技4

第4章

変形/移動処理

吉田 大海

4-1 画像を水平にしたいとき等に使う「回転変換」

収録フォルダ：回転変換

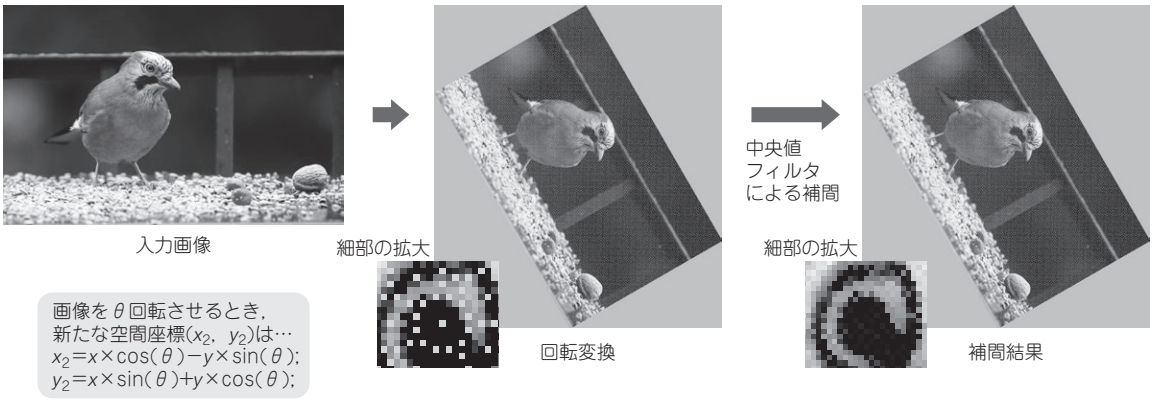

図1　回転変換…座標変換により画像を回転させることができる

画像の回転変換処理は，画像を設定した角度に回転させる処理です．写真を撮影をしたときに，うまく水平に撮れなかったときは，この変換処理によって補正できます．

● 仕組み

画像の回転処理の仕組みを**図1**に示します．入力画素の座標を(x，y)とすると，変換結果の座標(x_2，y_2)は以下の式で求まります．

$$x_2 = x \times \cos\theta - y \times \sin\theta$$
$$y_2 = x \times \sin\theta - y \times \cos\theta$$

ただし，この式をそのまま適用すると画像の水平位置が大きく移動してしまうため，必要に応じてx_2に$(Y-1) \times \sin\theta$を加えるようにします(Yは縦方向の画素数)．

変換結果の座標(x_2，y_2)は小数点になることがあります．このため切り捨てや切り上げによって空白になってしまう場所がでてきます．これを取り除きたい場合は，中央値フィルタ(第6章参照)などを適用します．

リスト1　画像を回転するプログラム(抜粋)

```
for( y = 0; y < Y; y++ ){
  for( x = 0; x < X; x++){

    // 入力画像から画素値を読み込む
    p[0] = img->imageData[img->widthStep*y + x*3 ];

    // 回転変換
    x2 =   x     * cos((2.0*3.14*double(t)) / 360.0)
         - y     * sin((2.0*3.14*double(t)) / 360.0)
         + (Y-1) * sin((2.0*3.14*double(t)) / 360.0);
    y2 =   x     * sin((2.0*3.14*double(t)) / 360.0)
         + y     * cos((2.0*3.14*double(t)) / 360.0);
    //   + (X-1) * sin((2.0*3.14*double(t)) / 360.0);

    // 画素値を出力画像として書き込む
    img2->imageData[img2->widthStep*y2+x2*3]   = p[0];
    img2->imageData[img2->widthStep*y2+x2*3+1]= p[0];
    img2->imageData[img2->widthStep*y2+x2*3+2]= p[0];
```

● 実行結果

画像を回転するプログラムを**リスト1**に示します．実行結果は**図1**に示した通りです．