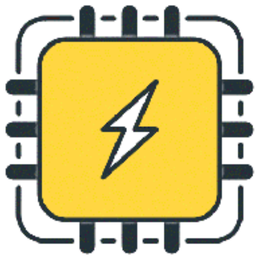
http://www.takagi.inf.uec.ac.jp/mmip/

マルチメディア処理

Multimedia Information Processing

第5回　マルチメディアデータの符号化とファイル形式



髙木一幸

講義概要

- マルチメディアデータ（文書、音声、画像、映像など）の表現方法と処理技術について、基礎的な内容を紹介・解説する。
- 授業時間中および宿題として、計算機を使った演習を行うことにより実際のデータの扱い方を学び、理解を深める。

第1回：授業の概要説明、
人間の感覚とマルチメディア処理（4/11）

髙木

第2回：文字・テキストの表現と処理（4/18）

第3回：音声のディジタル表現と処理（4/25）

第4回：画像・映像のディジタル表現（5/2）

第5回：マルチメディアデータの符号化とファイル形式（5/9）

第6回：2次元図形の表現と描画（5/16）

第7回：画像処理(1)画素ごとの濃淡変換（5/23）

第8回：画像処理(2)空間フィルタリング（5/30）

廣田

第9回：カメラと写真撮影（6/13）

第10回：3次元コンピュータグラフィックス（6/20）
(1)形状表現と透視投影

第11回：3次元コンピュータグラフィックス（6/27）
(2)照明効果とシェーディング

第12回：アニメーションと映像制作（7/4）

第13回：シミュレーションと可視化（7/11）

第14回：グラフィックパイプラインとシェーダ（7/18）

第15回：マルチメディアの応用例I（7/25）

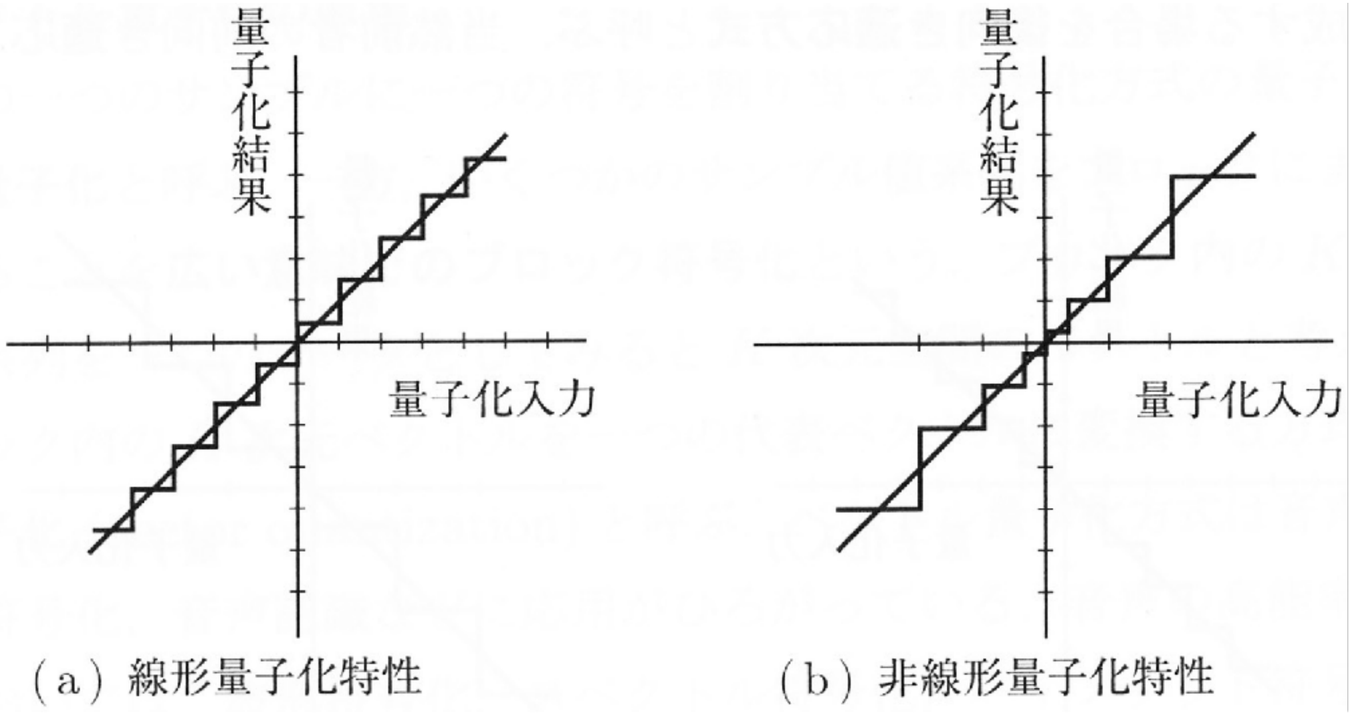
音声の符号化

波形符号化		
振幅方向の処理	非直線量子化	log PCM
	適応量子化	適応 PCM(APCM) 適応差分変調方式 (ADM)
時間軸上の処理	予測符号化	差分 PCM(DPCM) 適応差分 PCM(ADPCM) 適応予測符号化 (APC)
周波数軸上の処理	帯域分割符号化 (SBC:Sub-band coding) 適応変換符号化 (ATC:Adaptive transform coding)	
スペクトル符号化		
パラメトリック	LPC(Linear predictive coding) 方式 PARCOR(Partial auto-correlation coefficient) 方式 LSP(Line spectrum pair) 方式	
ノンパラメトリック	ケプストラム法	
生成音源 (ハイブリッド 符号化を含む)	パルス駆動線形予測法 (Pulse excited LPC:PELP) 残差駆動線形予測法 (Residual excited LPC:RELP) 音声駆動線形予測法 (Voice excited LPC:VELP) マルチパルス駆動線形予測法 (Multi-pulse excited LPC:MPC) 符号励振型線形予測法 (Code excited Linear Prediction:CELP)	

- PCM
- LPC
- CELP

©板橋秀一編著，音声工学、森北出版、2005年。

非直線量子化 log PCM 対数圧伸（log）PCM, μ-law



（a）線形量子化特性　（b）非線形量子化特性

©板橋秀一編著，音声工学、森北出版、2005年。

- 量子化ステップ幅…大振幅では粗く、小振幅で細かく…量子化器のステップ数を減少できる（量子化ビット数の減少）。
- 日本やアメリカの電話通信で利用されているPCM音声の圧縮方式の一つ。
- 人の聴覚特性が音の大きさに対して対数的であることを利用。
- WaveNetではμ-lawアルゴリズムを利用して音声波形を圧縮した後に8bitに量子化。

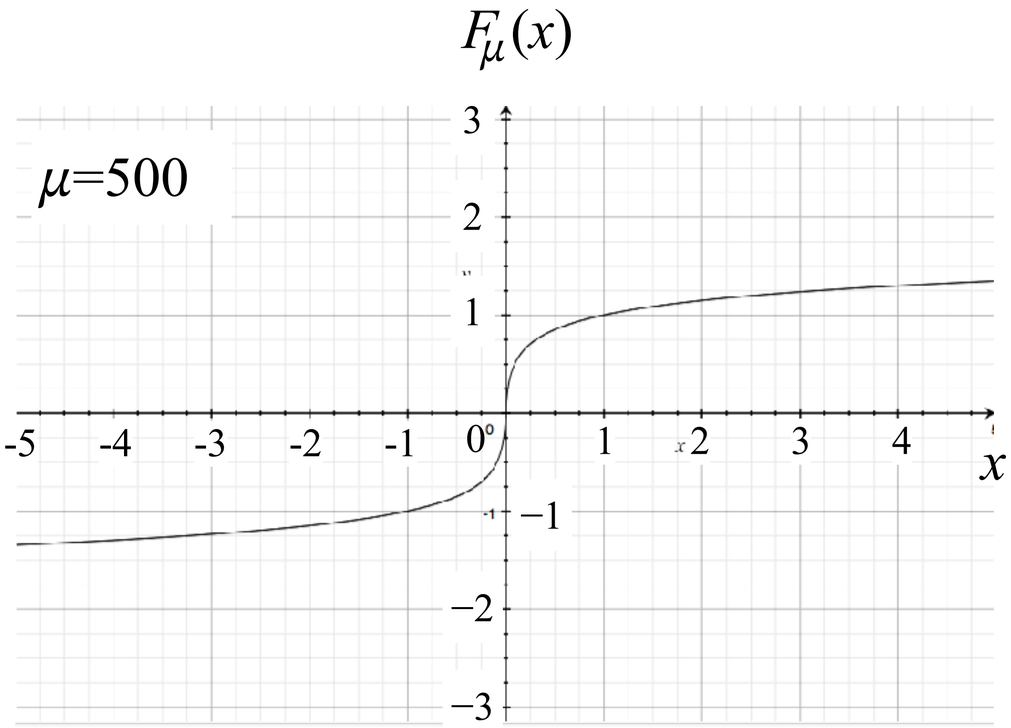
非直線量子化 log PCM　対数圧伸（log）PCM, μ-law

μ-lawの圧縮式　$$F_\mu(x) = \mathrm{sgn}(x)\frac{\log(1+\mu|x|)}{\log(1+\mu)}$$

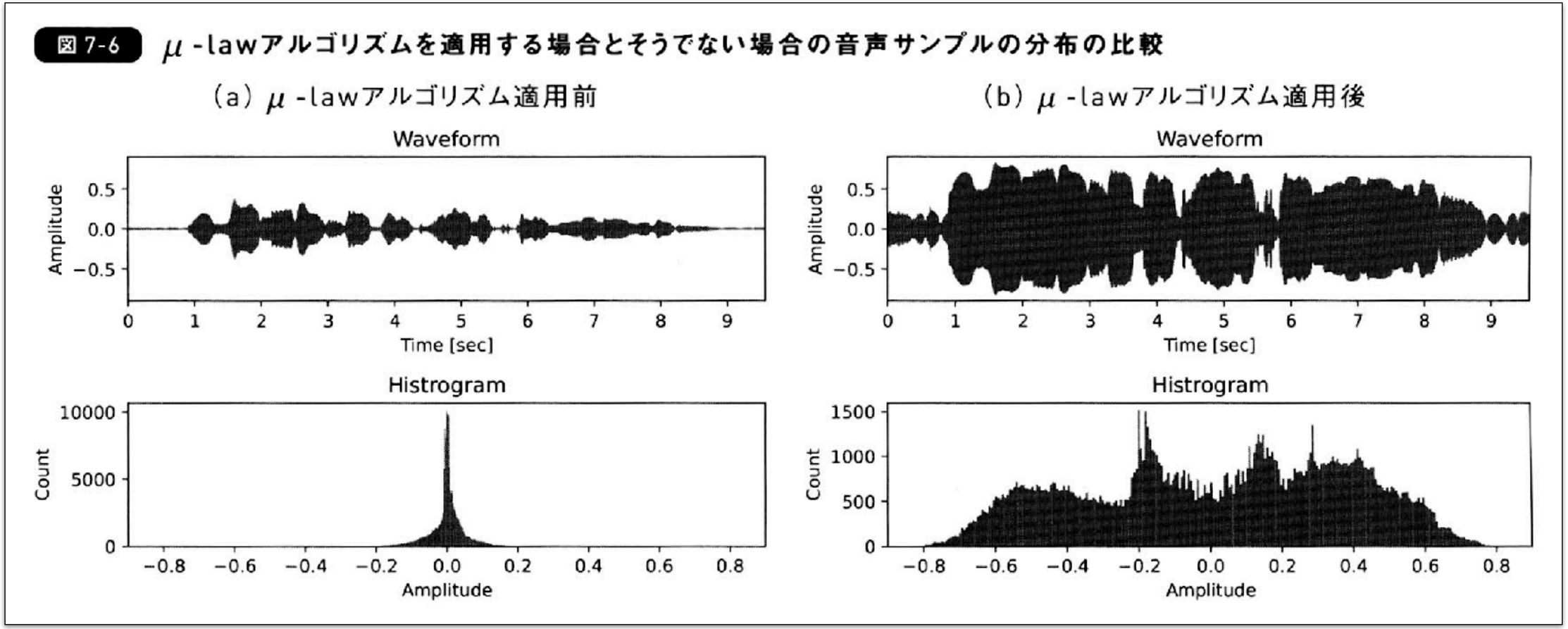
x：入力信号振幅（$-1 \leqq x \leqq 1$）

$\mathrm{sgn}(x)$：xの極性

μ：$\mu \to 0$：無圧縮、μ=100～500



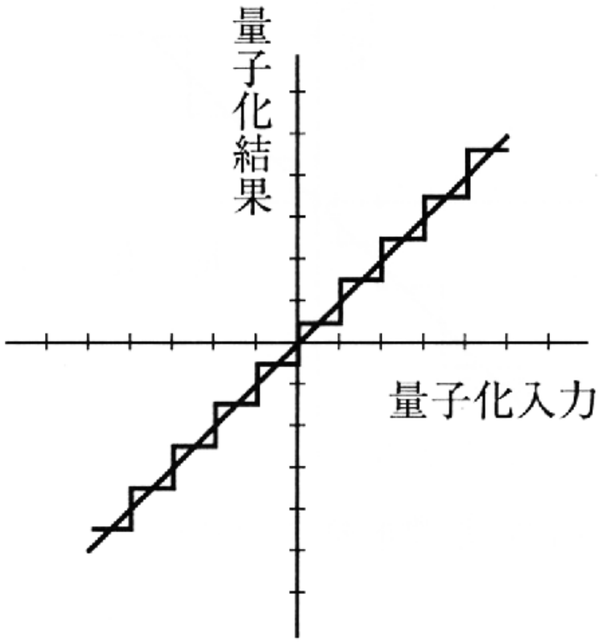
μ-lawの適用と非適用の音声波形サンプル



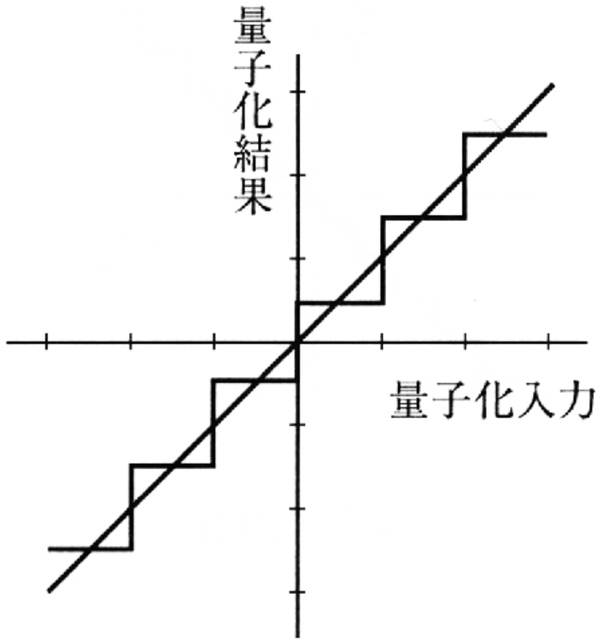
©山本龍一 (著), 高道慎之介 (著), Pythonで学ぶ音声合成 機械学習実践シリーズ、 2021年。

