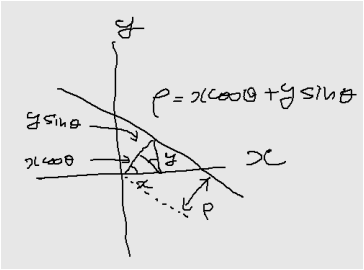
Hough 変換について
MATLAB,C 言語で HOUGH 変換をする場合、扱う座標系がことなるため、若干面倒である。ここでは、なるべく計算を簡略化するように考慮した座標計算アルゴリズムをベースに解説を行っていく。
Hough 変換をする場合
以下の
MATLAB xy 配列：x_{matlab}, y_{matlab}
C 言語 xy 配列：x_{clang}, y_{clang}
xy 座標：x, y
ρ θ座標：ρ, θ
MATLAB ρ θ配列：$\rho_{matlab}, \theta_{matlab}$
C 言語 ρ θ配列：$\rho_{clang}, \theta_{clang}$
を定義する。
変換手順は以下の通りである。
画像 MATLAB xy 配列→
houghf(xy 座標→変換→ρ θ座標→変換→C 言語 ρ θ配列)→
MATLAB ρ θ配列→変換式→ρ θ座標
→変換式→xy 座標

配列座標は、特に配列座標は、C 言語系の 0 から、MATLAB は 1 から始まるので、その点を明確にして行う。
xy 座標
直線の Hough 変換の定義では、
$\rho = x\cos\theta + y\sin\theta$



を使い、xy 空間のデータをρ θ座標にマッピングする。今、それぞれの変数の関係とコンピュータに実装する場合の配列変数の関係を以下のように定義する。
ρ θ配列、xy 配列の範囲

MATLAB	C 言語
$\theta_{matlab} = 1 \sim \theta_{max}$	$\theta_{clang} = 0 \sim \theta_{max} - 1$
$\rho_{matlab} = 1 \sim \rho_{max}$	$\rho_{clang} = 0 \sim \rho_{max} - 1$
$y_{matlab} = 1 \sim y_{max}$	$x_{clang} = 0 \sim x_{max} - 1$
$x_{matlab} = 1 \sim x_{max}$	$y_{clang} = 0 \sim y_{max} - 1$

実際のパラメータρ θ座標 xy 座標の範囲は、

$\theta = 0 \sim \pi - \frac{\pi}{\theta_{max}}$

$$\rho = -\frac{\sqrt{x^2{}_{max} + y^2_{max}}}{2} \sim \frac{\sqrt{x^2{}_{max} + y^2_{max}}}{2} - \frac{\sqrt{x^2{}_{max} + y^2_{max}}}{\rho_{max}}$$

$$x = -\frac{x_{max}}{2} \sim \frac{x_{max}}{2} - 1$$

$$y = -\frac{y_{max}}{2} \sim \frac{y_{max}}{2} - 1$$

と定義する。$\theta = 0 \sim \pi$ としても良いが、0 とπでは向きが異なるだけで同じ方向を示すためあえて、$\frac{\pi}{\theta_{max}}$ を引いている。これは、計算を簡略化する意味もある。

ρは、直線の式の定義より原点からの長さを意味する。画面内であらわされる直線のうちでもっとも長い線は対角線になる。ここでは、画像の中心を原点とするため、最大の距離は、対角線の長さ割る２で表現できる。

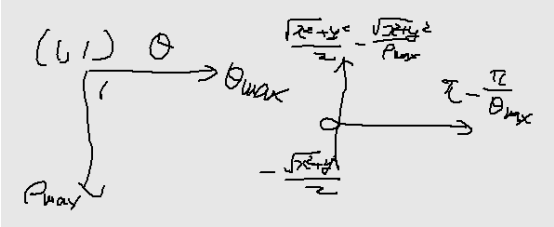
ここでは、MATLABρθ配列、C 言語ρθ配列、MATLAB xy 配列、C 言語 xy 配列 xy 座標の５つの座標を扱っていく。

