2値画像処理Ⅱ-特徴抽出と符号化-(3.3)

- 膨張 (expansion): 2値画像を構成する連結成分の輪郭画素を1層分増やすことで、連結成分を外側に厚く(大きく)する処理。
- 収縮 (contraction): 2値画像を構成する連結成分の輪郭画素を1層分取り除くことで、連結画素を内側に薄く(小さく)する処理。
- 元画像を $f(i,j)$、変換後の画像を $g_t(i,j)$ とすると、上記処理(8近傍処理の場合)は以下のようになる。

膨張処理 $g_t(i,j)=\begin{cases}1, & f(i+k,j+l)=1\\ 0, & その他\end{cases}$ 収縮処理 $g_t(i,j)=\begin{cases}0, & f(i+k,j+l)=0\\ 1, & その他\end{cases}$

但し, $-1\le k,l\le 1$ で $f(i+k,j+l)$ は $f(i,j)$ 自身と周囲 8近傍画素。

k か l のどちらかが 0の場合は周囲 4近傍画素となる。

- p.38最終行の「膨張した画像」は「収縮した画像」の誤り。
- 図3.11に膨張と収縮処理の適用例がある。
- 図3.11を参考にして、次の元画像を膨張後収縮処理しなさい。
- 次に、同じ元画像を収縮後膨張処理しなさい。
- 但し、処理は4近傍ではなく、8近傍行うものとする。
- また、各処理は1回のみとする。
- なお、画像の周囲は0画素で囲まれているものとする。つまり、実際の画素はないが仮想的に0画素があるものとして近傍を考える。

- 元画像

1	1	1	1	1	1	1	
1	1	1	1	1	1	1	
1	1	1	1	1	1	1	
1	1	1		1	1	1	1
1	1	1	1	1	1	1	
1	1	1		1	1	1	
1	1	1		1	1	1	

1	1	1
1		1
1	1	1

- 元画像を膨張処理した画像

- 元画像を膨張処理した後、収縮処理した画像

- 元画像を収縮処理した画像

- 元画像を収縮処理した後、膨張処理した画像

• 膨張後収縮の解答

1	1	1	1	1	1	1					
1	1	1	1	1	1	1					
1	1	1	1	1	1	1			1	1	1
1	1	1		1	1	1	1		1		1
1	1	1	1	1	1	1			1	1	1
1	1	1		1	1	1					
1	1	1		1	1	1					

元画像

1	1	1	1	1	1	1	1	1					
1	1	1	1	1	1	1	1	1					
1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1					
1	1	1	1	1	1	1	1	1					

膨張後の画像

1	1	1	1	1	1	1					
1	1	1	1	1	1	1					
1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1					
1	1	1	1	1	1	1					

膨張後収縮の画像

- 収縮後膨張の解答

1	1	1	1	1	1	1	
1	1	1	1	1	1	1	
1	1	1	1	1	1	1	
1	1	1		1	1	1	1
1	1	1	1	1	1	1	
1	1	1		1	1	1	
1	1	1		1	1	1	

1	1	1
1		1
1	1	1

元画像

1	1	1	1	1
1				1
1				1
1				1
1				1

収縮後の画像

1	1	1	1	1	1	1
1	1	1	1	1	1	1
1	1	1	1	1	1	1
1	1	1		1	1	1
1	1	1		1	1	1
1	1	1		1	1	1
1	1	1		1	1	1

収縮後膨張の画像

- 膨張と収縮処理には以下の性質がある。

- 膨張後収縮した画像と収縮後膨張した画像は異なる。
- 膨張後収縮した場合:

  ① 小さな孔(穴)は消滅する。(1画素とは限らない)

  ② 小さな溝(凹み)は消滅する。(上記同様、1画素とは限らない。)
- 収縮後膨張した場合:

  ① 小さな髭(飛び出している領域)は消滅する。(上記同様、1画素とは限らない。)

  ② 小さな領域(近傍が全て1の画素を持たない領域)は消滅する。

距離変換(distance transformation)

- 画像中にある対象物体内の各画素値をその画素から背景(対象物体以外の領域)までの最短距離で置換する処理を距離変換という。
- 変換により得られる画像を距離変換画像という。
- 対象物体内部の画素ほど大きな値に変換される。
- 距離計算は4近傍と8近傍で異なる。

