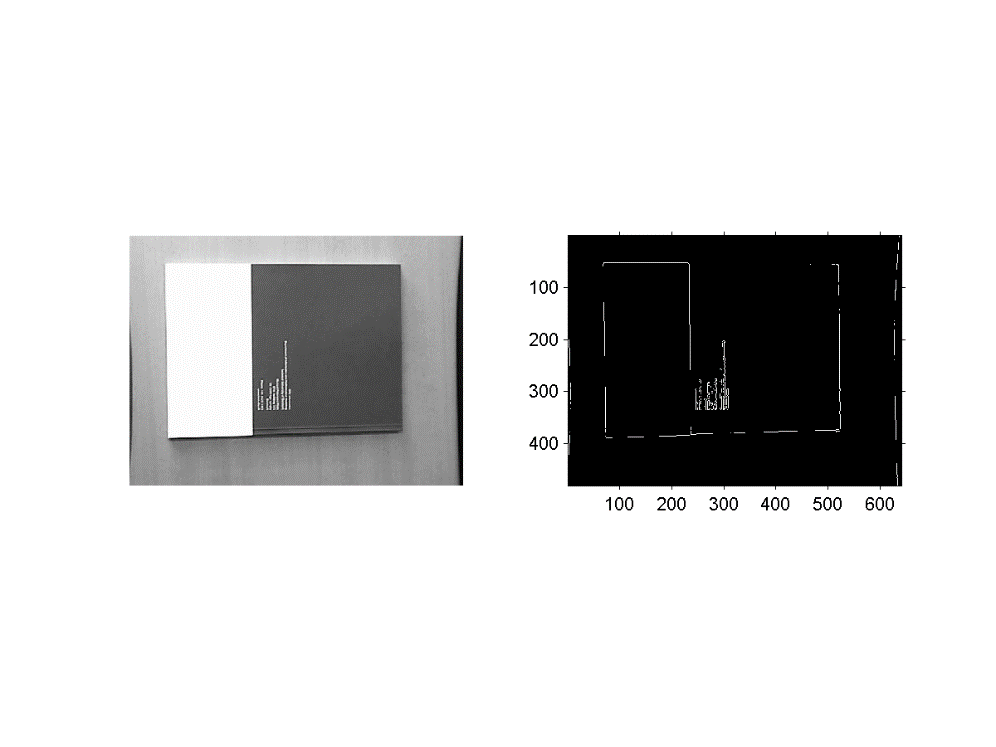
