2004－CVIM－146　(19)
2004／11／12

骨格線を利用したオブジェクト検索手法

中山　仁†　渡辺　俊典†　古賀　久志†

† 電気通信大学大学院情報システム学研究科
E-mail: †{zin,watanabe,koga}@sd.is.uec.ac.jp

概 要　オブジェクトより得られる骨格線を利用したオブジェクト検索手法を提案する．本手法では，階層化されたオブジェクトの骨格を木構造で表現する．木のノードに骨格のトポロジカルな特徴を持たせて，木のマッチングでオブジェクト間の類似度を求め，オブジェクト検索を実現する．トポロジカルな特徴は，オブジェクト形状が多少変形した場合でも変化しないため，ロバストなオブジェクト検索が行える．

キーワード　骨格，オブジェクト検索, 木のマッチング

Object retrieval using skeleton lines

Hitoshi NAKAYAMA†, Toshinori WATANABE†, and Hisashi KOGA†

† Graduate School of Infomation Systems, University of Electro-Communications
E-mail: †{zin,watanabe,koga}@sd.is.uec.ac.jp

Abstract　We propose a method to search objects using skeletons. In this method, the skeletons of objects are expressed by trees. Then the object retrival is realized by associating tree nodes with topological features and calculating the similarities between objects as tree-edit distances. Because topological features do not change against shape transformation to some extent, our object retrival metood has robustness.

Key words　skeleton, object retrieval, tree matching

1. はじめに

画像のオブジェクトを認識することは，コンピュータビジョンにおいて基本的な問題の１つである．オブジェクトの形状には，面積，重心，周囲長などの特徴を多く含む．そのためオブジェクトを認識する上で，オブジェクトの形状特徴は最も重要な特徴となり，大きな影響を与える．実際，産業応用画像処理（製品検査など）や医学，生物学における顕微鏡画像処理においても形状特徴を利用したものがオブジェクトの認識に用いられ，実用化されている．

オブジェクトの形状だけを用いて，認識する手法の一つに輪郭線と骨格線を用いるものがある．本研究では骨格を利用した手法について論じる．骨格は，1) 形状を線で表現することができる，2) 形状と同じ位相と幾何学的構造を持っている，3) 形状の復元が可能である，という特徴を持っており，形状を少ない情報で表現できるため，よい表現方法であると言える．骨格を用いる既存手法として，骨格の接続関係を木構造で表現し，木のマッチングによりコストを求め，オブジェクト間の類似度とするものが知られている [1]．この手法は，オブジェクトの類似度を木の類似度にマッピングすることにより，高速に認識が行え，形状変化に対して，ロバストなオブジェクト認識が行える．しかし，図 1 のようにどのノードを木の根にするかによって，木の構造が変化するため，形状が類似したオブジェクト間でも近い類似度が得られないという不安定さが指摘されている [2]．

本研究では，新たに Borgefors ら [3] による階層的

骨格を利用し，それを木構造で表現して，木のマッチングでオブジェクト間の類似度を求めるオブジェクト検索法を提案する．この階層化骨格はオブジェクトがある程度変形しても，上位階層の骨格は変形せず，下位階層の骨格のみ変化するという性質があるので，これを利用してロバストな検索手法を実現することを狙う．また，上位階層を根として木を作ることにより，木の根の選択方法が従来手法と異なり，自然に一意に定まるという特徴がある．

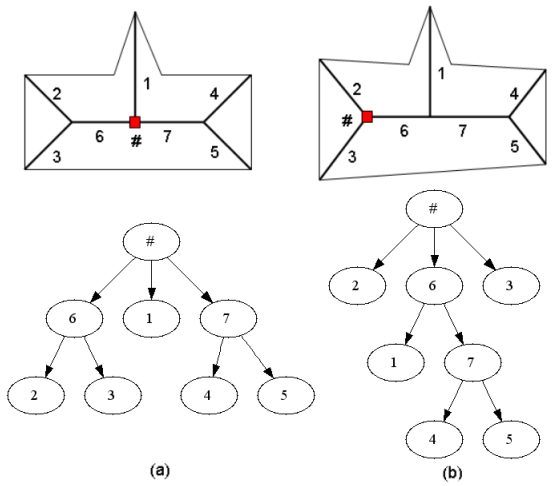

図 1　既存手法の木構造表現 (♯:根)

