2L-1

渦巻探索の順序を考慮したSSDA法による相互相関法の並列処理手法

尾本 良子 † 佐藤 章浩 † 佐田 宏史 † 前川 仁孝 † 伊與田 光宏 †
† 千葉工業大学 情報科学部 情報工学科

1 はじめに

動きベクトル検出処理 (ME 処理) は, 時間的に連続するフレーム間の対応を求める処理である. ME 処理の評価尺度の一つである相互相関法は, 照明変動にロバストである一方で計算量が多く, 誤差和計算の打ち切りにより高速化する SSDA 法 [1] を適用しても処理に時間がかかる. 誤差和計算の打ち切りを効率良く行う探索手法である渦巻探索 [2] は, 動きベクトルが小さい場合に有効であるが, 動きベクトルが大きい場合は打ち切りの効率が低下する. そこで本稿では, 渦巻探索を内側と外側から異なるプロセッサで探索することで打ち切りを効率的に行う並列処理手法を提案する.

2 動きベクトル検出処理 (ME 処理)

ME 処理の代表的な手法にブロックマッチング法 (BM 法) がある. BM 法は, 図 1 のように前フレーム中のある領域をテンプレートとして用いて, 現フレーム中で最も類似度が高いブロックを求める手法である. 求めたブロックに向かう方向と移動量が動きベクトルとなる. フレーム間の類似度を表す評価尺度の一つに, 相関係数 R_x を求めてマッチングを行う相互相関法がある. テンプレートの輝度値を $T(i)$, 探索画像中の被参照ブロックの輝度値を $F_x(i)$ とすると, R_x は式 (1) によって表すことができる. また, 式 (1) 中の F_{xM}, T_M は, それぞれ式 (2), 式 (3) によって表すことができる.

$$R_x = \frac{\sum_{i=0}^{N-1}(F_x(i) - F_{xM}) \cdot (T(i) - T_M)}{\sqrt{\sum_{i=0}^{N-1}(F_x(i) - F_{xM})^2 \sum_{i=0}^{N-1}(T(i) - T_M)^2}} \quad (1)$$

$$F_{xM} = \frac{1}{N}\sum_{i=0}^{N-1} F_x(i) \quad (2)$$

$$T_M = \frac{1}{N}\sum_{i=0}^{N-1} T(i) \quad (3)$$

相互相関法では, テンプレートの輝度分散値を類似度に用いるため輝度の変化に影響されず, 他の評価尺度と比べて精度の高いマッチングができる. しかし, 正規化処理の計算量が膨大であるため, 処理時間が長い.

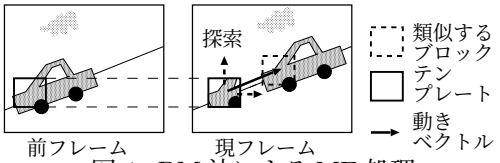

図 1: BM 法による ME 処理

3 相互相関法の高速化手法

BM 法において全探索の精度を保証しつつ高速化を図る手法として, 計算の打ち切りを行う SSDA 法や並列処理がある. これらの手法により, 相互相関法を評価尺度に用いた ME 処理を高速化できる.

A Parallel Normalized Correlation Coefficient Algorithm by a SSDA Method in Consideration of the Order of a Spiral Search
† Yoshiko Omoto, Akihiro Sato, Hiroshi Sata, Yoshitaka Maekawa, Mituhiro Iyoda
Dept. of Computer Science, Chiba Institute of Technology

3.1 渦巻探索を用いた SSDA 法による高速化

SSDA 法は, マッチングの際に求められる累積誤差和が閾値を超えた時点でそのブロックは類似度が低いと判断し, それ以降の誤差和の計算を打ち切り, テンプレートを移動する手法である. 相互相関法による SSDA 法では, 式 (1) を式 (4) のように変形して相関係数 R_x を求める [3].

$$R_x = H_x - \frac{\sum_{i=0}^{N-1}\{F_x(i) - T(i)\}^2}{K_x} \quad (4)$$

式 (4) の H_x, K_x を予め計算すると, R_x は誤差和計算により単調減少する関数となる. 計算が終了したブロックにおける R_x の最大値を閾値とし, 式 (4) 中で求める二乗誤差和計算の増加により, R_x が閾値を下回った時点で誤差和計算を打ち切ることができるため, 処理時間を短縮できる.

