大学で撮った写真を文字認識

目的：画像処理ライブラリの作成
勉強方針：深さ制限探索、一度の勉強に目的を設定してそれを時間内に絶対終わらせる
スケジュール
１　画像のフーリエ変換　学習深度
２　二値画像　学習深度
３　幾何学的変換　学習深度
４　パターン認識　学習深度
５　動画像処理　学習深度
６　画像計測　学習深度
７　ファイル構造　学習深度
８　画像処理クラスの設計を考える、これからのスケジュールを考える　学習深度

画像の基本構造
画素(Pixel)...二次元に配列された格子点。画像を構成する一つの小さい粒。
・画素の色の表現法
　・グレースケール画像(8bit 画像)
　　0 (黒)～255 (白) までの 256 階調で灰色濃淡を表した画像
　・RGB カラー画像(24bit 画像=8*3bit)
　　赤 (Red)、緑 (Green)、青 (Blue) の 3 原色を混ぜて色を表現する方式
　　混ぜる色の濃さはそれぞれ 8bit(0~255)で表す
・HSV 色空間
「色相(Hue)」「彩度(Saturation)」「明度(Value)」の 3 つの組み合わせで色を表現する手法

・標本化
アナログ画像(手描きのイラスト、アナログカメラの写真など)のアナログ信号を一定間隔で区切って、サンプルをとる
・量子化
区切ったものを何ビットの精度(色の精度？)で読み取るかを決める。
・符号化
実際にデジタルデータとして読み取る

・画像の読み込み方法
・ラスタ操作
画像データを一次元配列として走査（スキャン）する
使う場面：画素値の変換、単純二値化処理、ヒストグラム計算
・二次元配列操作
隣接画素を用いた走査
空間フィルタ、特徴点抽出、ラベリング

・K 平均法
あるデータの分布とクラスタ数 K が与えられたとき、データ間の距離を基準にしてデータを K 個のクラスタに分割する処理
RGB 画像の場合、各画素には RGB 値が割り当てられている。これを 3 次元ベクトルとみなすと一枚の RGB 画像は RGB 空間内の点の集合とみることできる。
類似色同士が塊を作って分布しているはずなので、これによって、画像の代表色の取得やそれによる画像分割、色を一番近い平均値に置き換えることで減色が可能になる。

・ヒストグラム
ヒストグラムとは、度数分布をグラフ化したもの。
画像処理におけるヒストグラムでは、横軸に画素値（階調値、色の濃さ）、縦軸にその画素数を取ります。
　・線形濃度変換
　　入力画像から出力画像への線形写像での濃度変換(線形は原点を通る直線といっていい)
　・γ 補正（非線形濃度変換）
　　画像のコントラストを調節し、視認しやすくする(画素地の偏りを γ の値を操ることでコントロールできる)
　　I’(x,y)=Imax*(I(x,y)/Imax)^(1/γ)
　　I(x,y)...変換前の濃度値

Imax は画素値の最大値 8bit 画像なら 255

・ヒストグラム伸縮化

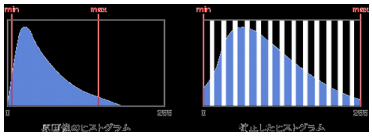
ヒストグラムより、引き伸ばしたい範囲 (a , b) を決定します。

濃度値 0 ～ a は無理矢理、濃度値を 0 にします。

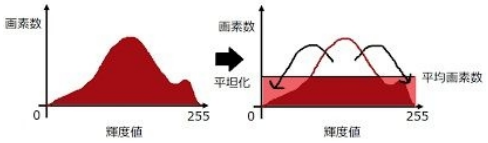
濃度値 b ～ 255 までは無理矢理、濃度値を 最大（255） にします。

濃度値 a ～ b の間の濃度値は、どうやって決めるかというのは、以下の式で決定します。

I’(x,y) = Imax*(I(x,y)-a/b-a)



・ヒストグラム平均化



I’(x,y) = Imax*(1/Height・Width) *0 から I(x,y)までの度数(画素数)の総和

Width,Height...画像の縦横のサイズ

空間フィルタリング

入力画像の注目する画素値だけでなく、その近傍（周囲）にある画素値も利用し、出力画像の画素値を計算する処理。

・空間フィルタリングのやり方

注目している画素とその近傍の画素の濃度値に重み付けをおこない、それらの和をとって注目している画素の新しい濃度値とするような処理を近傍処理という。重み付けに用いる値はオペレーターなどと呼ばれる。

