(19)日本国特許庁(JP)　(12)特許公報（Ｂ２）　(11)特許番号

特許第３０４５１９７号
（Ｐ３０４５１９７）

(45)発行日　平成12年5月29日(2000.5.29)　(24)登録日　平成12年3月17日(2000.3.17)

(51)Int.Cl.⁷	識別記号	ＦＩ	
Ｈ０３Ｍ	7/30	Ｈ０３Ｍ	7/30　Ｂ
Ｇ１０Ｌ	19/12	Ｇ１０Ｌ	9/14　Ｓ

請求項の数３　（全１７頁）

(21)出願番号　特願平3-188956

(22)出願日　平成3年7月29日(1991.7.29)

(65)公開番号　特開平5-37397
(43)公開日　平成5年2月12日(1993.2.12)
審査請求日　平成9年9月11日(1997.9.11)

(73)特許権者　000004226
日本電信電話株式会社
東京都千代田区大手町二丁目3番1号
(74)上記１名の代理人　100083806
弁理士　三好　秀和　（外1名）
(73)特許権者　392026693
エヌ・ティ・ティ移動通信網株式会社
東京都港区虎ノ門二丁目10番1号
(74)上記１名の代理人　100083806
弁理士　三好　秀和
(72)発明者　大室　仲
東京都千代田区内幸町一丁目1番6号　日本電信電話株式会社内
(72)発明者　守谷　健弘
東京都千代田区内幸町一丁目1番6号　日本電信電話株式会社内

最終頁に続く

(54)【発明の名称】ベクトル量子化器の符号帳設計方法

(57)【特許請求の範囲】
【請求項１】　学習用データを用いてベクトル量子化器の最適符号帳を設計するベクトル量子化器の符号帳設計方法であって、学習ベクトルを量子化して符号誤りのある伝送路を伝送すると仮定した場合に符号誤りをも考慮して各学習ベクトルを符号帳の中のそれぞれどの代表ベクトルに帰属させるかを更新し、前記仮定のもとにおける学習ベクトルと受信側の再生ベクトルとの歪みの平均値の総和の極小化に基づいて代表ベクトルを更新し、前記仮定のもとにおいて受信側の再生ベクトルと学習ベクトルとの歪みの平均値を反映する符号帳の評価関数を小さくするように符号帳インデックスの付け換えを行い、上記処理を交互に繰り返し用いることにより符号誤りのある場合の最適な符号帳を設計することを特徴とするベクトル量子化器の符号帳設計方法。

【請求項２】　代表ベクトルの初期数と代表ベクトルの初期値を決定し、前記学習ベクトルの帰属を更新する処理と前記代表ベクトルを更新する処理とを交互に繰り返し使用する処理と、前記符号帳インデックスの付け換えを行う処理を用いたインデックスの付け換えを行う処理と、前記代表ベクトルの数を増加させる処理を繰り返し行うことにより、所望の数の代表ベクトルを決定することを特徴とする請求項１記載のベクトル量子化器の符号帳設計方法。
【請求項３】　前記学習ベクトルの帰属を更新する処理と、前記代表ベクトルを更新する処理を交互に繰り返し使用する処理と、前記符号帳インデックスの付け換えを行う処理を用いたインデックスの付け換え処理の２つの処理を交互に繰り返し行った後、代表ベクトルの数を増加させる処理を行うことを特徴とする請求項１または２

記載のベクトル量子化器の符号帳設計方法。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】本発明は、音声または画像信号を符号化するためのベクトル量子化器を設計する方法に関する。

【０００２】

【従来の技術】音声信号や画像信号を符号化するための高能率な量子化法のひとつとして、ベクトル量子化やマトリクス量子化（以後、これらをベクトル量子化という言葉で代表させる）が知られている。ベクトル量子化においては、与えられたビット数に応じて、符号帳と呼ばれる部分に代表ベクトルパターンを蓄えておくが、この代表パターンをどのように決めるかによって量子化器の性能が左右される。

【０００３】ベクトル量子化における符号帳の設計手法としてＬＢＧアルゴリズムが知られている。この手法はベクトル量子化すべきベクトル空間の分布をよく表すような分布を持つ学習用ベクトルデータ系列を用意し、これらの学習用データを量子化したときの歪みの総和を極小にするように符号帳を設計する手法である。

【０００４】ＬＢＧアルゴリズムは、代表ベクトルの初期数を１とし、最適な代表ベクトルを求める操作と代表ベクトルの数を２倍に増加させる操作を交互に行って、目的とする数の代表ベクトルを得る。

【０００５】最適な代表ベクトルを求める操作は、まず学習用データがどの代表ベクトルに帰属するかを更新する処理を行い、その帰属状態において、歪みの総和が極小となるように、代表ベクトルを更新する。上記帰属更新処理と、代表ベクトルの更新処理を収束するまで交互に繰り返し行うことにより、代表ベクトルは次第に最適なものに近づいていく。この操作は Lloyd-Maxまたはｋ平均アルゴリズムと呼ばれる。

【０００６】ＬＢＧアルゴリズムは、学習ベクトルの集合をいくつかの部分集合に分割する操作を意味し、上記部分集合をクラスタ、分割操作をクラスタリングと呼ぶ。ＬＢＧアルゴリズムの詳細は文献 Y.Linde.A.Buzo. R.M.Gray著”An Algorithm forVector Quantizer Design”. IEEE Trans.Commun.COM-28 p.p.84-95,1980 に記載されている。

【０００７】

【発明が解決しようとする課題】ベクトル量子化では、代表ベクトルと実際に伝送される符号との間に何等関係がないため、伝送路において符号誤りが生じると著しく品質劣化が生じることが指摘されている。

【０００８】この問題を解決する一手法として、上記ＬＢＧアルゴリズムの歪み尺度に、符号誤りを考慮した尺度を用い、この尺度の総和が最小になるように学習すれば、符号誤りが生じても被害が比較的少ないと報告されている。

【０００９】文献：熊沢、笠原、滑川

「通信路誤りを考慮したベクトル量子化器の構成」電子通信学会論文誌　Vol.J67-B No.1 1984

しかし実際には、上記手法によっても、”最適”な解は得られず、局所最小解に陥って期待したほどに符号誤りに強い構造に設計できない。この理由は、ＬＢＧアルゴリズムが除々に最適解に近づける漸近的な手法であるため、容易に局所最小解に陥り、いったん陥るとそこから抜け出す手段を有しないからである。

【００１０】本発明は、上記に鑑みてなされたもので、その目的とするところは、符号誤りに強く、より高性能な符号帳を設計し得るベクトル量子化器の符号帳設計方法を提供することにある。

【００１１】

【課題を解決するための手段】上記目的を達成するため、本発明のベクトル量子化器の符号帳設計方法は、学習用データを用いてベクトル量子化器の最適符号帳を設計するベクトル量子化器の符号帳設計方法であって、学習ベクトルを量子化して符号誤りのある伝送路を伝送すると仮定した場合に符号誤りをも考慮して各学習ベクトルを符号帳の中のそれぞれどの代表ベクトルに帰属させるかを更新し、前記仮定のもとにおける学習ベクトルと受信側の再生ベクトルとの歪みの平均値の総和の極小化に基づいて代表ベクトルを更新し、前記仮定のもとにおいて受信側の再生ベクトルと学習ベクトルとの歪みの平均値を反映する符号帳の評価関数を小さくするように符号帳インデックスの付け換えを行う、上記処理を交互に繰り返し用いることにより符号誤りのある場合の最適な符号帳を設計することを要旨とする。

【００１２】また、本発明では、より”最適”に近い解、即ちより符号誤りに強い符号帳を得るために、前述のＬＢＧアルゴリズムにインデックス付けかえ操作を導入する。インデックス付けかえ操作は、ＬＢＧアルゴリズムにおいて、上記局所最小解から抜け出し、「真」の解に近づける手段を与える。言い換えると、インデックス付けかえ操作は、ダイナミックに状態を変える働きをする。

【００１３】ＬＢＧアルゴリズムにおいて、既に代表ベクトル（クラスタ）数N_iの符号帳が得られていると仮定する。もし、N_iが目的とする数に達していれば、処理を終了する。達していない場合には各クラスタの分割操作を行う。

【００１４】次に、分割された各クラスタの代表ベクトルを初期値として、各学習データの帰属更新処理と、代表ベクトルの更新処理を収束するまで交互に繰り返し行う（Lloyd-Max アルゴリズム）。このときの歪み尺度には、符号誤りを考慮した尺度を用いる。上記処理が収束すると、それ以上続けても歪みの総和は小さくならない。

【００１５】ここで、符号帳インデックスの付けかえ操

5

作を行う。符号誤りを考慮した歪み尺度とは、例えば、符号誤りがある伝送路を伝送したと仮定した場合の、受信側における歪みの平均値を表す。従って、前記 Lloyd-Maxアルゴリズムによっては、もはや歪みの総和を小さくできない場合でも、インデックスの付けかえ操作、例えば任意の２つのインデックスの交換によって、歪みの総和を小さくすることができる。一例として、任意の２つのインデックスを交換して、符号誤りを考慮した歪みの総和が小さくなれば交換し、小さくならなければ別のインデックスの組を交換してみる。この操作を繰り返し行い、どのインデックスの組を交換しても歪みの総和が小さくならないようになるまで続ける。ここで、インデックスの交換は、２つの間で交換してもよいし、３つ以上で交換してもよい。

