目次

★9　パターン情報処理の応用例 (2) — 画像処理いろいろ

画像を対象としてパターンを加工する処理を画像処理と呼ぶ．前回説明したフィルタリングも画像処理の手法の一つである．今回は，グレイスケール画像の画素値を変化させる方法 (**階調変換**) と，拡大縮小や回転のように画像の形を変える方法 (**幾何変換**) の一部 (画素値の補間と拡大縮小，アフィン変換) について述べる (☆ 1)．

☆ 1) この授業では，カラー画像，動画像，三次元画像等に関する話題は全て省略する．また，多岐にわたる画像処理とその応用のほんの一部しか紹介することができない．興味のある人は，この資料の末尾に記した参考文献等を参照してほしい．

★9.1　グレイスケール画像の階調変換としきい値処理

★9.1.1　画素値のヒストグラムと階調変換

画像の明るさを変化させる方法を考えよう．明るさの情報は，画素値の**ヒストグラム**を作ってみるとつかみやすい．例えば，図 1 左の暗い画像では小さな画素値が多い（この例では，画素値の最小値は 0（黒），最大値は 255（白）である）．この画像の画素値に一定数を加えると，図の真ん中に示すように明るい画像を作り出すことができる（この例では全ての画素値に 128 を加えている）．

ヒストグラム: histogram, 度数分布.

このように，画素値を変化させれば，画像の明るさを変化させることができる．このような操作を**階調変換**という．上記のように定数を加減するだけでなく，画素値に定数を掛けたり，非線形な関数を用いたりと様々な演算が利用される．図右の画像は，画素値の分布がなるべく一様分布に近づくようにヒストグラムを均等化（均一化ともいう）した結果である．明暗がはっきりして見やすくなっていることがわかる．

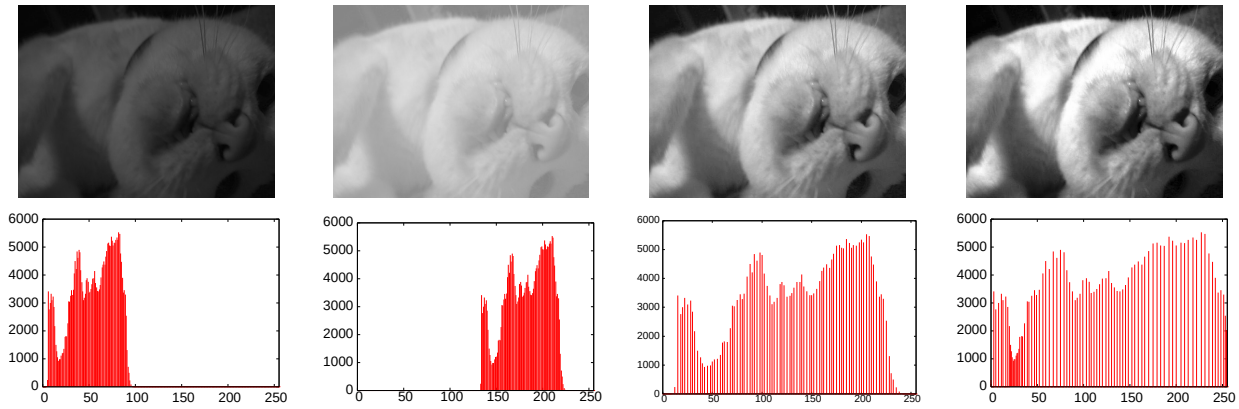


図 1: 画像と画素値のヒストグラム．上の画像の画素値のヒストグラムを下に示す．左から順に，元の画像，その画素値に定数を加えて作った画像，元の画素値に定数をかけて作った画像，ヒストグラムを均等化した画像.

