動画像に適したフラクタル画像圧縮プロセッサのVLSIによる実装

山内英樹[†,††] 武内良典[††] 今井正治[††]

静止画および動画像を効率良くフラクタル画像符号化するプロセッサのVLSIによる実装方法を提案する．本VLSIでは動画像のフラクタル画像符号化をフレーム内のみの情報で行うのではなく，別フレームのフラクタル情報を参照して行う．すなわち，基準フレームの最適ドメインブロックの探索結果を現フレームでの最適ドメインブロックの探索において利用する．これにより，動画像のフラクタル符号化において必要な演算量が大きく減少し，同時に符号量も削減する．また，本アーキテクチャを用いたVLSIはこれまでに報告されているフラクタル符号化VLSIと比べ小さなハードウェアコストでカラー動画の符号化時間の短縮と高圧縮率を可能にした．

VLSI Implementation of Fractal Image Compression Processor for Moving Pictures

Hideki Yamauchi,[†,††] Yoshinori Takeuchi[††] and Masaharu Imai[††]

This paper proposes an efficient VLSI implementation of fractal image coding processor for moving pictures. Proposed VLSI achieves high compression ratios and high speed moving picture coding. The average of compression ratios becomes 2–5 times higher, and the processing time is 10 times faster than those by conventional fractal techniques. Proposed VLSI architecture technique enables real-time encoding of full-motion videos, and the circuit size of VLSI is much smaller than previously proposed fractal processors.

1. はじめに

フラクタル画像圧縮[1)~3)]は画像内に存在する自己相似度の高い別の画像を抽出し，その相似情報を表すアフィン変換のパラメータを用いて画像を符号化する方式である．JPEG方式に匹敵するほどの符号効率と再現性を持ち，さらにブロックノイズやモスキートノイズが発生しない特長がある[4)]．そのうえ，アフィン変換の反復処理だけで画像の復号ができ，画像の拡大も容易に得られるという優れた特長がある[5)]．

しかし，フラクタル画像圧縮は相似度の高い最適画像の探索に膨大な演算量が必要で，符号化の処理時間が長いという欠点がある．たとえば512 × 512画素サイズの符号化に，動作周波数1 GHzのPC（パーソナルコンピュータ）を用いても数秒程度の時間が必要である．

† 三洋電機株式会社研究開発本部マイクロエレクトロニクス研究所
R&D Headquaters, Microelectronics Research Center, SANYO Electric Co., Ltd.
†† 大阪大学大学院基礎工学研究科
Depertment of Informatics and Mathmatical Science, Gratute School of Engineering Science, Osaka University

このため，これまでに高速にフラクタル画像圧縮する専用プロセッサのアーキテクチャがいくつか提案されたが[6)~8)]，ハードウェアコストが大きく，今日の半導体微細加工技術の進歩をもってしても実用化は困難である．この理由により，フラクタル画像圧縮方式は，JPEGやMPEGで用いられるDCT方式と比べ，普及が阻害されている．

このような状況で，PE（Processor Element）の並列化とPEを構成するデータパスの最適化を行い，レンジブロックの画像の特徴によりブロックをいくつかのクラスに分類し，ドメインブロックの探索をレンジブロックと同一なクラスのブロックに限定する構成を有するフラクタル画像圧縮VLSIのアーキテクチャが提案され[9)]，JPEG[10),11)]方式並みのハードウェアコストでフラクタル画像符号ができることが示された．

しかし，このVLSIアーキテクチャは静止画を効率良くフラクタル画像圧縮できるが，動画像の場合には処理時間が長くなり，リアルタイムでの符号化ができない．また，今後の動画像を扱う携帯端末用のVLSIでは，消費電力の低減が必要不可欠であり，画像圧縮の処理を低い動作周波数で実現できるVLSIのアーキテクチャが望まれている．

動画像のフラクタル符号化に関して，過去のフレームの符号化のときに用いたドメインブロックの位置情報を現在のフレームの符号化に用いるアイデアが提案されているが[12),13)]，これらの手法を実装した実現可能な VLSI アーキテクチャの検討はなされていない．

本論文では，動画像を効率良くフラクタル符号化するアルゴリズムの VLSI への実装方法を提案し，定量的な評価を行う．本 VLSI で用いる動画像のフラクタル画像符号化のアルゴリズムとして，フレーム内のみの画像情報で行うのでなく，基準フレームでのフラクタル情報を参照して行う方式を取り入れた．すなわち，基準フレームの最適ドメインの探索結果を現フレームの符号化処理において利用する．このアルゴリズムは，フラクタル符号化において最も計算量の多い range-domain ブロックのマッチング回数を減少させ，符号処理時間を短縮できる可能性がある．このアルゴリズムを VLSI に実装することで，小さなハードウェアコストで動画像をリアルタイムに符号化する効率的なフラクタル画像圧縮プロセッサが実現でき，さらに圧縮率の向上と符号画像の品質向上の可能性がある．

本論文において，2 章ではフラクタル画像符号化の原理を示す．3 章では本 VLSI のアーキテクチャ，4 章では本 VLSI で採用した動画符号化の手法，5 章では VLSI 設計の結果と評価について述べる．6 章はまとめと今後の課題を示す．

2. フラクタル画像符号化の原理

フラクタル画像符号化は Barnsley が提案した反復関数系（IFS: Iterated Function System）[1),2)] に基づいて，Jacquin により自然画像への応用[3)] が行われた．これは任意の初期画像に対して縮小変換を反復的に施して得られる画像が，原画像に近い画像に収束するという原理に基づいている．

2.1 IFS の原理

μ を原画像，ν を再生画像とする．$\omega(\mu)$ は，原画像 μ に対する変換を表し，$d(\mu,\nu)$ を原画像 μ と再生画像 ν 間の距離，すなわち相異度を表すものとする．任意の μ,ν に対して

$$d(w(\mu),\omega(\nu)) \leq s \cdot d(\mu,\nu), \quad s<1 \qquad (1)$$

が成立するとき，ω を縮小変換といい，s をその縮小率という．初期画像 μ_0 に縮小変換 ω を n 回反復的に施して得られる画像を $\omega^n(\mu_0)$ とする．また，μ_{orig} を符号化対象の原画像とする．このとき次式が成立する．

$$d(\mu_{orig},\omega^n(\mu_0)) \leq \frac{1}{1-s} \cdot d(\mu_{orig},\omega(\mu_{orig})) + s^n \cdot d(\mu_{orig},\mu_0) \qquad (2)$$

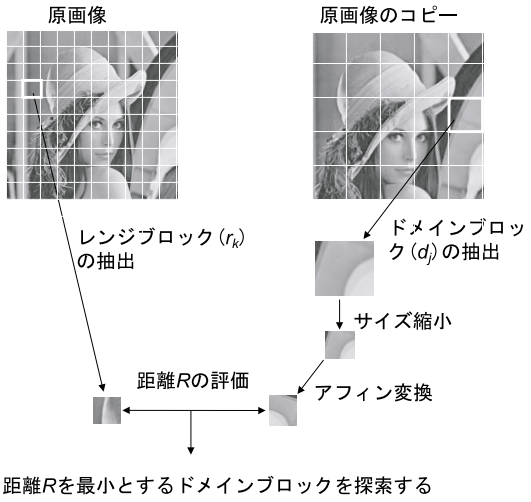

図 1 フラクタル画像符号化のアルゴリズム
Fig. 1 Fractal image encoding algorithm.

