PBL　第2回課題

16X3128

馬場　俊弥

１、最近傍補完の計算式

最近傍補完の計算式を以下に図1に示す。

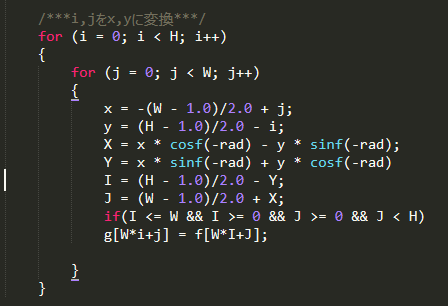


図１　最近傍補完の計算式

２、双線形補完の計算式

双線形補完の計算式を以下に図2に示す。

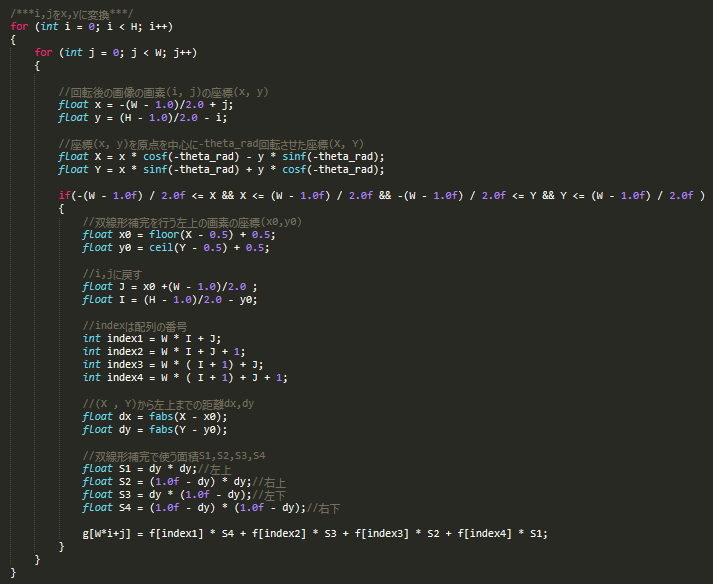


図2　双線形補完の計算式

３、最近傍補完を用いた回転後の画像

最近傍補完を用いた回転後の画像を以下に図3に示す。



図3　最近傍補完を用いた回転後の画像

４、双線形補完を用いた回転後の画像

双線形補完を用いた回転後の画像を以下に図4に示す。



図4　双線形補完を用いた回転後の画像

５、双線形補完を用いた画像回転プログラム

双線形補完を用いた画像回転プログラムを以下に示す。

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#define W 256

#define H 256

float theta = 45.0;

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

int main()

{

float f[W\*H]; /\*\* 原画像用の配列 \*\*/

float g[W\*H]; /\*\* トリミング後の画像の配列 \*\*/

FILE \*fp1 ,\*fp2;

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

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

fread( f , sizeof(float) , 256 \* 256 , fp1);

fclose( fp1 );

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

{

for (int 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( "kadai2\_45.raw" , "wb" );

fwrite( g , sizeof(float) , 256 \* 256 , fp2 );

fclose( fp2 );

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

return 0;

}

void rotateBilinear(float f[],float g[],float theta)

{

/\*\*\*i,jをx,yに変換\*\*\*/

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

{

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

{

//回転後の画像の画素(i, j)の座標(x, y)

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

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

//座標(x, y)を原点を中心に-theta\_rad回転させた座標(X, Y)

float X = x \* cosf(-theta\_rad) - y \* sinf(-theta\_rad);

float Y = x \* sinf(-theta\_rad) + y \* cosf(-theta\_rad);

if(-(W - 1.0f) / 2.0f <= X && X <= (W - 1.0f) / 2.0f && -(H - 1.0f) / 2.0f <= Y && Y <= (H - 1.0f) / 2.0f )

{

//双線形補完を行う左上の画素の座標(x0,y0)

float x0 = floor(X - 0.5) + 0.5;

float y0 = ceil(Y - 0.5) + 0.5;

//i,jに戻す

float J = x0 +(W - 1.0)/2.0 ;

float I = (H - 1.0)/2.0 - y0;

//indexは配列の番号

int index1 = W \* I + J;

int index2 = W \* I + J + 1;

int index3 = W \* ( I + 1) + J;

int index4 = W \* ( I + 1) + J + 1;

//(X , Y)から左上までの距離dx,dy

float dx = fabs(X - x0);

float dy = fabs(Y - y0);

//双線形補完で使う面積S1,S2,S3,S4

float S1 = dy \* dy;//左上

float S2 = (1.0f - dy) \* dy;//右上

float S3 = dy \* (1.0f - dy);//左下

float S4 = (1.0f - dy) \* (1.0f - dy);//右下

g[W\*i+j] = f[index1] \* S4 + f[index2] \* S3 + f[index3] \* S2 + f[index4] \* S1;

}

}

}

}