer von der ersten Eingabe abgezogen wird;
einen zweiten Subtrahierer, der eine Differenz aus einem zweiten Paar von Proben bzw. Abtastungssignalen des Eingangsdatensignals als ein drittes Ergebnis erzeugt und ausgibt;
einen zweiten Addierer, der das dritte Ergebnis und das zweite Ergebnis addiert, um so die Summe der Addition als ein viertes Ergebnis zu erzeugen und auszugeben;
eine zweite Ausgangslogik, in die das vierte Ergebnis eingeht und die basierend darauf einen zweiten Tiefpaßkoeffizienten erzeugt und ausgibt, so daß das Wavelet- bzw. Kleinwellen-Transformationsfilter den ersten und den zweiten Tiefpaßkoeffizienten ausgibt.

5. Filter nach Anspruch 4, in dem die erste Ausgangslogik bzw. Ausgabelogik eine Dividiereinrichtung aufweist.

6. Filter nach Anspruch 5, in dem die Divisionseinrichtung eine Bit-Verschiebeeinrichtung aufweist.

7. Filter nach Anspruch 4, in dem die zweite Ausgangs- bzw. Ausgabelogik eine Divisionseinrichtung aufweist.

8. Filter nach Anspruch 7, in dem die Divisionseinrichtung eine Bit-Verschiebungseinrichtung aufweist.

9. Wavelet-Transformationsfilter zum Transformieren von Eingabedatensignalen, das aufweist:
einen ersten Addierer zum Addieren eines ersten Proben- bzw. Abtastsignalpaares der Eingabedatensignale, um die Summe der Addition als ein erstes Ergebnis zu erzeugen und auszugeben;
einen ersten Multiplizierer, in dem das erste Ergebnis eingeht und der das erste Ergebnis mit einem ersten Faktor vervielfacht bzw. multipliziert, um das Produkt der Multiplikation als ein zweites Ergebnis auszugeben, wobei das zweite Ergebnis als ein erster Tiefpaßkoeffizient ausgegeben wird;
einen ersten Subtrahierer, in den ein Tiefpaßkoeffizient eingeht, der von dem zweiten Ergebnis subtrahiert wird, um die Differenz als ein drittes Ergebnis zu erzeugen, wobei der Tiefpaßkoeffizient von dem ersten Subtrahierer ausgegeben und als Eingabe zu dem ersten Subtrahierer zurückgeführt wird;
einen zweiten Subtrahierer, um ein zweites Paar von Proben bzw. Abtastsignalen des Eingabedatensignals voneinander zu subtrahieren, um die Differenz als ein, viertes Ergebnis zu erzeugen und auszugeben;
eine erste Divisionseinrichtung, die das vierte Ergebnis empfängt und das dritte Ergebnis durch einen zweiten Faktor dividiert, um das Ergebnis der Division als ein fünftes Ergebnis zu erzeugen und auszugeben;
einen zweiten Addierer, in den das dritte Ergebnis und das zweite Ergebnis eingeht und der das dritte und vierte Ergebnis addiert und die Summe als sechstes Ergebnis ausgibt;
eine zweite Divisionseinrichtung, die das sechste Ergebnis empfängt und das sechste Ergebnis durch einen dritten Faktor dividiert, um einen zweiten Tiefpaßkoef-fizienten zu erzeugen, wobei der erste Tiefpaßkoeffizient und der dritte Tiefpaßkoef-fizient von dem Filter ausgegeben werden.

10. Filter nach Anspruch 9, in dem die Multiplizierereinrichtung eine Verschiebungseinrichtung aufweist.

11. Filter nach Anspruch 9, in dem wenigstens eine der Divisionseinrichtungen eine Verschiebungseinrichtung aufweist.

12. Wavelet-Transformationsfilter zum Transformieren von Eingangsdatensignalen, das aufweist:
einen ersten Subtrahierer, um ein Paar von Tiefpaßkoeffizienten zu subtrahieren, um die Differenz als ein erstes Ergebnis zu erzeugen und auszugeben;
einen zweiten Subtrahierer, in den das erste Ergebnis eingeht und der das erste Ergebnis von einem Hochpaßkoeffizienten subtrahiert, um die Differenz als ein zweites Ergebnis zu erzeugen und auszugeben;
einen ersten Addierer, der das zweite Ergebnis mit einem Tiefpaßkoeffizienten addiert, der als zweite Eingabe in den ersten Addierer eingegeben wird, um die Summe als ein drittes Ergebnis zu erzeugen;
einen dritten Subtrahierer, um das zweite Ergebnis von der zweiten Eingabe, die auf dem Tiefpaßkoeffizienten basiert, zu subtrahieren, um ein viertes Ergebnis zu erzeugen; wobei das Filter eine erste Probe bzw. Abtastsignal, das auf dem dritten Ergebnis basiert und eine zweiten Probe bzw. Abtastsignal, das auf dem vierten Ergebnis basiert, ausgibt.

13. Filter nach Anspruch 12, das ferner einen ersten Abschneidemechanismus aufweist, der angekoppelt bzw. angeschlossen ist, um das dritte Ergebnis zu empfangen und das dritte Ergebnis abzuschneiden, um die erste Probe bzw. das erste Abtastsignal zu erzeugen.

14. Filter nach Anspruch 12, das ferner eine Divisionseinrichtung aufweist, die an das erste Ergebnis angekoppelt bzw. angeschlossen ist, um das erste Ergebnis durch einen ersten Faktor zu dividieren, um den ersten Eingang bzw. Eingabe zu erzeugen.

15. Filter nach Anspruch 12, das ferner eine Multipliziereinrichtung aufweist, die an den Tiefpaßkoeffizienten angekoppelt bzw. angeschlossen ist, um den Tiefpaßkoeffizienten zu multiplizieren, um den zweiten Eingang bzw. Eingabe zu erzeugen.

16. Codierer zum Codieren einer Vielzahl von Datensymbolen, der aufweist:
eine Formatierungseinheit, die angekoppelt bzw. angeschlossen ist, um die Vielzahl von Datensymbolen zu empfangen und die Vielzahl von Datensymbolen in einen Satz von formatierten Datensymbolen zu formatieren; wobei durch die Formatierung eine Anzahl von Datensymbolen in Symbole basierend auf einem Datenformat formatiert werden,
eine Signifikanz- bzw. Wertigkeitseinheit, die an die Formatierungseinheit angekoppelt bzw. angeschlossen ist, um eine Vielzahl von Entscheidungen in Antwort auf jedes der Vielzahl der Datensymbole zu erzeugen;
eine Speichereinrichtung, die angekoppelt bzw. angeschlossen ist, um die Vielzahl von Entscheidungen von der Signifikanz- bzw. Wertigkeitseinheit zum Speichern und Ausgeben zu empfangen.

17. Codierer nach Anspruch 16, in dem die Mehrzahl von Datensymbolen eine Mehrzahl von Koeffizienten aufweist.

18. Codierer nach Anpruch 16, in dem die Formatierungseinheit eine (Vor-)Zeichen-/Größenordnungseinheit aufweist, um die Mehrzahl von Datensymbolen in den Satz von formatierten Datensymbolen umzusetzen, die in (Vor-)Zeichen-/Größenordnungsform formatiert sind.

19. Codierer nach Anspruch 18, in dem die (Vor-)Zeichen-/Größenordnungseinheit aufweist:
eine Signifikanz- bzw. Wertigkeitsbestimmungseinrichtung, um eine Signifikanzanzeige für jedes Datensymbol auszugeben;
einen (Vor-)Zeichen- und Mantissengenerator, der angekoppelt bzw. angeschlossen ist, um jedes der Mehrzahl der Datensymbole zu empfangen und (Vor-)Zeichen und Mantissen für jedes der Mehrzahl von Datensymbolen zu erzeugen; und
eine Index-Zähleinrichtung zum Erzeugen eines Index für jedes der Mehrzahl von Datensymbolen.

20. Codierer nach Anspruch 19, in dem die Signifikanz- bzw. Wertigkeitsbestimmungseinrichtung einen Prioritätscodierer aufweist.

21. Codierer nach Anspruch 19, in dem die Signifikanz- bzw. Wertigkeitsbestimmungseinrichtung ein Paar von Prioritätscodierern und eine Auswähleinrichtung aufweist, um zwischen Ausgängen von jedem des Paares von Prioritätscodierern auszuwählen, um die Signifikanz- bzw. Wertigkeitsanzeige auszugeben.

22. Codierer nach Anspruch 16, der ferner einen Größenordnungsspeicher aufweist, der angekoppelt bzw. angeschlossen ist, um die Mehrzahl von formatierten Datensymbolen zur Speicherung zu empfangen.

23. Codierer nach Anspruch 22, in dem jedes der Mehrzahl von formatierten Datensymbolen in dem Größenordnungsspeicher auf der Grundlage eines Signifikanz- bzw. Wertigkeitspegels gespeichert wird.

24. Codierer nach Anspruch 22, in dem der Größenordnungsspeicher eine Mehrzahl von Speicherbereichen aufweist, wobei jedes der Mehrzahl von Speicherbereichen Daten speichert für einen unterschiedlichen Signifikanz- bzw. Wertigkeitspegel, der Größenordnungsspeicher ferner eine Mehrzahl von Zähleinrichtungen bzw. Zählern aufweist, wobei jeder der Mehrzahl von Zählern bzw. Zähleinrichtungen mit einem der Mehrzahl der Speicherbereiche verbunden ist, um Adressen für die Speicherung der formatierten Datensymbole zur Verfügung zu stellen.

Es folgen 54 Blatt Zeichnungen

Anhängende Zeichnungen

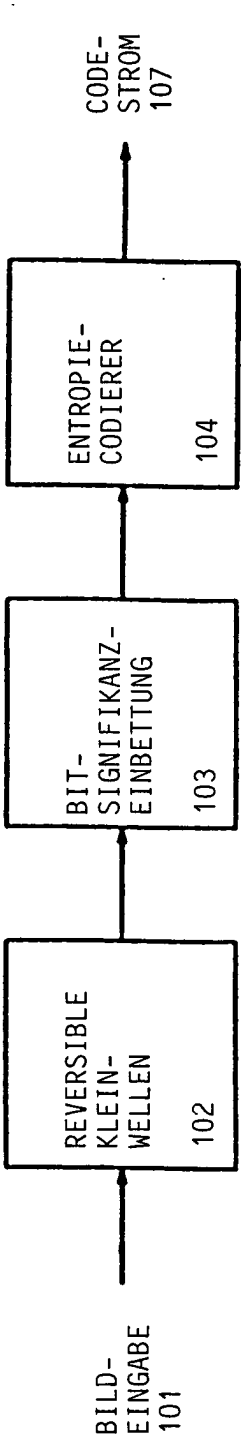


FIG. 1A

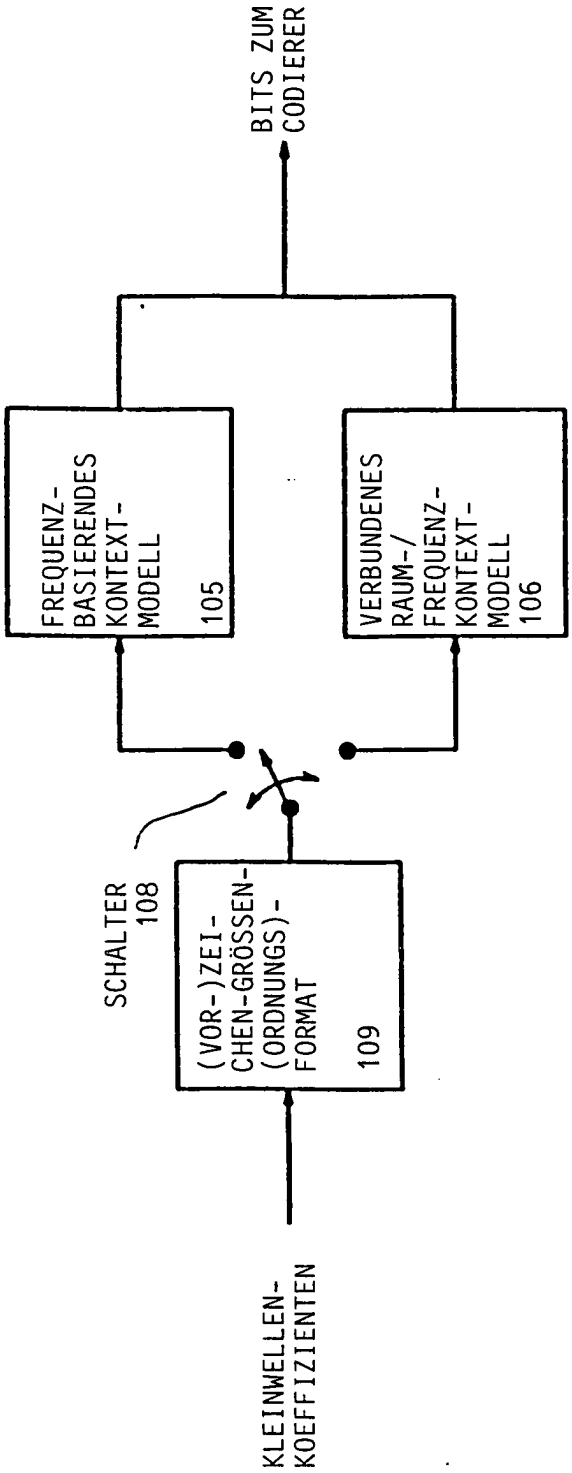

BITS ZUM CODIERER
FREQUENZ-BASIERENDES KONTEXT-MODELL 105
VERBUNDENES RAUM-/FREQUENZ-KONTEXT-MODELL 106
SCHALTER 108
(VOR-)ZEICHEN-GRÖSSEN-(ORDNUNGS)-FORMAT 109
KLEINWELLEN-KOEFFIZIENTEN

FIG. 1B

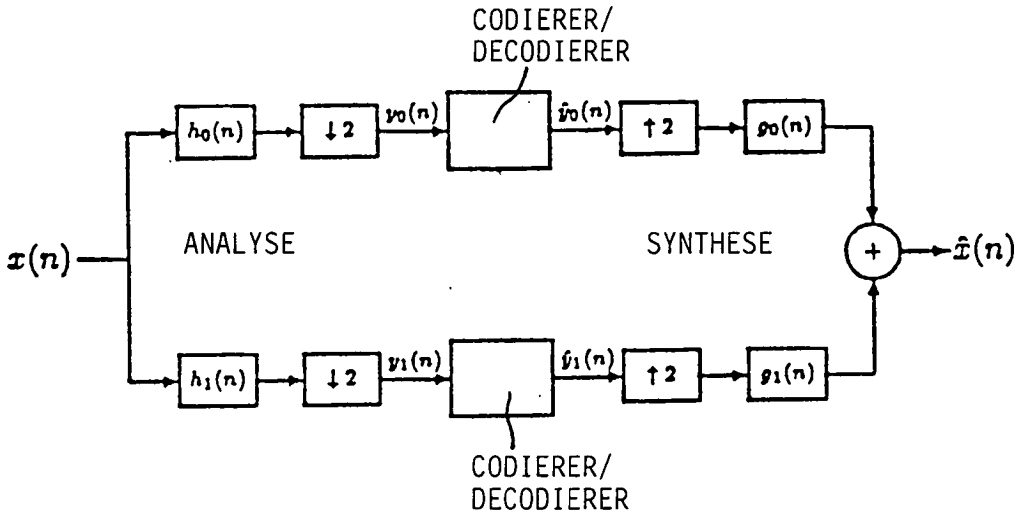
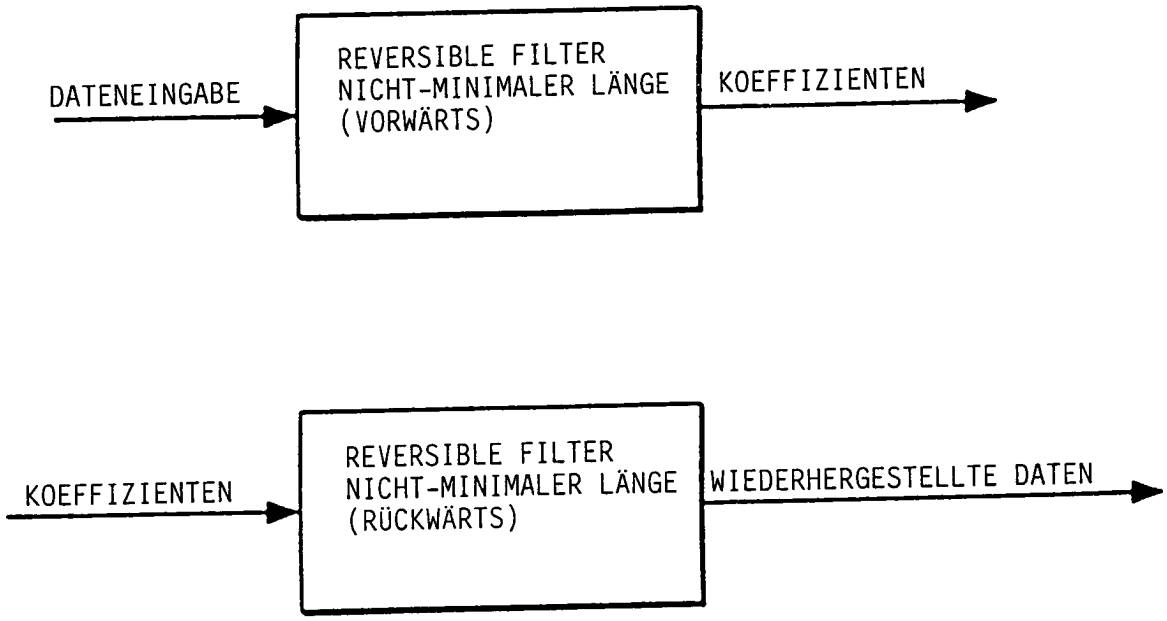
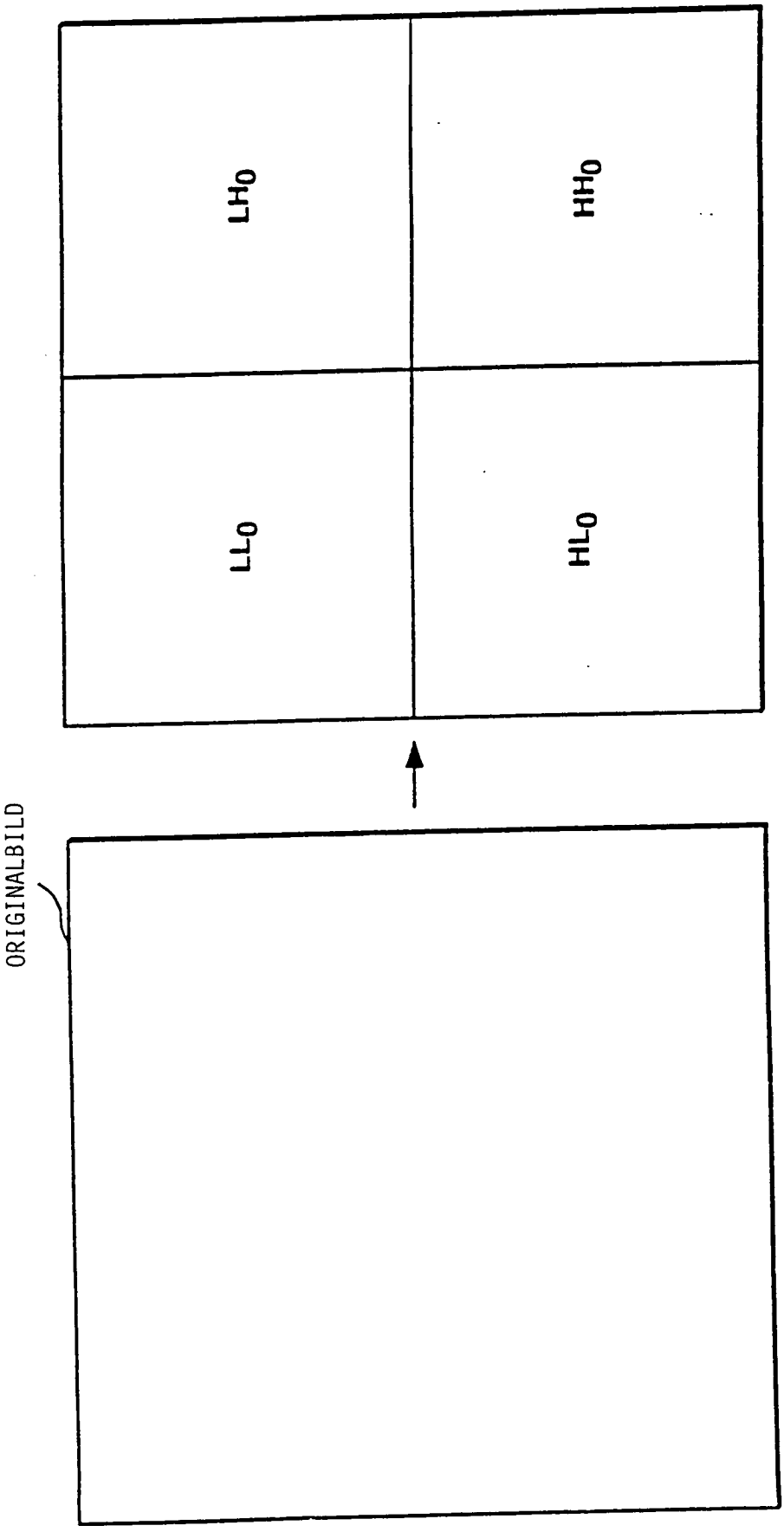

CODIERER/
DECODIERER
$h_0(n)$
↓2
$y_0(n)$
$\hat{y}_0(n)$
↑2
$g_0(n)$
$x(n)$
ANALYSE
SYNTHESE
+
$\hat{x}(n)$
$h_1(n)$
↓2
$y_1(n)$
$\hat{y}_1(n)$
↑2
$g_1(n)$
CODIERER/
DECODIERER

