〈第1章〉
C言語入門

1

本章ではフーリエ変換の基礎と画像再構成への応用を勉強するうえで，最低限必要な画像処理として平行移動，拡大と縮小，画像の回転についてはじめに述べる．画像再構成の実験には模擬計測データ（投影）を作成する必要があるが，双線形補間を用いた画像の回転処理でいろいろな画像から投影を作成することができCTの理解につながる．次に，CTに不可欠な固定座標系と回転座標系の関係を述べる．他に，線広がり関数の半値幅（FWHM）の計算，空間フィルタ処理，乱数を用いた透過光子数の推定を扱う．最後に，①バイナリデータをCSV形式のテキストデータとして出力（fprintf関数），②画像データの読み出し（fread関数）と書き込み（fwrite関数），③制御（繰り返し）としてのfor文，④制御（選択）としてのif else文などを用いた例題を載せる．

〔第1節〕 数学と画像の座標系

数学の座標系は，**図1-1**に示すように原点が中心にきて，x軸は右向きでy軸は上向きとなる．それに対して，画像の座標系は，画像の2次元データがメモリ上で左上から右下に順番に格納されているため，原点が左上となり，x軸は右向きでy軸は下向きになる．よって，画像上で数式を扱う場合，座標系の変換が必要となる．通常は，画像の中心に数学の座標系の原点をもってくる．ただし，取り扱う画像は幅と高さともに偶数の場合が多いので，原点は1画素分右下にずらす．8×8画素の画像を用いたときの座標系の変換の様子を**図1-2**に示す．画像の座標系では，2次元配列の番号であるiとjを用いている．両者を変換する式は

$$\begin{aligned} x &= j - N/2 \\ y &= N/2 - i \end{aligned} \tag{1-1}$$

となる．Nは正方形を仮定した画像の幅（高さ）である．横方向では，画像は0から$N-1$となるが，数学の座標系では$-N/2$から$N/2-1$となり，マイナスの画素が1つ多くなる．一方，縦方向では，画像は0から$N-1$となるが，数学の座標系では$-N/2+1$から$N/2$となり，プラスの画素が1つ多くなる．**図1-2**の画像では$-4 \le x \le 3$，$-3 \le y \le 4$の領域内で座標（2, 3）は灰色の画素となる．医用画像処理のプログラムを数学の座標で定式化しそれを画像として出力するときには，常に（1-1）式を意識する必要がある．なお，本書では数学の座標と画像の座標を区別するとき，前者を簡便に数学座標ということにする．そして，混同のおそれがなく，両者の区別が必要ない場合には数学の座標のことを単に座標と記載する．

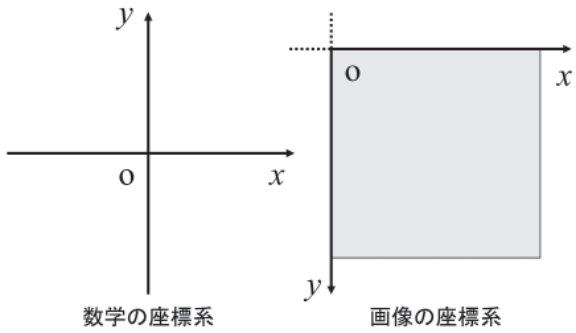

図 1-1　数学の座標系と画像の座標系

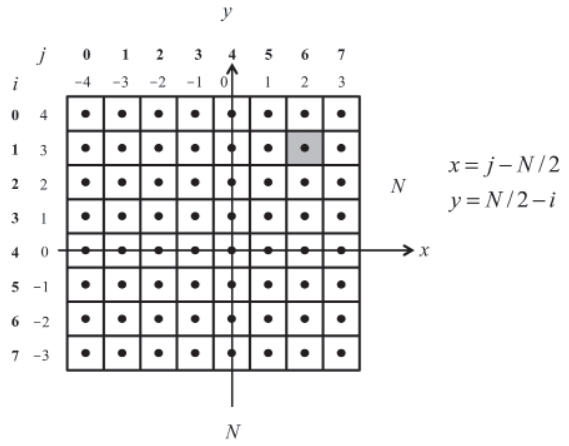

図 1-2　座標系の変換

〔第 2 節〕　平行移動

任意の点を横方向に dx，縦方向に dy だけ平行移動させることを数学座標で考える．移動前の原画像の座標を (x_0, y_0)，移動後の座標を (x, y) とすると，移動前後の座標の関係は次式になる．

