2007－CVIM－157 (27)
2007／1／12

代表サブテンプレートの位置を考慮した適応的ウィンドウスキップによる高速テンプレートマッチング法

坂田峻一† 外山史† 東海林健二† 宮道壽一†
†宇都宮大学工学部

あらまし テンプレートマッチングにおいて、現在用いられている高速化手法として適応的ウィンドウテンプレートマッチング法(Adaptive Window-Skipping Method : AWS)がある。AWSは、テンプレート内のサブテンプレート間の類似性を利用して、総当たり法同様の精度を保証してテンプレートマッチングを高速に行う手法である。AWSでは、前処理としてサブテンプレートの中から基準となる代表サブテンプレートを選択する。ここで、選ばれた代表サブテンプレートの位置によって処理時間が異なるが、AWSでは代表サブテンプレートの選択方法について特に考慮されていない。本研究では、AWSにおける代表サブテンプレートの選択方法についての検討を行い、より高速に探索することができる手法を提案する。実験の結果、従来のAWSよりも高速にマッチングを行うことができた。

A Fast Template Matching Using Adaptive Window Skipping Method Considering Posiiton of Reference Template

Shunichi Sakata† Fubito Toyama† Kenji Shoji† Juichi Miyamichi†
†Faculty of Engineering, Utsunomiya University

Abstract Adaptive Window-Skipping Method(AWS) is known as the fast template matching algorithm. This method can reduce the computational cost using similarity between subtemplates. AWS guarantees the same accuracy as an exhaustive search with template matching. In AWS preprocessing, a reference subtemplate is selected from the subtemplates. The position of the reference subtemplate affects the computational cost. But this position is not taken into account in AWS. In this paper, we propose a new method which takes into account the position of reference subtemplate. Experimental results show our method faster than AWS.

1 まえがき

画像から目的の物体の位置を見つけ出す技術にテンプレートマッチング法[1]がある。この手法は、図1のように既知の画像パターン（テンプレート）と入力画像上の多数の部分領域（ウィンドウ）との距離計算を行い、距離が最小のウィンドウの位置を求め、目的の物体の位置を検出する手法である。テンプレートマッチング法は、画像データベースからの目的画像の検索、映像からの物体の追跡・検索など幅広い分野で用いられている。しかし、テンプレートマッチング法はテンプレートとウィンドウの距離計算にかかる計算量が多く、また入力画像上でウィンドウを走査して照合を行うため、入力画像の枚数やテンプレートのサイズに比例して距離計算回数が大きくなり、計算時間が膨大になるという問題がある。

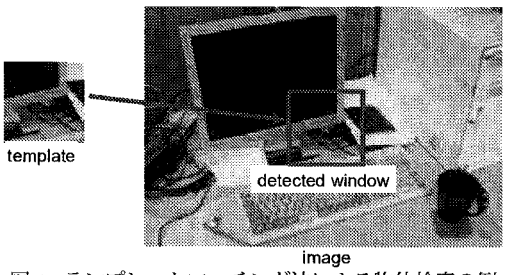

図 1: テンプレートマッチング法による物体検索の例

そのため、これまで多くの高速化手法が提案されてきた。代表的な手法として、テンプレートとウィンドウの距離計算の計算量を削減することにより、処理を高速化する残差逐次検定法(Sequential Similarity Detection Algorithm : SSDA)[2]や、テンプレートとウィンドウの照合の回数を削減することによって処理を高速化す

る2段階テンプレートマッチング法[3]、テンプレートの色ヒストグラムを利用して処理を高速化するアクティブ探索法[4]などが挙げられる。

SSDAは、テンプレートとウィンドウの距離計算の際、計算中の距離が単調増加することに注目し、計算途中で随時閾値θと比較し、距離が閾値θを超えたとき、その位置における計算処理を終了することで計算量を削減している。ここで、閾値θはそれまでに求めたテンプレートとウィンドウの距離の最小値である。SSDA法では、閾値θの初期値θ_0をテンプレートとウィンドウの距離の取り得る値の最大値にすることで最も類似する領域を出力することができる。また、$\theta_0 = \theta'$とすることで、テンプレートとの距離がθ'以上のウィンドウを処理の最初から省略できるため、θ'が小さい場合に探索処理を大幅に削減することができる。

2段階テンプレートマッチング法[3]は、テンプレートの部分領域であるサブテンプレートを用いて、各ウィンドウの距離下限値を求め、距離下限値が閾値θ以下のウィンドウのみテンプレート全体の照合を行うことで、計算量を削減する手法である。

