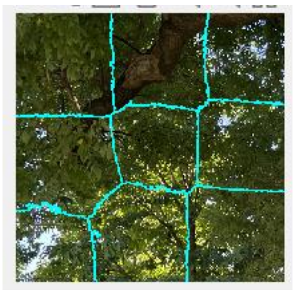
Yukun Duan
Computer Vision for HCI AU'22
Homework Assignment #7

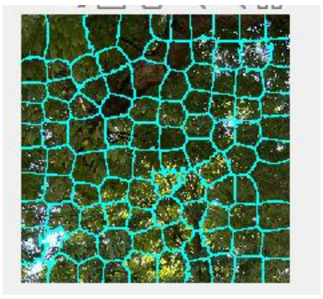
1.

I tested when the target number of superpixels at 10, 100, 200, 250, and 400, while the compactness value was at 3, 5, 10, 15, and 20. Here are the different boundaries overlaid:

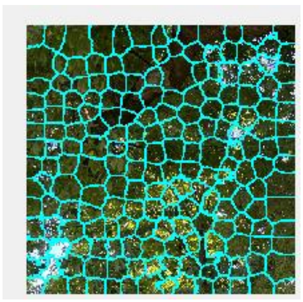
(10,10)



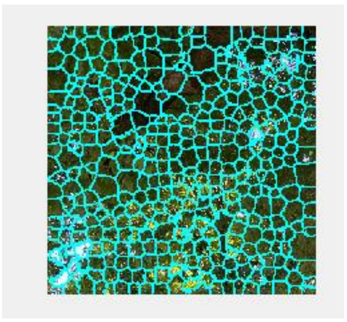
(100,10)



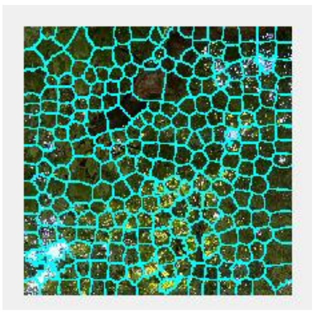
(200,10)



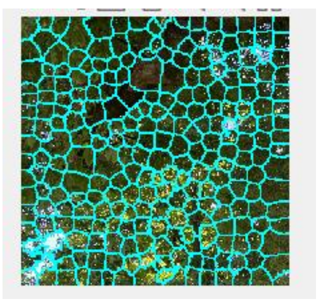
(400,10)



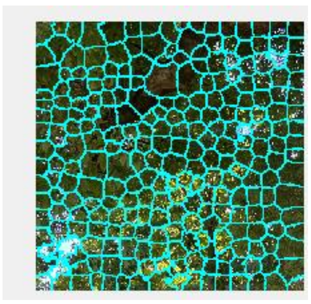
(250,3)



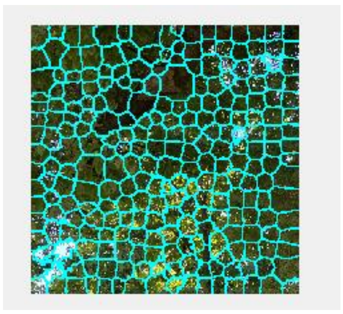
(250,5)



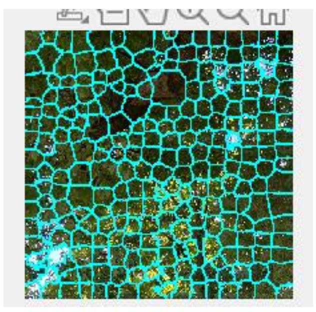
(250,10)



(250,15)



(250,20)