ρθ座標との値と INDEX との関係は、原点が異なるため以下のように定義する。

MATLABρθ配列	C 言語ρθ配列	ρθ座標
(1,1)	(0,0)	$\left(\frac{\sqrt{x^2_{max} + y^2_{max}}}{2} \quad -\frac{\sqrt{x^2_{max} + y^2_{max}}}{\rho_{max}} \quad 0\right)$
$(\rho_{max} \quad \theta_{max})$	$(\rho_{max} - 1 \quad \theta_{max} - 1)$	$\left(-\frac{\sqrt{x^2_{max} + y^2_{max}}}{2} \quad \pi - \frac{\pi}{\theta_{max}}\right)$

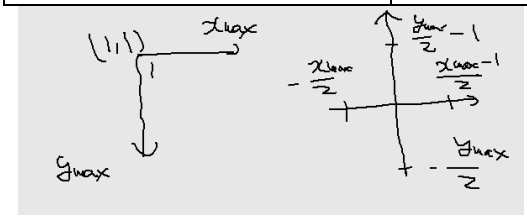
これらの関係よりρθ配列とρθ座標の関係を作ると

MATLABρθ配列からρθ座標への変換	C 言語ρθ配列からρθ座標への変換
$\theta = \frac{\pi}{\theta_{max}}(\theta_{matlab} - 1)$ $\rho = \frac{\sqrt{x^2_{max} + y^2_{max}}}{\rho_{max}}\left(-\rho_{matlab} + \frac{\rho_{max}}{2}\right)$	$\theta = \frac{\pi}{\theta_{max}}\theta_{clang}$ $\rho = \frac{\sqrt{x^2_{max} + y^2_{max}}}{\rho_{max}}\left(-\rho_{clang} + \frac{\rho_{max}}{2} - 1\right)$
ρθ座標から MATLABρθ配列への変換	**ρθ座標から C 言語ρθ配列への変換**
$\theta_{matlab} = \frac{\theta_{max}}{\pi}\theta + 1$ $\rho_{matlab} = -\frac{\rho_{max}}{\sqrt{x^2_{max} + y^2_{max}}}\rho + \frac{\rho_{max}}{2}$	$\theta_{clang} = \frac{\theta_{max}}{\pi}\theta$ $\rho_{clang} = -\frac{\rho_{max}}{\sqrt{x^2_{max} + y^2_{max}}}\rho + \frac{\rho_{max}}{2} - 1$



となる。
同様に、MATLAB での xy インデックスと画像座標の関係を整理すると、
xy 座標と xy 配列の関係も整理すると

MATLAB xy 配列	C 言語 xy 配列	xy 座標
(1,1)	(0,0)	$\left(-\frac{x_{max}}{2} \quad \frac{y_{max}}{2}-1\right)$
$(x_{max} \quad y_{max})$	$(x_{max}-1 \quad y_{max}-1)$	$\left(\frac{x_{max}}{2}-1 \quad -\frac{y_{max}}{2}\right)$



の対応付けをすればよい。この時、y軸が逆転しているのでθの値に注意する必要がある。
つまり、それぞれ xy の関係は、

MATLAB xy 配列から xy 座標への変換	C 言語 xy 配列から xy 座標への変換
$x = x_{matlab} - 1 - \frac{x_{max}}{2}$ $y = -y_{matlab} + \frac{y_{max}}{2}$	$x = x_{clang} - \frac{x_{max}}{2}$ $y = -y_{clang} + \frac{y_{max}}{2} - 1$
xy 座標から MATLAB xy 配列への変換	xy 座標から C 言語 xy 配列への変換
逆変換は、$x_{matlab} = x + \frac{x_{max}}{2} + 1$ 逆変換は、$y_{matlab} = -y + \frac{y_{max}}{2}$	$x_{clang} = x + \frac{x_{max}}{2}$ $y_{clang} = -y - 1 + \frac{y_{max}}{2}$