アクティブ探索法は、あるウィンドウでの照合結果を利用してその近傍ウィンドウの類似度の上限を求め、上限値が閾値を下回る場合に照合を省略する手法である。

また、その他の手法として適応的ウィンドウスキッピングテンプレートマッチング法（Adaptive Window-Skipping Method:AWS）[5][6]がある。AWSはテンプレートの部分領域であるサブテンプレートとウィンドウの部分領域であるサブウィンドウを比べることによってサブウィンドウを内部に含む各ウィンドウの距離下限値を計算し、その距離下限値が閾値θ以下のウィンドウのみテンプレート全体の照合を行うことで処理を高速化する手法である。AWSはテンプレートマッチング法と同じ精度を保証しながら、テンプレートマッチング法やSSDAよりも高速に処理を行うことができる。AWSでは、前処理としてサブテンプレートの中から基準となる代表サブテンプレートを選択する。この代表サブテンプレートとすべてのサブテンプレートの距離を前処理であらかじめ求めておき、探索処理時にこれらの距離と計算した代表サブテンプレートとサブウィンドウとの距離を用いて周辺ウィンドウの距離下限値を計算する。求めたウィンドウの距離下限値が閾値θを超えたとき、そのウィンドウ照合処理を省略する。ここで、代表サブテンプレートと他のすべてのサブテンプレートの距離が距離下限値を求める計算に使用されている。そのため、どのサブテンプレートを代表サブテンプレートとして選択するかによって距離下限値の値が変化し、処理時間が異なる。しかし、AWSでは代表サブテンプレートは単にテンプレートの左上隅のサブテンプレートを選択しており、代表サブテンプレートの選択方法については特に考慮されていない。

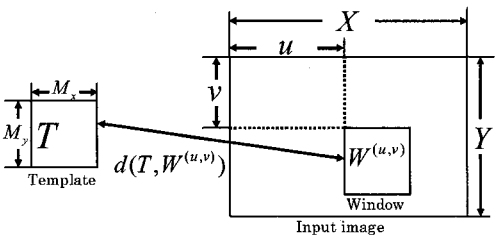

図 2: 問題の設定

そこで、本研究ではAWSにおいて代表サブテンプレートを選ぶ際、他のサブテンプレートとの距離の総和が最も少ないサブテンプレートを代表サブテンプレートとして選び、探索時に照合を省略することができるウィンドウをを増やすことで、処理を高速化する手法を提案する。本研究で提案した手法は実験の結果、従来のAWSよりも高速にマッチングを行うことができた。

2 問題の設定

本研究では、図2にのようにテンプレートを$\mathbf{T}$、入力画像のある場所(u,v)におけるウィンドウを$\mathbf{W}^{(u,v)}$とし、テンプレート$\mathbf{T}$とウィンドウ$\mathbf{W}^{(u,v)}$における距離を$d(\mathbf{T},\mathbf{W}^{(u,v)})$と記述する。また、この距離に式(1)で表される絶対値距離(Sum of Absolute Difference:SAD)を用いた。SADは計算量が小さいためによく用いられる距離である。

$$SAD: d(T,W^{(u,v)}) = \sum_{p=0}^{N-1} |T^p - W^p| \qquad (1)$$

ここで、T^p, W^pは、$\mathbf{T},\mathbf{W}^{(u,v)}$のそれぞれ$p$番目の画素の濃度値であり、$N$は、$\mathbf{T}$と$\mathbf{W}^{(u,v)}$の画素の総数である。テンプレートのサイズを$(M_x \times M_y)$,入力画像のサイズを$(X \times Y)$としたとき、$(u,v)$はそれぞれ、$0 \leq u < X - M_x$,$0 \leq v < Y - M_y$の範囲を動く。このため、ウィンドウの総数は、$(X - M_x) \times (Y - M_y)$になる。

3 テンプレートマッチング法

テンプレートマッチング法は、入力画像から与えられたテンプレートと最も類似するウィンドウの位置を検出する手法である。テンプレートマッチング法では、探索処理においてテンプレートを入力画像上で走査して、全てのウィンドウとの距離計算を行う。そのため、ウィンドウの総数$(X - M_x) \times (Y - M_y)$回の距離計算を行う。つまり、存在する全てのウィンドウにおいて距離$d(\mathbf{T},\mathbf{W}^{(u,v)})$を計算しなければならない。探索処理の結果、最も$d(\mathbf{T},\mathbf{W}^{(u,v)})$が低い位置$d(u,v)$を結果

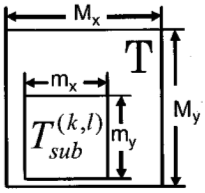

図 3: サブテンプレート

として出力する。テンプレートマッチング法はその処理手順上、入力画像のサイズやテンプレートのサイズに比例して計算量が膨大になってしまい、処理に要する時間が大きくなってしまうという問題がある。

4 AWS

適応的ウィンドウスキッピングテンプレートマッチング法(AWS)は、テンプレートの一部(サブテンプレート)とウィンドウの一部(サブウィンドウ)の距離から周辺ウィンドウの距離下限値を求め、その下限値が閾値θを上回る場合に照合の省略を行う。具体的には、探索処理においてテンプレートとウィンドウの距離計算の結果、その距離が閾値θより大きいときに、サブテンプレートとサブウィンドウの距離を用いて、近傍のウィンドウの距離下限値を求める。ウィンドウ全体に対しての距離計算を行う前に、そのウィンドウに設定された距離下限値が閾値θを超えるかどうかを判定し、距離下限値が閾値θを超える場合には、そのウィンドウでの距離計算を省略することで不要な距離計算をスキップし、処理を削減する。AWSは、テンプレートマッチング法と同じ精度を保証して、不要な距離計算を適応的にスキップしながら探索を行うことができる。