【００１６】この後のクラスタの分割操作に戻っても良いが、インデックスの付けかえ操作によって、 Lloyd-Maxの局所最小解を抜け出していると考えられるので、再度 Lloyd-Max、即ち、各学習データの帰属更新処理と、代表ベクトルの更新処理を収束するまで交互に繰り返し行う。

【００１７】この後、クラスタの分割操作に戻る。クラスタの分割処理と、上記歪み最小化操作を繰り返し、目的とする数の代表ベクトルが得られた時点で終了する。

【００１８】

【作用】本発明のベクトル量子化器の符号帳設計方法では、学習ベクトルを量子化して符号誤りのある伝送路を伝送すると仮定した場合に符号誤りをも考慮して各学習ベクトルを符号帳の中のそれぞれどの代表ベクトルに帰属させるかを更新し、この仮定のもとで学習ベクトルと受信側の再生ベクトルとの歪みの平均値の総和の極小化に基づいて代表ベクトルを更新し、受信側の再生ベクトルと学習ベクトルとの歪みの平均値を反映する符号帳の評価関数を小さくするように符号帳インデックスの付け換えを行い、上記処理を交互に繰り返し用いることにより符号誤りのある場合の最適な符号帳を設計している。

【００１９】既に報告されているＬＢＧアルゴリズムに誤りを考慮した歪み尺度を導入しただけでは、局所最小解に陥って、十分に符号誤りに強い符号帳を設計できないことが多いが、本発明の手法を用いることによって局所最小解に陥る危険を減らし、より高性能な符号帳を設計することができる。

【００２０】また、インデックス付けかえ操作は、符号誤りを考慮した距離尺度を用いたＬＢＧ法で、最終的にできあがった目的数の代表ベクトルを持つ符号帳に対して１回だけ適用することもでき、一見この方が計算量が少なくて済むように思われる。しかし実際には、このようなインデックス付けかえ操作の最適化は非常に難しい。なぜなら、インデックス付けかえ操作そのものが、容易に局所最小解に陥ってしまうからである。従って、本発明に比べてはるかに多くの計算時間を費やしても、

6

ＬＢＧ法によって作成した符号帳に比べ、ほとんど性能は向上しない。

【００２１】一方、本発明による方法では、ＬＢＧ法とインデックス付けかえ操作が相互に影響しあい、わずかな計算量の増加で、容易に最適に近い状態へと導くことができる。

【００２２】

【実施例】以下、本発明の一実施例を図面を参照して説明する。図１は本発明によって設計した符号帳を用いた、ベクトル量子化法の構成例を示したものである。送信側では、符号帳１に蓄えられたM個の代表ベクトルｃ（ｒ）と入力ベクトルｘとの歪みを歪み算出部２で算出し、歪みが最小となるインデックスｒを歪み判定部３で判定し、符号化部４で符号化して伝送する。一方、受信側では、符号化されて伝送されてくるインデックスｒを復号化部５で受信して、インデックスｒ’として復号化し、符号帳６を介して対応する代表ベクトルをｃ（ｒ’）として出力する。ここで、伝送路に符号誤りがない、すなわちｒ＝ｒ’ならば問題はないが、例えば無線通信のように品質の悪い伝送路の場合には、符号ｒのうちの１ないし２ビットが反転して検波されることも少なくない。そこで、仮にｒの１ないし２ビットが反転してもｃ（ｒ）とｃ（ｒ’）の間に大きな差が生じないように符号帳を設計すれば、符号誤りによる品質の劣化を最小限に抑えることができる。

【００２３】図２及び図３は、上記のように、符号誤りが生じても品質劣化が少ないという観点で高性能な符号帳の設計法の一例の概略をフローチャートで示したものである。これに先立ち、学習用のデータを用意する。学習用データは、実際の符号化において符号化されるべきデータが分布する空間を有限のデータ系列で効果的に表すように選択するのがよい。一般的には、学習データの数は多いほうがよい。学習用データの分布が実際に符号化されるデータの分布に比べて著しい偏りがある場合には、本発明によって設計した符号帳の性能が低下することになる。

【００２４】用意した学習用ベクトルデータ系列を

$$\{x_i\}, \qquad i=1, 2, \cdots, N$$

とする。図２においてMはクラスタ、すなわち代表ベクトルの数を表す。初期値としてMをM_oに設定する（ステップ１１０）。通常M_oは１に設定すると非常に都合がよいが、必ずしも１にする必要はない。次に、代表ベクトルの初期値を決定する（ステップ１２０）。M_o＝１の場合には任意の値に設定してよい。なぜなら、M_oのときの代表ベクトルの初期値は全く意味を持たないからである。なお、$M_o \neq 1$のときには、適当な初期値を設定しなければならない。一例として、適当に学習ベクトルの中からピックアップして初期値とする方法がある。次に、ループカウンタ Repを０にリセットする（ステップ１３０）。次にステップ１２０で決定された代表

ベクトルを初期値として、誤りを考慮した歪み尺度による Lloyd-Maxアルゴリズムにより、代表ベクトルを更新する（ステップ１４０）。ステップ１４０の更新操作の詳細を図４に示す。さらに図４における学習データの帰属更新処理（ステップ３１０）の詳細を図５に示す。いま、M個の代表ベクトルが決まっているとする。これらを

$\{c_j\}$, $j=1, 2, \cdots, M$

とする。学習データの帰属更新では、N個の学習用データが、それぞれどの代表ベクトルに帰属するかを決定する。図５において、第ｉ番目の学習データと第ｊ番目の代表ベクトルとの符号誤りを考慮した歪み尺度

$d(x_i, c_j)$

を計算する（ステップ４４０）。

【００２５】上記歪み尺度は次のように定義される。符号ｊを伝送したときに受信側で符号ｋが受信される確率を

$p(k \mid j)$

と表す。$p(k \mid j)$を簡易的に計算する一例として、通信路をランダムな二元対象通信路を仮定し、符号のビ＊ット数をｎ、符号ｋと符号ｊとの間のハミング距離をｈ、伝送路のランダムビット誤り率をｅとすると、

$p(k \mid j) = e^h (1-e)^{n-h}$

としてもよい。また、バースト誤りなど、ランダム誤りとは異なった特性の誤りが生じる可能性がある場合には、$p(k \mid j)$として、それらを考慮した誤り確率を用いてもよい。さらに、伝送するための符号化に誤り訂正符号を併用したときには、誤り訂正符号を併用したときの誤り確率を用いればよく、誤り検出符号を併用した場合には、受信側で誤りを検出し、修復処理をした後の誤り確率を用いればよい。また、実際の確率よりも多少誤り率を大きく見積もっても学習することにより、信頼性を高めることもできる。

【００２６】２つのベクトルx_iとc_jとの間の距離を表す尺度を

$d_v(x_i, c_j)$

で表すと、

【００２７】

【数１】

$$d(x_i, c_j) = \sum_{k=1}^{M} p(k \mid j)\, d_v(x_i, c_j)$$

【００２８】と定義する。これは符号誤りが生じたときの、受信側における

$d_v(x_i, c_j)$

の期待値を表す。なお、$d_v(x_i, c_j)$の定義のしかたは任意であるが、一般にはユークリッド距離や重み付きユークリッド距離が用いられる。x_i, c_jをそれぞれ

$x_i = (x_{i1}, x_{i2}, x_{i3}, \cdots, x_{ip})$

$c_j = (c_{j1}, c_{j2}, c_{j3}, \cdots, c_{jp})$

で表すと、重み付きユークリッド距離は

【００２９】

【数２】

$$d_v(x_i, c_j) = \sum_{r=1}^{p} w_{ir} (x_{ir} - c_{jr})^2$$

【００３０】で表される。重みのないユークリッドの場合には、

$w_{ir} = 1.0$

とすればよい。これらの距離を用いた場合には、上記d★（x_i, d_j）の計算を高速に実現できることが知られている。すなわち、

【００３１】

【数３】

$$d(x_i, c_j) = \sum_{k=1}^{M} p(k \mid j) \sum_{r=1}^{p} w_{ir} (x_{ir} - c_{jr})^2$$

【００３２】であるから、

【００３３】

【数４】

$$y_{jr} = \sum_{k=1}^{M} p(k \mid j)\, c_{jr}$$

$$a_{jr} = \sum_{k=1}^{M} p(k \mid j)\, (c_{jr})^2$$

【００３４】とおくことにより、y_{jr}, a_{jr}は入力に関係なく、量子化器の初期化時に一度計算すればよく、

【００３５】

【数５】

9　　10

$$d(x_i, c_j) = \sum_{r=1}^{p} w_{ir}(x_{ir}^2 - 2x_{ir}y_{jr} + a_{jr})$$

【００３６】となって、計算を簡略化できる。なお、符号帳を探索する尺度としては、上式において、x_{ir}^2 も計算する必要がない。

【００３７】上記ｄ（x_i，c_j）をｊが１からＭまで計算し、最もｄ（x_i，c_j）が小さくなった代表ベクトルに、学習用の第ｉ番目のデータを帰属させる。第ｉ番目のデータが第ｊ番目の代表ベクトルに帰属することを