$$\begin{aligned} x &= x_0 + dx \\ y &= y_0 + dy \end{aligned} \tag{1-2}$$

平行移動は移動後の画素値を基準に考える．**図 1-3**（a）の原画像を $f_0(x, y)$，原画像の 1 点の座標を (x_0, y_0) とする．（b）の移動後の画像を $f(x, y)$，移動後の 1 点の座標を (x, y) とする．原画像の座標 (x_0, y_0) と移動後の座標 (x, y) の関係は（c）のようになるので，移動後の画素値 $f(x, y)$ の値は次式になる．

$$f(x,y) = f_0(x_0, y_0) = f_0(x - dx, y - dy) \tag{1-3}$$

平行移動後の画素値は原画像から補間によって求める．補間には最近傍補間，双線形補間などがある．

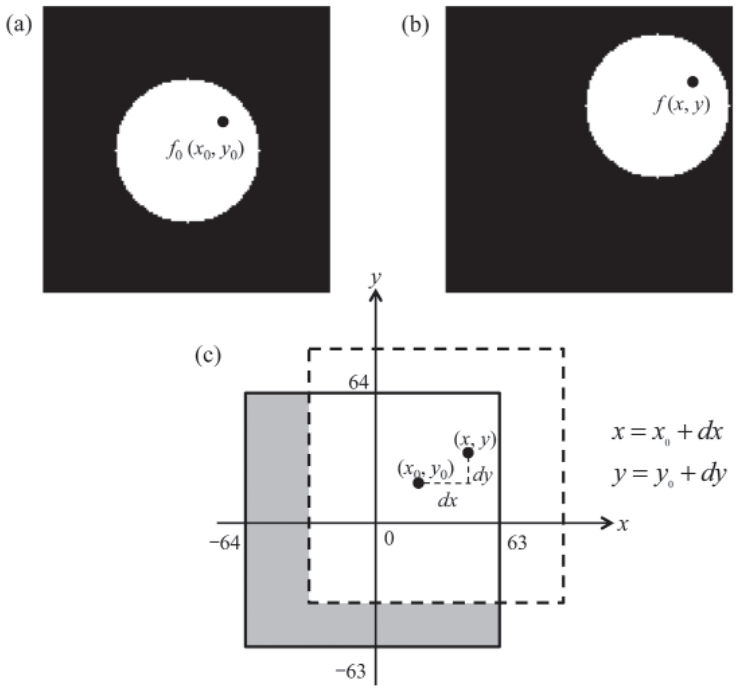

図 1-3　画像の平行移動

最近傍補間は求める座標の近傍画素から最も近い距離にある画素の値（画素値）を割り当てる（参照する）．**図 1-4** で右側を処理後の 8×8 画素の画像 $f(x, y)$，左側を 8×8 画素の原画像 $f_0(x, y)$ とする．黒色の画素を処理後の注目画素，灰色の画素を参照する原画像の画素，灰色の破線の領域は原画像の参照する座標 (x', y') とその近傍画素を拡大した図とする．最近傍補間は求める座標と最も近い近傍画素を参照するので図のように濃い灰色画素の画素値を参照する．処理画像の座標 (x, y) の画素値は次式で表される．

$$f(x,y) = f_0(x',y') = f_0(x_0,y_1) \tag{1-4}$$

このように，最近傍補間で補間を行う場合では，原画像から参照する座標 (x', y') を求め，原画像の座標の近傍画素のうち，最も近い画素の画素値を参照する．

双線形補間は求める座標 (x', y') の近傍にあるいくつかの画素の値に，(x', y') との距離の逆比に応じた重み付けをしてその総和値を割り当てる．双線形補間は濃い灰色画素の画素値をより大きく参照し，薄い灰色画素の画素値をあまり参照しない．処理画像の座標 (x, y) の画素値は次式で表される．

$$\begin{aligned} f(x,y) &= f_0(x',y') \\ &= (x_1 - x')(y_1 - y')f_0(x_0,y_0) + (x' - x_0)(y_1 - y')f_0(x_1,y_0) \\ &\quad + (x_1 - x')(y' - y_0)f_0(x_0,y_1) + (x' - x_0)(y' - y_0)f_0(x_1,y_1) \end{aligned} \tag{1-5}$$

座標と配列の関係は

$$(x_0,y_0) = (j_0,i_0),\quad (x_1,y_0) = (j_1,i_0),\quad (x_0,y_1) = (j_0,i_1),\quad (x_1,y_1) = (j_1,i_1) \tag{1-6}$$