式 (2) より，μ_{orig} が $\omega(\mu_{orig})$ に近い場合，右辺の第 2 項が十分小さくなるよう n を大きくとれば右辺は零に近づく．したがって，任意画像 μ_0 に対して ω を反復した $\omega^n(\mu_0)$ を求めれば，原画像 μ_{orig} に近似できる．

2.2 自然画像の符号化方法

画像のフラクタル符号化は，収束画像 $\omega^n(\mu_0)$ が原画像 μ_{orig} に近似するような適切な縮小変換 ω を求め，得られた ω を符号として用いるものである[4)]．

実際の自然画像では，全体の画像を分割し，分割した画像ブロック b_k ごとに縮小変換 ω_k を求める．

具体的な縮小変換 ω_k の作用は，原画像から一部の画像領域を抽出し，その抽出した領域を縮小して形成されたブロックの画素に対してアフィン変換 $s \cdot p_i + o$ を行うことである．ここで p_i は画素の強度を表す．したがって，抽出した領域の位置を示す情報とアフィン係数 s，o で ω_k が表現でき，これらが符号データとして用いられる．

以下，符号化の手順を具体的に説明する．**図 1** に示すように最初に原画像全体を互いに重なり合わないブロックに分割する．このブロックはレンジブロックと呼ばれ，式 (2) における μ_{orig} に相当する．次に同じ原画像から各辺の大きさがレンジブロックの整数倍のブロックを作成する．このブロックはドメインブロックと呼ばれ，ドメインブロックどうしはオーバラップしてもかまわない．このドメインブロックを集めたドメインプールを構成する．また，ドメインブロックを回転，反転して作成したブロックについてもドメインプールに付け加える．ドメインブロックが全体画像か

ら抽出された領域に相当する．

レンジブロック r_k の適切な縮小変換 ω_k は $d(r_k, \omega_k(\mu_{orig}))$ が最も小さくなるものであり，ドメインプールの中からレンジブロックとの相似度が最も高いドメインブロックを探索することにより求められる．相似度は，ドメインブロック d_j とレンジブロック r_k 間の距離 $R(d_j, r_k)$ を用いる．距離 $R(d_j, r_k)$ はドメインブロックをレンジブロックと同じサイズになるように縮小（間引き，または平均化），アフィン変換を施した後に，レンジブロックとの画素強度差の自乗の和（SSD: sum of square difference）により式 (3) で表される[4]．

$$R(d_j, r_k) = \sum_{i=1}^{n} (s \cdot a_i + o - b_i)^2 \tag{3}$$

ここに，n はレンジブロックを構成している画素数，a_i はサイズ縮小後のドメインブロックの画素強度，b_i はレンジブロックの画素強度，s，o はアフィン変換係数である．R を最小にする s，o は，R の偏微分から式 (4)，(5) で求まる．

$$s = \frac{n\sum_{i=1}^{n} a_i b_i - \sum_{i=1}^{n} a_i \sum_{i=1}^{n} b_i}{n\sum_{i=1}^{n} a_i^2 - (\sum_{i=1}^{n} a_i)^2} \tag{4}$$

$$o = \frac{\sum_{i=1}^{n} b_i - s\sum_{i=1}^{n} a_i}{n} \tag{5}$$

式 (4)，(5) を用いると距離 R は式 (6) で表される．

$$R(d_j, r_k) = \frac{n\sum_{i=1}^{n} b_i^2 - (\sum_{i=1}^{n} b_i)^2}{n} - \frac{n\sum_{i=1}^{n} a_i^2 - (\sum_{i=1}^{n} a_i)^2}{n} \cdot s^2 \tag{6}$$

ドメインプールの中で最小の距離 R を持つドメインブロックが最適ドメインブロックになる．すべてのレンジブロックについて求めた最適ドメインブロックの位置情報とそのアフィン係数 (s, o) で ω が表現され，符号化ができる．

2.3 quad-tree 手法

少ない符号量でフラクタル符号化するにはレンジブロックの個数が少ないこと，すなわちレンジブロックのサイズが大きいことが必要である．しかし，同一の画像内でサイズが大きいレンジブロックに類似したドメインブロックを見つけることは，絵柄が複雑な画像では困難である．そこで，少ない符号量で復号画像の再現性を高める手法として，quad-tree 手法[3] が用いられる．この手法は最初に，原画像を大きなサイズのレンジブロック（最大レンジブロック）に分割し，このレンジブロックから最適ドメインの探索を始める．もし，相似度の高いドメインブロックが得られない場合には，レンジブロックをさらに 4 分の 1 の大きさに再分割し，再度最適ドメインの探索を行う．この繰返しをレンジブロックがある決められた大きさ（最小レンジブロック）になるまで行う．最終的に，原画像は複数のサイズのレンジブロックから構成される．このレンジブロックの構成情報を quad-tree 情報，あるいは quad-tree 構成と呼ぶ．

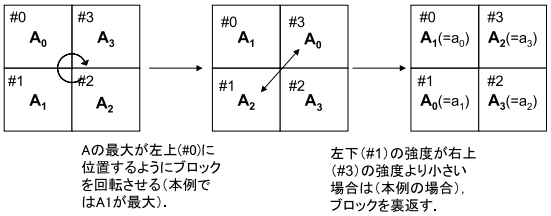

図 2 ブロックのクラス化
Fig. 2 Classifications of blocks.