2. 骨　　格

骨格は 2 値画像の距離変換によって求められる．距離変換とは，図形内の各画素について，背景から各画素までの距離を画素の値に変換する．距離変換された図形から 4 近傍（または 8 近傍）での値が最大値をとる画素の集合を求める操作を「骨格化」という．つまり，図形領域を S, 背景領域を $\bar{S}$ とし，d を距離関数とする．S の距離変換図形内の画素 (i,j) で最大値をとる画素の集合 S^* は，

$$S^* = \{(i,j) | d((i,j),\bar{S}) \geq d((u,v),\bar{S})\} \qquad (1)$$

であり，S^* が S の骨格となる．(u,v) は (i,j) の 4 近傍（または 8 近傍）の画素である．骨格化を行うことにより，図形を線で表現できる．また，骨格の各画素は，距離の情報を保持しているため，骨格は図形の情報を保持し，情報圧縮の手段としても用いられる．骨格の持つ特性として，

（1） 図形を線で表現する

（2） 形状と同じ位相と幾何学的構造を持つ

（3） 骨格から形状の復元が可能である

の 3 つがある．

3. 骨格の階層化

本章では，4 章で提案する階層化された骨格の木構造を生成するために行う，骨格の階層化について説明する．骨格の階層化には，文献 [3] で提案されている手法を用いる．

3.1 骨格のマルチスケール表現

骨格のマルチスケール表現を行うために，ピラミッド法の AND ピラミッドを用いる．AND ピラミッドは，$2^i \times 2^i$ の大きさの画像を 2×2 のブロックに分割し，ブロック内の画素がすべて黒画素であれば黒画素とし，それ以外であれば白画素として，解像度の低い $2^{i-1} \times 2^{i-1}$ 画像を作成する．$2^n \times 2^n$ サイズの入力画像を最下位階層の画像 L_n とすると，最下位階層の画像 L_n から最上位階層 L_3 まで解像度を減少させ，$(n-2)$ 階層のピラミッドを構築する．そして，AND-ピラミッドの各階層の画像ごとに骨格化アルゴリズムを用いて骨格化を行い，骨格のマルチスケール表現を行う (図 2)．本研究では，骨格化アルゴリズムとして [4] を用いる．距離関数は，市街区距離を使う．

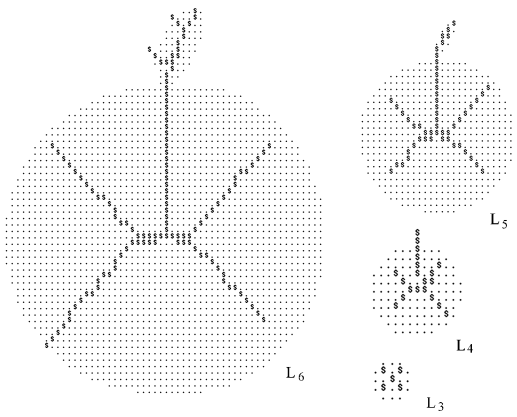

図 2　骨格のマルチスケール表現

3.2 トップダウン処理

上位の階層と下位の階層間の画素を関連付けるトップダウン処理により，最上位階層 L_3 から最下位下層 L_n までの各々の骨格画素に識別可能なラベルと付けていく．つまり，上位の階層 L_i の骨格画素は，下位の階層 L_{i+1} の 2×2 のブロック内の骨格