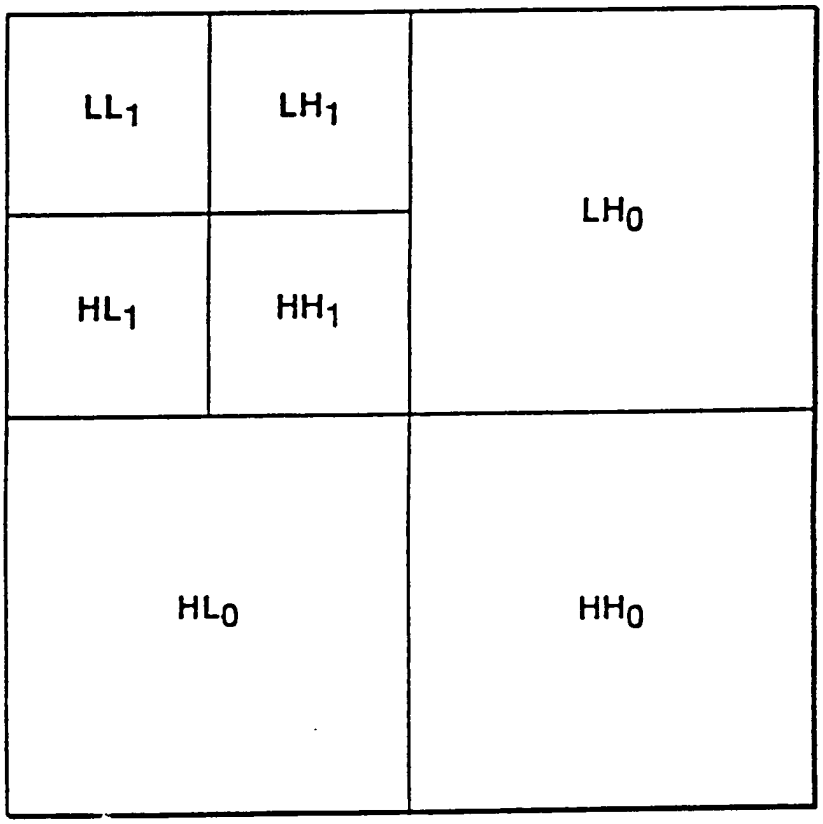
FIG. 2A


DATENEINGABE
REVERSIBLE FILTER
NICHT-MINIMALER LÄNGE
(VORWÄRTS)
KOEFFIZIENTEN
KOEFFIZIENTEN
REVERSIBLE FILTER
NICHT-MINIMALER LÄNGE
(RÜCKWÄRTS)
WIEDERHERGESTELLTE DATEN

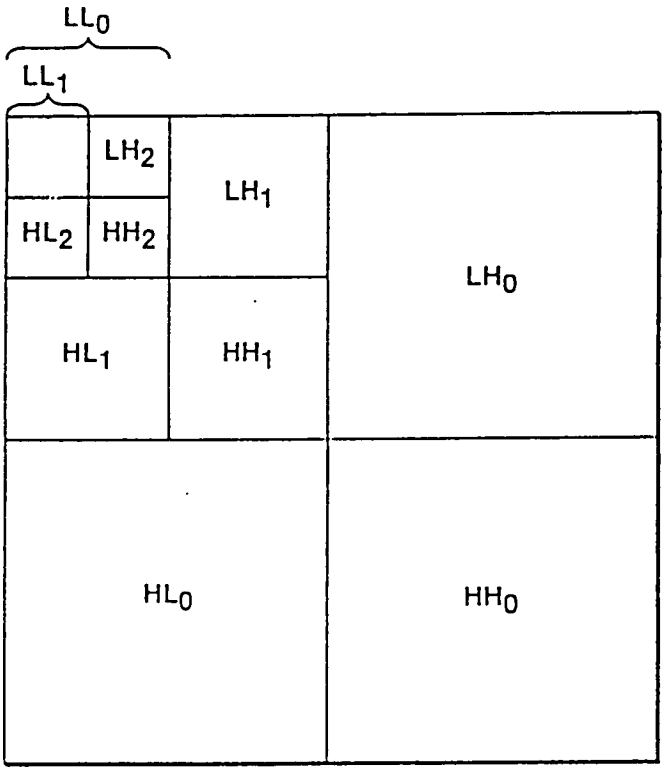
Figur 2B



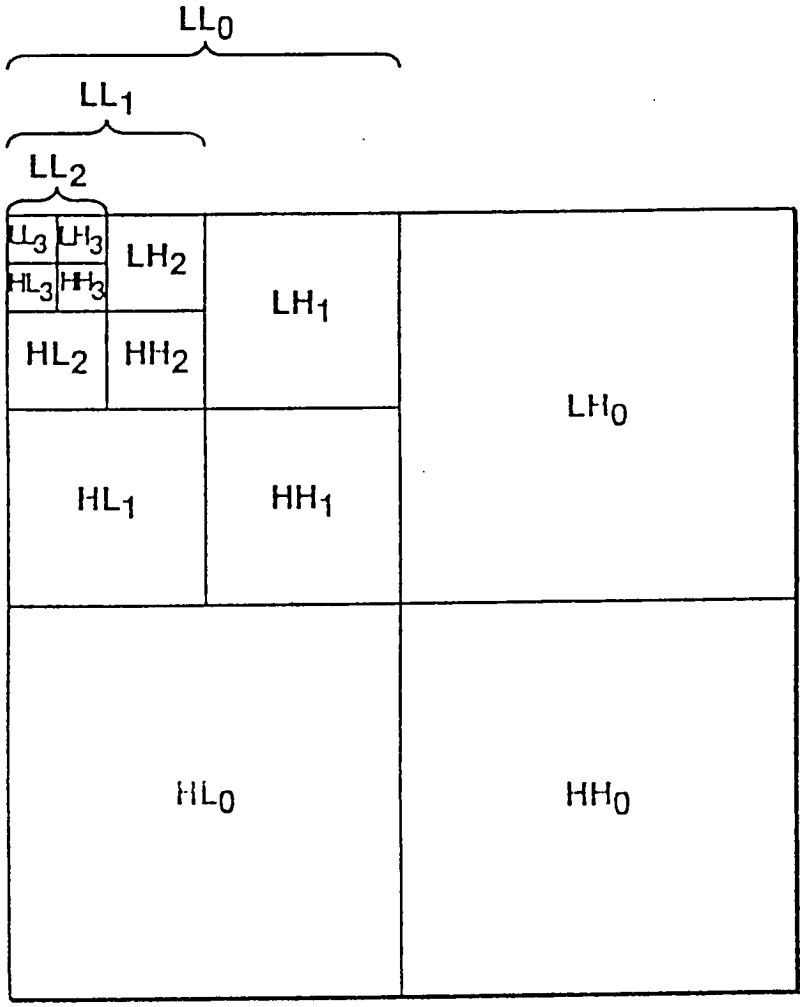
Figur 3A



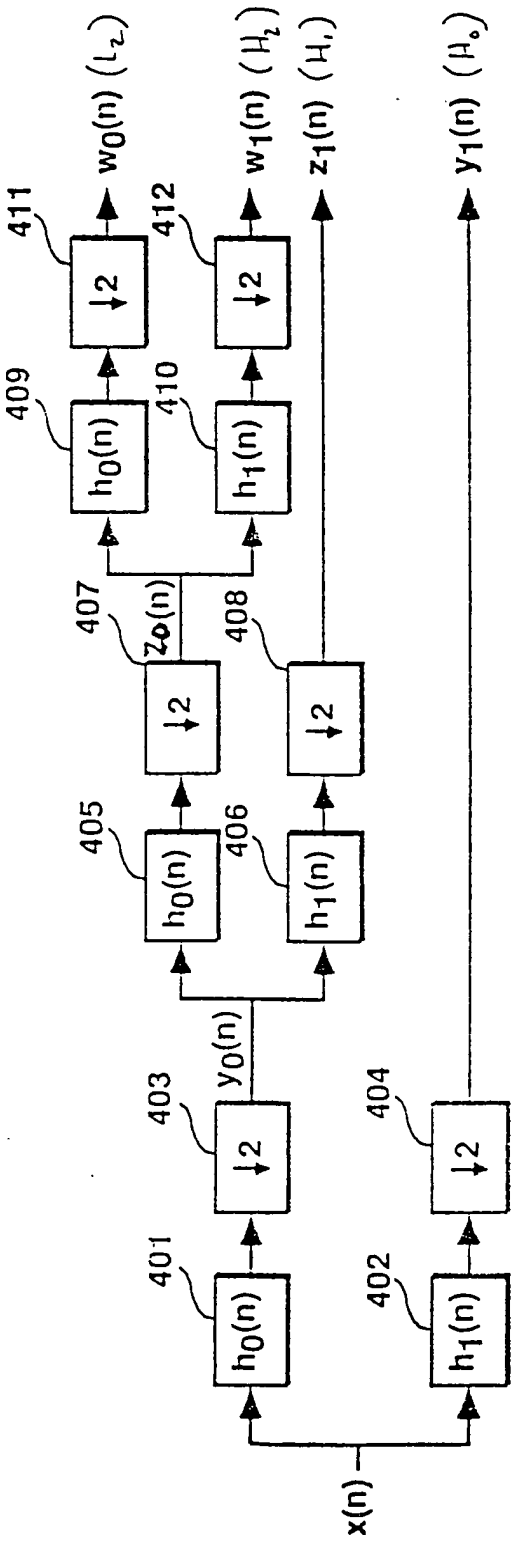
Figur 3B



Figur 3C

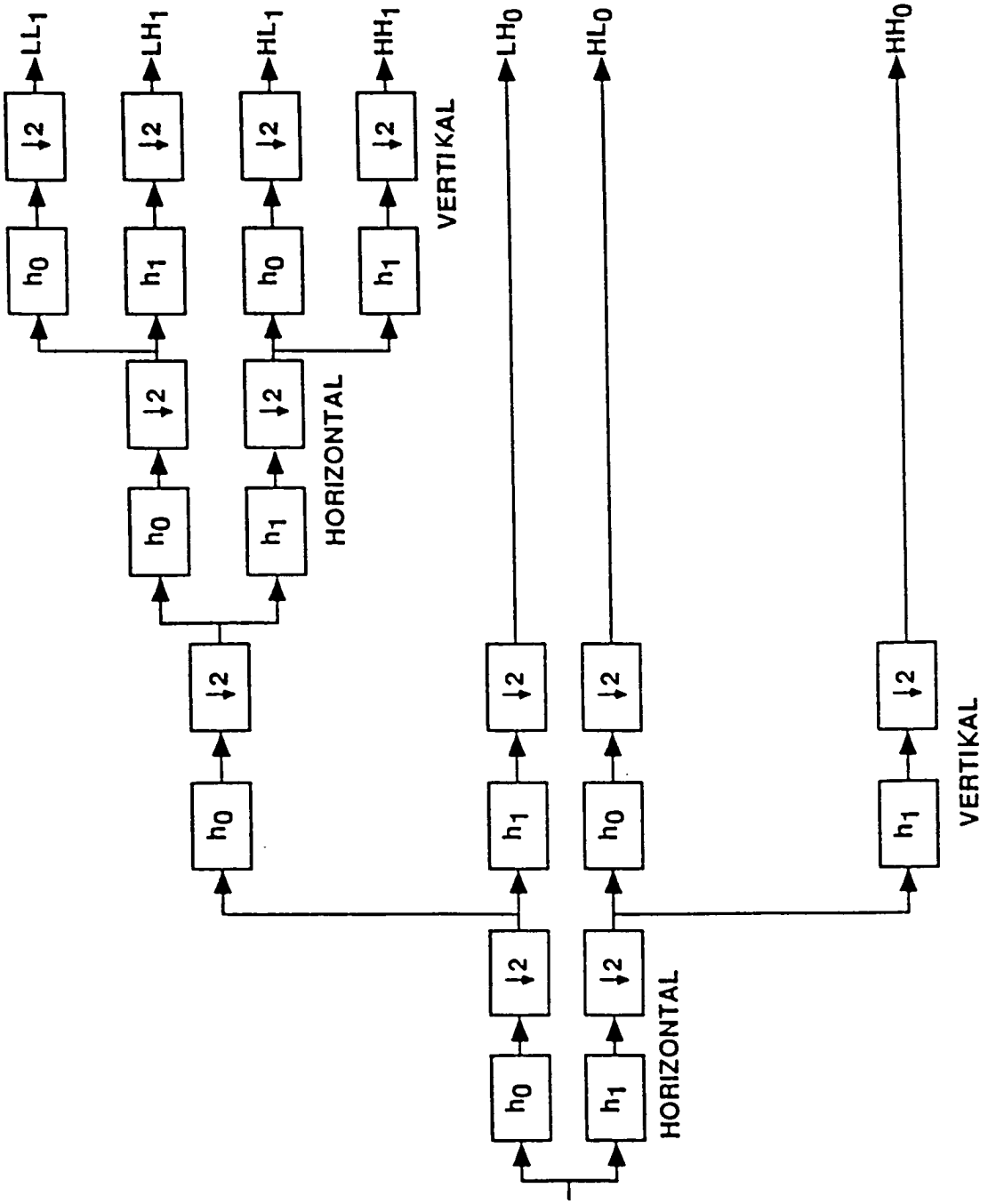


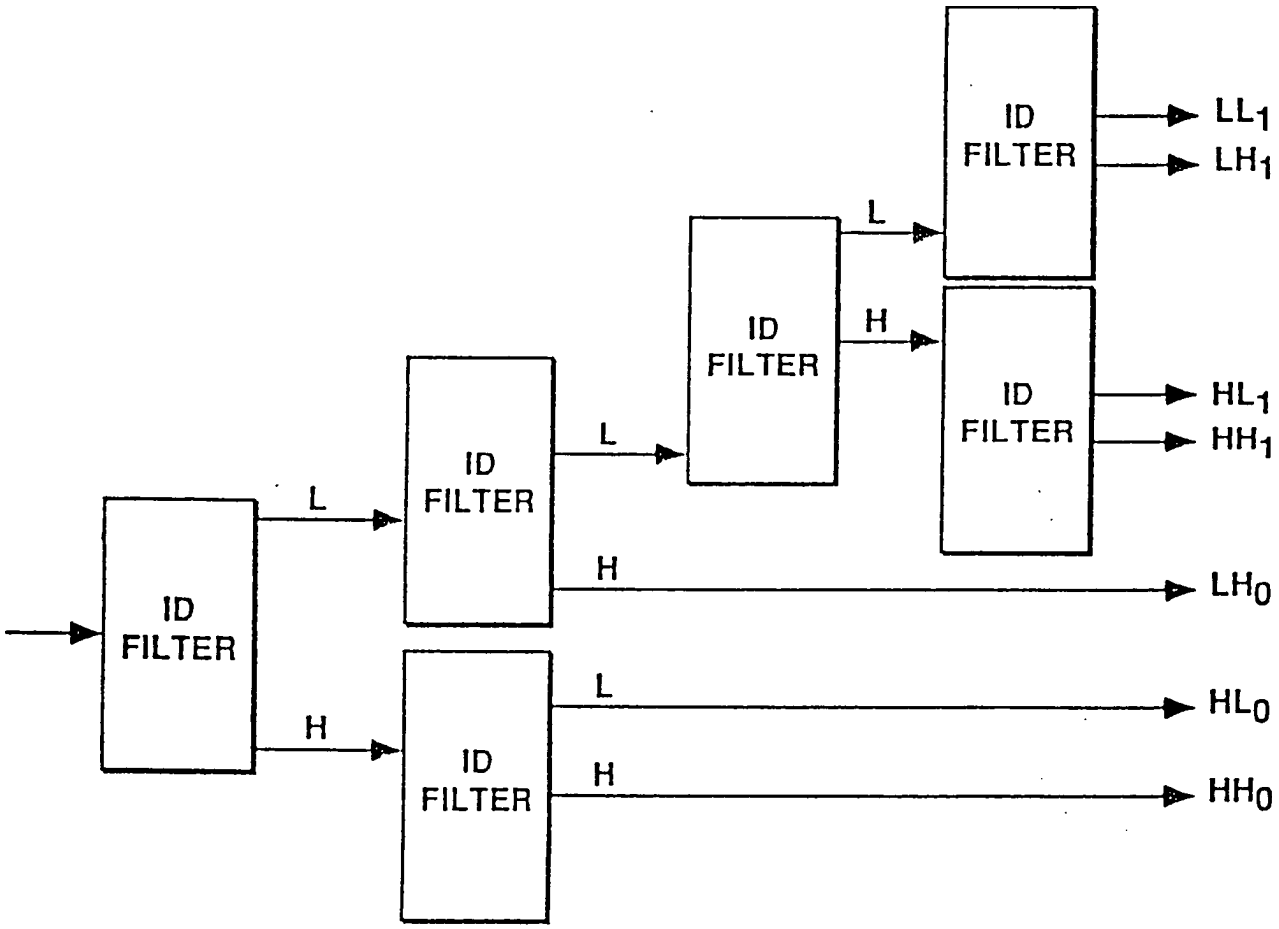
Figur 3D


x(n)
401
h0(n)
402
h1(n)
403
↓2
404
↓2
y0(n)
405
h0(n)
406
h1(n)
407
↓2
408
↓2
z0(n)
409
h0(n)
410
h1(n)
411
↓2
412
↓2
w0(n) (L2)
w1(n) (H2)
z1(n) (H1)
y1(n) (H0)

