社団法人 情報処理学会 研究報告
IPSJ SIG Technical Reports

2008－SLDM－136（8）
2008 / 10 / 7

[招待論文] 実時間画像処理 VLSI の研究

深山正幸

金沢大学理工学域電子情報学類 〒920-1192 石川県金沢市角間町
E-mail: miyama@t.kanazawa-u.ac.jp

あらまし 本稿では各種制約の中で最適な実時間画像処理 VLSI を実現する研究の一端として，実時間画像符号化 VLSI と画像認識用実時間動き推定 VLSI の実現手法について報告する．
キーワード 実時間，画像符号化，動き推定，VLSI

[Invited Paper] A Study on Real-Time Image Processing VLSI

Masayuki MIYAMA

School of Electrical and Computer Engineering, Kanazawa University
Kakumamachi, Kanazawa, Ishikawa, 920-1192 Japan
E-mail: miyama@t.kanazawa-u.ac.jp

Abstract This paper describes our study on VLSI image processing, which includes real-time image coding VLSI and real-time motion estimation VLSI for image recognition.
Keyword real-time, image coding, motion estimation, VLSI

1. はじめに

今や我々の生活は画像に溢れている．これは近年の画像処理技術とそれを支える半導体・集積回路・計算機・ネットワーク技術の発展によるものである．画像符号化，画像認識，画像改善といった高度な画像処理が身近になればなるほど，機器の小型化，低電力化，低コスト化が要求される．また画像処理にはしばしば実時間性が要求され，高解像度化や高精度化などの高性能化への要求は止まる所を知らない．これらの課題を解決するには特定の画像処理に向けた専用 VLSI の開発が有効な手段である．画像処理 VLSI の開発では演算量と精度だけではなく VLSI 実装上の制約を考慮したアルゴリズムと効率的なアーキテクチャの開発が重要である．ここで VLSI 実装上の制約とはチップ面積，動作周波数，消費電力，外部データ転送量などであり，これらはデバイスや半導体プロセスに依存する．そこで本稿では各種制約の中で最適な実時間画像処理 VLSI を実現する研究の一端として，実時間画像符号化 VLSI と画像認識用実時間動き推定 VLSI の実現手法について報告する．

2. 実時間画像符号化 VLSI

ここでは PC デスクトップ画像の低遅延伝送システムに用いられる SXGA 60 fps 対応 JPEG 2000 コーデック VLSI の高性能化手法[1]と，MPEG-4 動き検出 VLSI の消費電力を 1mW 以下に抑える低電力化手法[2]について述べる．

2.1. 高性能 JPEG 2000 コーデック VLSI [1]

JPEG 2000 は高圧縮率とブロックノイズのない高画質を特徴とする画像圧縮規格である[3]．図 1 に示すとおり JPEG 2000 の符号化は離散ウェーブレット変換 (DWT)，量子化，レート-歪み最適化付きエンベディッドブロック符号化 (EBCOT)の順で行われる．1 枚の画像はタイルに分割される．DWT はタイルを周波数ごとのサブバンドに分解する．各サブバンドのウェーブレット係数は量子化される．量子化後サブバンドはいくつかのコードブロックに分割される．EBCOT ではコードブロックからビットモデリングと算術符号化によりビットストリームが生成され，目標のビットレート内で最小の歪みになるよう切り詰められる．

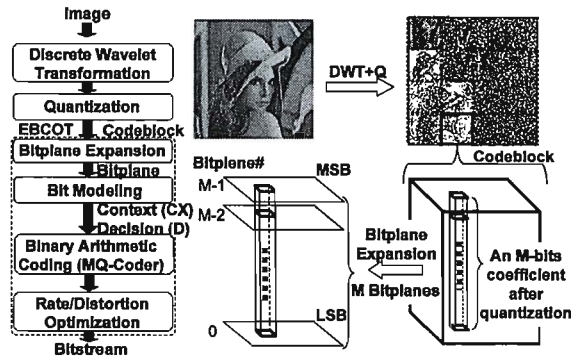


図 1. JPEG 2000 概要

図 2 に示すとおりエンベディッドブロック符号化(EBC)においてコードブロックはいくつかのビットプレーンに分解される．コードブロックは最上位のビットプレーンから最下位まで順に符号化される．1 枚のビットプレーンの符号化は 3 つのパスに分解され，順に処理される．それぞれのパスは決められた順でビット毎にビットプレーンを符号化する．あるビットの符号化は周囲の係数の現時点までの符号化結果に依存する．このため複数のビットを単純に並列処理できない．

