Changxing Cao

(1) convert them into grayscale images

convert them into grayscale images by using the formula luminance = 0.30R + 0.59G + 0.11B

```
function [gray]=grayimg(img)
%img=imread('TestImage1c.jpg');
R=double(img(:,:,1));
G=double(img(:,:,2));
B=double(img(:,:,3));
gray=0.30*R + 0.59*G + 0.11*B;
%fprintf('%f', max(max(gray3)));
%gray=double(gray3)/double(max(max(gray3)))*255.0;
```

detect edges using the Sobel's operator,Sobel operator:

$$Gx=\begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix}*A$$

$$Gy=\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix}*A$$

$$G=\sqrt{Gx^2+Gy^2}$$

Code:

```
clear;
sourcePic=imread('TestImage3.jpg');
%grayPic=mat2gray(sourcePic);

%have zrayl
%newGrayPic=rgb2gray(sourcePic); %convert 3D - 2D
grayPic=grayimg(sourcePic);
%figure;
%imshow(grayPic);
[m,n]=size(grayPic);
newGrayPic=zeros(m,n);
[m,n]=size(newGrayPic);
%newGrayPic2=rgb2gray(sourcePic);
sobelNum=0;
sobelThreshold=40; %1 40; 2 70-80; 3 40
for j=2:m-1
    for k=2:n-1
        sobelNum=sqrt((grayPic(j-1,k+1)+2*grayPic(j,k+1) ...
            +grayPic(j+1,k+1)-grayPic(j-1,k-1)-2*grayPic(j,k-1)-grayPic(j+1,k-1))^2+ ...
           (grayPic(j-1,k-1)+2*grayPic(j-1,k)+grayPic(j-1,k+1)-grayPic(j+1,k-1) ...
           -2*grayPic(j+1,k)-grayPic(j+1,k+1))^2);
        if(sobelNum > sobelThreshold)
            newGrayPic(j,k)=255;
        else
            newGrayPic(j,k)=0;
        end
    end
end
figure;
imshow(newGrayPic)
title('Sobel???????')
```

In order to detect accurate straight lines, we improve the detection with **canny** and improve canny with Sobel operator instead of [-1,1;-1,1].