Ｂｅ（ｉ）＝ｊ

で表す。

【００３８】上記操作をすべてのｉについて実行し、各学習用データをどの代表ベクトルに帰属させるかを決定する。

【００３９】次に、図４に戻り、歪みの総和Ｄを算出する（ステップ３２０）。歪みの総和Ｄは次のように定義される。

【００４０】

【数６】

$$D = \sum_{i=1}^{N} d(x_i, c_{Be(i)})$$

【００４１】または

【００４２】

【数７】

$$D = \sum_{j=1}^{M} \sum_{i \in (Be(i)=j)} d(x_i, c_j) \qquad (*1)$$

【００４３】上記２つの式は等価である。なお、上式２つめのΣはＢｅ（ｉ）＝ｊを満たすｉについての和を表す。上式で求めたＤがあらかじめ決められたしきい値よりも小さくなれば、ここで処理を終了し、小さくなければステップ３１０で更新した学習データの帰属に対して代表ベクトルが最適なものとなるように更新する。

【００４４】代表ベクトル更新のための式は、ベクトル＊

$$c'_{jr} = \frac{\sum_{k=1}^{M} p(k \mid j) \sum_{i \in (Be(i)=j)} w_{ir} x_{ir}}{\sum_{k=1}^{M} p(k \mid j) \sum_{i \in (Be(i)=j)} w_{ir}}$$

$$j = 1, 2, 3, \cdots, M$$

【００４６】となる。なお、本式は上記（＊１）式をc_{jr}で偏微分したものを０とおくことにより導出され、Ｄが極小となるc_{jr}を示す。この新代表ベクトルを用いて、再びステップ３１０に戻り、学習データの帰属更新処理を行う。

【００４７】図４の処理が収束して終了した時点で、図２における Lloyd-Maxアルゴリズムによる代表ベクトルの更新処理（ステップ１４０）が終了する。次のステップ１５０ではループカウンタ Repの値を調べ、あらかじめ決められた最大ループ回数Rep_max 以上ならステップ１８０へ、それ未満ならばステップ１６０へ進む。Rep_max は通常１で十分である。なお、 Rep_max が０のときは、従来のＬＢＧ法に等しい。

【００４８】ステップ１６０では、符号帳のインデックス付け換えを行う。その処理の詳細を図６に示す。

【００４９】図６においてｈはループのカウンタを表す。ｈの初期値は０に設定する（ステップ５１０）。符号帳評価関数Ｅ算出処理（ステップ５２０）では、現在の符号帳に対する初期評価関数Ｅを計算する。評価関数Ｅには、前述の歪みの総和Ｄ、すなわち（＊１）式を用いるべきである。しかし、あるインデックスとあるインデックスを交換するたびに全学習用データを用いて歪みの総和Ｄを計算しなおすことは、非常に多くの計算量を必要とする。そこで、簡略的な手段として、確率ｐ（ｋ｜ｊ）を用いて次の様に定義してもよい。

【００５０】

【数９】

＊間距離の定義式d_v（x_i，c_j）によって異なる。前述のように、重み付きユークリッド距離を用いると、新代表ベクトル

$c'_j = (c'_{j1}, c'_{j2}, c'_{j3}, \cdots, c'_{jp})$

は、

11 12

$$E=\sum_{k=1}^{M}\sum_{j=1}^{M} p(k\mid j)\, d_c(c_j, c_k)$$

【００５１】ただし、$d_e(c_j, c_k)$は代表ベクトルとc_kの間の距離を表し、一例として

【００５２】

【数１０】

$$d_c(c_j, c_k)=\sum_{r=1}^{M}(c_{jr}-c_{kr})^2$$

【００５３】のように、ユークリッド距離で定義してもよい。

【００５４】また、c_jが使われる頻度を考慮して

【００５５】

【数１１】

$$E=\sum_{k=1}^{M}\sum_{j=1}^{M} p(k\mid j)\, Pr(j)\, d_c(c_j, c_k)$$

【００５６】としてもよい。ただし、Ｐｒ（ｊ）はｊ番目の代表ベクトルが使用される確率を表す。このようにLloyd-Maxアルゴリズムの評価尺度と、インデックス付け換えの尺度を異なったものに定義すると、一方で歪みが減少したにもかかわらず、他方の処理で逆に歪みが増大する可能性も生じる。しかし、実験の結果、インデックス付け換えの評価関数を簡略化しても、悪影響は観察されない。

【００５７】上式で求められた初期評価関数値をＥ minに設定する（ステップ５３０）。次にｊ≠ｋなる任意の２つのインデックスを交換したと仮定したときの評価関数値Ｅを計算する（ステップ５４０，５５０）。ｊ，ｋの決め方は乱数を用いて決定してもよい。また、１≦ｊ≦Ｍ、１≦ｋ≦Ｍのすべての組み合わせについて評価関数Ｅを計算して、Ｅが最も小さくなったときのｊとｋを用いても良い。ｊ，ｋを交換して得られた評価関数の値ＥがＥ minよりも小さければ、インデックスｊとｋが割り当てられた代表ベクトルをお互いに交換する（ステップ５７０）。このとき、ループのカウンタｈは０にクリア、再びステップ５４０に戻る。逆にＥがＥ minよりも小さくならなかった場合には、ループカウンタｈの値に１を加える（ステップ５８０）。このとき、ｈの値があらかじめ決められた値Ｌよりも大きくなった場合には、どのインデックスを交換しても評価関数が小さくならないとみなして処理を終了する（ステップ５９０）。Ｌよりも小さい場合には、ステップ５４０に戻る。なお、ここでは、インデックスの交換を２つのインデックスで行っているが、より局所最小解の危険を避けるため、３つ以上で交換してみて、評価関数を計算してもよい。

【００５８】インデックス付けかえの後、ループカウンタ Repを１インクリメントし（ステップ１７０）、ステップ１４０に戻って再び誤り尺度 Lloyd-Maxアルゴリズムを適用して代表ベクトルを更新する。ステップ１４０と１６０を交互に行うことは、仮にステップ１４０で局所最小解に陥って十分な性能が得られなくとも、ステップ１６０におけるインデックスの付け換えにより局所最小を抜け出す可能性があり、より真の最小解に近づくことが期待される。計算量の増加が許されるならば、 Rep_max の値を２以上にするとよい。

$$c'_j=c_j+\Delta,\qquad j=M+1, M+2, \cdots, 2M$$

とする。Δは微小なベクトルである。

【００６２】こうしてクラスタ数を増やし、Ｍ’←Ｍとしてステップ１３０に戻り、以上の手順を繰り返す。

【００５９】ステップ１８０において、クラスタすなわち代表ベクトルの数Ｍが、あらかじめ割り当てられたビット数Ｂによって定まるクラスタ数2^Bに達していない場合には、１６－ａによってそれぞれのクラスタを分割する（ステップ１９０）。分割によってクラスタ数は２倍になる。これまでのクラスタ数をＭ、分割後のクラスタ数をＭ’とすると、

Ｍ’＝２Ｍ

分割により、Ｍ＋１から２Ｍまでのインデックスが新たに使用可能となる。ここで代表ベクトルの初期値をステップ２１０において決定してステップ１３０に戻る。なお、上記例では、クラスタを分割して２倍に増加させたが、必ずしも２倍でなくてもよいし、クラスタによって分割するものと、しないものがあってもよい。Ｍの初期値M_0を１に設定し、１回の分割によって２倍ずつにしていくと、ステップ１８０において、容易に$M=2^B$として終了することができ、都合がよい。

【００６０】代表ベクトルの初期値は次の様に決定する。

【００６１】１からＭまでの代表ベクトルの初期値は、現在の代表ベクトルをそのまま初期値とする。新しい初期値をc_j’とすると、

$$c'_j=c_j,\qquad j=1, 2, \cdots, M$$

である。また、新しく生成されたインデックスに対応する初期値は、

【００６３】ステップ１８０において、クラスタ数が目的とする数に達した場合には、学習の処理を終了し、そのときの代表ベクトルを最終的な符号帳の代表ベクトル

とする。

【００６４】なお、代表ベクトルの初期数が$M_0=2^B$である場合には、ステップ１９０，２００，２１０などのクラスタの分割に伴う処理が不要であることは明らかである。

【００６５】図３は、図２の手順の簡略版を示す。図２との違いは、ステップ２５０と２６０の処理を、それぞれ１回ずつ行っている点である。当然、図２の方が良好な結果が得られるが、計算を簡略化するために、図３の構成にしてもよい。また、基本的には図３の構成で行い、ステップ２７０において、$M=2^B$となる直前のループのみ図２のようにステップ１４０と１６０を交互に行ってもよい。

