

「C言語によるパターン認識プログラミング」 第1週

○パターン認識とは

パターン認識とは、画像処理により対象とする画像に何（パターン）が映っているのかを識別することである。

人間は目で見ただけで何であるかを瞬時に判断できるが、コンピュータではこれは難しい。

この授業では、簡単なパターン認識の手法を勉強し、自分で簡単なパターン認識の方法を考えてプログラムする。

○デジタル画像データ

画像データは、画素（ピクセル、ドットと呼ばれることもある）が集まっていて、一つ一つの画素は数値で表現されている。簡単な例で例えるならばオセロ板のようなものだと思えばよい。縦横にマス目（画素）が並び、そこに白か黒のコマを全て敷き詰める。



そこで黒（もしくは白）がどのような形（パターン）であるかをコンピュータにより自動的に調べるのがパターン認識である。

なお、ここでは簡単のため、一つの画素では白か黒のどちらかとし、つまりモノクロ画像を対象とする。ここでは扱わない白黒画像（白と黒の間であるグレーなどを含む画像）やカラー画像などは、一つの画素において、白と黒以外のグレーや各色を表すことができるものである。

例えば、3×3画素の画像データで次のようなパターンの場合、

□□□
■ ■ □
■ ■ ■

白を0、黒を1として、

0 0 0
1 1 0
1 1 1

のように表現して扱う。

なお、通常よく使われる画像データは、1600x1200 など非常に多量の画素からなり、かつ各画素において多くの色（16万色以上）が表現できるカラー画像である。

○エッジ抽出

特徴の前に、画像処理を体験的にも理解できるよう画像処理の基礎的な加工手法としてエッジ抽出を紹介する。

ここでは、画像データは白黒（0/1）データであるとする。

例えば 7x7 の画像において、次のような画像の黒い画素は四角のパターンと言える。

```
□□□□□□□
□■□□□□□
□■□□□□□
□■□□□□□
□■□□□□□
□■□□□□□
□□□□□□□
```

この場合パターン内部は塗りつぶされているが、周辺のいわゆるエッジ（周辺部分）のみを取り出した方が処理しやすいことがある。

先の画像からエッジを抽出した（理想的な）例を次に示す。

```
□□□□□□□
□■□□□□□
□■□□□□□
□■□□□□□
□■□□□□□
□■□□□□□
□□□□□□□
```

○一次微分によるエッジ抽出方法

エッジを抽出する画像処理技法は様々あるが、ここでは、最も簡単な一次微分によるエッジ抽出方法を紹介する。

まず、エッジの意味を考えてみよう。

対象の周辺とも考えることができるが、別の考え方として、エッジは変化する部分であると考えられることもできる。つまり、白□→黒■へ、もしくは逆に変化するところがエッジである。

さて、数学が嫌いな方もいるかもしれないが、変化の具合を示す数学的な道具に、「微分」がある。

微分とは変化の具合のことである。たとえば曲線の接線は、その曲線の傾きを表すことになり、グラフにおいて傾きとはどの程度変化しているかを示していて、傾きは微分で求めることができた。

画像処理における微分は、基本的に引き算である。

例えば、□■（0 1）という画像データの変化具合は、0→1 と変化しているので、傾きは 1 と言える。□□（0 0）は変化なしで、傾きは 0 と言える。つまり、各々(1-0=1), (0-0=0)と引き算で求めることができる。これが画像データにおける微分の基本である。

但し、これだけではちょっとまずい。

今、2 x 2 画素の画像を考えて、次の図のように、上二つが白(0)、下二つが黒(1)だとする。この画像の傾きはどうなるであろうか。

```
□□    0 0
■■    1 1
```

先ほどのように、横の変化を考えると、傾きは

0 (=0-0)

0 (=1-1)

