オーディオビジュアル複合情報処理 27－2
(1999. 12. 16)

二乗誤差を判定基準とするブロックマッチングの高速化

大脇 吉則　原田 雅樹　藤井 俊彰　木本 伊彦　谷本 正幸
名古屋大学大学院工学研究科
〒464-8603 名古屋市千種区不老町
Tel: 052-789-3163　　Fax: 052-789-3628
E-mail: ohwaki@tanimoto.nuee.nagoya-u.ac.jp

あらまし　ブロックマッチング法は画像処理の基本的な技術として様々な場面で用いられている．しかし，ブロックマッチングは計算量が多く，処理を行うのに時間がかかるという欠点がある．本稿では，二乗誤差をブロック間距離の判定基準として利用するブロックマッチングの高速化手法を提案する．二乗誤差の計算を行う前にブロック内要素の総和を利用した，簡単な計算によるブロック間距離の間接判定を行いブロックを選別することにより，二乗誤差の計算を行うブロック数を削減し，ブロックマッチングの精度を悪化させる事無く計算時間を短縮できることを示す．

キーワード　ブロックマッチング，高速化，二乗誤差，間接判定

Acceleration of Block Matching Which Uses Square Errors as a Criterion.

Yoshinori OHWAKI　Masaki HARADA　Toshiaki FUJII
Tadahiko KIMOTO　Masayuki TANIMOTO
Graduate School of Engineering, Nagoya University
Furo-cho, Chikusa-ku, Nagoya 464-8603 JAPAN
Tel: +81-52-789-3163　　Fax: +81-52-789-3628
E-mail: ohwaki@tanimoto.nuee.nagoya-u.ac.jp

Abstract　The block matching method is used in various scenes as a fundamental technology of the image processing. However, there are much calculation amount on block matching method, and there is a defect of taking time. In this paper, we propose high speeding technique of block matching which uses square error as a criterion of the distance between blocks. This paper shows that the indirect decision treatment by the simple calculation sorts out the block and this reduces the block which is calculated square error, and that it can reduce calculation time of the block matching.

key words　block matching, acceleration, square error, indirect decision

1 はじめに

ブロックマッチング法は,動画像符号化における動きベクトルの検出,視差画像での視差検出における対応点探索やフラクタル画像符号化でのドメインブロックの探索など様々な場面で用いられているが,処理時間がかかるという欠点を持っている.

ブロックマッチングの高速化をはかるためには,探索ブロックを少なくすればよい.しかしあらかじめ探索空間を小さく設定してしまうと,希望通りのマッチングの精度を得ることができなくなる.マッチングの精度を落とさずに探索ブロック数を削減するには,解とならないブロックのみを探索ブロックから省けばよい.そこで本稿では,ブロック内の要素の総和を用いて定められた探索空間の中からブロックマッチングの解とならないブロックのみを削減する手法を提案する.

2 二乗誤差を用いたブロックマッチング

ブロックマッチングとはあるブロック x にもっとも近いブロックをブロックの集合 Y から探索する処理である.本稿ではこれ以降 x をソースブロック,Y を探索空間,探索空間 Y 内のブロック y_j を探索ブロックと呼び,ソースブロック x,探索ブロック y_j の大きさ ($|x|$,$|y_j|$) を N,探索空間の大きさ ($|Y|$) を M とする.ブロック間距離を表す量としては,

- ブロック内画素の画素値の差の絶対値和
- ブロック内画素の二乗誤差の総和

がよく用いられているが,本稿ではこのうち二乗誤差を用いたブロックマッチングを扱う.ソースブロック x,探索空間 Y をそれぞれ

$$x = [x_1, x_2, ..., x_N]$$

$$Y = [y_1, y_2, ..., y_M]$$

$$y_j = [y_{j1}, y_{j2}, ..., y_{jN}]$$

とすると二つのブロック x と y_j のブロック間距離 $D(x, y_j)$ は二乗誤差を用いる場合

$$D(x, y_j) = \sum_{i=1}^{N} (x_i - y_{ji})^2 \tag{1}$$

と表すことができ,$D(x, y_j)$ が最小の値を取る時のブロック y_j が x にもっとも近いブロックとなる.D の最小値を求めるには,D の暫定的な最小値 D'_{min} と $D(x, y_j)$ を比較し,$D'_{min} > D(x, y_j)$ なら D'_{min} の値を $D(x, y_j)$ にするという操作をすべての探索ブロックについて行うが,D'_{min} と $D(x, y_j)$ の比較を行う際に,$D(x, y_j)$ の値を計算してから D'_{min} の値と比べるのではなく,$(x_i - y_{ji})^2$ を 1 つ加えるごとに D'_{min} との比較を行い,和が D'_{min} 以上になった時点で計算を打ち切るという手法が用いられている.そのアルゴリズムを図 1 に示す.