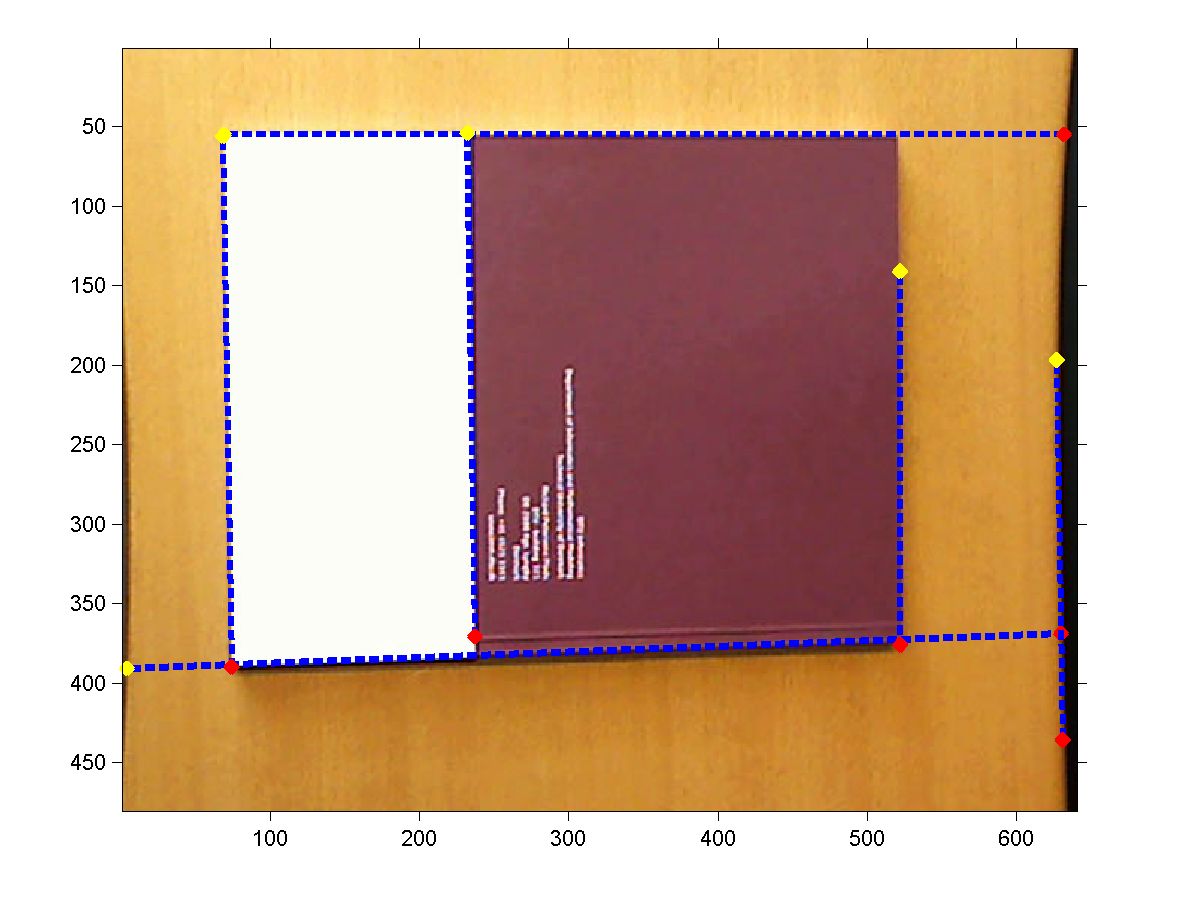