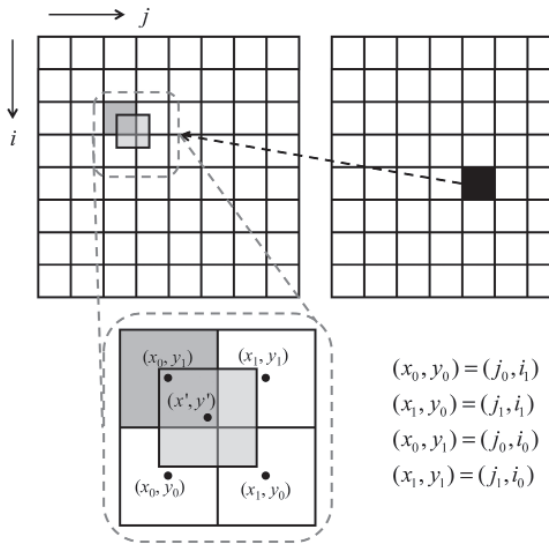

図 1-4　最近傍補間と双線形補間

であるから

$$f(x,y) = (j_1 - x')(i_1 - y')f_0(j_0, i_0) + (x' - j_0)(i_1 - y')f_0(j_1, i_0)$$
$$+ (j_1 - x')(y' - i_0)f_0(j_0, i_1) + (x' - j_0)(y' - i_0)f_0(j_1, i_1) \qquad (1\text{-}7)$$

となる．このように，双線形補間は原画像から参照する座標（x', y'）を求め，その座標から近傍画素との距離と画素値を参照する．

〔第 3 節〕　拡大と縮小

原点を中心とする拡大と縮小処理では，横方向の拡大（縮小）率を a_x，縦方向の拡大（縮小）率を a_y とすると，変換前の座標（x_0, y_0）と変換後の座標（x, y）の関係は次式で表される．

$$x = a_x \times x_0$$
$$y = a_y \times y_0 \qquad (1\text{-}8)$$

原画像を（1-8）式で演算し処理画像を求めると欠損が生じる．例として，**図 1-5** のように原画像の配列を img，処理画像の配列を ima とし，画像の中心を拡大の中心として 2 倍に拡大する処理を考える．配列の上と横にある数字は座標を示す．画像の拡大を（1-8）式の数式通りに行うと，img の①の画素は拡大前の座標（−2, 2）から拡大後の座標（−2×2, 2×2）=（−4, 4）となり，座標（−4, 4）に画素値を代入する．②以降も①と同様に ima に画素値を代入していくと，処理画像には画素値が入っていない画素が生じ欠損となる．**図 1-6** に実際に欠損が生じた画像と後に説明する正しい方法で拡大した画像を示す．これらの欠損を防ぐために，処理画像の座標を原画像のどの座標に対応するか調べる．例えば，ima の座標（3, 3）に入る画素値は，3/2 = 1.5 から img の座標（1.5, 1.5）の画素値を取り出す．しかし，img に座標（1.5, 1.5）は存在しないため補間で画素値を求める．処理画像 $f(x, y)$ の値は原画像の $f_0(x_0, y_0)$ と等しく，これらの関係は次式で表される．