MATLAB xy 配列から MATLAB　ρ配列への変換

$$\rho = \left(x_{matlab} - 1 - \frac{x_{max}}{2}\right)\cos\theta + \left(-y_{matlab} + \frac{y_{max}}{2}\right)\sin\theta$$

$$\rho_{matlab} = \frac{\rho_{max}}{2} - \frac{\rho_{max}}{\sqrt{x_{max}^2 + y_{max}^2}}\left(\left(x_{matlab} - 1 - \frac{x_{max}}{2}\right)\cos\theta + \left(-y_{matlab} + \frac{y_{max}}{2}\right)\sin\theta\right)$$

となる。
x,y 配列の計算は、

$$x_{matlab} = \frac{x_{max}}{2} + \frac{\rho}{\cos\theta} - \left(-y_{matlab} + \frac{y_{max}}{2}\right)\frac{\sin\theta}{\cos\theta} + 1$$

$$y_{matlab} = \frac{y_{max}}{2} - \frac{\rho}{\sin\theta} + \left(x_{matlab} - 1 - \frac{x_{max}}{2} \right) \frac{\cos\theta}{\sin\theta}$$

となる。
C 言語では、

$$\rho = \left(x_{cleng} - \frac{x_{max}}{2} \right) \cos\theta + \left(-y_{clang} - 1 + \frac{y_{max}}{2} \right) \sin\theta$$

$$\rho_{clang} = \frac{\rho_{max}}{2} - \frac{\rho_{max}}{\sqrt{x_{max}^2 + y_{max}^2}} \left(\left(x_{cleng} - \frac{x_{max}}{2} \right) \cos\theta + \left(\frac{y_{max}}{2} - y_{clang} - 1 \right) \sin\theta \right) - 1$$

逆変換用の式は、

$$x_{clang} = \frac{x_{max}}{2} + \frac{\rho}{\cos\theta} - \left(-y_{clang} - 1 + \frac{y_{max}}{2} \right) \frac{\sin\theta}{\cos\theta}$$

$$y_{clang} = \frac{y_{max}}{2} - \frac{\rho}{\sin\theta} + \left(x_{clang} - \frac{x_{max}}{2} \right) \frac{\cos\theta}{\sin\theta} - 1$$

画像のサイズ xmax,ymax
Hough 変換のパラメータの解像度　rho, theta

rho,thetax の求め方

例えば、 [rhoindex,thetaindex]=find(hbw==max(hbw(:)));
により最大値の配列
rhoindex,thetaindex と実際の rho,theta の関係は、

$\theta = \frac{\pi}{\theta_{max}} (\theta_{matlab} - 1)$ 、 $\rho = \frac{\sqrt{x_{max}^2 + y_{max}^2}}{\rho_{max}} \left(\frac{\rho_{max}}{2} - \rho_{matlab} \right)$ で計算できる。

計算した rho,theta から画像中に直線を引くには、
もし、sin θ が 0 であるならば、

$$y_{matlab} = -\frac{y_{max}}{2} \sim \frac{y_{max}}{2} - 1$$

として、

$$x_{matlab} = \frac{\rho}{\cos\theta} - \left(-y_{matlab} + \frac{y_{max}}{2} \right) \frac{\sin\theta}{\cos\theta} + \frac{x_{max}}{2} + 1$$

そうでないならば、

$$x_{matlab} = -\frac{x_{max}}{2} \sim \frac{x_{max}}{2} - 1$$

$$y_{matlab} = \frac{y_{max}}{2} - \frac{\rho}{\sin\theta} + \left(x_{matlab} - 1 - \frac{x_{max}}{2} \right) \frac{\cos\theta}{\sin\theta}$$