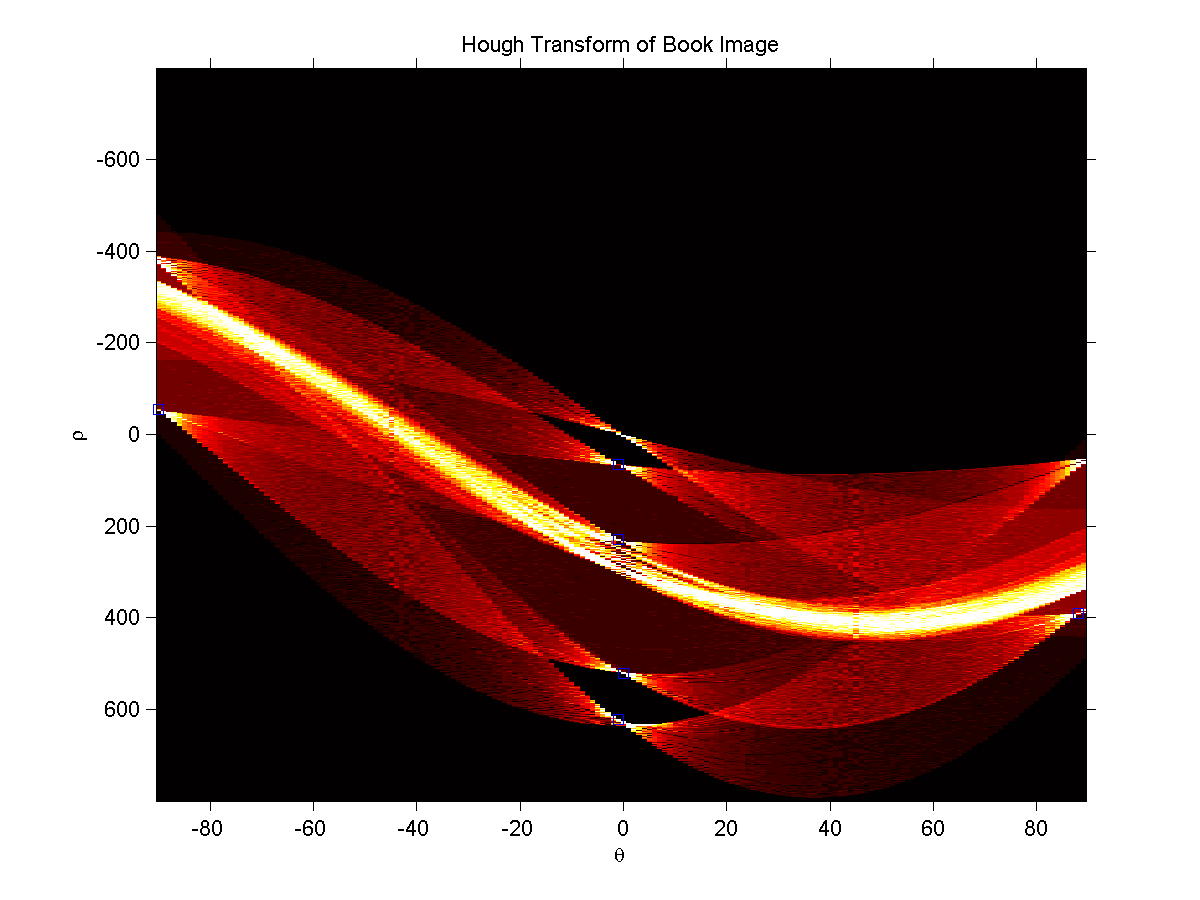
# Hough