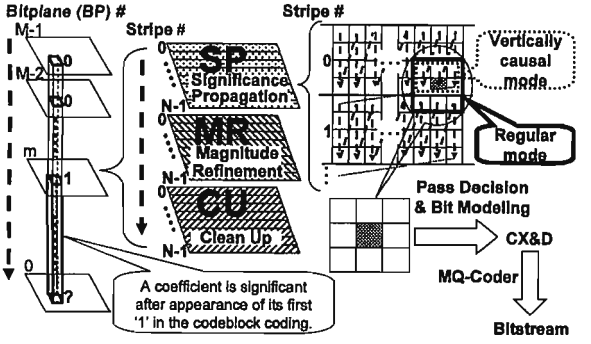


図 2．EBCOT 概要

EBC の高速化のためビットプレーン並列法を考案した．図 3 に示すとおり下位のビットプレーンの SP パスの符号化には上位のビットプレーンの CU パス後の符号化結果が必要である．そこで符号化結果を参照できるように上下のビットプレーンの符号化対象ビット位置をずらして並列処理する．さらにビットプレーン内の 3 つのパスも同様に前のパスの符号化結果が参照できるように符号化対象ビット位置をずらして並列処理する．本手法によりビットプレーン数を 10 枚としたとき 30 倍の高速化を達成できる．

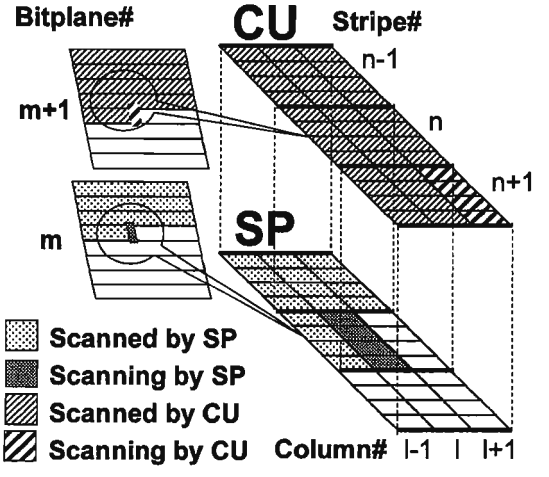


図 3．ビットプレーン並列手法

ビットプレーン並列手法を効率的に処理する EBC アーキテクチャを考案した．EBC のブロック図を図 4 に示す．各 BPC が対応するビットプレーンを符号化する．上位の BPC と下位の BPC は FIFO で接続され上位の符号化結果が下位に受け渡される．これにより上位と下位のビットプレーン符号化が自律的に同期する．

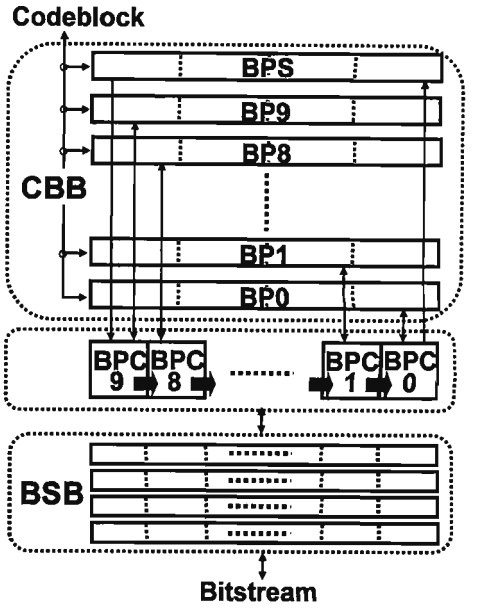


図 4．EBC ブロック図

EBC を含む JPEG 2000 コーデックを 0.18μm プロセスで VLSI 実装した．プロット図を図 5 に示す．チップサイズは 4.7mm 角であり，160MHz 動作時に 158MS の符号化性能を持つ．これは SXGA(1280×1024, 60fps, YCbCr4:2:2)を実時間符号化できる性能であり，PC デスクトップ画像の低遅延伝送に応用可能である．

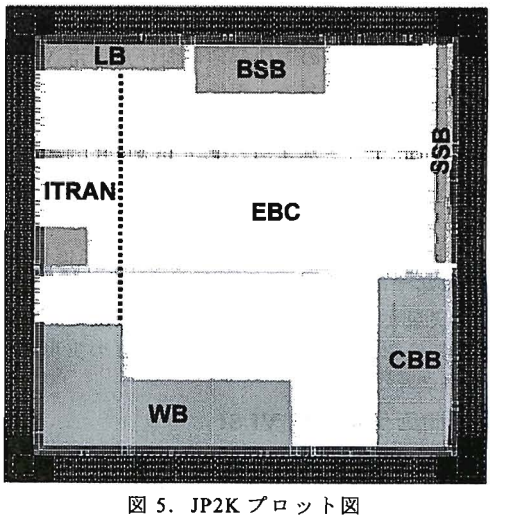


図 5．JP2K プロット図

2.2. 低電力 MPEG-4 動き検出 VLSI [2]

携帯端末で MPEG などの動画像を視聴する際，バッテリー駆動時間を長くするにはコーデックの低電力化が重要である．そこでコーデックの中でも電力消費割合の大きい動き検出部の低電力化を目的とし，消費電力 1mW を目標とする．動き検出アルゴリズムの演算量削減のため最急降下法を適用した．最急降下法は評価点における評価関数の下り勾配の最も急な方向に最小値の探索を進める方法であり，フルサーチと比較して探索点数を激減できる．このとき動きベクトル *(Vx,Vy)* の評価値 *E* は式(1)で表される．

$$E(V_x,V_y)=\sum_i\sum_j(TB_{i,j}-SW_{i+V_x,j+V_y})^2 \quad (1)$$

ここで *TB* は符号化対象フレームのテンプレートブロック，*SW* は参照フレームのサーチウィンドウである．*TB* と，*(Vx,Vy)* で示される *SW* 中の参照ブロックの対応画素同士の差分の二乗を取り，ブロック全体で和を取る．探索方向は以下の式(2)-(3)で表される評価関数 *E* の *x* 方向および *y* 方向の微係数を成分とするベクトル $-[\partial E/\partial x \;\; \partial E/\partial y]^T$（*T* は転置を表す）で表される．

$$\partial E/\partial x = -\sum_i\sum_j(TB_{i,j}-SW_{i+V_x,j+V_y})(SW_{i+1+V_x,j+V_y}-SW_{i-1+V_x,j+V_y}) \quad (2)$$

$$\partial E/\partial y = -\sum_i\sum_j(TB_{i,j}-SW_{i+V_x,j+V_y})(SW_{i+V_x,j+1+V_y}-SW_{i+V_x,j-1+V_y}) \quad (3)$$

最急降下法アルゴリズムのフローチャートを図 6 に示す．図中の 4 つのベクトルから初期ベクトルを選択し，その点における評価関数の勾配から探索方向を決定して一次元探索を行う．評価値最小の点で探索方向を変え一次元探索を行い，これを繰り返す．一次元探索の開始点が評価値最小であれば探索を終了する．

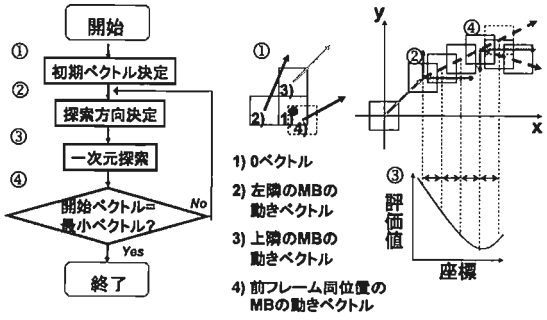


図 6．最急降下法アルゴリズム

演算量や記憶容量と画質の観点からパラメタを最適化した．シミュレーションには QCIF および CIF サイズのテスト画像を用いた．この結果，探索範囲±16，階層探索なし，一次元探索の繰返し回数 2 回，一括探索回数 4 回とした．ここで一括探索とは一次元探索において次の評価値が大きくなっても探索を終了せず決められた点数の探索を続行する方法である．またハードウェア簡単化のため探索方向を 8 方向に制限したところ予測画像の画質が向上した．最適化後のアルゴリズムと他のアルゴリズムの演算量と画質の比較を図 7 に示す．本手法(sdm_mb_h1)はスリーステップサーチ(tss)[4]と比較して演算量が低いにも関わらず高画質である．本手法は高速法 cote[4]より演算量の平均は高いが最悪は低く高画質である． 8×8 の 4 個のサブブロックの動きから 16×16 のマクロブロックの動きベクトルを求める方法を本手法に追加した方法(sdm_sb_h1)の画質劣化はフルサーチ(fs)と比較して無視できるほど小さい．

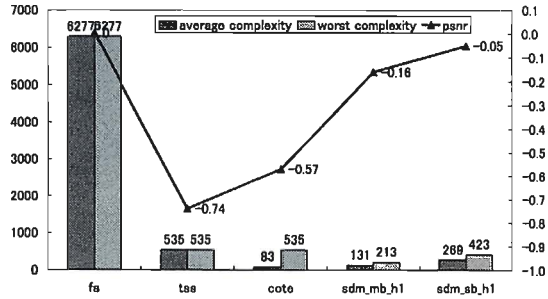


図 7．演算量と画質の比較

本アルゴリズムを効率的に実行するプロセッサ(ME4)のアーキテクチャを考案した．ME4 のブロック図を図 8 に示す．ME4 は 16 個の PE による SIMD 構成であり，評価ブロックの 1 行の計算を 1 サイクルで行う．PE は評価値計算，微係数計算，1/2 精度画素計算といった異なる計算を一つの回路で行える．任意のブロックの 1 行を 1 サイクルで供給できるようメモリが構成されている．1 次元探索で連続する評価値計算の間の無効サイクルを 0 とするよう制御回路が構成されている．また，1 次元探索の繰返しが途中で終った際にプロセッサをアイドル状態として電力を削減するためにゲーティッドクロックが用いられている．

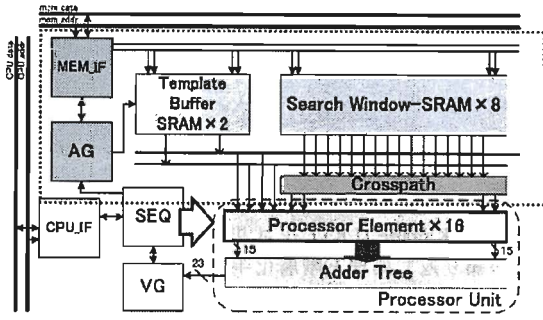


図 8．ME4 ブロック図

ME4 を 0.18μm プロセスで VLSI 実装した．チップ写真を図 9 に示す．コアサイズ 3.5mm×3.9mm，QCIF15fps 実行時のコアの消費電力は 0.4mW (0.85MHz, 1.0V)となった．低演算量アルゴリズムと高効率アーキテクチャにより携帯端末向け動き検出 VLSI の消費電力を 1mW 以下に抑える目標を達成した．

