計測自動制御学会東北支部 第198回研究集会 (2000.6.23)
資料番号 198-9

オプティカルフローを用いた人物抽出とそのVLSI化

Human Extraction Using Optical Flow and Its VLSI Implementation

○張山昌論, 渡邊郷史, 李昇桓, 亀山充隆

○Masanori Hariyama, Satoshi Watanabe, Seunghwan Lee, Michitaka Kameyama

東北大学

Tohoku University

キーワード： 絶対差分和 (sum of absolute differences), ロジックインメモリアーキテクチャ (logic-in-memory architecture), ハイレベルシンセシス (high-level synthesis), 最適アロケーション(optimal allocation)

連絡先： 〒980-8579 仙台市青葉区荒巻字青葉05 東北大学大学院情報科学研究科亀山研究室
張山昌論, Tel.: (022)217-7154, Fax.: (022)263-9167, E-mail: hariyama@kameyama.ecei.tohoku.ac.jp

1. まえがき

高速かつ信頼性が高い人物抽出は, 侵入者検出, 高安全自動車, 危険監視システムなどの実現のための重要な要素技術である. 本稿では人物抽出手法として接続情報に基づく手法を提案する. この方法では画像中に人間候補領域を仮定し, その領域の形状と人間のモデルを照合することにより, 人間領域かどうか判定する.

この手法においては, 人間領域候補数が膨大となり計算量が膨大となる問題があるため, 候補領域を限定することが重要となる. そこで, 本稿では, 移動物体抽出を用いて人間候補領域を限定することを考える. 移動領域抽出として, 従来は計算量が少ないという理由から, 従来は画像間の差分に基づく方法がよく用いられている. しかしながら, この手法では物体が移動することによる背景画像の輝度値の変化の影響により誤抽出が生じる可能性が高いという問題点がある.

このような問題を解決するために, 本研究ではオプティカルフローを用いた移動物体抽出を提案する. オプティカルフロー検出は, 異なる時刻に取得された画像間の対応を求める処理である. 参照画像上のある画素の対応点を求めるためには, その画素を中心とする参照ウィンドウを設定し,候補画像上で全ての候補ウィンドウとの間で, ウィンドウ内の画素の濃度値の差分絶対値の和(SAD)を計算し, SADが最小となる画素を対応点として選択する. このように, 周囲の点とのSADを比較して移動ベクトルを決定するため, 画像間差分に基づく方法よりも, よりロバスト性が高い移動物体抽出が可能となる. このような対応点探索を参照画像上の全ての画素に対して行う必要があり, 計算量が膨大となるため, SAD演算の専用VLSI化が望まれる.

そのVLSIプロセッサにおいては、画像メモリと

演算部の転送ボトルネックの解消が重要となる．そこで，本稿ではバッファから演算部への並列データ供給を可能にするためのメモリアロケーションを提案する．バッファは複数のメモリモジュールから構成され，異なるメモリモジュールに記憶されるデータには並列にアクセスが可能である．その並列性を最大限に活用するために，任意の位置のウィンドウ内の全ての画素を異なるメモリモジュールに記憶できる長方形メモリアロケーションを提案する．

また，バッファと演算器間の並列データ供給のための相互結合網の複雑さを最小化するための演算器アロケーションを提案する．1個のメモリモジュールにマッピングされた画素に対する全ての絶対差(AD)演算を，1個のPEにマッピングすることにより，1個のメモリモジュールは1個のPEにだけ接続され，その結果，メモリ・演算器間の通信が完全に局所化されたロジックインメモリ構造が得られる．

0.8μmCMOS設計ルールを用いてオプティカルフロー抽出VLSIプロセッサを設計したところ，256 × 256画像，16 × 16 ウィンドウを用いた場合の処理時間は5msとなり，ワークステーションでの処理時間にくらべ，5桁以上の高性能化が達成された．

2. アルゴリズム

2.1 移動物体領域抽出に基づく人物抽出

Fig. 1に提案する人物抽出の処理の流れを示す．本システムでは，固定されたカメラを用いる．そのカメラから異なる時刻の画像を取り込む．次に，各画像に対して領域分割を行う．その後，各領域が移動領域かどうかを決定するための移動領域抽出を行う．抽出された移動領域に対して，細線化を行い，接続情報に基づき人間のモデルと照合を行うことにより，人間領域かどうかの判定を行う[1]．移動抽出を行うことにより，照合を行う候補数を大