Code:

```
function [ m, theta, sector, canny1,  canny2, bin] = canny1step( src,  lowTh , highTh)
[ Ay,  Ax, dim ] = size(src);
if dim>1
    src = rgb2gray(src);
end
src = double(src);
m = zeros( Ay,  Ax);
theta = zeros( Ay,  Ax);
sector = zeros( Ay,  Ax);
canny1 = zeros( Ay,  Ax); %??????
canny2 = zeros( Ay,  Ax); %????????
bin = zeros( Ay,  Ax);
for y = 2:( Ay-1)
    for x = 2:( Ax-1)
        gx =  src( y-1,  x+1)  + 2*src( y, x+1)  + src( y+1,  x+1)  - ...
            src( y-1,  x-1)  -  2*src( y, x-1)  - src( y+1,  x-1);
        gy =  src( y+1,  x-1)  + 2*src( y+1, x)  + src( y+1, x+1) ...
            - src( y-1,  x-1)  - 2*src( y-1, x)  - src( y-1, x+1);
        m( y, x)  = ( gx^2+gy^2)^0.5 ;
        %-------------------------------
        theta( y, x)  = atand( gx/ gy)   ;
        tem = theta( y, x);
        %-------------------------------
        if (tem<67.5) &&(tem>22.5)
            sector( y, x)  =  0;
        elseif (tem<22.5) &&(tem>-22.5)
            sector( y, x)  =  3;
        elseif (tem<-22.5) &&(tem>-67.5)
            sector( y, x)  =   2;
        else
            sector( y, x)  =   1;
        end
        %-------------------------------
    end
end
%------------------------
%??????
%------> x
%   2 1 0
%   3 X 3
%y  0 1 2
for y = 2:( Ay-1)
    for x = 2:( Ax-1)
        if sector( y, x)==0  %?? - ??
            if (  m( y, x) > m( y-1, x+1)  ) &&(   m( y, x) > m( y+1, x-1)   )
                canny1( y, x)  = m( y, x);
            else
                canny1( y, x)  = 0;
            end
        elseif sector( y, x)==1   %????
            if (  m( y, x) > m( y-1, x)  ) &&(   m( y, x) > m( y+1, x)   )
                canny1( y, x)  = m( y, x);
            else
                canny1( y, x)  = 0;
            end
        elseif sector( y, x)==2  %?? - ??
            if (  m( y, x) > m( y-1, x-1)  ) &&(   m( y, x) > m( y+1, x+1)   )
                canny1( y, x)  = m( y, x);
            else
                canny1( y, x)  = 0;
            end
        elseif sector( y, x)==3  %???
            if (  m( y, x) > m( y, x+1)  ) &&(   m( y, x) > m( y, x-1)   )
                canny1( y, x)  = m( y, x);
            else
                canny1( y, x)  = 0;
            end
        end
    end %end for x
end %end for y
```

```
for y = 2:(Ay-1)
    for x = 2:(Ax-1)
        if canny1(y,x)<lowTh  %?????
            canny2(y,x) = 0;
            bin(y,x) = 0;
            continue;
        elseif canny1(y,x)>highTh  %?????
            canny2(y,x) = canny1(y,x);
            bin(y,x) = 1;
            continue;
        else  %???????8???????????????????
            tem =[canny1(y-1,x-1), canny1(y-1,x), canny1(y-1,x+1);
                     canny1(y,x-1),    canny1(y,x),   canny1(y,x+1);
                     canny1(y+1,x-1), canny1(y+1,x), canny1(y+1,x+1)];
            temMax = max(tem);
            if temMax(1) > highTh
                canny2(y,x) =temMax(1);
                bin(y,x) = 1;
                continue;
            else
                canny2(y,x) = 0;
                bin(y,x) = 0;
                continue;
            end
        end
    end %end for x
end %end for y
end %end of function
```

(2) detect straight line segments using the Hough Transform

Hough Transform

Create (-90º 90º)*(0, rho) space with 0 value, where rho is the sqrt(x2+y2).

Get non-0 (edge) data of the image after edge detection, get the (x,y)

Code:

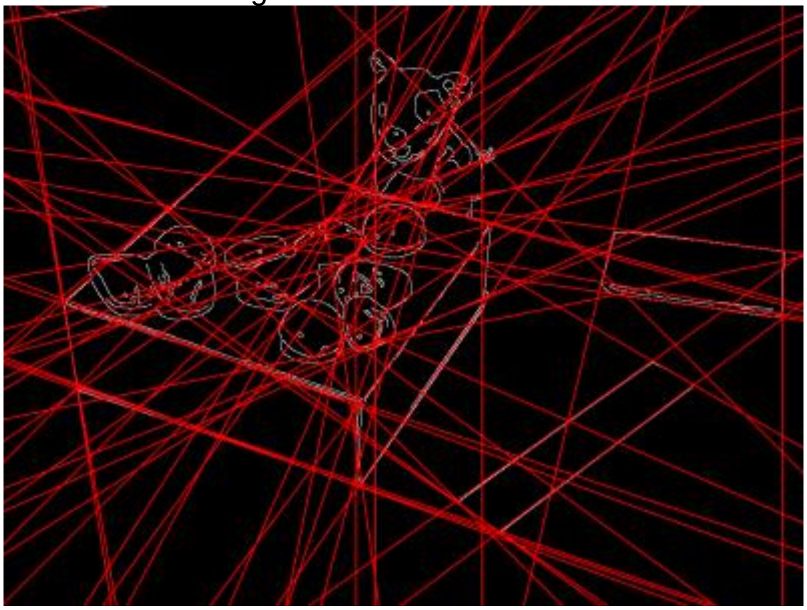
```
function [h, theta, rho] = houghTs(f, dtheta, drho)
if nargin < 3
   drho = 1;
end
if nargin < 2
   dtheta = 1;
end

f = double(f);
[M,N] = size(f);
theta =linspace(-90, 0, ceil(90/dtheta) + 1);
theta =[theta -fliplr(theta(2:end - 1))];
ntheta =length(theta);

D = sqrt((M- 1)^2 +(N- 1)^2);
q =ceil(D/drho);
nrho = 2*q - 1;
rho =linspace(-q*drho, q*drho, nrho);

[x, y, val] =find(f);
x = x - 1;  y = y - 1;

% Initialize output.
h = zeros(nrho, length(theta));

% To avoid excessive memory usage, process 1000 nonzero pixel
% values at a time.
for k = 1:ceil(length(val)/1000)
   first =(k - 1)*1000 + 1;
   last  = min(first+999, length(x));

   x_matrix     =repmat(x(first:last), 1, ntheta);
   y_matrix     =repmat(y(first:last), 1, ntheta);
   val_matrix   =repmat(val(first:last), 1, ntheta);
   theta_matrix =repmat(theta, size(x_matrix, 1), 1)*pi/180;

   rho_matrix = x_matrix.*cos(theta_matrix) + ...
      y_matrix.*sin(theta_matrix);
   slope =(nrho - 1)/(rho(end) - rho(1));
   rho_bin_index =round(slope*(rho_matrix - rho(1)) + 1);

   theta_bin_index =repmat(1:ntheta, size(x_matrix, 1), 1);

   h = h + full(sparse(rho_bin_index(:), theta_bin_index(:), ...
```