2.4 クラス化手法

フラクタル画像符号化でのレンジブロックとドメインブロックとの比較回数は，全探索の場合にはレンジブロック数 N_r とドメインブロック数 N_d の積 $N_r \cdot N_d$ である．したがって，フラクタル符号化時間の短縮には，この比較回数を削減することが有効である．少ない比較回数で良好な探索結果を得るため，レンジブロックおよびドメインブロックを画像の特徴で分類し，同一のクラスどうしでのみ探索するクラス化手法[14),15)] が用いられる．以下に，この手法を述べる．

まず，**図 2** に示すようにドメインブロックおよびレンジブロックを 4 個のサブブロック（#0，#1，#2，#3）に分割し，これらについて強度 A_i，偏差 V_i を式 (7)，(8) を用いて計算する．ここで，r_j^i は i 番目のサブブロックに属する j 番目の画像強度である．

$$A_i = \sum_{j=0}^{n-1} r_j^i \tag{7}$$

$$V_i = \frac{1}{n}\sum_{j=0}^{n-1} [(r_j^i)^2 - A_i^2] \tag{8}$$

次に，左上（#0）に，強度 A_i が最大のサブブロックが位置するようにブロックを回転させる．左下（#1）のサブブロックの強度が右上（#3）の強度より小さい場合にはブロックを裏返す．左上の位置にあるサブブロックの強度を a_0，左下を a_1，右下を a_2，右上を a_3 とした場合，4 個の a_i の相対関係は以下の 3 通りになり，1st クラス（$pfirst$）が決定される．

$pfirst\ 1 : a_0 \geq a_1 \geq a_2 \geq a_3$
$pfirst\ 2 : a_0 \geq a_1 \geq a_3 \geq a_2$
$pfirst\ 3 : a_0 \geq a_2 \geq a_1 \geq a_3$

また，V_i の相対関係は 24 個（4!=24）であり，それを 2nd クラス（$psecond$）で表す．$pfirst$，$psecond$ のクラス数はそれぞれ 3 個，24 個であり，全体で 72 個のクラスに分類される．また，$psym$ は回転した大きさ（0°，90°，180°，270°）とブロックを裏返したかどうかで 8 種類に分類される．

クラス化手法ではレンジブロックと同一の $pfirst$，$psecond$ を持つドメインブロックに探索を限定することで，フラクタル符号化時間の短縮が実現できる．この場合の，レンジブロックとドメインブロックの探索回数 F は次式で表される．ここで，$N_r(c)$，$N_d(c)$ は，それぞれクラス c のレンジブロック数，ドメインブロック数である．

$$F = \sum_{c=1}^{72} N_r(c) \cdot N_d(c) \tag{9}$$

3. フラクタル画像圧縮プロセッサのアーキテクチャ

3.1 LSI アーキテクチャ

提案するフラクタル画像符号化プロセッサの VLSI アーキテクチャを**図 3** に示す．

静止画のフラクタル符号化プロセッサ[9] に，動画像の処理を行う動画像制御部と基準フレーム情報格納部を追加した構成で実装している．符号化される画像データは入力インタフェース（Input IF）を経由して外付けの画像バッファメモリ（External image buffer memory）に格納される．また，符号化の前処理として，原画像を $1/n$（n：整数）に縮小した画像を作成し，画像バッファメモリ内に格納する．この縮小画像は，レンジブロックとの比較に用いられる縮小されたドメインブロックの画素データとして使用される．クラス化部（Classifier）ではドメインブロック，レンジブロックのクラス（$pfirst$，$psecond$），$psym$，および R の計算に用いられる係数（$\sum a_i$，$\sum b_i$，$\sum a_i^2$，$\sum b_i^2$）を計算し，その結果をクラスメモリに格納する．レンジ–ドメインブロック探索部（Range-Domain block matcher）ではドメインブロックとレンジブロックとの距離 R，アフィン変換係数 s，o を計算する．データ制御部（Data controller）では，マッチングするレンジブロックとドメインブロックを決定し，レンジ–ドメインマッチング部へ必要なデータを出力する．これらの動作を輝度（Y）成分，色差（U，V）成分について繰り返す．符号データパック部（Encode data packing）では各レンジブロックについてのドメインブロックの位置，アフィン変換係数を連結して符号化データ出力を作成し，出力インタフェース部（Output IF）から外部に出力する．

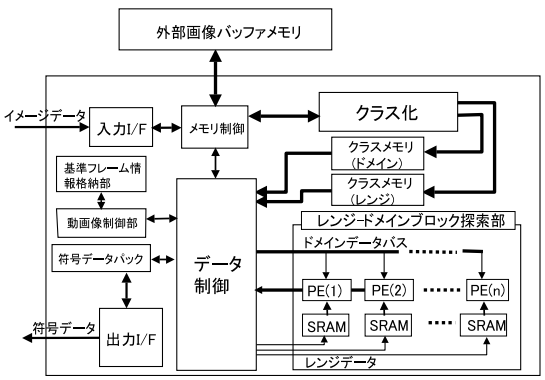

図 3 提案するフラクタル画像圧縮プロセッサの構成図
Fig. 3 Block diagram of the proposed fractal image compression processor.

3.2 レンジ–ドメインブロック探索部

レンジ–ドメインブロック探索部（Range-Domain block matcher）は，図 3 に示されるように，n 個の複数の PE（Processor Element）とこれと対になる SRAM から構成されており，n 個の同一クラスのレンジブロックの最適ドメインブロックを同時に探索することができる．以下，動作を説明する．

まず最初に，データ制御部から n 個のレンジブロックの画素データが，PE に接続されている SRAM に転送された後，1 つのドメインブロックの画素データがドメインデータバスを用いて転送されるたびに，n 個の PE で並列に range-domain マッチングが行われる．ここで PE に転送されるドメインブロックとレンジブロックのクラスは同一である．

PE 部では SRAM に画素データが格納されているレンジブロックと，ドメインブロック間の距離 R，アフィン係数を式 (4)〜(6) を用いて計算する．式 (4)〜(6) の計算に必要な $\sum a_i$，$\sum b_i$，$\sum a_i^2$，$\sum b_i^2$ のパラメータはレンジブロックあるいはドメインブロックに固有の値であり，距離 R の計算の前にクラス化部であらかじめ求められている．$\sum a_i b_i$ はレンジブロックとドメインブロックの関係で決まるので本 PE 部で計算する．

R の計算で最も処理量が多く，演算に時間がかかるのが $\sum a_i b_i$ の計算であり，これをいかに高速に行うかがプロセッサの高性能化のポイントになる．

PE 部の回路構成は**図 4** に示すように，主に $\sum a_i b_i$ の計算を行うデータパス部と，この結果を用いて R，s，o を計算する計算部に分かれる．データパス部と計算部はパイプラインで並列に動作する．データパス

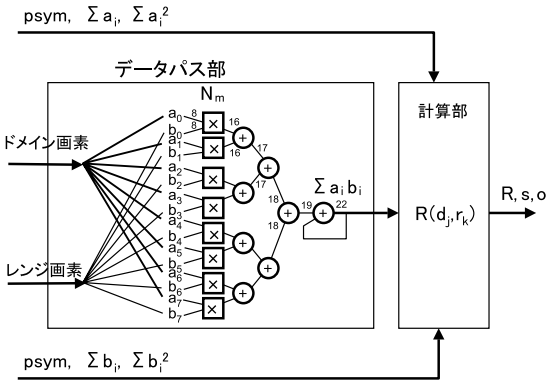

図 4 PE（Processor Element）の構成
Fig. 4 PE (Processor Element) architecture.

