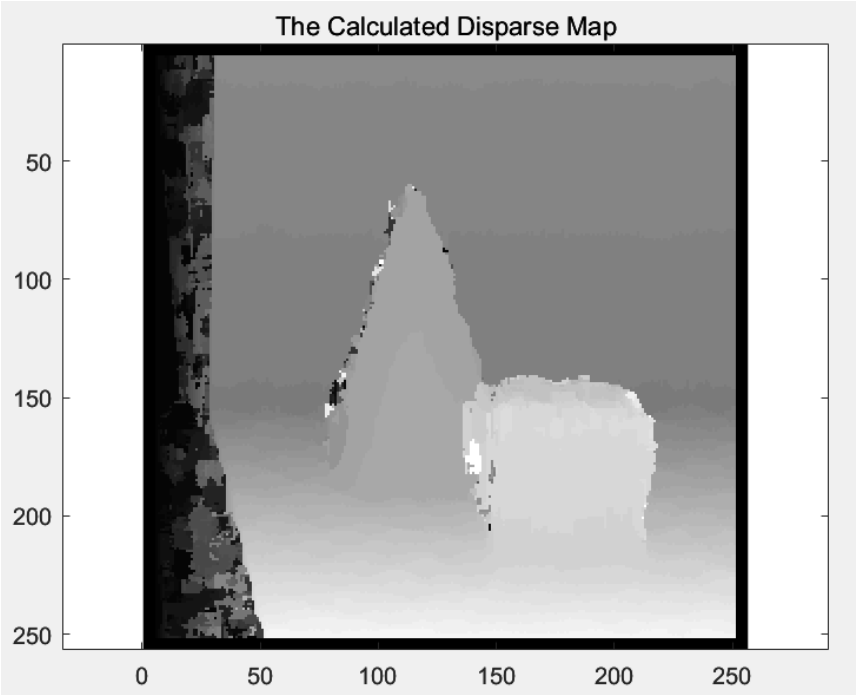
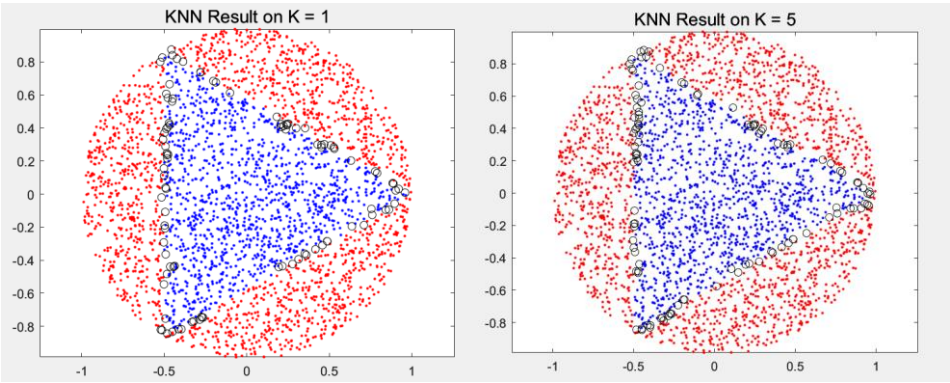
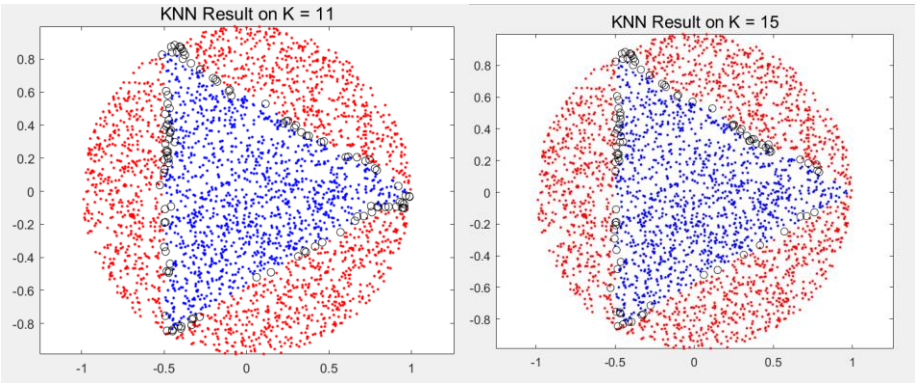
Yukun Duan
Computer Vision for HCI AU'22
Homework Assignment #10