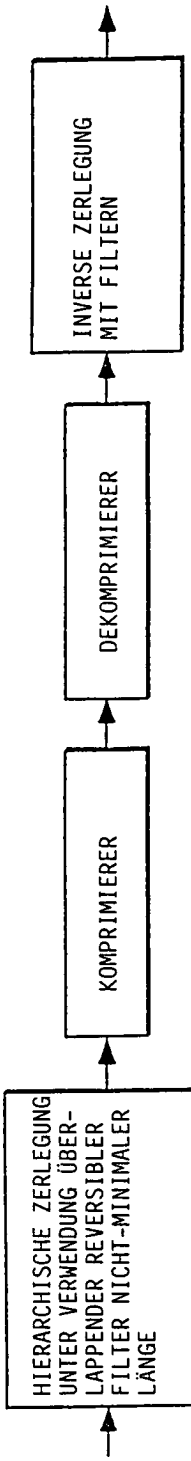
Figur 4A

Figur 4B

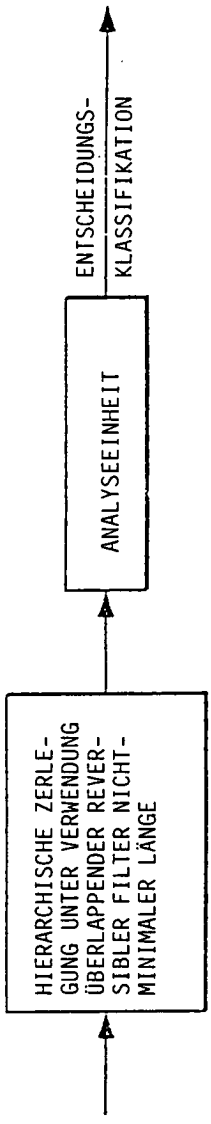

LL1
LH1
HL1
HH1
LH0
HL0
HH0
↓2
h0
h1
VERTIKAL
HORIZONTAL
VERTIKAL
HORIZONTAL

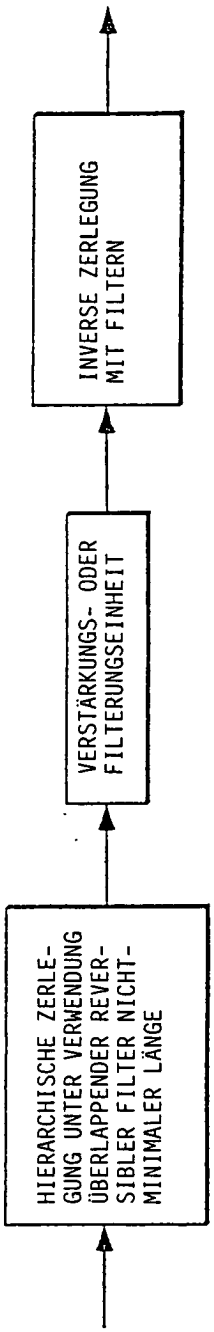

ID FILTER
L
H
ID FILTER
L
H
ID FILTER
L
H
ID FILTER
LL1
LH1
HL1
HH1
LH0
HL0
HH0

Figur 4C



Figur 4D





Figur 4E

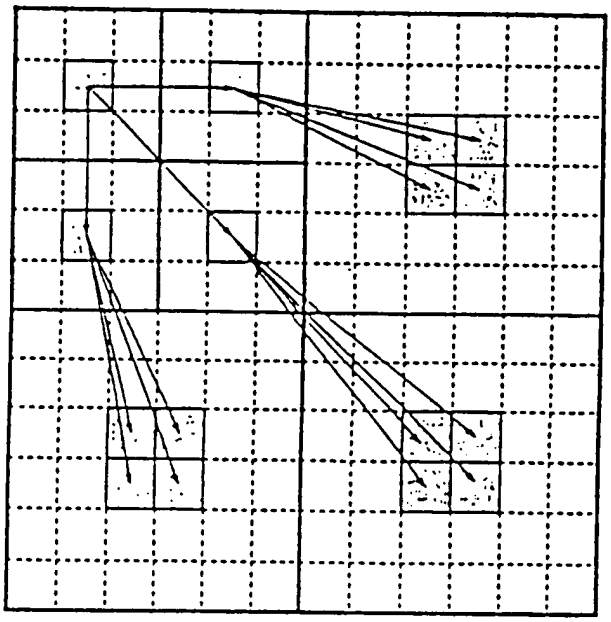


FIG. 5

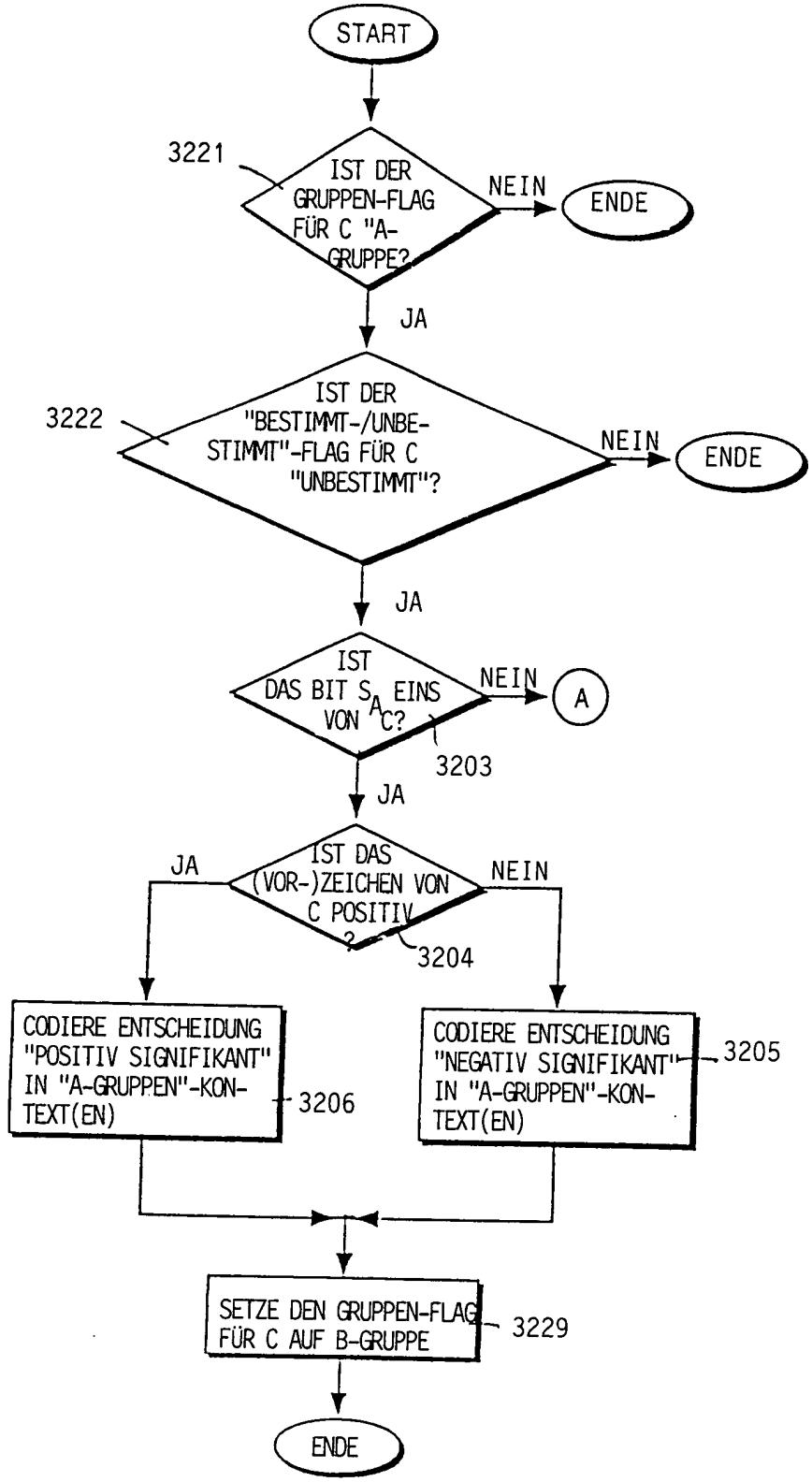

START
3221
IST DER GRUPPEN-FLAG FÜR C "A-GRUPPE?
NEIN
ENDE
JA
3222
IST DER "BESTIMMT-/UNBESTIMMT"-FLAG FÜR C "UNBESTIMMT"?
NEIN
ENDE
JA
IST DAS BIT S_A EINS VON C?
NEIN
A
3203
JA
JA
IST DAS (VOR-)ZEICHEN VON C POSITIV ?
NEIN
3204
CODIERE ENTSCHEIDUNG "POSITIV SIGNIFIKANT" IN "A-GRUPPEN"-KONTEXT(EN)
3206
CODIERE ENTSCHEIDUNG "NEGATIV SIGNIFIKANT" IN "A-GRUPPEN"-KONTEXT(EN)
3205
SETZE DEN GRUPPEN-FLAG FÜR C AUF B-GRUPPE
3229
ENDE

FIG. 6a

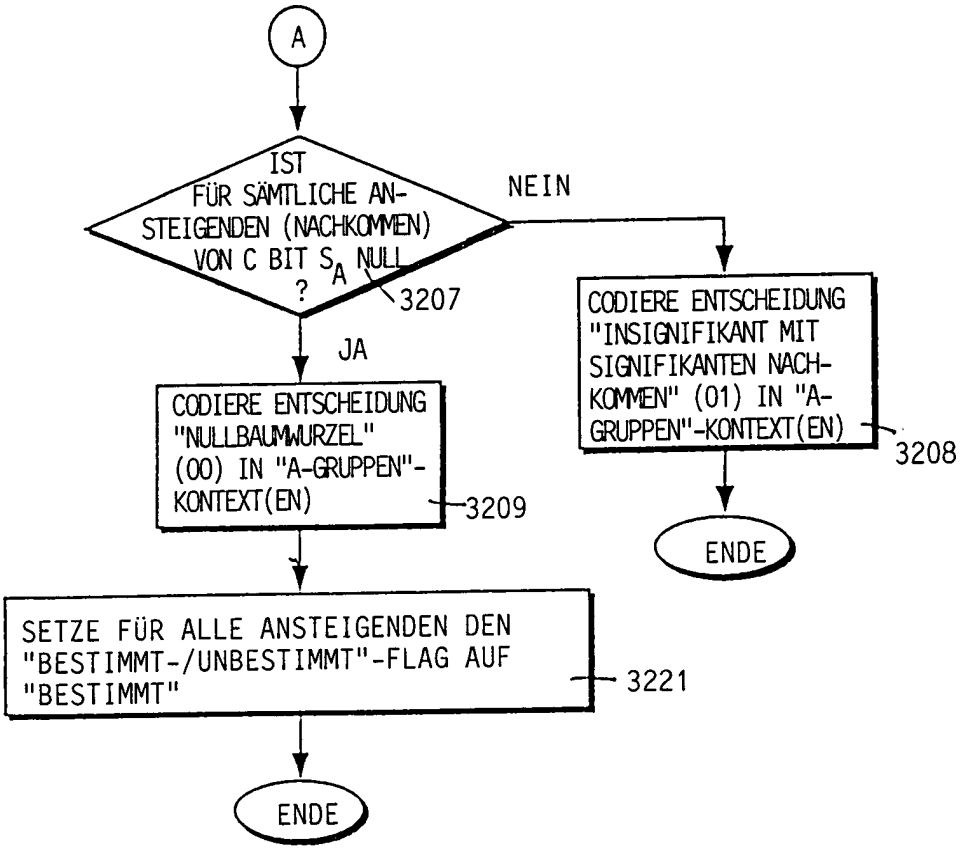


FIG. 6a (FORTGESETZT)

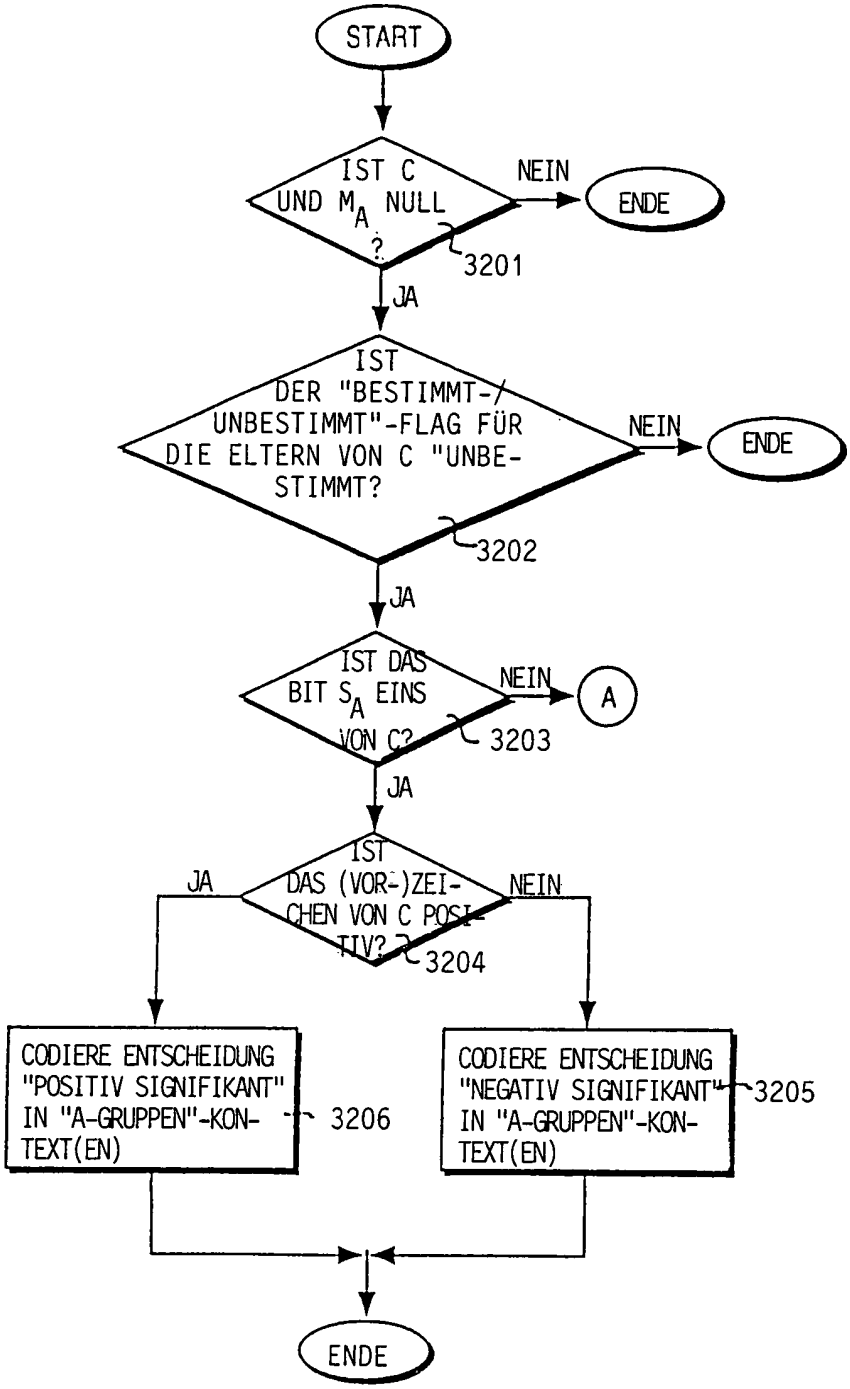

START
IST C UND M_A NULL ?
NEIN
ENDE
3201
JA
IST DER "BESTIMMT-/UNBESTIMMT"-FLAG FÜR DIE ELTERN VON C "UNBESTIMMT?
NEIN
ENDE
3202
JA
IST DAS BIT S_A EINS VON C?
NEIN
A
3203
JA
IST DAS (VOR-)ZEICHEN VON C POSITIV?
JA
NEIN
3204
CODIERE ENTSCHEIDUNG "POSITIV SIGNIFIKANT" IN "A-GRUPPEN"-KONTEXT(EN)
3206
CODIERE ENTSCHEIDUNG "NEGATIV SIGNIFIKANT" IN "A-GRUPPEN"-KONTEXT(EN)
3205
ENDE

FIG. 6b

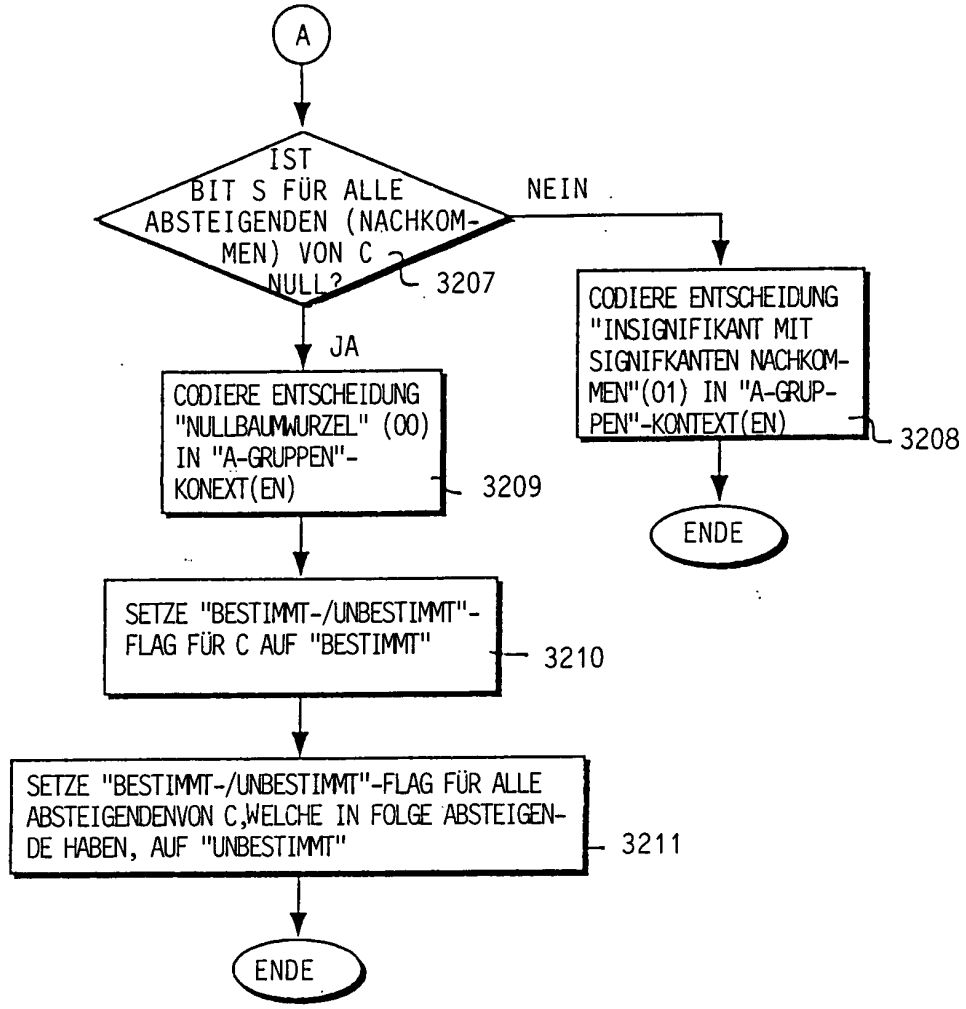


FIG. 6D (FORTGESETZT)

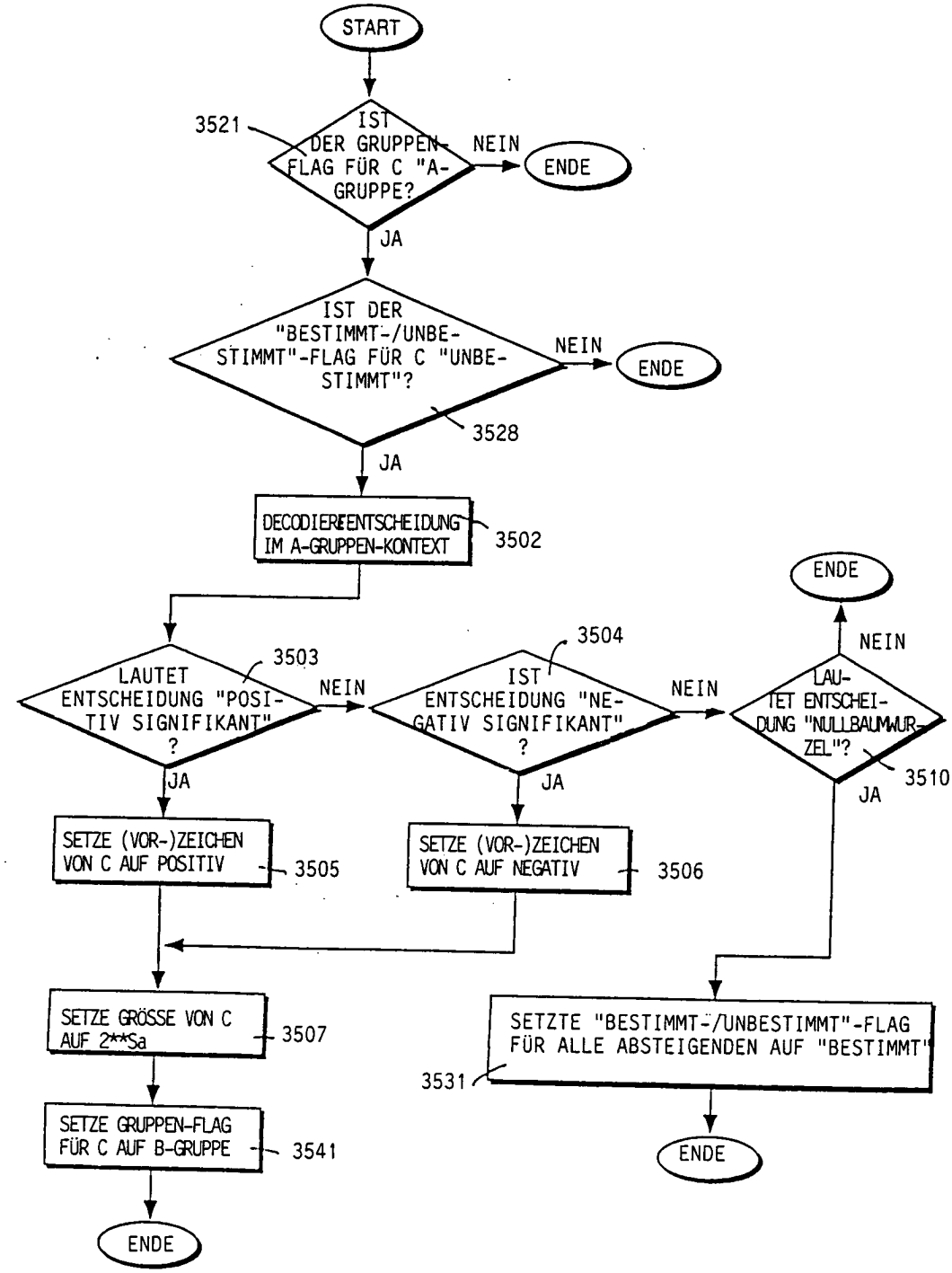

START
IST DER GRUPPEN-FLAG FÜR C "A-GRUPPE?
3521
NEIN
ENDE
JA
IST DER "BESTIMMT-/UNBESTIMMT"-FLAG FÜR C "UNBESTIMMT"?
NEIN
ENDE
3528
JA
DECODIERE ENTSCHEIDUNG IM A-GRUPPEN-KONTEXT
3502
LAUTET ENTSCHEIDUNG "POSITIV SIGNIFIKANT" ?
3503
NEIN
IST ENTSCHEIDUNG "NEGATIV SIGNIFIKANT" ?
3504
NEIN
LAUTET ENTSCHEIDUNG "NULLBAUMWURZEL"?
3510
NEIN
ENDE
JA
JA
JA
SETZE (VOR-)ZEICHEN VON C AUF POSITIV
3505
SETZE (VOR-)ZEICHEN VON C AUF NEGATIV
3506
SETZE GRÖSSE VON C AUF 2**Sa
3507
SETZE GRUPPEN-FLAG FÜR C AUF B-GRUPPE
3541
ENDE
SETZTE "BESTIMMT-/UNBESTIMMT"-FLAG FÜR ALLE ABSTEIGENDEN AUF "BESTIMMT"
3531
ENDE