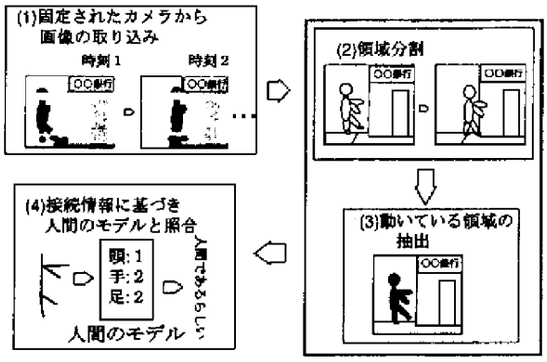

Fig. 1　移動物体領域抽出に基づく人物抽出の処理の流れ．

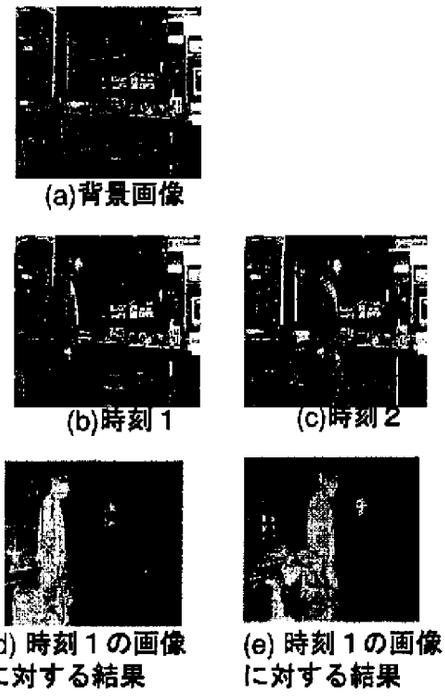

Fig. 2　背景差分に基づく移動物体抽出の例．

幅に減少でき，計算量の減少が可能となる．この処理においては，移動領域に対して人間かどうかの照合が行われるため，人間である領域を見逃さずに抽出できるような信頼性の高い人間抽出が必要となる．従来，人間抽出の手法としては，(1)背景画像との差分に基づく方法，(2)連続した画像間の差分に基づく方法，(3)オプティカルフローに基づく方法に大別される．従来は，計算量が少ないという理由から(1),(2)の方法がよく用いられている．しかしながら，これらの方法では，物体が移動することによる背景領域の輝度変化の影響により，背景

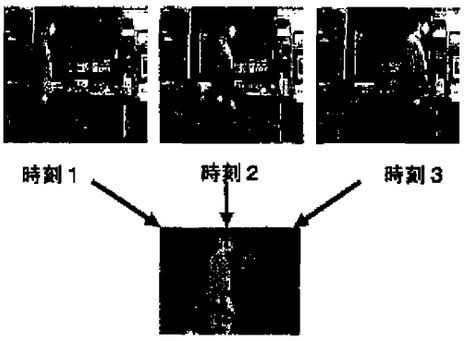

Fig. 3　連続した画像間の差分に基づく移動物体抽出の例.

部分であるにもかかわらず移動領域であるとし抽出される領域が多くなるという問題がある. Fig.2と3 に背景差分に基づく方法, 連続した画像間の差分に基づく方法の実験例を示す. 実験結果において, 移動したと判断された画素は白く示されている. この結果から,背景部分にも移動領域とみなされる画素が多く出現することがわかる. 一方, オプティカルフローに基づく移動領域抽出では, 各画素の輝度値の変化ではなく, 周囲の画素の変化量との比較を行い決定するため, よりロバストな移動領域抽出が行える. そこで, 本研究では, オプティカルフローを用いて移動領域を抽出する.