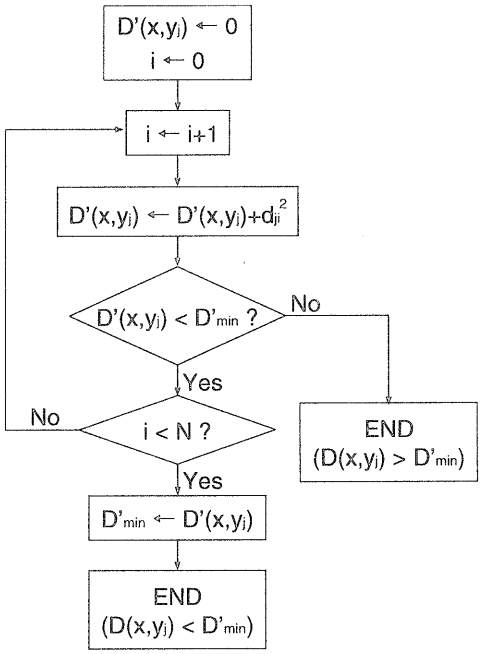


図 1: 従来法のアルゴリズム

3 提案法

3.1 間接判定による計算量の削減

ソースブロック s と探索空間 Y 内のブロック y_j の同位置における要素どうしの差の集合を d_j とおくと,

$$d_j = [d_{j1}, d_{j2}, ..., d_{jN}]$$

$$d_{ji} = x_i - y_{ji} \tag{2}$$

式 (1) は式 (2) を用いて,

$$D(x, y_j) = \sum_{i=1}^{N} d_{ji}^2 \tag{3}$$

と表すことができる.また,d_j の平均 ($E(d_j)$),分散 ($V(d_j)$),二乗平均 ($E(d_j^2)$) の関係は,

$$V(d_j) = E(d_j^2) - E(d_j)^2 \tag{4}$$

と表され,

$$E(d_j) = \frac{1}{N} \sum_{i=1}^{N} d_{ji} = \frac{1}{N} \left(\sum_{i=1}^{N} x_i - \sum_{i=1}^{N} y_{ji} \right) \tag{5}$$

$$E(d_j^2) = \frac{1}{N} \sum_{i=1}^{N} d_{ji}^2 = \frac{1}{N} D(x, y_j) \tag{6}$$

$$V(d_j) = \sum_{i=1}^{N} \{d_{ji} - E(d_j)\}^2 \geq 0 \qquad (7)$$

が成り立つので，式 (4), (5), (6),(7) より，

$$\frac{1}{N} D(x, y_j) - \left\{ \frac{1}{N} \left(\sum_{i=1}^{N} x_i - \sum_{i=1}^{N} y_{ji} \right) \right\}^2 \geq 0 \qquad (8)$$

この式の両辺を N 倍して整理すると，

$$D(x, y_j) \geq \frac{1}{N} \left(\sum_{i=1}^{N} x_i - \sum_{i=1}^{N} y_{ji} \right)^2 \qquad (9)$$

となり，ブロック x とブロック y_j の距離 $D(x, y_j)$ はそれぞれのブロックのブロック内の要素の総和の差の二乗を $\frac{1}{N}$ 倍したものより大きな値を取ることがわかる．このことを利用して，以下のアルゴリズムによりブロックマッチングの高速化をはかることができる．