となってしまう。

しかし変化の方向を下向き方向で考えると、1(=1-0) 1(=1-0)となる。

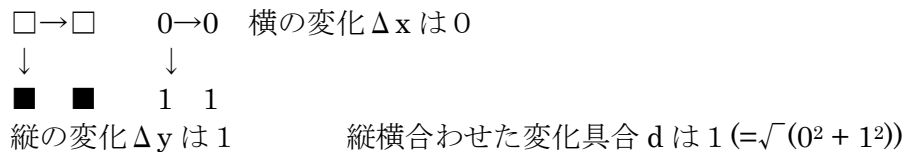
そこで、縦横両方の変化具合を合わせた値 d は、次のような計算で求めることができる。

$$\Delta x = f(x+1, y) - f(x, y)$$

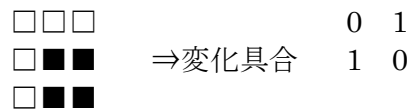
$$\Delta y = f(x, y+1) - f(x, y)$$

$$d = \sqrt{\Delta x^2 + \Delta y^2}$$

上記の 2x2 の画像例において、画素(0,0) (左上の画素) に注目した場合、 $\Delta x = 0 - 0 = 0$, $\Delta y = 1 - 0 = 1$ となり、微分値 d は $1 (= \sqrt{(0^2 + 1^2)})$ となる (C 言語ではルートの計算は sqrt() 関数を利用し、二乗は単に同じものを掛ければよい)。

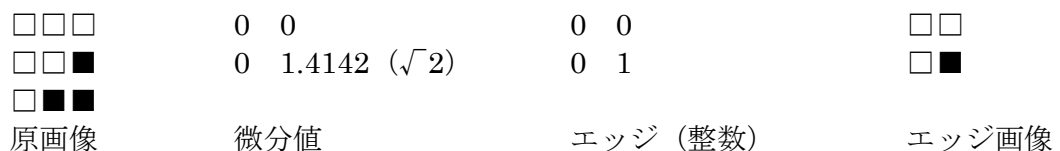


また、3x3 の次のパターンにおける (縦横の) 変化具合は、



となる。このように隣 (右と下) とのいわゆる差を取るので、画像 3x3 は 2x2 のエッジデータとなる。

また、3x3 で次のようなパターンの場合、エッジ (一次微分値) を計算し、1 以上を 1 としてエッジ画像を求めると次のようになる。



なお、画素の値は整数でなければいけないので、たとえば上記ではエッジの計算値が 1 以上の場合は 1 とし、それ以下の場合は 0 とした。

⇒ ★実際に手計算し、それからプログラムしてみよう！

まずは今日の課題用紙の間 1 で手計算をしてみよう。

次にサンプルファイルをダウンロードし、実行方法、エラーの見方&修正を。

それからエッジ抽出処理のプログラムを完成させよう！

◆パターン認識の手法概要

パターン認識は、何らかの画像を対象として、それがどのパターンであるのかを識別することである。たとえば、先のおセロで作ったものを見て、「ハート型」とであると判定することである。

パターンを認識もしくは識別するには、様々な方法がある。

まず、考え方としては非常に単純な方法が、「テンプレートマッチング」である。これは、識別すべき物体のサンプル画像を切り抜いてテンプレートとし、認識をしたい画像に当てはめていき、ぴったりと重なったら識別できたとする方法である。

考え方はシンプルだが、実際には様々な問題がある。

まず、通常「ぴったりと重なる」つまり全く同じパターンが現れるとは限らない。これはノイズ (雑音) が生じたり、観測誤差が生じることもあるばかりか、実際パターンが全く同じとは限らない。更に大きさが違ったり、回転されていたりすることもある。

もう一つのパターン認識の大きな考え方は特徴などによる構造マッチングである。この方法はテンプレートマッチングが、画像そのものを照合させていくのに対し、面積などのパターンの特徴を抽出しておき、これらを照合していく考え方である。