2.2　オプティカルフローを用いた移動領域抽出

オプティカルフロー検出は異なる時刻に取得された2枚の画像間の対応を求める処理である. Fig.4に示すように, 基準画像I_0上に, 座標(X_0, Y_0)のピクセルP_0を中心とする大きさ$N \times N$のウィンドウ

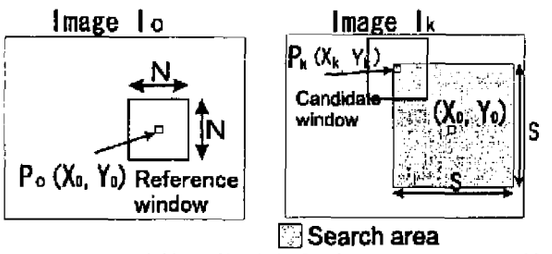

Fig. 4　SAD演算に基づくオプティカルフロー検出

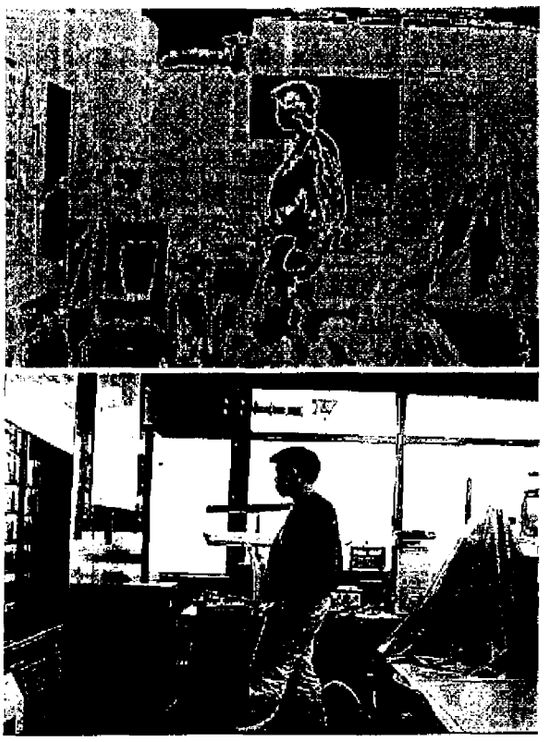

Fig. 5　オプティカルフローを用いた移動領域抽出の例.

を設定する. また, 候補画像I_k上に, 座標(X_0, Y_0)を中心とする大きさ$S \times S$のサーチエリアを設定する. 候補画像上にピクセルP_kを中心とする大きさ$N \times N$の候補ウィンドウを設定し, 中心ピクセルP_kがサーチエリアに含まれるような全ての候補ウィンドウに対して, 式(1)に示すSAD(Sum of Absolute Differences)を計算する.

$$SAD(P_0, P_k) = \sum_{i=-W/2}^{W/2} \sum_{j=-W/2}^{W/2} \left| I_0(X_0+i, Y_0+j) - I_1(X_1+i, Y_1+j) \right| \quad (1)$$

ここで, (X_k, Y_k)はピクセルP_kの座標, $I_k(X_k, Y_k)$はその画素値, Wはウィンドの一辺の長さである. SADが小さい程, 参照ウィンドウと候補ウィンドウが類似していることを示す. サーチウィンドウ内の各候補ウィンドウに対してSAD演算を行い, SADが最小となるウィンドウの中心画素を対応点とする.

オプティカルフローを用いた移動領域抽出の処理の流れを以下に示す.

Step 1: Fig. 1に示したように領域分割を行う. 領域数の増加は計算量の増加につながる. そこで, 領域分割においては, 従来のRGB表色系を用いた方法ではなく, HSI表色系を用いた手法を採用している. HSI表色系では色と輝度を分離でき, 同じ色の領域を1個の領域として抽出できるため, RGB表色系よりも領域数を減少できる.

Step 2: 領域の境界線を輪郭線として抽出する.

Step 3: 輪郭線の各画素に対して, オプティカルフロー検出を行う. 輪郭線は,領域内部の画素に比べ, 特徴がある画素であるので, より信頼性の高いオプティカルフローを抽出できる. 輪郭線上の全画素数をM, 輪郭線上の画素のうちフローベクトルの大きさが0ではない画素の数をNとする. 次式の条件を満たす領域を移動領域として抽出する.