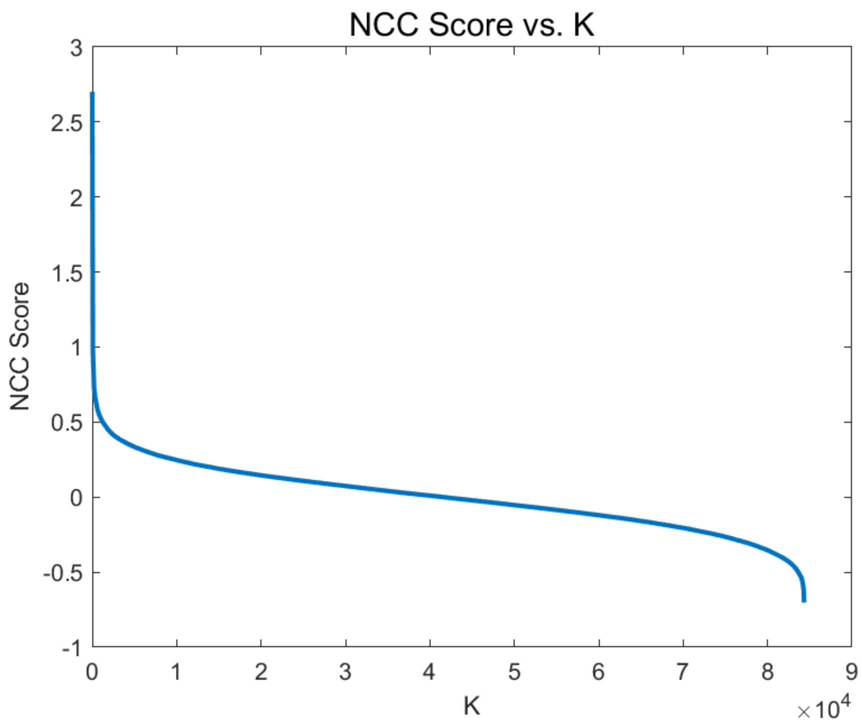


Here is the original picture:

Based on the test of different superpixel values and compactness values, I found that the target number of superpixels and the “compactness” parameter will all influence the regularity of the shape and size of the resulting segment. Normally, the larger these two values are, the greater number of segments we can obtain, and the shape is more likely irregular.

2.

The NCC list is calculated based on the image given on the slides. Here is the plot of NCC vs. K values and the corresponding results are shown below:


NCC Score vs. K
NCC Score
K
$\times 10^4$
3
2.5
2
1.5
1
0.5
0
-0.5
-1
0
1
2
3
4
5
6
7
8
9

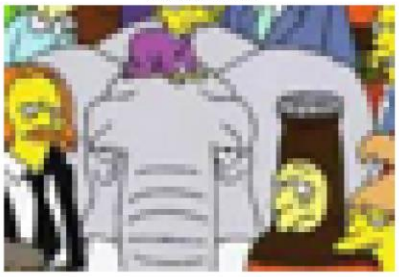
The 1st Patch

The 2st Patch

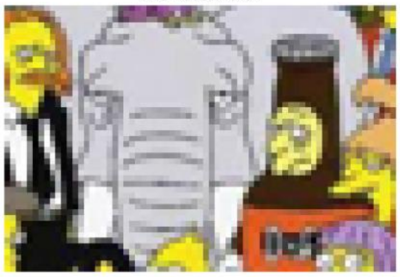
The 5st Patch

The 10st Patch

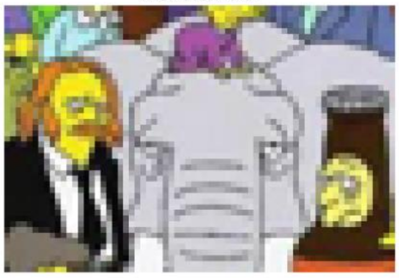
The 50st Patch



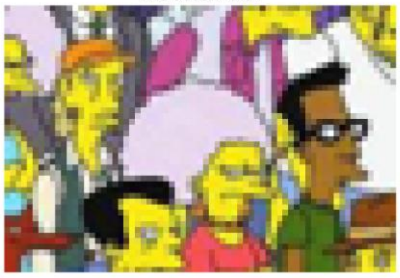
The 100st Patch



The 200st Patch



The 500st Patch



I found that the center of the first patch is [106 106]. The result looks reasonable for k equals 1 to 10. The 50th, 100th, and 200th results caught the elephant, but the center is a little bit far away from the origin in the template. The 500th result failed to catch the correct spot. The plot indicates that the most accurate match occurs in K < 200. The value of NCC drops significantly from 1-150, and afterward, the slope becomes smooth and flat for K > 400.