FIG. 6c

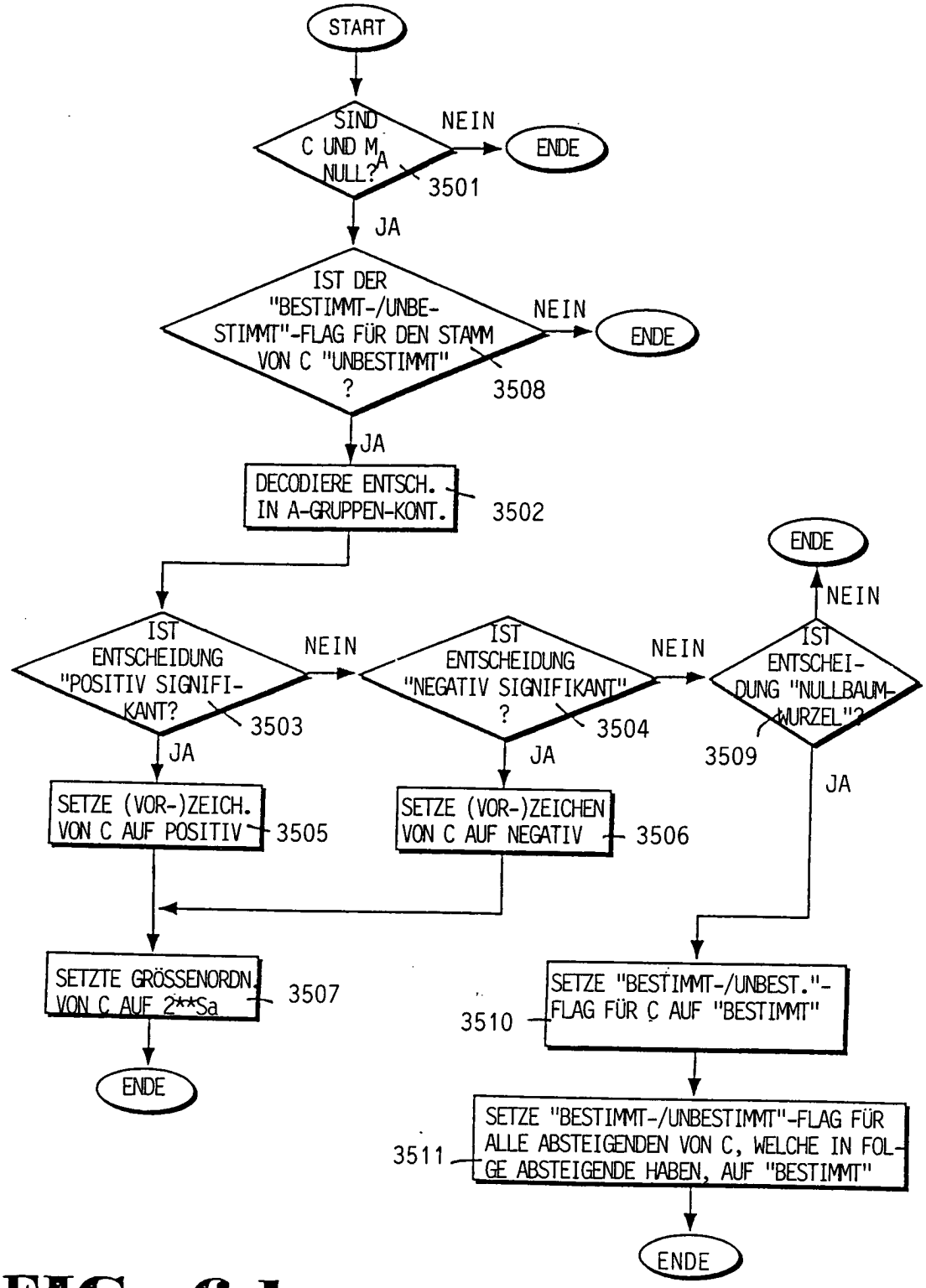

START
SIND C UND M_A NULL?
3501
NEIN
ENDE
JA
IST DER "BESTIMMT-/UNBESTIMMT"-FLAG FÜR DEN STAMM VON C "UNBESTIMMT" ?
3508
NEIN
ENDE
JA
DECODIERE ENTSCH. IN A-GRUPPEN-KONT.
3502
IST ENTSCHEIDUNG "POSITIV SIGNIFIKANT?
3503
NEIN
IST ENTSCHEIDUNG "NEGATIV SIGNIFIKANT" ?
3504
NEIN
IST ENTSCHEIDUNG "NULLBAUMWURZEL"?
3509
NEIN
ENDE
JA
JA
JA
SETZE (VOR-)ZEICH. VON C AUF POSITIV
3505
SETZE (VOR-)ZEICHEN VON C AUF NEGATIV
3506
SETZTE GRÖSSENORDN. VON C AUF 2**Sa
3507
ENDE
SETZE "BESTIMMT-/UNBEST."-FLAG FÜR C AUF "BESTIMMT"
3510
SETZE "BESTIMMT-/UNBESTIMMT"-FLAG FÜR ALLE ABSTEIGENDEN VON C, WELCHE IN FOLGE ABSTEIGENDE HABEN, AUF "BESTIMMT"
3511
ENDE

FIG. 6d

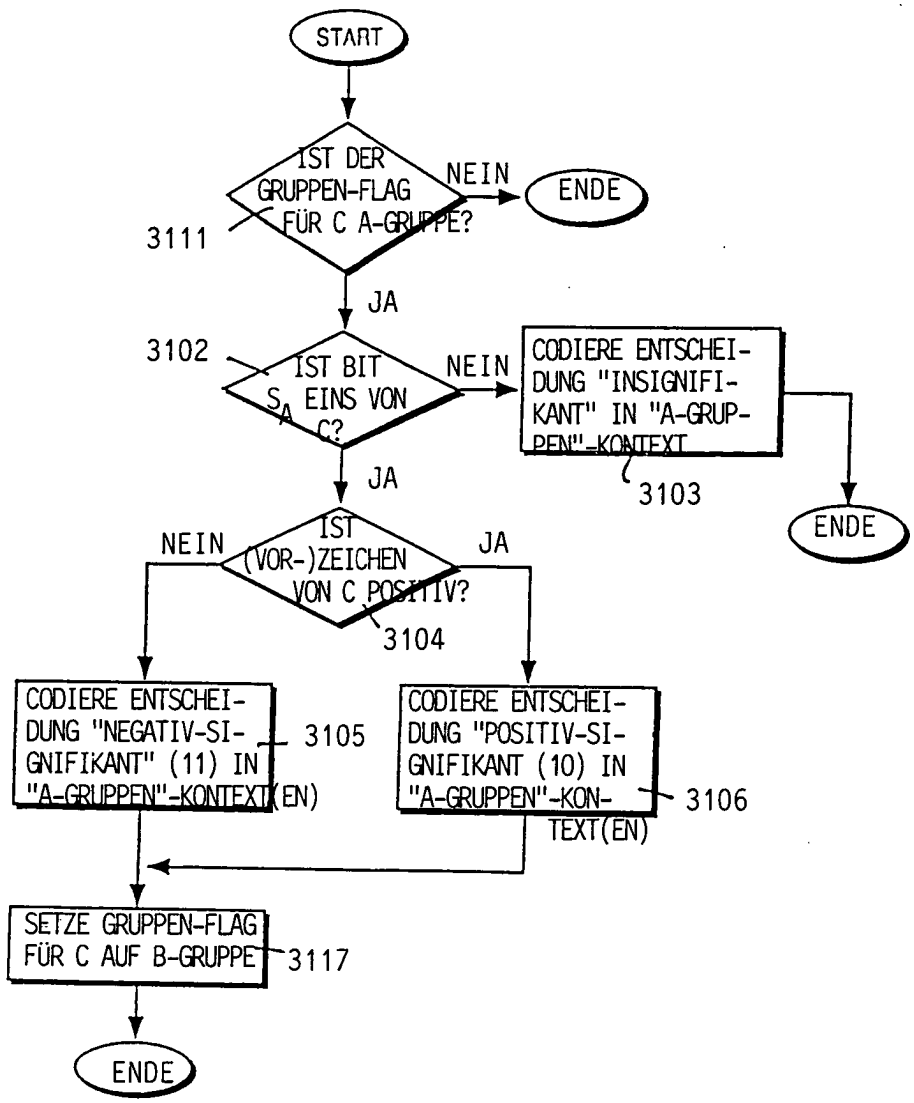

START
IST DER GRUPPEN-FLAG FÜR C A-GRUPPE?
NEIN
ENDE
3111
JA
3102
IST BIT S_A EINS VON C?
NEIN
CODIERE ENTSCHEIDUNG "INSIGNIFIKANT" IN "A-GRUPPEN"-KONTEXT
3103
ENDE
JA
NEIN
IST (VOR-)ZEICHEN VON C POSITIV?
JA
3104
CODIERE ENTSCHEIDUNG "NEGATIV-SIGNIFIKANT" (11) IN "A-GRUPPEN"-KONTEXT(EN)
3105
CODIERE ENTSCHEIDUNG "POSITIV-SIGNIFIKANT (10) IN "A-GRUPPEN"-KONTEXT(EN)
3106
SETZE GRUPPEN-FLAG FÜR C AUF B-GRUPPE
3117
ENDE

FIG. 7a

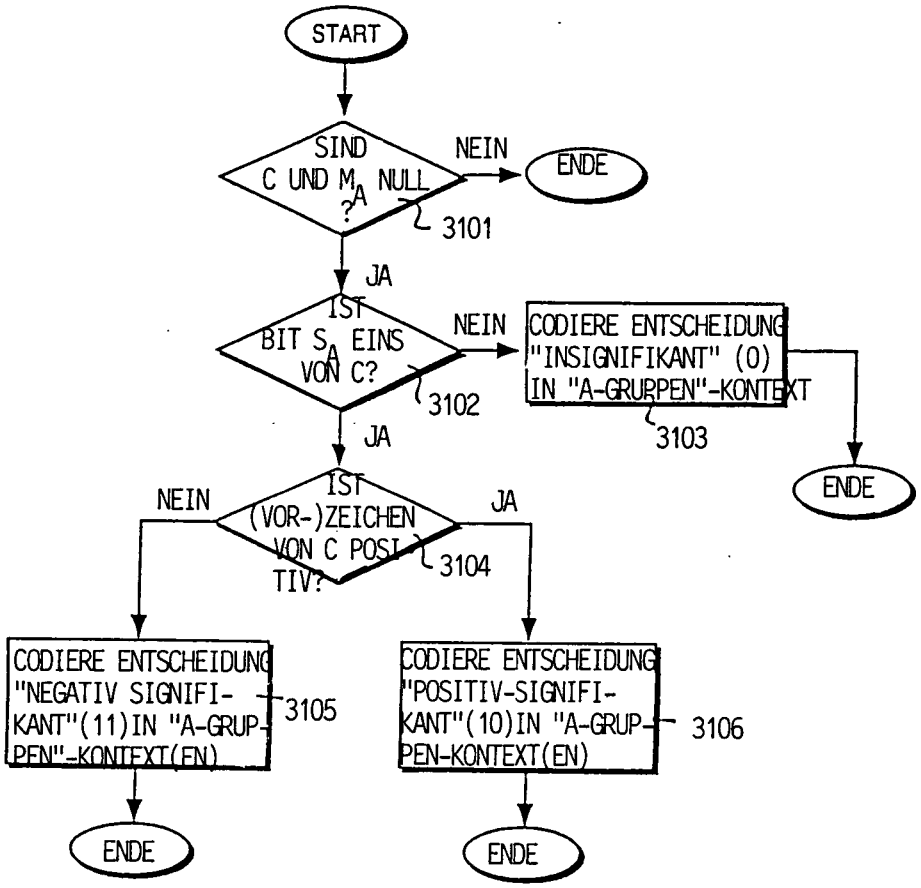

START
SIND C UND M_A NULL ?
NEIN
ENDE
3101
JA
IST BIT S_A EINS VON C?
NEIN
CODIERE ENTSCHEIDUNG "INSIGNIFIKANT" (0) IN "A-GRUPPEN"-KONTEXT
3102
3103
JA
ENDE
IST (VOR-)ZEICHEN VON C POSITIV?
NEIN
JA
3104
CODIERE ENTSCHEIDUNG "NEGATIV SIGNIFIKANT"(11)IN "A-GRUPPEN"-KONTEXT(EN)
3105
CODIERE ENTSCHEIDUNG "POSITIV-SIGNIFIKANT"(10)IN "A-GRUPPEN-KONTEXT(EN)
3106
ENDE
ENDE

FIG. 7b

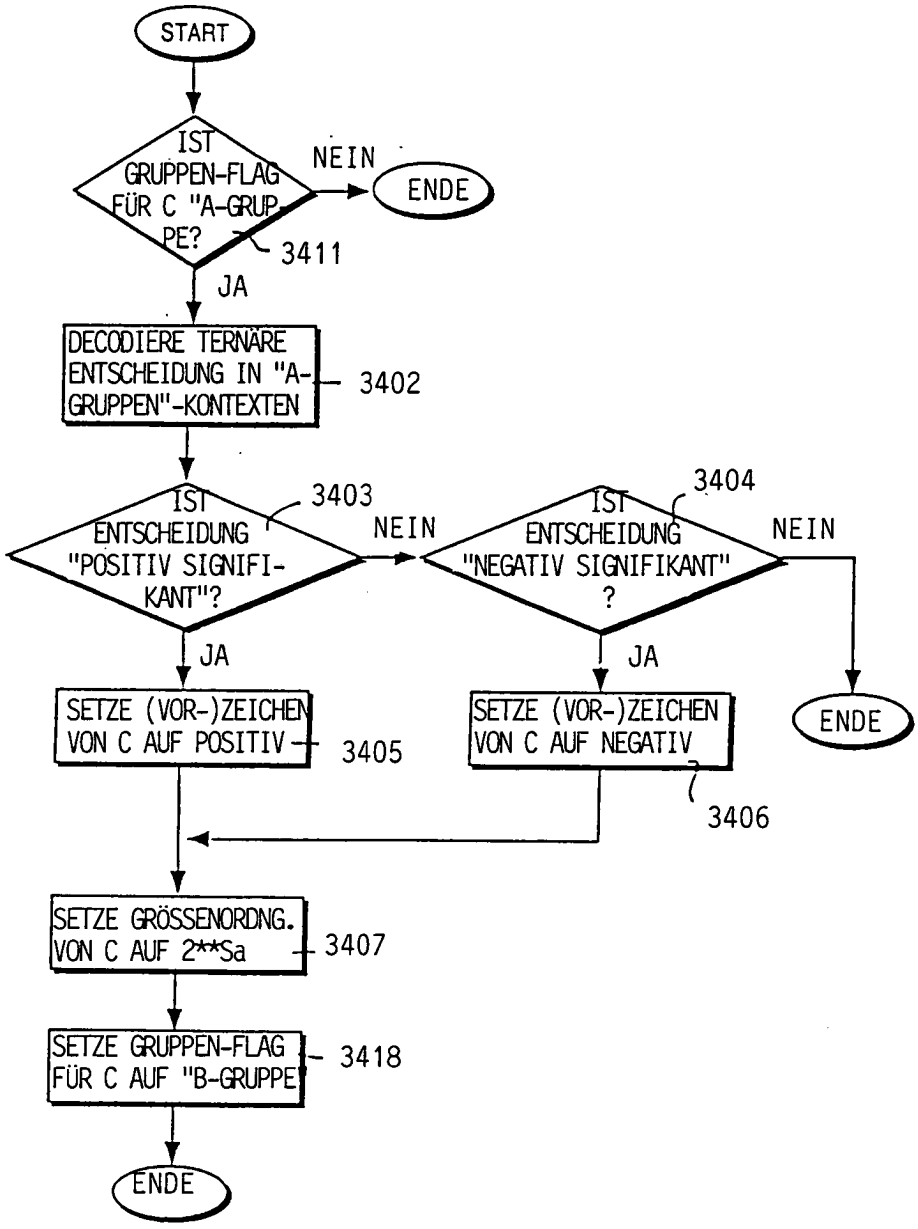

START
IST GRUPPEN-FLAG FÜR C "A-GRUPPE?
NEIN
ENDE
3411
JA
DECODIERE TERNÄRE ENTSCHEIDUNG IN "A-GRUPPEN"-KONTEXTEN
3402
IST ENTSCHEIDUNG "POSITIV SIGNIFIKANT"?
3403
NEIN
IST ENTSCHEIDUNG "NEGATIV SIGNIFIKANT"?
3404
NEIN
JA
JA
SETZE (VOR-)ZEICHEN VON C AUF POSITIV
3405
SETZE (VOR-)ZEICHEN VON C AUF NEGATIV
3406
ENDE
SETZE GRÖSSENORDNG. VON C AUF 2**Sa
3407
SETZE GRUPPEN-FLAG FÜR C AUF "B-GRUPPE"
3418
ENDE

FIG. 7c

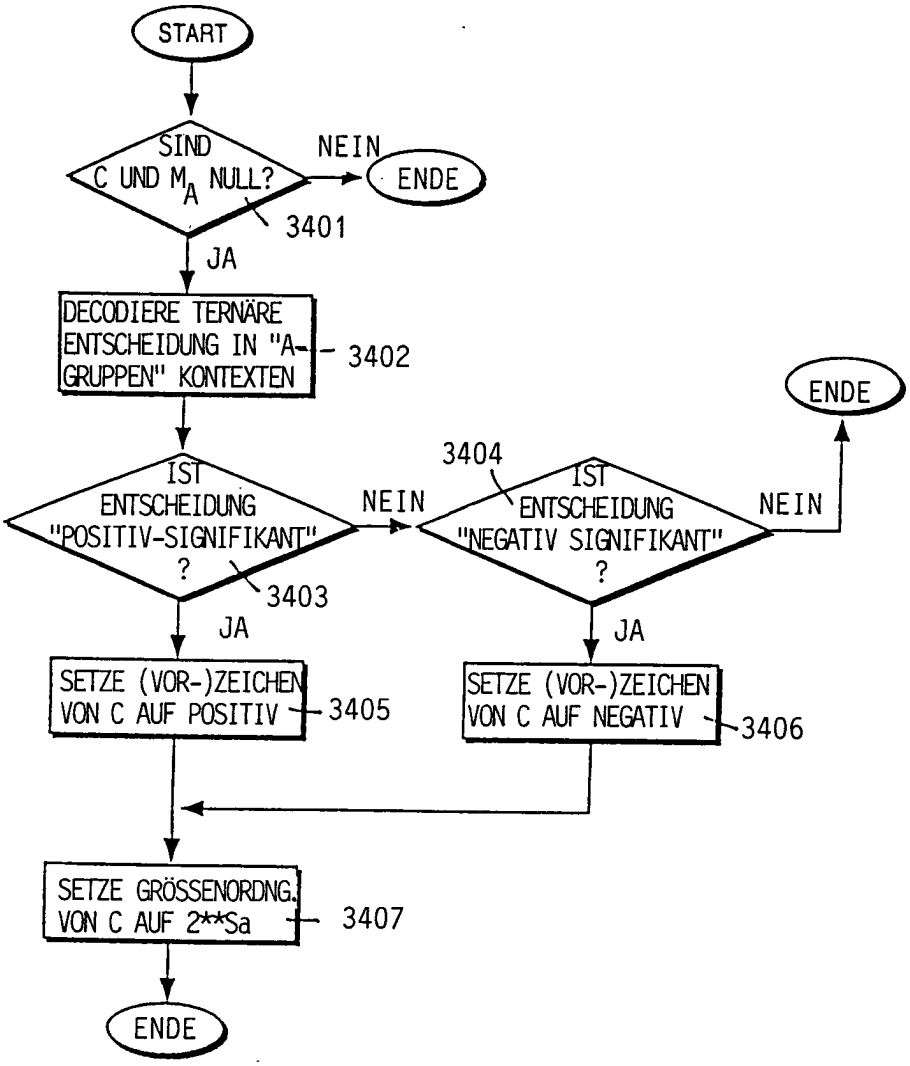

START
SIND C UND M_A NULL?
NEIN
ENDE
3401
JA
DECODIERE TERNÄRE ENTSCHEIDUNG IN "A-GRUPPEN" KONTEXTEN
3402
IST ENTSCHEIDUNG "POSITIV-SIGNIFIKANT" ?
NEIN
3403
JA
3404
IST ENTSCHEIDUNG "NEGATIV SIGNIFIKANT" ?
NEIN
JA
ENDE
SETZE (VOR-)ZEICHEN VON C AUF POSITIV
3405
SETZE (VOR-)ZEICHEN VON C AUF NEGATIV
3406
SETZE GRÖSSENORDNG. VON C AUF 2**Sa
3407
ENDE

FIG. 7d

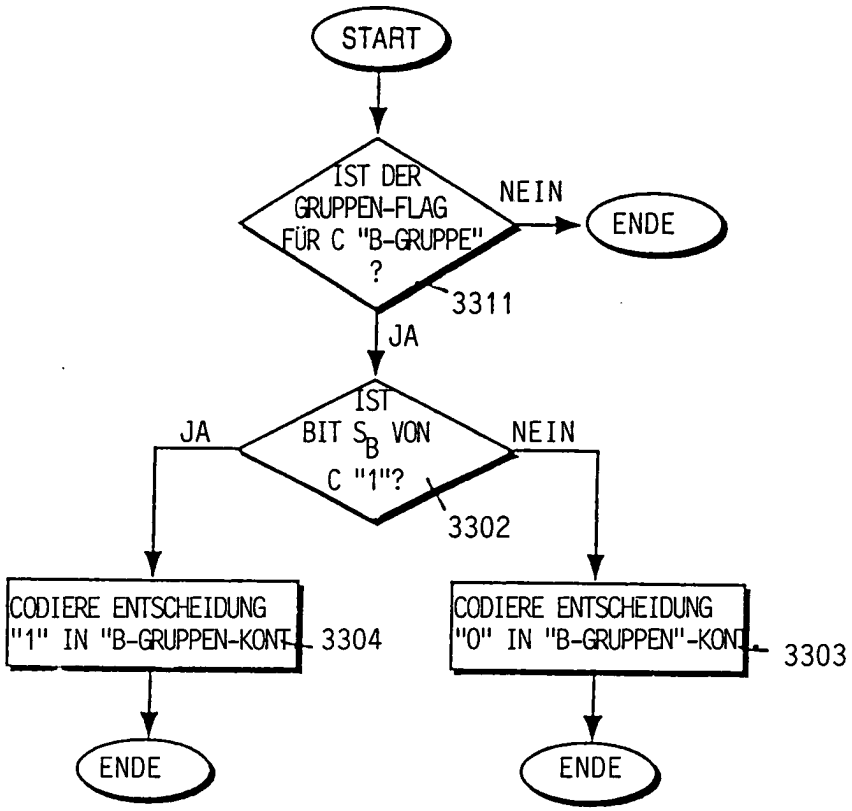

START
IST DER GRUPPEN-FLAG FÜR C "B-GRUPPE" ?
NEIN
ENDE
3311
JA
IST BIT S_B VON C "1"?
JA
NEIN
3302
CODIERE ENTSCHEIDUNG "1" IN "B-GRUPPEN-KONT
3304
CODIERE ENTSCHEIDUNG "0" IN "B-GRUPPEN"-KONT
3303
ENDE
ENDE