Hough Peak:

Find the max(x,y) in certain local region. Eg,(7*7);

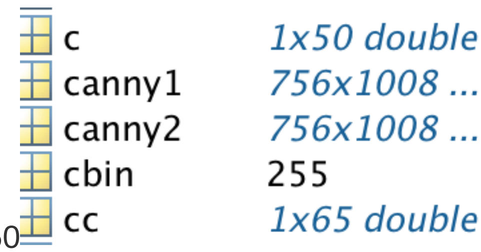
```
function [r, c, hnew] = hpeak(h, nhood)
if nargin < 2
   nhood =size(h)/50;
   % Make sure the neighborhood size is odd.
   nhood = max(2*ceil(nhood/2) + 1, 1);
end

hmax=max(h(:));
hnew = h; r =[]; c =[];
[hm, hn]=size(h);
localmaxa=80; %2nd 80; 3rd 5
localmaxb=10; %2nd 10; 3rd 5
for ha=1+localmaxa:localmaxa*2:hm-localmaxa
   for hb=1+localmaxb:localmaxb*2:hn-localmaxb

      hnewp=hnew(ha-localmaxa:ha+localmaxa, hb-localmaxb:hb+localmaxb);
      if(max(hnewp(:))~=0 &&max(hnewp(:))>=hmax*0.2) %2nd 0.2 3rd0.15
         [p,q]=find(hnewp==max(hnewp(:)));
      p=p+ha-localmaxa-1;
      q=q+hb-localmaxb-1;
      p=p(1);q=q(1);
      r(end+1)=p;c(end+1)=q;
      end
```

The result of straight lines detection:

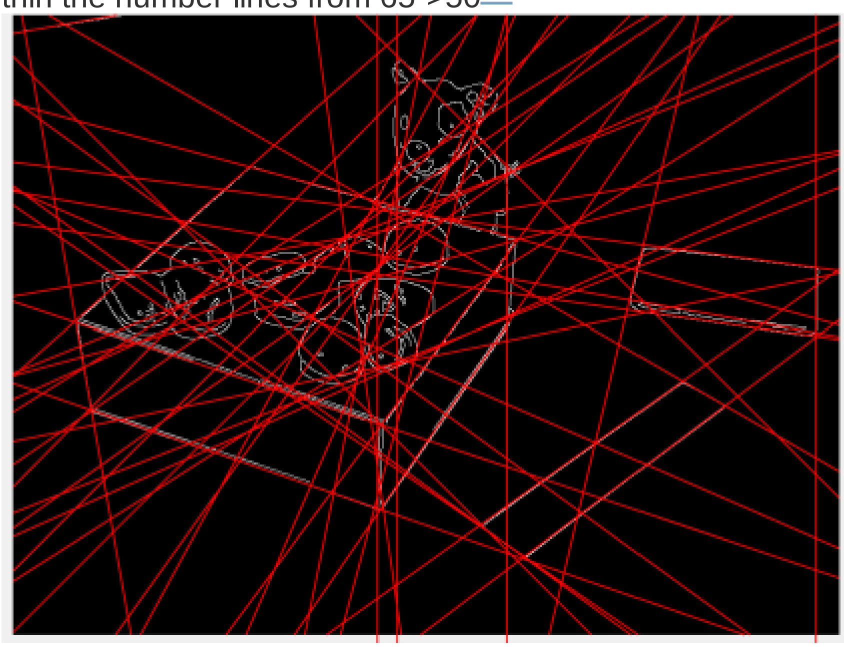


Then thin the number of lines:

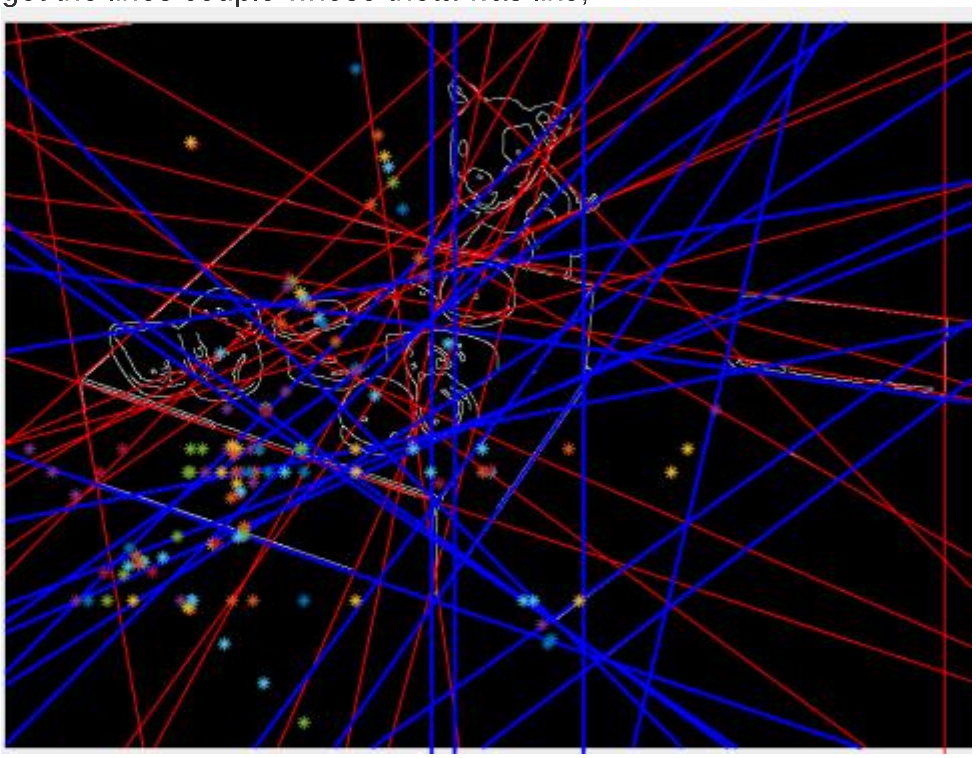
EG:



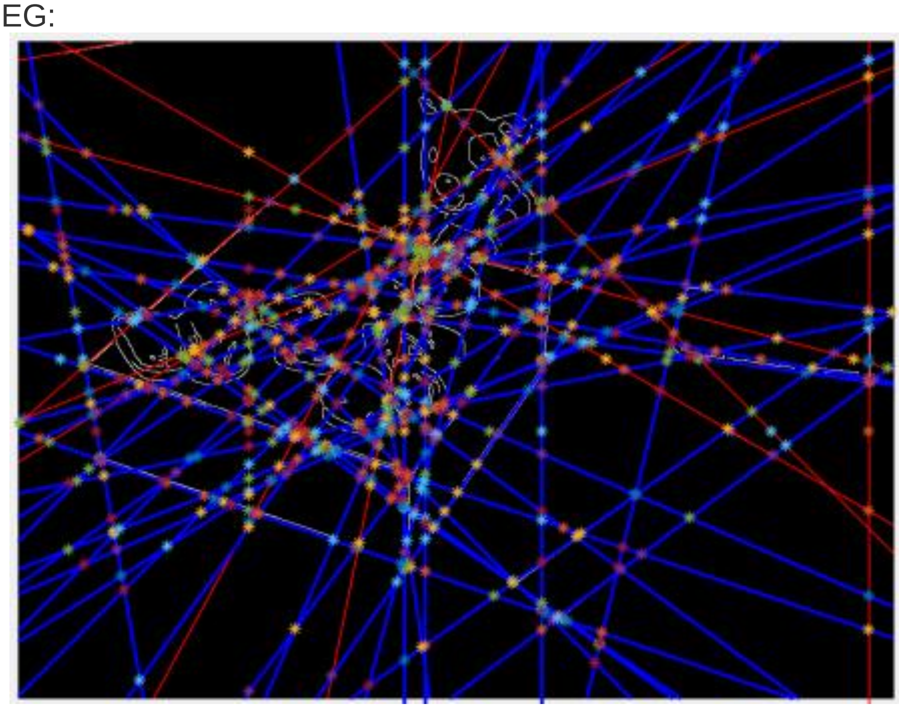
thin the number lines from 65->50



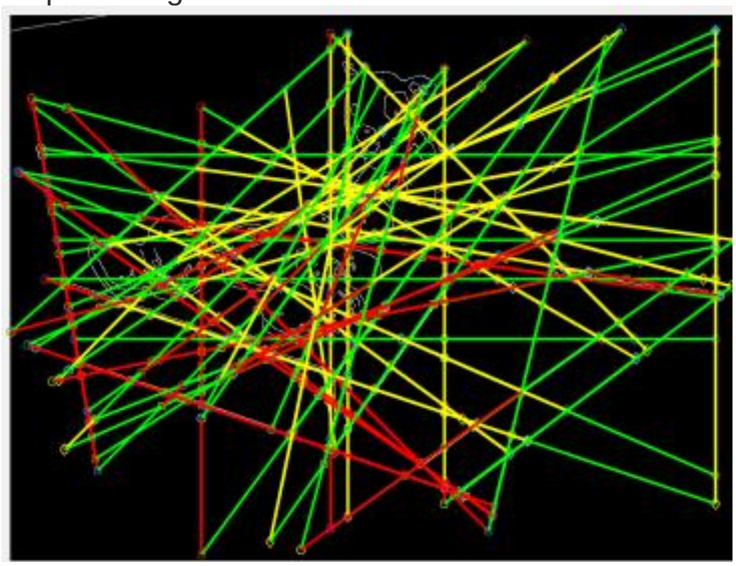
get the lines couple whose theta was like,