距離変換アルゴリズム

ア) 初期化：

元画像 $f(i, j)$ を距離変換初期化画像 $d'(i, j)$ に変換する。

$$d'(i, j) = \begin{cases} \infty, & f(i, j) = 1 \\ 0, & f(i, j) = 0 \end{cases}$$

イ) ラスタ走査：

左上から右下へのラスタ走査で、$d''(i, j)$ に変換する。

$$d''(i, j) = \min[d'(i, j), d''(i-1, j)+1, d''(i, j-1)+1] \quad (4\text{近傍})$$

$$\begin{aligned} d''(i, j) = \min[&d'(i, j), d''(i-1, j)+1, d''(i, j-1)+1, \\ &d''(i-1, j-1)+1, d''(i+1, j-1)+1] \quad (8\text{近傍}) \end{aligned}$$

ウ) 逆ラスタ走査：

右下から左上への逆ラスタ走査で、$d(i, j)$ に変換する。

$$d(i, j) = \min[d''(i, j), d(i+1, j)+1, d(i, j+1)+1] \quad (4\text{近傍})$$

$$\begin{aligned} d(i, j) = \min[&d''(i, j), d(i+1, j)+1, d(i, j+1)+1, \\ &d(i+1, j+1)+1, d(i-1, j+1)+1] \quad (8\text{近傍}) \end{aligned}$$

注) p.40中ほどの「ステップ (*b*)」は「ステップ (イ)」の誤り。

4近傍処理

0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	0	0	0	0	0	0
0	1	1	1	1	1	1	1	0	0	0	0	0	0
0	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	0	0	0	0	0	0
0	1	1	1	1	1	1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0

2値画像

0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	∞	∞	∞	∞	∞	∞	∞	0	0	0	0	0	0
0	∞	∞	∞	∞	∞	∞	∞	0	0	0	0	0	0
0	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	0
0	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	0
0	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	0
0	∞	∞	∞	∞	∞	∞	∞	0	0	0	0	0	0
0	∞	∞	∞	∞	∞	∞	∞	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0

初期化後

0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	0	0	0	0	0	0
0	1	2	2	2	2	2	2	0	0	0	0	0	0
0	1	2	3	3	3	3	3	1	1	1	1	1	0
0	1	2	3	4	4	4	4	2	2	2	2	2	0
0	1	2	3	4	5	5	5	3	3	3	3	3	0
0	1	2	3	4	5	6	6	0	0	0	0	0	0
0	1	2	3	4	5	6	7	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0

ラスター変換後

0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	0	0	0	0	0	0
0	1	2	2	2	2	2	1	0	0	0	0	0	0
0	1	2	3	3	3	3	2	1	1	1	1	1	0
0	1	2	3	4	4	4	3	2	2	2	2	1	0
0	1	2	3	3	3	3	2	1	1	1	1	1	0
0	1	2	2	2	2	2	1	0	0	0	0	0	0
0	1	1	1	1	1	1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0

逆ラスター変換後

8近傍処理

0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	0	0	0	0	0	0
0	1	1	1	1	1	1	1	0	0	0	0	0	0
0	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	0	0	0	0	0	0
0	1	1	1	1	1	1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0

2値画像

0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	∞	∞	∞	∞	∞	∞	∞	0	0	0	0	0	0
0	∞	∞	∞	∞	∞	∞	∞	0	0	0	0	0	0
0	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	0
0	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	0
0	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	0
0	∞	∞	∞	∞	∞	∞	∞	0	0	0	0	0	0
0	∞	∞	∞	∞	∞	∞	∞	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0

初期化後

0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	0	0	0	0	0	0
0	1	2	2	2	2	2	1	0	0	0	0	0	0
0	1	2	3	3	3	2	1	1	1	1	1	1	0
0	1	2	3	4	3	2	2	2	2	2	2	1	0
0	1	2	3	4	3	3	3	3	3	3	2	1	0
0	1	2	3	4	4	4	4	0	0	0	0	0	0
0	1	2	3	4	5	5	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0