FIG. 8a

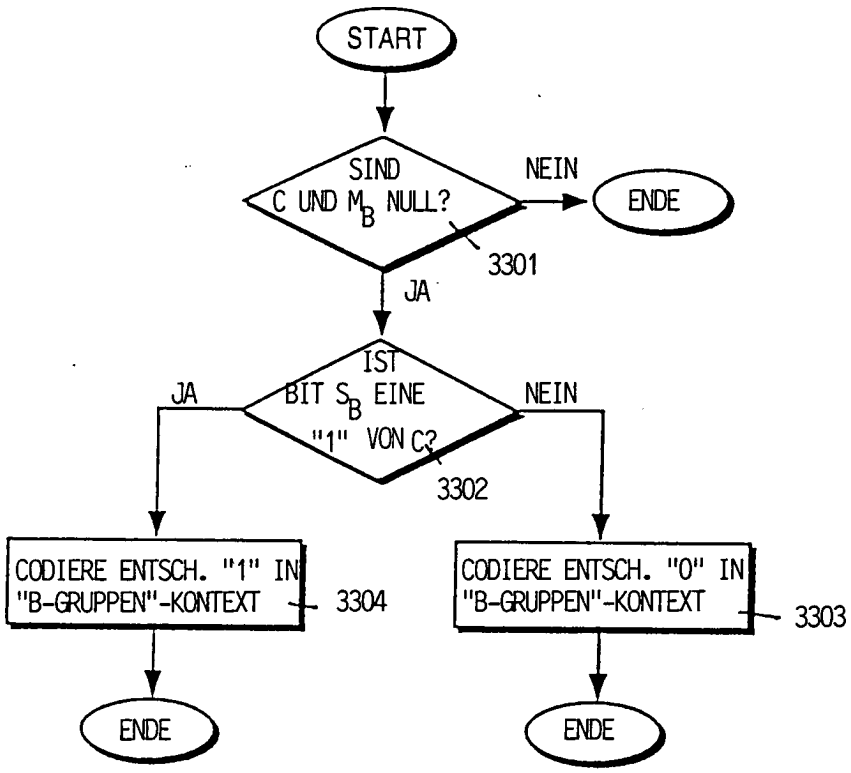

START
SIND
C UND M_B NULL?
NEIN
ENDE
3301
JA
IST
BIT S_B EINE
"1" VON C?
JA
NEIN
3302
CODIERE ENTSCH. "1" IN
"B-GRUPPEN"-KONTEXT
3304
CODIERE ENTSCH. "0" IN
"B-GRUPPEN"-KONTEXT
3303
ENDE
ENDE

FIG. 8b

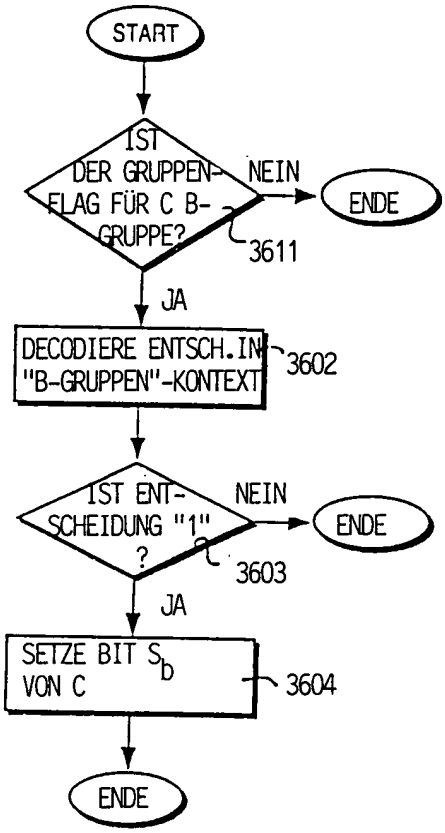

START
IST DER GRUPPEN-FLAG FÜR C B-GRUPPE?
NEIN
ENDE
3611
JA
DECODIERE ENTSCH.IN "B-GRUPPEN"-KONTEXT
3602
IST ENT-SCHEIDUNG "1" ?
NEIN
ENDE
3603
JA
SETZE BIT S_b VON C
3604
ENDE

FIG. 9a

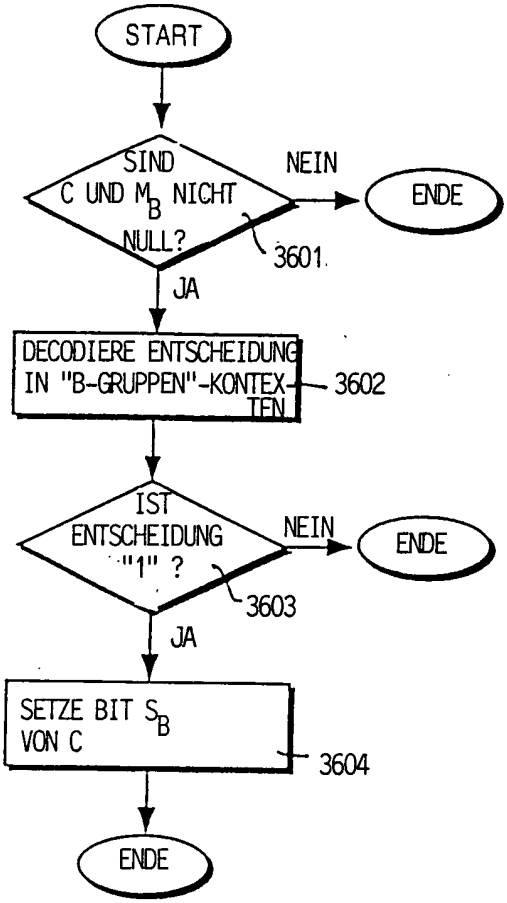

START
SIND C UND M_B NICHT NULL?
NEIN
ENDE
3601
JA
DECODIERE ENTSCHEIDUNG IN "B-GRUPPEN"-KONTEXTEN
3602
IST ENTSCHEIDUNG "1" ?
NEIN
ENDE
3603
JA
SETZE BIT S_B VON C
3604
ENDE

FIG. 9b

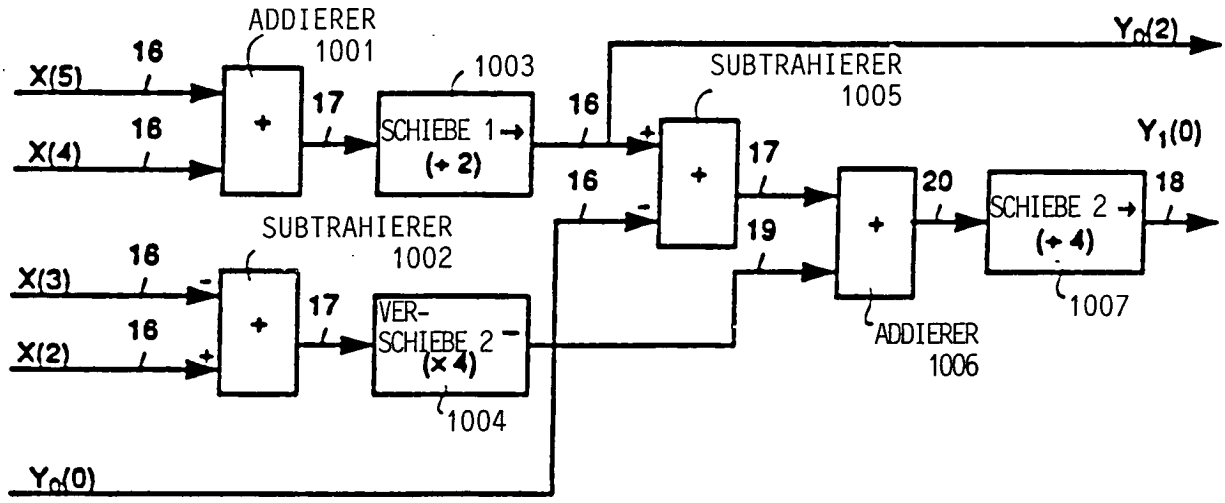

ADDIERER
1001
1003
SUBTRAHIERER
1005
$Y_0(2)$
$Y_1(0)$
X(5)
X(4)
X(3)
X(2)
16
17
SCHIEBE 1 →
(÷ 2)
SUBTRAHIERER
1002
VER-
SCHIEBE 2 ←
(× 4)
1004
20
18
SCHIEBE 2 →
(÷ 4)
1007
ADDIERER
1006
19
$Y_0(0)$

FIG. 10

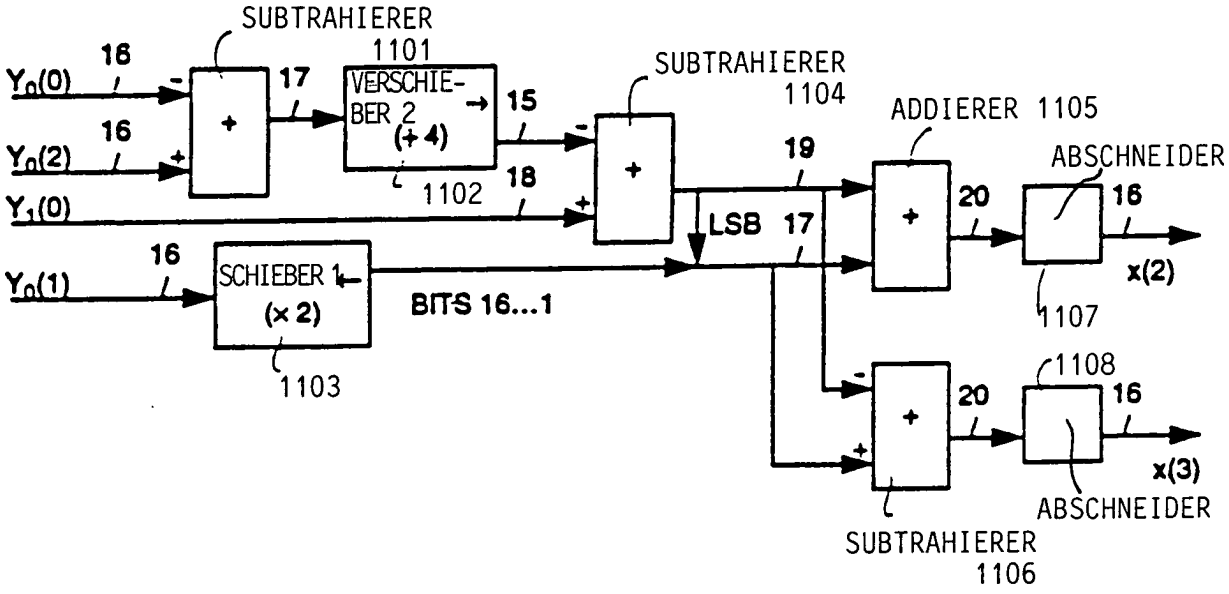

SUBTRAHIERER
1101
16
$Y_0(0)$
16
$Y_0(2)$
$Y_1(0)$
17
VERSCHIE-
BER 2
(+4)
15
1102
18
SUBTRAHIERER
1104
19
ADDIERER 1105
ABSCHNEIDER
20
16
x(2)
1107
LSB 17
16
$Y_0(1)$
SCHIEBER 1
(×2)
1103
BITS 16...1
1108
20
16
x(3)
ABSCHNEIDER
SUBTRAHIERER
1106

FIG. 11

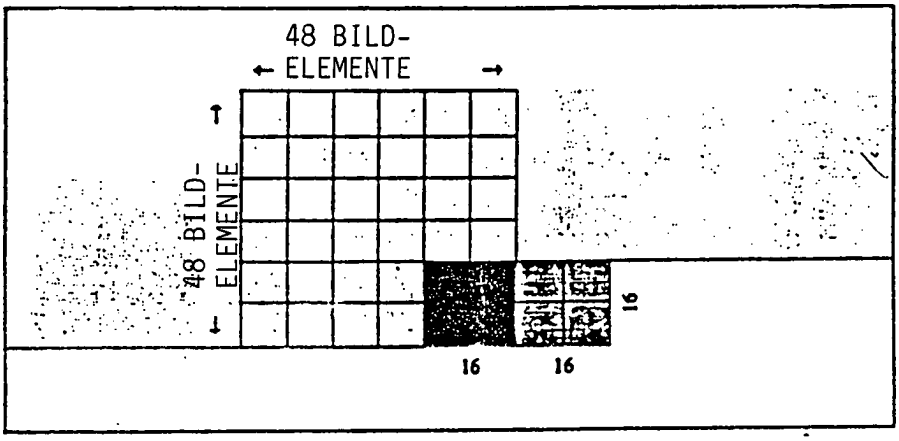


FIG. 12

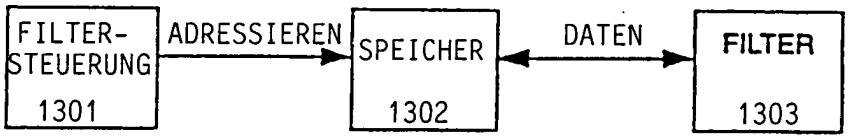

FILTER-
STEUERUNG
1301
ADRESSIEREN
SPEICHER
1302
DATEN
FILTER
1303

FIG. 13

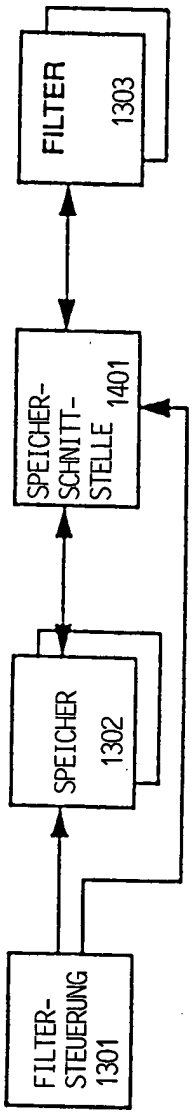

FILTER
1303
SPEICHER-SCHNITT-STELLE
1401
SPEICHER
1302
FILTER-STEUERUNG
1301

FIG. 14

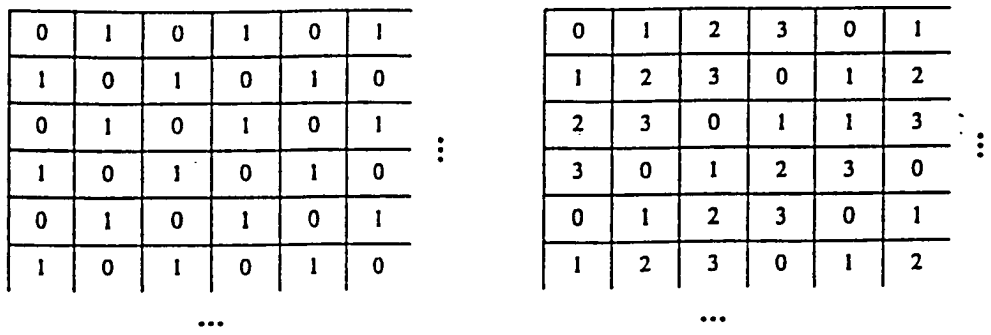
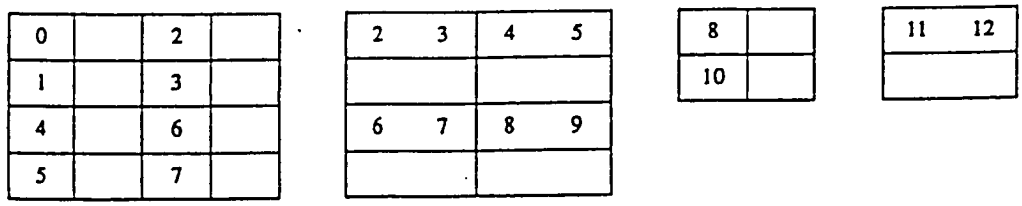


FIG. 15



PEGEL BZW. NIVEAU 1 HORIZONTAL

PEGEL 1 VERTIKAL

PEGEL 0 HORIZ.

PEGEL 0 VERT.

FIG. 16

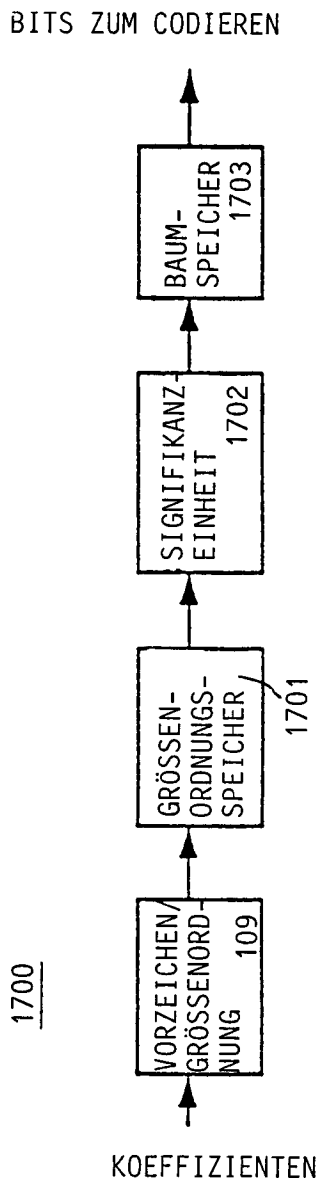

BITS ZUM CODIEREN
BAUM-SPEICHER 1703
SIGNIFIKANZ-EINHEIT 1702
GRÖSSEN-ORDNUNGS-SPEICHER
1701
VORZEICHEN/GRÖSSENORD-NUNG 109
1700
KOEFFIZIENTEN

FIG. 17

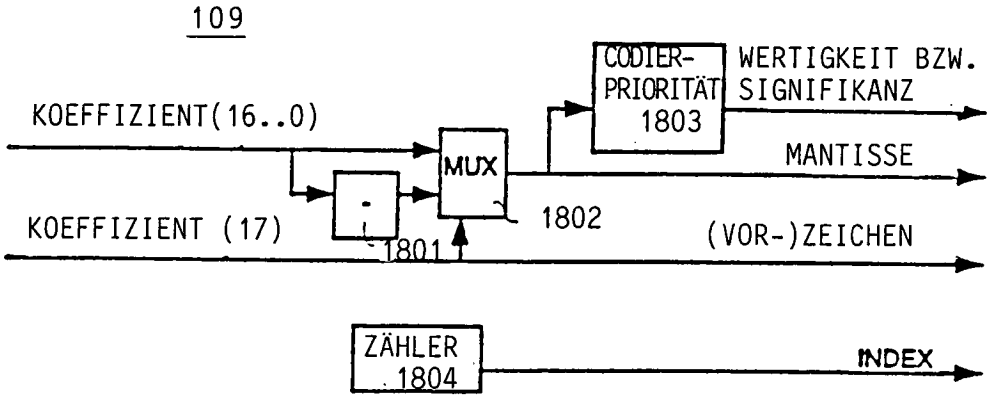

109
KOEFFIZIENT(16..0)
KOEFFIZIENT (17)
MUX
1801
1802
CODIER-PRIORITÄT 1803
WERTIGKEIT BZW. SIGNIFIKANZ
MANTISSE
(VOR-)ZEICHEN
ZÄHLER 1804
INDEX

FIG. 18

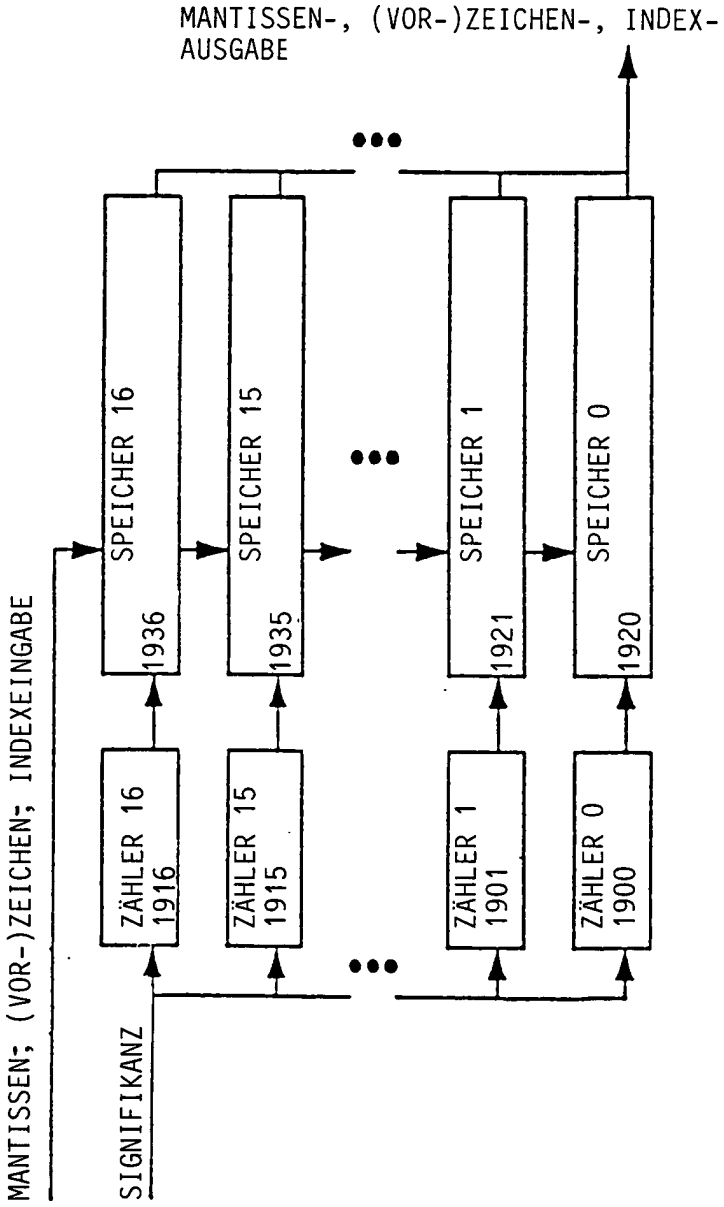
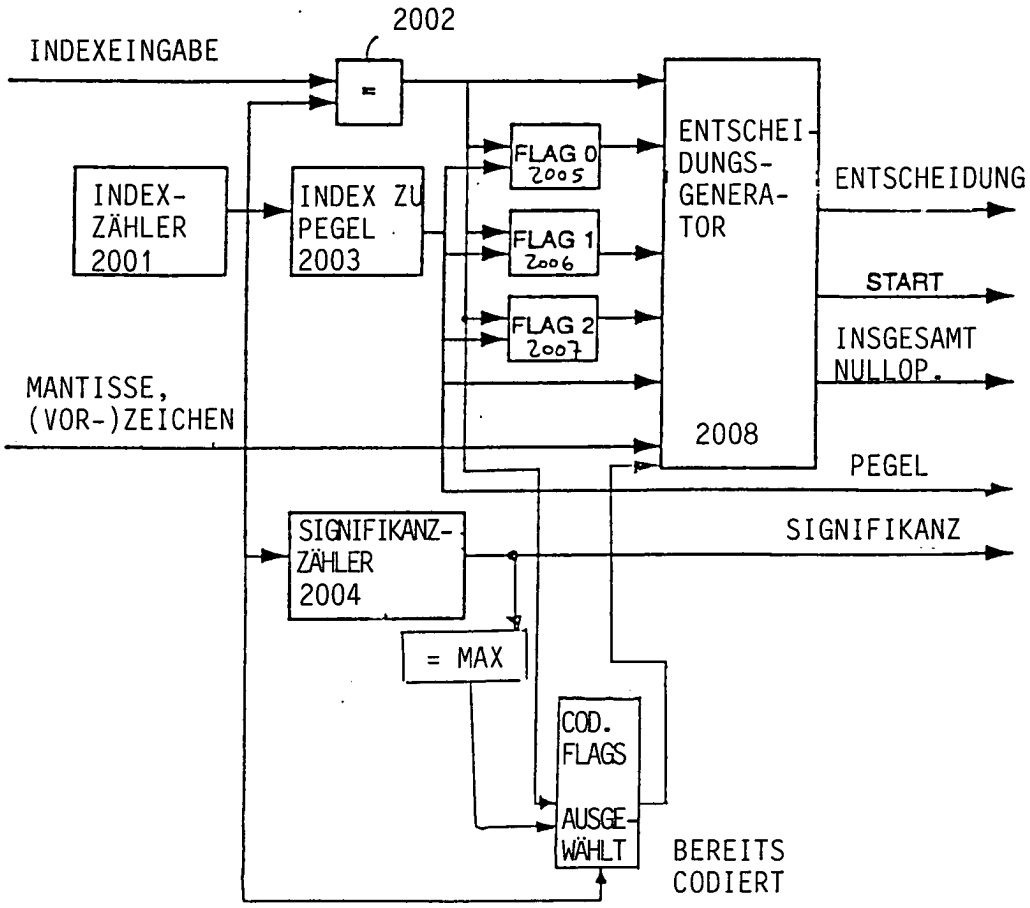