【００６６】ここまでは、一般的なベクトル量子化器の符号帳の学習方法について説明した。しかし、上記のような単純構成のベクトル量子化器では、割り当てるビット数の増加にもとなって、量子化するための計算量と記憶量が指数関数的に増大し、現在のハードウェア技術を考慮しても、実時間処理するためや、計算コストを下げるためには、量子化ビット数を１０から１３ビット程度以下に抑える必要がある。しかし、実際には、良好な量子化品質を得るために、２０ビットないし３０ビットを必要とすることも少なくない。このような問題に対して、少ない計算量で、しかも品質の劣化が少ない方法として、多段ベクトル量子化法が知られている。多段ベクトル量子化法の構成を図７に示す。図７は、説明を簡単にするために、３段の場合について説明した。多段ベクトル量子化法とは、複数の小規模なベクトル量子化器を縦続接続する方法である。各小規模なベクトル量子化器は、大きく分けて、それぞれ符号帳、ベクトル加算器、歪み判定部の３つの部分から成る。なお、１段目については、ベクトル加算器を省略してよい。１段目のベクトル量子化器の動作は、既に述べた通常のベクトル量子化器と同じである。符号帳５０の中に、M_1個の代表ベクトルを蓄え、これらを順に歪み判定部５３に送る。ｊ番目の代表ベクトルを$c_{jr}^{(1)}$（ｒはベクトルの次元を表す）とすると、歪み判定部５３では、$c_{jr}^{(1)}$と入力ベクトルx_{ir}との歪みを判定し、最も歪みの小さくなった代表ベクトルを１段目の量子化値（ベクトル）$q_{ir}^{(1)}$とする。２段目以降のｓ段目では、ｓ－１段目までの量子化値$q_{ir}^{(s-1)}$にｓ段目の符号帳に蓄えられたｊ番目の代表ベクトル$c_{jr}^{(s)}$を加えた、

$$*q_{ir}^{(s-1)}+c_{jr}^{(s)}$$

と入力ベクトルx_{ir}との歪みが最小になるｊ’番目の代表ベクトル$c_{j\cdot r}^{(s)}$をｓ段目の代表ベクトルとし、

$$q_{ir}^{(s)}=q_{ir}^{(s-1)}+c_{j\cdot r}^{(s)}$$

をｓ段目までの量子化値とする。

【００６７】このときの歪み尺度は、ユークリッド距離などの任意の尺度でよい。後述する例えば（＊２）式のような符号誤りを考慮した歪み尺度を用いると、符号誤りが生じても劣化の少ない量子化器とすることができる。

【００６８】この操作を段数分だけ繰り返し、最終段までの量子化値が決定されると、端子５８より入力ベクトルx_{ir}の量子化値x_{ir}^{*}として出力される。なお、上記説明では、各段において代表ベクトル（各段までの量子化値）を最適なものひとつに決定したが、途中の段では、いくつか量子化候補を残しておき、最終段までの量子化値が最適になるように決定した方が、若干の計算量増大につながるけれども、量子化性能はよい。なお、多段ベクトル量子化法の詳細については、文献：B.H.Juang,A.H.Gray著”Multiple Stage Vector Quantization for Speech Coding”Proc.ICASSP82,p.p597-600（1982）に記載されている。

【００６９】このような多段ベクトル量子化器の符号帳の設計にも、歪み尺度を変更するだけで、容易に本発明を利用できる。すなわち、多段ベクトル量子化器の符号帳の設計は、１段目から順に設計していくのであるが、１段目は通常のベクトル量子化器と実質的に同等であるので、本発明をそのまま利用可能である。２段目以降についても、図５における歪み尺度算出処理（ステップ４４０）の歪みの定義式、および図４における代表ベクトル更新処理（ステップ３４０）の更新式、ならびに図６における符号帳評価関数Ｅ算出処理ステップ５２０，５５０の定義式を変更し、各段の符号帳設計において本発明を適用するだけである。

【００７０】まず、歪み尺度の定義は、前述のように、入力ベクトルx_{ir}と量子化値（ベクトル）

$$q_{ijr}^{(s)}=q_{ir}^{(s-1)}+c_{jr}^{(s)}$$

とのベクトル間距離を重み付きユークリッド距離で表すと、

【００７１】

【数１２】

$$d_v(x_i,q_{ij}^{(s)})=\sum_{r=1}^{p}w_{ir}(x_{ir}-q_{ijr}^{(s)})^2$$

$$=\sum_{r=1}^{p}w_{ir}\{x_{ir}-(q_{ir}^{(s-1)}+c_{jr}^{(s)})\}^2$$

【００７２】となる。なお、このときx_{ir}と$q_{ijr}^{(s)}$をいったんケプストラムになおしてからユークリッド距

15

離をとってもよい。ベクトル間距離を上記$d_v(x_i, q_{ij}^{(s)})$のように表すと、符号誤りを考慮した歪み尺度は、

$$d(x_i, q_{ij}^{(s)}) = \sum_{k=1}^{M} p(k \mid j)\, d_v(x_i, q_{ij}^{(s)}) \quad (*2)$$

【００７４】となり、上記$d(x_i, q_{ij}^{(s)})$をｊが１からＭ（Ｍは代表ベクトルの数）まで計算し、最も$d(x_i, q_{ij}^{(s)})$が小さくなったｊ’番目の代表ベクトルに学習用の第ｉ番目のデータを帰属させる。なおここで、前段までの量子化値（ベクトル）$q_{ir}^{(s-1)}$は、

【００７５】

【数１４】

$$q_{ir}^{(s-1)} = \sum_{f=1}^{s-1} c_{j_s r}^{(f)}$$

【００７６】のように各段の代表ベクトルの和で表してもよいし、

【００７７】

【数１５】

$$y_{jr}^{(s)} = \sum_{k=1}^{M} p(k \mid j)\, c_{jr}^{(s)}$$

$$a_{jr}^{(s)} = \sum_{k=1}^{M} p(k \mid j)\, (c_{jr}^{(s)})^2$$

【００８２】を事前に計算しておくことにより、実際の量子化時に処理を高速化することができる。

【００８３】歪みの総和は、前述のように、

【００８４】

【数１８】

$$D = \sum_{i=1}^{N} d(x_i, q_{Be(i)}^{(s)})$$

※

$$D = \sum_{j=1}^{M} \sum_{i \in (Be(i)=j)} d(x_i, q_{Be(i)}^{(s)}) \quad (*3)$$

【００８７】で表す。ここで、

Ｂｅ（ｉ）＝ｊ

は、第ｉ番目の学習データベクトルが、ｓ段目において代表ベクトルｊに帰属することを表す。また、Ｎは学習データ数である。

16

＊【００７３】

【数１３】

＊

【００７８】とおいて、

【００７９】

【数１６】

$$q_{ir}^{(s-1)} = \sum_{f=1}^{s-1} y_{j_f r}^{(f)}$$

【００８０】即ち、受信したときの平均値（期待値）で表してもよい。一般には、後者のほうが性能が良い。また、$d(x_i, q_{ij}^{(s)})$の式を展開し、

【００８１】

【数１７】

※【００８５】または

【００８６】

【数１９】

【００８８】代表ベクトルの更新は、次の式によって行う。

【００８９】

【数２０】

17 18

$$c'_{jr} = \frac{\sum_{k=1}^{M} p(k \mid j) \sum_{i \in (Be(i)=j)} w_{ir} (x_{ir} - q_{ir}^{(s-1)})}{\sum_{k=1}^{M} p(k \mid j) \sum_{i \in (Be(i)=j)} w_{ir}}$$

$$j = 1, 2, 3, \cdots, M$$

【００９０】通常のベクトル量子化器の符号帳の設計と異なるのは、x_{ir}から１段前までの量子化値$q_{ir}^{(s-1)}$を差し引いて平均することである。なお、通常のベクトル量子化の場合、および多段ベクトル量子化における１段目の量子化の場合でも、１段前までの量子化値を０と考えれば、多段ベクトル量子化の特殊な場合であると考えることができる。

＊【００９１】図６における符号帳評価関数Ｅ算出処理（ステップ５２０，５５０）の定義式は、前述のように、歪みの総和Ｄすなわち、（＊３）式で表すのが望ましい。しかし、簡単のため

【００９２】

【数２１】

$$E = \sum_{k=1}^{M} \sum_{j=1}^{M} p(k \mid j)\, d_c(c_j^{(s)}, c_k^{(s)})$$

【００９３】としてもよいし、$c_j^{(s)}$が使われる頻度を考慮して

※【００９４】

【数２２】

$$E = \sum_{k=1}^{M} \sum_{j=1}^{M} p(k \mid j)\, Pr(j)\, d_c(c_j^{(s)}, c_k^{(s)})$$

【００９５】としてもよい。ただし、Ｐｒ（ｊ）は、ｊ番目の代表ベクトルが使用される確率を、d_c（$c_j^{(s)}$，$c_k^{(s)}$）は代表ベクトル$c_j^{(s)}$と$c_k^{(s)}$の間の距離を表し、一例としてユークリッド距離で定義してもよい。