ここで, THは実験的に求められたしきい値である.

$$\frac{M}{N} \geq TH \qquad (2)$$

Fig. 5にオプティカルフローを用いた移動領域抽出の実験例を示す. Fig. 5上の入力画像において, 中央の人間が左に移動している. Fig. 5下において, 黒い領域が移動領域として抽出された領域である. ほぼ人間だけが抽出されていることがわかる. この結果においては, 人間の脚部が移動領域として抽出されていない. これは, 画像を撮影した間隔が小さい(30frame/sec)ため,画像の変化が小さいことが原因であると考えられる. この問題を解決するために, 今後複数の画像対オプティカルフローを連鎖し, より長い時間でオプティカルフローを抽出する方法を検討中である.

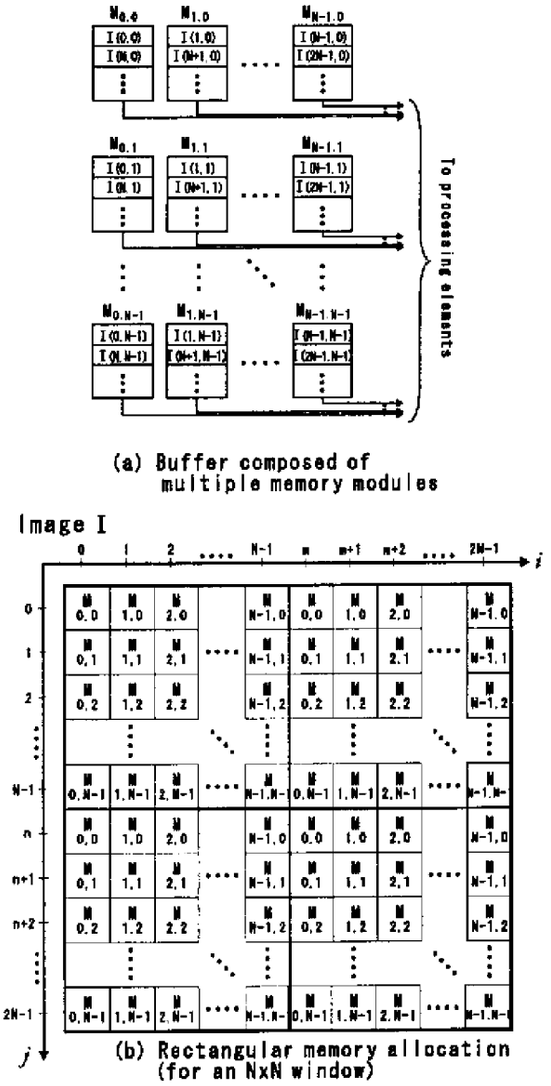

Fig. 6 長方形メモリアロケーションを用いた並列アクセス.

3. オプティカルフロー抽出用ロジックインメモリ構造VLSIプロセッサ

3.1 長方形メモリアロケーションに基づくバッファへの並列アクセス

SAD演算においてはウィンドウ内の画素に対して並列にAD演算を行える画素レベルの並列性と, 異なる候補ウィンドウに対して並列にSAD演算を行えるというウィンドウレベルの並列性がある. このような並列性を活用するために, ウィンドウ内のAD演算を並列に行い, 異なるウィンドウに対

するSAD演算をパイプライン化するそのためには,各コントロールステップにおいて,ウィンドウ内の全画素をバッファから演算部に転送する必要がある.並列アクセスのための有効な方法の一つとして,Fig.6(a)に示すように複数のメモリモジュールを用いてバッファを構成する方法がある.この方法では,異なるメモリモジュールに記憶される画素に対して並列にアクセスを行うことができる.このような並列性を最大限に活用するためには,ウィンドウ内の全ての画素を異なるメモリモジュールにマッピングするようなメモリアロケーションを求めることが重要となる.特に,以下の条件A1,A2,A3を満たすメモリアロケーションを最適なメモリアロケーションと定義する[2].

A1: 候補ウィンドウの位置によらずウィンドウ内の全ての画素は異なるメモリモジュールに記憶される.

A2: 各ピクセルは1個のメモリモジュールにだけマッピングされる.

A3: メモリモジュール数が最少である.