MANTISSEN-, (VOR-)ZEICHEN-, INDEX-AUSGABE
SPEICHER 16
1936
SPEICHER 15
1935
SPEICHER 1
1921
SPEICHER 0
1920
ZÄHLER 16
1916
ZÄHLER 15
1915
ZÄHLER 1
1901
ZÄHLER 0
1900
MANTISSEN; (VOR-)ZEICHEN; INDEXEINGABE
SIGNIFIKANZ

FIG. 19

FIG. 20


2002
INDEXEINGABE
=
INDEX-
ZÄHLER
2001
INDEX ZU
PEGEL
2003
FLAG 0
2005
FLAG 1
2006
FLAG 2
2007
ENTSCHEI-
DUNGS-
GENERA-
TOR
2008
ENTSCHEIDUNG
START
INSGESAMT
NULLOP.
MANTISSE,
(VOR-)ZEICHEN
PEGEL
SIGNIFIKANZ-
ZÄHLER
2004
SIGNIFIKANZ
= MAX
COD.
FLAGS
AUSGE-
WÄHLT
BEREITS
CODIERT

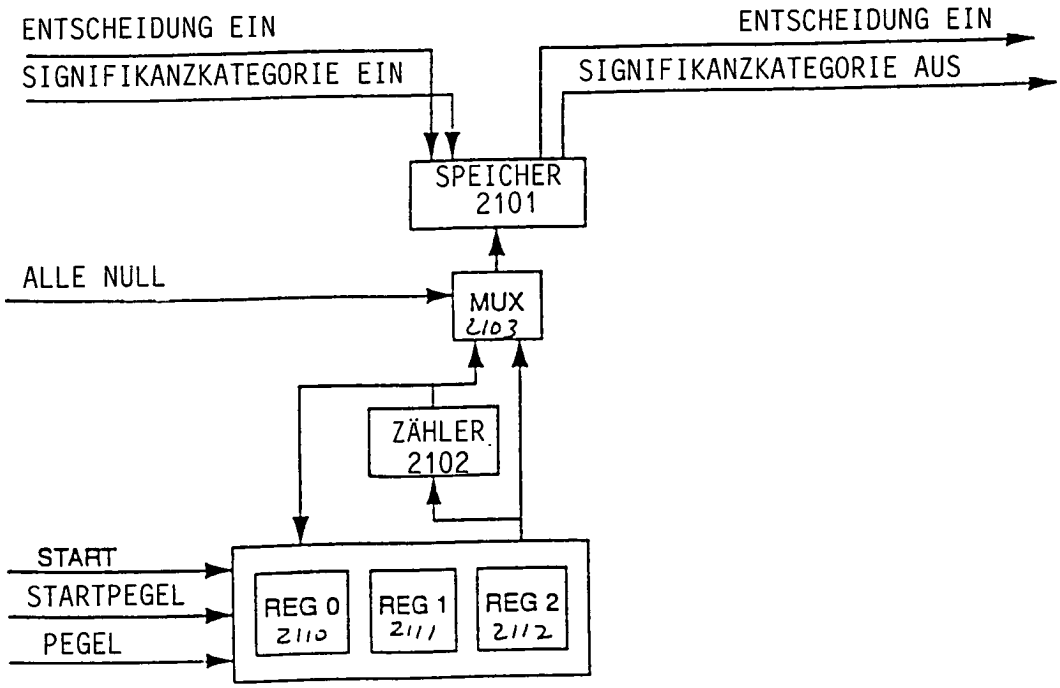

ENTSCHEIDUNG EIN
SIGNIFIKANZKATEGORIE EIN
ENTSCHEIDUNG EIN
SIGNIFIKANZKATEGORIE AUS
SPEICHER
2101
ALLE NULL
MUX
2103
ZÄHLER
2102
START
STARTPEGEL
PEGEL
REG 0
2110
REG 1
2111
REG 2
2112

FIG. 21


VORWÄRTS-
KLEINWEL-
LENFILTER
1000
ANORD-
NUNGS-
EINHEIT
2200
KONTEXT-
MODELL
105

FIG. 22

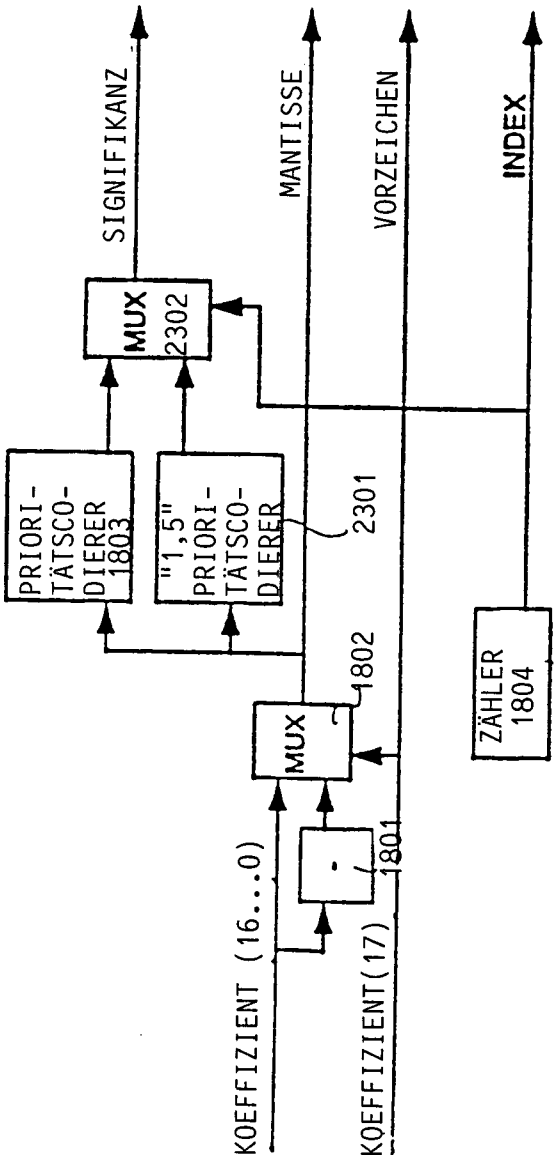

SIGNIFIKANZ
MANTISSE
VORZEICHEN
INDEX
MUX
2302
PRIORI-
TÄTSCO-
DIERER
1803
"1,5"
PRIORI-
TÄTSCO-
DIERER
2301
MUX
1802
ZÄHLER
1804
1801
KOEFFIZIENT (16...0)
KOEFFIZIENT(17)

FIG. 23

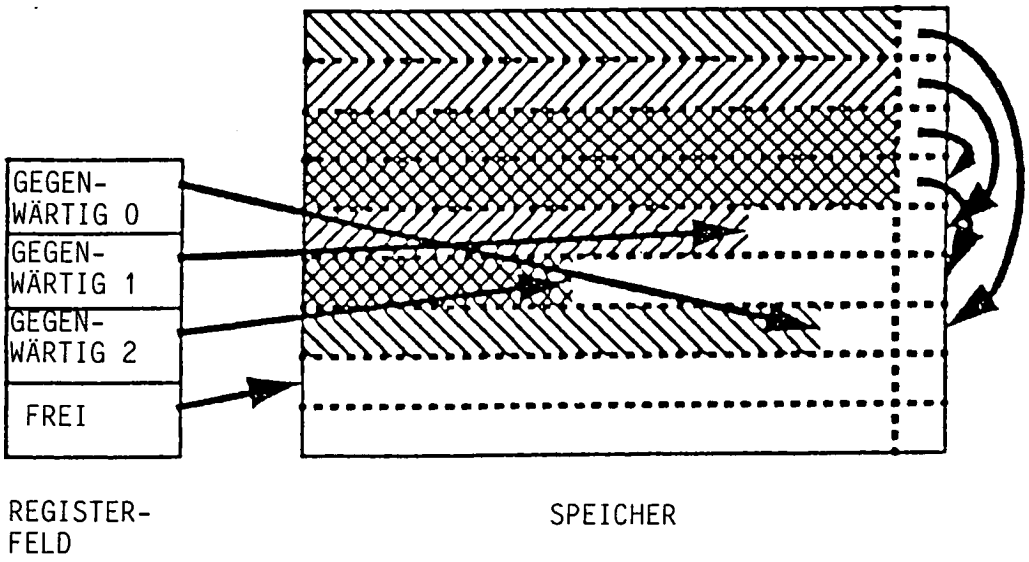

GEGEN-
WÄRTIG 0
GEGEN-
WÄRTIG 1
GEGEN-
WÄRTIG 2
FREI
REGISTER-
FELD
SPEICHER

FIG. 24

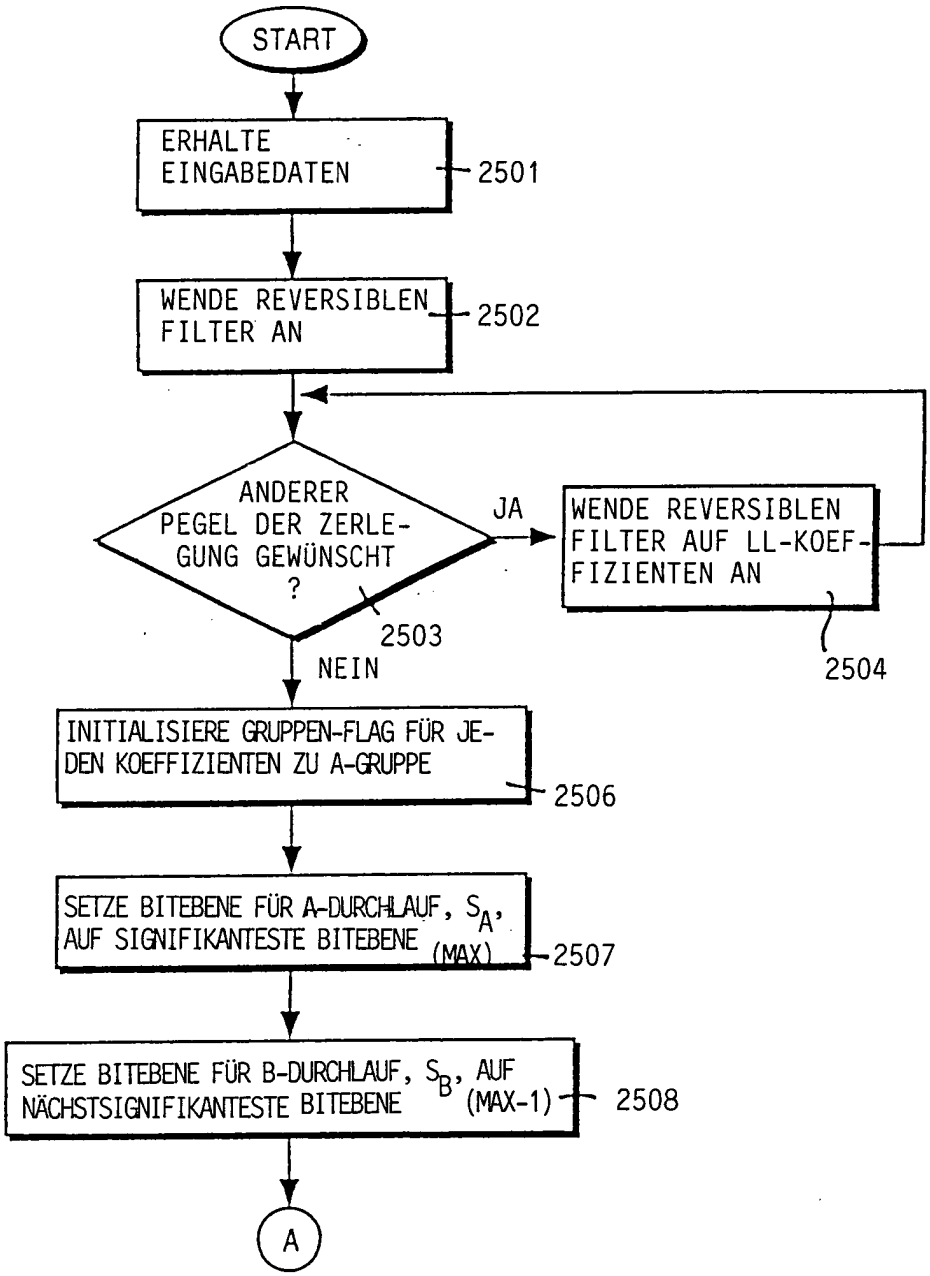

START
ERHALTE EINGABEDATEN
2501
WENDE REVERSIBLEN FILTER AN
2502
ANDERER PEGEL DER ZERLEGUNG GEWÜNSCHT ?
JA
WENDE REVERSIBLEN FILTER AUF LL-KOEFFIZIENTEN AN
2503
2504
NEIN
INITIALISIERE GRUPPEN-FLAG FÜR JEDEN KOEFFIZIENTEN ZU A-GRUPPE
2506
SETZE BITEBENE FÜR A-DURCHLAUF, S_A, AUF SIGNIFIKANTESTE BITEBENE (MAX)
2507
SETZE BITEBENE FÜR B-DURCHLAUF, S_B, AUF NÄCHSTSIGNIFIKANTESTE BITEBENE (MAX-1)
2508
A

FIG. 25a

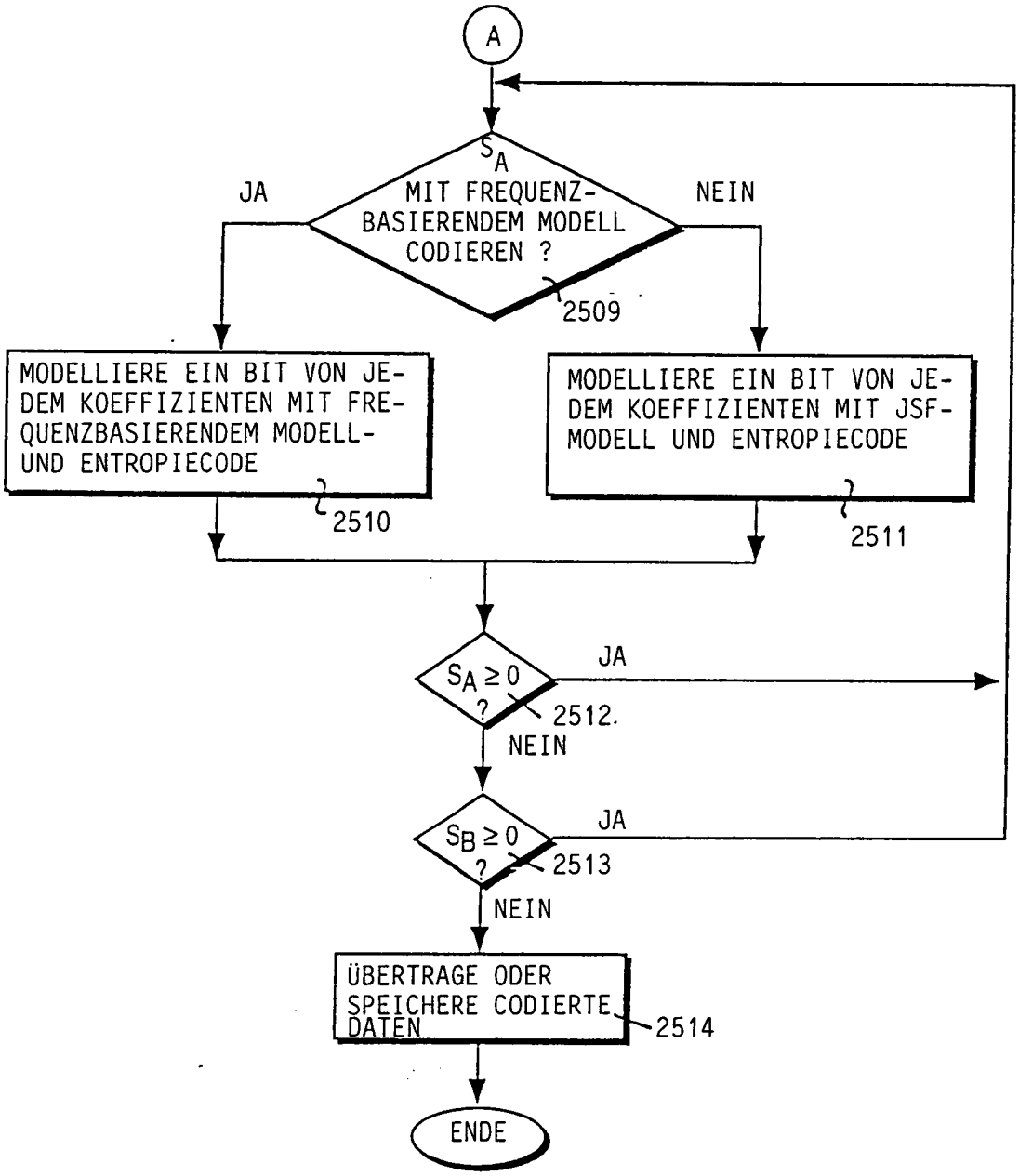

A
S_A MIT FREQUENZ-BASIERENDEM MODELL CODIEREN ?
2509
JA
NEIN
MODELLIERE EIN BIT VON JEDEM KOEFFIZIENTEN MIT FREQUENZBASIERENDEM MODELL- UND ENTROPIECODE
2510
MODELLIERE EIN BIT VON JEDEM KOEFFIZIENTEN MIT JSF-MODELL UND ENTROPIECODE
2511
$S_A \geq 0$?
2512
JA
NEIN
$S_B \geq 0$?
2513
JA
NEIN
ÜBERTRAGE ODER SPEICHERE CODIERTE DATEN
2514
ENDE

FIG. 25b

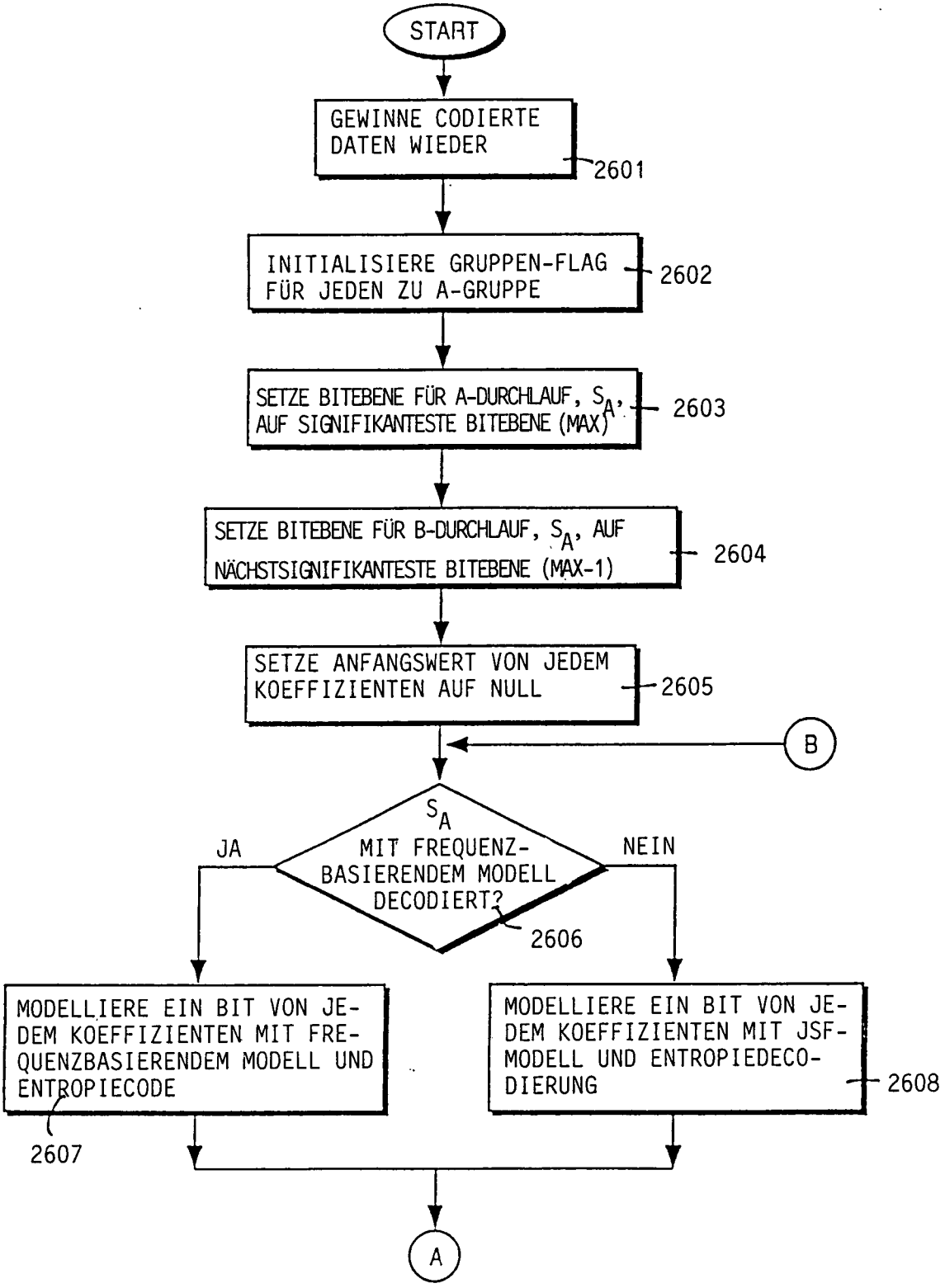

START
GEWINNE CODIERTE DATEN WIEDER
2601
INITIALISIERE GRUPPEN-FLAG FÜR JEDEN ZU A-GRUPPE
2602
SETZE BITEBENE FÜR A-DURCHLAUF, S_A, AUF SIGNIFIKANTESTE BITEBENE (MAX)
2603
SETZE BITEBENE FÜR B-DURCHLAUF, S_A, AUF NÄCHSTSIGNIFIKANTESTE BITEBENE (MAX-1)
2604
SETZE ANFANGSWERT VON JEDEM KOEFFIZIENTEN AUF NULL
2605
B
S_A MIT FREQUENZ-BASIERENDEM MODELL DECODIERT?
2606
JA
NEIN
MODELLIERE EIN BIT VON JEDEM KOEFFIZIENTEN MIT FREQUENZBASIERENDEM MODELL UND ENTROPIECODE
2607
MODELLIERE EIN BIT VON JEDEM KOEFFIZIENTEN MIT JSF-MODELL UND ENTROPIEDECODIERUNG
2608
A