【００９６】以上の変更のもとに、本発明を多段ベクトル量子化器の符号帳設計にも使用することができる。

【００９７】前にも述べたが、実際の通信をする場合には、なんらかのエラー訂正符号をつけて送信する場合が多い。しかし、通信路のビットレートの制約によっては、十分なエラー訂正符号を付加することができないことも多い。そのような場合に、本発明の方法が威力を発揮する。誤り訂正符号を用いないで本発明のみによって全段の符号誤り対策をしてもよいが、部分的に誤り訂正符号を用い、本発明と組み合わせる方法も効果的である。多段ベクトル量子化の場合には、１段目が最も符号誤りに対して品質の劣化が大きく、順に２段目、３段目・・・と符号誤りに対する感度が下がっていくという性質がある。そこで、誤り訂正符号を十分付加することができない場合には、１段目を誤り訂正符号によって強力に保護し、２段目以降は誤り訂正符号によってよりも、本発明において符号誤り率を適当に定めて符号誤りに強い符号帳を設計することにより、エラーのないところでの品質劣化が少なく、エラーが生じても品質の劣化が少ない符号化器を構成することができる。例えば、本発明によって、１段目はビット誤り率をゼロまたは、十分小さな値で符号帳を学習し、２段目以降は、伝送路の符号誤り率程度、または、２段目以降のはじめの方の段を実際の符号誤り率よりも若干高めに、後ろの方の段は実際の誤り率か、若干低めになるように設定して符号帳を設計し、実際の符号化の場面では、１段目の符号のみに誤り訂正符号をつけて伝送すると能率がよい。

【００９８】誤り訂正符号の中には、誤り訂正能力は若干低いが、誤り検出能力の高いものが存在する。これは、受信した複数のビットの組のなかで、どのビットが反転しているかを検出することはできないが、どれかが反転していることを検出できるものである。このように、訂正はできなくても、誤っていることさえわかれば、受信側でその符号を使うことをやめ、しかるべき処置をとることができる。例えば、音声信号などの場合には、時間軸において、前後のデータと相関が強いため、受信したまちがった符号を使うよりも、前後の誤りのない受信信号（符号）から、誤りを検出した時点の信号を推定するほうが劣化を小さく抑えることができる場合が多い。例えば、１フレーム（伝送単位）前のデータをそのまま繰り返して使うか、前後の再生ベクトルを線形に補間して当該時点の再生ベクトルとする方法がよく用いられる。このように、本発明を用いて符号帳を設計する場合には、伝送路の誤り率、誤り訂正符号、および、誤り検出符号を用いた時の修復方法をも考慮に入れるとよい。

【００９９】本発明は、例えば音声の符号化に応用する

とよい。本発明の方法が威力を発揮する音声符号化の方式のひとつに、ＣＥＬＰ方式がある。この方式は、入力音声信号を線形予測分析し、その結果得られた線形予測係数を例えば線スペクトル対（ＬＳＰ）と呼ばれるパラメータに変換する。ＬＳＰはその量子化特性が非常に優れているので、よく用いられるが、ＬＳＰ以外にも、ＰＡＲＣＯＲ係数と呼ばれるパラメータに変換する方法もある。ここでは、これらのパラメータを総称して、線形予測パラメータと呼ぶ。ＣＥＬＰ方式ではまず、この線形予測パラメータを量子化する。次に、上記線形予測パラメータから得られる線形フィルタを、音源信号で駆動して得られる信号と、入力信号との歪みが最小となるように音源信号を決定する。線形予測パラメータの量子化、音源信号の量子化とも、通常はベクトル量子化を用いるため、ともに本発明の手法は優れた結果をもたらす。

【０１００】線形予測パラメータの量子化に本発明を適用する方法の一例を述べる。まず、入力音声信号の線形予測分析を行って得られた線形予測係数を線スペクトル対（ＬＳＰ）に変換する。線形予測分析の次数をｐとすると、ＬＳＰはｐ次のベクトルとして得られ、このベクトルをある符号で表すことができるように量子化する。なお、ＬＳＰは、音声スペクトルの包絡特性を表すパラメータである。音声信号からＬＳＰを算出する方法については、例えば、文献：吉井貞熙著「ディジタル音声処理」（東海大学出版会）に記載されている。

【０１０１】ここでは、一例として、線形予測分析次数１０次、ＬＳＰを計算する時間周期（フレーム周期）は２０ｍｓ程度で行うとよい。なお通常、分析次数は１０次～１６次、フレーム周期は１０～４０ｍｓ程度に設定する。学習用のデータは、計算時間の許す限り多くのデータを準備するとよいが、複数話者の異なった発声およそ１５００秒もあれば十分である。ＬＳＰをベクトル量子化する際に、十分な品質を維持するためには、１ベクトルあたり２０～３０ビット必要であるので、１段のベクトル量子化器で実現することは、計算量と記憶量の観点から現実的でない。そこで一例として、３段の多段ベクトル量子化を用い、例えば１段目を８ビット、２段目を８ビット、３段目を６ビットの、１ベクトルあたり２２ビットとする。１ベクトルあたりの必要ビット数は駆動音源の量子化方法によって異なり、ＣＥＬＰでは２０～２２ビットで十分な品質が得られる。

【０１０２】符号誤りを考慮したＬＢＧ学習、および、実際の量子化時における距離尺度の誤り率の設定は、用途によって異なるが、例えば移動体通信用では、１．０～３．０％程度に設定するとよい。ただし、符号誤り率を高く設定すると、符号誤りのないところでの品質が劣化するため、符号誤りのないときの品質を重視したい場合には、誤り率は上記値よりも低めに設定するとよく、また、符号誤りが生じたときの劣化防止を重視するのであれば、上記値よりも若干高めに設定してもよい。また、１段目は相対的に符号誤りに対する感度が高いので、１段目のみ誤り訂正符号または誤り検出符号を使用すると効率がよい。この場合には１段目の学習・量子化時の誤り率の設定は、十分低いかまたは０に設定してよく、２段目以降は伝送路に合わせた誤り率を設定する。なお、誤り検出符号を用い、受信側で誤りが生じたことのみが検出された時には、受信した符号からベクトルを再生することをやめ、前後の時間のベクトルから誤りを検出した時刻のベクトルを推定するとよい。ベクトル間の歪み尺度には、ユークリッド距離や重み付きユークリッド距離、ケプストラム歪み等を用いるとよい。

【０１０３】本発明によって、ベクトル量子化器の符号帳を設計する方法の効果を調べるために、以下の条件による実験を、コンピュータシミュレーションによって行った。

【０１０４】本実験では、無記憶ガウス情報源をベクトル量子化して、従来法との性能を比較する。ベクトル次元は８次元、量子化ビット数は８ビットで、ベクトル量子化器は１段で構成した。ベクトル間距離尺度はユークリッド距離を用いた。学習データは１０，０００ベクトル、テスト用には学習外の５，０００ベクトルを用い、全体のＳＮ比で評価した。学習時および量子化時の符号誤り率は１．０％に設定し、テストでは、チャネル誤り率を０．０～１０％まで変化させてＳＮ比の変化を調べた。比較した方法は、本発明の方法の他、従来法として符号誤り率を全く考慮しないＬＢＧ法、歪み尺度として符号誤りを考慮した尺度（誤り率１．０％に設定）を用いたＬＢＧ法、また、上記誤りを考慮したＬＢＧ法が終了したあとに、できあがった符号帳に対してインデックス付けかえを適用した方法について調べた。結果を図７に示す。全く符号誤りを考慮しない場合には、△印と破線で示すように、符号誤りとともに急速に劣化する。一方、本発明による方法では、符号誤りのないところでわずかに品質が劣化しているものの、符号誤りが生じても劣化を少なく抑えている。また、●印と実線で示される誤り尺度のＬＢＧ法でも、十分良好な品質が維持されているが、本発明のほうが優れた結果を示している。また、ＬＢＧ法終了後にインデックス付けかえを行う方法（□印と一点鎖線）では、誤り尺度ＬＢＧに比べてほとんど改善していない。

【０１０５】以上より、本発明が、符号誤りのある場合のベクトル量子化器の設計において、非常に優れた方法であることが示された。なお、本実験では無記憶ガウス情報源を用いて性能を比較したが、実際の音声信号や画像信号のような相関が強い信号の場合には、さらに大きな効果が期待される。