4.1 サブテンプレート

AWSにおけるサブテンプレートの概念を図3に示す。サブテンプレートは、テンプレートの一部の領域を切り出したもので、そのサイズは(m_x, m_y)である。本研究では、テンプレート$\mathbf{T}$の位置(k,l)におけるサブテンプレートを$\mathbf{T}_{sub}^{(k,l)}(0 \leq k < M_x - m_x, 0 \leq l < M_y - m_y)$と表記する。

文献[6]より、AWSではサブテンプレートサイズ(m_x, m_y)はそれぞれ、テンプレートサイズの約8割から9割のときに最も少ない実行時間となることがわかっている。そのため、本研究では、(m_x, m_y)はそれぞれ(M_x, M_y)の85 %の値とした。

4.2 AWSの処理の概要

AWSの処理手順は、大きく分けて前処理と探索処理に分けられる。前処理では、サブテンプレートの中から基準となる代表テンプレート$\mathbf{R}_{sub}$を選ぶ。このとき、AWSは代表サブテンプレートの位置は考慮しておらず、テンプレートの左上隅のサブテンプレート$\mathbf{T}_{sub}^{(0,0)}$を代表サブテンプレートとして選択する。その後、代表サブテンプレートと他の全てのサブテンプレートとの距離を計算する。その他前処理では、ウィンドウの距離下限値の初期化や閾値θの初期化を行う。次に、探索処理において、テンプレートを入力画像上で走査し、対応するウィンドウとの距離を計算する。このとき、図4のように$d(T,W)$ではなく、まず$d(R_{sub}, \mathbf{W}_{sub})$を計算する。次に、求めた$d(\mathbf{R}_{sub}, \mathbf{W}_{sub})$とあらかじめ前処理で求めておいたサブテンプレート間距離$d(\mathbf{R}_{sub}, \mathbf{T}_{sub})$を利用して、**4.3**で述べる三角不等式を使った手法を用いて$\mathbf{W}_{sub}$を内部に含む全てのウィンドウ$\mathbf{W}'$の距離下限値を求める。これにより、一度のサブテンプレートとサブウィンドウの照合で、多数の近傍ウィンドウの距離下限値が計算できる。求めた$\mathbf{W}^{(u,v)}$の距離下限値が閾値θを上回れば、位置(u,v)におけるテンプレート$\mathbf{T}$とウィンドウ$\mathbf{W}$間の照合を省略できる。閾値θは、これまで求めたテンプレートとウィンドウの距離の最小値である。閾値の初期値θ_0をとりうる値の最大値に設定することによって、最も類似する領域を検出することができる。以上の処理により、不要な処理を適応的にスキップし、計算量の大幅な削減を実現できる。

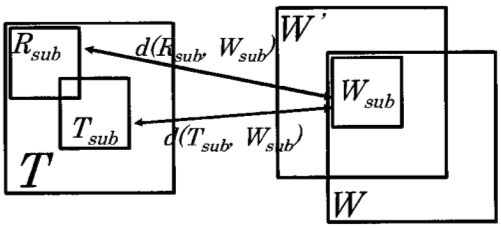

図 4: R_{sub}とW_{sub}の距離からT_{sub}とW_{sub}の距離を推定

4.3 距離下限値の計算

探索処理において、代表サブテンプレート$\mathbf{R}_{sub}$と入力画像における位置(m,n)におけるサブウィンドウ$\mathbf{W}_{sub}^{(m,n)}$の距離計算を行った場合、テンプレート$\mathbf{T}$と$\mathbf{W}_{sub}^{(m,n)}$を含むウィンドウ$\mathbf{W}^{(m-k,n-l)}$(ただし、$0 \leq k < M_x - m_x, 0 \leq l < M_y - m_y$)の距離下限値は次のようにして求めることができる。

前処理の段階において、サブテンプレート間距離$d(\mathbf{R}_{sub}, \mathbf{T}_{sub}^{(k,l)})$は計算済みであり既知である。$\mathbf{R}_{sub}$とサブウィンドウ$\mathbf{W}_{sub}^{(m,n)}$の距離$d(\mathbf{R}_{sub}, \mathbf{W}_{sub}^{(m,n)})$を計算した際、図5に示す三つの距離に対し三角不等式が成立し、$d(\mathbf{T}_{sub}^{(k,l)}, \mathbf{W}_{sub}^{(m,n)})$に対し次式を導くことができる。

$$d(\mathbf{T}_{sub}^{(k,l)}, \mathbf{W}_{sub}^{(m,n)}) \geq d(\mathbf{R}_{sub}, \mathbf{W}_{sub}^{(m,n)}) - d(\mathbf{R}_{sub}, \mathbf{T}_{sub}^{(k,l)}) \quad (2)$$