ラスター変換後

0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	0	0	0	0	0	0
0	1	2	2	2	2	2	1	0	0	0	0	0	0
0	1	2	3	3	3	2	1	1	1	1	1	1	0
0	1	2	3	4	3	2	2	2	2	2	2	1	0
0	1	2	3	3	3	2	1	1	1	1	1	1	0
0	1	2	2	2	2	2	1	0	0	0	0	0	0
0	1	1	1	1	1	1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0

逆ラスター変換後

骨格抽出(skeleton extraction)

- 距離変換画像で極大値を持つ画素の集合を対象物体の骨格(skeleton)といい、この集合を取り出す処理を骨格抽出(skeleton extraction)という。
- p.40の”skelton”は”skeleton”の誤り(“l”の後に”e”が必要)。
- 距離変換画像 $d(i, j)$ に対して、次の条件を満足する画素が骨格を構成する。

$$d(i, j) \geq \max[d(i-1, j), d(i+1, j), d(i, j-1), d(i, j+1)]$$

(4近傍)

$$d(i, j) \geq \max[d(i-1, j), d(i+1, j), d(i, j-1), d(i, j+1), \\ d(i-1, j-1), d(i+1, j-1), d(i-1, j+1), d(i+1, j+1)]$$

(8近傍)

- p.41図3.13に4近傍の処理結果がある。下記図で8近傍処理による骨格画素に〇を付けなさい。
- 注) 図3.13は4近傍による距離変換画像であり、8近傍処理の距離変換画像は異なる。

								1	1
1	1	1	1	1					
1	2	2	2	2	1	1	1		
1	2	3	3	3	2	2	2	1	1
1	2	3	3	2	1	1	2	2	1
1	2	2	2	1			1	1	1
1	1	1	1	1					

- 8近傍処理による骨格抽出(解答)

								①	①
1	1	1	1	1					
1	2	2	2	2	1	1	1		
1	2	③	③	③	2	②	②	1	1
1	2	③	③	2	1	1	②	②	1
1	2	2	2	1			1	1	1
1	1	1	1	1					

骨格処理の性質

- 4近傍処理の結果と8近傍処理の結果は異なる。
- 一連結成分の骨格が連結しているとは限らない。(骨格が切れることもある。)
- 骨格画素と背景までの距離があれば、元画像(2値画像)が復元できる。
- 上記性質より画像情報の圧縮に用いられる。
- 骨格は図形の主要形状を示すため、文字や図形の認識(特にパターンマッチング)に有効である。

- p.41図3.13の結果から元画像を復元してみよう。

								①	①
①									
	②								
		③	③	③		②	②		①
		③	③				②	②	
	②								①
①				①					

- 骨格からの復元(解答)

								①	①
①									
	②								
		③	③	③		②	②		①
		③	③				②	②	
	②								①
①				①					

骨格画像

								①	①
①									
	②	2	2	2					
	2	③	③	③	2	②	②		①
	2	③	③	2			②	②	
	②	2	2						①
①				①					

1回目の復元画像

								①	①
①	1	1	1	1					
1	②	2	2	2	1	1	1		
1	2	③	③	③	2	②	②	1	①
1	2	③	③	2	1	1	②	②	1
1	②	2	2	1			1	1	①
①	1	1	1	①					

2回目の復元画像

細線化(thinning)

- 図形の線幅を細めて幅1画素の中心線を抽出することを細線化といい、この画像を細線化画像という。

(a) 波形伝搬法：輪郭点を構成する画素で端点(1画素と連結している画素)以外の消去可能な点を消去することで輪郭部から等距離にある中心線を構成する画素を抽出する方法。(図3.14)

(b) マスクパターンを用いる方法：図3.15のパターンを逐一当てはめ、中心画素を削除(1から0に変更)することで細線化する方法。図3.15のパターンは、輪郭点(周辺3画素が0)でかつ、端点ではない点(少なくとも周辺2画素は1)を削除するという考えに基づいている。

- 上記マスクパターンを用いる方法では、細線化の結果は図形の中心になるとは限らない。この方法を改良したものにヒルディッチ(Hilditch)の方法がある。
- 細線化の条件