一般に,1個の画素を複数のメモリモジュールに重複して記憶すれば並列アクセスが可能となるが,メモリ容量が大きくなる.そこで,最小のメモリ容量で並列アクセスを可能にするために条件A2を考える.また,メモリモジュール数が多くなる程,アドレスバスの本数や,アドレスデコーダなどの制御のためのハードウェア量が増加する.そのため,条件A3によりメモリモジュール数の最小化を行う.

最適なメモリアロケーションとして,長方形アロケーションを提案する.$m \times n$ウィンドウに対する長方形メモリアロケーションにより,画素(i,j)はメモリモジュール$M_{Q(i),R(j)}$にマッピングされる.ここで,マッピング関数$Q(i)$, $R(j)$ はFig.6に示すように,次式で与えられる.

$$Q(i) = i \bmod m \quad , \quad R(j) = j \bmod n, \quad (3)$$

Fig.6(b)に示すように,$m \times n$ウィンドウ内の画素は全て異なるメモリモジュールに記憶されるため条件A1を満たす.また,マッピング関数$Q(i)$, $R(j)$はそれぞれ,i,j に対してただ一つ出力が決定される1価関数であるので,画素(i,j)はただ一つのメモリモジュールにだけマッピングされる.さらに,長方形メモリアロケーションでは$m \times n$個のメモリモジュールを必要とし,これは$m \times n$ウィンドウ内の画素を並列にアクセスするために必要な最少のメモリモジュール数である.したがって,長方形メモリアロケーションは最適なメモリアロケーションであることが分かる.

3.2 相互結合網の簡単化のための演算器アロケーション

SAD演算部を設計する際に最も大きな問題は並列データ転送を可能にする,できるだけ簡単な相互結合網を設計することである.メモリモジュールとPEの間の相互結合網を最も簡単にするために,長方形演算器アロケーションを提案する.長方形演算器アロケーションにより,候補画像上の画素(i,j)に関するAD演算はFig.7に示す2次元PEアレイ上の$PE_{Q(i),R(j)}$にマッピングされる,ここで,$Q(i)(= i \ mod \ N)$と$R(j)(= j \ mod \ N)$は長方形メモリアロケーションのためのマッピング関数である.画素(i,j) は長方形メモリアロケーションにより$M_{Q(i),R(j)}$にマッピングされるので,メモリモジュール$M_{Q(i),R(j)}$に記憶される画素に関するAD演算が$PE_{Q(i),R(j)}$にマッピングされる.その結果,メモリモジュール$M_{Q(i),R(j)}$は$PE_{Q(i),R(j)}$にだけ接続されることになる.

Fig.8に長方形メモリアロケーションの例を示す.メモリモジュール$M_{k,l}(k,l$は$0 \leq k,l \leq 3$を満たす整数)に記憶される画素に関するAD演算は$PE_{k,l}$にだけ割り当てられることがわかる.

また,参照ウィンドウ内の画素は繰り返し使用

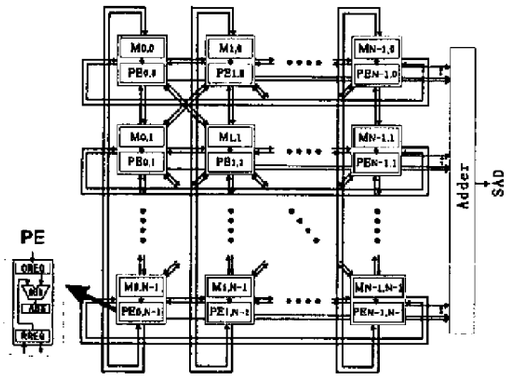

Fig. 7　長方形演算器アロケーションに基づくSAD演算部($N \times N$ウィンドウ).

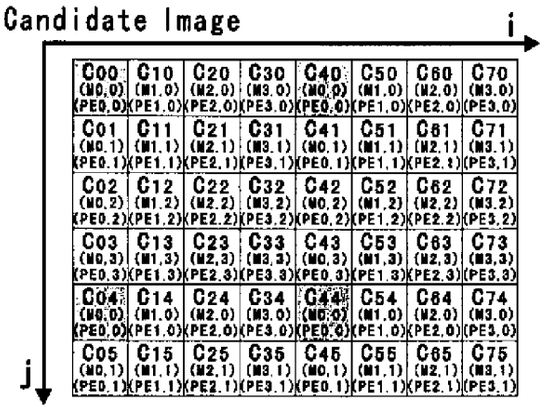

Fig. 8　4×4 ウィンドウに対する長方形メモリアロケーションと長方形演算器アロケーション.