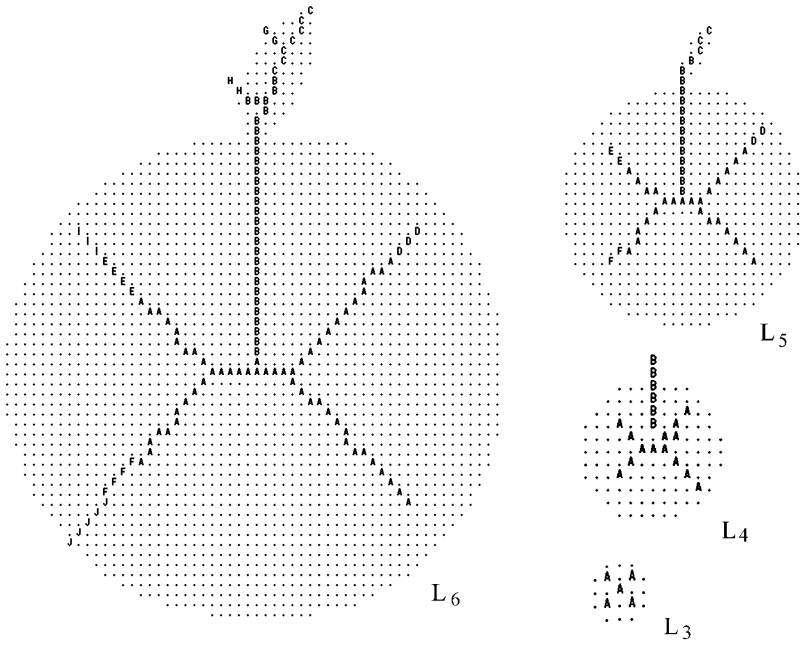

図 3　骨格の階層化

画素に対応するため，2×2 のブロック内に存在する骨格画素には上位の骨格画素のラベルと同じラベル付ける．対応しない骨格画素は L_{i+1} で発生した骨格であり，新たなラベルが付ける．このことにより，各々の骨格画素の発生する階層が分かり，骨格の階層化が行われる．図 2 に対して，骨格の階層化を行い，骨格を 1 画素幅に整えた結果を図 3 で示す．

4．階層的骨格の木構造表現

本研究では，階層的骨格の木構造表現をオブジェクト認識に適用することを提案する．木構造に表現することによって，オブジェクトの類似度を木の類似度にマッピングし， 形状変動に対して，ロバストなオブジェクト認識が行える．

4.1　木構造の生成

木構造の生成は，すべての階層で行う．つまり最上位階層 L_n とした時に $(n-2)$ 個の木を生成する．各階層における木構造の表現方法は，最上位階層 L_3 で現れるラベルを根として，根に隣接しているラベルを子とする．その子を親として，隣接しているラベルを子とする．この処理をすべてのラベルが木構造に記述されるまで繰り返すことで木構造を生成する．例えば，図 3 の階層 L_5 での木構造の記述

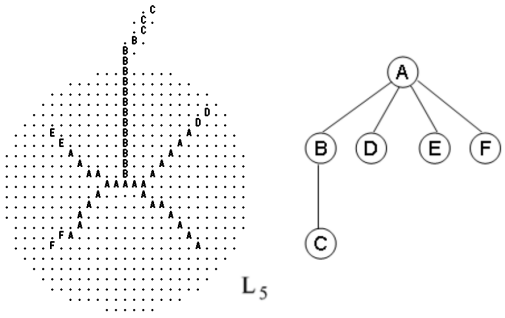

図 4　木構造の生成例

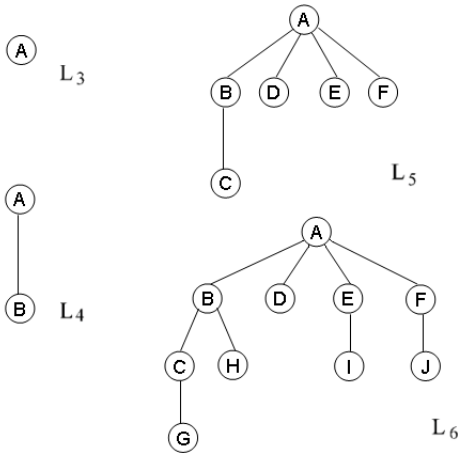

図 5　木構造の生成

は，図 4 のように，最上位階層 L_3 で現れるラベル A を根として，A に隣接している B,D,E,F を子として木に記述する．次に B を親として，B に隣接している C を子として木を記述する．これによりすべてのラベルが木に記述されたため，これが L_5 の木となる．また，図 3 のすべての階層で木の生成を行った結果を図 5 で示す．