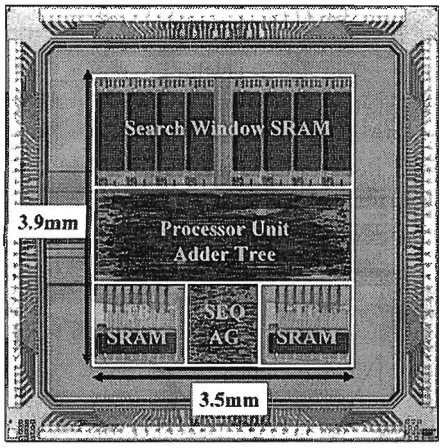


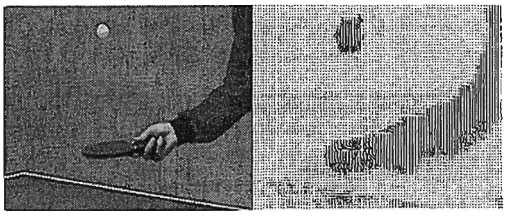
図 9．ME4 チップ写真

3. 画像認識用実時間動き推定 VLSI

車載，ロボット，監視といった応用に向けた高度な画像認識処理では時間的空間的に高解像度で高精度な実時間動き推定が重要である．ここでは画素毎の動きを求めるオプティカルフロー検出 VLSI[5]と画像全体の支配的な動きを求めるアフィン動きモデル推定 VLSI[6]について述べる．

3.1. オプティカルフロー検出 VLSI [5]

オプティカルフローは動画像における連続 2 フレーム間での画素毎の動きベクトルであり，動画像認識処理に使用される．図 10 にオプティカルフローの処理例を示す．静止した背景とラケットを振る腕とピンポン玉の動きが得られている．



(a)原画像(Table tennis)　(b)オプティカルフロー
図 10．オプティカルフロー例

Pyramidal Lucas and Kanade (PLK)アルゴリズム[7]は，Lucas and Kanade (LK)アルゴリズムに検出精度改善のための繰り返し処理と階層化手法を導入したものである．PLK アルゴリズムは他のアルゴリズムと比較して，演算量とメモリ容量とフレーム遅延が小さく，精度が高いという特徴を持つ．これらの H/W 実装上の利点から，本研究では PLK アルゴリズムを採用した．PLK アルゴリズムにおいて，オプティカルフロー $\mathbf{u}=(u,v)$ は式(4)を最小化する動きベクトルとして定義される．

$$E(u,v)=\sum_{(x,y)\in\Omega}\left(I(x,y)-J(x+u,y+v)\right)^2 \quad (4)$$

ここで I と J はそれぞれ連続する現フレームと次フレームの画素の輝度値であり，(x,y) は注目画素の座標，Ω は注目画素を中心とする小領域(ウィンドウ)を表す．I フレーム，J フレームでのウィンドウをそれぞれウィンドウ A，ウィンドウ B と呼ぶ．ウィンドウ A とウィンドウ B は $\mathbf{u}$ だけずれている．ここでウィンドウの一辺の画素数をウィンドウサイズ W と定義する．最小二乗法により E を最小化する $\mathbf{u}_{opt}$ を求める．式(4)を $\mathbf{u}$ で微分し左辺を 0 とおき，右辺の J をテイラー展開して整理すると，式(5)が導かれる．

$$\mathbf{u}_{opt}=\mathbf{G}^{-1}\cdot\mathbf{b}$$

$$\begin{cases}\mathbf{G}=\sum_{(x,y)\in\Omega}\begin{bmatrix}I_x^2(x,y) & I_x(x,y)I_y(x,y)\\ I_x(x,y)I_y(x,y) & I_y^2(x,y)\end{bmatrix}\\ \mathbf{b}=\sum_{(x,y)\in\Omega}\begin{bmatrix}I_x(x,y)I_t(x,y)\\ I_y(x,y)I_t(x,y)\end{bmatrix}\end{cases} \quad (5)$$

ここで，I_x，I_y，I_t はそれぞれ x，y，t 方向の輝度勾配であり，$\mathbf{G}$ と $\mathbf{b}$ をそれぞれ空間輝度勾配行列，ミスマッチベクトルと呼ぶ．式(5)を解くことでオプティカルフロー $\mathbf{u}_{opt}$ を得る．図 11 に PLK アルゴリズムのフローチャートを示す．ここで L は階層数，K は繰り返し回数を表す．最初に階層画像を作成する．そしてウィンドウ A の輝度値から I_x，I_y が，ウィンドウ A とウィンドウ B の輝度値から I_t が計算される．次に，$\mathbf{G}$ と $\mathbf{b}$ が計算され，最後に $\mathbf{u}$ が計算される．これらの処理は最上位レベルからレベル 1(原画像)まで繰り返され，最終的なオプティカルフローが得られる．