パターン認識においては、どのような形 (パターン) でも認識できるようにすることはほぼ不可能である。たとえば、●と×の2種類のパターン (形状) を認識したいとした場合、★のようなパターンは認識できなくても問題ない。この場合、●と認識しても、またどちらでもないとして認識しても構わない。

さて面積という特徴では大きさしかわからないので、必ずしも形がわかるわけではない。しかし、

ここでは対象とするパターンは、ほぼ決められた一定の大きさであるとする。このような条件を付ければ、たとえば、●と×というパターンを考えた場合、明らかに面積は違う。つまり、認識したい対象の画像において、面積を観測し、その面積が大きければ●、小さければ×と判定することができる。具体的に考えると、たとえば●の面積はおよそ 220 程度だと考えられ、×の面積はおよそ 60 程度だと考えられる場合、あるパターンの面積を観測した結果、180 だとしたら、そのパターンは●であると判定させればよい。より具体的には、たとえば●の面積は 200～240 で、×は 30～90 とし、それ以外の面積の場合はこの 2 種類のパターン以外であると判定させるなどとする。

特徴としてよく利用される幾何学的特徴としては、面積以外にも「密度」、「周囲長」や「アスペクト比（縦横比；縦の長さと横の長さの比）など様々なものがある。

◇面積

パターンの面積のことである。

ここではパターン（物体）は、黒(1)で表現することになっているので、単純に全画素の値を合計すれば簡単に求めることができる。

例えば、次のような 2x2 画素のパターンの場合、

□■	0 1
■■	1 1

黒の画素は全部で三つなので、面積は 3 となる。これをプログラムで実現するには、 $0+1+1+1$ というように全画素を合計すればよい。

⇒★実際のプログラムで確かめてみよう！

※まずはプログラム全体の構造についておおよそ理解しておこう。

次に面積を求めるプログラムをサンプルプログラムで確認しよう。ただし、上半分と下半分の面積（下記参照）を各々抽出して合計しています。

◇上下の面積比

面積を対象画像の上半分と下半分で、各々抽出し、それら面積もしくは面積比を特徴とする。

つまり全体の面積がもし同じ程度だとしても、上半分の面積もしくは下半分の面積で比べると違いが生じることもまる。また、上下に対象なパターンは、面積比はほぼ 1.0 になり、上下に対象ではないパターン（たとえば▲など）は面積比は 1.0 とは異なる。

ここでいう面積比とは、上半分の面積を S_u 、下半分の面積を S_d とした時、 S_u/S_d で計算した値のことをいう。

例えば、次のような 2x2 画素のパターンの場合、

□■	0 1
■■	1 1

上部分の黒の画素は 1 個で、下部分では二つなので、上下それぞれ面積は、 $1 (=0+1)$ 、 $2 (=1+1)$ となる。この場合の上下の面積比は、 $1/2=0.5$ となる。

◇周囲長

周囲長とは、そのパターン（物体）の周囲の長さのことである。

周囲長を求めるにもさまざまな方法が考えられる。一般的には、パターンの周囲を追跡して求め、このつながりの距離を求める。なお、斜めにつながっている場合は、距離は 1 ではなく、 $\sqrt{2}=1.4142$ とすることが多い。

しかし、少し考えてみると、周囲長をもう少し簡単に求めることができる。

今我々はエッジ抽出をすることができる。エッジも基本的にはパターンの周囲とほぼ同等である。よって、エッジ抽出をしたエッジ画像において、エッジ部分の面積を求めればよいことになる。なお、実際に計測してみるとわかるが、必ずしも正確なエッジではなく、かつ画素数が少ないこともあり、周囲長は▲、●、◆を識別するための良い特徴ではない。