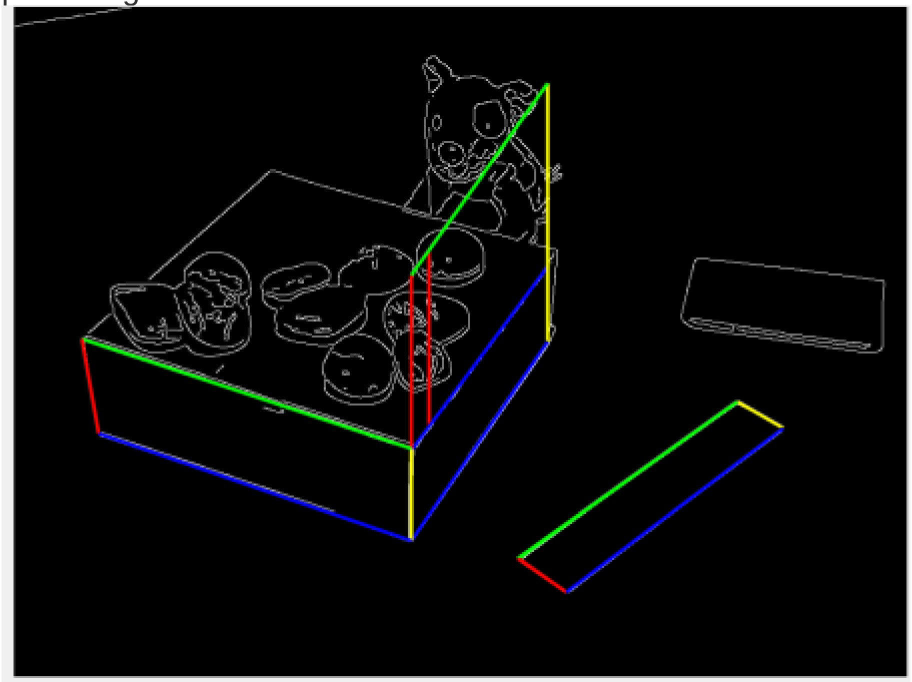
And get the lines couple's points where they cross each other.
Calculate the function of two variable equation
Detect point of parallelograms
EG:



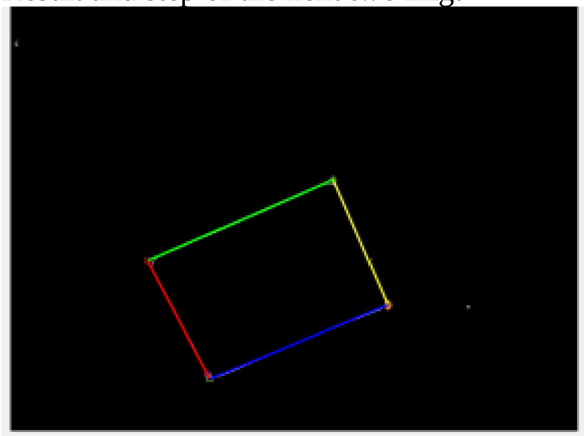
(3) detect parallelograms from the straight-line segments detected in step (2).
Firstly, determine pairs of parallel lines and from the parallel lines determine candidate parallelograms by computing the lines' intersection points.
Then compute the number of edge points (in percentage) that are present on each of the four sides of the candidate parallelograms; if the percentage is high, it is a parallelogram.
All parallelogram:



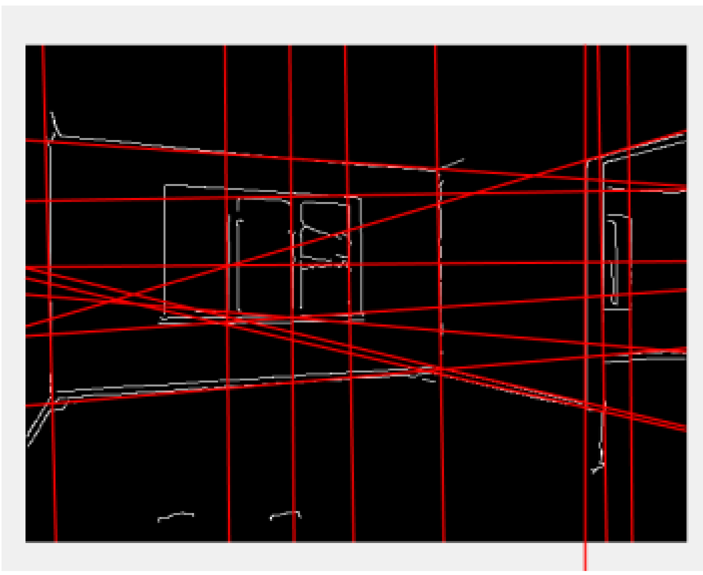
parallelogram that is accurate:



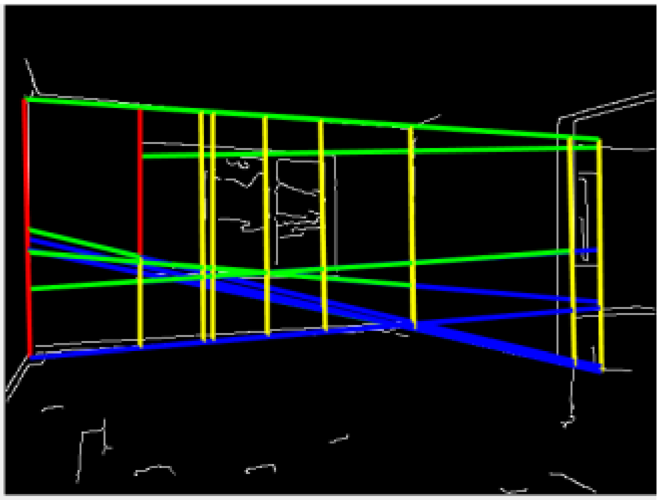
Result and step of the next two img:



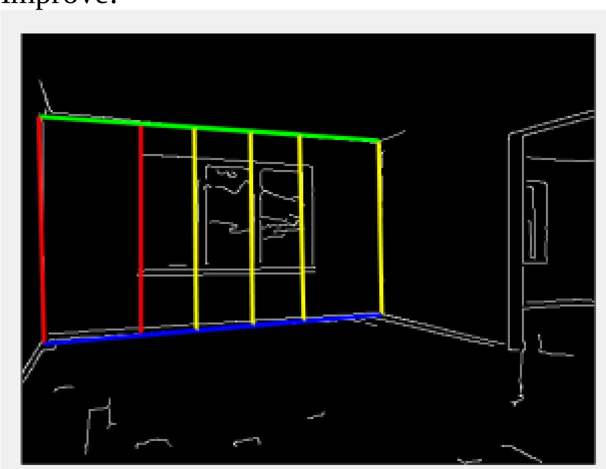
1.straight lines:



2.parallelogram:



Improve:



Extra Credits (10 points). This part is optional.