例　入力画像 I　　オペレーター K(カーネル)

I(0,0),I(1,0),I(2,0),I(3,0)　　K(0,0),K(1,0),K(2,0)

I(0,1),I(1,1),I(2,1),I(3,1)　　K(0,1),K(1,1),K(2,1)

I(0,2),I(1,2),I(2,2),I(3,2)　　K(0,2),K(1,2),K(2,2)

I(0,3),I(1,3),I(2,3),I(3,3)

出力画像 I'の端以外の画素値

注目画素とその周囲の画素を同じ位置のカーネル要素とかけて和をとる

これを畳み込み演算という

I’(1,1) = K(0,0)*I(0,0)+K(1,0)*I(1,0)+K(2,0)*I(2,0)

+K(0,1)*I(0,1)+K(1,1)*I(1,1)+K(2,1)*I(2,1)

+K(0,2)*I(0,2)+K(1,2)*I(1,2)+K(2,2)*I(2,2)

出力画像 I’の端の画素値

処理方法は特に定義されていないため、自分で定義

輪郭抽出に使う場合　出力画像の端の画素値は全て 0 にする

平滑化処理に使う場合 入力画像の画素値をそのまま出力画像の画素値にする

平滑化

・平均値フィルタ

注目画素の近傍の画素値の平均値を計算し、その値を新しい画素値とする

平均値フィルタをかけると画像にぼかしがかかった感じになる

平均値フィルタのオペレーター

1/9 * 1,1,1

1,1,1

1,1,1

↑これで注目画素と 8 近傍の計 9 画素の平均値を求めている

・ガウシアンフィルタ

ガウス分布を利用して「注目画素からの距離に応じて近傍の画素値に重みをかける」という処理を行い、自然な平滑化を実現

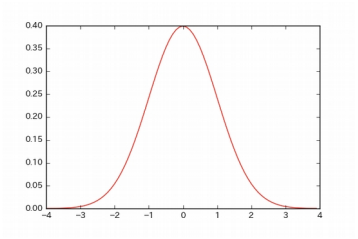
真ん中の注目画素が最も重みが大きくなり、外側の画素ほど重みが小さくなる。

この特性により、画素付近の情報をより残したまま画像をぼかすことが可能。
＊＊＊＊＊

二次の確率密度関数

$$g(x,y,\sigma)=\frac{1}{\sqrt{2\pi}\sigma}exp(-\frac{x^2+y^2}{2\sigma^2})$$

一変量の確率変数 X が、平均 μ、分散 σ^2 の正規分布に従うときの確率密度関数。9 つの画素値の濃度で平均と分散をとってこの関数をつくり、注目画素からの関数上の距離に応じて近傍の画素値に重みをかける。



正規分布確率密度関数

$$f(x)=\frac{1}{\sqrt{2\pi}\sigma}\exp(\frac{-(x-\mu)^2}{2\sigma^2})$$

・メディアンフィルタ
メディアン(Median：中央値)とは、「データを値が小さい順に並べた時に、真ん中にあるデータ」のこと
注目画素と 8 近傍の計 9 画素の中央値を求め、それを新しい画素とする。

輪郭抽出

一次微分
一次微分を計算することで、注目画素の左右・上下の画素値の変化の傾き()を求め、微分した画素値が大きい箇所が輪郭
一次微分 Ix(x,y),Iy(x,y)を求める方法
Ix(x,y) = I(x+1,y) - I(x,y)
Iy(x,y) = Ix(x,y+1) – I(x,y)
このやり方だと、二つの画素の中間における微分値は Ix(x+0.5,y),Iy(x,y+0.5)になり、少し誤差が出る。
両端の画素値を用いることでこれを回避する方法がある
Ix(x,y) = I(x+1,y) – I(x-1,y)
Iy(x,y) = Ix(x,y+1) – I(x,y-1)
行列での表現

$$Kx=\begin{matrix}0,0,0\\-1,0,1\\0,0,0\end{matrix}\qquad Ky=\begin{matrix}0,-1,0\\0,0,0\\0,1,0\end{matrix}$$

縦横方向の一時微分から画素値を求める
I’(x,y) = (Ix(x,y)^2 + Iy(x,y)^2)^(1/2)

プレヴィットフィルタ
一次微分＋平滑化処理

$$Kx=\begin{matrix}-1,0,1\\-1,0,1\\-1,0,1\end{matrix}\qquad Ky=\begin{matrix}-1,-1,-1\\0,0,0\\1,1,1\end{matrix}$$