【０１０６】

【発明の効果】以上説明したように、本発明によれば、局所最小解に陥る危険を低減し、より高性能な符号帳を

21

設計することができるとともに、またわずかな計算量の増加により容易に最適に近い状態に導くことができる。

【図面の簡単な説明】

【図１】本発明によって設計した符号帳を用いてベクトル量子化器を構成する一例を示すブロック図である。

【図２】本発明による符号帳の学習法の流れの概略を示すフローチャートである。

【図３】本発明による符号帳の学習法の流れの概略を示すフローチャートである。

【図４】図３の処理における Lloyd-Maxアルゴリズムによる代表ベクトルの更新処理であるステップ１４０の詳細な処理を示すフローチャートである。

【図５】図４の処理における学習データの帰属更新処理であるステップ３１０の詳細な処理を示すフローチャートである。

22

【図６】図２に示す符号帳インデックス付け換え処理であるステップ１６０の詳細な処理を示すフローチャートである。

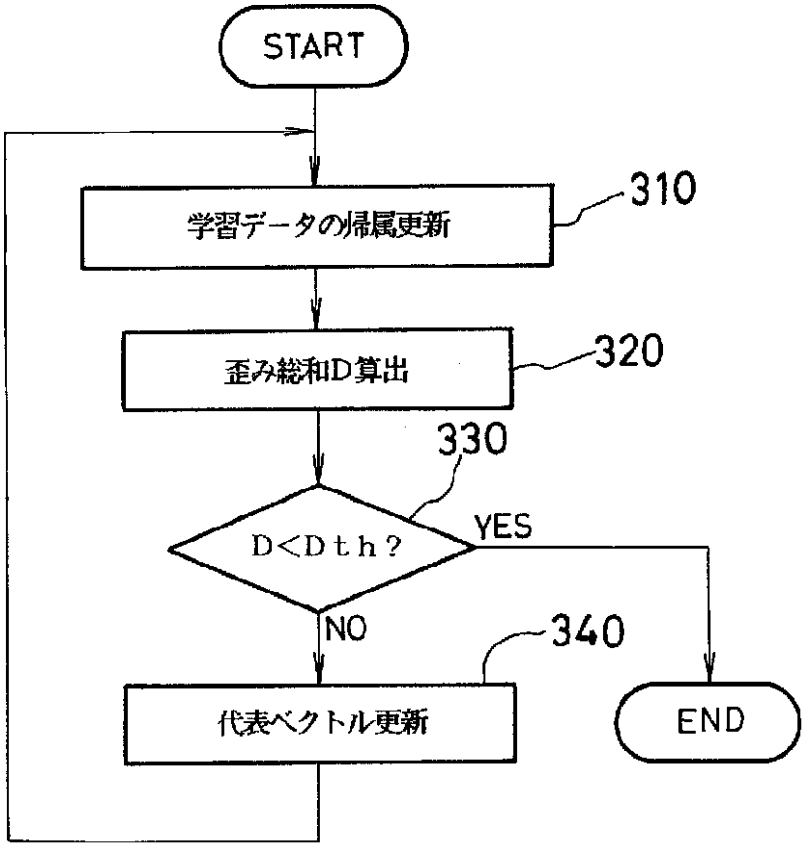
【図７】多段ベクトル量子化の構成例を示すブロック図である。

