・画像回転アルゴリズム

プログラムは以下のようになっている。

X = x \* cosf(rad) - y \* sinf(rad);

Y = x \* sinf(rad) + y \* cosf(rad);

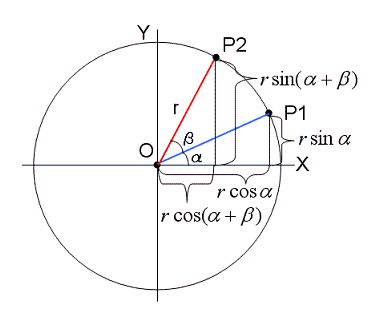


図1　P1をβ回転させた点P2を表す図

なぜ上のような式が成り立つかを説明するために、角度αで原点からの距離がrの点P1(x,y)をβ回転させた点をP2(X,Y)とすると、加法定理より、以下の式が成り立つ。

X = rcos(α+β) = rcosαcosβ - rsinαsinβ)

Y = rsin(α+β) = rsinαcosβ + rcosαsinβ

また、x = rcosα,y = rsinαより

X = xcosβ – ysinβ

Y = ycosβ + xsinβ

よって、画像回転の式が求められることが分かる。

・原画像

　今回使用した原画像を以下に図2に示す。



図２　原画像

・回転画像1（45度回転）

原画像を45度回転させた画像を以下に図3に示す。



図3 45度回転画像

・回転画像２（-60度回転画像）

原画像を-60度回転させた画像を以下に図4に示す。



図4 -60度回転画像

参考文献

回転行列

http://www.sist.ac.jp/~kanakubo/research/hosoku/kaiten\_gyoretu.html

また、今回作成したプログラム（45度回転）を以下に示す。

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#define W 256

#define H 256

void rotateCoordinate(unsigned char f[],unsigned char g[],float theta){

int i, j;

int I, J;

　　 float x;

float y;

float X;

float Y;

float rad = theta \* M\_PI / 180.0;

/\*\*\*i,jをx,yに変換\*\*\*/

for (i = 0; i < H; i++)

{

for (j = 0; j < W; j++)

{

x = -(W - 1.0)/2.0 + j;

y = (H - 1.0)/2.0 - i;

X = x \* cosf(rad) - y \* sinf(rad);

Y = x \* sinf(rad) + y \* cosf(rad);

I = (H - 1.0)/2.0 - Y;

J = (W - 1.0)/2.0 + X;

if(I <= 256 && I >= 0 && J >= 0 && J < 256)

g[W\*I+J] = f[W\*i+j];

}

}

}

int main()

{

int i, j;

unsigned char f[256\*256]; /\*\* 原画像用の配列 \*\*/

unsigned char g[256\*256]; /\*\* トリミング後の画像の配列 \*\*/

FILE \*fp1 ,\*fp2;

float theta;

theta = 45.0;

/\*\*原画像ファイル読み込み\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

fp1 = fopen("lenna\_uchar\_256-256.raw" , "rb");

fread( f , 1 , 256 \* 256 , fp1);

fclose( fp1 );

for (i = 0; i < H; ++i)

{

for (j = 0; j < W; ++j)

{

g[W\*i+j] = 0;

}

}

rotateCoordinate(f,g,theta);

/\*\*トリミング画像ファイル書き込み\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

fp2 = fopen( "kadai1\_45.raw" , "wb" );

fwrite( g , 1 , 256 \* 256 , fp2 );

fclose( fp2 );

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

return 0;

}