部は N_m 個の乗算器を持っており，$\sum a_i b_i$ の演算を高速に行う．たとえば，乗算器の個数が 8 個，レンジブロックのサイズが 8 × 8 のとき，8 サイクルで演算を完了する．乗算器の演算ビット幅は入力が 8 ビット，出力が 16 ビットである．加算器の演算ビット幅は図 4 の中に示してある．

3.3 アーキテクチャの最適化

リアルタイムの動画像圧縮では，制約時間内に符号化の処理を終了しなければならない．フラクタル符号化処理時間のほとんどを占める最適ドメインの探索時間は，画像サイズ，符号化条件，プロセッサの構成に依存する．符号化条件には，レンジブロックサイズ，ドメインブロックサイズ，ドメインブロックのオーバラップ量 D_{ol} がある．ここでドメインオーバラップ D_{ol} は，ドメインブロックの重なり具合を表す値で，ドメインブロックの一辺の長さ l_d と隣り合うドメインブロック間の距離 l_t を用いて式 (10) で表される．ドメインオーバラップが大きいほどドメインブロック数は多くなり，復号画像の品質は高まるが探索時間は長くなる．

$$D_{ol} = \frac{l_d}{l_t} \tag{10}$$

プロセッサの構成には PE の並列度と乗算器の個数がある．

ここでは，**表 1** に示すパラメータの条件下で，CIF 画像を 30 fps または 15 fps でリアルタイムで画像圧縮できることを制約条件として与え，この制約を満たす PE の個数と乗算器の個数の関係を求めた．必要な PE の並列度と乗算器の個数の関係が**図 5** に示される．この計算では，ドメイン画素データを PE 部に転送する時間を考慮してあり，表 1 に示してあるドメインデータ転送バス幅と動作周波数の積から決まるデータ転送速度 400 メガバイト/秒と，転送するドメインブロックのデータ量から求めている．ブロックサイズが 8 × 8 のドメインブロックのデータ量は，1 画素を 1 バイトとすると 64 バイトであり，1 ドメインブロックあたりのデータ転送時間は 160 nsec になる．同様に，レンジ画素データを PE に接続されている SRAM に格納する時間は，レンジデータ転送バス幅と動作周波数の積から決まるデータ転送速度 200 メガバイト/秒と，レンジブロックのデータ量 16 バイトから，1 レンジブロックあたり 80 nsec になる．

表 1 乗算器の並列数と PE の個数の関係を求めるために用いたパラメータ
Table 1 Parameters used to determine the relation between the parallelism of multipliers and the number of PEs.

画像サイズ	352 × 288 (CIF)
ドメインオーバラップ	4
レンジブロックサイズ	4 × 4
ドメインブロックサイズ	8 × 8
ドメインデータ転送バス幅	128 bit
レンジデータ転送バス幅	64 bit
動作周波数	25 MHz
乗算処理時間	1 サイクル

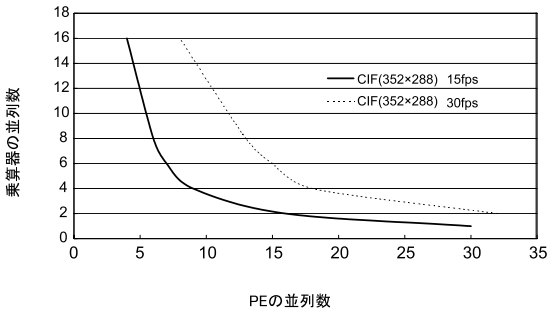

図 5 処理時間の制約を与えた場合の乗算器の並列数と PE の個数の関係
Fig. 5 Relation between the parallelism of the multipliers and the number of PEs when the processing time is restricted.

4. 本フラクタル画像圧縮プロセッサで用いる動画像のフラクタル画像圧縮方式

本フラクタル画像圧縮プロセッサの特徴は，動画像のフラクタル符号化において，基準となる別の画像フレームのドメイン情報およびアフィン変換係数を用いることである．基準となるフレームのフラクタル符号化の情報は，図 3 に示される基準フレーム情報格納部に，次の基準フレームが設定されるまで保持される．

本フラクタル画像圧縮プロセッサで採用している動画像のフラクタル符号化の手順を**図 6** に示す．この処

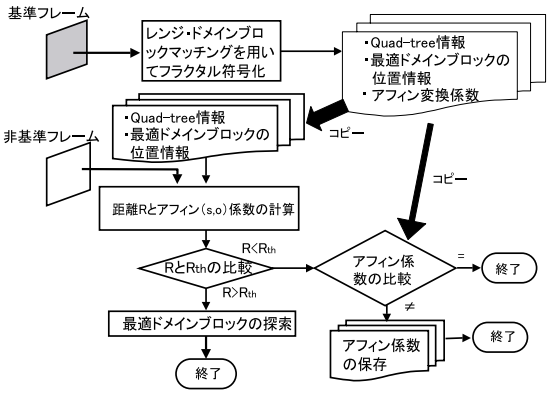

図 6 本フラクタル画像圧縮プロセッサで用いている動画像のフラクタル符号化手順
Fig. 6 Sequence of fractal movie encoding used in the proposed VLSI.

理の制御は，図 3 にある動画像制御部で行われる．最初に，動画を構成するフレームを基準フレームと非基準フレームに分類する．基準フレームのフラクタル符号化は，2 章で示した静止画をフラクタル符号化する従来どおりの手法を適用し，フレーム内での range-domain ブロックマッチングにより行う．ここで得られた探索結果の quad-tree 構成，最適ドメインブロック位置情報，アフィン変換係数は別フレームの符号化で用いるため保存しておく．

次に，非基準フレームでの quad-tree 構成と最適ドメインブロック位置の決定は，range-domain ブロックマッチングを用いては行わず，保存されている基準フレームの値を代用する．非基準フレームでのアフィン変換係数と距離 R は，基準フレームのドメインブロック位置情報と非基準フレームの画素データから求める．

アフィン変換係数と距離 R は，ドメインブロック位置が基準フレームと同一であっても，画素データが基準フレームと違えば基準フレームの値とは異なる．

フレーム内でドメインブロックの探索を行わず，別フレームのドメイン情報を用いることは，画質の劣化を生じさせる恐れがある．そこで符号化品質の劣化を抑制するために，上記方法で求めた距離 R があるしきい値 R_{th} より小さい場合のみ（$R < R_{th}$），ここで得られたアフィン係数を用いるが，$R > R_{th}$ の場合は基準フレームのドメイン情報を用いずに，基準フレームで行った方法と同様に，フレーム内での range-domain ブロックマッチングを用いて，フラクタル符号化を行う．ここで，距離 R のしきい値 R_{th} との比較はレンジブロックごとに行う．

非基準フレームのレンジブロックには，3 種類のモードが存在する．ドメインブロック情報とアフィン情報を必要としないブロック，アフィン情報のみを必要とするブロック，ドメインブロック情報とアフィン情報を必要とするブロックである．