(a) トポロジカルな性質(連結性、孔など)は保存する。

(b) 線幅は1とする。

(c) 端点は削除しない。

(d) 細線化された線は図形の中心にくる。

(e) 細線化された線上に髭は生じない。

(f) 元画像を回転しても中心線の形態は変わらない。

(g) 図形の交差部で図形は歪まない。

輪郭線抽出(border following)

- 図形の輪郭画素を抽出する処理を輪郭線抽出といい、対象図形を1とすると、手法は次のとおり。

① 画像をラスタ走査して、0から1への変化点(対象図形の開始点)を検出する。

② 開始点の周囲8画素の値を調べ、新たな変化点を求める。この走査を方形走査という。

注) ここで、背景を0、対象図形を1とした場合、変化点は0から1へ変化する点となるが、背景を1 、対象図形を0とした場合は1から0への変化点となる。

また、周囲8画素の検索は時計回りと反時計回りがあり、検索方向と対象図形の追跡方向は一致する。

③ 周囲8画素が全て0であれば、孤立点である。

④ 中心画素に抽出済のマークを付け、方形走査で検出した変化点に移動する。

⑤ ②に戻り、開始点を新たな変化として方形走査を、抽出済マークの画素が発見されるまで繰り返す。

⑥ 抽出済マークの付いた画素が輪郭画素となる。

⑦ 上記手法を全ての図形に対する輪郭が検出されるまで繰り返す。このとき、同じ図形の輪郭を抽出することのないように、0から抽出済マークへ変化する画素は開始点とはしない。

輪郭抽出の特徴

(a) 方形走査を時計回りに行うと、図形の輪郭も時計回りに追跡される。方形走査を反時計回りに行うと、図形の輪郭も反時計回りに追跡される。

(b) 孔の輪郭は方形走査とは逆回りに追跡される。つまり、方形走査を時計回りに行うと、孔の輪郭は反時計回りに、方形走査を反時計回りに行うと孔の輪郭は時計回りに追跡される。

(c) 輪郭線の総数は、図形の輪郭数＋孔の輪郭数である。

注) 原画像と元画像はどちらも使用される。画像処理では処理の元になる画像という意味で元画像を用いることが多い。原画像は写真や絵画で原画として使用されることが多い。

線分(line segment)の検出

- 図形から線要素を抽出する処理を線分検出という。
- 図3.18に示す3x3のマスクパターンで次の値を調べる。

$$E_i = 2(\sum_{i=1}^{3} B_i) - \sum_{i=1}^{3} (A_i + C_i)$$

- $E_i \ (i = 1,2,3,4)$ の中で最大値を与える$B_1 \ B_2 B_3$ の方向に線分が存在する。
- 但し、$E_i \ (i = 1,2,3,4)$ にあまり差がない場合、線分は存在しない。

3.4 線図形の形状分析と符号化

- 線図形の特徴点には次のものがある。

① 端点 : 連結数1の画素。但し、線図形を対象としているため、1画素と連結している画素となる。

② 連結点 : 連結数2の画素。2画素と連結。

③ 分岐点 : 連結数3の画素。3画素と連結。

④ 交差点 : 連結数4の画素。4画素と連結。

(図3.8参照) 上記は線図形を対象としているため、連結数0の内部点はない。

- 画像処理により抽出された線図形にはノイズが含まれるため、次の処理により整形を行う。

(a) 短い枝の除去：分岐点から着目点(端点)までの距離が短い髭部分を除去する。

(b) 短い切断部の接続：端点から着目点(近隣にある端点)までの距離が短く、かつ方向が近い(線分の方向と端点を結ぶ方向がほぼ一致する)場合、端点を結ぶ。

(c) 小さな凹凸の平滑化：線分の方向に対する凹凸をなくして、線分を滑らかにする。

線図形の認識

- 線図形がどのような関数で表現できるのかを調べることを線分の認識という。
- 線図形を表現する関数を求めるには次の方法を用いる。

(a) 最小二乗法を用いる方法: 与えられた線分点列に対して直線や円(範囲を限定すると円弧)の方程式を想定し、最小二乗法により関数の係数を決定する方法。

図3.19参照。但し、図3.19は適用する点列の範囲を決定する式 $|f(x,y)| < C$ である。

線図形の認識

- 最小二乗法を用いて以下のように係数を決定する。

直線の場合

与えられた点列 $(x_i, y_i)(i = 0,1,\ldots,n-1)$に対して

最も近似する直線 $y = ax + b$を求めるためには、

$S = \sum_{i=0}^{n-1}(y_i - ax_i - b)^2$を最小とする aとbを $\frac{\partial S}{\partial a} = 0$及び $\frac{\partial S}{\partial b} = 0$より求める。