【図８】本発明の効果を調べるためにコンピュータシミュレーションによる符号誤りの実験を行った結果を示す図である。

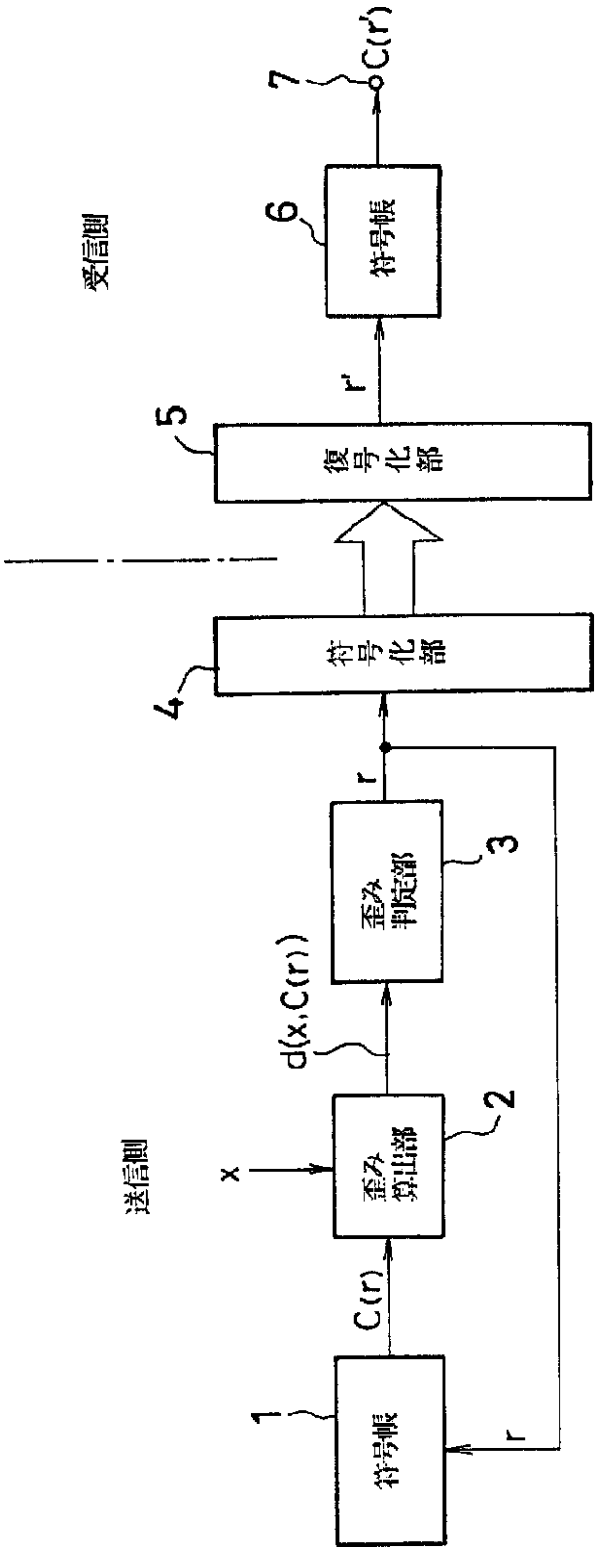
【符号の説明】

1　符号帳
2　歪み算出部
3　歪み判定部
4　符号化部
5　復号化部
6　符号帳

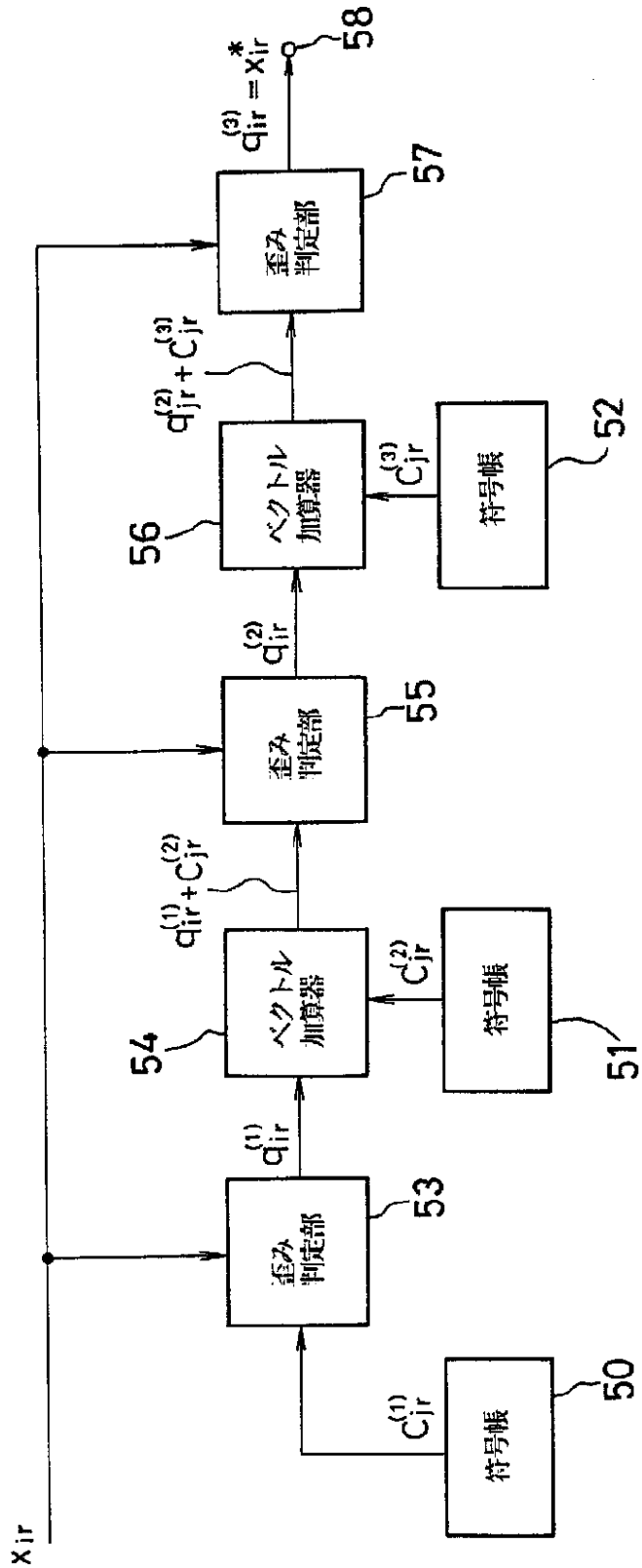
【図４】



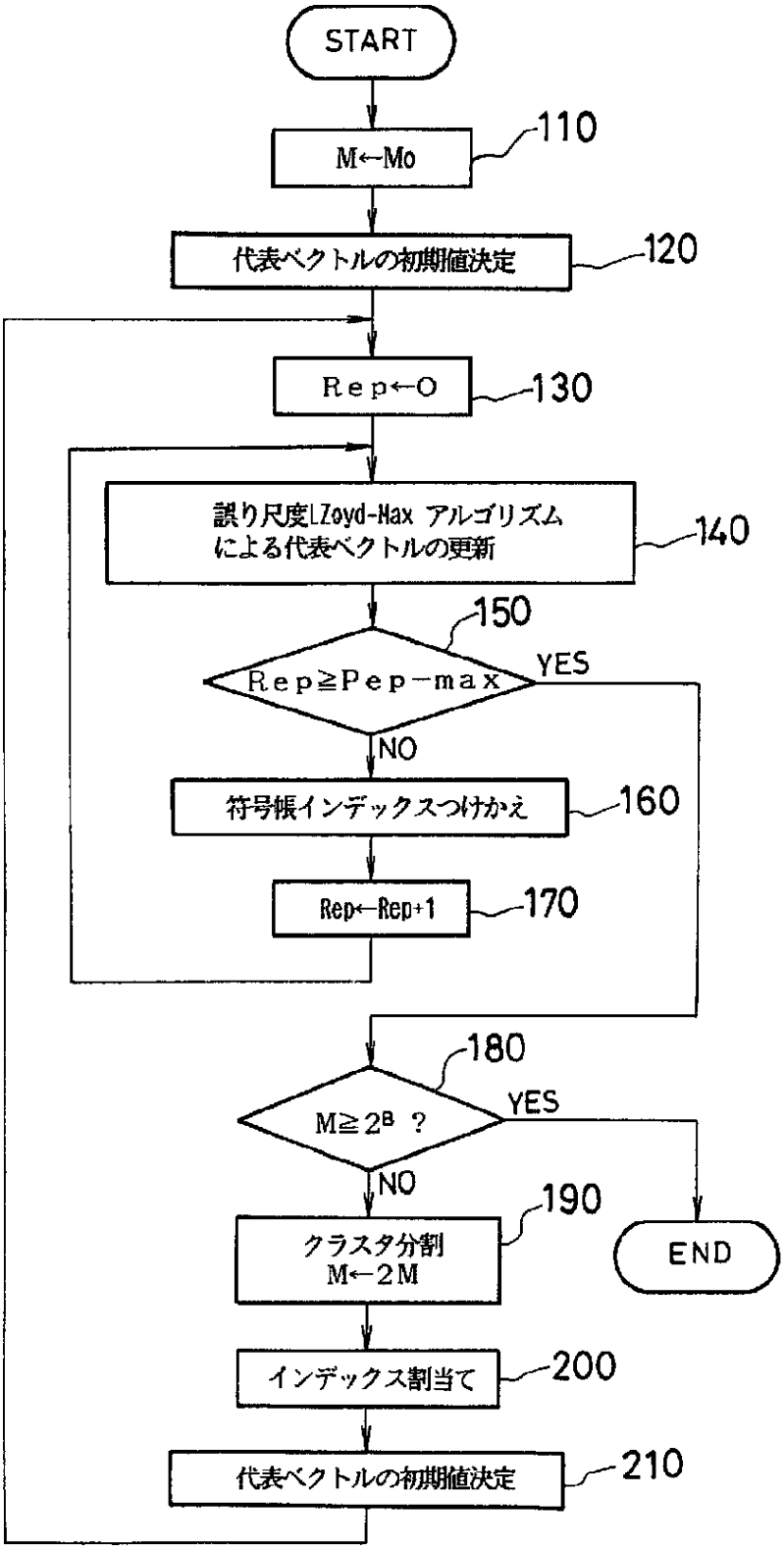
【図１】


送信側
受信側
符号帳
歪み算出部
歪み判定部
符号化部
復号化部
符号帳
x
C(r)
d(x,C(r))
r
r'
C(r')

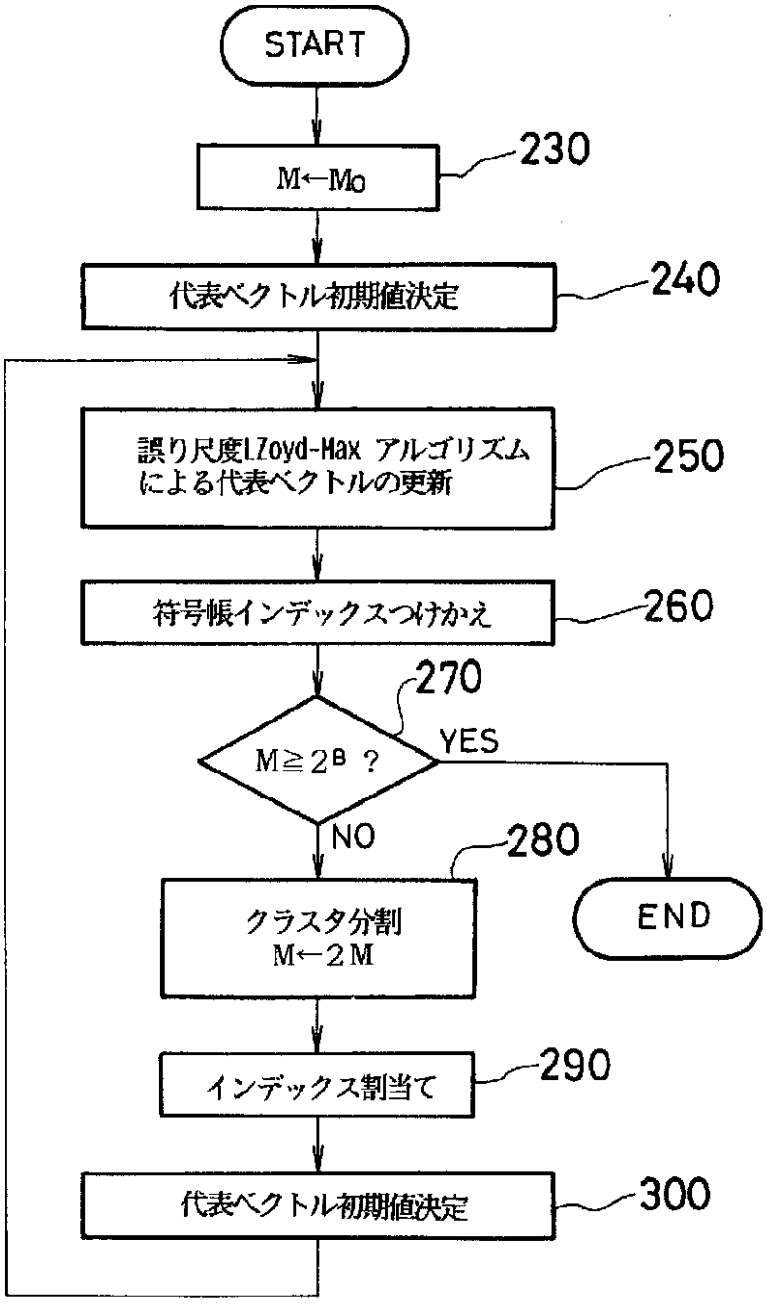
【図７】


符号帳
ベクトル加算器
歪み判定部
x_{ir}
$C_{jr}^{(1)}$
$C_{jr}^{(2)}$
$C_{jr}^{(3)}$
$q_{ir}^{(1)}$
$q_{ir}^{(2)}$
$q_{jr}^{(2)}+C_{jr}^{(3)}$
$q_{ir}^{(3)}=x_{ir}^{*}$