- 元の音声波形は振幅の分布が 0 付近に集中。
- μ-lawアルゴリズムを適用することによって，分布広がっている：量子化ビットを有効に使うことができる。

適応量子化(APCM) Adaptive Pulse Code Modulation



(a) 小信号時の量子化特性



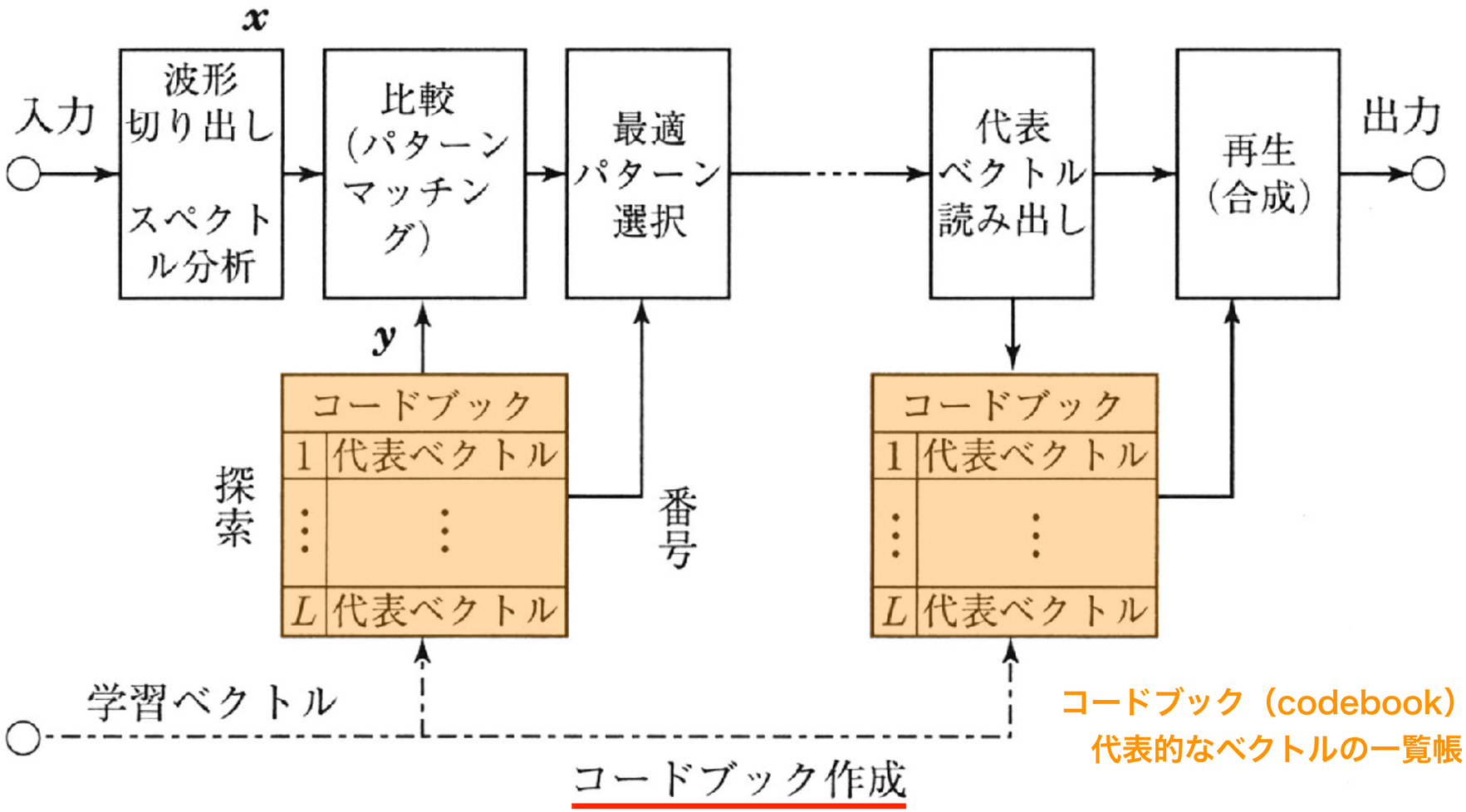
(b) 大信号時の量子化特性

- log PCM：大振幅の入力レベルでのSN比が犠牲に
- APCM：入力レベルに応じて量子化ステップを切り替える

量子化ステップの切り替え情報の伝達が必要

©板橋秀一編著，音声工学、森北出版、2005年。

ベクトル量子化（VQ） Vector Quantization



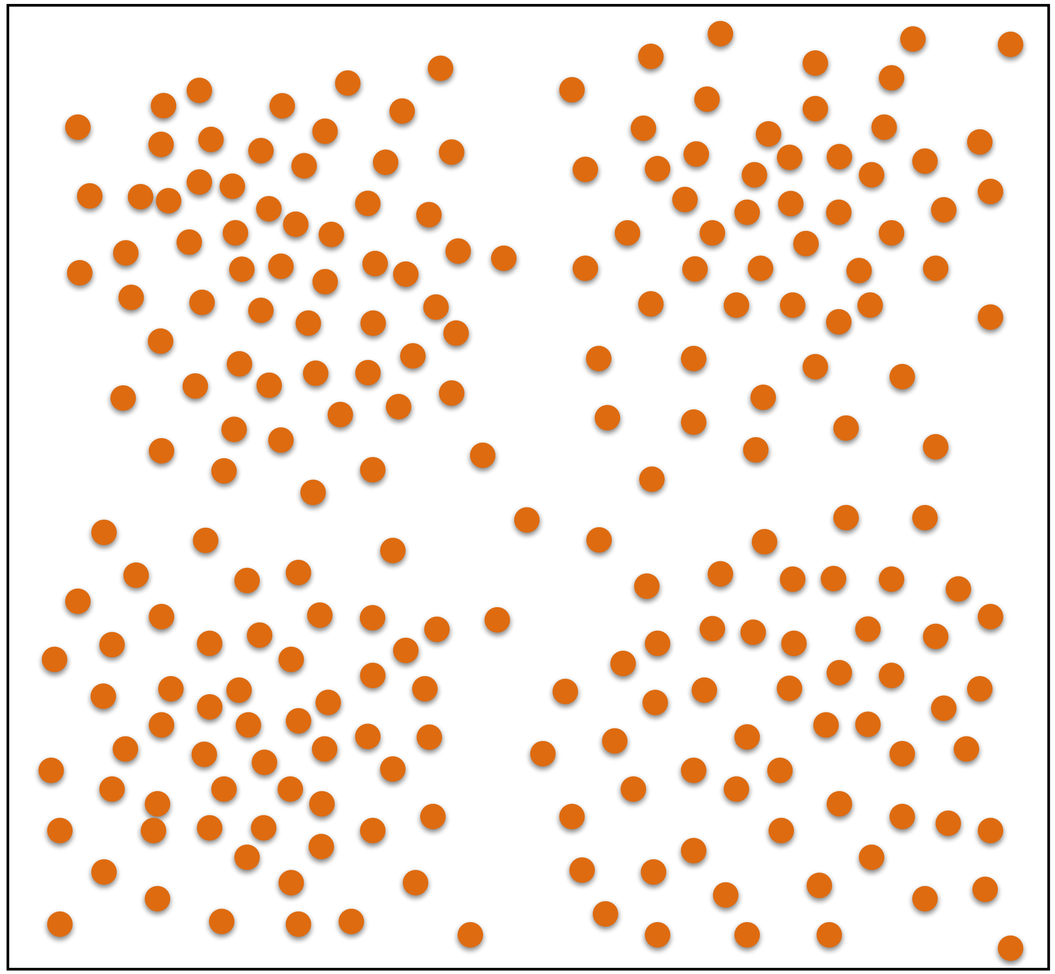
コードブック（codebook）
代表的なベクトルの一覧帳

©板橋秀一編著，音声工学、森北出版、2005年。

ベクトル量子化（VQ） Vector Quantization

$$R^n \rightarrow N$$

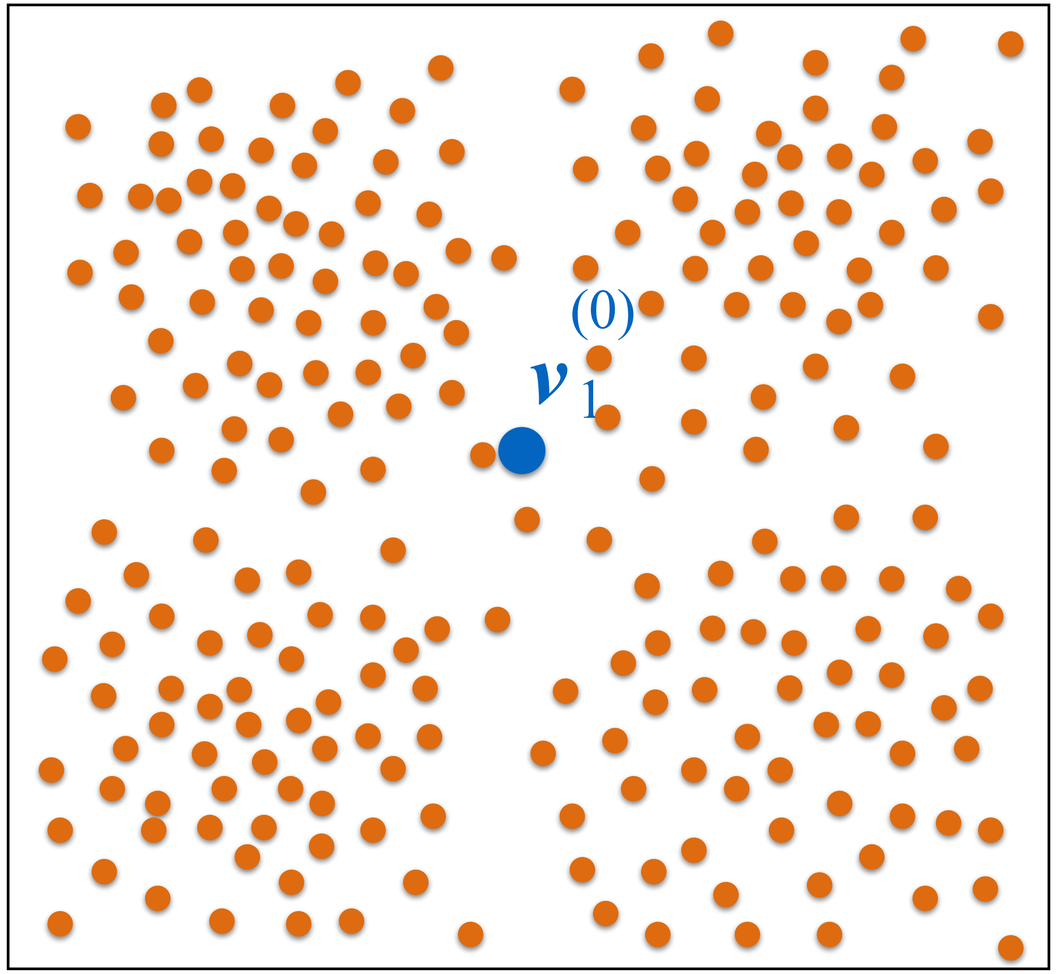
$$\boldsymbol{x} \in R^2$$



ベクトル量子化（VQ） Vector Quantization

コードブック学習：LBGアルゴリズム　Linde Buzo Gray

Step 1 初期化：データの重心を計算し代表ベクトルとしてサイズ１のコードブックを作成

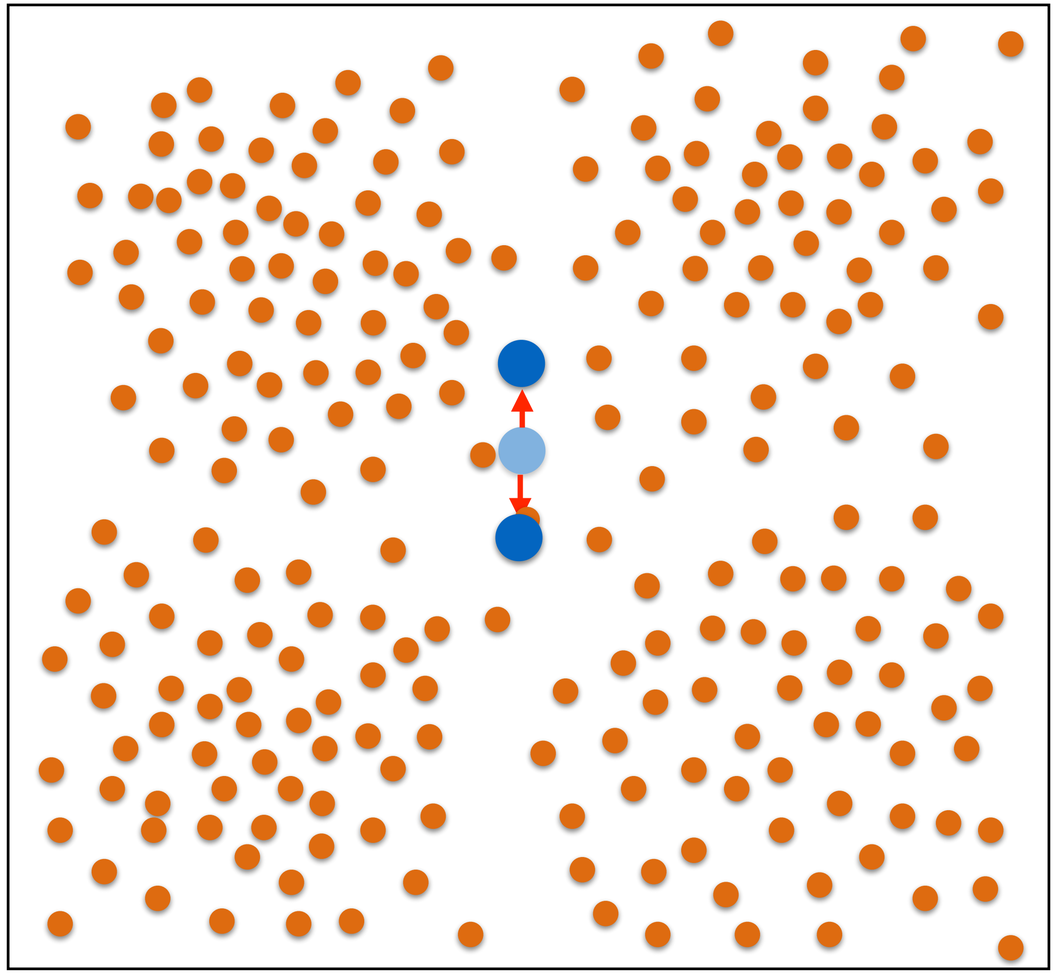


$v_1^{(0)}$ 初期(0)のサイズ１の
コードブックの
第１符号の代表ベクトル

ベクトル量子化（VQ） Vector Quantization

コードブック学習：LBGアルゴリズム Linde Buzo Gray

Step 2 コードブックの代表ベクトルに微小値を加え、近接した代表ベクトルに分割



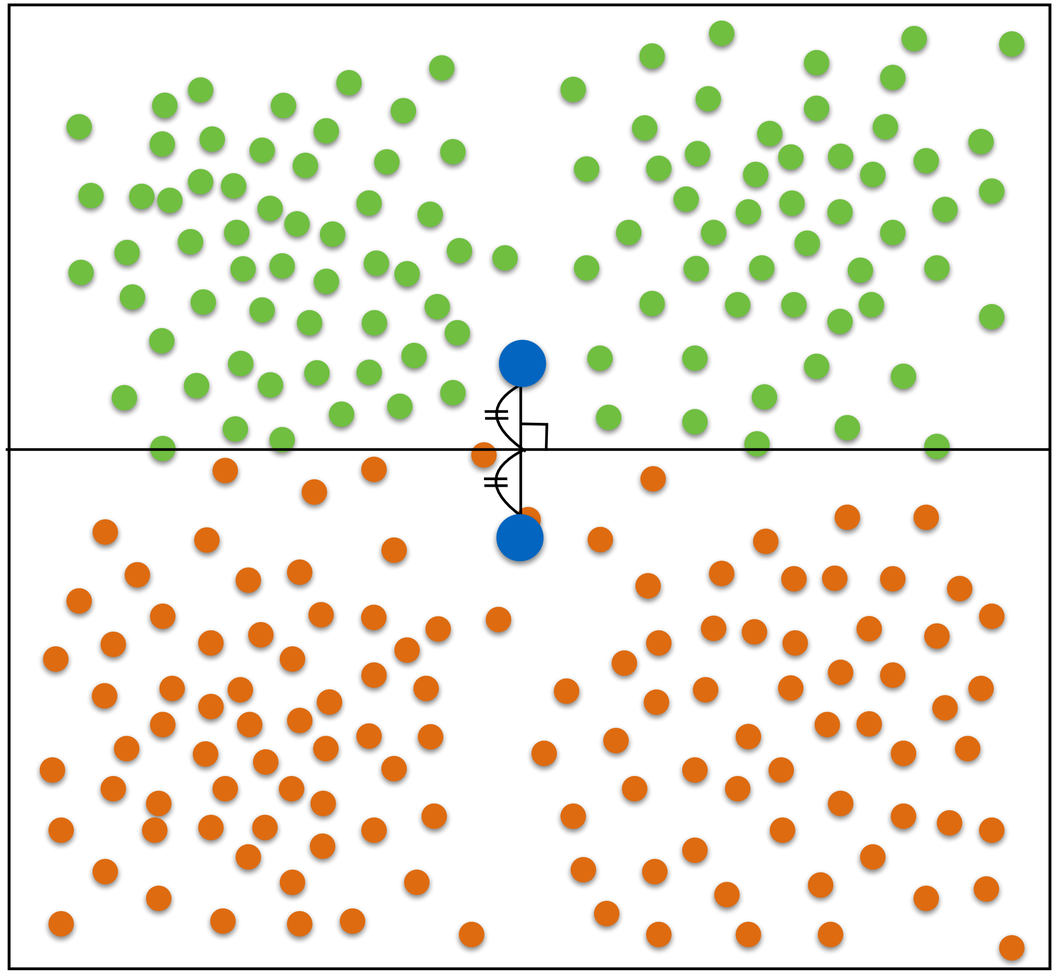
$$\boldsymbol{v}_1^{(0)} + \delta$$

$$\boldsymbol{v}_1^{(0)} - \delta$$

ベクトル量子化（VQ） Vector Quantization

コードブック学習：LBGアルゴリズム　Linde Buzo Gray

Step 3　新たに作られた代表ベクトルで学習データを分類。



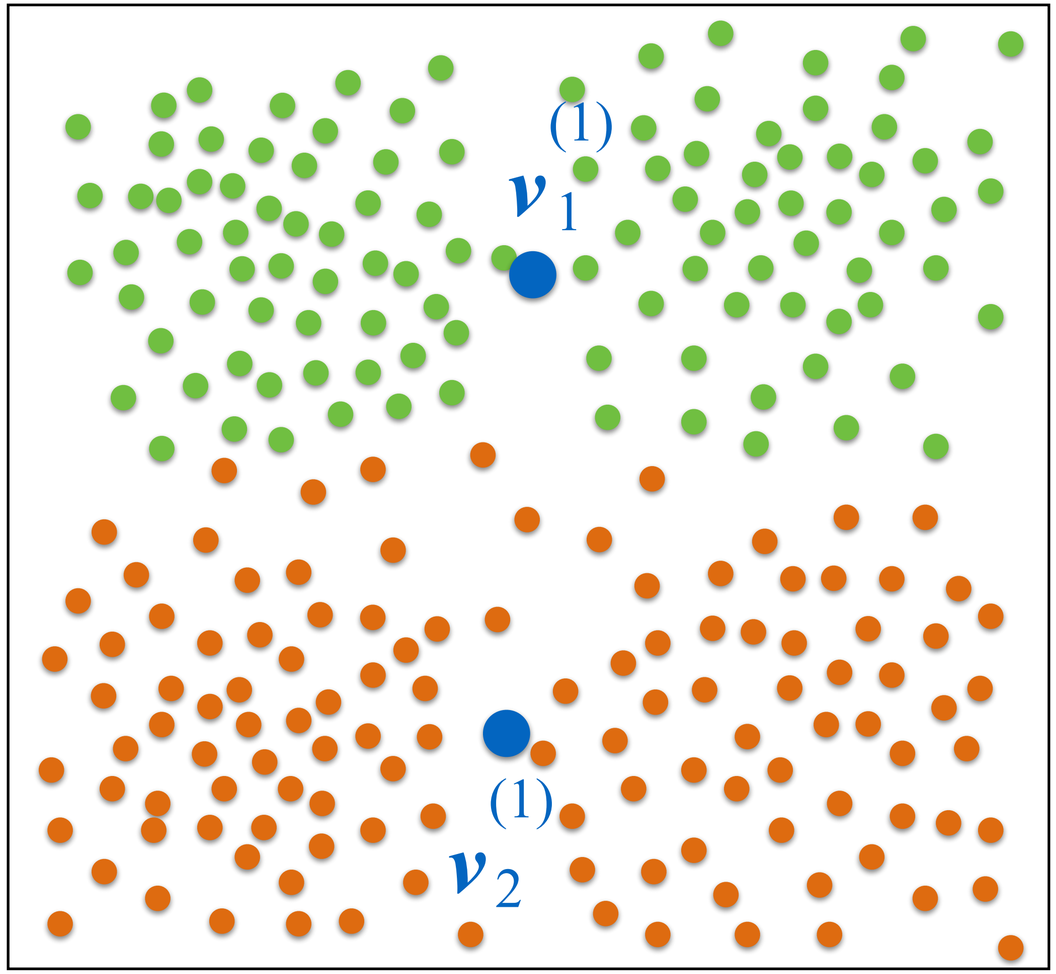
$$\boldsymbol{v}_1^{(0)} + \delta$$

$$\boldsymbol{v}_1^{(0)} - \delta$$

ベクトル量子化（VQ） Vector Quantization

コードブック学習：LBGアルゴリズム　Linde Buzo Gray

Step 4　分類された学習データの重心を計算し、代表ベクトルとする。



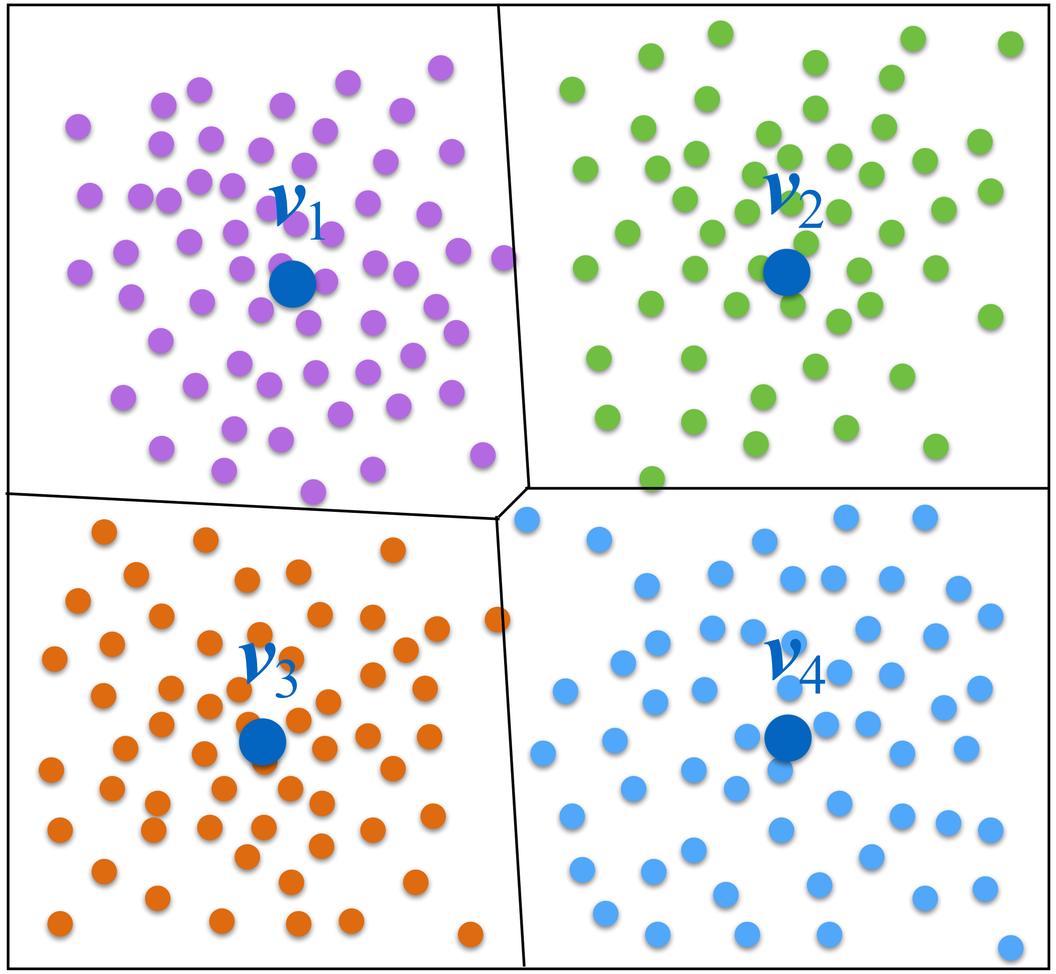
$v_1^{(1)}$　第１段階のサイズ2の
コードブックの
第1符号の代表ベクトル

$v_2^{(1)}$　第１段階のサイズ2の
コードブックの
第2符号の代表ベクトル

ベクトル量子化（VQ） Vector Quantization

コードブック学習：LBGアルゴリズム　Linde Buzo Gray