として計算すれば、画像とオーバーラップ可能になる。

MATLAB で使う場合

HOUGH 変換を行う場合、画像サイズ xmax,ymax この場合、横が x サイズ、縦が y サイズそれに、ρ、θの分解能 rhomax,thetamax を入れる。
size 関数でサイズを検出する場合には、
[ymax,xmax]=size(bw)と、順序が異なるので注意する必要がある。

```
xmax=160;
ymax=260;
rhomax=100;
thetamax=360;
xymax=sqrt(xmax*xmax+ymax*ymax);
bw=zeros(ymax,xmax);
bw(:,ymax/2)=1; % means rho=0;
bw=imrotate(bw,55,'crop');% means theta=55deg
hbw=houghf(bw,rhomax,thetamax);
[rhoindex,thetaindex]=find(hbw==max(hbw(:)));
rho=(rhomax/2-rhoindex)*xymax/rhomax
theta=pi/thetamax*(thetaindex-1)
rho=rho(1);theta=theta(1);theta*180/pi
%%%%%%%%%%%%%%%%%%%%%%%
xvec=rho/cos(theta)-(-[-xymax xymax]+ymax/2)*sin(theta)/cos(theta)+xmax/2+1;
yvec=ymax/2-rho/sin(theta)+([-xymax xymax]-xmax/2-1)*cos(theta)/sin(theta);
imshow(bw);
hold on
plot(xvec,[-xymax xymax]);
plot([-xymax xymax],yvec);
plot(xvec,[-xymax xymax]);
hold off
```

ここで theta は、回転角度で単位は radian。　rho は、画像中心からの距離を示している。
xmax,ymax は任意の大きさの画像を意味し、rhomax,thetamax は、それぞれどれだけの分解能を望むかにより決定される。

```
% bugfixed for houghf.c version 2002.10.05 http://www.gerox.com
% version up can apply non square image　version 2004.02.07 http://www.gerox.com
xmax=30;
ymax=50;
rhomax=200;
thetamax=180;
bw=eye(ymax,xmax);
xymax=sqrt(xmax^2+ymax^2);
bw=imrotate(bw,60,'crop');
close all;
figure(1)
imshow(bw)
hold on;
hline1=plot([1 1],[-xymax xymax],'linewidth',2);
hold off
drawnow;
c=houghf(bw,rhomax,thetamax);
figure(2)
imshow(c,[ ]);axis square;
set(2,'windowbuttondown',['aaa=get(gca,''currentpoint'');',...
'theta = pi/thetamax*(aaa(1,1)-1);',...
```

```
'rho = (rhomax/2-aaa(1,2))*xymax/rhomax;',...
'x=(rho/cos(theta)-(-[-xymax xymax]+ymax/2)*sin(theta)/cos(theta))+xmax/2+1;',...
'set(hline1,"Xdata",x);',...
'drawnow;']);
set(1,'doublebuffer','on');
set(2,'doublebuffer','on');
```

ρ θ座標→matline→MATLAB xy 配列→
houghf(C 言語　xy 配列→変換→xy 座標→変換→ρ θ座標→変換→C 言語　ρ θ配列)→
MATLAB ρ θ配列→変換式→ρ θ座標→変換式→xy 座標