1. (初期設定)
   ブロック x の要素の総和を計算し，sum_x に保存する．

$$sum_x \leftarrow \sum_{i=1}^{N} x_i$$

   D'_{min} の初期値として $D(x, y_1)$ を設定する．

$$D'_{min} \leftarrow D(x, y_1)$$

2. (マッチング操作)
   探索空間内の y_1 以外のすべてのブロック y_j について以下の操作を行う．

   2-1 (間接判定)
   ND'_{min} と $\left(sum_x - \sum_{i=1}^{N} y_{ji}\right)^2$ の大きさを比較する．
   $ND'_{min} \leq \left(sum_x - \sum_{i=1}^{N} y_{ji}\right)^2$ ならば次のブロックについて同様の操作を行う．
   そうでなければ次へ進む．

   2-2 (直接判定)
   D'_{min} と $D(x, y_j)$ の大きさを比較する．
   $D'_{min} \leq D(x, y_j)$ ならば次のブロックについて同様の操作を行う．
   そうでなければ次へ進む．

   2-3 (最小値の更新)

$$D'_{min} \leftarrow D(x, y_j)$$

3. (終了)

3.2 間接判定処理の高速化

提案法では，間接判定を行うために従来法では行っていない各ブロックの要素の総和を求めている．この計算を通常のように行うと，表 1 のようになる．この計算時間は，探索空間が大きい場合にはあまり影響はないが，探索空間が小さい時には，表 1 のように無視できないレベルであり，ここで費す時間のために従来法よりも時間がかかってしまう場合がある．

表 1: 計算時間

探索空間	提案法 (秒)			従来法 (秒)
	和の計算	その他	合計	
9×9	2.06	1.51	3.57	1.70
17×17	2.06	3.86	5.92	4.73
33×33	2.06	9.56	11.62	13.01
65×65	2.06	21.07	23.13	31.47

探索空間内のブロックがそれぞれ重なっている場合，図 2 のように横方向に 1 画素ずれているだけのブロックなら y_j と y_k が重なっている部分の要素の和は共通であるので，すでに y_j のブロック内の要素の総和が求まっているなら y_k のブロック内の要素の総和は y_j のブロック内の要素の総和から A の要素を引き B の要素を加えることによって求めることができる．また縦方向にも同様に考え，B の要素の総和は C の要素の総和から D を引き E を足すことにより求まる．

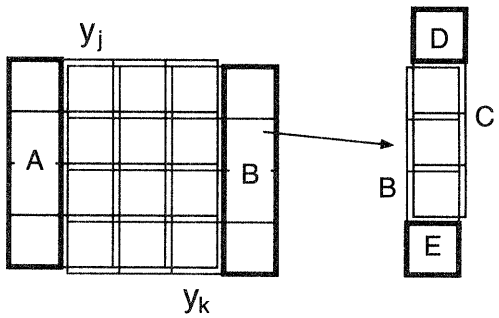


図 2: ブロックの重なり

探索空間内の左上のブロックから計算をしていくとすると，図 2 の y_k を求める時には y_j, A, C はすでに求まっているので，1 ブロックのブロック内の要素の総和を求めるための計算量は，まず B を求めるのに加算が 1 回，そして y_k を求めるのに加算が 1 回減算が 1 回必要となる．よって合計 4 回の加減算のみでブロック内の要素の総和を求めることができるので，この部分での大幅な高速化が期待できる．

4 シミュレーション実験

4.1 実験条件

動画像の 2 フレーム間の動きベクトルをブロックマッチングにより求める実験を以下の条件で行う．

使用画像 salesman 及び flower garden をグレイスケールに変換したもの

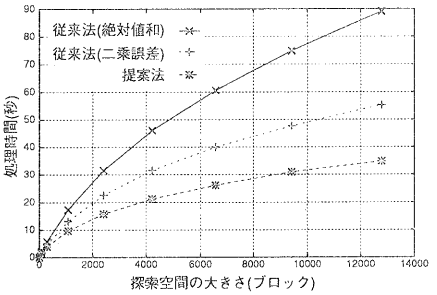
ブロックサイズ 8 画素×8 画素

マッチング精度 1 画素

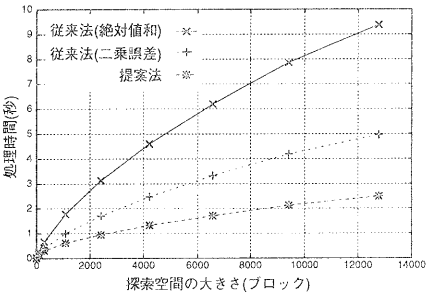
ブロックの探索順序 探索空間の中心から渦巻状

4.2 実験結果

図 3 は探索空間の大きさをを変化させたときの処理時間を示している．図中従来法 (絶対値和) はブロック間距離としてブロック間の各要素の差の絶対値和を用いたものを，従来法 (二乗誤差) はブロック間距離としてブロック間の各要素の二乗誤差の和を用いたものを，提案法は 3 章で提案した手法を表す．