されるため, 各PE内の参照レジスタに記憶している. これらの画素を再利用するためには, 異なる候補ウィンドウに対して演算を行う毎に, 各PE間で画素を転送する必要がある. 長方形演算器アロケーションにより, このPE間の相互結合網簡単も簡単化される. Fig.4に示したように, 隣接した候補点に対して連続的にSADが計算される. そのため,Fig.9に示すように, step u において参照ウィンドウ内の画素Rと候補ウィンドウ内の画素Vの間でAD演算が行われたとすると step $u+1$においてはRとVの8近傍の画素中の1画素との間でAD演算が行われる. ここで,Vの8近傍画素のどの画素に対してAD演算が行われるかは,エピポーララインの傾きにより決定される. この場合,RはVがマッピングされているPEから, Vの8近傍画素がマッピングされている8個のPEのどれか1個のPEに転送されることになる. Fig. 8からわかるように, 長方形演算器アロケーションは候補画像上の隣接した画素に関するAD演算をFig.7に示す2次元PEアレイの隣接したPEにマッピングする. したがって,参照ウィンドウ内の画素のPE間での転送は,常に隣接したPE間で行うことができる[3].

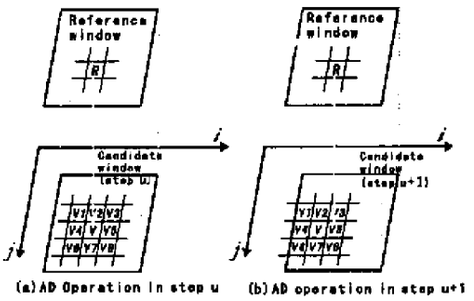

Fig. 9　連続したSAD演算.

4. 評価

Fig. 10に0.8μmCMOS設計ルールを用いて設計されたステレオビジョンVLSIプロセッサのレイアウトを示す. 256 × 256のサイズの画像に対するオプティカルフロー抽出を5msecで行える. その結果, ワークステーションでの処理と比較して, 40万倍の高速化が達成される. また, 並列メモリアクセスのためのメモリアロケーションを用いない場合と比較して, 16倍の高速化が達成される.

5. むすび

SAD演算に基づくオプティカルフローの抽出の信頼性は, SAD演算のウィンドウサイズに依存する. そのため, より信頼性の高いオプティカルフローを抽出するために, 画像に応じて最適なウィンドウサイズを決定するアルゴリズム及び, それに適合したアーキテクチャを検討することが重要となる.

参考文献

1) 張山昌論, 来山信康, 亀山充隆: 高安全知能集積システム用画像認識プロセッサの構成, 計測自動制御学会東北支部, 179-6,1998.

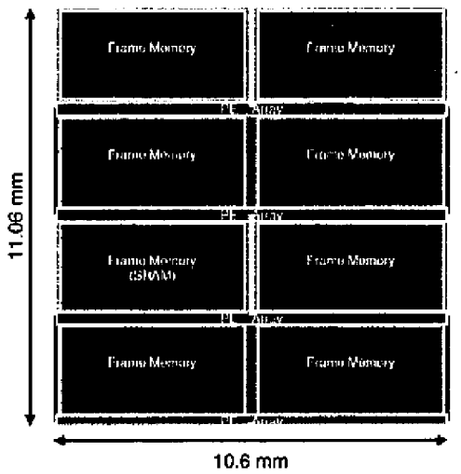

Fig. 10　VLSIプロセッサのレイアウト

2) Lee Seung, Masanori Hariyama, Michitaka Kameyama, "A Three-Dimensional Instrumentation VLSI Processor Based on a Concurrent Memory-Access Scheme," IEICE Trans. Electron, vol.E80-C, No.11, pp.1491-1498,1997.

3) 張山昌論, 李昇桓, 亀山充隆, "転送ボトルネックのないセンサ・メモリアーキテクチャに基づくモーションステレオVLSIプロセッサの構成," 電気学会論文誌, vol.120-E, no.5, pp.237-244,2000.