Step 5 コードブックサイズが所定の大きさになってなければ、**Step 2**から繰り返す



コードブック（符号帳）

M：コードブックサイズ

c_i：コード（符号），$i = 1, \cdots, M$

$\boldsymbol{v}_i$：代表ベクトル，$i = 1, \cdots, M$

第 i クラスターの歪

$$E_i = \frac{1}{|K_i|} \sum_{\boldsymbol{x} \in K_i} (\boldsymbol{x} - \boldsymbol{v})$$

K_i：$\boldsymbol{v}_i$ のクラスター

総歪 $$E = \sum_{i=1}^{M} E_i$$

線形予測符号化法 Linear Predictive Coding

$$x_t = -\sum_{n=1}^{p} \alpha_n x_{t-n} + Gu_t$$

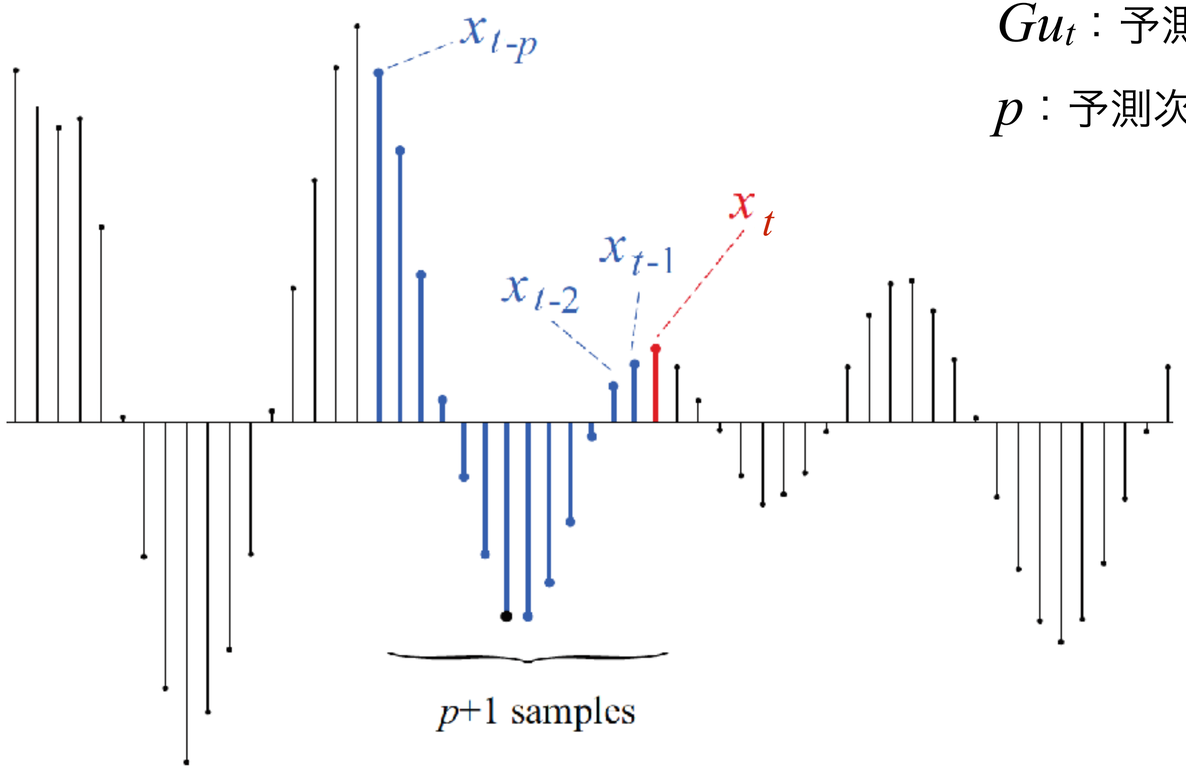
離散信号の値を過去の標本値の線形結合として予測

x_t：時刻 t の標本値

α_n：係数

Gu_t：予測誤差（Gは係数）：音源情報

p：予測次数



線形予測符号化法　Linear Predictive Coding

$$x_t = -\sum_{n=1}^{p} \alpha_n x_{t-n} + G u_t$$

伝送データ：$\alpha_n, n = 1, \ldots p,\ Gu_t$

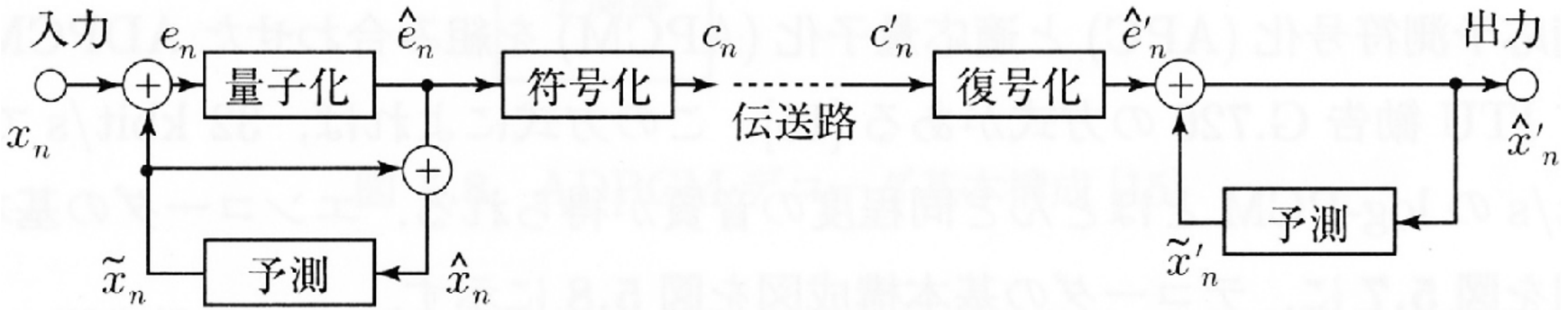
- 10ms程度の音声サンプルからα_nとGu_tを推定
- サンプルそのものの伝送より送信データが少
  - 16kHzサンプリング、16bit量子化の場合
    - サンプル伝送：10ms × 16 × 16bit × 100 = 256kbps
    - LPC（p=8）：（8 × 8bit ＋ 10ms ×16× 8bit）× 100 = 134.4kbps

α_n フィルタ　　Gu_t 音源

©板橋秀一編著，音声工学、森北出版、2005年。

適応予測符号化（APC） Adaptive Predictive Coding

- 音声信号は隣接するものから数点離れた標本間でも大きな相関
- **DPCM**（**D**ifferential **PCM**）：隣接標本間の差分を符号化
  - 予測係数（1～3次）の線形予測　$\tilde{x}_t = a_1 x_{t-1} + a_2 x_{t-2} + a_3 x_{t-3}$
  - 予測次数が大 → 差分信号 e の変化幅小 → 少ビット数で量子化可



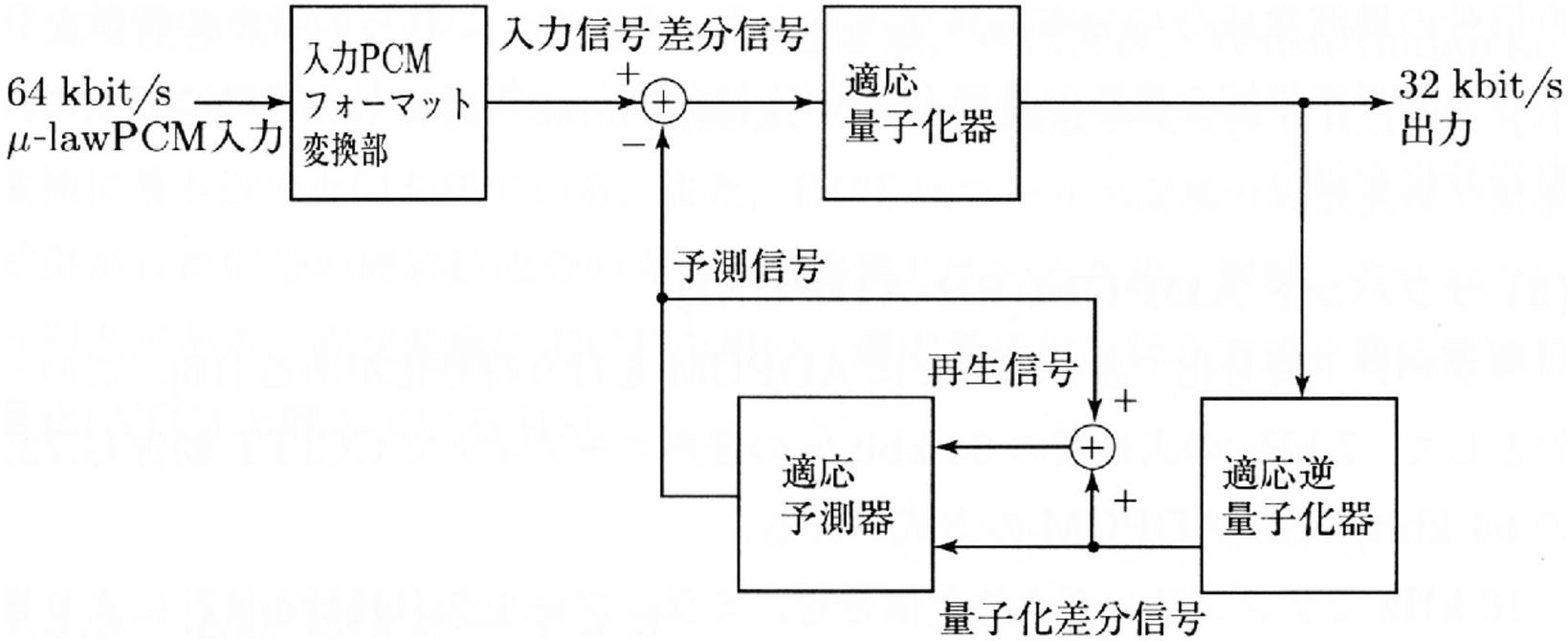
©板橋秀一編著，音声工学、森北出版、2005年。

- APC：予測次数を音声区間毎に変化させる方式

適応差分PCM（ADPCM） Adaptive Differential Pulse Code Modulation

ADPCM符号器（encoder）

国際電話網



©板橋秀一編著，音声工学、森北出版、2005年。

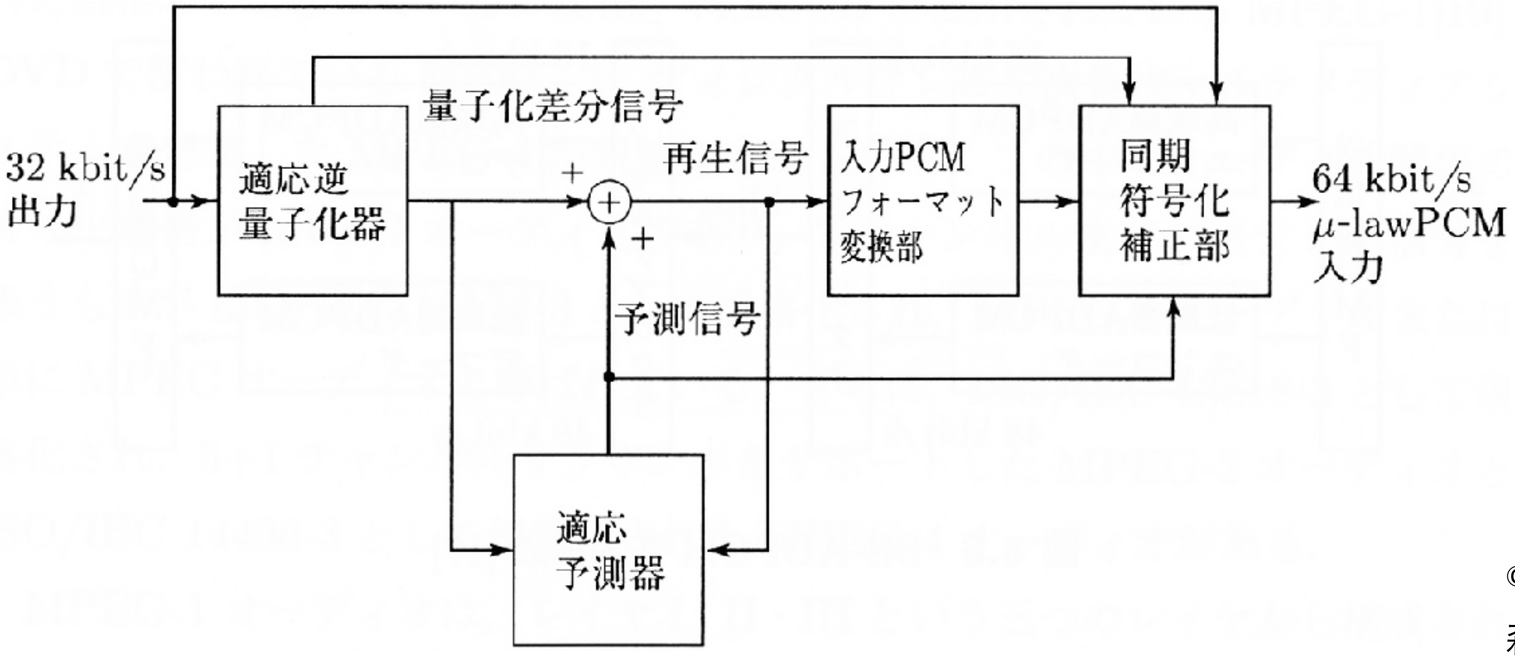
1. μ-lawで量子化されたPCM入力信号を線形PCMに変換
2. 入力信号から入力の予測信号を減算して差分信号（4ビット）を得る
3. フィードバックループで逆量子化＋予測