IHOUGH テスト houghdemo0.m

```
% houghtransform sample program by Gerox (c) 2004
close all;
clear all
xmax=80;
ymax=160;
rhomax=1000;
thetamax=180;
%%%%%%%%%Define line parameter%%%%%%%%%%%%%%%%%%
rho=20;
theta=60*pi/180;
%%%%%%%%%make image%%%%%%%%%%%%%%%%%%
bw=zeros(ymax,xmax);
if(abs(sin(theta))>0.01)
   x=[-xmax/2 xmax/2-1];
   y=rho/sin(theta)-x*cos(theta)/sin(theta);
   xindex=x+1+xmax/2;
   yindex=round(ymax/2-rho/sin(theta)+(xindex-1-xmax/2)*cos(theta)/sin(theta));
else
   y=[ymax/2-1 -ymax/2];
   x=rho/cos(theta)-y*sin(theta)/cos(theta);
   yindex=-y+ymax/2;
   xindex=round(rho/cos(theta)-(ymax/2-yindex)*sin(theta)/cos(theta)+xmax/2+1);
end
bw=matline(bw,yindex(1),xindex(1),yindex(2),xindex(2));
subplot(2,2,1);imshow(bw);
title(['Original ¥rho=',num2str(rho),' ¥theta=',num2str(theta*180/pi)]);
%%%%%%%%Hough transform %%%%%%%%%%%%%%%%%%%%%%%%%%
hbw=houghf(bw,rhomax,thetamax);
%%%%%%%%Calculate rho and theta from Hough pace %%%%%%%%%%%%%%%
[rhoindex,thetaindex]=find(hbw==max(hbw(:)));
rhoindex=mean(rhoindex);
thetaindex=mean(thetaindex);
rho=(rhomax/2-rhoindex)*sqrt(xmax*xmax+ymax*ymax)/rhomax
theta=pi/thetamax*(thetaindex-1)
[rho theta*180/pi]
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
subplot(2,2,3);imshow(hbw,[]);
title('Hough space image');ylabel('¥rho');xlabel('¥theta');
```

```
%%%%%%%Reconstruct line from calculated rho and theta %%%%%%%%%%%%
bw2=zeros(ymax,xmax);
if(abs(sin(theta))>0.01)
   x=[-xmax/2 xmax/2-1];
   y=rho/sin(theta)-x*cos(theta)/sin(theta);
   xindex=x+1+xmax/2;
   yindex=round(ymax/2-rho/sin(theta)+(xindex-1-xmax/2)*cos(theta)/sin(theta));
else
   y=[ymax/2-1 -ymax/2];
   x=rho/cos(theta)-y*sin(theta)/cos(theta);
   yindex=-y+ymax/2;
   xindex=round(rho/cos(theta)-(ymax/2-yindex)*sin(theta)/cos(theta)+xmax/2+1);
end
bw2=matline(bw2,yindex(1),xindex(1),yindex(2),xindex(2));
subplot(2,2,4);imshow(bw2);
title(['Ihough ¥rho=',num2str(rho),' ¥theta=',num2str(theta*180/pi)]);
subplot(2,2,2);imshow(bw);title('Overlapped image');
hold on
plot(xindex,yindex,'linewidth',5);
hold off
```

直線の作成には、MATLINE 関数を使用

```
% create line in matrix data matline(xyarea(x,y),x1,y1,x2,y2)
function xyarea = matline(xyarea,x1,y1,x2,y2)
for i=1:length(x1)
   xysize = size(xyarea);
   dx = abs(x2(i)-x1(i));
   dy = abs(y2(i)-y1(i));
   if(dx > dy)
      if(x1(i) > x2(i))
        step =sign((y1(i) > y2(i))-0.5);
        s=x1(i);x1(i)=x2(i);x2(i)=s;y1(i)=y2(i);
      else
         step = sign((y1(i) < y2(i)) -0.5);
      end
      if min(xysize >= [x1(i) y1(i)]) & min([x1(i) y1(i)])>0, xyarea(x1(i),y1(i)) = 1;end
      s = floor(dx / 2);
      while (x1(i) <= x2(i))
         x1(i) = x1(i) + 1;
         s = s - dy;
         if s < 0
            s = s + dx;
            y1 = y1 + step;
         end
         if min(xysize >= [x1(i) y1(i)]) & min([x1(i) y1(i)])>0, xyarea(x1(i),y1(i)) = 1;end
      end
   else
      if(y1(i) > y2(i))
         step =sign((x1(i) > x2(i))-0.5);
         s=y1(i);y1(i)=y2(i);y2(i)=s;x1(i)=x2(i);
      else
```

```
      step = sign((x1(i)<x2(i))-0.5);
    end
    if min(xysize >= [x1(i) y1(i)]) & min([x1(i) y1(i)])>0, xyarea(x1(i),y1(i)) = 1;end
    s = floor(dy / 2);
    while (y1(i) <= y2(i))
      y1(i) = y1(i) + 1;
      s = s - dx;
      if s < 0
        s = s + dy;
        x1(i) = x1(i) + step;
      end
      if min(xysize >= [x1(i) y1(i)]) & min([x1(i) y1(i)])>0, xyarea(x1(i),y1(i)) = 1;end
    end
  end
end
```