次に、SADは単調増加なので、テンプレートとウィンドウの距離は、サブテンプレートとサブウィンドウの距離より大きい。すなわち、以下の式(3)が成立する。

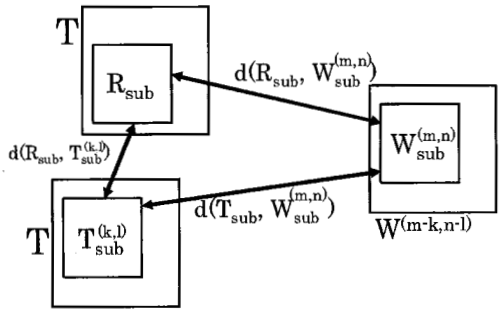

図 5: 三つの距離の関係

$$d(\mathbf{T}, \mathbf{W}^{(m-k,n-l)}) \geq d(\mathbf{T}^{(k,l)}_{sub}, \mathbf{W}^{(m,n)}_{sub}) \geq d(\mathbf{R}_{sub}, \mathbf{W}^{(m,n)}_{sub}) - d(\mathbf{R}_{sub}, \mathbf{T}^{(k,l)}_{sub}) \quad (3)$$

式(3)は$d(\mathbf{R}_{sub}, \mathbf{W}^{(m,n)}_{sub}) - d(\mathbf{R}_{sub}, \mathbf{T}^{(k,l)}_{sub})$がテンプレートTとウィンドウ$\mathbf{W}^{(m-k,n-l)}$の距離下限値であることを意味する。この距離下限値が、閾値θを超えた場合、ウィンドウ$\mathbf{W}^{(m-k,n-l)}$とテンプレートTの距離は閾値θより必ず大きくなり、位置$(m-k, n-l)$は求める位置にはなり得ない。そのため、その位置での処理を省略することができる。

4.4 代表サブテンプレートの位置について

AWSは探索処理の際、多数のウィンドウに対して距離下限値を設定しその値が閾値θを超えた場合、処理をそのウィンドウに対する処理を省略することで高速に出力を得る。また、AWSは前処理の段階でサブテンプレートの中から代表サブテンプレートを一つ選定する。**4.3**より、AWSは、選ばれた代表サブテンプレートとサブウィンドウの距離を利用して、ウィンドウの距離下限値を求める。そのため、どのサブテンプレートを代表サブテンプレートとして選択するかによって処理時間が異なる。しかし、AWSでは代表サブテンプレートは単に左上隅のサブテンプレート$\mathbf{T}^{(0,0)}_{sub}$を代表サブテンプレートとして選択しており、代表サブテンプレートの位置に関しては特に考慮されていない。

5 提案手法

本研究で提案する、代表サブテンプレートの位置を考慮した適応的ウィンドウスキップによる高速テンプレートマッチング法は、AWSにおいて代表サブテンプレートを選ぶ際、距離下限値の値が大きくなるような代表サブテンプレートを選ぶことで、処理の高速化を行う。具体的には、代表サブテンプレートとして、他のサブテンプレートとの距離の総和が最も少ないサブテンプレートを選ぶ。これにより、式(3)の$d(\mathbf{R}_{sub}, \mathbf{T}^{(k,l)}_{sub})$の値が平均的に小さくなる。その結果、ウィンドウに設定される距離下限値の値が大きくなる。AWSでは距離下限値が閾値θより大きければ照合を省略することができるので、距離下限値の値が大きければ省略できるウィンドウが多くなり、処理時間を短縮することができる。

5.1 代表サブテンプレートの決定方法

提案手法では、照合の走査処理を始める前処理として、代表サブテンプレートの位置と走査開始位置と走査方向を決定する。本手法では、代表サブテンプレートの位置を決定するパラメータとして、以下の式(4)で表されるサブテンプレート$\mathbf{T}^{(q,r)}_{sub}$と他の全てのサブテンプレートとの距離の総量$D^{(q,r)}_{total}$を利用する。

$$D^{(q,r)}_{total} = \sum_{s=0}^{M_x-m_x-1} \sum_{t=0}^{M_y-m_y-1} d(\mathbf{T}^{(q,r)}_{sub}, \mathbf{T}^{(s,t)}_{sub}) \quad (4)$$

$D^{(q,r)}_{total}$は、位置(q,r)のサブテンプレートと、他の全てのサブテンプレートとの距離の総和である。$D^{(q,r)}_{total}$を求めるにはサブテンプレートの総数$(M_x-m_x)\times(M_y-m_y)$回の距離計算を行う必要がある。提案手法では、前処理の段階でテンプレート内の全てのサブテンプレートに対して$D^{(q,r)}_{total}$を求める。$D^{(q,r)}_{total}$の値が、最も小さいサブテンプレート$\mathbf{T}^{(q,r)}_{sub}$を代表サブテンプレート$\mathbf{R}_{sub}$とする。ここで、$D^{(q,r)}_{total}$の値が最も小さいサブテンプレートとは、他のサブテンプレートとの距離の総和が最も少ないサブテンプレートであり、$d(\mathbf{R}_{sub}, \mathbf{T}^{k,l}_{sub})$の値が平均的に小さくなる。

5.2 提案手法の処理の流れ

提案手法もAWSと同じく、処理手順は大きく分けて前処理と探索処理に分かれている。前処理と探索処理は以下のような処理手順によって行われる。

< 前処理手順 >

(1) 入力画像データベース、テンプレートを読み込む。
(2) テンプレートからすべてのサブテンプレートを切り出す。
(3) 全てのサブテンプレートにおいて、他の全てのサブテンプレートとの距離を計算し、その総和を求める。
(4) 求めた距離の総和が最も少ないサブテンプレートを代表サブテンプレート$\mathbf{R}_{sub}$とする。
(5) $\mathbf{R}_{sub}$と全ての$\mathbf{T}^{(k,l)}_{sub}$の距離$d(\mathbf{R}_{sub}, \mathbf{T}^{(k,l)}_{sub})$を求める。
(6) 全てのウィンドウの距離下限値を0に初期化する。
(7) 閾値θを閾値の初期値θ_0に初期化する。