ITU勧告G.726方式、APC+APCM

適応差分PCM（ADPCM） Adaptive Differential Pulse Code Modulation

ADPCM符号器（encoder）

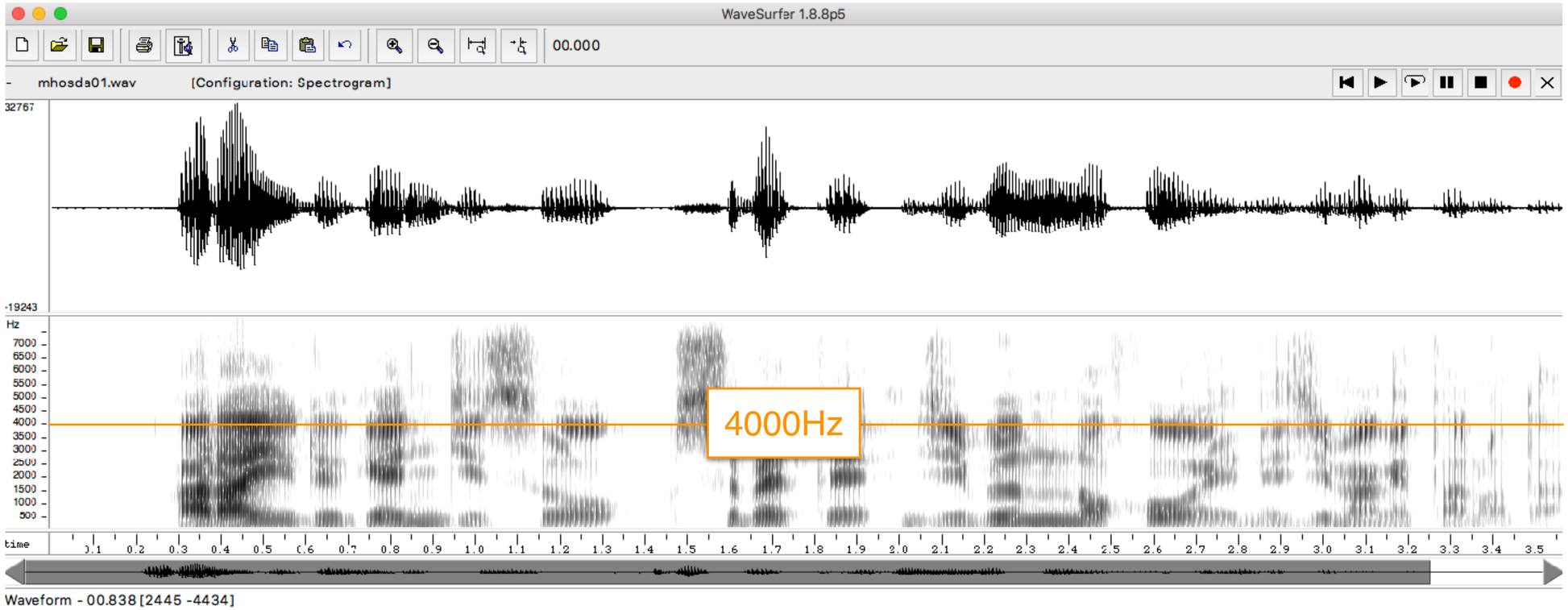
国際電話網



©板橋秀一編著，音声工学、森北出版、2005年。

1. エンコーダのフィードバックループと同構成で信号再生
2. 再生信号はPCM変換部で**μ-law**で量子化されたPCM符号に符号化
3. 「同期符号化補正部」では累積歪の補正

サブバンドADPCM（SB-ADPCM） Sub-Band-ADPCM

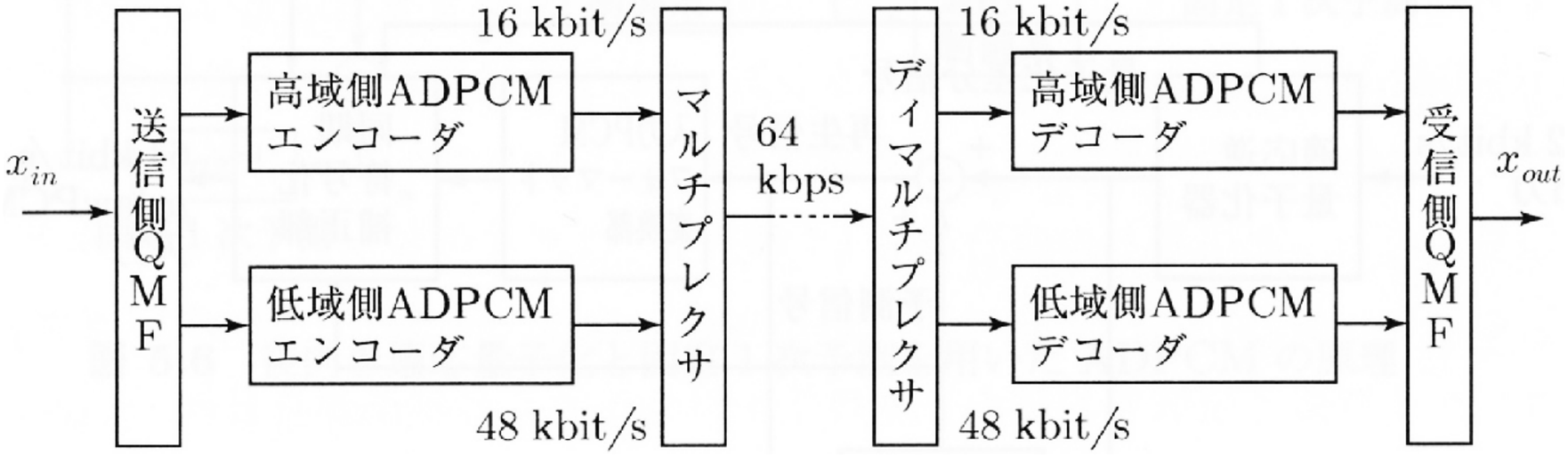


- 音声信号は高域よりも低域の周波数成分が多い
- 時間毎に周波数成分の分布が異なる
- 複数の帯域（band）に分割し、各バンドの特徴に応じて符号化

サブバンドADPCM（SB-ADPCM）

ITU-T勧告G.722 SB-ADPCM

Sub-Band-ADPCM

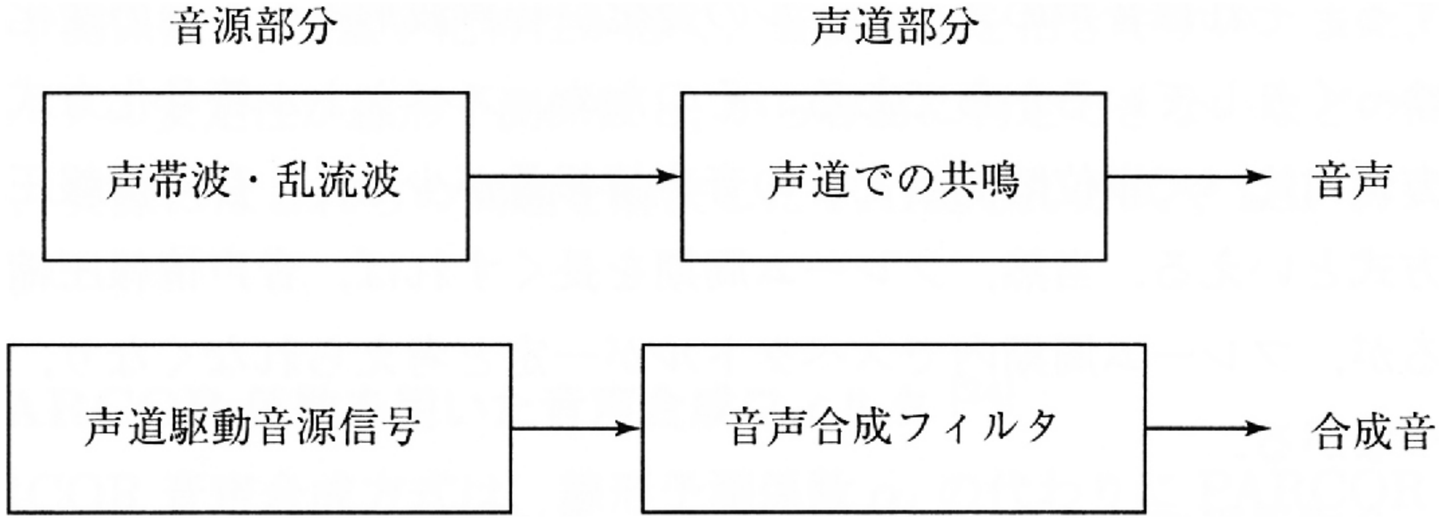


- 低域は0～4 kHz、高域は4～8 kHz
- ADPCMにより低域を48 kbit/s、高域側を16 kbit/s で符号化

©板橋秀一編著，音声工学、森北出版、2005年。

スペクトル符号化方式（分析合成方式）

音声の生成過程をモデル化し，その特徴パラメータを用いて音声を合成

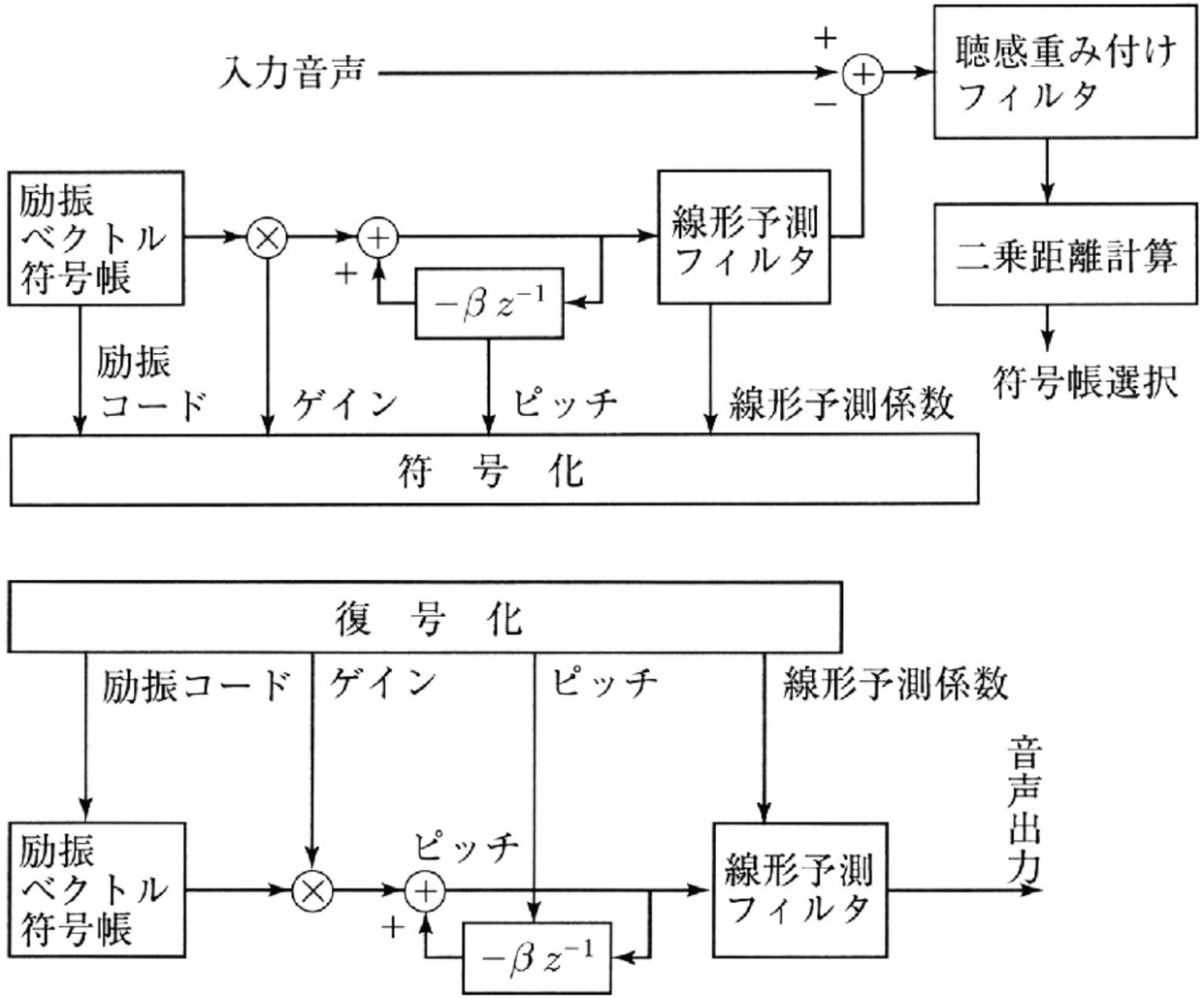


このような考え方で音声を分析・合成する方式：**Vocoder**　W.H.Dudley 1939

©板橋秀一編著，音声工学、森北出版、2005年。

符号励振線形予測法（CELP） Code Excited Linear Prediction

ベクトル量子化された励振信号を用いて線形予測を行う分析合成形の符号化方法。

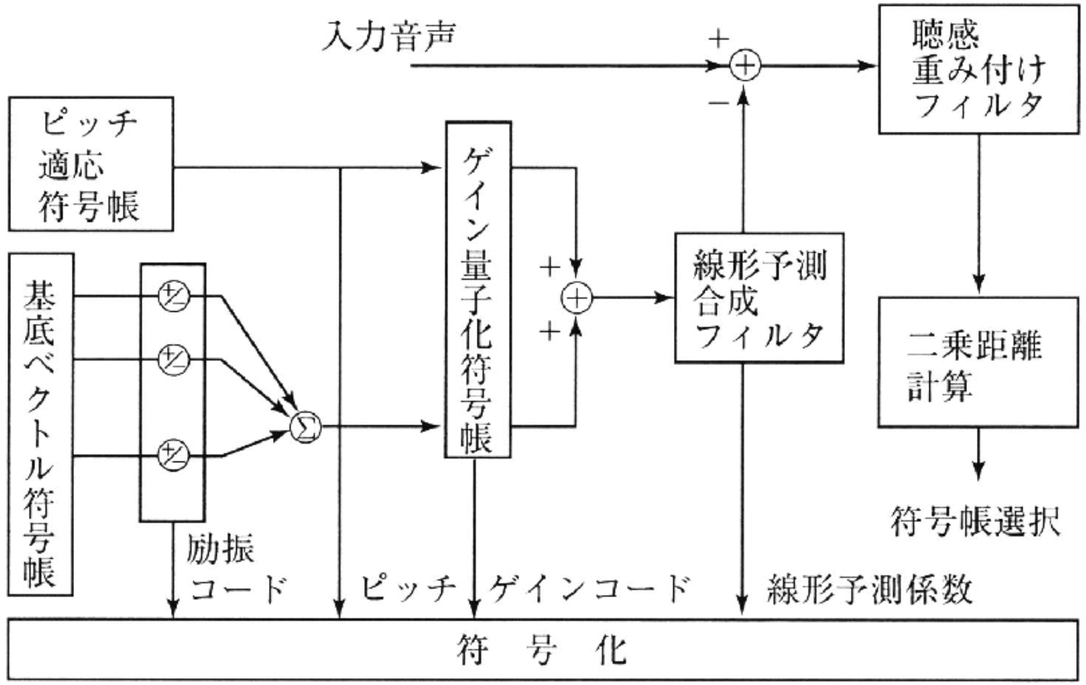


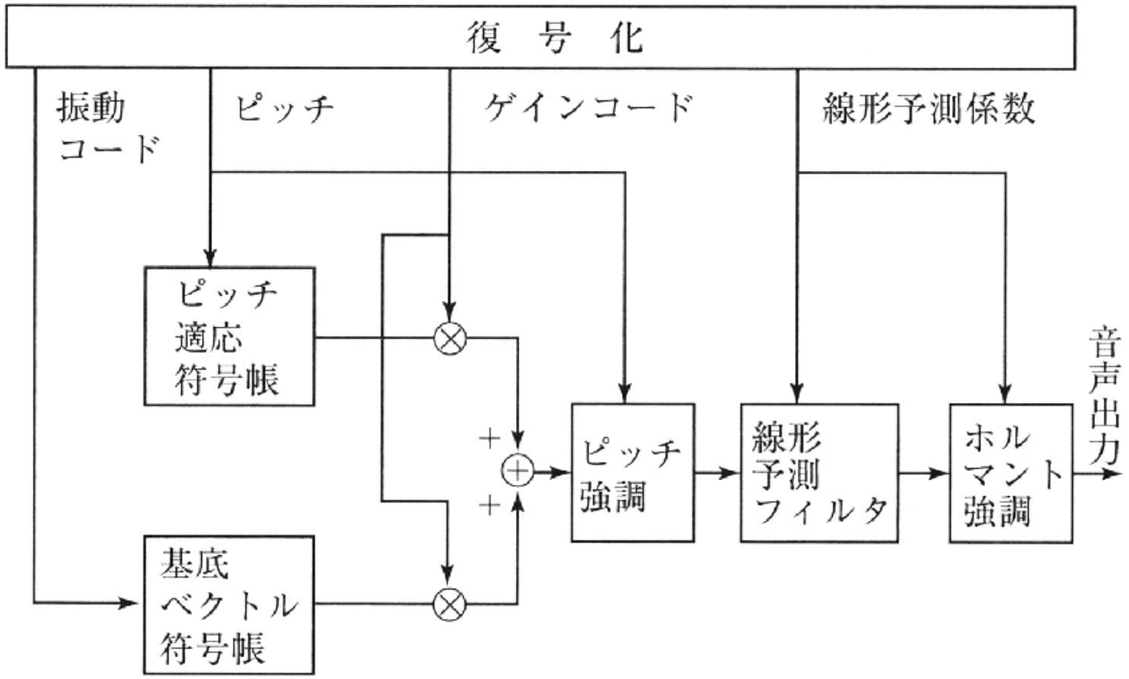
©板橋秀一編著，音声工学、森北出版、2005年。

ベクトル加算励振線形予測法（VSELP）

Vector Sum Excited Linear Prediction

ベクトル量子化されたピッチ周期成分と雑音的成分を混合した励振信号を用いて線形予測を行う分析合成形の符号化方法。米国および日本のディジタル携帯電話用音声符号化標準方式。