houghf.c

```
#include <stdio.h>
#include <math.h>
#include "mex.h"
/*
function res=hough(im,rhomax,thetamax)
% Name:          hough(im,RHO_MAX,THETA_MAX)
% Version:     v1.0
% Author:      Dimitrios Ioannou
%         dioan@robotsg.nuceng.ufl.edu
%         dioan@alder.circa.ufl.edu
% Date:        08/23/95
% Arguments:
%         im: is the input,binary, image. If the
%         image is not binary pixels having
%         non-zero values are considered.
%         RHO_MAX: is an integer number specifying
%         the rho quantization.
%         THETA_MAX: is an integer number
%         specifying the theta quantization
% Purpose:
%         perform the hough transform of a binary
%         image
% Dependencies:
%         None
% Example:       v=hough(im,256,256)
%         input is the image im, and the
%         quantization is d_rho=X/256 and d_theta=pi/256
%         if the size of the image is 256 by 256
%         d_rho=1.
%
%         v is the number of votes in the
%         paremeter space. v(i,j) is the number
%         of votes for the strip having distance from
%         the center of the image equal to
%         (i-RHO_MAX/2)*d_rho (d_rho=X/RHO_MAX, the
%         image is X by X  pixels),and its normal has
%         angle j*d_theta,(d_theta=pi/THETA_MAX)
%
%         for a 256 by 256 image, the center of the
%         image is the center of the pixel (128,128)
%         i=1 => rho=(i-1-128)*d_rho=-128*d_rho
%         i=256 => rho=(i-1-128)*d_rho=127*d_rho
%         this essentially means that:
%         'the image is not symmetric around its center'.
%
```

```
Arranged by Gerox(c) for mex file 1999
Arranged by Gerox(c) for sparse matrix 1999.08.22
Arranged by Gerox(c) for uint8 matrix 1999.08.25
Optimaized for speed & bug fix by Gerox(c) 2002.10.05
Now can be apply non-square image by Gerox(c) 2004.02.07
------------------
close all;
data=zeros(50,50);
data(10,:)=1;
figure(1);set(1,'doublebuffer','on');
tdat1=[ ];tdat2=[ ];
for theta=1:5:180
    data1=imrotate(data,theta,'nearest','crop');
    tic;hout=houghf(data1,50,50);tdat1=[tdat1 toc];
    subplot(2,2,1);imagesc(data1);axis square
    subplot(2,2,2);imagesc(hout);axis square
    tic;hout=houghf(data1,50,50);tdat2=[tdat2 toc];
    subplot(2,2,3);imagesc(data1);axis square
    subplot(2,2,4);imagesc(hout);axis square
    drawnow;
end
mean(tdat1),mean(tdat2)
------------------
*/
void mexFunction(int nlhs,mxArray *plhs[],int nrhs,const mxArray *prhs[]) {
    int i,ii,jj;
    int theta,rho;
    int rhomax,thetamax,X,Y;
    double pi = 3.14159265358979;
    double *im;
    double d_theta;
    int *smat,*cmat;
    double dd_rho,tmp;
    double rhomax_2;
    int X_2,Y_2;
    double *yy;
    int jjY;
    double jjX_2;
    double iiY_2;
    if ((nrhs != 3) || (nlhs != 1)) {
        mexPrintf("function res=hough(im,RHO_MAX,THETA_MAX)¥n");
        mexPrintf("Based on m-script file (Author:Dimitrios Ioannou: hough.m v1.0)¥n");
        mexPrintf(" which is available from www.mathworks.com under user contributed library¥n");
        mexPrintf("Arranged for MEX version by Gerox(c) Copyright 1999 %s¥n",__DATE__);
        mexErrMsgTxt("¥n");
    }
    rhomax = (int)mxGetScalar(prhs[1]);
    thetamax = (int)mxGetScalar(prhs[2]);
    //[M,N]=size(im);
    X = (int)mxGetN(prhs[0]);
    Y = (int)mxGetM(prhs[0]);
//  if (X != Y) {
//      mexPrintf("Input image is not square. Exiting!¥n");
//      return;
//  } else {
        if ((X % 2) == 1) {
            mexPrintf("Input image size has to be even in pixels. Exiting!¥n");