< 探索処理手順 >

(1) 探索していないウィンドウが存在するのであれば、それを切り出す。
(2) 切り出したウィンドウの距離下限値が閾値θより大きければ、照合を省略し、(1)へ。
(3) 代表サブテンプレートに対応するサブウィンドウを切り出す。

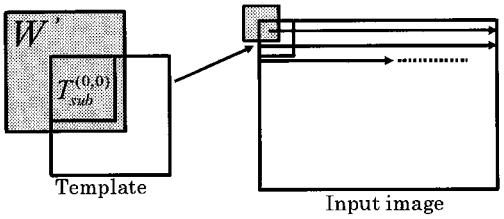

図 6: 代表テンプレートの位置と走査方向の関係

(4) 代表サブテンプレートとサブウィンドウの距離(部分距離)を計算する。
(5) 部分距離が閾値θより大きいとき、周囲のウィンドウの距離下限値を更新し、(1)へ。
(6) 部分距離が閾値θより小さいとき、テンプレートとウィンドウの残りの距離も計算し、テンプレートとウィンドウの距離(全体距離)を求める。
(7) 全体距離が閾値θより小さいとき、閾値θをこの全体距離の値に更新し、この位置を記録する。
(8) (1)へ戻る。

5.3 走査開始位置と走査方向の決定方法

代表テンプレートの位置と、ウィンドウを走査する順序は提案手法の効率を考える上で非常に重要である。例えば、左上隅のサブテンプレート$\mathbf{T}_{sub}^{(0,0)}$を代表サブテンプレートとして利用した場合、距離下限値を計算するウィンドウ$\mathbf{W}'$は照合したウィンドウの左上の方向になる。すなわち、図6のように左上から右下へ走査した場合には、距離下限値を計算するウィンドウ$\mathbf{W}'$は、既に走査し終わったウィンドウとなってしまうため、照合を省略した効果がなくなってしまう。そこで、提案手法では代表テンプレートのテンプレート内の位置に応じて、ウィンドウの走査の開始位置と走査方向を変更する。

5.3.1 走査開始位置の決定方法

提案手法において、探索処理における走査開始位置はテンプレートにおける代表サブテンプレートの位置(q,r)から決定する。具体的には、テンプレート内の代表サブテンプレートの位置が(q,r)であるとき、走査開始位置(a,b)は、以下の式(5)で表される。

$$\begin{aligned} a &= X - M_x - (X - M_x) \times (q/(M_x - m_x)) \\ b &= Y - M_y - (Y - M_y) \times (r/(M_y - m_y)) \end{aligned} \quad (5)$$

式(5)における走査開始位置(a,b)は、左上隅を原点としたときの、テンプレート内における代表サブテンプレートの位置の比$(M_x : q, M_y : r)$を右下隅を原点としたときの入力画像とウィンドウの比に置き換えたものである。図7のように、代表サブテンプレート$\mathbf{R}_{sub}$がテンプレート内部の左上に存在する場合、開始位置は入力画像中の右下の部分になる。ウィンドウの走査処理は求めた(a,b)から開始される。

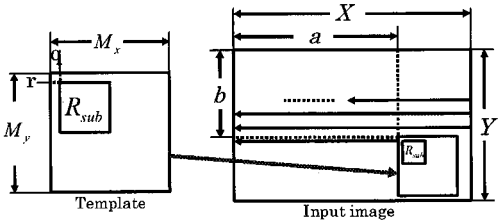

図 7: 走査開始位置の決定

5.3.2 走査方向の決定方法

テンプレートの走査方向に関しても、代表サブテンプレートの位置(q,r)によって決定される。具体的には、$(M_x - m_x)/2 \leq q < M_x - m_x$、$(M_y - m_y)/2 \leq r < M_y - m_y$、つまり走査開始位置が入力画像内の左上の領域にあるとき、図8(a)のように左上から右下への方向に走査する。$0 \leq q < (M_x - m_x)/2$、$(M_y - m_y)/2 \leq r < M_y - m_y$、つまり走査開始位置が入力画像内の右上の領域内にあるとき、図8(b)のように右上から左下への方向に走査する。$(M_x - m_x)/2 \leq q < M_x - m_x$、$0 \leq r < (M_y - m_y)/2$、つまり走査開始位置が入力画像内の左下の領域にあるとき、図8(c)のように左下から右上への方向に走査する。$0 \leq q < (M_x - m_x)/2$、$0 \leq r < (M_y - m_y)/2$、つまり走査開始位置が入力画像内の右したの領域にあるとき、図8(d)のように右下から左上への方向に走査する。それぞれの走査方向において、画像の隅まで走査し終わった場合、反対側の画像の隅に戻りそこから走査開始位置までの処理を行うことで、画像全体を走査する。これにより、効率的にウィンドウをスキップしながら走査処理を行うことができる。