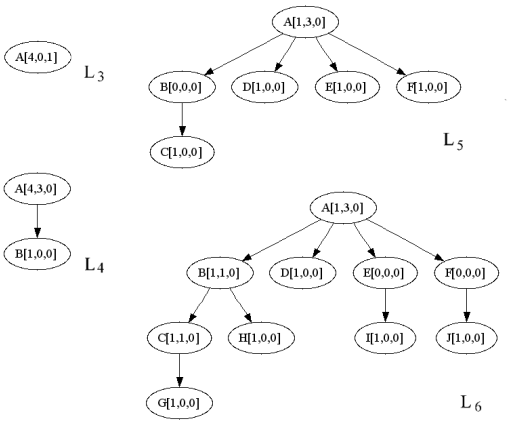

図 6　ノードの挿入

4.2　木のノード

木のノードは骨格のラベルの表す領域に対応する．骨格の特徴量としてオブジェクト形状が多少変形した場合でも変化しないトポロジカルな特徴量を持たせる．具体的には，木のノードに対する領域に含まれる骨格画素から端点数，分岐点数，交差点数を抽出し，特徴量とする．

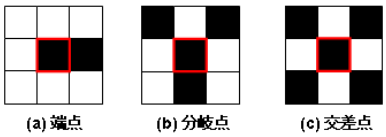

図 7　骨格の特徴

- 端点
  線の端の点 (図 7(a))
- 分岐点
  線がＹ字状に分岐する点 (図 7(b))
- 交差点
  2 本の線が十字状に交わる点 (図 7(c))

木のノードに特徴量を与えた例を図 6 で示す．

4.3　マッチングアルゴリズム

木のマッチングには，Oommen ら [5] で提案さらている木の DP マッチングを用いて，マッチングを行い，木のノードの間の類似度を求める．木のマッチングにおけるノードの削除，挿入，置換コストは，分岐点と交差点の必要な骨格上の線の本数及び，端点数に注目して以下のように決定した．分岐点と交差点の追加・削除に必要な骨格上の線の本数は，端点数に比べて重要であるため，重み w をかける．以降では，端点，分岐点，交差点の各ノード数を $\sharp end, \sharp branch, \sharp cross$ とおく．x, y をノードの識別子とする．

- ノード $x(\sharp end, \sharp brach, \sharp cross)$ の削除コスト

骨格上において，分岐点は 1 本，交差点は 2 本の線 (図 8 の点線) が削除されることによって，消滅する．そのため，削除されるコスト c は，

$$c = (\sharp branch + \sharp cross \times 2) \times w + \sharp end \qquad (2)$$

とする．

- ノード $x(\sharp end, \sharp brach, \sharp cross)$ の挿入コスト

骨格上において，分岐点は 1 本，交差点は 2 本の線 (図 8 の点線) が追加されることによって，発生する．そのため，挿入されるコスト c は

$$c = (\sharp branch + \sharp cross \times 2) \times w + \sharp end \qquad (3)$$

とする．

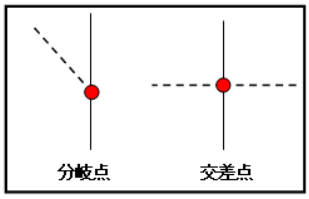

図 8　分岐点と交差点

- ノードの $x(\sharp end_1, \sharp branch_1, \sharp cross_1)$ を $y(\sharp end_2, \sharp branch_2, \sharp cross_2)$ に 置換するコスト