4.1 本手法による符号量の削減

フラクタル符号化の符号量は式 (11) で表される．

$$L = \sum N(r) \cdot (l_s + l_o + l_d(r) + l_t) \qquad (11)$$

ここで $N(r)$ はサイズが $r \times r$ のレンジブロックの個数，l_s，l_o はアフィン変換係数 $(s,\ o)$ のビット数，l_d はドメインブロックの位置情報を示すのに必要なビット数である．レンジブロックがどのモードに属しているかの情報ビットが l_t である．

非基準フレームでの各レンジブロックについて，最適ドメインブロック位置情報として基準フレームで得られている値を用いれば，このレンジブロックの最適ドメインブロックの位置情報が削減できる．ドメイン位置情報ビット：l_d はドメインブロックプール内のドメインブロックの個数で決まるが，VGA サイズの場合は 10〜12 bit で表されることが多く，これらビットの削減によるフラクタル符号量の削減効果は大きい．また，アフィン変換係数の値が基準フレームの値と同一であれば，アフィン情報ビット：l_s，l_o が削減できる．アフィン変換係数 s，o には，画質的な観点から経験的にそれぞれ 5 bit，6 bit が割り当てられている．

4.2 本手法の評価実験

本節では，本フラクタル画像圧縮プロセッサに実装するアルゴリズムである基準フレームの探索結果を用いて動画像を符号化した結果と，本アルゴリズムを用いない通常のフレーム内のみの情報を用いて符号化した従来方式による結果を比較評価する．

評価項目は圧縮率，画像圧縮時間，および画像の符号化品質である．ここで，圧縮率（Compression ratio）は原画像の符号量を C_{orig}，フラクタル符号化後の符号量を C_{enc} とすると以下の式で定義される．符号化効率が高いほど，圧縮率の値は高くなる．

$$Compression\ ratio = \frac{C_{orig.}}{C_{enc}} \qquad (12)$$

画像圧縮時間の評価は，フラクタル符号化を完了するまでに，range-domain ブロックマッチング部で行われたフレームごとの，レンジブロックとドメインブロックの探索回数で行う．画像の符号化品質は，得られた符号データ（最適ドメインブロック情報，アフィン係数）を用いて，反復アフィン変換を 20 回繰り返して得られた再生画像と原画像との PSNR（Peak Signal to Noise Ratio）で評価する．

実験に用いる動画像データを**表 2** に示す．

表 2 実験に用いる動画像データ
Table 2 Movie data used for the experiments.

画像名	画素数	フレーム数	フレームレート（fps）	ピクセルあたりのビット数	特徴
Akiyo	176×144	15	10	24	アナウンサが原稿を読んでいる場面
*Aki*1	352×288	300	15	24	アナウンサがゆっくり歩いてる場面
friends	352×288	300	15	8	子供が遊んでいる場面
drive	352×288	300	15	8	走行車から見た風景
maiko	352×288	300	15	8	右から左に移動している風景
Lenna	512×480	-	1	24	静止画

cif: 352 × 288 画素

動画像 *Akiyo*（176 × 144 画素，10 フレーム，24 bit/ピクセル）の輝度（Y）成分を用いて得られた符号化結果を**図 7** に示す．レンジブロックのサイズは 4 × 4 と 8 × 8，ドメインブロックのサイズは 8 × 8 と 16 × 16 である．ドメインオーバラップは 2 であり，ドメインブロックどうしは縦，横それぞれ隣のドメインブロックと半分の重なりがある．s のビット数は 5 bit，o のビット数は 6 bit である．基準フレームは動画像の一番先頭のフレーム#0 とした．フレーム#1 以降はフレーム#0 の探索結果を用いて符号化する非基準フレームである．

図 7 (a) より，range-domain ブロックのマッチング回数は約 5 分の 1 に減少し，符号化に必要な演算量が減ることが知られる．図 7 (b) より，基準フレーム#0 のドメイン情報を用いて符号化した非基準フレーム#1〜#8 の符号量は減少し，圧縮率が約 2 倍高くなっていることが知られる．図 7 (c) は，従来方式の場合と比較して，どれだけ劣化するかを示している．劣化の度合いはフレーム番号が進むほど大きくなっているが，しきい値 R_{th} を小さく設定することで画像の劣化が抑制できることが知られる．$R_{th} = 5$ の場合には画像の劣化がほとんど生じていないことが知られる．

図 8 に，動画像 *Aki*1（352 × 288 ピクセル，15 fps，300 フレーム）のレンジブロックのサイズを変化させて得られた，全フレームで平均した PSNR と符号量の関係を示す．一般に，符号化品質（PSNR）を向上させると符号量は大きくなるが，同図より，本方式では従来方式より少ない符号量で所望の符号化品質（PSNR）を得られることが知られる．これから，非基準フレームの符号化を基準フレームのドメイン情報を用いて符号化したことで符号量が減少できたことが示される．

また，動画像 *Aki*1 を用いた実験から得られた探索回数と符号化品質の関係を**図 9** に示す．フラクタル符号化で類似性の高いブロックの探索回数を多くすることで符号化品質を高めることができるが，本手法は従来方式より少ない探索回数で，要求される符号品質を満足できることが本実験結果より示される．

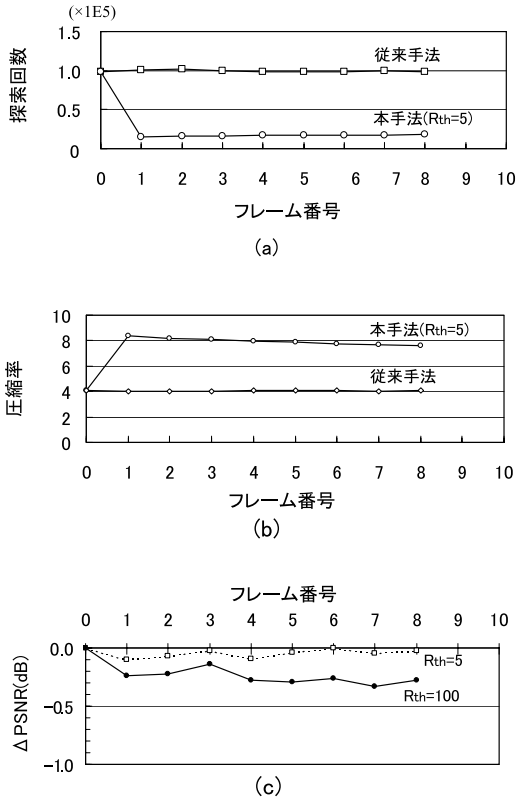

図 7 画像 *Akiyo* を用いた符号化実験の結果，(a) 探索回数，(b) 圧縮率，(c) 画像品質（PSNR）の劣化
Fig. 7 Results of the encoding experiments of image Akiyo, (a) Matching number, (b) Compression ratio, (c) Degradation of the image quality (PSNR).

