**Homework 3 Report**

**Name: Ruohuan Xu ID: 10453903**

I follow the steps in the homework assignment and implement the requires from it. In the end, sorting the distance values and select the top 5% most likely correspondences. Report the number of correct and incorrect correspondences included in this set. Repeat for the top 10% all the way to 100% in 5% increments.

1. Top5% most likely correspondences:

Correct number:76 Wrong number: 13452

2. Top10% most likely correspondences:

Correct number:140 Wrong number: 26915

3. Top15% most likely correspondences:

Correct number:204 Wrong number: 40379

4. Top20% most likely correspondences:

Correct number:273 Wrong number: 53837

5. Top25% most likely correspondences:

Correct number:339 Wrong number: 67298

6. Top30% most likely correspondences:

Correct number:415 Wrong number: 80750

7. Top35% most likely correspondences:

Correct number:477 Wrong number: 94215

8. Top40% most likely correspondences:

Correct number:535 Wrong number: 107685

9. Top45% most likely correspondences:

Correct number:588 Wrong number: 121159

10. Top50% most likely correspondences:

Correct number:629 Wrong number: 134645

11. Top55% most likely correspondences:

Correct number:678 Wrong number: 148124

12. Top60% most likely correspondences:

Correct number:727 Wrong number: 161602

13. Top65% most likely correspondences:

Correct number:783 Wrong number: 175074

14. Top70% most likely correspondences:

Correct number:811 Wrong number: 188573

15. Top75% most likely correspondences:

Correct number:843 Wrong number: 202068

16. Top80% most likely correspondences:

Correct number:880 Wrong number: 215559

17. Top85% most likely correspondences:

Correct number:913 Wrong number: 229053

18. Top90% most likely correspondences:

Correct number:960 Wrong number: 242534

19. Top95% most likely correspondences:

Correct number:1018 Wrong number: 256003

20. Top100% most likely correspondences:

Correct number:1061 Wrong number: 269487

Source code:

%Read the Teddy stereo pair

teddy\_left = double(imread('teddyL.pgm'));

teddy\_right = double(imread('teddyR.pgm'));

height = size(teddy\_left, 1);

width = size(teddy\_left, 2);

%Define the derivative matrix

matrix = [-1 0 1; -1 0 1; -1 0 1];

%Use matrix to compute Ix and its transpose to compute Iy

left\_Ix = conv2(teddy\_left, matrix, 'same');

left\_Iy = conv2(teddy\_left, matrix', 'same');

right\_Ix = conv2(teddy\_right, matrix, 'same');

right\_Iy = conv2(teddy\_right, matrix', 'same');

%Use these first order derivaties to compute I2x, I2y and Ixy at each pixel

left\_IxH = left\_Ix.\*left\_Ix;

left\_Iy2 = left\_Iy.\*left\_Iy;

right\_Ix2 = right\_Ix.\*right\_Ix;

right\_Iy2 = right\_Iy.\*right\_Iy;

left\_Ixy = left\_Ix.\*left\_Iy;

right\_Ixy = right\_Ix.\*right\_Iy;

%Define Gaussian smoothing

gaussian\_size = 5;

halfsize = (gaussian\_size - 1)/2;

standard\_deviation = 1; %set standard deviation of the distribution as 1.0 and kernel size 5¡Á5

[x,y] = meshgrid(-halfsize:halfsize, -halfsize:halfsize);

G\_1 = exp(-(x.^2 + y.^2)/(2\*standard\_deviation^2));

G = G\_1./sum(G\_1(:)); %average the derivative values in 5X5 windows centered at each pixel

%Apply Gaussian smoothing

left\_Ixg = conv2(left\_IxH, G, 'same');

left\_Iyg = conv2(left\_Iy2, G, 'same');

right\_Ixg = conv2(right\_Ix2, G, 'same');

right\_Iyg = conv2(right\_Iy2, G, 'same');

left\_Ixyg = conv2(left\_Ixy, G, 'same');

right\_Ixyg = conv2(right\_Ixy, G, 'same');