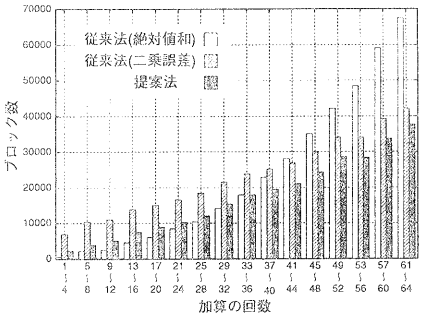
(a) flower garden (720 × 480)



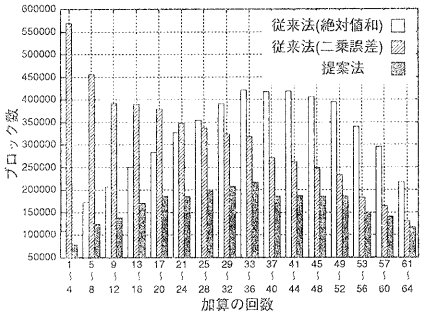
(b) salesman (352 × 288)

図 3: 探索空間の大きさを変えた時の処理時間の変化

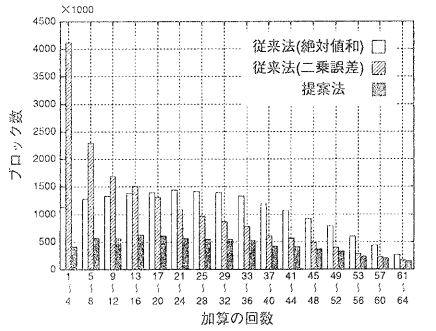
4.1 節の条件で 2 つの画像についてそれぞれ 4 種類の探索範囲における計算量を求めた．結果を表 2 および図 4，5 に示す．



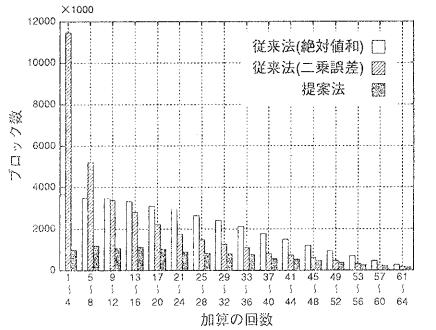
(a) 探索空間：9 × 9 ブロック



(b) 探索空間：33 × 33 ブロック

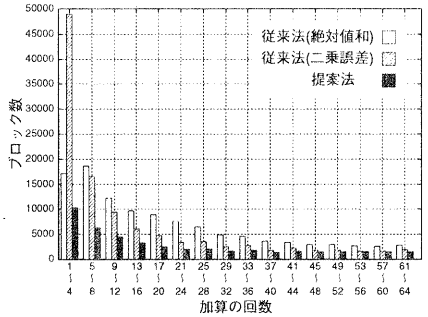


(c) 探索空間：65 × 65 ブロック

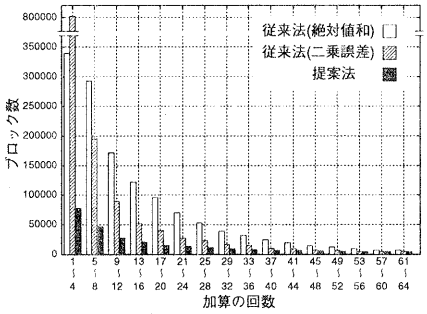


(d) 探索空間：97 × 97 ブロック

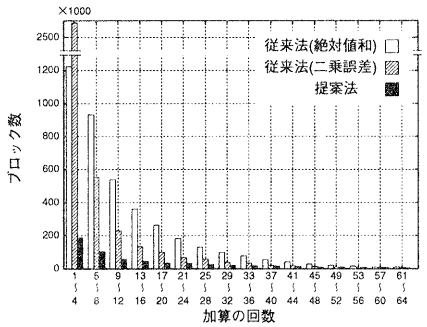
図 4: 様々な探索空間サイズにおける，ブロック間距離判定時の加算の回数とそのブロック数 (flower garden)