5.4 前処理の計算量

提案手法は、代表サブテンプレートの位置を決定するため、前処理の段階でサブテンプレートをテンプレート上で走査するように全てのサブテンプレートに対して他の全てのサブテンプレート間距離を求める。その

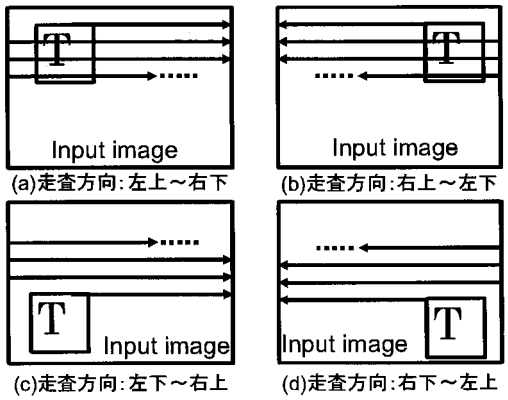

図 8: 走査方向の決定

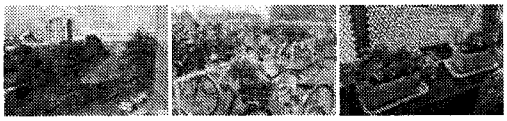

図 9: 実験に用いた入力画像の例

図 10: 実験に用いたテンプレートの例

ため、AWSに比べて前処理に多くの計算量が必要になる。この計算量は、テンプレートサイズとサブテンプレートサイズに応じて決定される。サブテンプレートサイズ(m_x, m_y)が小さい程、サブテンプレートの数が増えるので、AWSと比べて距離計算を行う回数が多くなり、より多くの計算処理が必要になる。従って、提案手法は前処理時間を無視することができる、画像データベースに対するテンプレートマッチングに対して有効である。

6 実験

提案手法の効果を確かめるために、テンプレートマッチング法(TM)、AWS、提案手法の各手法との処理実行時間の比較を行った。TMは、閾値θを利用した距離計算の省略を行わず、全てのウィンドウにおいて、テンプレートとウィンドウの距離計算を行う方法である。本実験において、AWSの代表サブテンプレートの位置は、文献[6]と同様に$\mathbf{T}_{sub}^{(0,0)}$とし、探索処理における走査開始位置はウィンドウの位置領域の右下隅$(X - M_x, Y - M_y)$、ウィンドウの走査方向は右下から左上への方向とした。

本実験で使用した入力画像は、384 × 256pixelの20種類の異なるカラー画像であり、画像の色表現にはRGBそれぞれ8bitの値を用いた。また、入力画像には10 %のガウスノイズを加えて使用した。実験に使用した入力画像の例を図9に、テンプレートの例を図10に示す。テンプレートサイズは80 × 80pixelとした。サブテンプレートサイズは、参考文献[6]より、最も良好な結果を得ることができるサイズとしてテンプレートサイズの85 %である、68 × 68pixelとした。実験で使用したPCのスペックを、表1 に示す。

本実験では、20枚の入力画像から成る画像データベースを作成し、提案手法、AWS、TMそれぞれの手法で処理を行い、画像データベースからテンプレートの位置を出力させた。テンプレートは画像データベースを構成する20枚の画像それぞれに用意し、計20種類のテンプレートを使用した。実験はそれぞれのテンプレートに対して行い、それぞれの手法に対して合計20回の探索を行った。全ての実験において画像データベースから画像を検索する順序は同じとした。なお、実験の結果、全ての手法において出力されたテンプレートの位置は同一であり、正しいテンプレートの位置を検出することができた。

表 1: 実験に使用したコンピュータのスペック

OS	Vine Linux 2.6
CPU	Intel Xeon 3.06GHz
Memory	2GB
Compiler	GNU C++(-O3オプション使用)

表 2: 探索処理時間の比較

	提案手法	AWS	TM
平均実行時間 (sec)	145.598	154.5315	260.201
最短実行時間 (sec)	16.98	34.54	236.55
最長実行時間 (sec)	359.83	386.28	270.4

表2より、提案手法が最も短い実行時間で処理を行えたことがわかる。本実験において、提案手法は平均してAWSの94.22 %の実行時間で処理を終えることができており、約6 %の処理時間の削減に成功した。20回の実験のうち、提案手法がAWSよりも高速に処理を行えたのは17回であり、全体の85 %であった。残りの3例に関しては、AWSが高速に処理を行うことができた。これらの例に関しては、**7**で考察する。実験結果より、平均的に提案手法の方が高速に処理を行うことができた。

また、実験において、AWSと提案手法における前処理にかかった平均時間はAWSで1.00(sec)、提案手法で4.79(sec)であった。従って、20枚程度の画像データベースからの検索において、前処理の時間を含めても、提案手法が高速であることがわかる。また、前処理の時間は入力画像の枚数に影響しないので、処理を行う画像データベースの規模が大きくなるほど影響は小さくなり、提案手法の有効性は増すと言える。

7 考察

7.1 提案手法の方が多くの実行時間が必要になる場合

本実験において行った実験では、ほとんどの場合(全体の85 %)において、提案手法の方が少ない実行時間で処理を終えることができた。しかし、残りの実験では提案手法よりもAWSが高速に処理を行えるという結果になった。これは、提案手法とAWSで選択された代表サブテンプレート間の距離の違いと、代表サブテンプレートの位置の違いが計算時間に影響を与えているためであると考えられる。それぞれの場合について、以下に詳しく検討する。