以上の結果から，本フラクタル画像圧縮プロセッサに実装した動画像圧縮のアルゴリズムは，少ない演算量で動画像の圧縮ができ，符号化時間の短縮に効果があること，また符号品質の向上や符号量の削減にも効果があることが知られる．

4.3 カラー画像への適用

カラー画像の符号化は，各フレームごとに YUV 成分を独立にフラクタル符号化することで行う．画像

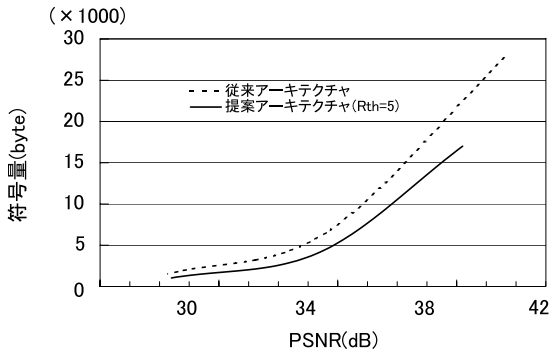

図 8 動画像 *Aki*1 のフラクタル符号化での PSNR と符号量の関係
Fig. 8 Relation between the PSNR and the coding size when movie *Aki*1 is encoded.

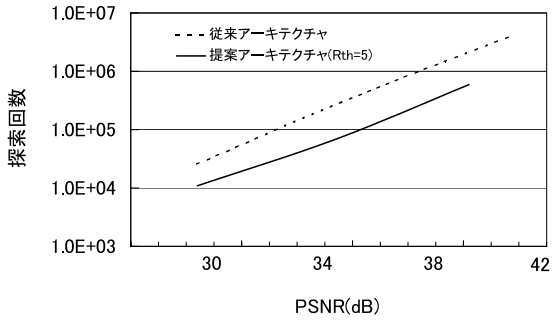

図 9 動画像 *Aki*1 のフラクタル符号化での PSNR と探索回数の関係
Fig. 9 Relation between the PSNR and the number of matching when movie *Aki*1 is encoded.

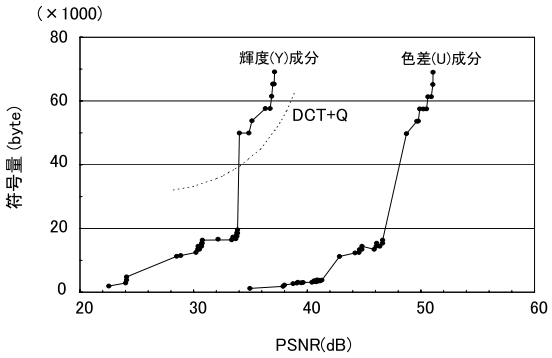

図 10 輝度，色差成分をフラクタル符号化した結果
Fig. 10 Results of fractal encoding of luminance and chrominance data.

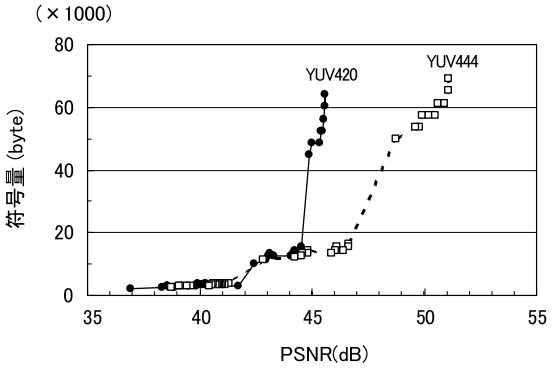

図 11 YUV440 と YUV42 とフラクタル符号化した結果
Fig. 11 Results of fractal encoding of YUV440 images and YUV420 images.

Lenna（512 × 480 ピクセル，24 bit/ピクセル）の輝度（Y）成分，色差（U）成分を符号化したときの符号化画質（PSNR）と符号量の関係を**図 10** に示す．色差（V）成分についての結果は図示してないが色差（U）成分と同様な結果である．ここで符号化画質や符号量は，レンジブロックのサイズやアフィン変換係数 s，o のビット長を変化させて調整した．

フラクタル符号化方式で符号化した結果と DCT 方式で符号化した結果の比較を行う．ここではジグザグスキャンとハフマン符号化は行っていない．離散コサイン変換（DCT）と量子化（Q）により Y 成分を圧縮した場合の符号化品質 PSNR と符号量の関係を図 10 のフラクタル符号化の結果に重ねて示す．本図より，輝度（Y）成分の比較において，符号量が 40 キロバイト以下では，フラクタル符号化の方が，DCT + Q による符号化より画質が良いことが知られる．この結果より，フラクタル符号化は，符号量が小さくても実用的な画質が得られることが知られる．

本研究では，フラクタル画像圧縮に可変長符号化を組み合わせていないが，フラクタル符号化と可変長符号を複合することで，さらに符号量が削減できる可能性がある．

また，色差成分の画素を水平方向，垂直方向に 1/2 に間引いた YUV420 フォーマットの場合の画像品質と符号量の関係を**図 11** に示す．画像品質が 44.5 dB 以下では YUV420 フォーマットと，色差成分の画素を水平方向，垂直方向に間引かない YUV444 フォーマットでの符号量の差は見られない．これはフラクタル符号での符号量を決めているレンジブロックの個数が YUV444 と YUV420 の場合で同程度になっているためである．

5. VLSI 設計とその評価

5.1 VLSI 設計

図 3 として提案したアーキテクチャの有効性を VLSI 設計を行い検証する．外部フレームメモリはファーストページの DRAM を利用する．マッチング部のレンジブロックメモリは 64 bit × 16 word の SRAM，ク

表 3 LSI の諸元
Table 3 Specification of the LSI.

プロセス	0.25 μm，5 層メタル CMOS
回路規模	115 K ゲート
内蔵 SRAM	400 Kbit
最大動作周波数	43 MHz

表 4 設計したフラクタル画像プロセッサのハードウェアコスト
Table 4 Hardware cost of designed fractal image processor.

回路ブロック	回路規模（gates）
入力 IF	4,300
メモリ制御	4,200
クラス化	13,500
データ制御	8,200
レンジド–メインブロック探索部	67,000
符号データパック部	3,000
出力 IF	5,200
基準フレーム情報格納部	6,400
動画像制御部	6,000
ロジック合計	114,800
クラスメモリ（レンジ）	256 Kbit
クラスメモリ（ドメイン）	128 Kbit
レンジデータメモリ	1 Kbit × 16

（PE の並列数：6，PE 内の乗算器数：8）

ラスメモリは 64 bit × 4 Kword の SRAM を用いた．

CIF サイズの静止画を 15 フレーム/秒でリアルタイムに画像圧縮できるように，レンジ–ドメイン探索部での PE 数を 6，PE 部内の乗算器数を 8 とした．ドメインデータ転送バス幅は 128 bit，レンジデータ転送バス幅は 64 bit とした．

回路設計は米国 *Synopsys* 社の論理合成ツール *DesignCompiler* を使用した．使用ライブラリは 0.25 μm，5 層メタル CMOS である．最大動作周波数は 43 MHz である．**表 3** に LSI の諸元，**表 4** に回路規模の内訳を示す．