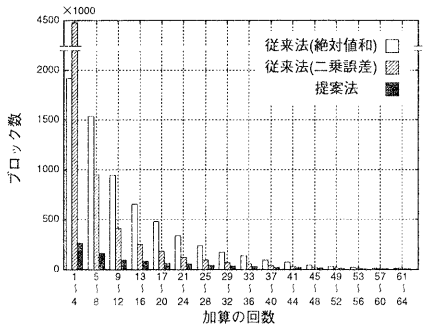
(a) 探索空間：9 × 9 ブロック



(b) 探索空間：33 × 33 ブロック



(c) 探索空間：33 × 33 ブロック



(d) 探索空間：97 × 97 ブロック

図 5: 様々な探索空間サイズにおける，ブロック間距離判定時の加算の回数とそのブロック数 (salesman)

表 2: 間接判定により削減されたブロック数

flower garden

探索空間	削減ブロック数	総ブロック数	削減率
9×9	92518	413424	22.4%
33×33	2345692	5091075	46.1%
65×65	10328040	17453475	59.2%
97×97	22764522	34051171	66.9%

salesman

探索空間	削減ブロック数	総ブロック数	削減率
9×9	65648	115668	56.8%
33×33	1042148	1316601	79.1%
65×65	3391336	3992625	84.9%
97×97	5766160	6708617	86.0%

表 2 は提案法の間接判定によって削減されたブロックの数及び総ブロック数に対する割合である．

図 4，5 は二乗誤差及び差の絶対値の和が D'_{min} より大きくなるまでに行った加算の回数とブロック数の関係を表している．二乗誤差の総和及び絶対値の総和が D'_{min} より小さかったブロックについてはこの図には含まれていない．また，提案法において間接判定の結果二乗誤差の計算を行わなかったブロックについても図には含まれていない．

5 考察

図 3 および表 2 に示すように，flower garden，salesman とも探索空間が大きくなるにつれて，探索ブロックの削減率は大きくなっているが，計算時間の削減率はどちらの画像とも探索空間の大きさによらずほぼ一定となってる．図 4，5 に示されているように，探索空間の大きいブロックでは，探索空間の小さいブロックに比べて，加算の回数が少ないブロックでの削減率が大きく，加算の回数が多いブロックでは，削減率が悪くなってるという傾向があるので，削減したブロックの全体的な割合が大きくなっていても，計算量の削減率は探索範囲の大きさによらずほぼ一定であるためと思われる．

本手法では，ブロック間距離が近いブロックが探索空間の探索の早い段階で現われるほど，d'_{min} の減少が早く間接判定に於いて，より多くの探索ブロックを削減することができ，また，実際のブロック間距離の判定においても少ない計算量で処理を打ち切れると予想される．本稿において，flower garden に比べて動きの少ない saleseman の方が探索ブロックの削減率がよかったことからもブロック間距離が近いブロックをいかに早く見つけるかが重要であるといえる．今回の実験では，探索ブロックの探索順序は，探索空間の中心から渦巻状に探索を行っていったが，探索の開始地点を固定するのではなく，周囲のブロックの動きベクトルから予測を行い，そこを開始地点とすれば更によい結果が得られると思われる．

また，ブロック間距離の計算時に d_{ji} が大きいものか

ら計算を行うことができれば，より早い段階で計算を打ち切れることが予想される．

6 むすび

本稿では，二乗誤差をブロック間距離の判定基準として用いるブロックマッチングをブロック内の要素の総和を用いてブロック間距離の間接判定を行うことで探索ブロックを削減し，高速化をはかる手法を提案した．シミュレーションの結果，提案法を用いることによって，探索空間の大きさにより，20%～85%の探索ブロックを削減し，探索空間のサイズによらず処理時間が40%～50%削減されることが確認された．

参考文献

[1] 大脇, 藤井, 木本, 谷本,, “画像のブロックマッチングの高速化,” 電子情報通信学会, 1999.

[2] 安居院, 長尾, “画像の処理と認識,” 昭晃堂, 1992