代表サブテンプレート間距離の違いが、計算時間影響を与えているかどうかを調べるため、**6**で行った実験

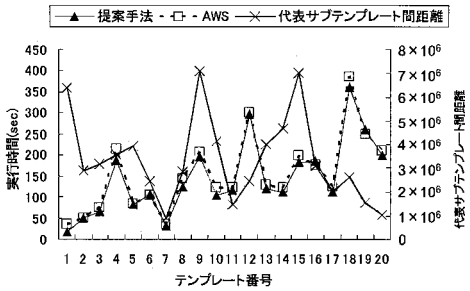

図 11: 実行時間と代表サブテンプレート間距離の関係

に対し、20枚の各テンプレートそれぞれを探索するのに要した、提案手法とAWSの実行時間および、提案手法で選択された代表サブテンプレートとAWSで選択された代表サブテンプレートの距離を図11に示す。図11より、AWSの方が高速に処理を終えている例(テンプレート番号7,16,19)では、代表サブテンプレート間距離が比較的小さい値になっていることがわかる。これは、提案手法とAWSでは、代表サブテンプレート同士の距離の差が大きいほど、AWSに比べて提案手法の処理削減効果が大きくなるためと考えられる。提案手法では、他のサブテンプレートとの距離の総和が最も少ないサブテンプレートを代表サブテンプレートとしている。それにより、サブテンプレート間距離が全体的に小さくなり、処理を削減できるウィンドウの数が増える。そのため、代表サブテンプレート同士の距離の差が大きいほど提案手法のAWSに比べての処理削減効果が大きくなる。

計算時間に影響を与えるもう1つの要因として、代表サブテンプレートの位置の違いによる、走査開始位置と走査方向の違いが考えられる。

例えば提案手法の走査開始位置が入力画像の左上隅の場合、AWSでの走査開始位置は右下隅であるため、ウィンドウを検索する順序が逆になる。このとき、テンプレートと非常に距離の小さいウィンドウが画像の右下隅の付近にあった場合、AWSでは、初期の探索でこのウィンドウとの距離計算を行い、非常に小さい距離値が得られる。このような小さな距離値が得られた場合、閾値θの値が小さな値に設定されるため以降の計算処理を高速に行うことができる。一方、左上隅から探索した提案手法は、最後の方で非常に距離の小さいウィンドウとの距離計算を行うため、それまでのウィンドウとの距離計算において、閾値θが大きいため、右下から開始するAWSよりも高速に処理することができない場合が生じる。

以上のように、代表サブテンプレート間の距離の違いと、位置の違いが処理時間に影響するため、提案手法がAWSよりも遅くなることがあるが、平均的には

表 3: 代表サブテンプレート間距離が大きい場合の結果

	提案手法	AWS
平均実行時間(sec)	1.582	2.632
最短実行時間(sec)	0.56	0.42
最長実行時間(sec)	5.63	7.66

AWSよりも高速に処理することができる。

走査開始位置と走査方向の差と、代表サブテンプレート間の距離の二つについて考察したが、これらの場合は画像1枚に対する処理に対してある程度偶発的に生じるものである。探索を行う入力画像内に閾値θよりも距離の小さなウィンドウが存在しない場合、距離下限値の値が大きくなる提案手法の方が速く処理を行うことができると考えられる。また、処理を行うデータベースの規模が大きくなるほど閾値θの更新が発生しない画像に対する処理が増えるため、提案手法の優位性が大きくなると考えられる。

7.2 閾値θの更新がない場合について

7.1では、提案手法がAWSよりも遅くなることがある要因として、代表サブテンプレート間の距離の違いと、位置の違いが処理時間に影響することが挙げられると述べた。そこで、これらの要因による処理時間の影響をできるだけ少なくした場合に、どの程度提案手法が有効であるかを調べる。代表サブテンプレート間の位置の違い、すなわち走査線開始位置と方向の違いによる影響を少なくするためには、閾値θの更新がないようにすればよい。すなわち閾値θの初期値を正解テンプレートの距離値に設定すれば、閾値θの更新が行わずに、**7.1**で述べた閾値θの更新までの処理の差による実行時間の差が現れないことになる。一方、代表サブテンプレート間の距離の違いを少なくするためには、**6**で行った実験のうち、代表サブテンプレート間の距離の大きいテンプレートを用いればよい。

以上のように、閾値θの初期値を正解テンプレートとの距離値に設定し、代表サブテンプレート間の距離の大きいテンプレートを用いた時の、提案手法とAWSの実行時間の違いを表3に示す。表3は**6**で行った実験において、代表サブテンプレート間距離の大きいテンプレートを用いて18枚の画像に対して処理を行ったときの、画像1枚に対するそれぞれの手法の平均実行時間、最短実行時間、最長実行時間を示している。

表3より、代表サブテンプレート間距離が大きいテンプレートでの実験結果では、平均実行時間は提案手法の方が短くなっており、AWSの60.1 %の実行時間で処理を終えている。なお、このテンプレート用いた実験では全ての場合において提案手法の方が高速に処理を行うことができた。この結果から、閾値θの更新が発生せず、提案手法とAWSの代表サブテンプレート間距