◇ヒストグラム（度数分布）

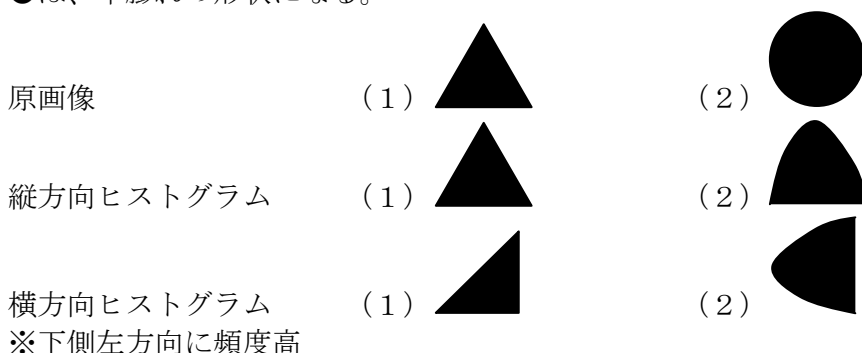
画像データは、二次元なので、ヒストグラムは縦方向と横方向の二つを求めることができる。ここではパターン（物体）は黒(1)であるので、たとえば縦方向のヒストグラムの場合、ある列（縦一列）において、黒の画素の数を数えることにより求められる。

ヒストグラムはこのままでは特徴としては使い難い。これは単に二次元データが一次元になっただけであるので、二つの一次元データを比べるのが難しいからである。

ではどのようにヒストグラムを利用すればよいであろうか？

例えば、今、▲、●の2つのパターンを識別することを考える。縦方向のヒストグラムを考えると、▲も●もどちらも中膨れした形状（ほぼ▲）になるはずで、識別は難しそうである。

では横方向のヒストグラムを考えてみよう。▲は、上が小さく&下が大きくなり（∟のような形状）、●は、中膨れの形状になる。



よって、大きく3つに分けて考えればよい。つまり、縦方向のヒストグラムの場合は左・中・右部分、横方向のヒストグラムの場合は上・中・下部分に分ける。そして各々の部分において度数がおおよそ小さいのか、中くらいなのか、大きいのかを判断する。なお部分に分けるには、簡単に単にそれらの部分の合計を適当に判断させてもそれなりの精度が出せる。

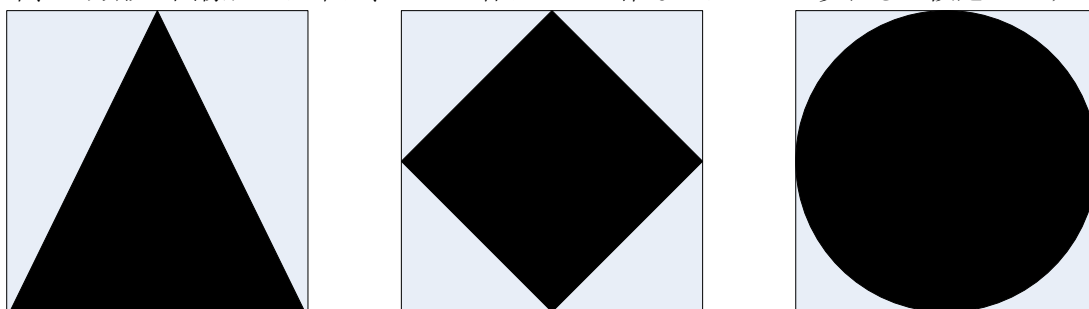
次に今度は、▲、●、◆の三つのパターンを識別することを考える。残念ながら●、◆は、縦方向・横方向共にヒストグラムで区別するのは難しそうであり、別の特徴で区別すべきである。

◇パターン認識の例；特徴の選択

面積という特徴について、どのようなパターンが識別できるかを考えてみよう。

認識したいパターンを▲と◆の2種類として考えてみよう。

今、正方形の画像データ中に、ほぼ一杯にほぼ正確なパターンが現れると仮定してみよう。



簡単な算数の問題であるが、このような理想的なパターンだとすれば、▲も◆も、どちらも面積は同じで、20x20画素の正方形の画像とした場合、どちらも面積は200となる。