相互相関法による SSDA 法の閾値は, 過去に求めた評価値の最大値を用いるため, 計算を効率良く打ち切るには探索の初期段階で類似度が高いブロックを探索し, 閾値を更新する必要がある. 小さい動きベクトルの発生率が高い動画像を効率良く探索する手法に渦巻探索がある. これは, 探索領域の中心から外側へ渦巻状に探索することで, 探索の早い段階で誤差が小さいブロックを探索でき, 効率良く計算を打ち切ることができる. しかし, 図 2 に示すように大きな動きベクトルが生じる場合, 誤差の小さいブロックは中心付近に存在しないため誤差の小さいブロックの探索が遅くなり, 計算の打ち切り効率が低下する. このような場合は, 周囲の画素の相関性が高いことを利用して計算済みの周囲の動きベクトルから現画素の動きベクトルを予測し, 予測した付近から探索することで打ち切り効率の低下を防ぐ [4].

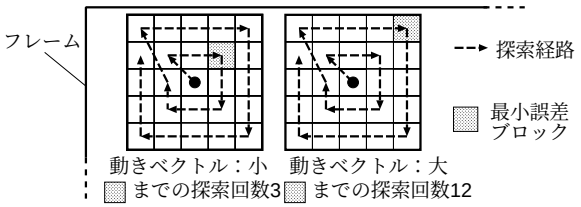

図 2: 渦巻探索による探索順序

3.2 並列処理による高速化

ME 処理は各画素間の依存関係がなく, 各画素を独立に処理できるため, 並列処理が可能である. 並列処理による ME 処理の高速化には, フレームやブロックを各プロセッサに分割する手法など, 従来より様々な手法がある [5]. SSDA 法を用いた BM 法を並列処理により高速化する場合, SSDA 法により計算を打ち切るタイミングが各ブロックで異なるため, 並列化の際にフレームをプロセッサ数で等分割すると各プロセッサの処理量が偏る. このため, 処理量の少ないプロセッサがアイドル状態となり並列効率が低下する. ブロック単位で並列処理するために, 図 2 中の 2 ブロックにそれぞれ 1 プロセッサ割り当てた場合は, 最小誤差ブロックまでの探索回数が, プロセッサ 1 は 3, プロセッサ 2 は 13 となる. よって, 計算を打ち切るタイミングがプロセッサ 1 の方が早くなりプロセッサ 1 はプロセッサ 2 の処理が終わるまでアイドル状態と

なる. このような場合, 各プロセッサ毎に割り当てる領域サイズを変化させ, 処理量を均等にすることで並列処理効率を向上させることができる.

4 探索順序を考慮したBM法の並列処理手法

従来の動きベクトルの予測を用いた渦巻き探索では, 現画素と周囲の画素が物体と物体の境目である場合, 周囲画素との相関性が低くなり予測精度が低下する. このため, 探索開始位置が最小誤差ブロック付近でなくなるため誤差が小さいブロックを探索するタイミングが遅くなり, SSDA 法における誤差和計算の打ち切り効率が低下する. また従来の並列処理手法では, 各プロセッサの処理量を均等にするためのオーバーヘッドが生じるため, 各プロセッサの処理量が均等になったとしても使用プロセッサ数以上の高速化を得ることは困難である. 従来の高速化手法を改善するために, 本稿では SSDA 法における打ち切りの効率を向上させることにより, 並列処理における使用プロセッサ数以上の高速化を図る. 動きベクトルが小さい場合は誤差の小さいブロックが内側に, 動きベクトルが大きい場合は誤差の小さいブロックが外側に存在することに注目し, 図 3 のように外側と内側から異なるプロセッサで並列に探索する手法を提案する.

本手法では, 1 フレームに対し 2 プロセッサを割り当て, 内側から探索するプロセッサは逐次と同様に探索する. 外側から探索するプロセッサは, 内側からの探索順序が最も遅くなるブロックから探索する. このように探索することで, 動きベクトルが小さい場合には, 内側から探索するプロセッサにより 1 プロセッサで探索する場合と同様に誤差が少ない部分から探索できるため, 逐次処理とほぼ同じタイミングで計算を打ち切ることができる. また, 動きベクトルが大きい場合には, 外側から探索するプロセッサにより, 誤差が少ない部分から探索することで, 逐次処理よりも早い段階で良い閾値を得ることができるため, 打ち切りの効率低下を防ぐことができる. 図 3 のように探索領域が 5×5 画素の場合, 探索する画素数は 25 である. 1 プロセッサで探索する場合に最小誤差ブロックまでの探索回数が 23 であったとすると, 外側からの探索では探索回数 2 で最小誤差ブロックに到達する. 外側からの探索によって更新された閾値は, 内側からの探索における閾値としても適用できる. このため, 1 プロセッサで探索する場合は最小誤差和ブロックに到達するまでの探索回数は 23 であるが, 2 プロセッサで探索する場合は最小誤差ブロックに到達する探索回数が 2 となる. よって動きベクトルが大きい場合は, 2 プロセッサにより探索することで誤差和計算の計算量が半分以下になる場合がある. このような探索手法により, 動きベクトルが大きい場合でも探索領域を 1 プロセッサで探索する従来の並列処理手法より効率的に計算が打ち切られ, 使用プロセッサ数倍以上の高速化が可能となる.