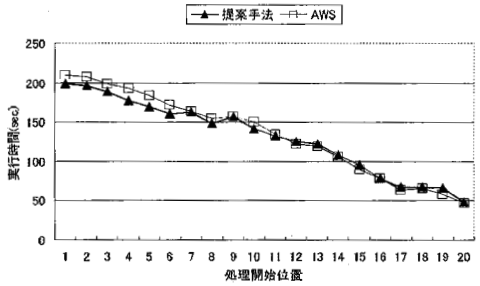

図 12: 処理開始位置を変えて実行した結果

離が大きい場合での、提案手法の優位性が確認できる。

この実験で確認できた提案手法の優位性は、探索処理において閾値θの更新が発生しない処理全てに対して有効なものである。処理を行うデータベースの規模が大きいほど処理を行う画像の枚数が多くなるので提案手法の優位性は、大きくなるものと考えられる。

7.3 実行時間と処理開始位置の関係について

画像データベースに対して探索処理を行う場合、画像データベースにおける、画像を検索する順序は実行時間に影響を与えるため、重要である。正解のテンプレート位置を探索した場合、閾値θは最も低い値となるため、その後の探索処理は全て適応的にスキップされる。そのため、探索処理中の初期で、正解のテンプレート位置を含む画像を探索した場合と後期で探索した場合では、処理に要する時間が大きく異なると考えられる。

そこで、同一のテンプレートを利用して、画像データベースの処理開始位置を変更しながら検索を行った。なお、この実験で利用したテンプレート位置が存在するデータベース上の画像の位置は20番目であり、走査は画像番号に対して降順に行い、1番目の画像を処理し終わったら20番目の画像に戻り、降順に処理をしていない画像を処理していくものとする。実験の結果を図12に示す。

図12より、画像の検索順序によって実行時間に差違が生じており、正解のテンプレート位置を含む画像を処理全体の後期で行うような場合ほど、実行時間が大きくなっていることがわかる。これは、提案手法やAWSでは、正解のテンプレート位置を含む画像を何枚目に処理するかが実行時間に大きな影響を与え、初期に処理できるほど実行時間は短くなることを表している。

8 むすび

本研究では、テンプレートマッチングの高速化手法であるAWSにおいて、代表サブテンプレートの選択方法についての検討を行い、代表サブテンプレートの位置を考慮した高速テンプレートマッチング法を提案した。本研究で提案した代表サブテンプレートの位置を考慮した適応的ウィンドウスキップによる高速テンプレートマッチング法は、AWSにおいて代表サブテンプレートを選定する際、他のサブテンプレートとの距離の総和が最も少ないサブテンプレートを選び設定される距離下限値の値を大きくし、処理を省略できるウィンドウを増やすことで、より高速に処理を行うことができる。

実験結果より、提案手法は多数の画像を含む画像データベースからのテンプレート位置の探索において、テンプレートマッチング法と同じ精度を保証しながらAWSよりも高速探索することができた。提案手法は閾値θの更新がない、最大限に処理をスキップすることができる場合の処理ではAWSよりもさらに高速に処理を行え、そのような処理が多くなる大規模画像データベースに対して有効であることがわかった。

また、画像データベースに対して処理を行う場合、画像データベース上の処理の開始位置によって実行時間に明確な差違が生じることがわかった。提案手法やAWSは、正解のテンプレート位置の探索が処理の初期であるほど高速に処理を行うことができる。このことから、新たな高速化手法として、対象データベースにおける処理開始位置を自動的に選定する手法などが考えられる。これは今後の課題とする。具体的には、処理を階層化し、まず大まかに探索処理を行い、その結果を受けて探索を行う順番を変更した後、全体の探索処理を行う手法などが考えられる。

参考文献

[1] 尾上守男(編), 画像処理ハンドブック, 昭晃堂, 1987.

[2] D.I. Barnea and H.F. Silverman, "A class of algorithms for fast digital image registration," IEEE Trans.Comput., vol.C-21, no.2, pp.179-186, 1972.

[3] G.J. Vanderbrug and A. Rosenfeld, "Two-stage template matching," IEEE Trans. Comput., vol.C-26, no.4, pp.384-393, 1977.

[4] 村瀬 洋, V.V. Vinod, "局所色情報を用いた高速物体探索 — アクティブ探索法," 信学論(D-II), vol.J81-D-II,no.9,pp.2035-2042, Sept. 1998.

[5] 川西隆仁, 黒住隆行, 柏野邦夫, 高木茂, "サブテンプレート間距離を用いた適応的スキップによる高速テンプレートマッチング法 — スキッピングテンプレートマッチング法," 画像の認識・理解シンポジウム(MIRU2004), Vol.2, pp.19-23,2004.

[6] 川西隆仁, 久野和樹, 木村昭吾, 黒住隆行, 柏野邦夫, 高木茂, "サブテンプレート間距離を用いた適応的ウィンドウスキップによる高速テンプレートマッチング法," 電子情報通信学会論文誌 D-II Vol. J88-D-II, No.8, pp.1389-1397,2005.