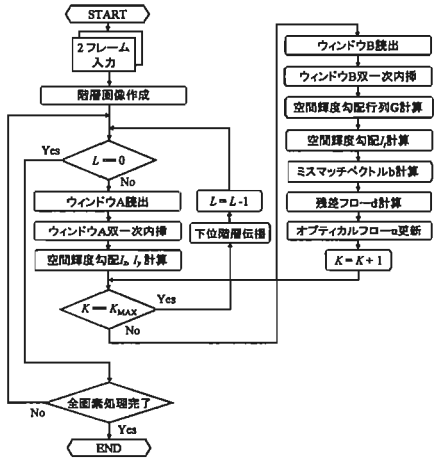


図 11．PLK アルゴリズムフローチャート

演算量と精度の観点からシミュレーションによりパラメタ値を決定し，L=3，W=11，K=1 とした．またフローの大きさに上限を設けて精度の劣化無くウィンドウ B 用のメモリ容量を削減した．最適化したアルゴリズムに基づき VLSI アーキテクチャを考案した．PLK プロセッサのブロック図を図 12 に示す． 本プロセッサは階層画像作成部(PIC)，空間輝度勾配算出部(SGM)，ミスマッチベクトル算出部(MMV)，オプティカルフロー算出部(OPF)などから構成されており，各ブロックが並列にパイプライン処理を行う．SGM と MMV 内の演算器をウィンドウサイズ W と同じ数だけ配置することで，1 サイクルでウィンドウ 1 行分の計算ができる並列度となっている．

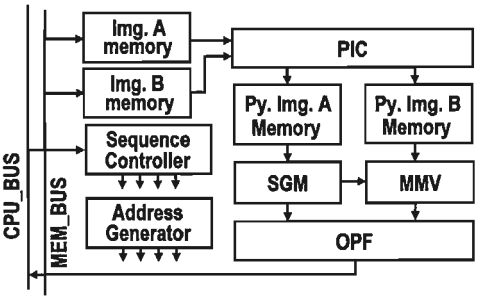


図 12．PLK ブロック図

VGA30fps を小チップ面積，高精度で実時間処理できるオプティカルフロープロセッサをスタンダードセルで実装を行った．SGM, MMV, OPF の演算部を 90nm プロセスで設計した．その結果，SGM は 1.50×0.84mm^2，MMV は 1.50×0.70mm^2，OPF は 0.50×0.84mm^2 となった．また全ブロックを含めたコアサイズは 3.50×3.00mm^2 であり，動作周波数は 332MHz である．

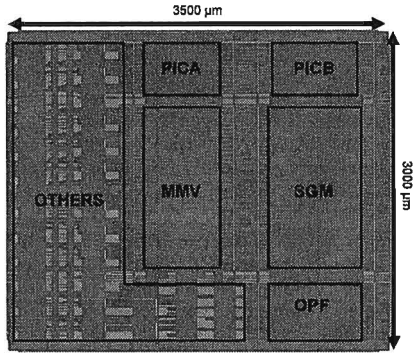


図 13．PLK プロット図

3.2. 実時間アフィン動き推定 VLSI

アフィン動き推定において動きは 2 次元のアフィン変換で表される．2 枚の連続画像から以下の 6 つのアフィンパラメタから成るアフィン動きモデル A が推定される．

$$A^t = (a_1\ a_2\ a_3\ a_4\ a_5\ a_6) \tag{6}$$

これらのパラメタ値の組合せでいろいろな動きを表現できる．図 14 は拡大の動きの例である．各点の動きベクトルは以下の式で表される．

$$\begin{cases} u_i = a_1 + a_2 x_i + a_3 y_i \\ v_i = a_4 + a_5 x_i + a_6 y_i \end{cases} \tag{7}$$