円の場合

$S = \sum_{i=0}^{n-1}\{r^2 - (x_i - a)^2 - (y_i - b)^2\}^2$を最小とする aとbを $\frac{\partial S}{\partial a} = 0,\ \frac{\partial S}{\partial b} = 0$及び

$\frac{\partial S}{\partial r} = 0$より求める。

(a) ハフ(Hough)変換を用いる方法:

図3.20の直線αを求める。OBとx軸とのなす角度はθであるから、OBの単位ベクトルは $(\cos\theta, \sin\theta)$となる。

また、直線αの原点Oからの距離はρであるから、点Bの座標は$(\rho\cos\theta, \rho\sin\theta)$となる。直線$\alpha$上の任意の点を$(x, y)$とすると、直線$\alpha$上の任意のベクトルは$(x-\rho\cos\theta, y-\rho\sin\theta)$となる。

一方、直線αとOBは直交するから、ベクトルの内積は0となる。従って、$(x-\rho\cos\theta)\cos\theta+(y-\rho\sin\theta)\sin\theta=0$

$\therefore x\cos\theta+y\sin\theta=\rho(\cos^2\theta+\sin^2\theta)=\rho$　　$p.$47の(3.19)

となる。

つまり、直線 αは$x\cos\theta + y\sin\theta = \rho$と記述できる。
ここで、θとρを固定すると、直線 αはXY平面上で(x, y)をパラメータとする直線となる。一方、(x, y)を固定してθとρをパラメータとして変更すると、XY平面上では(x, y)を通る様々な直線 (原点からの距離 ρ 及び直線への垂線と x軸とのなす角度 θが異なる直線) を表す。つまり、$\theta\rho$平面上では(x, y)を通る曲線となる。
ここで、$\rho = x\cos\theta + y\sin\theta = A\sin(\theta + \phi)$と変形できる。

但し、$A = \sqrt{x^2 + y^2}, \phi = \tan^{-1}\frac{y}{x}$

つまり、$\rho = x\cos\theta + y\sin\theta$は$\theta\rho$平面上では正弦波となる。

複数の点に対する直線 $\rho = x\cos\theta + y\sin\theta$を考え、
これを$\theta\rho$平面上に描画すると、 もし、複数の点を通る直線が同一直線 であれば、
$\rho = x\cos\theta + y\sin\theta$の$\theta$と$\rho$は同じ値を取る。つまり、
$\theta\rho$平面上の複数の曲線は 1点で交わる。(*p*.47図3.20)
交点における曲線の数 が多いほど、同一直線 を構成する点(x, y)が多く存在することを 示し、
直線性は高くなる。

従って、次のようにすれば、ハフ変換を用いて直線を抽出できる。

与えられた点列 $(x_i, y_i)(i = 0,1,\ldots,n-1)$に対して$\rho = x_i \cos\theta + y_i \sin\theta$を$\theta\rho$平面の正弦曲線に変換する。全ての点列 $(x_i, y_i)(i = 0,1,\ldots,n-1)$に対して$\theta\rho$平面の正弦曲線を描画し、多くの曲線が1点$(\theta_j, \rho_j)$に集中していれば、$\rho_j = x\cos\theta_j + y\sin\theta_j$という直線が画像中に存在していることが判る。

線図形の符号化

- 線図形を符号で表現することを線図形の符号化といい、次の方法がある。

(a) チェイン符号化：フリーマン(Freemann)が考案した方法で、線分の方向を図3.21(a)に示す8方向で符号化する方法。符号化には3ビットを用いる。図3.21(b)に例がある。