一次微分と同じく画素値は
I’(x,y) = (Ix(x,y)^2 + Iy(x,y)^2)^(1/2)

ソベールフィルタ
プレウィットフィルタの平均化をかける際に注目画素との距離に応じて重み付けを変化させたもの
プレウィットよりぼかしを軽減できる

$$\begin{matrix}-1,0,1\end{matrix}\qquad\begin{matrix}-1,-2,-1\end{matrix}$$

Kx = -2,0,2　　Ky = 0, 0, 0
-1,0,1　　1, 2, 1

ラプラシアンフィルタ
二次微分を利用して画像から輪郭を抽出する空間フィルタ
Ixx(x,y) = {I(x+1,y) – I(x,y)} - {I(x,y) – I(x-1,y)}
Iyy(x,y) = {I(x,y+1) – I(x,y)} - {I(x,y) – I(x,y-1)}

∇2I(x,y) = Ixx(x,y) + Iyy(x,y) = I(x+1,y) + I(x,y+1) + I(x-1,y) + I(x,y-1) – 4*I(x,y)
よって

K4 = 0,1,0 / 1,-4,1 / 0,1,0　　K8 = 1, 1,1 / 1,-8,1 / 1, 1,1

※K8 は周囲 8 画素を使用した二次微分のカーネル

エンボスフィルタ
輪郭の画素を凸(白)、輪郭の周囲の画素を凹(黒)にすることで輪郭を際立たせる

K = -2,-1,0 / -1,1,1 / 0,1,2

LOG フィルタ
平滑化(ガウシアンフィルタ)＋輪郭抽出(ラプラシアンフィルタ)の合わせ技
ラプラシアンフィルタは二次微分の働きをするため、ノイズが強調されやすい。だがガウシアンフィルタであらかじめ画像を平滑化しノイズを消去することでそれを抑えられる

$$LoG(x,y) = \frac{x^2+y^2-\sigma^2}{2\pi\sigma^6} exp(-\frac{x^2+y^2}{2\sigma^2})$$

この式は、ガウシアンフィルタの式を二階微分することで求まる。

Canny エッジ検出器
①輪郭の検出漏れや誤検出が少ない。
②各点に一本の輪郭を検出する。
③真にエッジの部分を検出できる。
という利点がある

① Gaussian フィルタで画像を平滑化する。
②平滑化された画像の微分を計算する。
③微分した結果から勾配の大きさと方向の計算する。
傾きの大きさ(Ix^2*Iy^2)^(1/2)　方向 θ = arctan(Iy/Ix)
④　Non maximum Suppression 処理をする。
輪郭を細線化する。
注目画素の画素値と、輪郭の勾配方向に隣り合う 2 つの画素値を比較します。
そして、3 つの中で注目画素の画素値が最大でない場合、画素値を 0(黒)に置き換えます。
⑤　Hysteresis Threshold 処理をする
2 つの閾値（最大閾値・最小閾値）で「信頼性の高い輪郭」と「信頼性の低い輪郭」を選び画素を評価し、信頼性の低い輪郭を除去し、途切れている部分をつなぐ。

二値画像
色を 0（黒）と 1（白）の二階調（1bit）で表した画像
・二値化処理
ある閾値を設定してそれ以上か未満かで画素値を白か黒に分ける処理

・適応二値化処理
閾値を固定せず、注目画素と周囲にある画素の画素値の平均値を閾値とする。
これにより、画素ごとに異なる閾値を設定できる。

・大津の二値化処理
① 入力画像 I（8bit 画像）のヒストグラムを求めます。
②ヒストグラムから、画素値の最大値 Imax、最小値 Imin(濃さのことだと思われる)、平均値 μ0 を求めます。
③ Imin～Imax の範囲内で、ある閾値 T を選びます。
④ 閾値 T でヒストグラムを 2 つのクラス（Class）に分けます。
⑤ クラス 1 の分散 σ_1^2、平均値 μ1、画素数 n1 を求めます。
⑥ クラス 2 の分散 σ_2^2、平均値 μ2、画素数 n2 を求めます。
⑦ 以下の式からクラス内分散 σ_w^2 とクラス間分散 σ_b^2 を求めます。
σ_w^2 = n1*σ_1^2+n2*σ_2^2/n1+n2
σ_b^2 = n1*(μ1-μ0)^2+n2*(μ2-μ0)^2/n1+n2
⑧ 手順⑦で求めた 2 つの分散から、分離度 S（クラス内分散とクラス間分散の比)を求めます
S = σ_b^2/σ_w^2
⑨ 手順③～⑧を繰り返し、分離度 S を Imin～Imax の範囲内にある全ての T の分だけ求めます。
⑩ 分離度 S が最大になるときの T を二値化処理に用いる閾値に決定します。