H, the Hough transform matrix. theta (in degrees) and rho are the arrays of rho and theta values over which hough generates the Hough transform matrix.



6 peaks



Q8.

I\_edge = edge(I\_gray, 'canny',[0.07 0.31],0.9);% recognize the full edge of the book

% I\_edge = edge(I\_gray, 'sobel');%10

% I\_edge = edge(I\_gray, 'prewitt');%10

% I\_edge = edge(I\_gray, 'roberts');%missed some lines, but line on text.

# Ransac

1. Make a function that estimates a line, in homogeneous coordinates, given two points.

function l = estimate\_line(p1, p2)

x = [p1;1];

y = [p2;1];

l = cross(x,y);

end

2. Make a function that determines if a given 2D point is inlier or outlier wrt. a given line. The

threshold should be supplied as parameter to this function, such that it can easily be tuned latter.

function [ n ] = isinlier(l,p,t)

l = l/norm(l(1:2));

p = [p;1];

if abs(dot(l',p)) < t

n=1;

else

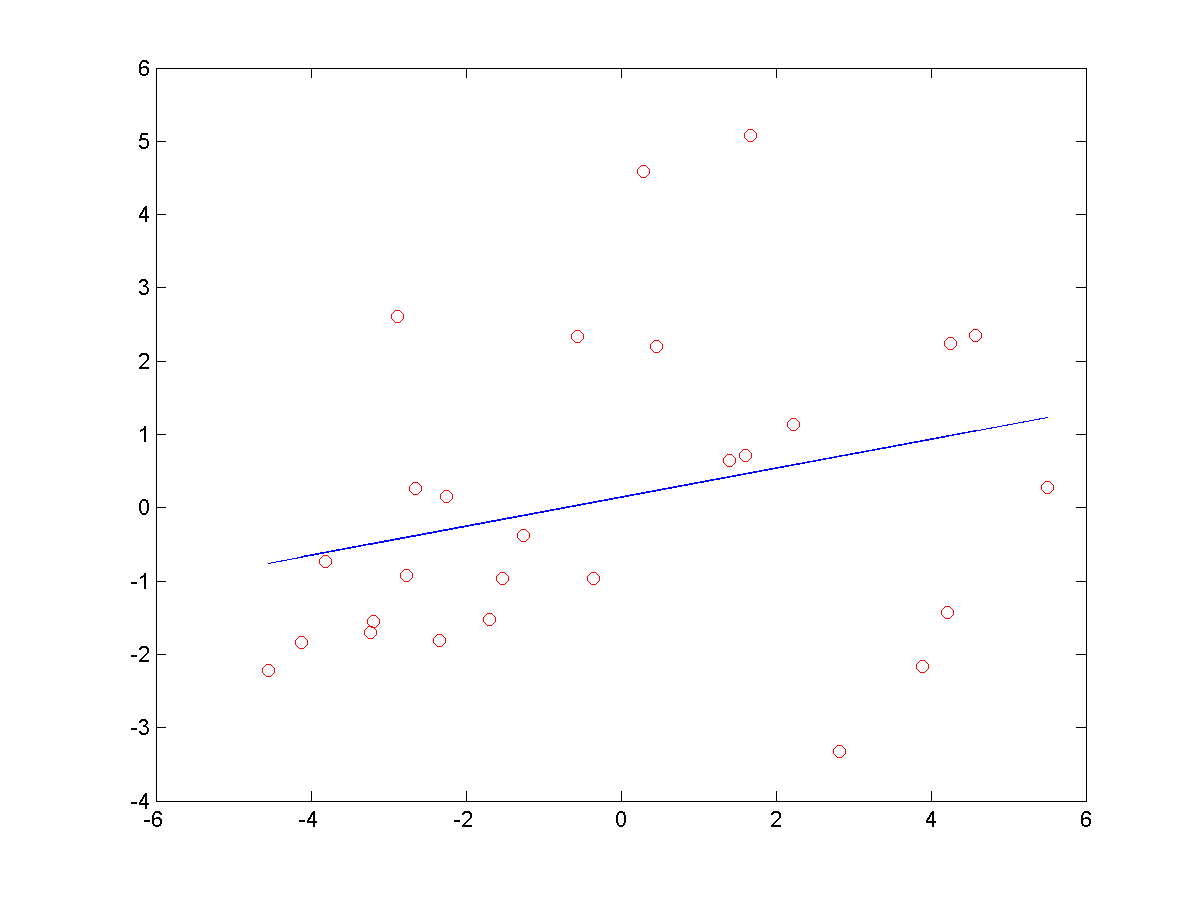
n=0;

end

end

3. Make a function that calculates the consensus, i.e. number of inliers, for a line wrt. to a set of

points.



p=RanLine(13,14); %generate non-homogeneous random points

t=1;

inlier = count\_inliers(l, p, t)

4. Make a function that randomly draws two of n 2D points.

function [p1 p2] = Ran2( p)

i =length(p)

r= randsample(i,2,'false');