よって、▲と◆は、面積という特徴では、区別はできないことになる。

では、どのような特徴だったらこの二つを区別することができるだろうか？

次に、◆と●の2種類のパターンについて考えてみよう。

このような理想的な（正確な）パターンの場合、●の面積は円の面積の公式(πr^2)から約314となるはずで、◆の200よりも大きい。

理想的ではない現実的なパターンは、ノイズや誤差等があるので、これを十分に考慮して考えなければいけない。例えば、◆の面積は180~220、●の面積は「280~340」などと想定することが考えられる。これらの値は、必ずしも正確に決めることはできない。が、十分考慮して理由ある値として設定すべき

である。

◇実際のパターン認識

パターン認識をする場合、実際には一つの特徴を使うわけではない。つまり、複数の特徴を使用し、これらを総合的に判断してどのパターンかを識別する。

ここでは、▲、●、◆の画像パターンを識別することを考えてみよう。また、各々のパターンは必ずしも正確な図形ではなく、また雑音も生じているとし、実際にパターン認識の方法を考えてみよう。つまり、単純なテンプレートマッチングでは難しく、また面積特徴だけでは全てのパターンを識別することは難しい。また、ヒストグラムだけでも難しい。

ここまでで、面積及び上下部分の面積（比）とヒストグラム（{原画像，エッジ画像}×{縦，横}の4種類）を特徴として利用することができる（ここでは周囲長は良い特徴ではないので利用しないとする）。但し、ヒストグラムは、このままではなく、上記の様に更に工夫して利用する必要がある。

では、どうしたらよいであろうか？いずれも単独の特徴量ではうまくいかないので、複数の特徴をうまく組み合わせればある程度解決可能である。例えば▲パターンの場合の各特徴を考えてみると、面積は中くらいで、原画像の縦方向のヒストグラムが左から小・大・小で、横方向が上から小・中・大、エッジ画像の縦方向ヒストグラムは小・小・小、横は小・小・大(中)となるはずである。このように考えて▲、●、◆を区別できる特徴を組み合わせればよい。

なお考え方、つまりパターン認識の方法の正解は一つではなく、複数存在する。但し絶対完璧な認識ができる方法はほぼ困難である。また例えば大中小などの判断なども必ずしも正解があるわけではない。より柔軟に様々な変形パターンがなるべく正確に認識できるように考えてみよう。人間でも●だか◆だか判別できないパターンもあり、100%識別するパターン認識アルゴリズムはほぼ困難（不可能に近い）であることは十分に頭に入れておこう。

特徴によるパターン認識の方法を考える場合、次の点に注意しておこう。

複数のテスト画像を同時に比べるのではなく、ある一つの画像がどのパターンなのかを調べるようにすることに注意しよう。たとえば、▲画像と●画像を比べて、どちらが▲パターンでどちらが●パターンなのかを調べたいわけではない。あくまでも一つの画像を対象に▲パターンなのか●パターンなのかを認識したいわけである。

⇒★各特徴の抽出プログラムを完成させ、実際にどのような数値になるのか確かめよう！

サンプルプログラムでは、上記で紹介した特徴について、およそ完成しています。

未完成な部分としては、面積抽出部分の文法エラー、エッジ抽出、縦横のヒストグラム抽出、3分割ヒストグラムの大中小の判断が未完成です。特徴の抽出プログラムのこれらを完成させて、▲パターンらしき画像、●パターンらしき画像、◆らしき画像の各々に対して、実際の各特徴の値がどのようなになるのかを確かめよう。

なお、サンプルプログラムでは、20x20の画像の中に、およそ18x18に収まるパターンとされていることに注意しておこう。

☆次週パターン認識プログラム全体を完成させるので、これらの特徴値を参考にしながら、どの特徴を利用し、どのように認識させるのかを考えてきてください。