            return;
        }
        if ((Y % 2) == 1) {
            mexPrintf("Input image size has to be even in pixels. Exiting!¥n");
            return;
        }
//  }
    im = mxGetPr(prhs[0]);
//  Since we need theta from 0 to pi
//  check result of
//  mesh(hough(ones(120,120),120,120));
```

```
//  mesh(houghf(ones(120,120),120,120));
    d_theta = pi / (double)thetamax;
//  theta=0:d_theta:pi-d_theta;
    smat = (int*)mxCalloc(thetamax,sizeof(int));
    cmat = (int*)mxCalloc(thetamax,sizeof(int));
    dd_rho = (double)rhomax/ (double)sqrt(X*X+Y*Y);
    for (i = 0,tmp=0.0;i < thetamax;tmp += d_theta,i++) {
        smat[i] = (int)(sin(tmp) * dd_rho * (1 << 16));
        cmat[i] = (int)(cos(tmp) * dd_rho * (1 << 16));
    }
//  [x,y]=find(im);
    rhomax_2 = (double)(rhomax >> 1);
    X_2 = (int)(X >> 1);
    Y_2 = (int)(Y >> 1);
    plhs[0] = mxCreateDoubleMatrix(rhomax,thetamax,mxREAL);
    yy = mxGetPr(plhs[0]);
    if(mxIsDouble(prhs[0])) {
        for(jj = 0;jj < X;jj++) {
            jjY = jj * Y;
            jjX_2 = (double)(jj - X_2);
            for (ii = 0;ii < Y;ii++) {
                if(im[jjY + ii] > 0) {
                    iiY_2 = (double)(Y_2-ii-1);
                    for(theta = 0;theta < thetamax;theta++) {
                        rho = (int)(rhomax_2-1-((int)(jjX_2 * cmat[theta] +iiY_2 * smat[theta])>>16));
                        if((rho < 0) || (rho > rhomax-1)) continue;
                        yy[theta * rhomax + rho] += 1.0;
                    }
                }
            }
        }
    } else {
        int *ir;
        int *jc;
        int cnt;
        int col;
        if(mxIsSparse(prhs[0])) {
//          [x,y]=find(im);
            ir = mxGetIr(prhs[0]);
            jc = mxGetJc(prhs[0]);
            for (col = 0;col < X;col++) {
                int start = jc[col];
                int stop = jc[col+1];
                if (start == stop) continue;
                jjX_2 = (double)(col - X_2);
                for (cnt = start; cnt < stop; cnt++)  {
//                  mexPrintf("¥t(%d,%d) = %g¥n", ir[cnt]+1, col+1, data[total++]);
                    iiY_2 = (double)(-ir[cnt] + Y_2 - 1);
                    for(theta = 0;theta < thetamax;theta++) {
                        rho = (int)(rhomax_2 - 1 - ((int)(jjX_2 * cmat[theta] + iiY_2 * smat[theta])>>16));
                        if((rho < 0) || (rho > rhomax-1)) continue;
                        yy[theta * rhomax + rho] += 1.0;
                    }
                }
            }
        } else {
            unsigned char *imi = (unsigned char*)mxGetPr(prhs[0]);
            if(mxIsUint8(prhs[0])) {
                for(jj = 0;jj < X;jj++) {
                    jjY = jj * Y;
                    jjX_2 = (double)(jj - X_2);
                    for (ii = 0;ii < Y;ii++) {
                        if(imi[jjY + ii] > 0) {
                            iiY_2 = (double)(Y_2 - ii - 1);
                            for(theta = 0;theta < thetamax;theta++) {
                                rho = (int)(rhomax_2 - 1 - ((int)(jjX_2 * cmat[theta] + iiY_2 * smat[theta])>>16));
                                if((rho < 0) || (rho > rhomax-1)) continue;
                                yy[theta * rhomax + rho] += 1.0;
```

```
                        }
                    }
                }
            }
        } else {
            mxFree(smat);
            mxFree(cmat);
            mexErrMsgTxt("Input must be double or sparse or uint8 Matrix");
        }
    }
  }
  mxFree(smat);
  mxFree(cmat);
}
```