5.2 動画像による圧縮実験

本節では，設計した動画像フラクタル画像圧縮プロセッサを評価し，従来方式の圧縮方式を用いている静止画フラクタル符号処理プロセッサ[9]（以後，従来構成 VLSI と記す）との比較を行う．評価に用いたカラー動画像の種類と結果を**表 5** に示す．動画像の構成は，352 × 288 画素（24 ビット/ピクセル），YUV444，フレーム数：300，フレームレート：15 fps，20 秒間の動画である．動画像フラクタル画像圧縮プロセッサ（以後，本 VLSI と記す）において，最適化ドメインの探索にはクラス化手法を適用した．基準フレームは 10 フレームごとに設け，これ以外のフレームは非基準フレームである．表 5 で示される圧縮率，探索回数，および符号時間は 300 フレームの平均値，PSNR は輝度（Y）成分について 300 フレームの平均値である．符号化時間は，動作周波数をともに 25 MHz，ドメインデータ転送バス幅を 128 bit，レンジデータ転送バス幅を 64 bit としてロジックシミュレーションにより算出した．

表 5 の#1 から#6 は，*Aki*1 の結果を示している．#6 では，レンジブロックサイズは 16 × 16，8 × 8，4 × 4 の 3 種類，ドメインブロックのオーバラップ D_{ol} は 4 を用いた．このときの従来構成 VLSI でのブロックマッチングの平均回数は 528,000 回，1 フレームあたりの符号量は 6,376 バイトである．本 VLSI では，1 フレームあたりのブロックマッチング回数は 287,000 回，1 フレームあたりの符号量は 4,812 バイトであり，従来構成 VLSI よりマッチング回数は約 46%減少，符号量は約 32%減少していることが知られる．

また，#6 での全フレームの符号量，符号画質（PSNR），および range-domain ブロック探索回数を**図 12** に示す．符号量および探索回数が減少しているフレームが非基準フレームである．同図より，画質（PSNR）は基準フレームと非基準フレームで大きな変化がなく，非基準フレームでの画質の劣化が小さいことが知られる．

#7 から#12 は動きの激しい動画像の画像圧縮結果であり，これらの画像でも符号化時間の短縮や圧縮率の向上が実現でき，画像圧縮性能が改善されていることが知られる．

これらの結果から，本 VLSI は従来構成 VLSI と符号画質（PSNR）を同等にしたうえで，圧縮率は 1.5〜2 倍程度高く，実行した探索回数は約 1/2〜1/10 に減少，画像圧縮時間が従来構成 VLSI より短縮，符号量も減少することが知られる．

5.3 VLSI アーキテクチャの考察

前節では，従来構成 VLSI との比較のために動作周波数を 25 MHz として処理時間を算出したが，設計した本 VLSI は 43 MHz まで動作する．したがって，この動作周波数で動作させれば処理時間はさらに短くできる．

30 fps でリアルタイムに符号化するには，平均符号化時間は 33 msec 以下でなければならないが，表 5 の本 VLSI の結果から，符号化時間はこの条件を満足しており，リアルタイムに符号化できることが示される．また，画素数が CIF の約 8 倍多い XGA サイズ（1024 × 768 画素）を符号化する場合，一般にフラクタル符号化の演算量は画素数の二乗に比例し，同一のハードウェアであれば処理時間も画素数の二乗に比例するので，処理時間は CIF 画像の約 8 倍となるが，本 VLSI

表 5 動画像符号化結果の従来構成 VLSI との比較
Table 5 Comparison between the results of movie encoding obtained by the conventional method and the proposed method.

動画像	レンジブロック 最小	レンジブロック 最大	ドメイン D_{ol}	圧縮率 本 VLSI/従来 VLSI	PSNR（dB） 本 VLSI/従来 VLSI	探索回数（× 1E3） 本 VLSI/従来 VLSI	符号化時間（msec） 本 VLSI/従来 VLSI
#1 Aki1	4 × 4	4 × 4	4	12.2/7.5	39.6/40.6	689/4754	12.5/90.2
#2 Aki1	4 × 4	4 × 4	2	12.4/8.0	38.6/39.5	184/1204	3.2/23.8
#3 Aki1	8 × 8	8 × 8	4	49.6/31.9	34.4/34.6	73/332	4.6/22.5
#4 Aki1	8 × 8	8 × 8	2	51.1/34.3	33.4/33.4	21/87	1.9/6.2
#5 Aki1	16 × 16	16 × 16	4	200/135	29.4/29.4	11.2/26.1	5.2/9.0
#6 Aki1	4 × 4	16 × 16	4	63.2/47.7	37.3/37.6	287/528	19.1/32.8
#7 friends	4 × 4	4 × 4	4	12.3/9.8	28.9/29.2	1661/2868	27.6/45.5
#8 friends	8 × 8	8 × 8	4	38.7/31.7	23.9/24.0	153/217	8.5/12.5
#9 friends	4 × 4	16 × 16	4	14.9/15.5	28.7/28.8	1695/1752	28.1/32.7
#10 friends	8 × 8	16 × 16	4	44.6/41.6	23.9/23.9	167/192	5.0/5.8
#11 drive	8 × 8	8 × 8	4	39.4/31.7	23.4/23.4	152/221	8.9/15.0
#12 maiko	8 × 8	8 × 8	4	38.0/31.7	21.1/21.1	176/228	9.2/15.3

R_{th}=100，352 × 288 画素，YUV444，24 ビット/ピクセル，フレーム数 300，フレームレート 15 fps，20 秒動画，符号化前の画像符号量：304,128 バイト

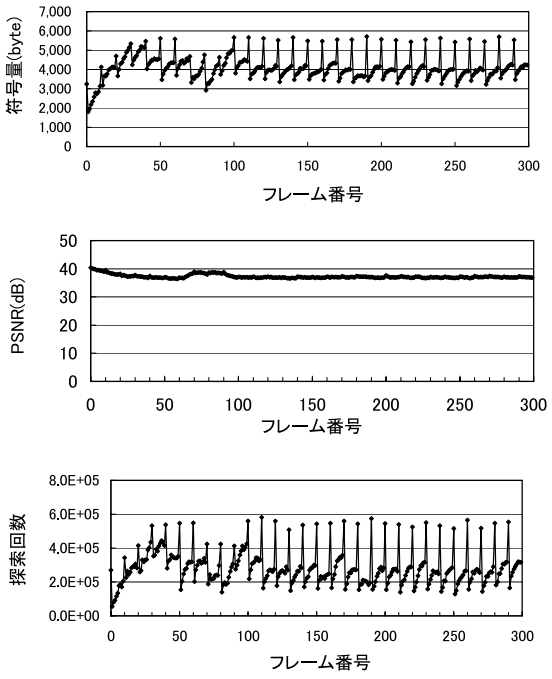

図 12 設計した VLSI により，動画像データ *Aki*1 をフラクタル画像圧縮した結果
Fig. 12 Results of fractal image compression of the movie data *Aki*1 by the designed VLSI.