%Compute the Harris operator response function

corner\_left = zeros(size(teddy\_left));

corner\_right = zeros(size(teddy\_right));

harris\_size = 5;

halfsize2 = (harris\_size - 1)/2;

threshold = 98000; %start the value for the threshold from 2000,

%but there will be more than 5000 corners so rise the vaule thershold

%until the corners will be 300-500

%The Harris operator f = det(M)/trace(M)^2

for i = halfsize2+1:height-halfsize2-1

for j = halfsize2+1:width-halfsize2-1

%create the matrix M

left\_IxH = sum(sum(left\_Ixg(i-halfsize2:i+halfsize2, j-halfsize2:j+halfsize2)));

left\_IyH = sum(sum(left\_Iyg(i-halfsize2:i+halfsize2, j-halfsize2:j+halfsize2)));

left\_IxyH = sum(sum(left\_Ixyg(i-halfsize2:i+halfsize2, j-halfsize2:j+halfsize2)));

right\_IxH = sum(sum(right\_Ixg(i-halfsize2:i+halfsize2, j-halfsize2:j+halfsize2)));

right\_IyH = sum(sum(right\_Iyg(i-halfsize2:i+halfsize2, j-halfsize2:j+halfsize2)));

right\_IxyH = sum(sum(right\_Ixyg(i-halfsize2:i+halfsize2, j-halfsize2:j+halfsize2)));

%Compare with the threshold, set the corner value and set the rest

%as 0

corner\_left(i,j) = (left\_IxH\*left\_IyH - left\_IxyH\*left\_IxyH)/(left\_IxH+left\_IyH);

%corner\_left(i,j) = left\_IxyH/(left\_IxH^2+left\_IyH^2+2\*left\_IxyH);

if corner\_left(i,j)< threshold

corner\_left(i,j) = 0;

end

corner\_right(i,j) = (right\_IxH\*right\_IyH - right\_IxyH\*right\_IxyH)/(right\_IxH+right\_IyH);

%corner\_right(i,j) = right\_IxyH/(right\_IxH^2+right\_IyH^2+2\*right\_IxyH);

if corner\_right(i,j)< threshold

corner\_right(i,j) = 0;

end

end

end

%Apply non-maximum suppression on the responses of the Harris operator in 3X3 windows.

suppression\_size = 3;

halfsize3 = (suppression\_size - 1)/2;

corner\_count1 = 1;

corner\_count2 = 1;

for i = halfsize3+1:size(teddy\_left,1)-halfsize3-1

for j = size(teddy\_left,2)-halfsize3-1:-1:halfsize3+1

%judge whether the pixel has the maximum response

if corner\_left(i,j) ~= 0

neighborhood = corner\_left(i-halfsize3:i+halfsize3, j-halfsize3:j+halfsize3);

if sum(sum(corner\_left(i,j) < neighborhood)) == 0

corner\_left2(corner\_count1,:) = [corner\_left(i,j) i j]; %save the corners'value and states

corner\_count1 = corner\_count1+1;

else

corner\_left(i,j) = 0; %Make sure that the order processed pixels does

end %not affect the output

end

if corner\_right(i,j) ~= 0

neighborhood = corner\_right(i-halfsize3:i+halfsize3, j-halfsize3:j+halfsize3);

if sum(sum(corner\_right(i,j) < neighborhood)) == 0

corner\_right2(corner\_count2,:) = [corner\_right(i,j) i j];

corner\_count2 = corner\_count2+1;

else

corner\_right(i,j) = 0;

end

end

end

end

total\_left = length(corner\_left2);

total\_right = length(corner\_right2);

%Using SAD to compute the distance

sad\_size = 3;

halfsize4 = (sad\_size - 1)/2;

corner\_count3 = 1;

for i = 1:length(corner\_left2)

for j = 1:length(corner\_right2)

xl = corner\_left2(i,2);

xr = corner\_right2(j,2);

yl = corner\_left2(i,3);

yr = corner\_right2(j,3);

%set 3X3 window for sad

sad\_left = teddy\_left(xl-halfsize4:xl+halfsize4, yl-halfsize4:yl+halfsize4);

sad\_right = teddy\_right(xr-halfsize4:xr+halfsize4, yr-halfsize4:yr+halfsize4);

distance(corner\_count3,:) = [sum(sum(abs(sad\_left-sad\_right))) i j];

corner\_count3 = corner\_count3+1;

end

end

%Compute the correct and incorrect

k = 20;

corner\_count5 = 0;

%get top #% distance

top\_distance = ceil((k/100)\*length(distance));

distance = sortrows(distance);

distance = distance(1:top\_distance,:);

gt = imread('disp2.pgm');

gt = double(gt)./4;

for i = 1:length(distance)

disparity = distance(i,3);

disparity\_gt = gt(corner\_left2(distance(i,2),2), corner\_left2(distance(i,2),3));

if abs(disparity-disparity\_gt) <= 1

corner\_count5 = corner\_count5 + 1;

end

end

disp(['Top' num2str(k) '% most likely correspondences: '])

disp(['Correct number:' num2str(corner\_count5) ' Wrong number: ' num2str(length(distance)-corner\_count5)])