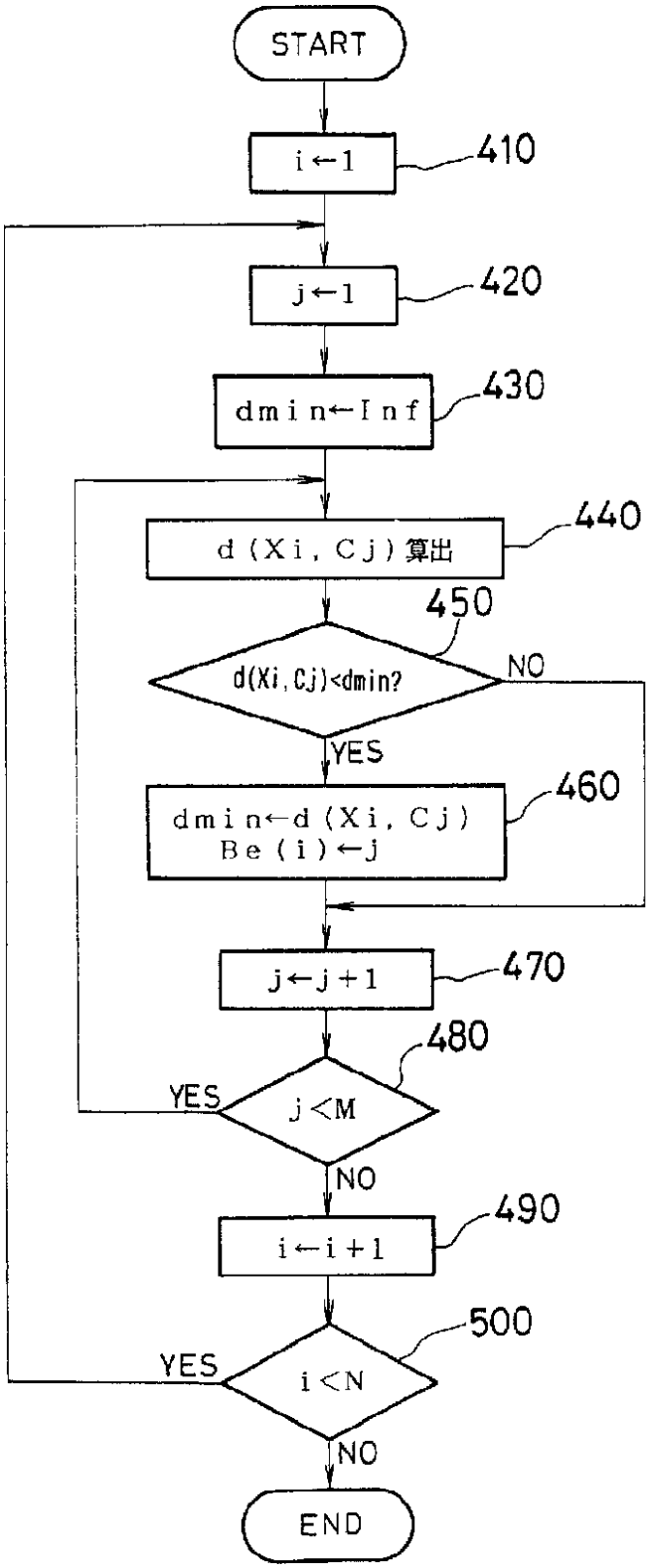
【図２】


START
M←Mo
110
代表ベクトルの初期値決定
120
Rep←O
130
誤り尺度LZoyd-Max アルゴリズム
による代表ベクトルの更新
140
150
Rep≧Pep−max
YES
NO
符号帳インデックスつけかえ
160
Rep←Rep+1
170
180
M≧2B ?
YES
NO
190
クラスタ分割
M←2M
END
インデックス割当て
200
代表ベクトルの初期値決定
210

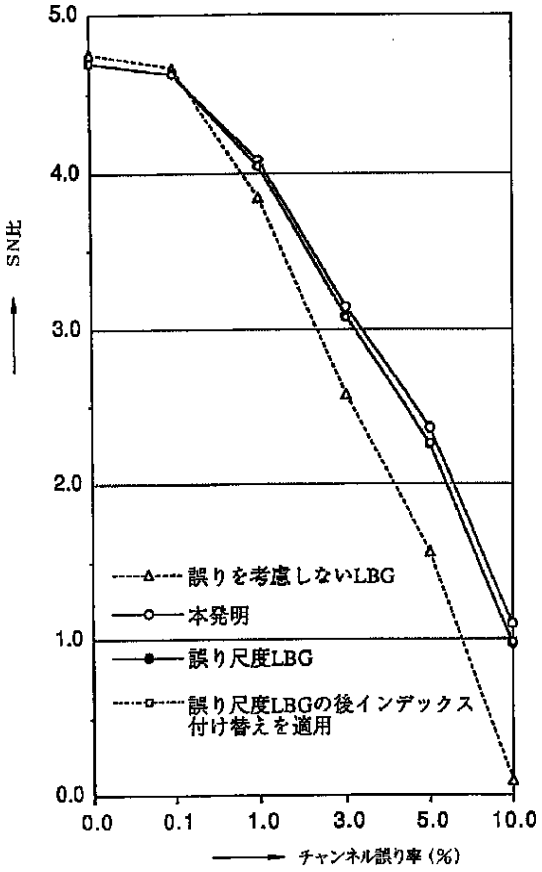
【図３】


START
M←Mo
230
代表ベクトル初期値決定
240
誤り尺度LZoyd-Max アルゴリズム
による代表ベクトルの更新
250
符号帳インデックスつけかえ
260
270
M≧2B ?
YES
NO
280
クラスタ分割
M←2M
END
インデックス割当て
290
代表ベクトル初期値決定
300

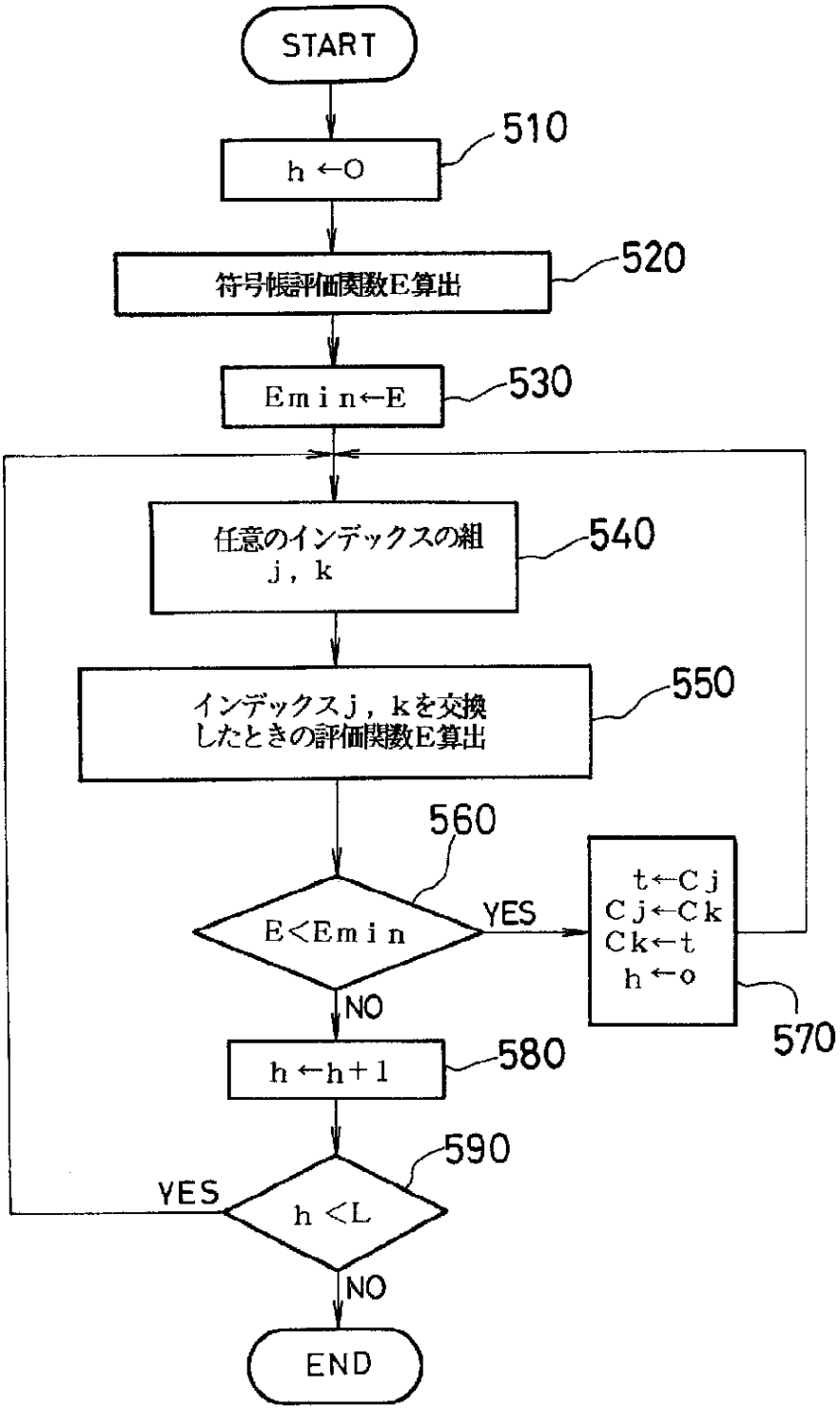
【図５】


START
i←1
410
j←1
420
dmin←Inf
430
d(Xi,Cj)算出
440
450
d(Xi,Cj)<dmin?
NO
YES
dmin←d(Xi,Cj)
Be(i)←j
460
j←j+1
470
480
YES
j<M
NO
i←i+1
490
500
YES
i<N
NO
END

【図８】


5.0
4.0
3.0
2.0
1.0
0.0
SN比
誤りを考慮しないLBG
本発明
誤り尺度LBG
誤り尺度LBGの後インデックス付け替えを適用
0.0
0.1
1.0
3.0
5.0
10.0
チャンネル誤り率(%)

【図６】



フロントページの続き

(72)発明者　間野　一則
東京都千代田区内幸町一丁目１番６号
日本電信電話株式会社内

(72)発明者　須田　博人
東京都千代田区内幸町一丁目１番６号
日本電信電話株式会社内

審査官　　石井　研一

(56)参考文献　　特開　昭62－188575（ＪＰ，Ａ）
特開　平１－205638（ＪＰ，Ａ）
特開　平１－232829（ＪＰ，Ａ）
特開　平１－259626（ＪＰ，Ａ）
電子通信学会技術研究報告、信学技報Ｖｏｌ．84、Ｎｏ．60、ｐｐ17－23、ＣＳ84－32、「通信路歪を低減するベクトル量子化の伝送符号の割り当てに関する一検討」相澤ほか

(58)調査した分野(Int.Cl.⁷，ＤＢ名)
H03M　7/30