置換によって骨格上で挿入，削除される線の本数 $line$ は，

$$line = |\sharp branch_2 - \sharp branch_1 + (\sharp cross_2 - \sharp cross_1) \times 2| \qquad (4)$$

である．これだけでは図 9 のような 2 つの分岐点と交差点を区別できない．両者は構造が異なるが，

置換しようとすると，$|\sharp branch_2 - \sharp branch_1| \neq 0$，$|\sharp cross_2 - \sharp cross_1| \neq 0$ にもかかわらず，式 (4) の $line$ が 0 となる．そこで式 (4) の $line$ が 0 の場合は，構造の違いをコストに特別に反映させるようにする．分岐点数:2 と交差点数:1 の構造の違いは，分岐点数:2 の中心の 1 本による違いであるため，1 とする．よって，構造の違いとして与えるコストは，

$$line = |\sharp cross_2 - \sharp cross_1| \quad (5)$$

とする．最終的に置換コスト c は

$$c = line \times w + |\sharp end_2 - \sharp end_1| \quad (6)$$

とする．

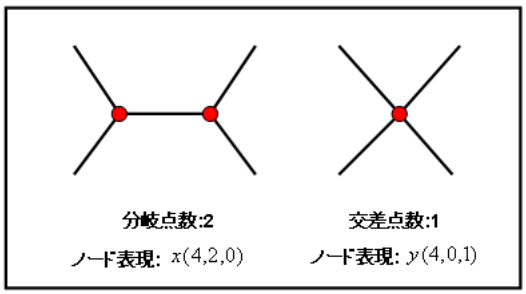

図 9　分岐点数と交差点数の関係

上記の削除，挿入，置換コストの合計が，階層 L_i でのコスト $cost_{L_i}$ となる．すべての階層で木のマッチングを行い，それぞれの階層で得られた木のマッチングコストに重み g_i をかけて，足した合計をオブジェクトの類似度として定義する．

$$cost = \sum_{i=3}^{n} (g_i * cost_{L_i}) \quad (7)$$

g_i は，重みであり，上位の階層ほど重要であるため，$g_n < g_{n-1} < \cdots < g_4 < g_3$ を満たす重みを与える．

5. 実験結果

本手法の有効性を検証する為に，オブジェクトのシルエット画像を用いて，それぞれのオブジェクト間の類似度を求めた実験結果を示す．テスト画像としては，図 10 で示す，64×64 のサイズの 8 個のシルエット画像で実験を行った．また，分岐点と交差点の必要な骨格上の線の本数に与える重み定数は，$w = 2$ とし，それぞれの階層で得られたコストを足し合わせて，オブジェクト間のコスト（類似度）とするときの重み定数は，$g_3 = 4, g_4 = 3, g_5 = 2, g_6 = 1$ とした．

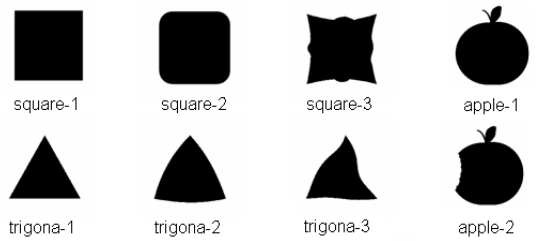

図 10　実験画像

実験結果は以下の表で示す．この実験結果より，三角形や四角形などの形状が変形したオブジェクトであっても，同様の形状を持つオブジェクトには他のオブジェクトに比べて，近い類似度が得られた．これにより，オブジェクトの形状が多少変形した場合でも，ロバストな検索を行えることを示した．

6. むすび

本研究では，階層化されたオブジェクトの骨格を木構造で表現し，木のマッチングでオブジェクト間の類似度を求め，オブジェクト検索する手法を提案した．そして，実験によってオブジェクトの形状の変形した場合でも，ロバストなオブジェクト検索を行えることを示した．

今後は，コストの計算の方法や木のマッチング操作方法にまだ問題があるため，最適化を進める．

文献

[1] K.Siddiqi, A.Shokoufandeh,S.Dickison,and S.Zucker, "Shock Graphs and Shape Matching" , Int'l J.Computer Vision,vol35,no.1,pp.13-32,1999.