Attached Code:

```
% Yukun Duan
% CSE5524 - HW7
% 10/15/2022


%% Problem 1

%%
myimage = imread('pic.jpg');
[L, NumLabels] = superpixels(myimage, 10, "Compactness",10);
% Display the superpixel boundaries overlaid on the original image
BW = boundarymask(L);
imshow(imoverlay(myimage,BW,'cyan'),'InitialMagnification',67);
pause;
%%
```

```
myimage = imread('pic.jpg');
[L, NumLabels] = superpixels(myimage, 100, "Compactness",10);
BW = boundarymask(L);
imshow(imoverlay(myimage,BW,'cyan'),'InitialMagnification',67);
pause;
%%
myimage = imread('pic.jpg');
[L, NumLabels] = superpixels(myimage, 200, "Compactness",10);
BW = boundarymask(L);
imshow(imoverlay(myimage,BW,'cyan'),'InitialMagnification',67);
pause;
%%
myimage = imread('pic.jpg');
[L, NumLabels] = superpixels(myimage, 400, "Compactness",10);
BW = boundarymask(L);
imshow(imoverlay(myimage,BW,'cyan'),'InitialMagnification',67);
pause;

%%
myimage = imread('pic.jpg');
[L, NumLabels] = superpixels(myimage, 250, "Compactness",3);
BW = boundarymask(L);
imshow(imoverlay(myimage,BW,'cyan'),'InitialMagnification',67);
pause;
%%
myimage = imread('pic.jpg');
[L, NumLabels] = superpixels(myimage, 250, "Compactness",5);
BW = boundarymask(L);
imshow(imoverlay(myimage,BW,'cyan'),'InitialMagnification',67);
pause;
%%
myimage = imread('pic.jpg');
[L, NumLabels] = superpixels(myimage, 250, "Compactness",10);
BW = boundarymask(L);
imshow(imoverlay(myimage,BW,'cyan'),'InitialMagnification',67);
pause;
%%
myimage = imread('pic.jpg');
[L, NumLabels] = superpixels(myimage, 250, "Compactness",15);
BW = boundarymask(L);
imshow(imoverlay(myimage,BW,'cyan'),'InitialMagnification',67);
pause;
%%
myimage = imread('pic.jpg');
```

```matlab
[L, NumLabels] = superpixels(myimage, 250, "Compactness",20);
BW = boundarymask(L);
imshow(imoverlay(myimage,BW,'cyan'),'InitialMagnification',67);
pause;



%% Problem 2
img = double(imread('search.png'));
template = double(imread('template.png'));
ncc = imageIteration(img, template);
for i = [1 2 5 10 50 100 200 500]
    dispCropImg(img, ncc(i,2:3), [47 69])
    title(sprintf('The %dst Patch',i),'FontSize',14)
    saveas(gcf,sprintf('patch_%d.png',i))
    pause;
end
disp(ncc(1,:))
displayPlot(reshape(ncc(:,1), 1, []), size(ncc, 1))



%% 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);
    ncc = sum(sum(arr, [1 2])/(size(template,1)*size(template,2)-1),
'all');
end

% Calculate the sorted list of NCC given image and template
function nccList = imageIteration(img, template)
    ncc = zeros(size(img) - size(template) + 1);
    halfr = floor(size(template,1)/2);
    halfc = floor(size(template,2)/2);
    for r = halfr+1:size(img, 1)-halfr
        for c = halfc+1:size(img, 2)-halfc
            crop = img(r-halfr:r+halfr, c-halfc:c+halfc, :);
            ncc(r-halfr,c-halfc) = calculateNCC(crop, template);
        end
    end
    yind = double(repmat((halfr+1:size(img, 1)-halfr)', 1, size(ncc,2)));
    xind = double(repmat(halfc+1:size(img, 2)-halfc,size(ncc,1),1));
    nccList = cat(3,ncc,yind,xind);
    nccList = reshape(nccList, [], 3);
    nccList = sortrows(nccList, 1, 'descend');
end

% Display the cropped image given coordinate (r, c) and result size (r, c)
function dispCropImg(img, coordinate, size)
    r = coordinate(1); % row of center pixel
    c = coordinate(2); % col of center pixel
    halfr = floor(size(1)/2);
    halfc = floor(size(2)/2);
    imshow(img(r-halfr:r+halfr, c-halfc:c+halfc, :)/255,
'InitialMagnification','fit');
end

% Plot the NCC score where arr = [y1 y2 y3 ...]
function displayPlot(arr, len)
    x = 1:1:len;
    y = arr(1:len);
    plot(x,y,'Linewidth',2)
    xlabel('K')
    ylabel('NCC Score')
    title('NCC Score vs. K','FontSize',14)
    saveas(gcf,"NCC_Score.png");
end
```