(b) 方向差分チェイン符号化：チェイン符号化では符号の割り当てに3ビット用いるが、符号の変化は±1が多い、そこで、符号の差分を計算し、確率の高い事象(方向)に少ないビット数を割り当てることで平均ビット数を小さくする方法(これを、エントロピー符号化という)である。(図3.22)

(c) ゾーン符号化：線分の方向を予測し、予測誤差を符号化する方法。アルゴリズムは次のとおり。

① 現在の着目点$p_{i-1}(x_{i-1}, y_{i-1})$ を原点として象限θと象限内のゾーンk の分割を行う。

② 次の点$p_i'(x_i', y_i')$ を $\overrightarrow{p_{i-2}p_{i-1}} = \overrightarrow{p_{i-1}p_i'}$ となるように予測する。ここで、$p_i'(x_i', y_i')$ の象限とゾーン番号を$\theta(p_i'), k(p_i')$ とする。

③ 実測点 $p_i(x_i, y_i)$ の象限とゾーン番号が $\theta(p_i), k(p_i)$ であれば、その差分 $d\theta = \theta(p_i) - \theta(p_i'), dk = k(p_i) - k(p_i')$ を符号化する。(図3.23)

方向差分チェイン符号化とゾーン符号化は1988年にCCITT (Comite Consultatif International Telegraphique et Telephonique：国際電信電話諮問委員会)で標準勧告された

3.5 団塊図形に対する特徴の抽出

- 広がりを持った図形、あるいは、ある程度固まりを持った図形を団塊図形といい、次のようにして抽出することができる。

(a) ラベリング(labeling): 同一連結成分に同一番号を割り当てる処理。次の手順で行う。

① 画像をラスタ走査し、ラベルが割り当てられていない連結成分を探す。

② 見つかった連結成分に同一ラベルを割り当てる。

③ 全ての連結成分にラベルが割り当てられるまで上記処理を繰り返す。

ラベリングにより、連結成分の数を計測できる。

(図3.24参照)

(b) 縮退(shrinking):連結成分の面積を小さくし、1点にする処理を縮退処理という。また、1点となった画素を縮退画素という。細線化処理結果に対して、端点除去を行えばよい。処理形態には次の種類がある。

① 単連結型縮退:孔を含まない連結成分(単連結成分)に対してのみ縮退処理を施す。

② 多重連結型縮退:孔を含む連結成分(多重連結成分)に対しても縮退処理を施す。単連結成分も縮退する。

縮退処理の結果、画像中の連結成分を簡単に計数できる。(p.52 1行目の縮対は縮退の誤り)