[2] T.B.Sebastian, P.N.Klein,and B.B.Kimia, "Recognition of Shapes by Editing Their Shock Graphs" , IEEE Trans.Pattern Analysis and Machine Intelligence,vol.26,no.5,pp.550-571,2004.

[3] G.Borgefors, G.Ramella and G.Sanniti di Baja , "Hierarchical Decomposition of Multiscale Skeletons ",IEEE Trans.Pattern Analysis and Machine Intelligence,vol.23,pp.1296-1312,2001.

[4] G.Sanniti di Baja and E.Thiel , "Skeletonization algorithm running on path-based distance maps ",Image and Vision Computing,vol.14,pp.47-57,1996.

[5] B.J.Oommen and R.K.S.Loke ,"On the Pattern Recognition of Noisy Subsequence Trees" ,IEEE Trans.Pattern Analysis and Machine Intelligence, vol.23, no.9, pp.929-946,2001.

表 1　階層 L_3の木のマッチングコスト

	square-1	square-2	square-3	trigona-1	trigona-2	trigona-3	apple-1	apple-2
square-1	0	0	0	3	3	3	0	0
square-2	0	0	0	3	3	3	0	0
square-3	0	0	0	3	3	3	0	0
trigona-1	3	3	3	0	0	0	3	3
trigona-2	3	3	3	0	0	0	3	3
trigona-3	3	3	3	0	0	0	3	3
apple-1	0	0	0	3	3	3	0	0
apple-2	0	0	0	3	3	3	0	0

表 2　階層 L_4の木のマッチングコスト

	square-1	square-2	square-3	trigona-1	trigona-2	trigona-3	apple-1	apple-2
square-1	0	0	2	7	5	7	3	3
square-2	0	0	2	7	5	7	3	3
square-3	2	2	0	5	3	5	3	3
trigona-1	7	7	5	0	2	2	8	8
trigona-2	5	5	3	2	0	0	6	6
trigona-3	7	7	5	2	0	0	8	8
apple-1	3	3	3	8	6	8	0	2
apple-2	3	3	3	8	6	8	2	0

表 3　階層 L_5の木のマッチングコスト

	square-1	square-2	square-3	trigona-1	trigona-2	trigona-3	apple-1	apple-2
square-1	0	2	4	9	7	8	9	9
square-2	2	0	4	7	5	6	7	7
square-3	4	4	0	5	3	4	5	5
trigona-1	9	7	5	0	2	3	6	6
trigona-2	7	5	3	2	0	2	6	6
trigona-3	8	6	4	3	2	0	3	3
apple-1	9	7	5	6	6	3	0	2
apple-2	9	7	5	6	6	3	2	0

表 4　階層 L_6の木のマッチングコスト

	square-1	square-2	square-3	trigona-1	trigona-2	trigona-3	apple-1	apple-2
square-1	0	4	8	9	8	8	15	15
square-2	4	0	4	5	4	4	11	11
square-3	8	4	0	3	4	4	9	9
trigona-1	9	5	3	0	1	3	12	12
trigona-2	8	4	4	1	0	2	11	11
trigona-3	8	4	4	3	2	0	9	9
apple-1	15	11	9	12	11	9	0	2
apple-2	15	11	9	12	11	9	2	0

表 5　オブジェクト間の類似度

	square-1	square-2	square-3	trigona-1	trigona-2	trigona-3	apple-1	apple-2
square-1	0	8	22	60	49	57	42	42
square-2	8	0	18	52	41	49	34	34
square-3	22	18	0	40	31	39	28	28
trigona-1	60	52	40	0	11	15	60	60
trigona-2	49	41	31	11	0	6	53	53
trigona-3	57	49	39	15	6	0	51	51
apple-1	42	34	28	60	53	51	0	12
apple-2	42	34	28	60	53	51	12	0