1.
The disparity map is printed below. The brighter area (which looks like a square) locates in the front of the room, and the cone that locates further looks darker than the square in the front. The ground at the center of the image looks darker, and the ground becomes brighter when it's getting closer to the front. Overall, the brightness indicates how far the object is from the camera, and the disparity map is a good measurement of how close the object is to the camera.



2.
The accuracy at K = 1 is: 96.767%
The accuracy at K = 5 is: 96.333%
The accuracy at K = 11 is: 96.267%
The accuracy at K = 15 is: 96.900%





We can find that some points at the boundary seem to hesitate. When we increase K, we hope to use more information from surrounding points to decide which class it belongs to. The points that are far from boundaries have accurate classification. The result at K=15 has the highest accuracy, and values between 1 and 15 seem to have less accuracy on this test set.

Attached code:

```
% Yukun Duan
% CSE5524 - HW10
% 11/4/2022


%% Problem 1
left = double(imread('left.png'));
right = double(imread('right.png'));
map = zeros(size(left));
[r,c] = size(left);
l = 11; % window size
h = floor(l/2);

for y = 1:r-l+1
    for x = 1:c-l+1
```

```
        leftwindow = left(y:y+11-1, x:x+11-1);
        ncclist = zeros(1,x-max(1, x-50)+1);
        for z = 1:x-max(1, x-50)+1
            rightwindow = right(y:y+11-1, x-z+1:x-z+11);
            ncclist(z) = calculateNCC(leftwindow, rightwindow);
        end
        [m, m_ind] = max(ncclist);
        map(y+h, x+h) = m_ind;
    end
end
figure;
imagesc(map, [0 50]);
axis equal;
colormap("gray")
title("The Calculated Disparse Map")
saveas(gcf,"depth.png");
pause;


%% Problem 2
load('train.txt')
load('test.txt')
x_train = train(:,1:2);
y_train = train(:,3);
x_test = test(:, 1:2);
y_test = test(:, 3);

K = [1 5 11 15];

for k = K
   predict = KNN(x_train, y_train, x_test, y_test, k);
   plot(x_test(predict==y_test & predict == 1,1), x_test(predict==y_test &
predict == 1,2), 'b.')
   axis equal;
   hold on
   plot(x_test(predict==y_test & predict == 2,1), x_test(predict==y_test &
predict == 2,2), 'r.')
   plot(x_test(predict~=y_test,1 ), x_test(predict ~= y_test,2), 'ko')
   hold off
   title(sprintf('KNN Result on K = %d',k),'FontSize',14)
   saveas(gcf,sprintf("k%d.png",k));
   pause;
end
```

```
%% function
% return the value of NCC of two eqo_sized image.
function ncc = calculateNCC(origin, template)
    p_mean = mean(origin, [1 2]);
    t_mean = mean(template, [1 2]);
    p_sigma = std(origin, 0, [1 2]);
    t_sigma = std(template, 0, [1 2]);
    arr = ((origin - p_mean).*(template - t_mean))./(p_sigma.*t_sigma);
    % prevent inf value from dividing 0
    arr(p_sigma * t_sigma == 0) = min(arr, [], 'all');
    ncc = sum(sum(arr, [1 2])/(size(template,1)*size(template,2)-1),
'all');
end

% return a column of predicted labels
function predict = KNN(x_train, y_train, x_test, y_test, k)
    index = knnsearch(x_train, x_test, 'K', k);
    predict = y_train(index);
    predict = mode(predict, 2);
    accuracy = sum(predict == y_test)/size(y_test, 1);
    fprintf('The accuracy at K = %u is: %.3f%%\n', k, accuracy*100)
end
```