FIG. 26a

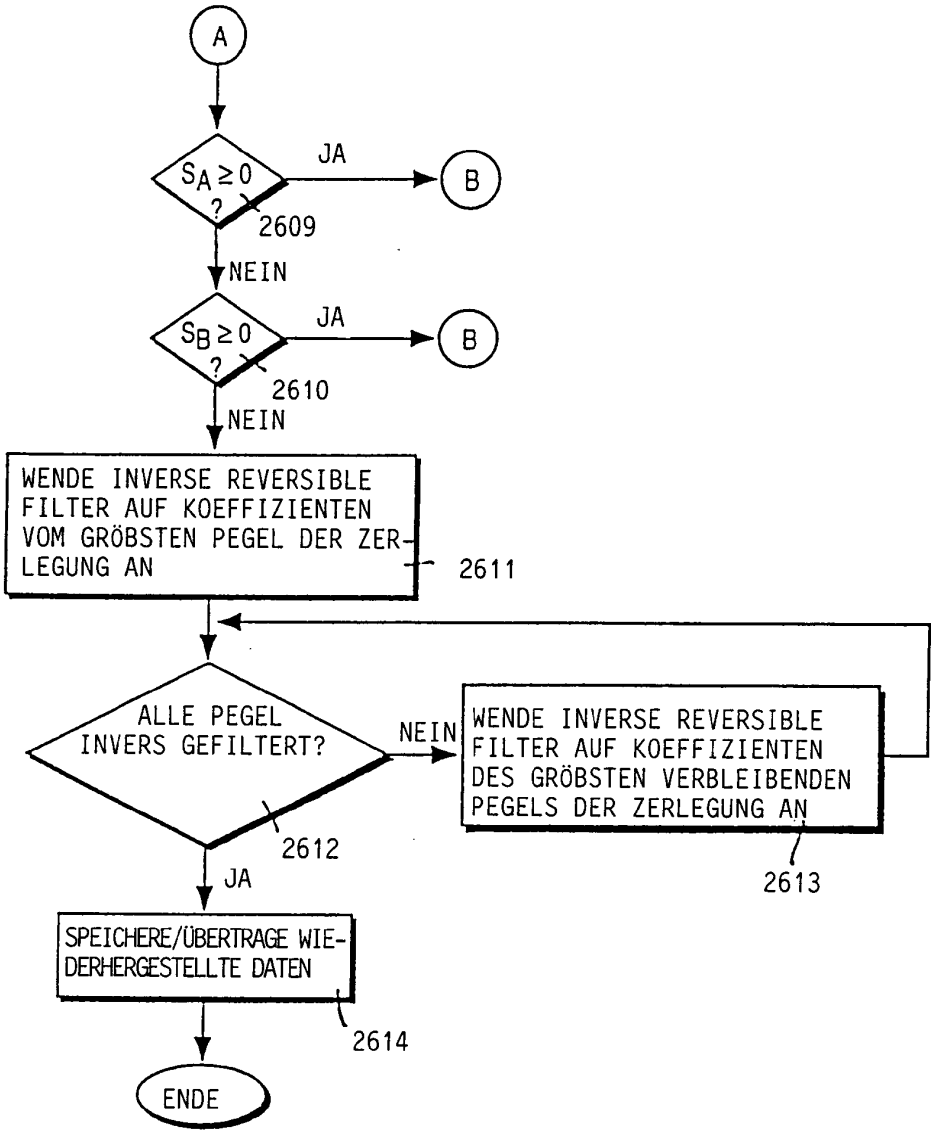

A
$S_A \geq 0$?
JA
B
2609
NEIN
$S_B \geq 0$?
JA
B
2610
NEIN
WENDE INVERSE REVERSIBLE FILTER AUF KOEFFIZIENTEN VOM GRÖBSTEN PEGEL DER ZERLEGUNG AN
2611
ALLE PEGEL INVERS GEFILTERT?
NEIN
WENDE INVERSE REVERSIBLE FILTER AUF KOEFFIZIENTEN DES GRÖBSTEN VERBLEIBENDEN PEGELS DER ZERLEGUNG AN
2612
2613
JA
SPEICHERE/ÜBERTRAGE WIEDERHERGESTELLTE DATEN
2614
ENDE

FIG. 26b

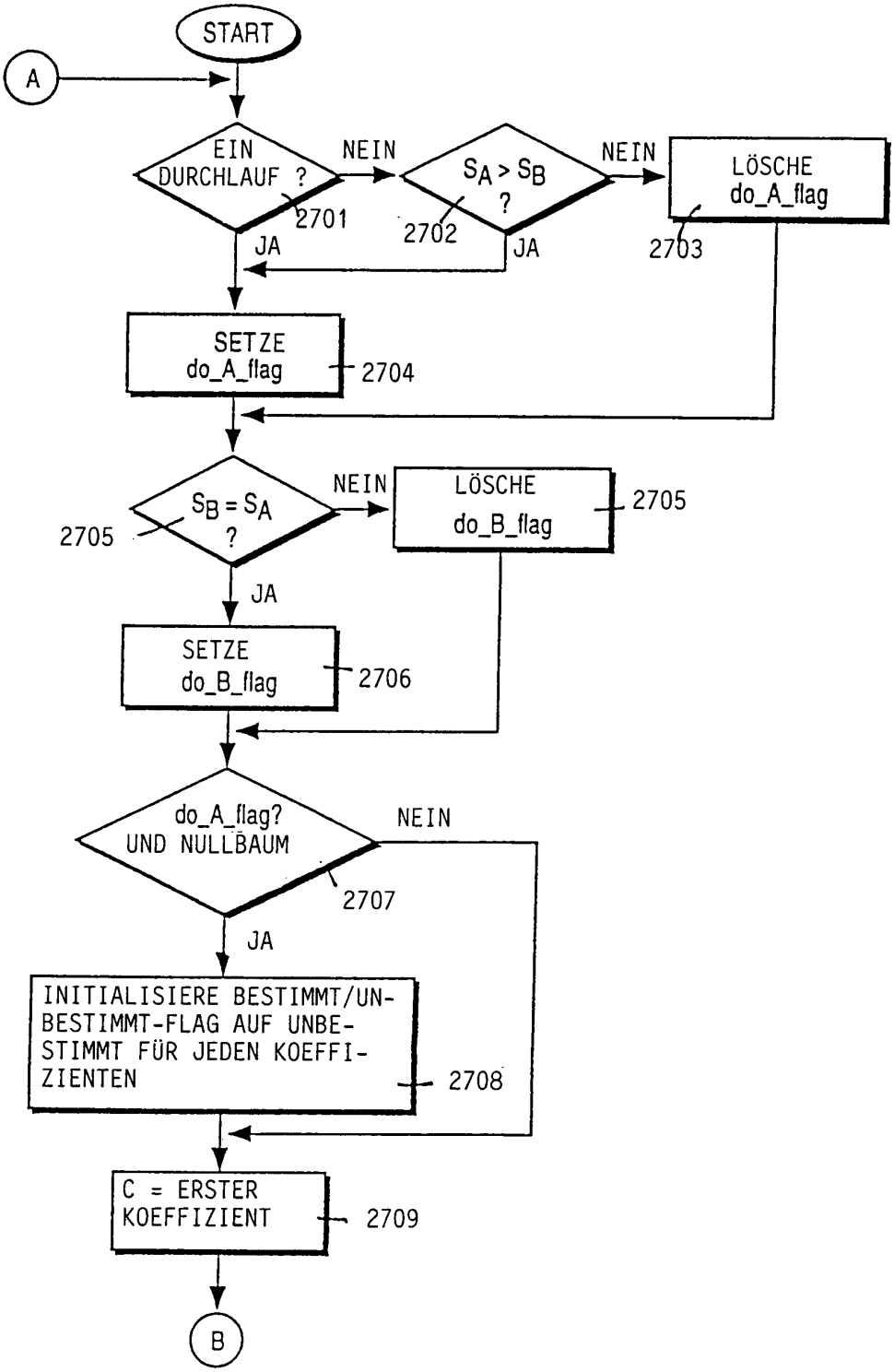

START
A
EIN DURCHLAUF ?
NEIN
2701
JA
$S_A > S_B$?
NEIN
2702
JA
LÖSCHE do_A_flag
2703
SETZE do_A_flag
2704
$S_B = S_A$?
2705
NEIN
LÖSCHE do_B_flag
2705
JA
SETZE do_B_flag
2706
do_A_flag? UND NULLBAUM
NEIN
2707
JA
INITIALISIERE BESTIMMT/UN-BESTIMMT-FLAG AUF UNBE-STIMMT FÜR JEDEN KOEFFI-ZIENTEN
2708
C = ERSTER KOEFFIZIENT
2709
B

FIG. 27a

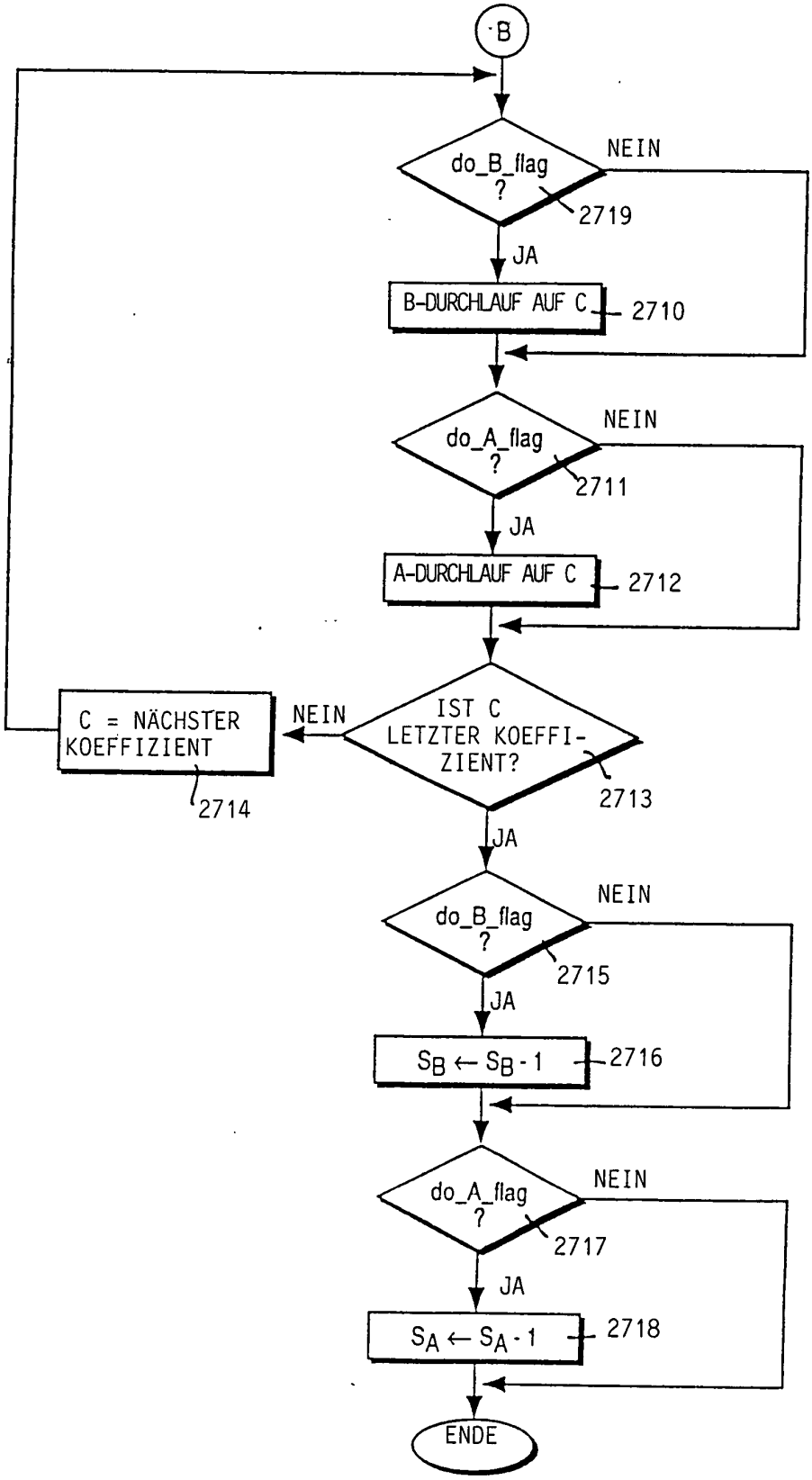

B
do_B_flag ?
NEIN
2719
JA
B-DURCHLAUF AUF C
2710
do_A_flag ?
NEIN
2711
JA
A-DURCHLAUF AUF C
2712
IST C LETZTER KOEFFI-ZIENT?
NEIN
2713
C = NÄCHSTER KOEFFIZIENT
2714
JA
do_B_flag ?
NEIN
2715
JA
SB ← SB - 1
2716
do_A_flag ?
NEIN
2717
JA
SA ← SA - 1
2718
ENDE

FIG. 27b

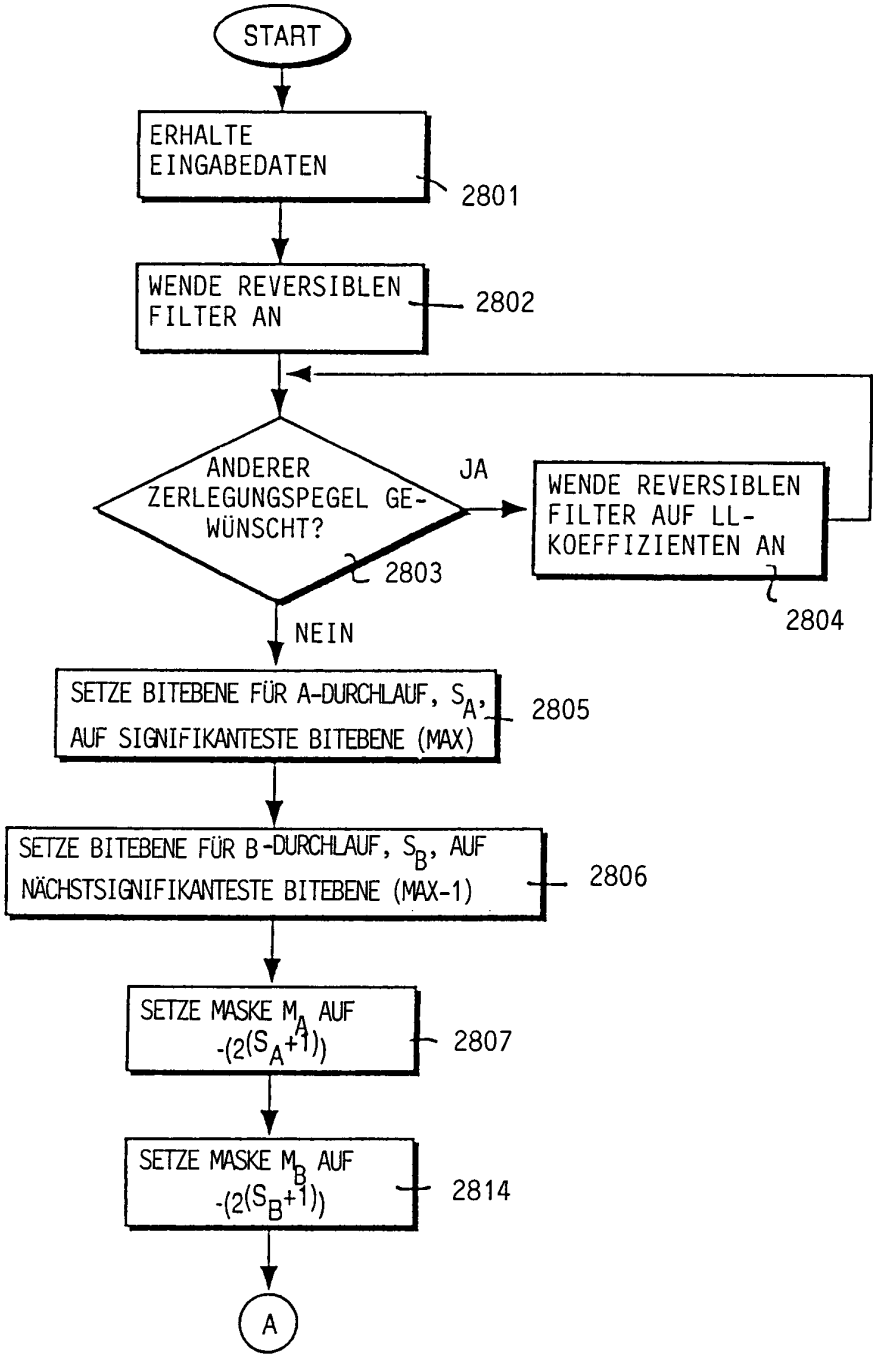

START
ERHALTE EINGABEDATEN
2801
WENDE REVERSIBLEN FILTER AN
2802
ANDERER ZERLEGUNGSPEGEL GEWÜNSCHT?
JA
2803
WENDE REVERSIBLEN FILTER AUF LL-KOEFFIZIENTEN AN
2804
NEIN
SETZE BITEBENE FÜR A-DURCHLAUF, S_A, AUF SIGNIFIKANTESTE BITEBENE (MAX)
2805
SETZE BITEBENE FÜR B-DURCHLAUF, S_B, AUF NÄCHSTSIGNIFIKANTESTE BITEBENE (MAX-1)
2806
SETZE MASKE M_A AUF $-(2^{(S_A+1)})$
2807
SETZE MASKE M_B AUF $-(2^{(S_B+1)})$
2814
A

FIG. 28a

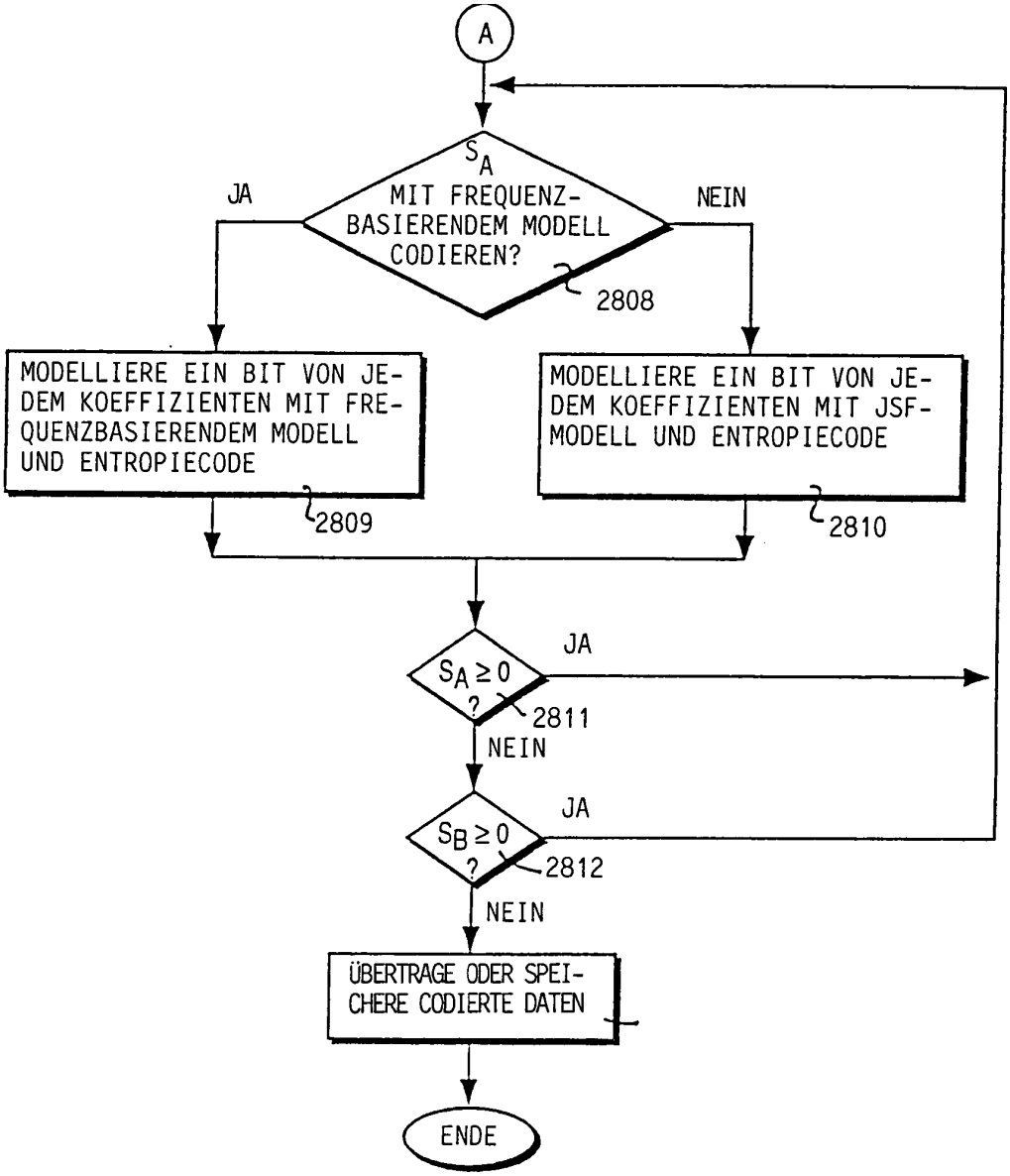

A
S_A MIT FREQUENZ-BASIERENDEM MODELL CODIEREN?
2808
JA
NEIN
MODELLIERE EIN BIT VON JEDEM KOEFFIZIENTEN MIT FREQUENZBASIERENDEM MODELL UND ENTROPIECODE
2809
MODELLIERE EIN BIT VON JEDEM KOEFFIZIENTEN MIT JSF-MODELL UND ENTROPIECODE
2810
$S_A \geq 0$?
2811
JA
NEIN
$S_B \geq 0$?
2812
JA
NEIN
ÜBERTRAGE ODER SPEICHERE CODIERTE DATEN
ENDE