Figure 1
Fi Ec Vi Ins To Desl Win He

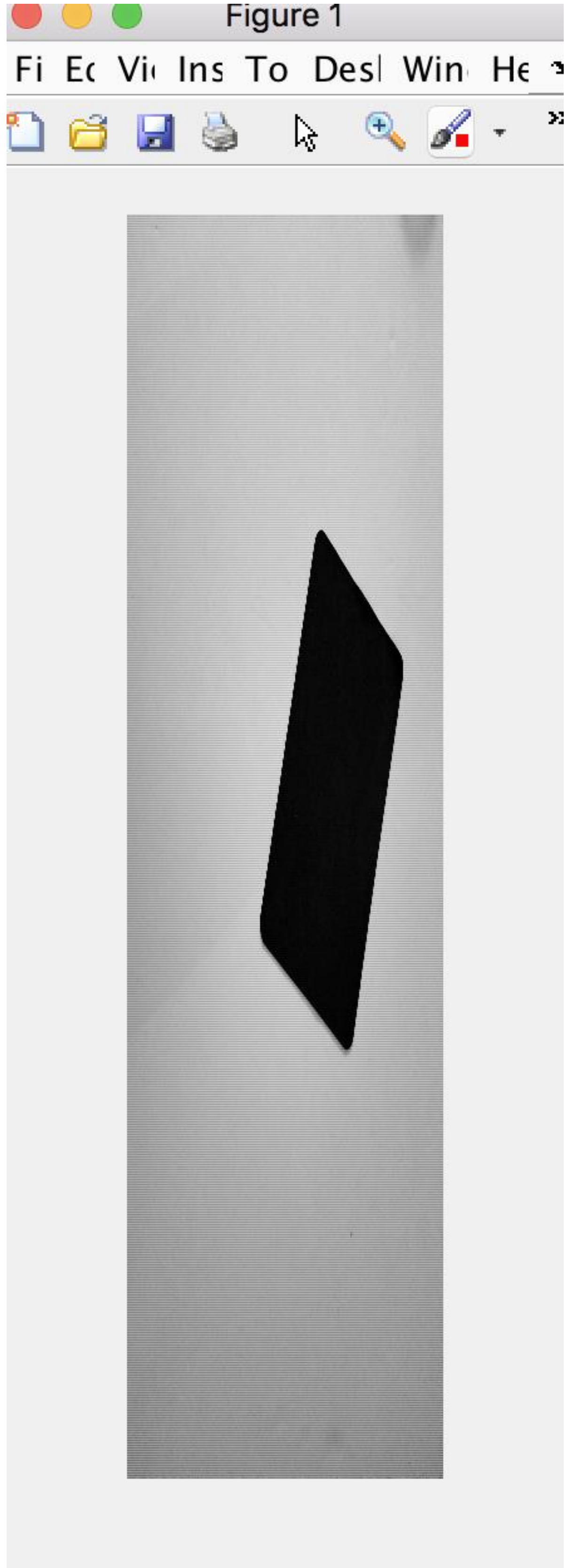


Figure 1
Fi Ec Vi Ins To Desl Win He

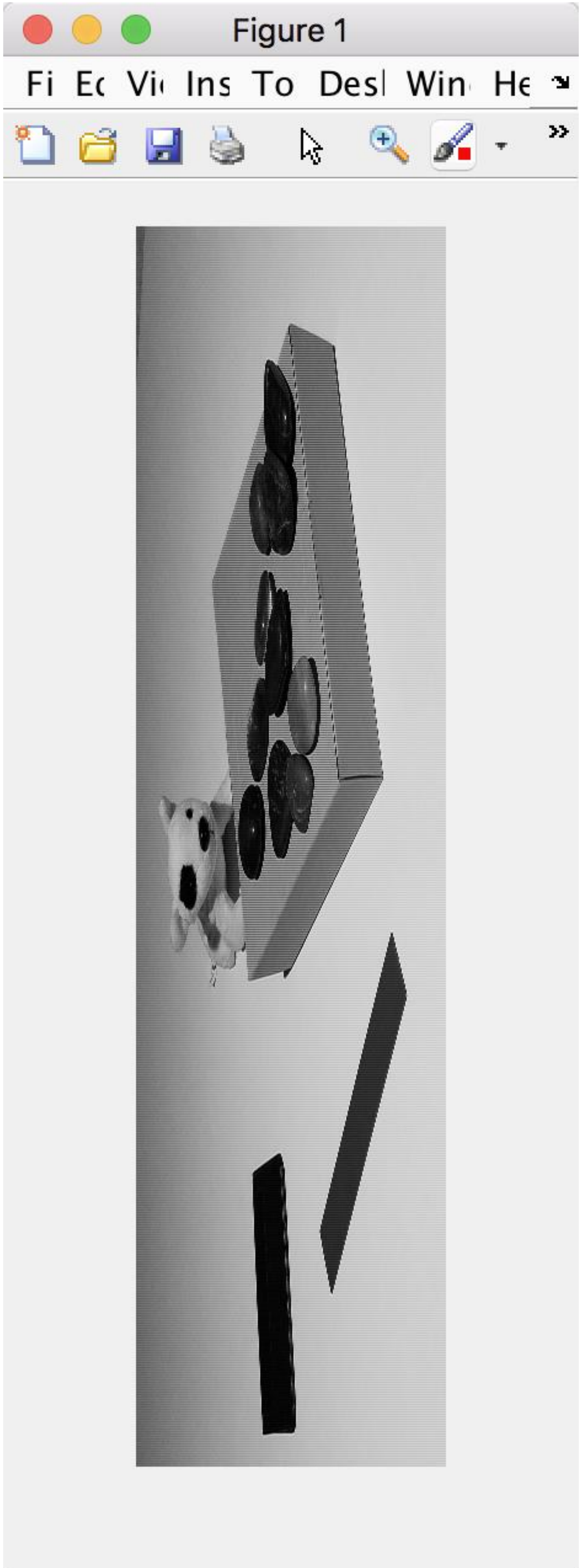


Figure 1
Fi Ec Vi Ins To Desl Win He