(c) 画像中にある連結成分の形状を表す特徴量を形状特徴量といい、次のものがある。

① 面積：連結成分を構成する画素数。

② 周囲長：連結成分を構成する輪郭画素の数。厳密には輪郭画素間の距離の総和。このため、斜め方向の画素間では $\sqrt{2}$ をかけて補正を行う。

③ 円形度：連結成分がどれだけ円に近いかを測る尺度。次式で表され、真円で最大値1を取る。

$$\frac{4\pi(\text{面積})}{(\text{周囲長})^2}$$

$p.52\ (3.22)$のxは不要 or ×(乗算記号)の誤り。

④ 絶対最大長:連結成分を構成する輪郭画素間距離の最大値。

⑤ 幅:絶対最大長に直交する直線に沿った連結成分の最大長さ。

⑥ 方向:水平線(画像の横軸 or x 軸)と絶対最大長の方向とのなす角度。

⑦ 円相当径(Heywood diameter):連結成分と面積の等しい円の直径。

$$\pi r^2 = [\text{連結成分の面積}]\text{となる}\ 2r$$

(d) 凹凸性：連結成分の概略形状を表す性質の一つ。連結成分を凹凸性により分類すると、次のようになる。

① 凸図形(convex)：連結成分の任意の2点を結ぶ線分が全て連結成分の内点から構成される図形。

② 凹図形(concave)：凸ではない図形。

③ 凸閉胞(convex hull)：連結成分を囲む最小の凸図形。最小外接凸図形ともいう。

(e) モーメント(moment)特徴量：原点を中心としたモーメントと重心回りのモーメント(重心モーメントあるいは、セントラルモーメント)がある。

$(p+q)$次のモーメント：$m(p,q)=\sum_{j=0}^{n-1}\sum_{i=0}^{n-1} i^p j^q f(i,j)$

重心モーメント：$M(p,q)=\sum_{j=0}^{n-1}\sum_{i=0}^{n-1}(i-i_g)^p (j-j_g)^q f(i,j)$

但し、$f(i,j)$ $(i,j=0,1,\ldots\ldots,n-1)$は画素$(i,j)$の値、

(i_g,j_g)は連結成分の重心

連結成分の主軸 (連結成分が伸びている方向)と

i軸とのなす角θは次式で与えられる。

$$\theta=\frac{1}{2}\tan^{-1}\left\{\frac{2M(1,1)}{M(2,0)-M(0,2)}\right\}$$

0次モーメント: $m(0,0) = M(0,0) = \sum_{j=0}^{n-1} \sum_{i=0}^{n-1} f(i, j)$

画像$f(i, j)$の面積

1次モーメント: $m(1,0)$ or $m(0,1)$

$$\frac{m(1,0)}{m(0,0)} = \frac{\sum_{j=0}^{n-1} \sum_{i=0}^{n-1} if(i, j)}{\sum_{j=0}^{n-1} \sum_{i=0}^{n-1} f(i, j)} = i_g, \quad \frac{m(0,1)}{m(0,0)} = \frac{\sum_{j=0}^{n-1} \sum_{i=0}^{n-1} jf(i, j)}{\sum_{j=0}^{n-1} \sum_{i=0}^{n-1} f(i, j)} = j_g$$

$\left\{ \frac{m(1,0)}{m(0,0)}, \frac{m(0,1)}{m(0,0)} \right\}$は連結成分の重心

2次モーメント(慣性モーメント)

$$M(2,0) + M(0,2) = \sum_{j=0}^{n-1} \sum_{i=0}^{n-1} (i - i_g)^2 f(i, j) + \sum_{j=0}^{n-1} \sum_{i=0}^{n-1} (j - j_g)^2 f(i, j)$$

$$= \sum_{j=0}^{n-1} \sum_{i=0}^{n-1} \{(i - i_g)^2 + (j - j_g)^2\} f(i, j)$$

(d) フーリエ記述子(Fourier descriptor) : 輪郭線を表現する(周期)関数をフーリエ級数展開し、展開した係数で連結成分の特徴を表す方法。

① Z型フーリエ記述子 : ZahnとRoskiesにより発見された方法で、偏角関数表現を用いる。図形の平行移動、回転、拡大、縮小に対して不変な量。

② G型フーリエ記述子 : Granlundにより発見された方法で、複素関数による位置座標表現を用いる。

③ P型フーリエ記述子 : 位相を意味するPhaseから名づけられた方法で、偏角の指数関数表現を用いる。図形の平行移動、回転、拡大、縮小に対して不変な量となる。また、閉曲線にも適用可能。

Z型フーリエ記述子

p.54図3.28(a)に示す点Aから輪郭線上でl離れた距離にある点における輪郭線の接線の偏角を$\theta(l)$とすると、輪郭線の長さがLのとき、$\theta(l)$は周期関数となる。つまり、$\theta(l+L)=\theta(l)+2\pi$となる。しかしながら、$l=0$を代入すると$\theta(L)=\theta(0)+2\pi$となり、輪郭線を一周すると、元の値に戻らない。そこで、正規化偏角関数$\theta_n(l)=\theta(l)-2\pi\dfrac{l}{L}$を導入する。すると、

$\theta_n(L)=\theta(L)-2\pi\dfrac{L}{L}=\left\{\theta(0)+2\pi\right\}-2\pi=\theta(0)$ となり、輪郭線を一周すると偏角は元に戻る。この正規化偏角関数$\theta_n(l)$を離散フーリエ展開し、その係数を特徴量とする。

G型フーリエ記述子

始点から輪郭線上でl離れた距離にある点を$z(l)=(x(l),y(l))$とする。この点を複素関数で考えると、$z(l)=x(l)+jy(l)$となるので、この複素関数$z(l)$を離散フーリエ展開し、その係数を特徴量とする。