アーキテクチャは PE の並列度や乗算器の並列度を高めることで容易に符号化時間を短縮することが可能であり，XGA サイズの動画像でもリアルタイム符号化が可能である．

さらに，本フラクタル画像圧縮プロセッサは少ない演算量で動画像の圧縮ができるので，実験で用いた動作周波数を下げてもリアルタイムの動画圧縮が可能である．たとえば，表 5 の#3 の実験では 1 フレームあたり 4.6 msec の時間で符号化できることが示されており，動作周波数を 1/5 の 5 MHz に下げても 23 msec の時間でリアルタイムに符号化できることが推論される．これより，本 VLSI アーキテクチャは LSI の低消費電力化にも効果があることが分かる．

表 5 の#2 では，PSNR は従来構成 VLSI よりが 0.9 dB 減少しているが，#1 のようにドメインオーバラップを大きくしドメインブロックを増加させることで，PSNR は従来構成 VLSI より高くなる．このときでも，符号化時間は 12.5 msec であり，従来構成 VLSI での 23.8 msec より約 2 倍早くできることが示されている．

これまでに，PE（Processor Element）アレイで構成したアーキテクチャを持つフラクタル符号化 LSI が提案されている[6]．この LSI では，レンジブロックごとに PE を持つ構成で，QCIF 画像（176 × 144 画素）をリアルタイムに符号化できるが，そのハードウェア量は 290 K ゲートである．我々の提案した構成ではその約 1/3 の小さなハードウェア量で約 4 倍の大きさの CIF 画像（352 × 288 画素）のリアルタイム符号化が可能である．

これらより，本提案の LSI アーキテクチャは動画像のフラクタル符号化を効率良くできることが知られる．

6. ま と め

カラー動画のフラクタル画像符号化をフレーム内のみの情報で行うのではなく，別フレームのフラクタル情報を参照して行う本フラクタル画像圧縮プロセッサは，動画像のフラクタル符号化における必要な演算量を約 1/2〜1/10 に減少させ，同時に符号量も 20〜70%に削減できる．また，小さなハードウェア量で実

現でき，これまでに報告されているフラクタル符号処理プロセッサと比べてカラー動画の符号化時間の短縮と高圧縮率化が可能である．今後は，フラクタル方式よる動画像の符号化において高画質化，可変長符号化による高効率化の改善を行う予定である．

参 考 文 献

1) Barnsley, M.F.: *Fractals Everywhere*, Academic Press Inc. (1988).
2) Jacquin, A.E.: Fractal image coding: A Review, *Proc. IEEE*, Vol.81, No.10, pp.1451–1465 (Oct. 1993).
3) Jacquin, A.E.: Image coding based on a fractal theory of iterated contractive image transformation, *IEEE Trans. Image Processing*, Vol.1, pp.18–30 (1992).
4) Fisher, Y.: *Fractal Image Compression Theory and Application*, Springer Verlag, New York (1995).
5) Lu, N.: Fractal Imaging, Academic Press Inc. (1997).
6) He, Z.L., Liou, M.L. and Fu, K.W.: VLSI architecture for real-time fractal video encoding, *IEEE ISCAS*, Vol.2, pp.738–741 (1996).
7) 季 信行，阿曽 具：フラクタル画像圧縮の VLSI アーキテクチャ，信学技報，CPSY98-82, pp.9–14 (1998).
8) 松浦昭洋，永野秀尚，名古屋彰：フラクタル画像圧縮再構成可能アーキテクチャによる実現法，信学技報，CPSY98-83，pp.15–22 (1998).
9) Yamauchi, H., Takeuchi, Y. and Imai, M.: VLSI Architecture for Real-Time Fractal Image Coding Processor, *IEICE TRANS. FUNDAMENTALS*, Vol.E83-A, No.3 (2000).
10) Chen, W.H., Smith, C.H. and Fralick, S.C.: A fast computational algorithm for the discrete cosine transformation, *IEEE Trans. Commun.*, COM-25 (9), pp.1004–1009 (1977).
11) Okada, S., Okada, S., Matsuda, Y., Yamada, T. and Kiyozaki, K.: System on a chip for Digital Still Camera, *Digest of technical paper of IEEE 1999 International Conference on Consumer Electronics*, pp.86–87 (1999).
12) Paul, B. and Hayes, M.: Fractal-based compression of motion video sequences, *ICIP94*, pp.755–759 (1994).
13) Fisher, Y., Rogovin, D. and Shen, T.P.: Fractal (self-VQ) encoding of video sequences, *Proc. SPIE, VCIP94* (1994).
14) Jacobs, E.W., Boss, R.D. and Fisher, Y.: Fractal-based image compression, Technical Report 1362, Naval Ocean Systems Center, San Diego, CA (Jun. 1990).
15) Fisher, Y., Jacobs, E.W. and Boss, R.D.: Fractal image compression using iterated transform, Technical Report 1408, Naval Ocean Systems Center, San Diego, CA (1991).

(平成 12 年 9 月 22 日受付)
(平成 13 年 2 月 1 日採録)

山内 英樹（正会員）

昭和 56 年豊橋技術科学大学電気・電子工学科卒業．昭和 58 年同大学院修士課程修了．昭和 58 年三洋電機（株）中央研究所で半導体レーザの研究開発に従事，平成 3 年同社マイクロエレクトロニクス研究所にて，VLSI 設計および VLSI CAD の研究に従事．IEEE，電子情報通信学会各会員．

武内 良典（正会員）

昭和 62 年東京工業大学工学部電気・電子工学科卒業．平成 4 年同大学院博士後期課程修了．博士（工学）．平成 8 年大阪大学大学院基礎工学研究科情報数理系専攻講師．平成 11 年同大学院基礎工学研究科情報数理系専攻助教授．ディジタル信号処理，VLSI 設計および VLSI CAD の研究に従事．IEEE，電子情報通信学会各会員．

今井 正治（正会員）

昭和 49 年名古屋大学工学部電気工学科卒業．昭和 54 年同大学院博士後期課程修了（工学博士）．同年豊橋技術科学大学奉職．平成 6 年同教授．平成 8 年大阪大学大学院基礎工学研究科情報数理系専攻教授．その間，昭和 59 年から昭和 60 年にかけて米国サウスカロライナ大学工学部電気計算機工学科客員助教授（文部省在外研究員）．これまで組合せ最適化アルゴリズム，ハードウェア/ソフトウェア協調設計等の研究に従事．平成 3 年より日本電子機械工業会および IEEE/DASC において EDA 標準化作業に従事．現在，情報処理学会設計自動化研究会主査．IEEE，ACM，電子情報通信学会，人工知能学会各会員．