©板橋秀一編著，音声工学、森北出版、2005年。

ベクトル加算励振線形予測法（VSELP）

Vector Sum Excited Linear Prediction

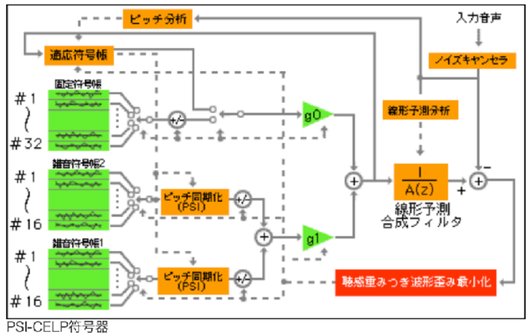
7.VSELPとPSI-CELP

移動無線伝送～7.VSELPとPSI-CELP

VSELPは北米および日本のフルレート方式に採用されているもので、CELP音声符号化方式です。雑音符号帳に工夫を凝らすことにより、必要なメモリ量が少ない、符号誤りの影響が小さい、合成フィルタの演算量が少ない、などの長所もありますが、その反面、量子化雑音が比較的大きいという短所もあります。

PSI-CELPはCELP系の音声符号化方式で、PDCハーフレート方式に採用されています。その大きな特長は、雑音符号帳のピッチ同期化です。ピッチ同期化では、雑音符号帳の先頭から音声の基本周期分だけを取り出して、繰り返すことにより、忠実に音声を再生します。これにより、VSELPの半分のビットレートで同等の品質を達成することが可能となりました。

また、PSI-CELPでは符号誤りによるひずみを軽減するため、誤り感度に応じて、冗長度の異なる誤り訂正符号化（ビット選別誤り保護）を行っています。



PSI-CELP符号器

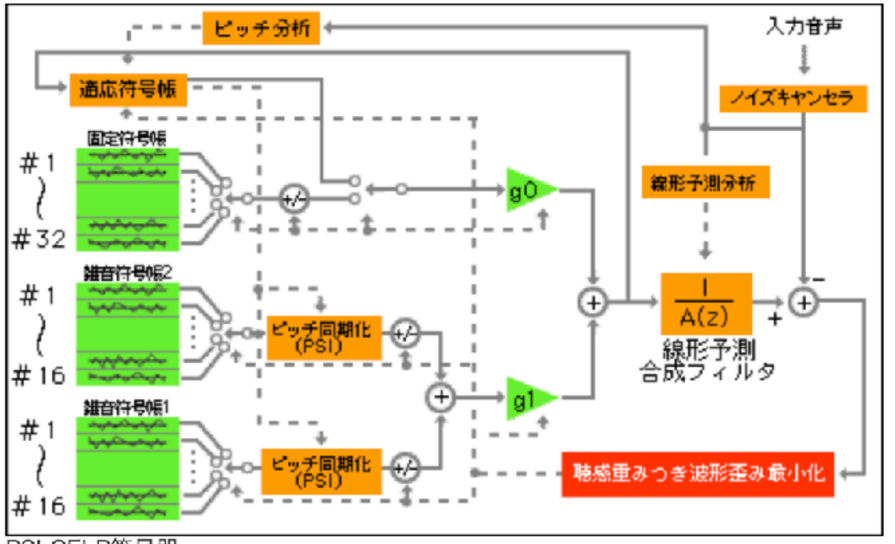
※1999年3月作成

移動無線伝送～7.VSELPとPSI-CELP

VSELPは北米および日本のフルレート方式に採用されているもので、CELP音声符号化方式です。雑音符号帳に工夫を凝らすことにより、必要なメモリ量が少ない、符号誤りの影響が小さい、合成フィルタの演算量が少ない、などの長所もありますが、その反面、量子化雑音が比較的大きいという短所もあります。

PSI-CELPはCELP系の音声符号化方式で、PDCハーフレート方式に採用されています。その大きな特長は、雑音符号帳のピッチ同期化です。ピッチ同期化では、雑音符号帳の先頭から音声の基本周期分だけを取り出して、繰り返すことにより、忠実に音声を再生します。これにより、VSELPの半分のビットレートで同等の品質を達成することが可能となりました。

また、PSI-CELPでは符号誤りによるひずみを軽減するため、誤り感度に応じて、冗長度の異なる誤り訂正符号化（ビット選別誤り保護）を行っています。



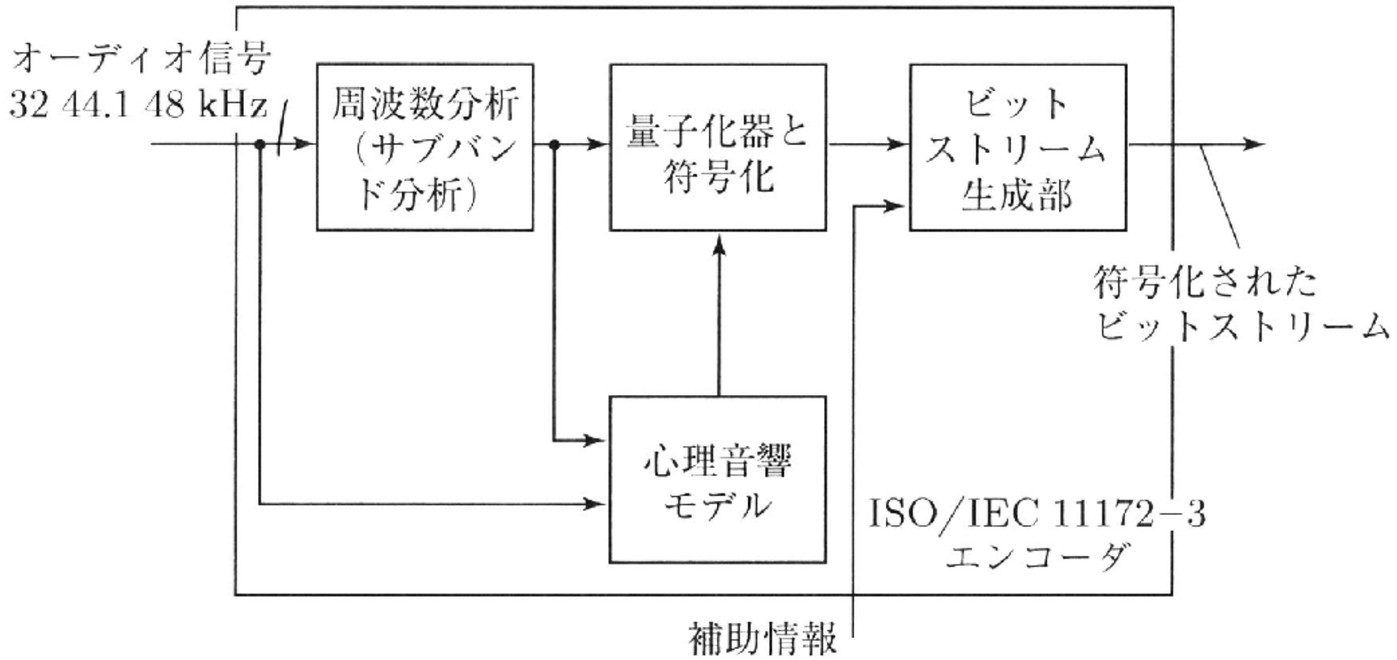
PSI-CELP符号器

© https://www.nttdocomo.co.jp/corporate/technology/rd/tech/network/radio_transmission/07/index.html

MPEGオーディオ

MPEG: Moving Picture Experts Group

動画とオーディオの圧縮符号化規格を作成しているISO/IECの作業部会



©板橋秀一編著，音声工学、森北出版、2005年。

- MPEG-1オーディオ：２チャンネル

  MPEG-2オーディオ：5＋1サラウンド

- MPEG-4オーディオ：多くのオーディオ符号化方式をサポート

静止画像圧縮の原理

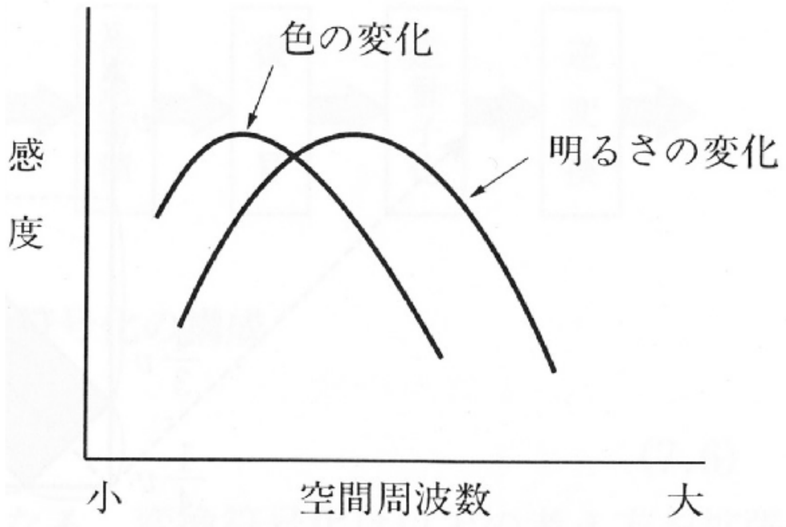
情報の統計的性質の利用

各画素の情報は統計的に大きな偏りを持つ

（例）隣接画素の色や明るさは近いことが多い

人間の視覚特性の利用

感度が鋭い部分の情報は正確に、鈍い部分は粗く



MANDRIL
SIDBA標準画像データベース

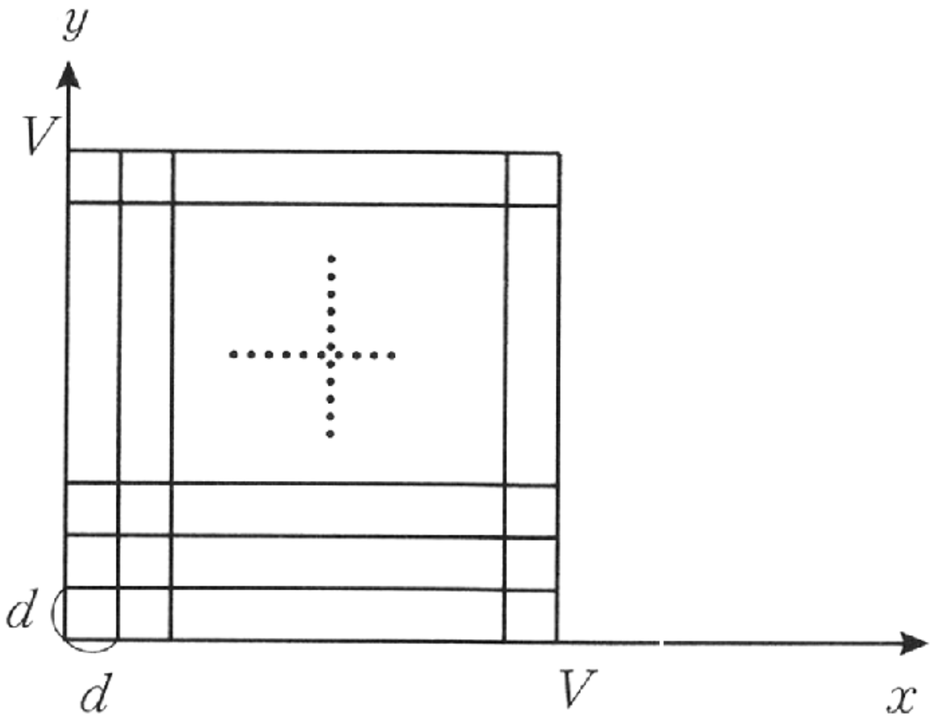
256×256、RGB各8ビット

=256×256×3×8

=1,572,864 bit=192 KB

©古井貞熙・酒井義則著，画像・音声処理技術、電波新聞社、2004年。

変換符号化の有効性



- 近接画素の色と明るさは相関が大
- x, y：近接画素の明るさ，　$0 \leqq x, y \leqq V$
- 画素の明るさを間隔 d で量子化
- $(V/d)^2$の矩形領域を表現するのに必要な

  ビット数：　$\log_2 (V/d)^2$

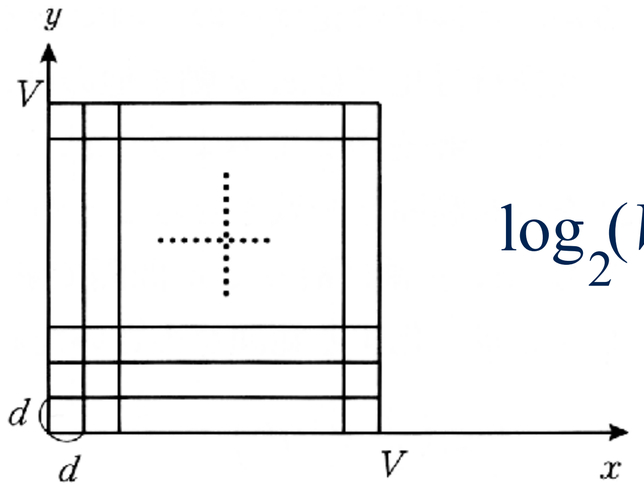
©古井貞熙・酒井義則著，画像・音声処理技術、電波新聞社、2004年。

x, y が一辺 V の正方形内に均一に分布するなら

このビット数に無駄はない…

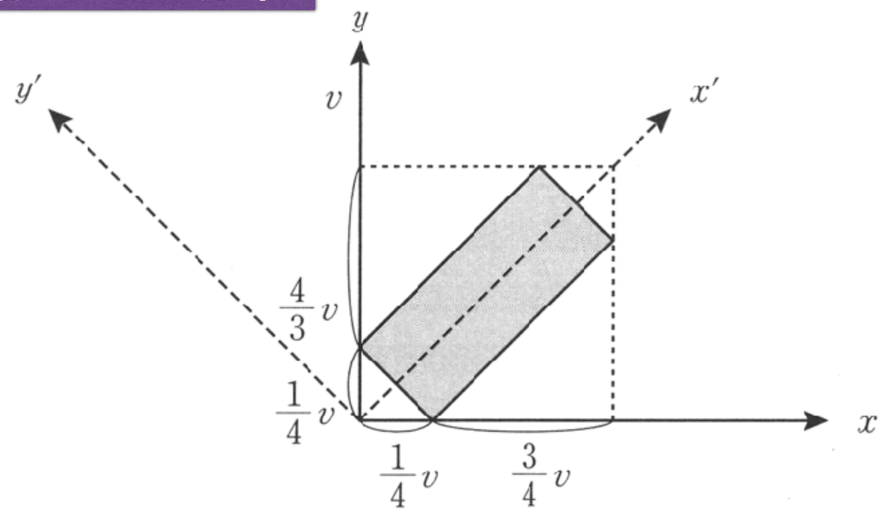
変換符号化の有効性　軸変換して無駄を減らす

$0 \le x, y \le V$ の範囲で画素の明るさを間隔 d で量子化する場合に必要なビット数は...

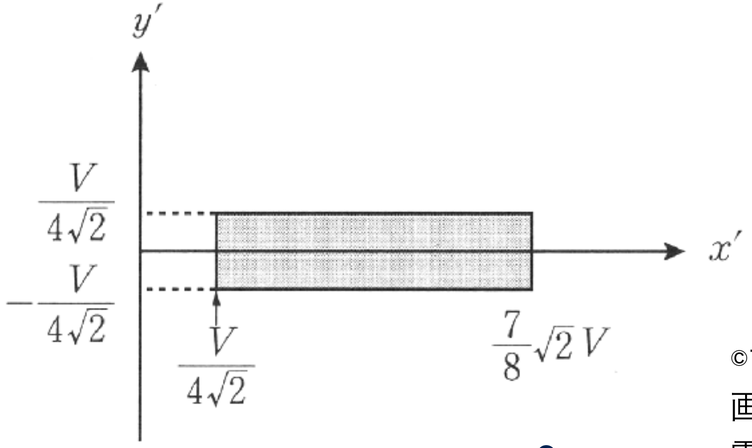


$$\log_2 (V/d)^2$$

偏りのある分布



変数変換後の量子化範囲



$$\log_2 \frac{3}{8} (V/d)^2$$

x と y を別々に量子化するのは無駄（空白領域がある）

©古井貞熙・酒井義則著, 画像・音声処理技術、電波新聞社、2004年。

変換符号化

・画素の明るさを $x_1, x_2, \ldots, x_n$ で表して

$$y_k = \sum_{j=1}^{N} h_{jk} x_j$$

$$k = 1, \cdots, N$$

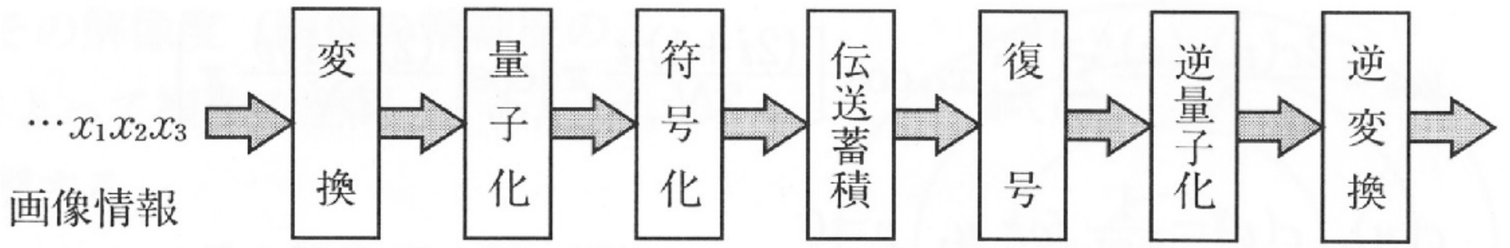
とする。

・h_{jk} をうまく選んで、y_k をできる限り独立となるようにする。

・次に、y_k を量子化して、適当に２値情報に変換する。

画像には離散コサイン変換が適している。
静止画像の国際標準のJPEGで採用。

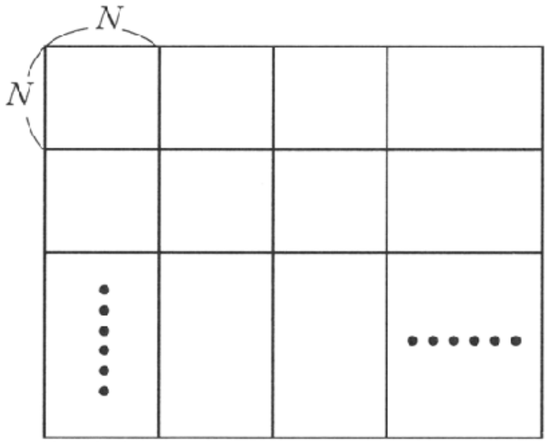
・復号側では、再び y_k を求め、逆変換することにより x_k を求める。



©古井貞熙・酒井義則著，画像・音声処理技術、電波新聞社、2004年。

静止画像に対する離散コサイン変換

DCT Discrete Cosine Transform



x_{ij} ： $N \times N$ のブロック内の画素の明るさ

$0 \leqq i,\ j \leqq N-1$

y_{uv} ： DCT係数、$0 \leqq u, v \leqq N-1$

$$y_{uv}=\frac{2c(u)c(v)}{N}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}x_{ij}\cos\left[\frac{(2i+1)u}{2N}\pi\right]\cos\left[\frac{(2j+1)v}{2N}\pi\right]$$

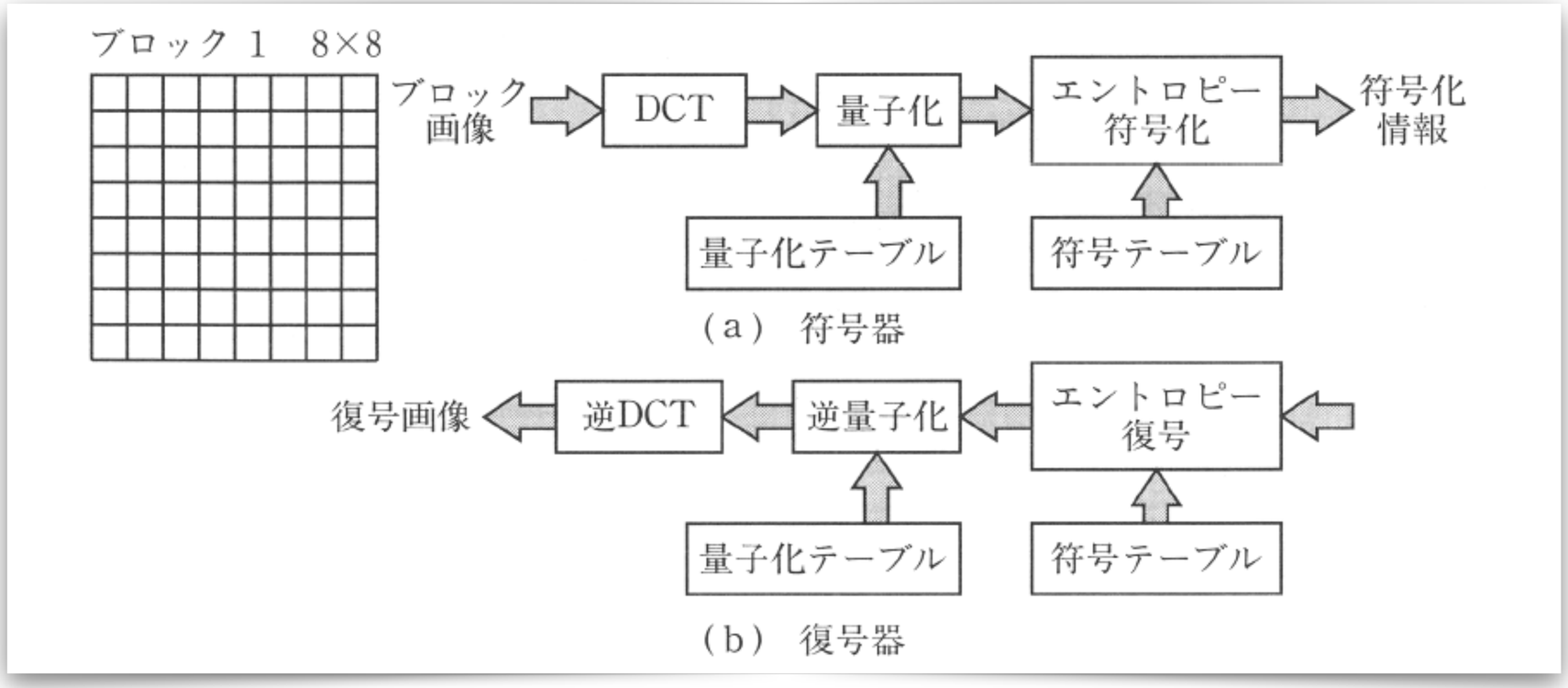
$$c(u),\ \ c(v)=\frac{1}{\sqrt{2}}\ for\ u,\ \ v=0$$

$$1\quad else$$

©古井貞熙・酒井義則著,
画像・音声処理技術、
電波新聞社、2004年。

静止画像の符号化：JPEG

Joint Photographic Experts Group



JPEGの基本構成

©古井貞熙・酒井義則著,
画像・音声処理技術、
電波新聞社、2004年。

ISO/IECによって定められた。最初の標準は1992年。

DCT係数と量子化テーブル

(a) DCT 係数 y_{uv}

	0	1	2	3	4	5	6	7
0	23	120	42	20	10	7	1	−2
1	60	−2	−15	−10	−3	0	−1	3
2	−7	10	−10	3	3	1	−1	
3	24	−6	8	2	1	−1	2	
4	−7	4	5	−2	1	1	0	
5	5	3	2	−2	0	1	0	
6	5	−2	0	1	0	1	1	
7	−6	−3	0	0	0	0	2	0

(b) 量子化テーブルの例[7]

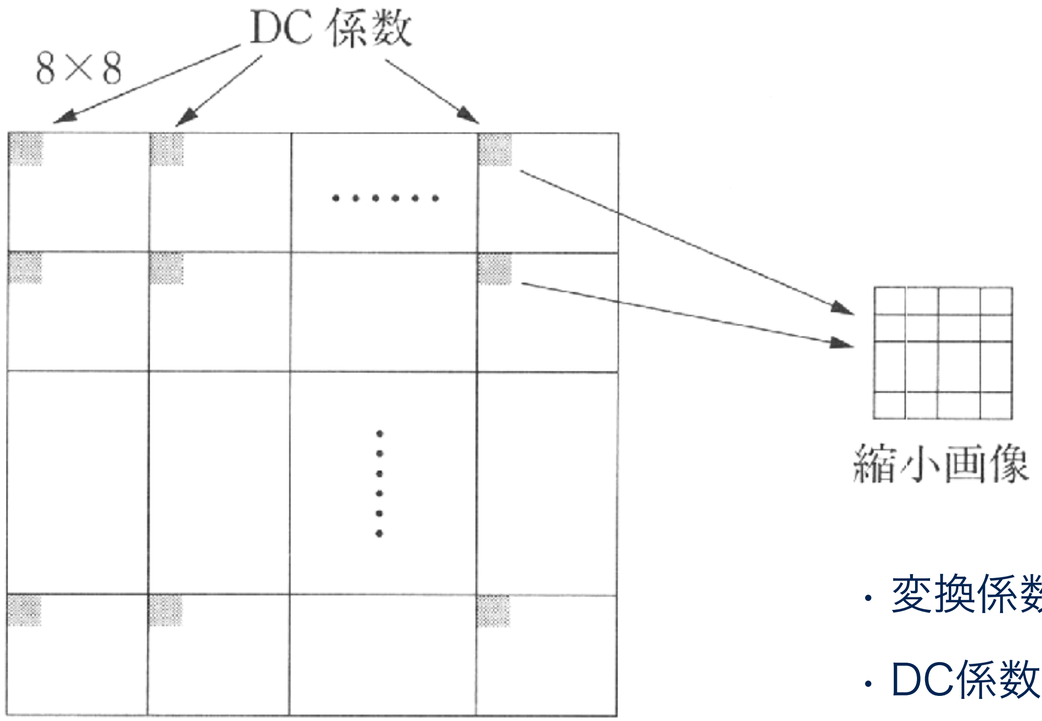
8	6	5	8	12	20	26	30
6	6	7	10	13	29	30	28
7	7	8	12	20	29	35	28
7	9	11	15	26	44	40	31
9	11	19	28	34	55	52	39
12	18	28	32	41	52	57	46
25	32	39	44	52	61	60	51
36	46	48	49	56	50	52	50

(c) 近似後の DCT 係数

29	20	8	3	1	0	0	0
10	−5	−2	−1	0	0	0	0
−1	1	−1	0	0	0	0	0
3	−1	1	0	0	0	0	0
−1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

- 高周波成分は視覚感度が悪い
- 量子化間隔：低周波で小、高周波で大
- DCT係数を量子化テーブルの値で割る

直流成分の符号化



©古井貞熙・酒井義則著,
画像・音声処理技術、
電波新聞社、2004年。

画素値が0〜255ならば

DCT係数の変動範囲は2048以内

- 変換係数の第1行第1列は直流（DC）成分
- DC係数はDCT係数の中でも特に値が大きい
- 他の係数（AC係数、交流成分）と分けて符号化
- 全ブロックのDC係数を集めた1つのブロック
- 縮小画像に対してDPCM（差分パルス符号変調）を適用
- DPCMの値は–2047〜2047

DC係数（直流成分）の符号化

・DC係数の差分値は-2047～+2047の範囲に入るので、ハフマン符号を割り当てる。

（例）DC差分値が–6の場合、グループ識別子は”100”、グループ内付加ビットは”001”

符号としては ”100001”。

表 7-4　DC 係数のハフマン符号

グループ番号	DC 係数の差分値	識別符号（ハフマン符号）	付加ビット数
0	0	00	0
1	−1, 1	010	1
2	−3, −2, 2, 3	011	2
3	−7, −6, −5, −4, 4, 5, 6, 7	100	3
4	−15, ⋯ −8, 8⋯15	101	4
5	−31, ⋯ −16, 16⋯31	110	5
6	−63, ⋯ −32, 32⋯63	1110	6
7	−127, ⋯ −64, 64⋯127	11110	7
8	−255, ⋯ −128, 128⋯255	111110	8
9	−511, ⋯ −256, 256⋯511	1111110	9
10	−1023, ⋯ −512, 512⋯1023	11111110	10
11	−2047, ⋯ −1024, 1024⋯2047	111111110	11

ハフマン符号（エントロピー符号）

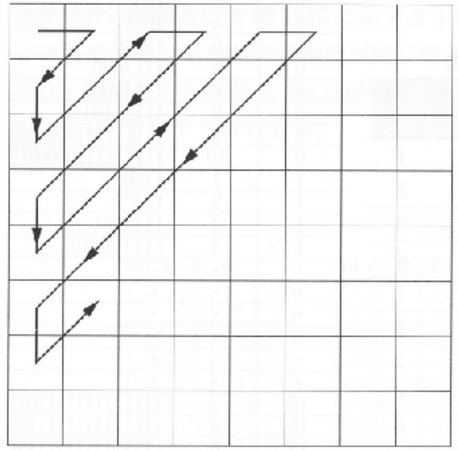
・高頻度記号には短ビット列、
低頻度記号には長ビット列を割り当て。

・メッセージ全体の符号化データ量を削減。

・接頭符号である。

・ある文字に対応するビット列が、他の文字に対応するビット列の接頭辞になることはない。この特徴のおかげで、デコーダはビット列を先頭から一度読むだけで元のメッセージを一意にデコードできる。

AC係数（交流成分）の符号化



7	60	0	7	0	0	0	0
15	−10	0	0	0	0	0	0
6	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

- シグザグスキャン（周波数成分：左上角＝低い、右下角＝高い）
- ブロック内の明るさあるいは色の変化成分
- 変化成分の小さなブロックでは”0”が多い
  - “0”の連続はその連続の長さを符号化
  - 非 ”0” 成分（有効係数）は数値を符号化

1. run length 0 と有効係数 ”60”
2. run length 0 と有効係数 ”15”
3. run length 0 と有効係数 ”6”
4. run length 0 と有効係数 ”−10”
5. run length 1 と有効係数 ”7”
6. EOB

©古井貞熙・酒井義則著,
画像・音声処理技術、
電波新聞社、2004年。

AC係数（交流成分）の符号化

表 7-5　AC 係数のグループ化

グループ番号	AC 係数	付加ビット数
1	−1, 1	1
2	−3, −2, 2, 3	2
3	−7, −6, −5, −4, 4, 5, 6, 7	3
4	−15, ⋯ −8, 8⋯15	4
5	−31, ⋯ −16, 16⋯31	5
6	−63, ⋯ −32, 32⋯63	6
7	−127, ⋯ −64, 64⋯127	7
8	−255, ⋯ −128, 128⋯255	8
9	−511, ⋯ −256, 256⋯511	9
10	−1023, ⋯ −512, 512⋯1023	10

©古井貞熙・酒井義則著，画像・音声処理技術、電波新聞社、2004年。

表 7-6　AC 係数のハフマン符号

無効係数のランレングス	有効係数のグループ番号	ハフマン符号
0	EOB	1010
0	1	00
0	2	01
0	3	100
0	4	1011
0	5	11010
0	6	1111000
0	7	11111000
0	8	1111110110
0	9	1111111110000010
0	10	1111111110000011
1	1	1100
1	2	11011
1	3	1111001
1	4	111110110
・	・	・
・	・	・
・	・	・
・	・	・
・	・	・
・	・	・
15	7	1111111111111011
15	8	1111111111111100
15	9	1111111111111101
15	10	1111111111111110

AC係数（交流成分）の符号化

7	60	0	7	0	0	0	0
15	−10	0	0	0	0	0	0
6	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

1. run length 0 と有効係数 ”60”
   1. 1111000+111100
2. run length 0 と有効係数 ”15”
   1. 1011+1111
3. run length 0 と有効係数 ”6”
   1. 100+110
4. run length 0 と有効係数 ”−10”
   1. 1011+0101
5. run length 1 と有効係数 ”7”
   1. 111001+111
6. EOB
   1. 1010

111100011110010111111110011010110101111001111 1010

(48 bit)

色情報の符号化

(1) カラー画像の色情報

$$Y = 0.299R + 0.587G + 0.114B$$ ：明るさ

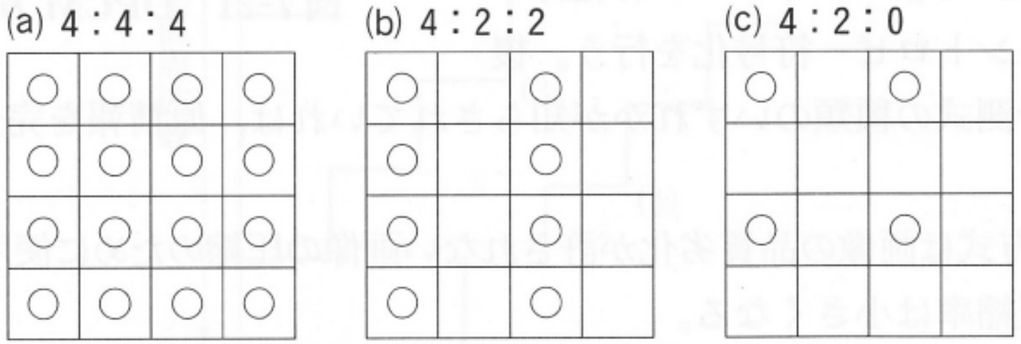
$$C_r = 0.169R - 0.331G - 0.500B$$ ：色差信号

$$C_b = 0.500R - 0.419G - 0.081B$$ ：色差信号

３成分個別にDCTおよび量子化、エントロピー符号化

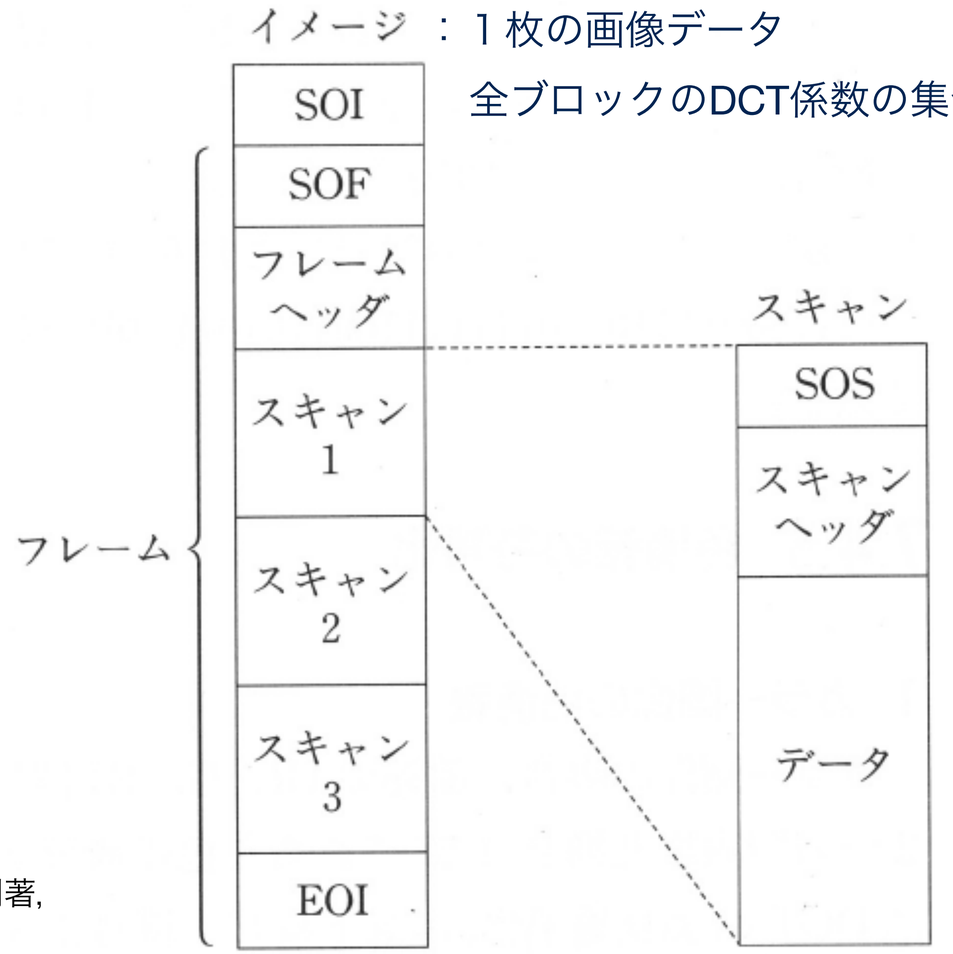
(2) 色差成分の解像度

明るさ Y に比べて小さくすることが可能 → ３種類の解像度



©古井貞熙・酒井義則著,
画像・音声処理技術、
電波新聞社、2004年。

JPEG基本方式のデータ形式



<<ノンインターリーブ方式>>

©古井貞熙・酒井義則著,
画像・音声処理技術、
電波新聞社、2004年。

```
SOI: Start Of Image
SOF: Start Of Frame
SOS: Start Of Scan
EOI: End Of Scan
```