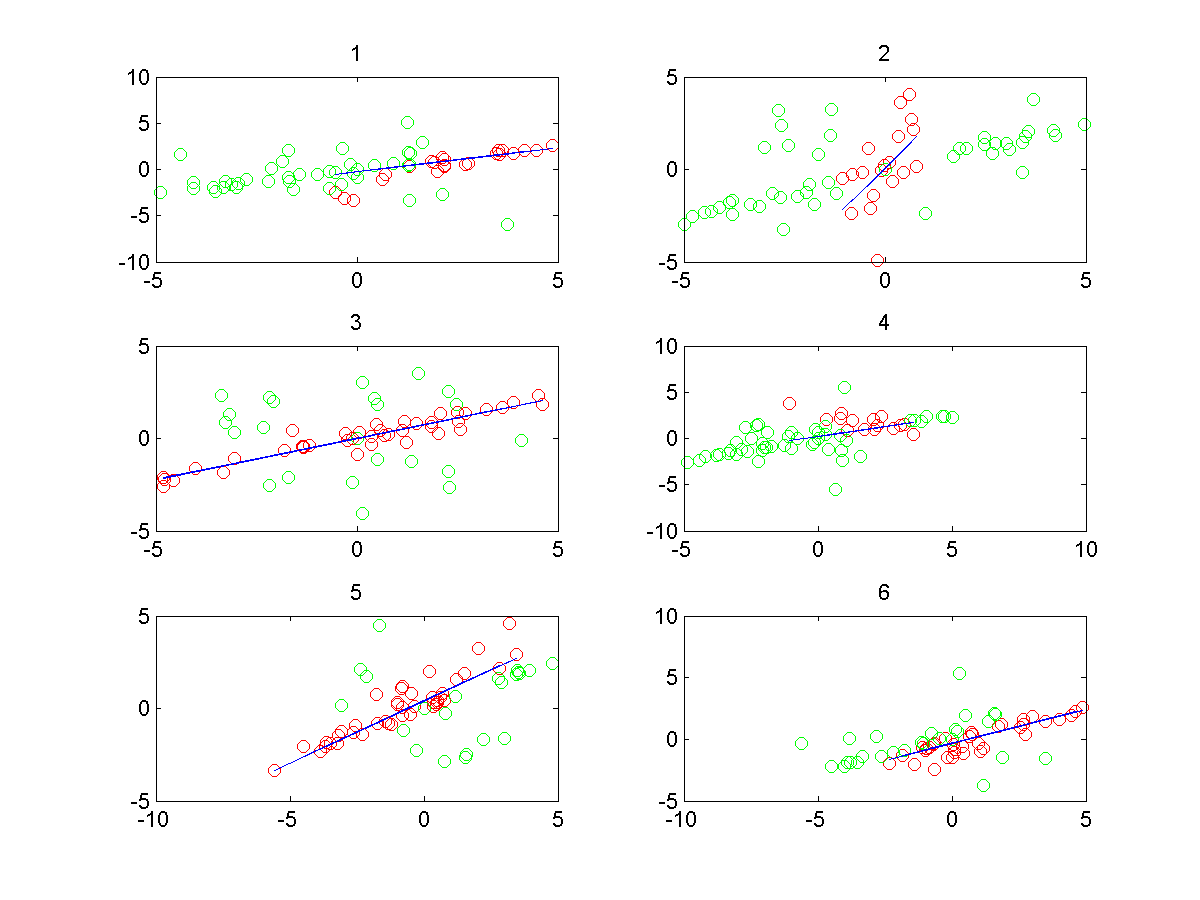
p1 = p(:,r(1));

p2 = p(:,r(2));

end

5. Assemble the functions made above to a working Ransac algorithm for estimating lines. The

number of iterations is set manually. 5th iter (estimated prob of outlier 0.2698, Iter needed 7)



iter =6;

t=1; %the smaller the t is, the more iter needed

for i=1:iter

in =33;

out=30;

p=RanLine(in,out);

[p1,p2]=Ran2(p);

l = estimate\_line(p1, p2);

[inlier(i) inp{i} outp{i}] = count\_inliers(l, p, t);

end

inlier

[highestCon I]= max(inlier)

inp = inp{I}

outp = outp{I}

m = in+out;

%estimated prob of outlier

iks = 1-highestCon/m

% iteraction needed based on the estimated prob of outlier

N=log(1-0.99)/log(1-(1-iks)^2)

plot(inp(1,:), inp(2,:),'ro',outp(1,:), outp(2,:),'go');hold on

pfit = polyfit(inp(1,:),inp(2,:),1) % coeficients

yfit = polyval(pfit,inp(1,:)); %evaluate coef

plot(inp(1,:),yfit,'b')

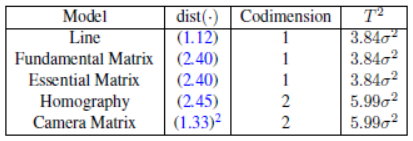
title('RansacFit');

xlabel(iks)

6. Experiment with the algorithm, what is a good threshold for distinguishing between inliers and

outliers?

95% confident level



A point is inlier if:

