CSE 5524 – Homework #8
10/21/2013
Manjari Akella

1). Compute and display the Harris pixel-wise cornerness function values for the image checker.jpg using a 3σ Gaussian window function with a standard deviation of σI = 1, a 3σ Gaussian derivative with a standard deviation of σD = 0.7, and a trace weighting factor of α = 0.05. Use the given code to display the pixel-wise cornerness function R. Then remove small values in R (< 1e6) and do non-maximum suppression to identify the corner points and display them on the image.

- Used the nlfilter function to do non maximal suppression

Output

Figure 1: Harris Cornerness function values

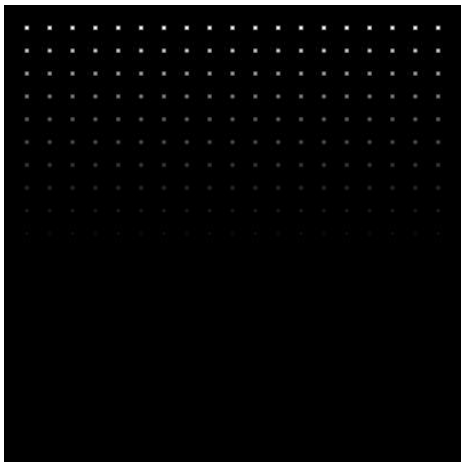

Figure 2: Harris Cornerness function values(threshold=1e6)

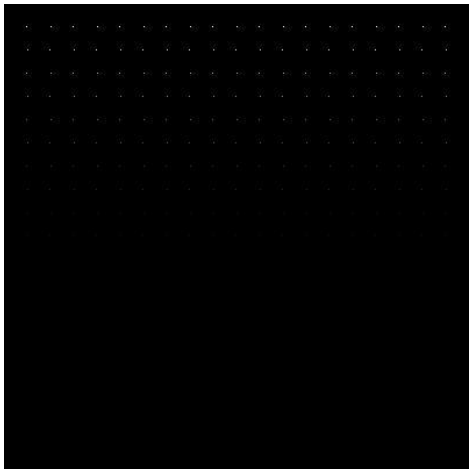

Figure 3: Harris Cornerness function values(non-max suppression)

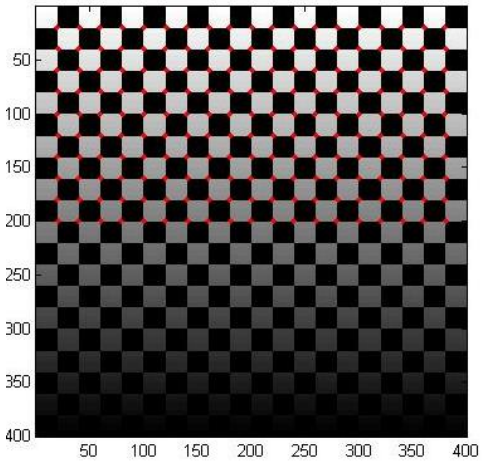

Figure 4: Corner Points

2). Compute and display the Shi-Tomasi pixel-wise cornerness function values for the image tower.jpg using a 3σ Gaussian window function with a standard deviation of σI = 1 and a 3σ Gaussian derivative with a standard deviation of σD = 0.7. This time do NOT threshold R, but just do non-maximum suppression and pick the top 150 corner points (150 largest R values after non-max suppression) and display them on the image.

Output

Figure 5: Shi-Tomasi Cornerness function values

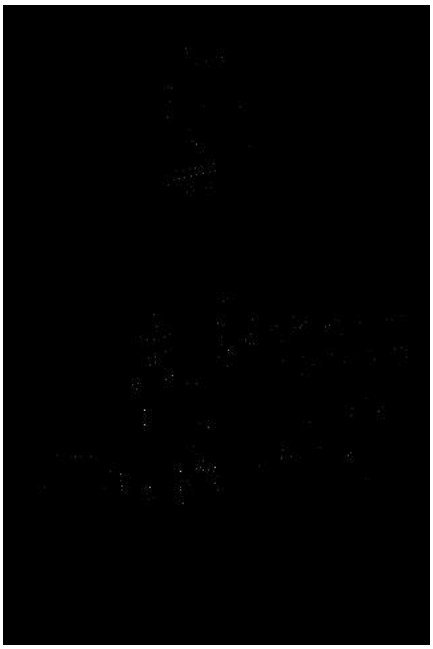

Figure 6: Shi-Tomasi (non-maximal suppression+top150)

Figure 7: Shi-Tomasi Corner points

3) Repeat Problem 2, but instead use the Harris response function and compare its 150 top corner points (after non-maximum suppression) to the 150 Shi-Tomasi points.

- The top 150 points of both Harris response function and Shi-Tomasi function were almost the same
- This can be seen from Figure 10

Output

Figure 8: Harris Cornerness function values

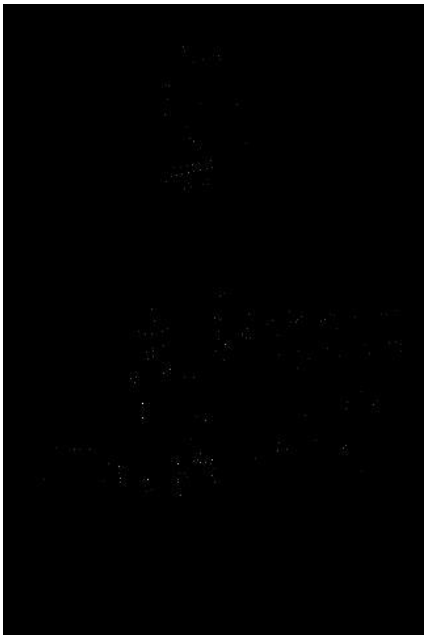

Figure 9: Harris(non-maximal suppression+top150)

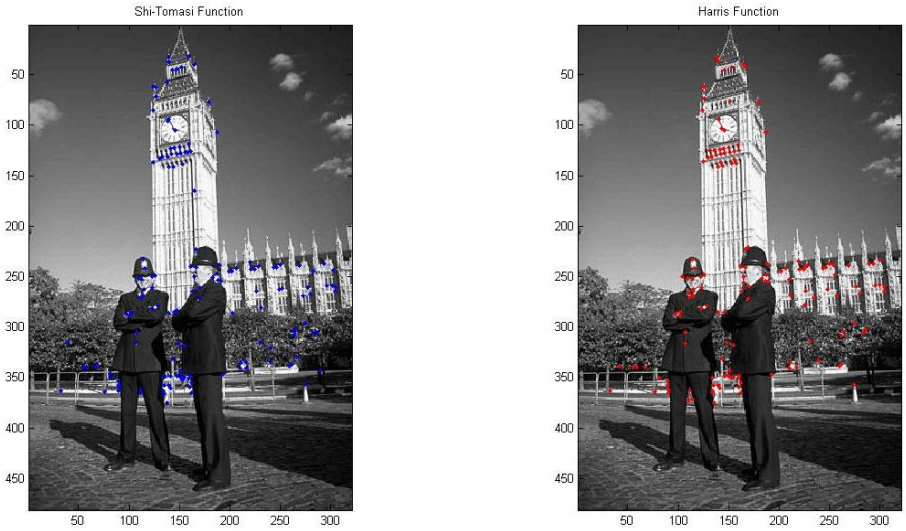

Figure 10: Shi Tomasi, Harris Response Top 150 Points

4) Implement the FAST feature point detector using a radius of r = 3, intensity threshold of T = 10, and a consecutive number of points threshold of n* = 9. Run the detector on the image tower.jpg. Display the image and overlay the FAST feature points.

- Ignored border pixels where making a circle wasn't possible

Output

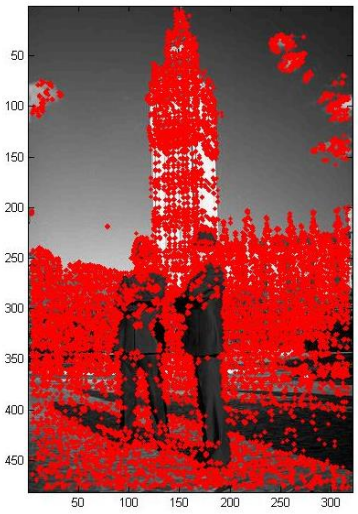

Figure 11: FAST, T=10, n*=9

5) Repeat the above problem with T = {20, 30, 50} and compare all four results.

- For T=10, a lot of points lit up. This can be seen in Figure 11. A lot of pixels were flagged as interesting. Pretty much everything except the sky(where intensity is same) and the people was flagged as interesting
- The number of points flagged decreased significantly as T went up. This can be seen in the Table 1
- For T=50, the number of points flagged was the least. Also, we can see from Figure 14 that these points (pixels) are actually picking up features such as edges of the building, corners of the clock tower, edges and buttons of the people etc.

T	No. of points
10	13722
20	7435
30	4664
50	2197

Table 1: Pixels flagged at various T

Output

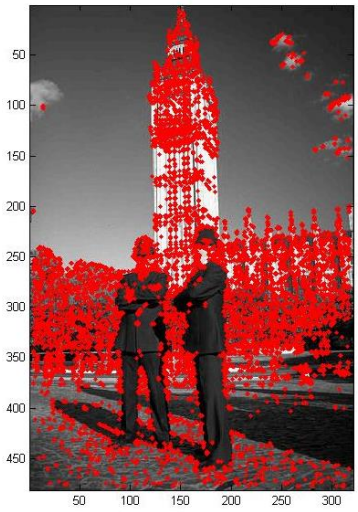

Figure 12: FAST, T=20, n*=9

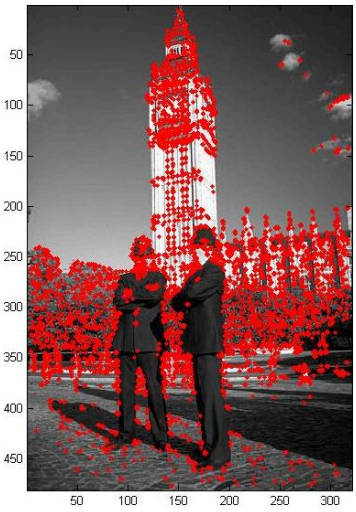

Figure 13: FAST, T=30, n*=9

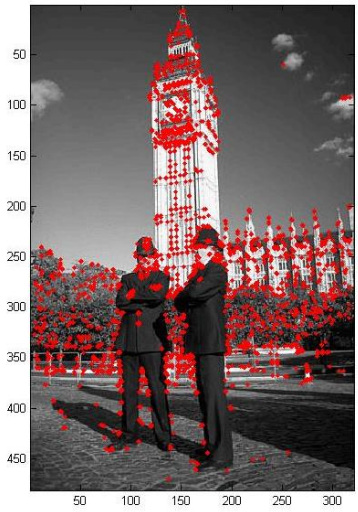

Figure 14: FAST, T=50, n*=9

CODE

1). function gaussDeriv2D.m

```
function [ Gx,Gy ] = gaussDeriv2D( sigma )
% Creates a 2D Gaussian Derivative mask in x and y directions
% Returns and displays them
range = ceil(3*sigma);
hsize = 2*ceil(3*sigma)+1;
for i = 1:hsize
    for j = 1:hsize
        x = i-range-1;
        y = j-range-1;
        t_1=x/(2*pi*(sigma^4));
        t_2=exp(-(x^2+y^2)/(2*sigma*sigma));
        Gx(j,i) = t_1*t_2;
    end
end
%generate a 2-D Gaussian kernel along y direction
Gy=Gx';
end
```

2). function check.m

```
function [ flag ] = check( A,N)
% Checks to see if largest contiguous pixel patch under consideration is >= N
or not. Returns flag = 0 if not else returns flag = 1
% A is a row vector, N is the threshold for number of contiguous pixels
% Values in A are : -1 = pixel is below threshold(T)
%                   +1 = pixel is above threshold(T)
%                   0 = either not a valid pixel or not above/below
%                   threshold
    countA = 0;
    countB = 0;
    % Break matrix so that circularity problem can be avoided
```

```
ind = find(A==0,1,'first');
% Check if ind is empty (i.e-no element is 0)
if (isempty(ind))
    ind=0;
    % Find ind for which A(i),A(i+1) are different
    for i=(1:size(A,2)-1)
         if(A(i)~=A(i+1))
            ind=i+1;
            break;
         else
            continue;
         end
    end
   % If ind still 0, this indicates, entire vector has same value
   if(ind==0)
        % if entire vector is 1(above)
        if(A(1,1)==1)
            countA=size(A,2);
        % else if entire vector is -1(below)
        elseif(A(1,1)==-1)
            countB=size(A,2);
        end
   else
        temp = A(1:(ind-1));
        newA = [A(ind:end),temp];
        i1 = find(newA==1,1,'first');
        i2 = find(newA==-1,1,'first');
        for i=i1:size(newA,2)
            if(newA(1,i)==0)
                break;
            elseif(newA(1,i)==-1)
                break;
            else
                countA = countA+1;
            end
        end
        for i =i2:size(newA,2)
            if(newA(1,i)==0)
                break;
            elseif(newA(1,i)==1)
                break;
            else
                countB = countB+1;
            end
        end
   end
else
    % if ind is not empty, break vector
    temp = A(1:(ind-1));
    newA = [A(ind:end),temp];
    i1 = find(newA==1,1,'first');
    i2 = find(newA==-1,1,'first');
    for i=i1:size(newA,2)
        if(newA(1,i)==0)
            break;
        elseif(newA(1,i)==-1)
            break;
```

```
            else
                countA = countA+1;
            end
        end
        for i =i2:size(newA,2)
            if(newA(1,i)==0)
                break;
            elseif(newA(1,i)==1)
                break;
            else
                countB = countB+1;
            end
        end
    end

    % Compare to N and set flag
    if ((countA>=N)||(countB>=N))
        flag=1;
    else
        flag=0;
    end
end
```

3). HW8.m

```
% Manjari Akella
% CSE5524 - HW8
% 10/21/2013

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Question 1
clear all;
close all;
clc;
%Define standard deviation values and alpha
sigmaD = 0.7;
sigma = 1;
a=0.05;
% Read image
I = double(imread('given_pics/checker.jpg'));
% Calculate gaussian derivatives
[Gx,Gy] = gaussDeriv2D(sigmaD);
% Run filter on image
Ix = imfilter(I,Gx,'replicate');
Iy = imfilter(I,Gy,'replicate');
% Compute Ix^2,Iy^2,Ix*Iy
Ix2 = Ix.^2;
Iy2 = Iy.^2;
Ixy = Ix.*Iy;
% Weighting gaussian
WG = fspecial('gaussian', 2*ceil(3*sigma)+1, sigma);
GIx2 = imfilter(Ix2,WG,'replicate');
GIy2 = imfilter(Iy2,WG,'replicate');
GIxy = imfilter(Ixy,WG,'replicate');
% Compute R
```

```
R = (GIx2.*GIy2)-((GIxy).^2)-(a.*((GIx2+GIy2).^2));
% Display R
Rimg = R - min(0, min(R(:)));
Rimg = Rimg/max(Rimg(:));
figure('Name','Q1: R','NumberTitle','off');
imshow(Rimg, 'Border', 'Tight');
colormap gray;
% Threshold R
R(R<1e6) = 0;
% Display R
Rimg = R - min(0, min(R(:)));
Rimg = Rimg/max(Rimg(:));
figure('Name','Q1: R(threshold=1e6)','NumberTitle','off');
imshow(Rimg, 'Border', 'Tight');
colormap gray;
% Non-maximal Suppression
R_new = nlfilter(R, [3 3], @(x) all(x(5) >= x([1:4 6:9])));
RR = R_new.*R;
% Display RR
Rimg = RR - min(0, min(RR(:)));
Rimg = Rimg/max(Rimg(:));
figure('Name','Q1: R(threshold=1e6 + Non-maximal
suppression)','NumberTitle','off');
imshow(Rimg, 'Border', 'Tight');
colormap gray;
% Show on original image
[r,c] = find(RR~=0);
figure('Name','Q1: Corner Points','NumberTitle','off');
imagesc(I);
colormap gray;
axis('image');
hold on;
for i=1:size(r,1)
    plot(c(i),r(i),'r.');
end
hold off;
pause;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Question 2
clear all;
close all;
clc;
%Define standard deviation values
sigmaD = 0.7;
sigma = 1;
% Read image and pre-allocate R
I = double(imread('given_pics/tower.jpg'));
R = zeros(size(I,1),size(I,2));
% Calculate gaussian derivatives
[Gx,Gy] = gaussDeriv2D(sigmaD);
% Run filter on image
Ix = imfilter(I,Gx,'replicate');
Iy = imfilter(I,Gy,'replicate');
% Compute Ix^2,Iy^2,Ix*Iy
Ix2 = Ix.^2;
Iy2 = Iy.^2;
```

```
Ixy = Ix.*Iy;
% Weighting gaussian
WG = fspecial('gaussian', 2*ceil(3*sigma)+1, sigma);
GIx2 = imfilter(Ix2,WG,'replicate');
GIy2 = imfilter(Iy2,WG,'replicate');
GIxy = imfilter(Ixy,WG,'replicate');
% Compute R (SHI-TOMASI)
for i=1:size(I,1)
    for j=1:size(I,2)
        M = [GIx2(i,j),GIxy(i,j);GIxy(i,j),GIy2(i,j)];
        d = eig(M);
        R(i,j) = min(d);
    end
end
% Display R
Rimg = R - min(0, min(R(:)));
Rimg = Rimg/max(Rimg(:));
figure('Name','Q2: R','NumberTitle','off');
imshow(Rimg, 'Border', 'Tight');
colormap gray;
% Non-maximal Supression
mask = nlfilter(R, [3 3], @(x) all(x(5) >= x([1:4 6:9])));
R_supp = R.*mask;
% Pick top 150 points
temp=sort(R_supp(:),'descend');
R_new=(R_supp>=temp(150)).*R_supp;
% Display
Rimg = R_new - min(0, min(R_new(:)));
Rimg = Rimg/max(Rimg(:));
figure('Name','Q2: R(Non-maximal suppression+top 150)','NumberTitle','off');
imshow(Rimg, 'Border', 'Tight');
colormap gray;
% Display
[r,c] = find(R_new~=0);
figure('Name','Q2/Q3: Corner Points(Shi-Tomasi+Harris)','NumberTitle','off');
subplot(121),imagesc(I);
colormap gray;
axis('image');
title('Shi-Tomasi Function');
hold on;
for i=1:size(r,1)
    plot(c(i),r(i),'b.');
end
hold off;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Question 3
% Compute R (HARRIS FUNCTION)
a=0.05;
RH = (GIx2.*GIy2)-((GIxy).^2)-(a.*((GIx2+GIy2).^2));
% Display
Rimg = RH - min(0, min(RH(:)));
Rimg = Rimg/max(Rimg(:));
% Non-maximal Supression
maskH = nlfilter(RH, [3 3], @(x) all(x(5) >= x([1:4 6:9])));
R_suppH = RH.*maskH;
% Pick top 150 points
```

```
temp=sort(R_suppH(:),'descend');
R_newH=(R_suppH>=temp(150)).*R_suppH;
% Display
[rH,cH] = find(R_newH~=0);
subplot(122),imagesc(I);
colormap gray;
axis('image');
title('Harris Function');
hold on;
for i=1:size(rH,1)
    plot(cH(i),rH(i),'r.');
end
hold off;
figure('Name','Q3: R','NumberTitle','off');
imshow(Rimg, 'Border', 'Tight');
colormap gray;
Rimg = R_newH - min(0, min(R_newH(:)));
Rimg = Rimg/max(Rimg(:));
figure('Name','Q3: R(Non-maximal Suppression+top 150)','NumberTitle','off');
imshow(Rimg, 'Border', 'Tight');
colormap gray;
imshow(Rimg, 'Border', 'Tight');
colormap gray;
pause;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Question 4
clc;
clear all;
close all;
% Load image
Im = double(imread('given_pics/tower.jpg'));
% Define radius, no. of consecutive points, intensity thresholds
N = 9;
T = 10;
fastX=0;
fastY=0;
count = 1;
nr = size(Im,1)-4;
nc = size(Im,2)-4;
% For each pixel, check 16 pixel neighbourhood(1=above,-1=below,0=pixel
% doesn't exist or neither above nor below
for i=4:nr
        for j=4:nc
% 1 i-3,j
            if(Im(i-3,j)>(Im(i,j)+T))
                A(1,1)=1;
            elseif(Im(i-3,j)<(Im(i,j)-T))
                A(1,1)=-1;
            else
                A(1,1)=0;
            end

% 2 i-3,j+1
            if(Im(i-3,j+1)>(Im(i,j)+T))
                A(2,1)=1;
```

```
            elseif(Im(i-3,j+1)<(Im(i,j)-T))
                A(2,1)=-1;
            else
                A(2,1)=0;
            end

% 3 i-2,j+2
            if(Im(i-2,j+2)>(Im(i,j)+T))
                A(3,1)=1;
            elseif(Im(i-2,j+2)<(Im(i,j)-T))
                A(3,1)=-1;
            else
                A(3,1)=0;
            end
% 4 i-1,j+3
            if(Im(i-1,j+3)>(Im(i,j)+T))
                A(4,1)=1;
            elseif(Im(i-1,j+3)<(Im(i,j)-T))
                A(4,1)=-1;
            else
                A(4,1)=0;
            end
% 5 i,j+3
            if(Im(i,j+3)>(Im(i,j)+T))
                A(5,1)=1;
            elseif(Im(i,j+3)<(Im(i,j)-T))
                A(5,1)=-1;
            else
                A(5,1)=0;
            end
% 6 i+1,j+3
            if(Im(i+1,j+3)>(Im(i,j)+T))
                A(6,1)=1;
            elseif(Im(i+1,j+3)<(Im(i,j)-T))
                A(6,1)=-1;
            else
                A(6,1)=0;
            end
%  7 i+2,j+2
            if(Im(i+2,j+2)>(Im(i,j)+T))
                A(7,1)=1;
            elseif(Im(i+2,j+2)<(Im(i,j)-T))
                A(7,1)=-1;
            else
                A(7,1)=0;
            end

% 8 i+3,j+1
            if(Im(i+3,j+1)>(Im(i,j)+T))
                A(8,1)=1;
            elseif(Im(i+3,j+1)<(Im(i,j)-T))
                A(8,1)=-1;
            else
                A(8,1)=0;
            end
```

```
% 9 i+3,j
            if(Im(i+3,j)>(Im(i,j)+T))
                A(9,1)=1;
            elseif(Im(i+3,j)<(Im(i,j)-T))
                A(9,1)=-1;
            else
                A(9,1)=0;
            end

% 10 i+3,j-1
            if(Im(i+3,j-1)>(Im(i,j)+T))
                A(10,1)=1;
            elseif(Im(i+3,j-1)<(Im(i,j)-T))
                A(10,1)=-1;
            else
                A(10,1)=0;
            end
% 11 i+2,j-2
            if(Im(i+2,j-2)>(Im(i,j)+T))
                A(11,1)=1;
            elseif(Im(i+2,j-2)<(Im(i,j)-T))
                A(11,1)=-1;
            else
                A(11,1)=0;
            end
% 12 i+1,j-3
            if(Im(i+1,j-3)>(Im(i,j)+T))
                A(12,1)=1;
            elseif(Im(i+1,j-3)<(Im(i,j)-T))
                A(12,1)=-1;
            else
                A(12,1)=0;
            end
% 13 i,j-3
            if(Im(i,j-3)>(Im(i,j)+T))
                A(13,1)=1;
            elseif(Im(i,j-3)<(Im(i,j)-T))
                A(13,1)=-1;
            else
                A(13,1)=0;
            end
 % 14 i-1,j-3
            if(Im(i-1,j-3)>(Im(i,j)+T))
                A(14,1)=1;
            elseif(Im(i-1,j-3)<(Im(i,j)-T))
                A(14,1)=-1;
            else
                A(14,1)=0;
            end
% 15 i-2,j-2
            if(Im(i-2,j-2)>(Im(i,j)+T))
                A(15,1)=1;
            elseif(Im(i-2,j-2)<(Im(i,j)-T))
                A(15,1)=-1;
            else
                A(15,1)=0;
            end
```

```
% 16 i-3,j-1
            if(Im(i-3,j-1)>(Im(i,j)+T))
                A(16,1)=1;
            elseif(Im(i-3,j-1)<(Im(i,j)-T))
                A(16,1)=-1;
            else
                A(16,1)=0;
            end

        % All 0 means no pixel breaches threshold
        if(nnz(A)~=0)
           flag = check(A',N);
           if (flag~=0)
                fastX(1,count)=j;
                fastY(1,count)=i;
                count=count+1;
           end
        end

        end
end
figure('Name','Q4: T=10','NumberTitle','off'); imagesc(Im);
colormap('gray');
axis('image');
hold on;
for i=1:size(fastX,2)
    plot(fastX(1,i),fastY(1,i),'r.');
end
hold off;
pause;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Question 5
clc;
clear all;
close all;
% Load image
Im = double(imread('given_pics/tower.jpg'));
% Define radius, no. of consecutive points, intensity thresholds
N = 9;
Threshold = [20,30,50];
nr = size(Im,1)-4;
nc = size(Im,2)-4;
% For each pixel, check 16 pixel neighbourhood(1=above,-1=below,0=pixel
% doesn't exist or neither above nor below
for itr=1:size(Threshold,2)
    T = Threshold(1,itr);
    fastX=0;
    fastY=0;
    count = 1;
    for i=4:nr
        for j=4:nc
% 1 i-3,j
            if(Im(i-3,j)>(Im(i,j)+T))
                A(1,1)=1;
            elseif(Im(i-3,j)<(Im(i,j)-T))
```

```
            A(1,1)=-1;
        else
            A(1,1)=0;
        end

% 2 i-3,j+1
        if(Im(i-3,j+1)>(Im(i,j)+T))
            A(2,1)=1;
        elseif(Im(i-3,j+1)<(Im(i,j)-T))
            A(2,1)=-1;
        else
            A(2,1)=0;
        end

% 3 i-2,j+2
        if(Im(i-2,j+2)>(Im(i,j)+T))
            A(3,1)=1;
        elseif(Im(i-2,j+2)<(Im(i,j)-T))
            A(3,1)=-1;
        else
            A(3,1)=0;
        end
% 4 i-1,j+3
        if(Im(i-1,j+3)>(Im(i,j)+T))
            A(4,1)=1;
        elseif(Im(i-1,j+3)<(Im(i,j)-T))
            A(4,1)=-1;
        else
            A(4,1)=0;
        end
% 5 i,j+3
        if(Im(i,j+3)>(Im(i,j)+T))
            A(5,1)=1;
        elseif(Im(i,j+3)<(Im(i,j)-T))
            A(5,1)=-1;
        else
            A(5,1)=0;
        end
% 6 i+1,j+3
        if(Im(i+1,j+3)>(Im(i,j)+T))
            A(6,1)=1;
        elseif(Im(i+1,j+3)<(Im(i,j)-T))
            A(6,1)=-1;
        else
            A(6,1)=0;
        end
%  7 i+2,j+2
        if(Im(i+2,j+2)>(Im(i,j)+T))
            A(7,1)=1;
        elseif(Im(i+2,j+2)<(Im(i,j)-T))
            A(7,1)=-1;
        else
            A(7,1)=0;
        end

% 8 i+3,j+1
```

```
            if(Im(i+3,j+1)>(Im(i,j)+T))
                A(8,1)=1;
            elseif(Im(i+3,j+1)<(Im(i,j)-T))
                A(8,1)=-1;
            else
                A(8,1)=0;
            end

% 9 i+3,j
            if(Im(i+3,j)>(Im(i,j)+T))
                A(9,1)=1;
            elseif(Im(i+3,j)<(Im(i,j)-T))
                A(9,1)=-1;
            else
                A(9,1)=0;
            end

% 10 i+3,j-1
            if(Im(i+3,j-1)>(Im(i,j)+T))
                A(10,1)=1;
            elseif(Im(i+3,j-1)<(Im(i,j)-T))
                A(10,1)=-1;
            else
                A(10,1)=0;
            end
% 11 i+2,j-2
            if(Im(i+2,j-2)>(Im(i,j)+T))
                A(11,1)=1;
            elseif(Im(i+2,j-2)<(Im(i,j)-T))
                A(11,1)=-1;
            else
                A(11,1)=0;
            end
% 12 i+1,j-3
            if(Im(i+1,j-3)>(Im(i,j)+T))
                A(12,1)=1;
            elseif(Im(i+1,j-3)<(Im(i,j)-T))
                A(12,1)=-1;
            else
                A(12,1)=0;
            end
% 13 i,j-3
            if(Im(i,j-3)>(Im(i,j)+T))
                A(13,1)=1;
            elseif(Im(i,j-3)<(Im(i,j)-T))
                A(13,1)=-1;
            else
                A(13,1)=0;
            end
 % 14 i-1,j-3
            if(Im(i-1,j-3)>(Im(i,j)+T))
                A(14,1)=1;
            elseif(Im(i-1,j-3)<(Im(i,j)-T))
                A(14,1)=-1;
            else
                A(14,1)=0;
```

```
            end
% 15 i-2,j-2
            if(Im(i-2,j-2)>(Im(i,j)+T))
                A(15,1)=1;
            elseif(Im(i-2,j-2)<(Im(i,j)-T))
                A(15,1)=-1;
            else
                A(15,1)=0;
            end
% 16 i-3,j-1
            if(Im(i-3,j-1)>(Im(i,j)+T))
                A(16,1)=1;
            elseif(Im(i-3,j-1)<(Im(i,j)-T))
                A(16,1)=-1;
            else
                A(16,1)=0;
            end

        % All 0 means no pixel breaches threshold
        if(nnz(A)~=0)
           flag = check(A',N);
           if (flag~=0)
                fastX(1,count)=j;
                fastY(1,count)=i;
                count=count+1;
           end
        end

        end
    end
    figure('Name',strcat('Q4: T=',num2str(T)),'NumberTitle','off');
    imagesc(Im);
    colormap('gray');
    axis('image');
    hold on;
    for i=1:size(fastX,2)
        plot(fastX(1,i),fastY(1,i),'r.');
    end
    hold off;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%% END %%%%%%%%%%%%%%%%%%%%
```