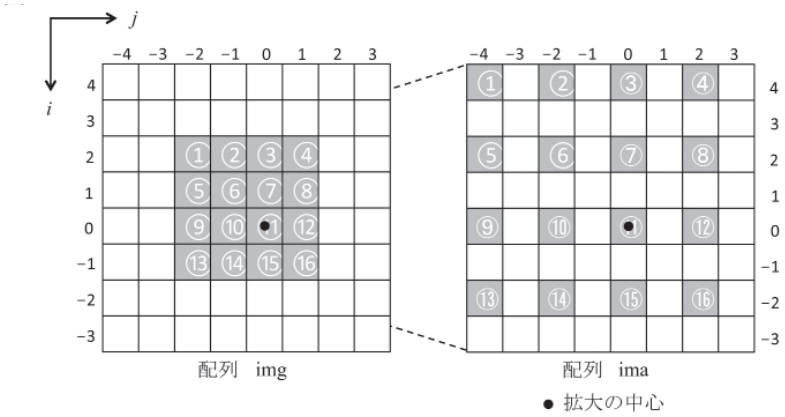

図 1-5　画像の拡大と縮小

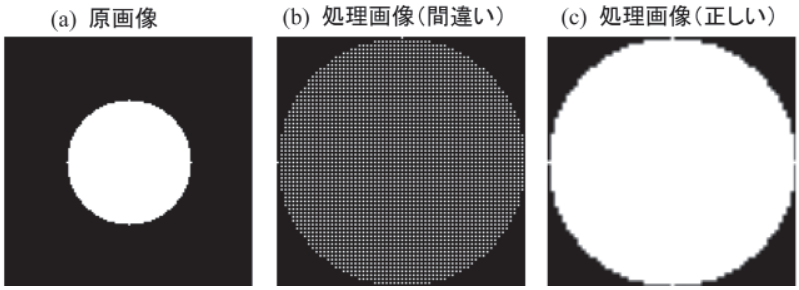

図 1-6　画像の拡大例

$$f(x,y) = f_0(x_0,y_0) = f_0(\frac{x}{a_x},\frac{y}{a_y}) \tag{1-9}$$

配列表示で処理画像の座標を（j, i）とすると次式で表される．

$$f(j,i) = f_0(x_0,y_0) = f_0(\frac{j}{a_x},\frac{i}{a_y}) \tag{1-10}$$

さらに，拡大の中心は画像の中心にしているので，原画像の座標（x_0, y_0）と処理画像の座標（j, i）の関係は

$$\begin{cases} j - N/2 = a_x(x_0 - N/2) \\ N/2 - i = a_y(N/2 - y_0) \end{cases} \tag{1-11}$$

となり，（x_0, y_0）は次式で表される．

$$\begin{cases} x_0 = \dfrac{(j - N/2)}{a_x} + N/2 \\ y_0 = N/2 - \dfrac{(N/2 - i)}{a_y} \end{cases} \tag{1-12}$$

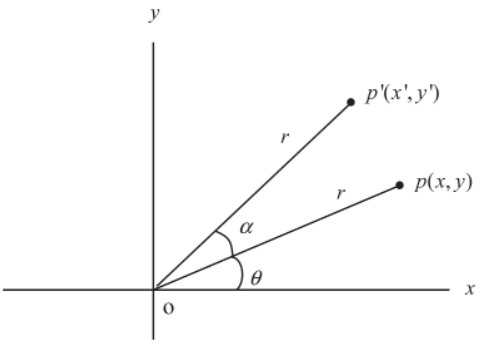

図 1-7　座標の回転

〔第 4 節〕　座標の回転

図 1-7 の座標上の点 p は x 軸から反時計回りに θ 回転した位置にある．点 p がさらに反時計回りに α 回転し p' の位置になるとき点 p と点 p' の座標の関係がどうなるか調べてみよう．p の座標を (x, y)，p' の座標を (x', y') とすると図から

$$\begin{aligned} x' &= r\cos(\theta+\alpha) \\ y' &= r\sin(\theta+\alpha) \end{aligned} \tag{1-13}$$

三角関数の加法定理

$$\begin{aligned} \cos(\alpha+\beta) &= \cos\alpha\cos\beta-\sin\alpha\sin\beta \\ \sin(\alpha+\beta) &= \sin\alpha\cos\beta+\cos\alpha\sin\beta \end{aligned} \tag{1-14}$$

を用いると

$$x' = r\cos(\theta+\alpha) = r\cos\theta\cos\alpha - r\sin\theta\sin\alpha$$

$$y' = r\sin(\theta+\alpha) = r\sin\theta\cos\alpha + r\cos\theta\sin\alpha$$

ここで

$$x = r\cos\theta, \quad y = r\sin\theta$$

であるから

$$x' = x\cos\alpha - y\sin\alpha$$

$$y' = x\sin\alpha + y\cos\alpha$$

行列では以下のように表される．

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{1-15}$$

図 1-8 の実線は（1-15）式で反時計回りに回転させた画像，背景の点線は画像を格納する座標を示す．最近傍補間は回転後の座標 (x', y') を整数化し背景の座標に原画像の値を割り振るので，(x', y') によっ

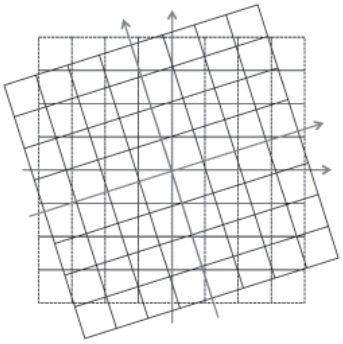

図 1-8　画像の回転処理に使用する配列

ては整数化の際に背景の座標に値が入らない可能性があり欠損を生じる．(1-15) 式に基づいたプログラム P1-01 rotate_nearest1.c を以下に示す．原画像の配列を img，回転後の画像の配列を ima，回転後の座標 (x', y') は (xr, yr) としている．

P1-01 rotate_nearest1.c

```
for (i = 0 ; i < N ; i++) {
   y = N/2-i;
   for (j = 0 ; j < N ; j++) {
      x = j-N/2;
      xr = x*cos(ang) - y*sin(ang);              ①（x, y）座標の回転
      yr = x*sin(ang) + y*cos(ang);
      jx = (int)(xr + N/2+0.5);                  ②数学座標から画像座標に変換し最近傍補間
      iy = (int)(N/2 –yr+0.5);
      if ((0 <= iy && iy < N) && (0 <= jx && jx < N))        ③正方形内に入るかの判別
         ima[iy][jx] = img[i][j];                ④画像の格納
      }
}
```

プログラムに慣れるまでは，②数学座標から画像座標に変換し最近傍補間は，別の変数 xr2，yr2 を用い，①数学座標から画像座標に変換 xr2 =(xr + N/2)，yr2 = (N/2 –yr)，②最近傍補間 jx = (int)(xr2 + 0.5)，iy = (int)(yr2+0.5) と分けて記載するとよい．**図 1-9** に処理画像を示す．画像は数式通りに反時計回りに回転しているが，値のない欠損が（b）5°，（c）10°，（d）30° と回転角が大きくなるにつれ観察される．

これを避けるには（1-15）式を逆にして回転後の画像の画素が原画像のどの画素に対応するか，という観点からプログラムを作成する．(1-15) 式の行列は正規直交行列なので逆行列は転置して求められ，回転後の座標 (x', y') から回転前の座標は

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} x' \\ y' \end{pmatrix} \tag{1-16}$$

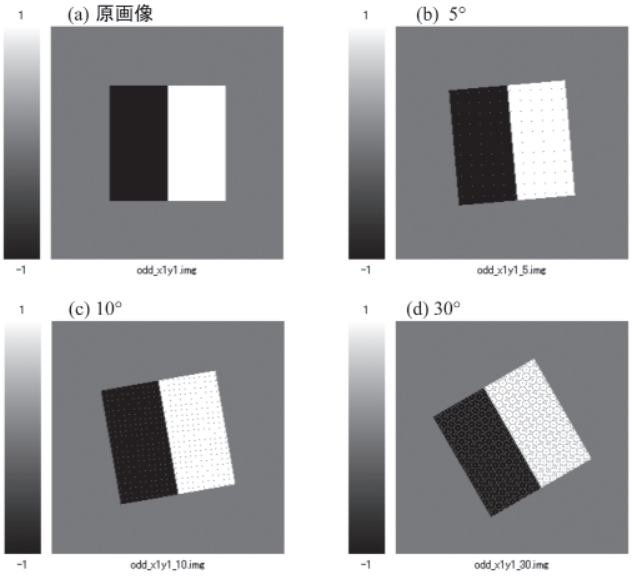

図 1-9　最近傍補間による画像の回転（欠損が生じる処理）

となる．(1-16) 式に基づいた P1-2 rotate_nearest2.c を以下に示す．異なるのは太字にした①と④のみである．

P1-2 rotate_nearest2.c

```
    for (i = 0 ; i < N ; i++) {
      y = N/2-i;
      for (j = 0 ; j < N ; j++) {
        x = j-N/2;
        xr = x*cos(ang) + y*sin(ang);          ①回転後の座標から回転前の座標を逆変換で求める
        yr = -x*sin(ang) + y*cos(ang);
        jx = (int)(xr + N/2+0.5);              ②数学座標から画像座標に変換し最近傍補間
        iy = (int)(N/2 –yr+0.5);
        if ((0 <= iy && iy < N) && (0 <= jx && jx < N))      ③正方形内に入るかの判別
          ima[i][j] = img[iy][jx];             ④画像の格納
        }
    }
```

図 1-10 に P1-2 rotate_nearest2.c による処理画像を示す．画像は反時計回りに回転し欠損は生じない．

〔第 5 節〕　固定座標系と回転座標系

画像の回転は（x, y）座標系 1 つで行えるが，図 1-11 のように CT では被検者に固定した座標とその回りを回転する検出器の座標を表す 2 つの座標系が必要になり，前者を固定座標系（x, y），後者を回転