FIG. 28b

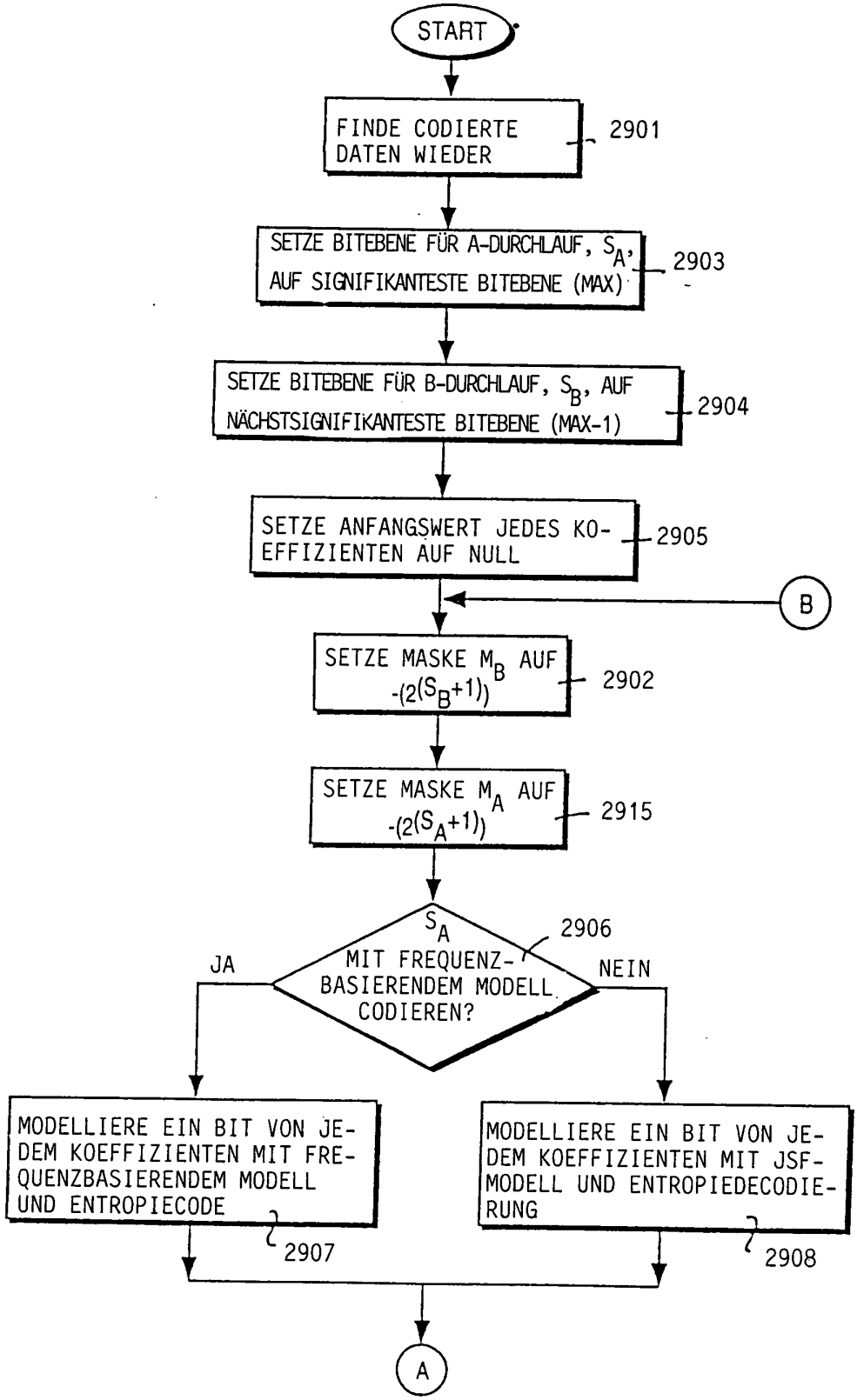

START
FINDE CODIERTE DATEN WIEDER
2901
SETZE BITEBENE FÜR A-DURCHLAUF, S_A, AUF SIGNIFIKANTESTE BITEBENE (MAX)
2903
SETZE BITEBENE FÜR B-DURCHLAUF, S_B, AUF NÄCHSTSIGNIFIKANTESTE BITEBENE (MAX-1)
2904
SETZE ANFANGSWERT JEDES KOEFFIZIENTEN AUF NULL
2905
B
SETZE MASKE M_B AUF $-(2^{(S_B+1)})$
2902
SETZE MASKE M_A AUF $-(2^{(S_A+1)})$
2915
S_A MIT FREQUENZBASIERENDEM MODELL CODIEREN?
2906
JA
NEIN
MODELLIERE EIN BIT VON JEDEM KOEFFIZIENTEN MIT FREQUENZBASIERENDEM MODELL UND ENTROPIECODE
2907
MODELLIERE EIN BIT VON JEDEM KOEFFIZIENTEN MIT JSF-MODELL UND ENTROPIEDECODIERUNG
2908
A

FIG. 29a

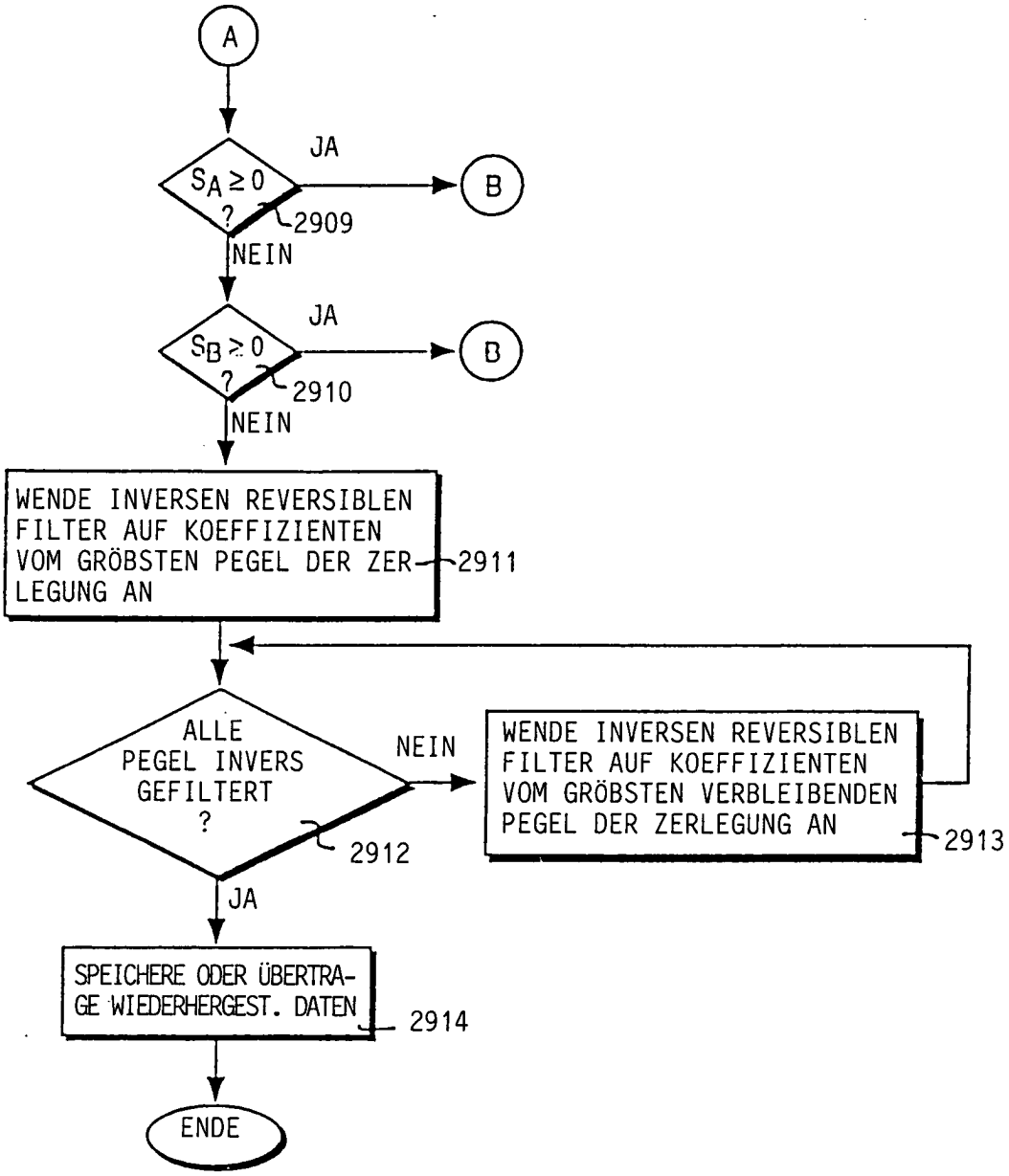

A
JA
$S_A \geq 0$?
2909
B
NEIN
JA
$S_B \geq 0$?
2910
B
NEIN
WENDE INVERSEN REVERSIBLEN FILTER AUF KOEFFIZIENTEN VOM GRÖBSTEN PEGEL DER ZERLEGUNG AN
2911
ALLE PEGEL INVERS GEFILTERT ?
NEIN
2912
WENDE INVERSEN REVERSIBLEN FILTER AUF KOEFFIZIENTEN VOM GRÖBSTEN VERBLEIBENDEN PEGEL DER ZERLEGUNG AN
2913
JA
SPEICHERE ODER ÜBERTRAGE WIEDERHERGEST. DATEN
2914
ENDE

FIG. 29b

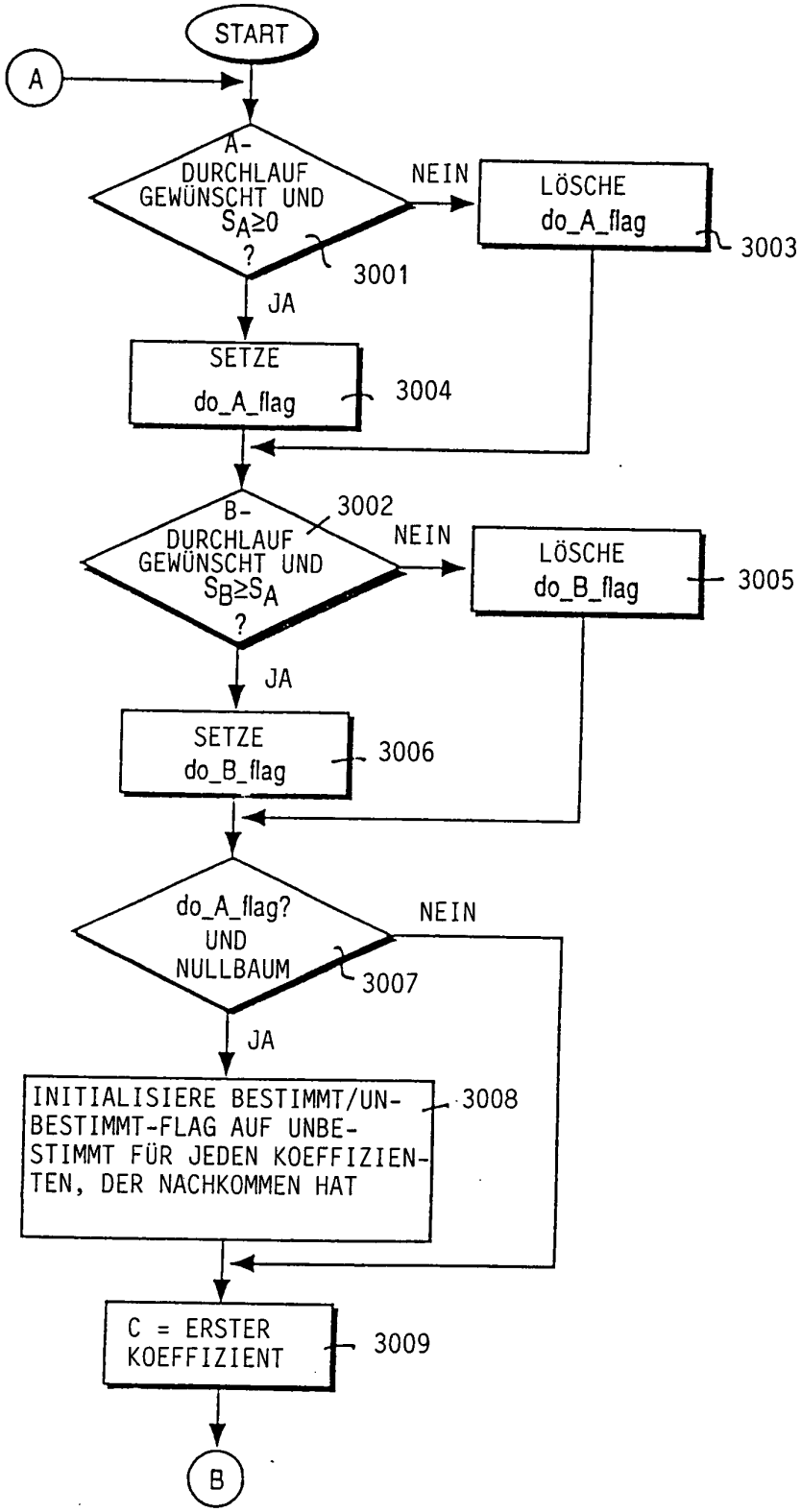

START
A
A-DURCHLAUF GEWÜNSCHT UND $S_A \geq 0$?
3001
NEIN
LÖSCHE do_A_flag
3003
JA
SETZE do_A_flag
3004
B-DURCHLAUF GEWÜNSCHT UND $S_B \geq S_A$?
3002
NEIN
LÖSCHE do_B_flag
3005
JA
SETZE do_B_flag
3006
do_A_flag? UND NULLBAUM
3007
NEIN
JA
INITIALISIERE BESTIMMT/UNBESTIMMT-FLAG AUF UNBESTIMMT FÜR JEDEN KOEFFIZIENTEN, DER NACHKOMMEN HAT
3008
C = ERSTER KOEFFIZIENT
3009
B

FIG. 30a

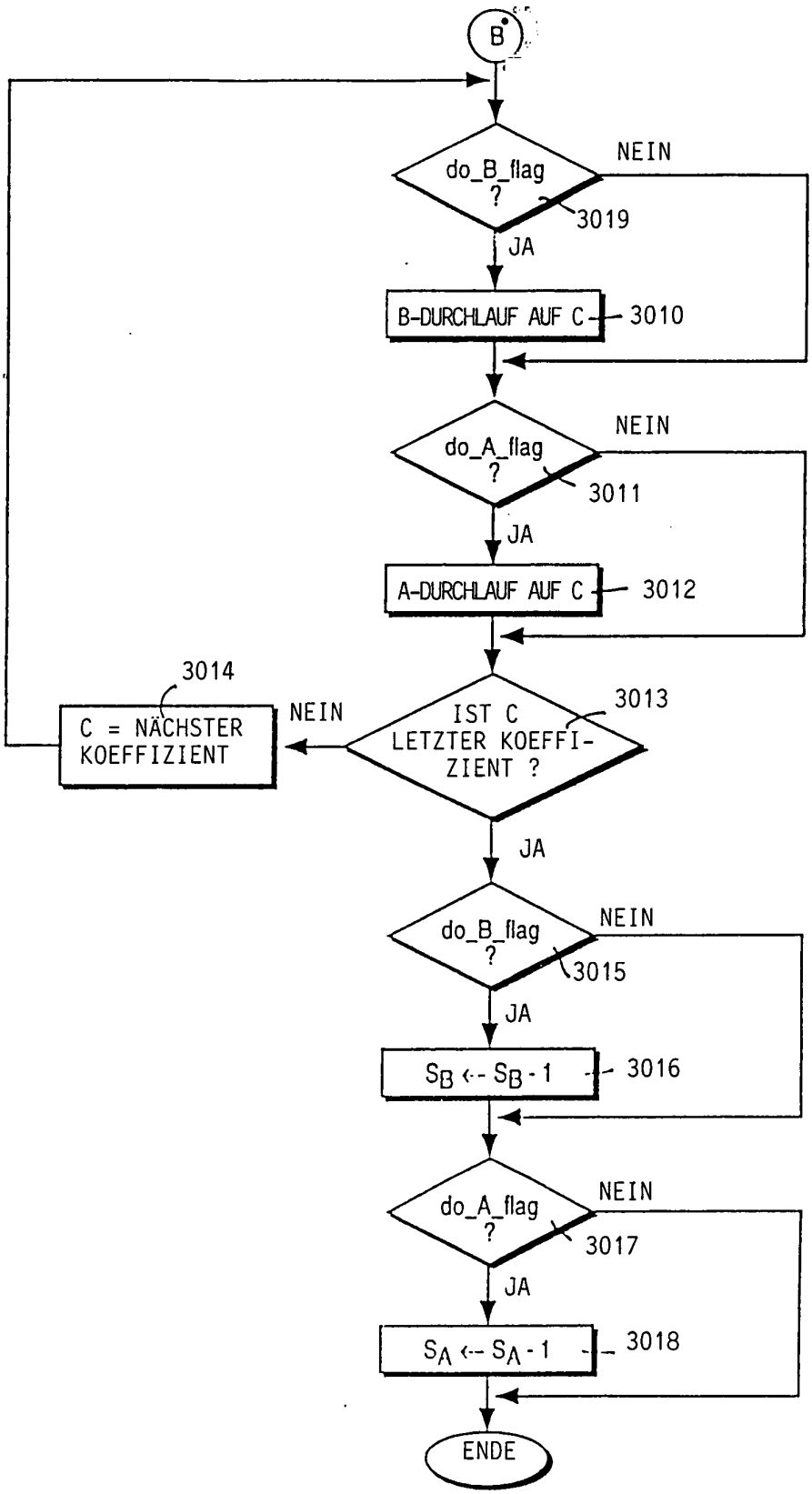

B
do_B_flag ?
NEIN
3019
JA
B-DURCHLAUF AUF C
3010
do_A_flag ?
NEIN
3011
JA
A-DURCHLAUF AUF C
3012
IST C LETZTER KOEFFI-ZIENT ?
3013
NEIN
C = NÄCHSTER KOEFFIZIENT
3014
JA
do_B_flag ?
NEIN
3015
JA
SB <-- SB - 1
3016
do_A_flag ?
NEIN
3017
JA
SA <-- SA - 1
3018
ENDE

FIG. 30b

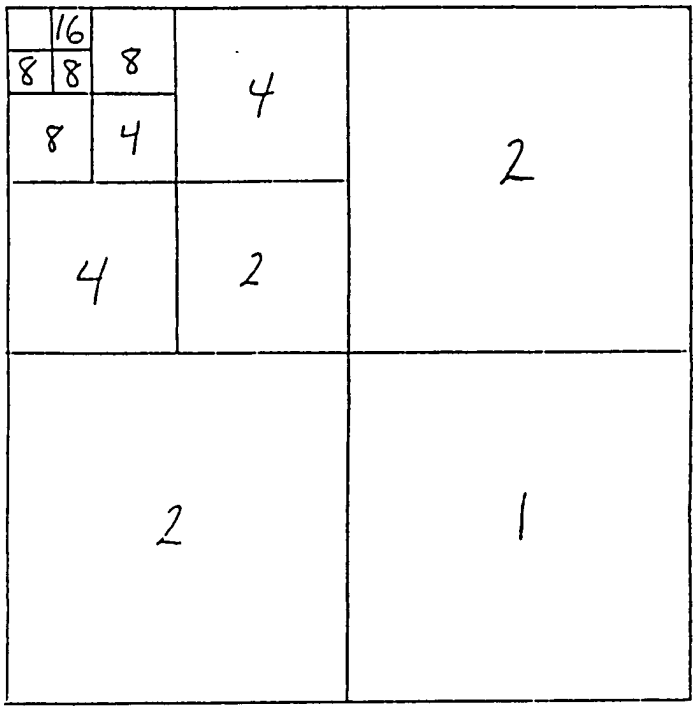


FIG. 31