★ 9.1.2 しきい値処理

階調変換の極端な場合として，画素値の基準（しきい値）を定め，画素値がしきい値以下ならその画素は真っ黒に，さもなくば真っ白にする，といった処理を考えることができる．このような処理を**しきい値処理**という．しきい値がひとつであれば画素値を 2 種類の値に分類することになる（この場合を特に**二値化**という）が，N 個あれば $N+1$ 値に分類することもできる．

しきい値: threshold.
二値化: binarization.

図 2 に，しきい値処理によって画像中の図と地を分離した例を示す．この例ではしきい値 (128 としている) を手動で設定しているが，実用的には，何らかの方法で自動的に決定する必要がある．そのため，画素値の分布などから適切なしきい値を求める手法や，位置毎にしきい値を適応的に変化させる手法など様々な手法が研究されている．

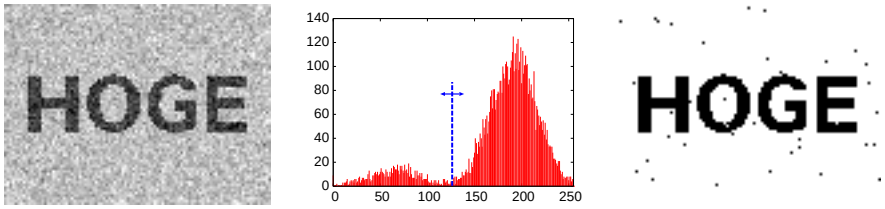


図 2: しきい値処理（二値化）の例．左から順に，元画像，そのヒストグラム，二値化した画像．

しきい値処理は，フィルタリングなどと組み合わせて画像の特徴をとらえるためにもよく用いられる．例えば，エッジ検出の結果をしきい値処理すれば，物体の輪郭線を抽出することができる．図 3 は，画像に

0	1	0
1	−4	1
0	1	0

というマスクを用いたフィルタリング (☆ 2) を行ってその値の絶対値をとり，その結果にしきい値処理を行ったものである．元の画像の中で画素値の変化が激しいところが黒く抜き出されている．

☆ 2) これは，画素値のラプラシアンに相当し，横方向の二階微分と縦方向の二階微分の和になっている．

図 3: フィルタリングとしきい値処理の組み合わせの例．左から順に，元画像，フィルタリングして絶対値をとったもの (見やすくするため白黒を反転してある)，それを二値化したもの．

★ 9.2　画像の幾何変換

★ 9.2.1　画素値の補間と画像の拡大

画像の拡大縮小その他の幾何変換を行うためには，画素値を補間する必要がある．話を簡単にするために，まずは 1 次元のパターンの伸長を例にとる．

$x = 0, 1, 2, 3, 4$ の 5 点で $f(0), \ldots, f(4)$ という値をとるパターンを考え（図 4 左），このパターンを，$x = 0$ を基準として，適当な倍率 a で変数の軸方向に伸長させたいとする．図中の 5 つの赤丸は，$a = 1.25$ の場合（$a > 1$ なのでパターンは伸びる）にこれらの点がどこに移るかを示している．例えば，元の座標（x 軸）で $x = 1$ の位置にあった値 $f(1)$ は，新しい座標（X 軸）では $X = 1.25$ に移っている．同様に，$f(2)$ は，$X = 1.25 \times 2 = 2.5$ の位置に移っている．このとき，元の座標 x と新しい座標 X の間には，次の関係が成り立っている．

$$X = ax \qquad\qquad x = \frac{X}{a} \qquad (1)$$

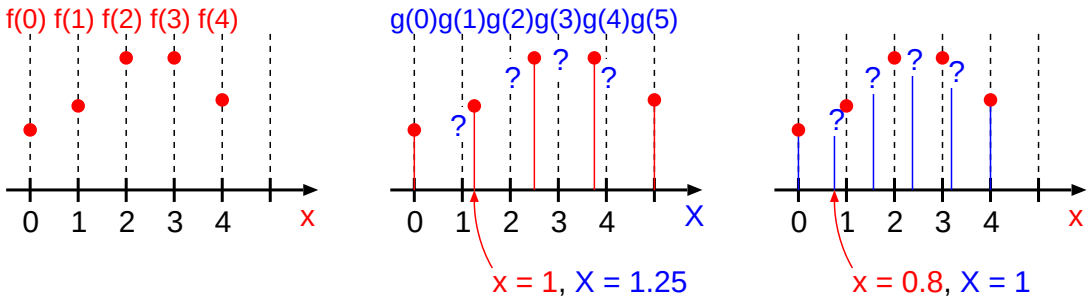


図 4: 1 次元パターンの伸縮

いま考えているのは変数が離散的なパターンなので，これでおわりというわけにはいかない．伸長結果の方も，変数が整数（$X = 0, 1, \ldots$）のときの値で表す必要があるからである．例の場合，図より $g(0) = f(0)$ および $g(5) = f(4)$ だから，これら 2 点の値はすぐにわかる．しかし，$g(1)$ から $g(4)$ までの 4 点は

$$g(1) = f(0.8) \qquad g(2) = f(1.6) \qquad g(3) = f(2.4) \qquad g(4) = f(3.2) \qquad (2)$$

となっており，これらの値はわからない．このように，離散的なパターンを変数軸方向に伸縮させようとすると，$f(0.8)$ や $f(1.6)$ のような値を何とかして計算する必要がある．

この $f(0.8)$ のような値を計算するには，その前後で値のわかっているもの（$f(0)$ や $f(1)$）を利用するのがよさそうである．既知の値を利用して間の値を補うので，このような計算を**補間**という．最も単純な補間の方法は，「最も近い既知の値をそのまま用いる」というものである．これを**最近傍法**という．もう少し工夫を考えると，前後 2 点を結ぶ直線を引いて考えることもできる．これを**線形補間法**という．図 5 左は，式 (2) の例でこれらの補間法を説明したものである．

補間: interpolation. **内挿**とも．最近傍法: nearest neighbor method. 線形補間法: linear interpolation method.

ここまでは 1 次元パターンの伸長を考えたが，2 次元の画像を拡大する場合も，話は同じである．原点を基準として x 軸方向に a 倍，y 軸方向に b 倍することで元画像の位置 (x,y) の画素が位置 (X,Y) に移るとしたら，

$$(X,Y)=(ax,by) \qquad (x,y)=\left(\frac{X}{a},\frac{Y}{b}\right) \tag{3}$$

という関係が成り立つので，これを利用して最近傍法や線形補間法を適用すればよいのである（図 5 右）．ただし，前後 2 点ではなく近傍 4 点の画素値を利用する．例えば，(x,y) の画素値を求める際には，

$$(\lfloor x\rfloor,\lfloor y\rfloor) \quad (\lfloor x\rfloor+1,\lfloor y\rfloor) \quad (\lfloor x\rfloor,\lfloor y\rfloor+1) \quad (\lfloor x\rfloor+1,\lfloor y\rfloor+1) \tag{4}$$

という 4 点を用いる．ここで $\lfloor x\rfloor$ は，x を超えない最大の整数を表す (☆ 3)．

☆ 3) 例えば $\lfloor\pi\rfloor=\lfloor 3.99\rfloor=3$ である．床関数といい，正の実数の場合「切り捨て」に相当する．ガウスの記号 $[\cdot]$ で表すこともある．また，$\lceil x\rceil$ というのもある（天井関数）．

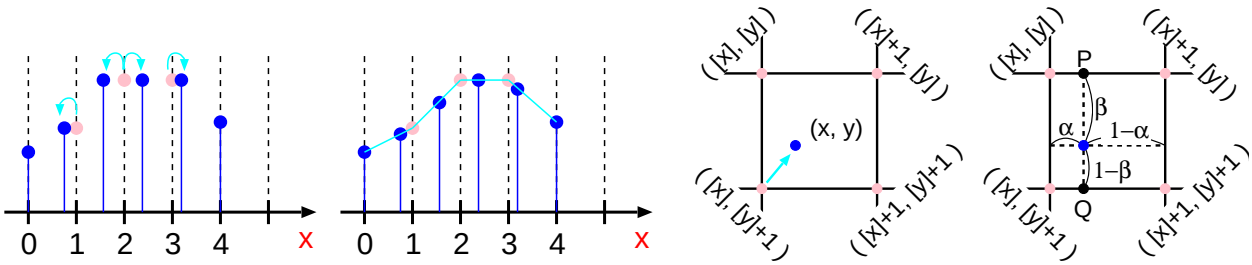


図 5: 最近傍法と線形補間法．左の 2 つは最近傍法／線形補間法による 1 次元パターンの補間を示し，右の 2 つは最近傍法／線形補間法による画素値の補間の例を示す．

線形補間法での画素値の補間法をちゃんと説明すると次のようになる．点 (x,y) が図のような位置にあるとする． $0\le\alpha,\beta\le 1$ である．点 P,Q は左右の点を $\alpha:1-\alpha$ に内分する位置にあるので，その画素値は，x 軸方向の線形補間より

$$(P\text{ の画素値})=(1-\alpha)f(\lfloor x\rfloor,\lfloor y\rfloor)+\alpha f(\lfloor x\rfloor+1,\lfloor y\rfloor) \tag{5}$$
$$(Q\text{ の画素値})=(1-\alpha)f(\lfloor x\rfloor,\lfloor y\rfloor+1)+\alpha f(\lfloor x\rfloor+1,\lfloor y\rfloor+1) \tag{6}$$

と表せる．これらを用いて，今度は y 軸方向の線形補間を考えると，

$$((x,y)\text{ の画素値})=(1-\beta)(P\text{ の画素値})+\beta(Q\text{ の画素値}) \tag{7}$$

が得られる (☆ 4)．

実際に画像を拡大した例を図 6 に示す．最近傍法では隣接画素に同じ値が代入されることがあるためにガタガタした印象であるが，線形補間の方は少しぼけているがより自然な拡大画像となっている．

このように線形補間法ではある程度画質のよい拡大を実現できる．しかし実際の画像処理では，よりよい画質を求めて，非線形関数やスプラインを利用したり，4 近傍だけでなくさらにその周囲の画素値も利用したりする改良手法が用いられることもある (☆ 5)．

☆ 4) 縦横二方向で線形補間を行なうので，この方法は双線形補間法 (bi-linear interpolation method，双一次〜，バイリニア法などとも) と呼ばれることもある．

☆ 5) 3 次多項式を用いるバイキュービック法がよく知られている（興味のある人は調べてみよう）．たいていの画像処理ソフトウェアでは，バイキュービック法を含めた 3 手法の中から使用する手法を選べるようになっている．

図 6: 画像の拡大．左から順に，元画像，最近傍法による拡大，線形補間による拡大．縦横とも 1.25 倍．

Q1. $f(3,6)=88, f(4,6)=64, f(3,7)=72, f(4,7)=104$ として，$f(3.75, 6.5)$ の値を最近傍法および線形補間法で求めよ．

★ 9.2.2　画像の縮小とエイリアシング

画像の縮小は，拡大と同様に式 (3) を考えて（ただし $0 < a, b < 1$(☆ 6)）画素値の補間を行なえばよい．しかし，a, b が小さい場合は，画素値を間引く，すなわち標本化することになるので，エイリアシングを起こさないように注意する必要がある．一般的には，縮小する前に平滑化を行なって高周波数の成分を除去する（ローパスフィルタリングする）方法が用いられる．

☆ 6) もちろん，$1 < a$ かつ $0 < b < 1$ とすれば横に拡大しつつ縦に縮小することになる．また，a, b を負の値にすれば画像を反転することになる．

図 7: 画像の縮小．左から順に，元画像，最近傍法，線形補間，ローパス＋最近傍法．縦横とも 0.5 倍．

★ 9.2.3 アフィン変換

画素値の補間さえできれば，式 (3) のように元画像の座標 (x, y) と変換後の新しい座標 (X, Y) とを関係づける式を定めることで，様々な幾何変換が可能となる．特に，以下の式による変換がよく用いられる．これを（2 次元の）**アフィン変換**という．

$$\begin{pmatrix} X \\ Y \end{pmatrix} = \boldsymbol{A} \begin{pmatrix} x \\ y \end{pmatrix} + \boldsymbol{t} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix} \tag{8}$$

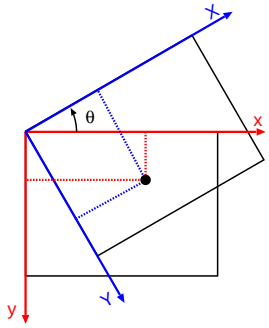
アフィン変換は，行列 $\boldsymbol{A}$ とベクトル $\boldsymbol{t}$ の決め方次第で**拡大縮小**，**平行移動**，**回転**，**反転**などを表せる．すなわちこれらを一般化した変換となっている．例えば，

$$\boldsymbol{A} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}, \qquad \boldsymbol{t} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{9}$$

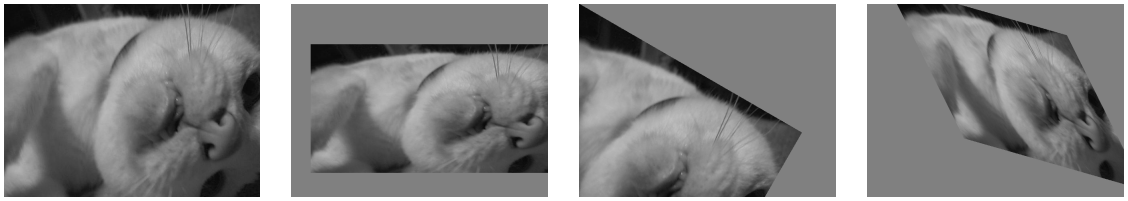
とおけば，右図のように θ 回転する変換となる．図 8 に例を示す．はじめの例（左から 2 つ目の画像）は，元画像を縦に $\frac{2}{3}$ に縮小してから右下に平行移動したものとなっている．次の例は，右図に示すように原点周りに $\frac{\pi}{6}$ だけ回転したものとなっている (☆ 7)．最後のものは，一般のアフィン変換の例である．

紹介はしないが，幾何変換としては，アフィン変換でも表せないようなもっと複雑なもの（例えばレンズを通して歪めるような変換）もいろいろある．(x, y) と (X, Y) とを関係づける式を定めて画素値を補間する，という方法はどんな幾何変換でも共通である．

幾何変換: geometric transformation.
アフィン変換: affine transformation.



☆ 7) 上図のように反時計周りに回転した座標系で元の画像を観察すると，画像自体は時計回りに回転して見える．



$$\boldsymbol{A} = \begin{pmatrix} 1 & 0 \\ 0 & \frac{2}{3} \end{pmatrix} \quad \boldsymbol{t} = \begin{pmatrix} 50 \\ 100 \end{pmatrix}$$

$$\boldsymbol{A} = \begin{pmatrix} \frac{\sqrt{3}}{2} & -\frac{1}{2} \\ \frac{1}{2} & \frac{\sqrt{3}}{2} \end{pmatrix} \quad \boldsymbol{t} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$\boldsymbol{A} = \begin{pmatrix} 0.7 & 0.4 \\ 0.2 & 0.8 \end{pmatrix} \quad \boldsymbol{t} = \begin{pmatrix} 50 \\ -50 \end{pmatrix}$$

図 8: 画像のアフィン変換．左端は元画像．元画像の範囲からはみ出した所はグレイで塗りつぶしてある．