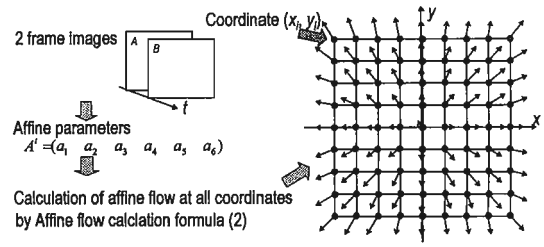


図 14．アフィン動きモデル

PSM アルゴリズム[8]は M-estimator を導入したロバストなアルゴリズムである．動きモデルを導出するエラー関数を以下に示す．

$$r_i = J(x_i + u_i, y_i + v_i) - I(x_i, y_i) + \xi \tag{8}$$

これは式(4)と同じであるが，領域全体の輝度変化 ξ を含んでいる．重み付き最小二乗法で領域全体のエラーを最小化する動きモデル θ を求める．

$$\begin{aligned} \boldsymbol{\theta} &= \mathbf{G}^{-1} \cdot \mathbf{G}_s \\ &= \left(\sum_{(x_i, y_i) \in F} w_i \boldsymbol{\chi}_i{}^t \boldsymbol{\chi}_i \right)^{-1} \left(\sum_{(x_i, y_i) \in F} w_i \boldsymbol{\chi}_i{}^t y_i \right) \end{aligned} \tag{9}$$

$$\begin{cases} \boldsymbol{\theta} = (\mathbf{A}^t \quad \xi) = (a_1 \quad a_2 \quad a_3 \quad a_4 \quad a_5 \quad a_6 \quad \xi) \\ y_i = -I_t \\ \boldsymbol{\chi}_i = (I_x \quad I_x x_i \quad I_x y_i \quad I_y \quad I_y x_i \quad I_y y_i \quad 1) \end{cases} \tag{10}$$

動きモデルと各点の動きの一致度により重みを決める．各点の重み w_i は評価値 q_i と定数 C を用いて計算する．

$$w_i = \begin{cases} (C^2 - q_i^2)^2 & if\ |q_i| < C \\ 0 & otherwise \end{cases} \tag{11}$$

$$q_i = \frac{\sum_{(x_i, y_i) \in \eta_i} \|\nabla I(x_i, y_i)\| DFDcomp(x_i, y_i)}{\sum_{(x_i, y_i) \in \eta_i} \|\nabla I(x_i, y_i)\|} \tag{12}$$

$$DFDcomp(x_i, y_i) = I(x_i + u_i, y_i + v_i, t+1) - I(x_i, y_i, t) + \xi \tag{13}$$

図15はPSMアルゴリズムのフローチャートを表す．階層処理の導入により大きな動きに対応している．最初に階層画像を作成し，最上位階層から推定を始める．各階層で繰返し動きモデルが計算される．最下位階層の動きモデル計算が終った後，各点の重みが計算され上位レベルに伝播される．動きモデル計算と重み計算が繰返されて画面全体において支配的な動きの正確なモデルが得られる．

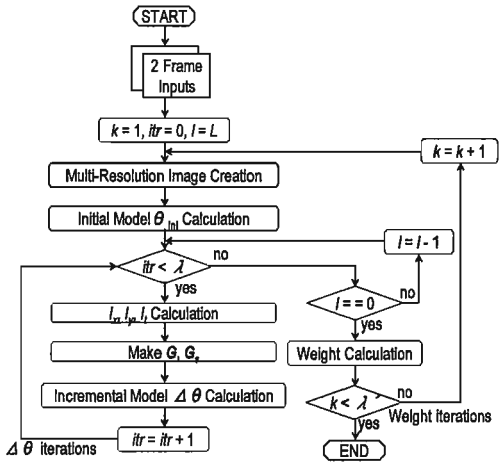


図 15. PSM アルゴリズムフローチャート

オリジナルのアルゴリズムに重みの2値化と画面分割法を導入することによりプロセッサと外部メモリ間のデータ転送量と内部メモリ量を削減した．画素サンプリング法によりクロック周波数を50%削減した．最適化したアルゴリズムに基づくプロセッサのアーキテクチャを図16に示す．本プロセッサは120MHz動作時にVGA 30 fpsの動きモデルを実時間計算可能である．

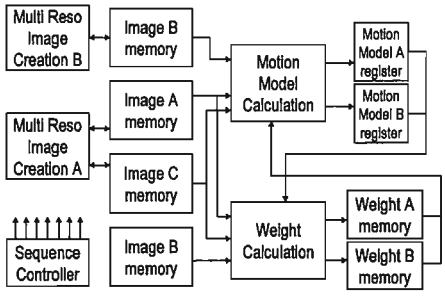
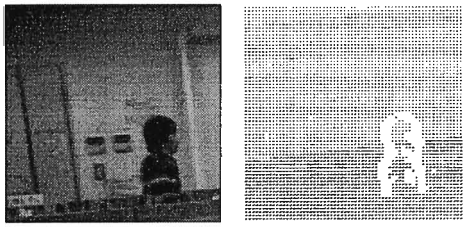


図 16. PSM アーキテクチャ

本プロセッサを FPGA 実装し，カメラから得られた画像を実時間処理した結果を図 18 に示す．カメラは横方向に平行移動しており，人物はカメラと反対方向に動いている．画面全体で支配的なカメラの動きが推定され，人物の周辺が重み 0 となって白く抜けていることが分かる．



(a) Shot image　　(b) Motion vector

図 18. 実験結果

4. まとめ

本稿では各種制約の中で最適な実時間画像処理 VLSI を実現する研究の一端として，実時間画像符号化 VLSI と画像認識用実時間動き推定 VLSI の実現手法について報告した．オプティカルフロー検出はジェスチャー認識や超解像度など，アフィン動き推定は動領域分割やオブジェクト符号化など様々な応用が可能であり，今後の課題である．本稿が読者の研究開発に役立てば幸いである．

謝　辞

本研究は東京大学大規模集積システム設計教育研究センターを通し，株式会社日立製作所および大日本印刷株式会社，シノプシス株式会社，日本ケイデンス株式会社，セロクシカ株式会社の協力で行われたものである．本研究の一部は株式会社ナナオとの共同研究である．本研究の一部は半導体理工学研究所および神戸大学吉本研究室との共同研究である．本研究の一部は科研費（19560339）の助成を受けたものである．

文　献

[1] M.Miyama, Y.Inoie, T.Kasuga, R.Inada, M.Nakao, Y.Matsuda, "A 158 MS/s JPEG 2000 Codec with a Bit-Plane and Pass Parallel Embedded Block Coder for Low Delay Image Transmission," IEICE TRAN. FUNDAMENTALS, vol.E91-A, no.8, pp.2025-2034, August 2008.

[2] M.Miyama, J.Miyakoshi, Y.Kuroda, K.Imamura, H.Hashimoto, M.Yoshimoto "A Sub-mW MPEG-4 Motion Estimation Processor Core for Mobile Video Application," IEEE JOURNAL OF SOLID-STATE CIRCUITS, vol.39, no.9, pp.1562-1570, September 2004.

[3] "Information Technology; JPEG 2000 Image Coding System– Part 1: Core Coding System", ISO/IEC 15444-1, Aug.2000.

[4] P.Kuhn, "Algorithms, Complexity Analysis and VLSI Architectures for MPEG-4 Motion Estimation," Kluwer Academic Publishers, 1999.

[5] Y. Murachi, H. Ishihara, Y. Fukuyama, R. Yamamoto, M. Miyama, H. Kawaguchi, Y. Matsuda, and M. Yoshimoto, "A VGA 30-fps Realtime Optical-Flow Processor Core for Moving Picture Recognition," IEICE TRAN. ELECTRONICS, vol.E91-C, no.4, pp.457-464, April 2008.

[6] Y. Yunbe, M. Miyama, and Y. Matsuda, "A VGA 30 FPS Affine Motion Estimation Processor for Real-Time Video Segmentation," IASTED Circuits and Systems Conference Proceedings, Hawaii, USA, August 2008.

[7] J. Y. Bouguet, "Pyramidal Implementation of the Lucas Kanade Feature Tracker Description of the Algorithm," Intel Corporation, Microprocessor Research Labs, OpenCV Documents, 1999.

[8] J.M. Odobez & P. Bouthemy, "Robust multiresolution estimation of parametric motion models applied to complex scenes," publication interne n° 1994, 788.