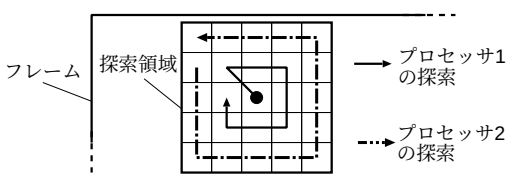

図 3: 提案手法による探索方法

5 評価

評価環境に共有メモリ型並列計算機 Sun Enterprise 4500 を用いた. CPU は UltraSPARC-IIs 400MHz であり, メモリは 4GB, OS は SunOS5.6, 最適化オプションに-fast を使用した. データは, テストシーケンスである foreman を用いた. 図 4 に提案手法と従来手法による ME 処理の実行時間を示す. 図 4 より, 提案手法による ME 処理の高速化率は最大で 3.8 倍, 全体で 2.6 倍の高速化率が確認できた. 提案手法では, 動きベクトルの大きさによらず早い段階で誤差の小さいブロックを探索できるため良い閾値が早く得られ, 閾値の更新も早い段階で行うことができる. これにより, 更新した良い閾値が内側と外側からの探索の両方に対して適用され, 早いタイミングで誤差和計算の打ち切り効率が向上したため高速化したと考えられる. 特に大きな動きベクトルが多く発生するフレームでは, 外側からの探索により 1 プロセッサによる探索よりも早い段階で誤差の小さいブロックを探索する回数が大幅に増えるため, 打ち切り効率が非常に高くなる. よって, 計算量が半分以下になる場合が生じ, 使用プロセッサ数以上の高速化率が得られたと考えられる.

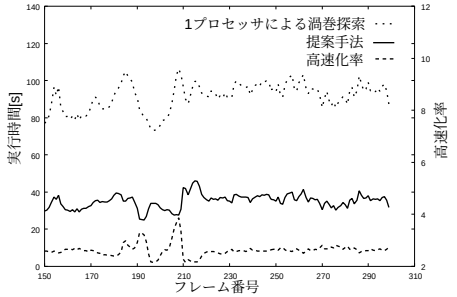

図 4: ME 処理の実行時間と高速化率

6 おわりに

本稿では, 動きベクトル検出処理における SSDA 法を適用した相互相関法の探索順序を考慮した並列処理手法を提案して評価した. 評価の結果, 1 プロセッサの探索に対して最大で 3.8 倍, 全体で 2.6 倍の高速化率が得られ, 提案手法の有効性を確認できた. 今後は更なる高速化を図るため, 動きベクトルが小さい場合の誤差和計算を削減する必要がある.

参考文献

[1] D.I.Barnea, “A class of algorithms for fast digital image registration”, IEEE Trans., Vol.C-21, pp.179-186, 1972.

[2] Th. Zahariadis, D. Kalivas, ”A Spiral Search Algorithm for fast estimation of block motion vectors”, in the proceedings of VIII European Signal Processing Conference, Vol.II, pp.1079-1082, 1996.

[3] 池田 光二, 吉田 昌司, 中島 啓介, 浜田 長晴, “正規化相関演算の単調関数化による高速テンプレートマッチング”, 信学論, Vol.J83-D-II, No.9, pp.1861-1869, 2000.

[4] 森吉 達治, 宮崎 孝, 黒田一朗, “動き適応探索順設定による動きベクトル検索の高速化”, 信学技報, Vol.99, No.25, pp.33-40, 1999.

[5] Daniel Grosu, “Motion Estimation in MPEG-2 Video Encoding Using a Parallel Block Matching Algorithm”, Proc. of the 6th International Symposium on Automatic Control and Computer Science, Vol.II, pp. 154-159, 1998