- 「スキャン」は全ブロックにおけるDCT係数の集合を表します。
- スキャンにおける色差信号の送り方には、「インターリーブ」と「ノンインターリーブ」の２種類があります。
- 「ノンインターリーブ」では色差信号は別のスキャンになり、「インターリーブ」では同一スキャンとして送られます。

動画像の符号化：MPEG

Moving **P**icture **E**xperts **G**roup

ISO/IECが定めたマルチメディア情報符号化の国際標準

- **MPEG-1**
  - 1.5Mb/s程度までの比較的低ビットレートの動画像
- **MPEG-2**
  - 標準TV、HDTVを含む広い応用範囲の動画像
- **MPEG-4**
  - 次世代携帯端末、動画像の編集機能等多用途
- **MPEG-7**
  - 動画検索が主目的のメタデータ表現機能の標準

動画像の構成

- 30（29.97）枚／秒のフレーム
- 1フレームは525本（標準）、1125本（ハイビジョン）
- 各フレームは静止画と同様、横×縦の画素より構成される
- **インターレス方式**：1フレームが２フィールドから構成される

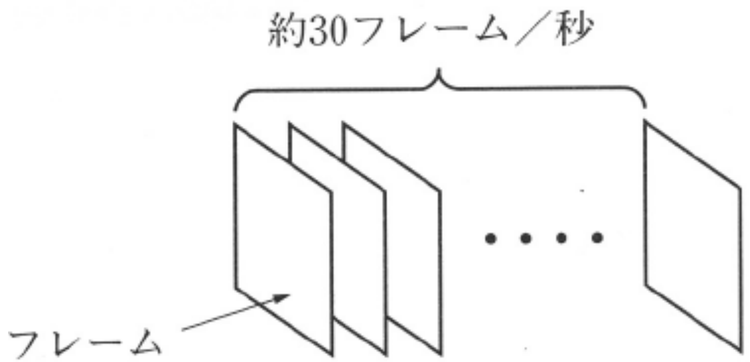


図 9-1　テレビジョンの構成

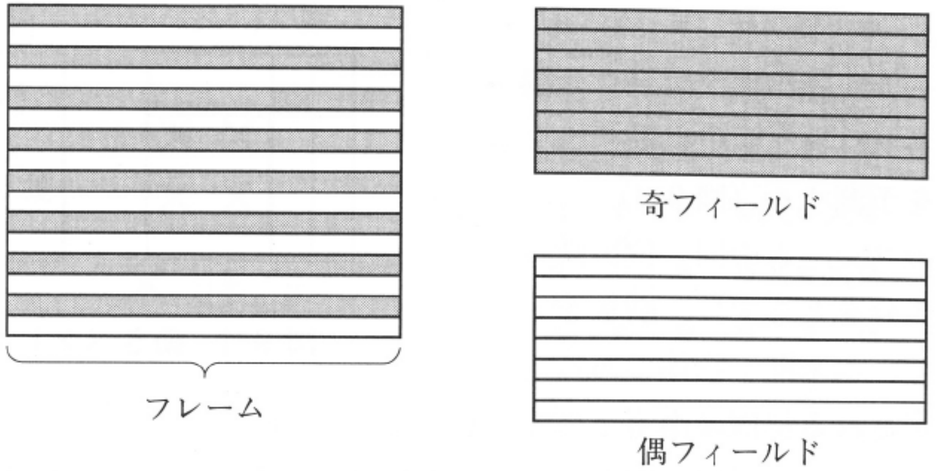


図 9-2　フレームの構造

©古井貞熙・酒井義則著,
画像・音声処理技術、
電波新聞社、2004年。

MPEGの基本技術：フレーム間予測符号化

- 前後フレームの情報を利用
- フレーム予測
  - 前フレームの同一位置画素
    との差分を符号化
  - 予測符号化

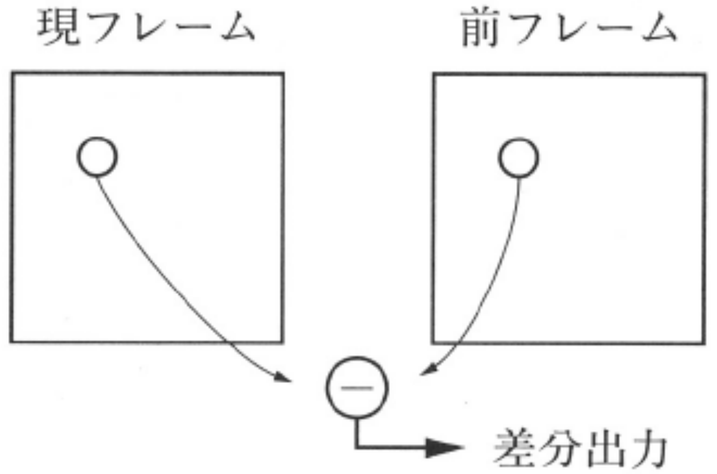


図 9-3　フレーム間予測符号化

MPEGではさらにこの技術を発展させた
「補償フレーム間予測符号化」を採用

MPEGの基本技術：動き補償予測符号化

Motion Compensation Prediction Coding

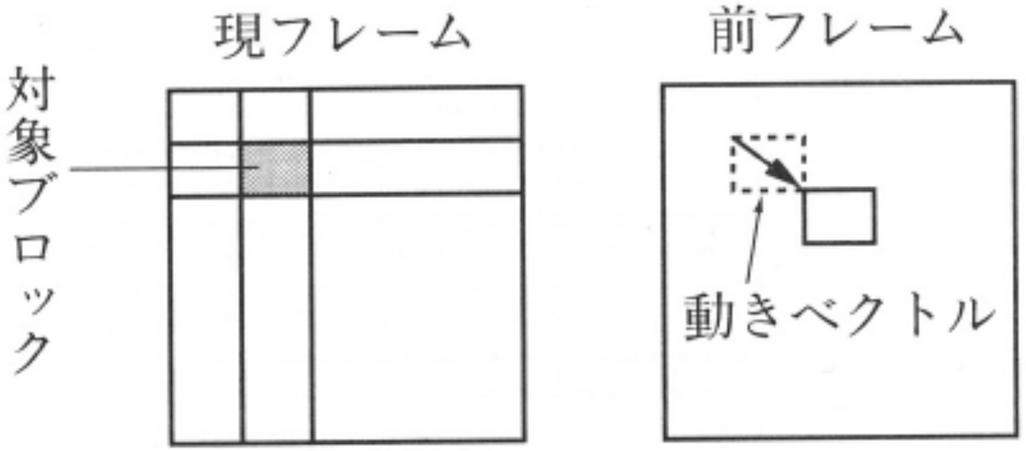


図 9-4　動きベクトル予測

©古井貞熙・酒井義則著，画像・音声処理技術、電波新聞社、2004年。

1. フレームをブロックに分割。
2. ブロック単位に動きベクトル予測
3. 前後フレームの同一位置画素との差分をとらず、動きを考慮して本来物体であろうと推定される位置の画素との差分をとる。
4. 画素差分とともに動きベクトルも符号化。
5. **参照フレーム**：動きを考慮して差分を取るフレーム

参照フレームは必ずしも直前のフレームではない（MPEGの場合）。動き予測を行う場合、

・動きベクトルのとり得る範囲をどの程度考慮するか？

・どのように動きベクトルを求めるか？

の２点が重要となる。

ブロックマッチング法

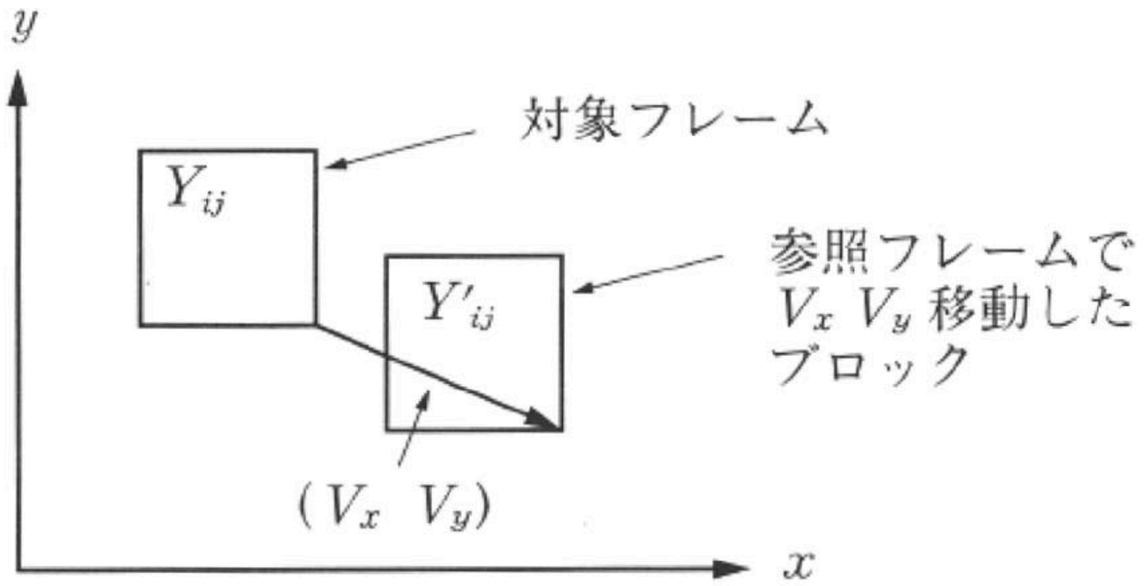


図 9-5　ブロックマッチング

符号化対象フレームのブロック内の画素値と、

参照フレーム内で当該ブロックを平行移動（V_x , V_y）したブロックの画素の値との差分合計をとり、

これが最小となる動きベクトルを求める。

対象フレーム画素：$Y_{i,j}$

参照フレーム画素：$Y'_{i,j}$

ブロック内画素数：$N \times N$

$$\underset{V_x\ ,\ V_y}{\operatorname{argmin}} \sum_{i=1}^{N}\sum_{j=1}^{N} \left| Y_{i,j} - Y'_{i+Vx,j+Vy} \right|$$

ブロックマッチング法 対象画素と参照画素の位置関係

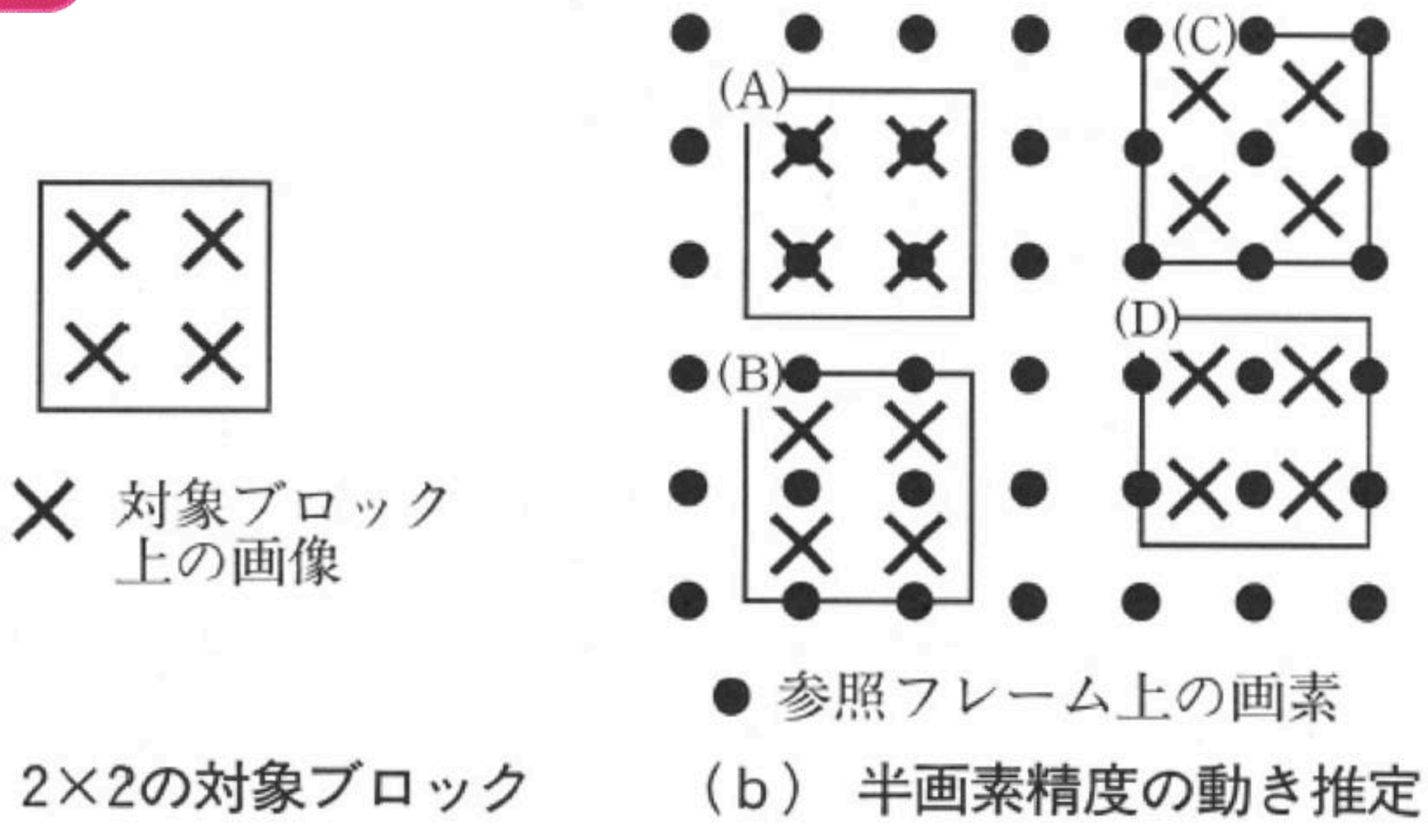


図 9-6 動き推定精度と画素配置[1]

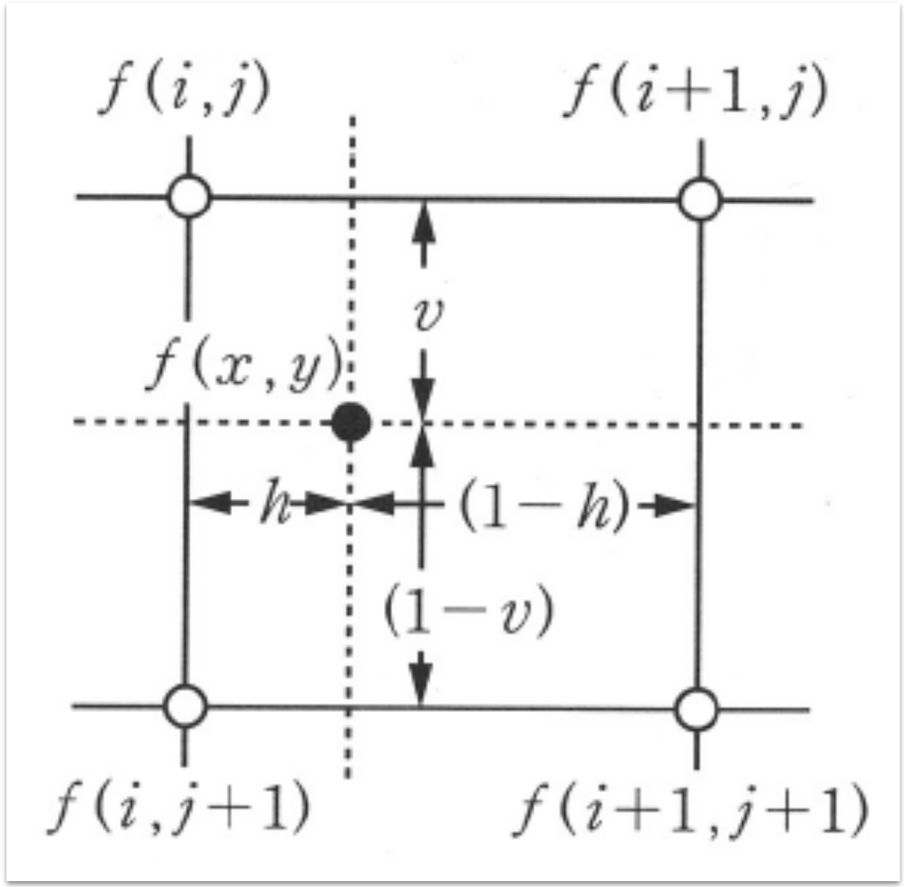
説明のためのこの例では、動きベクトルを求める単位となるブロックサイズは2×2。

図(b)の(A)では、動きベクトルの単位とは画素の整数倍。

図(b)の(B) (C) (D)では2分の1単位の動きベクトルの計算を想定。

このような場合は、実際に存在する画素の中間の画素値を補間して求める必要がある。

双一次画素補間 — 画素値の補間



・双一次補完を用いて画素の中間の画素値を補間

・補間点 (x, y) についての x の少数部分を h 、y の少数部分を v とする。周辺画素を用いて

$$f(x,\ y)=(1-h)(1-v)\cdot f(i,\ j)+h(1-v)\cdot f(i+1,\ j)$$
$$+(1-h)v\cdot f(i,\ j+1)+hv\cdot f(i+1,\ j+1)$$

を求める。

ブロックマッチングは単純で精度も良いが、あくまでも求めているのが画素の差分が最小となる点で、必ずしも正しい動きベクトルではないことに注意。

©古井貞熙・酒井義則著，画像・音声処理技術、電波新聞社、2004年。

MPEGのピクチャ構成

Ｉピクチャ　1枚の静止画として符号化

Pピクチャ　直前のＰピクチャあるいはＩピクチャを参照してフレーム間動き予測

Bピクチャ　前後のＩピクチャおよびＰピクチャを参照してフレーム間動き予測

【理由】予測誤り波及範囲の限定、早回し・逆再生対応

早回し：Ｉピクチャだけ、あるいはＩとＰピクチャだけ復号
逆回し：Ｉピクチャだけを逆に復号

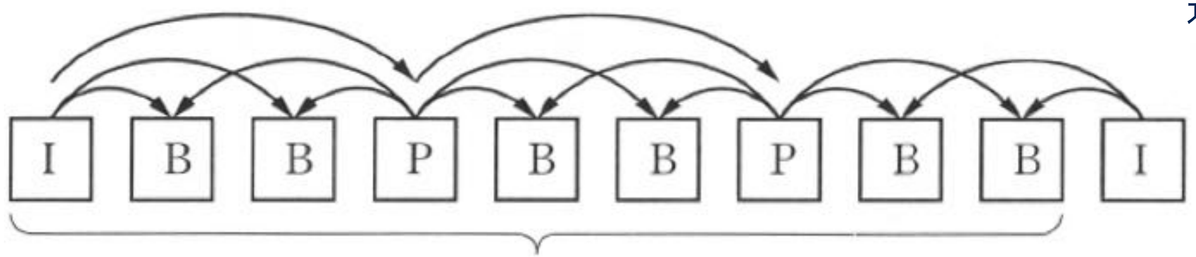


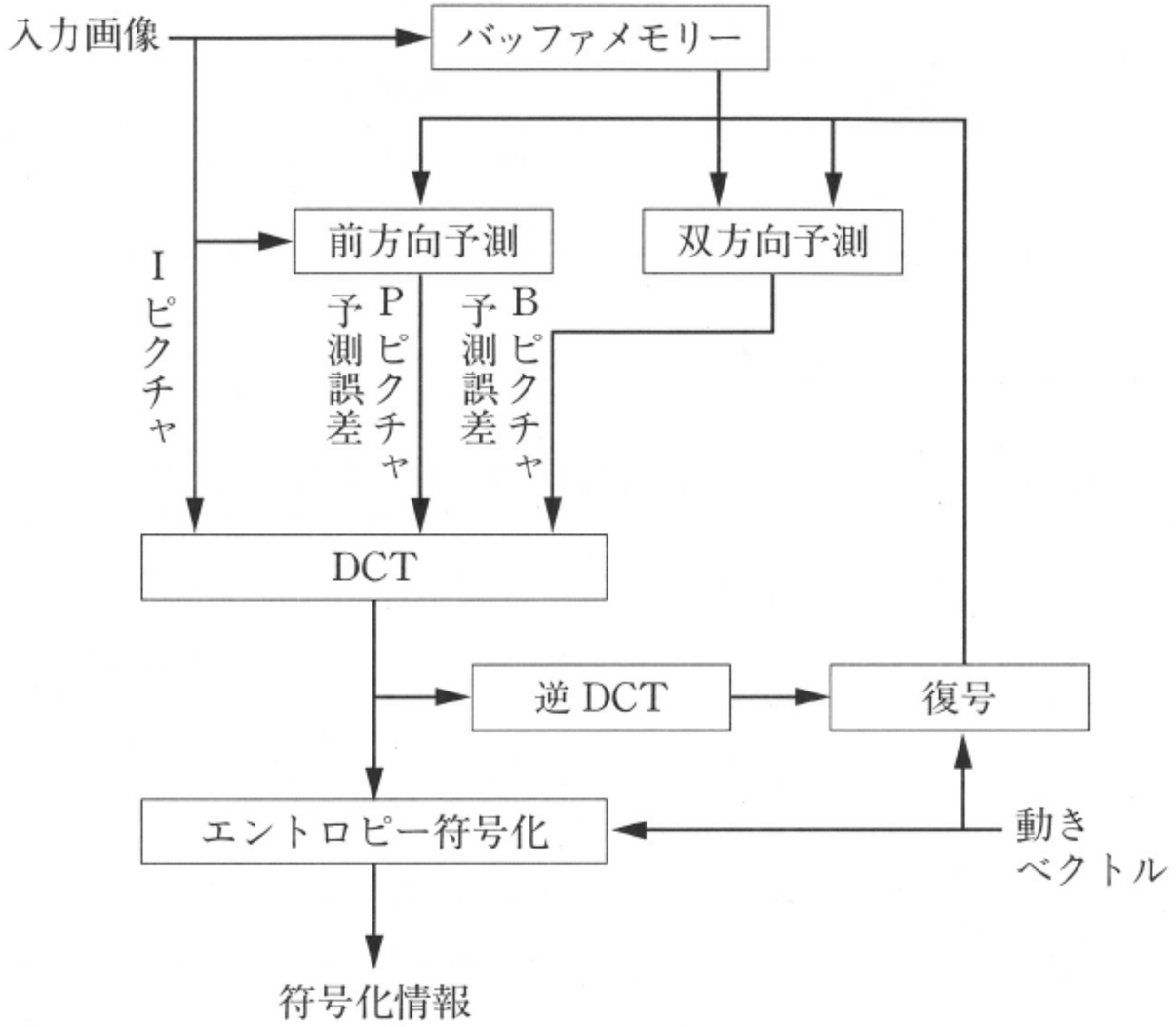
図 9-8　MPEG の構成例

Group **O**f **P**icture

©古井貞熙・酒井義則著，画像・音声処理技術、　電波新聞社、2004年。

MPEG符号器の基本構成

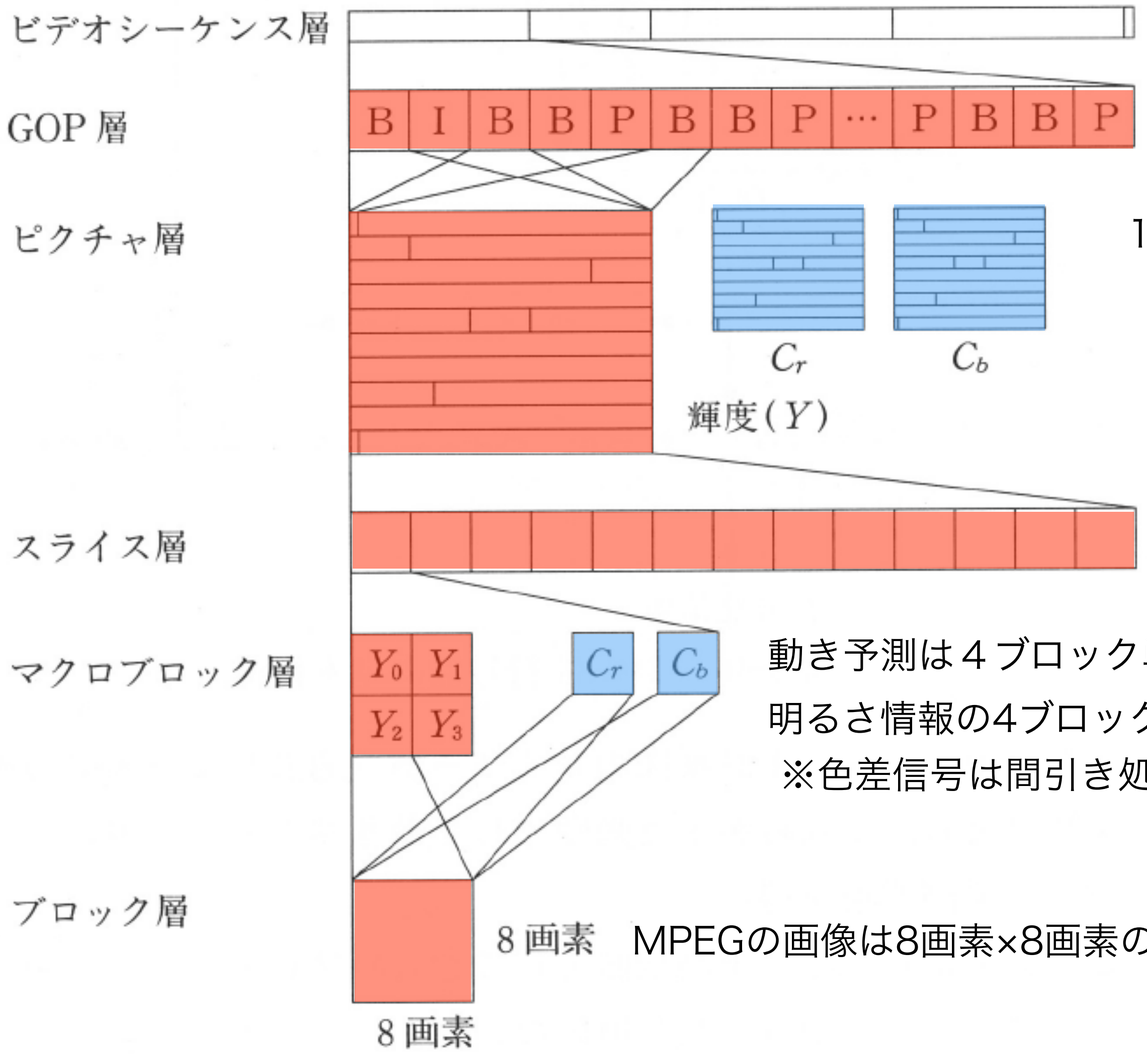
JPEGに「動き補償フレーム間予測符号化」を付加した構成



©古井貞熙・酒井義則著，画像・音声処理技術、　電波新聞社、2004年。

- BピクチャとPピクチャについてはブロック単位に参照ピクチャからの動きベクトルを求める。
- Bピクチャを符号化する場合、バッファメモリーから符号化すべきフレームを呼び出し、次に予測画像用メモリーから過去および未来の参照フレームを呼び出し、これらを予測画像として動きベクトルを用いて双方向予測を行い、予測誤差を得る。
- Pピクチャに対しては予測画像として過去のフレームのみを使用。
- Iピクチャに対しては予測画像を用いない。
- 予測誤差情報とともにPピクチャ，Bピクチャの動きベクトルをエントロピー符号化して送る。

MPEG-1のデータ構造



Iピクチャを含む複数のピクチャ
GOPが構成されている。

１ピクチャは複数のスライスからなる

マクロブロックの集合【スライス】

動き予測は４ブロック単位で処理【マクロブロック】
明るさ情報の4ブロックが色差情報の1ブロック
※色差信号は間引き処理されているため（p.41）

MPEGの画像は8画素×8画素の単位で処理【ブロック】

©古井貞熙・酒井義則著,
画像・音声処理技術、
電波新聞社、2004年。

マルチメディアデータの符号化とファイル形式

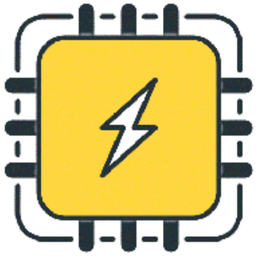
本日の要点

1. マルチメディアデータの符号化は情報の統計的性質と人間の感覚特性を利用。

2. 音声信号は、時間周波数領域で予測適応処理を行い、さらに分析合成方式を用いることにより、高い圧縮率を実現。

3. 静止画像は、人間の視覚特性を利用し、ブロック単位で周波数分析を行うことによって符号化。

4. 動画像は、画像の動きを予測することにより、高い圧縮率でも綺麗な動画を実現し、様々な再生操作に対応。

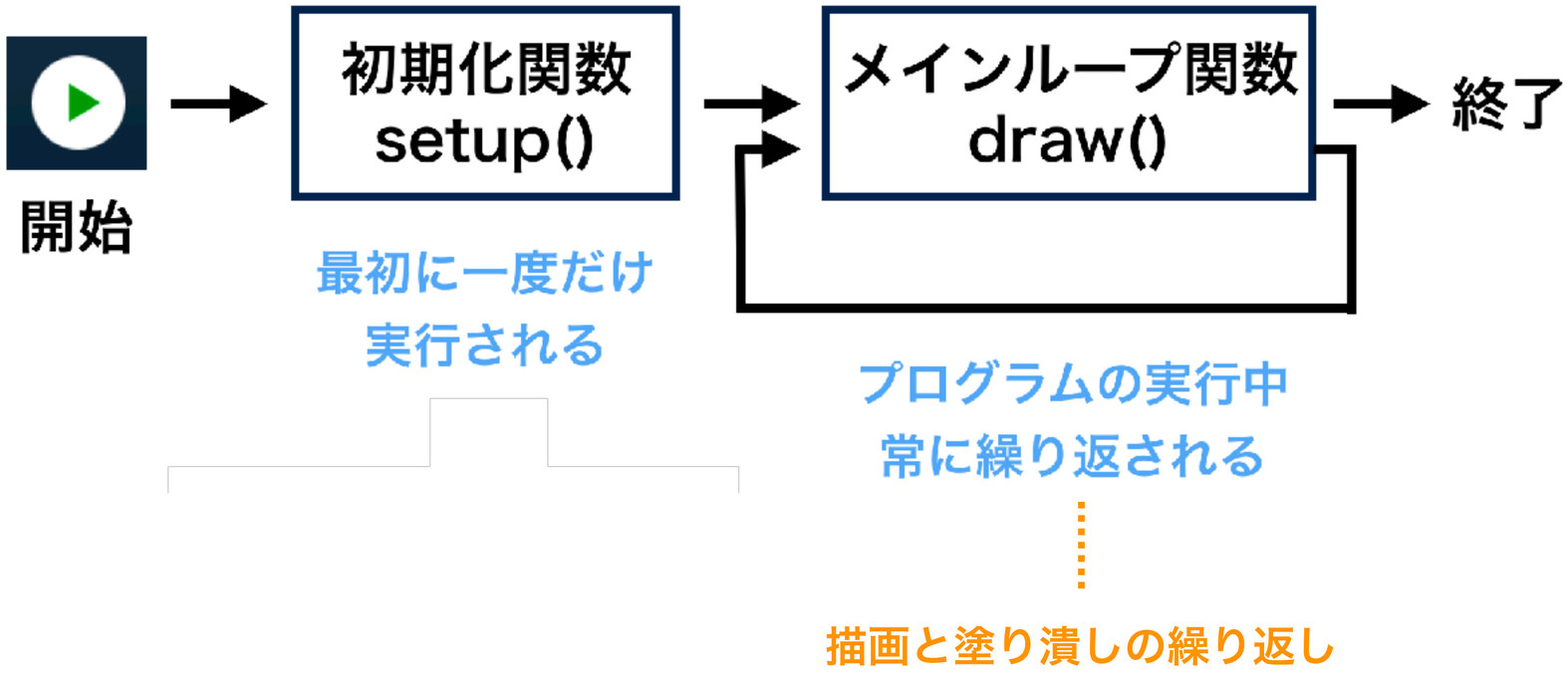


Processingでアニメーション

パラパラマンガ

表示する図形の位置や大きさなどを少しづつ変化させる

特定のタイミングで適切な命令を実行するような仕組みが必要



Processingでアニメーション

1. 時間間隔を置いて図形を描く　円を浮かび上がらせる

```
void setup() {
  size(800, 600);
  frameRate(60);
  background(0);
  noStroke();
}

void draw() {
  float x = random(0, width);
  float y = random(0, height);
  float dist = dist(x, y, width/2, height/2);
    if(dist < height/2) {
     noStroke();
     fill(63, 127, 255);
  }
  else {
    noFill();
    stroke(31, 127, 255);
  }

  ellipse(x,y,10,10);
}
```

frameRate()

フレームレート（毎秒表示されるフレーム数）= 60

円の中心 (x, y) は乱数で決定

円の中心 (x, y) が画面の中心から高さの半分以内の距離にあれば塗りつぶす

外側の円は塗りつぶさず縁だけ描く

©田所淳著，Processingクリエイティブ・コーディング入門—コードが生み出す創造表現、技術評論社、2017年。

Processingでアニメーション

2. 位置によるアニメーション　円をバウンドさせる

```
void setup() {
  size(800, 600);
  frameRate(60);
  fill(0, 127, 255);
  noStroke();
  locationX = 0;
  direction = -1;
}

void draw() {
  background(0);
  ellipse(locationX, height/2, 20, 20);
  locationX = locationX + 10*direction;
  if(locationX < 0 || locationX > width) {
    direction = direction * -1;
  }
}
```

円中心のX座標

円の動く方向（座標積算方向）

画面を黒で塗り潰す

画面の中央を左右に移動

１フレームで10ピクセル移動

左端、あるいは右端を超えたら方向を逆転

Processingでアニメーション

3. 大きさによるアニメーション　円の大きさと色を変化させる

colorMode(), frameCount

```
float diameter;
float hue;

void setup() {
  size(800, 600);
  frameRate(60);
  colorMode(HSB, 360, 100, 100, 100);
  noStroke();
}

void draw() {
  background(0);
  diameter = 400 + sin(frameCount*0.1)*200;
  hue = sin(frameCount*0.1)*120;
  fill(hue, 100, 100);
  ellipse(width/2, height/2, diameter, diameter);
}
```

色指定をHSBモードに設定
- Hueの変化範囲は [0, 360]
- Saturationは [0, 100]
- Brightnessは [0, 100]
- α値（透明度）は [0, 100]

frameCountは表示済みフレーム総数を表すシステム変数

Processingでアニメーション

4. 座標変換で図形を動かす　四角形を回転

```
float angle = 0;

void setup() {
  size(800, 600);
  frameRate(60);
  fill(0, 127, 255);
  noStroke();
}

void draw() {
  background(0);
  rotate(angle);
  rectMode(CENTER);
  rect(400, 300, 300, 300);
  angle += 0.1;
}
```

rotate(), rectMode()

画面を黒で塗り潰す
図形を回転する（画面の原点を中心に）
矩形を描く基点を変更する

Processingでアニメーション

5. 座標変換で図形を動かす　任意の座標で回転

```
float angle = 0;

void setup() {
  size(800, 600);
  frameRate(60);
  fill(0, 127, 255);
  noStroke();
}

void draw() {
  background(0);
  translate(width/2, height/2);
  rotate(angle);
  rectMode(CENTER);
  rect(0, 0, 300, 300);
  angle += 0.1;
}
```

translate()

原点を画面の中央に移動

Processingでアニメーション

6. 複数の図形を同時に回転　座標系の保存と復元

一度変更した座標を元に戻さないと、続く変更が蓄積してしまう。

→ pushMatrix(), popMatrix()

```
float angle = 0;

void setup() {
  size(800, 600);
  frameRate(60);
  fill(0, 127, 255);
  noStroke();
}

void draw() {
  background(0);

  pushMatrix();      現在の座標変換行列をスタックに保存
  translate(width/4, height/4);
  rotate(angle);
  rectMode(CENTER);
  rect(0, 0, 100, 100);      小さな正方形
  popMatrix();     元の座標変換行列をスタックから復元

  pushMatrix();
  translate(width/2, height/2);
  rotate(angle);
  rectMode(CENTER);
  rect(0, 0, 200, 200);          大きな正方形
  popMatrix();

  angle += 0.1;
}
```