膨張収縮処理
膨張(収縮)処理は二値画像の白領域を増やす(減らす)処理
注目画素の近傍に白色（黒）の画素が 1 つでも存在すれば、注目画素を白色(黒)に置き換え

マスク処理

マスク処理は、特定の部分のみを表示（抽出）し、それ以外の部分を表示しない（黒色画像または白色画像）ようにする画像処理
画像から抜き出す範囲を指定する画像を基準画像という。
基準画像は二値画像が使われる。

ハフ変換
Hough 変換は、画像中に含まれる直線や図形を「多数決」で検出する手法
基本的に二値化してノイズを取り除いて行われる
・直線を検出する場合
y=ax+b の形では x 軸に垂直な直線は a=∞になって厄介だから、
xcosθ+ysinθ=p の形（ヘッセの標準形）で表す
・円を検出する場合
教科書通りの

やり方
①式のパラメーターを点を指定して一つ決め (直線の場合(p,θ)の p
②もう一つのパラメータを自由に動かして当てはまる点があるか調べる。
③あった場合そのパラメータに投票
④票数が多いところが信用性の高い図であることがわかる。

補間法
画像の拡大のこと

最近傍法
画像を拡大した際に最近傍にある画素をそのまま使う線形補間法
他の補間法と比較して処理速度が速い反面、画質が劣化しやすい。
I’(x,y) = I([x/a],[y/a]) []は四捨五入 a は拡大倍率

バイリニア補間法
拡大画像の座標(x', y')における画素値を求める手順

①拡大画像の座標(x', y')を拡大率 a で割り、(x'/a,y'/a)を求めます。
②元画像における(x'/a ,y'/a)の周囲 4 画素の画素値 I(x,y), I(x+1,y), I(x,y+1), I(x+1, y+1)を取得します。
③周囲 4 画素それぞれと(x'/a ,y'/a)との距離を求めます。
④距離によって重み付け（0～１）を行います。（距離が小さいほど重みは大きい）
⑤周囲 4 画素の画素値の加重平均を拡大画像の座標(x', y')における画素値とします。

バイキュービック補間法
バイリニア補間法の性能強化バージョン。周囲 16 画素の画素値を使用するところと、重みづけの式が違う。
重み k の式

$$k = \begin{cases} 1-(a+3)d^2+(a+2)d^3 & (0 \le d < 1) \\ -4a+8ad-5ad^2+d^3 & (1 \le d < 2) \\ 0 & (d \ge 2.0) \end{cases} \quad (1)$$

テンプレートマッチング
入力画像からテンプレート画像(部分画像)と最も類似する箇所を探索する処理

SSD
画素値の差分の二乗和（二乗誤差）」で類似度を評価します。
この場合、値が最小になる場所が類似度が最も高いことになる。
部分画像と比べる位置をひとつづつずらしてこの処理を行い、値が最小の時を結果として出す。

SAD
「画素値の差分の絶対値の和」で類似度を評価。値が最小になる場所が類似度が最も高い。
メリット：計算量が少ない、外れ値の影響を受けにくい
デメリット：照明の影響をかなり受けやすい

NCC
「正規化相互相関」で類似度を評価

$$NCC(d_x,d_y) = \frac{\sum\sum[I(d_x+x,d_y+y)T(x,y)]}{\sqrt{\sum\sum[I(d_x+x,d_y+y)]^2}\sqrt{\sum\sum[T(x,y)]^2}}$$

dx,dy　ひとつづつずらす部分画像の左上
x,y　テンプレート画像の比較する画素値
式の上と下が似ているほど値が 1 に近づく
NCC は、画像をベクトルとみなして内積を計算するため、値がベクトルの長さに対して影響を受けません。

そのため、「照明変化に強い」という優れた特徴があります。

ZNCC
「零平均正規化相互相関」で類似度を評価

$$ZNCC(d_x, d_y) = \frac{\sum\sum[(I(d_x+x, d_y+y) - \mu_I)(T(x,y) - \mu_T]}{\sqrt{\sum\sum[I(d_x+x, d_y+y) - \mu_I]^2}\sqrt{\sum\sum[T(x,y) - \mu_T]^2}}$$

NCC と同じく照明変化に強い

特徴量
Haar-Like 特徴量