Answers to questions in Lab 3: Image segmentation

Name: Badi Mirzai Program: Systems Control and Robotics

Instructions: Complete the lab according to the instructions in the notes and respond to the questions stated below. Keep the answers short and focus on what is essential. Illustrate with figures only when explicitly requested.

Good luck!

Question 1: How did you initialize the clustering process and why do you believe this was a good method of doing it?

Answers:
I initialized the clusters by randomly assigning values between the RGB values 0-256 in the 3D color space. This means that we will have valid colors with the cluster centers in different places in the image spectrum.
A better alternative would perhaps have been to assign the cluster centers prior to calling the K-mean clusters algorithm, by setting the centers at the dominant colors of the image.

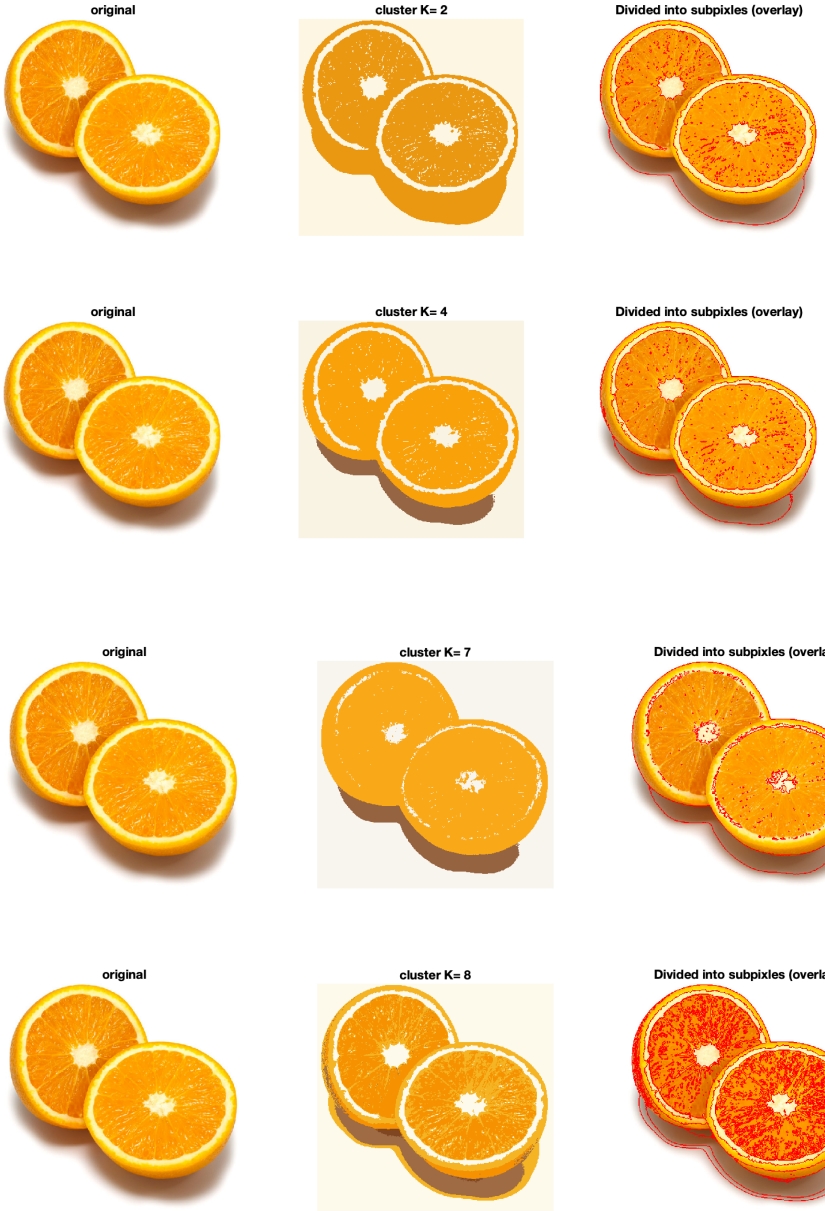
Question 2: How many iterations L do you typically need to reach convergence, that is the point where no additional iterations will affect the end results?

Answers:
The number of iterations required depends on the type of convergence, in other words how exact we want the clustering to be. The desired threshold between the difference between the previous and current clusters determines how exact we want the clustering to be.
In the code, though, I used the number of iterations, instead of the threshold, and it could be seen that the more complex the image, the more iterations is needed for getting the same result in clustering.
A more complex image with more colors need more iterations until they reach a convergence, while a simpler image doesn't need as many. The number of clusters K is also a facor for the convergence rate; higher K (many cluster centers) will result to more iterations that's needed. Since we initialize the cluster centers, it's therefore harder to determine exactly the number of iterations needed.
The Gaussian blur (defined by sigma) also has an impact on the convergence, since when we blur the image we will remove some colors, which leads to less color difference and therefor lesser amount of iterations needed to reach criteria: delta<threshold.

In my experiments, the tiger image required more iterations than the Orange image.

Question 3: What is the minimum value for K that you can use and still get no superpixel that covers parts from both halves of the orange? Illustrate with a figure.

Answers:
The results can vary quite a lot from each time we run the same simulations. But when setting K=8 (eight cluster centers) we can distinguish the two orange slices most of the time. This is since there's not much super pixels covering both halves (which might give the wrong impression that they belong segment/cluster as an object).



Question 4: What needs to be changed in the parameters to get suitable superpixels for the tiger images as well?

Answers:

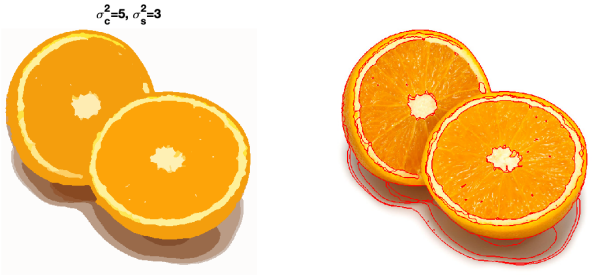
The tiger image contains more colors and variations, ad is thus a more complex image to cluster around. We would therefor need to have more cluster centers (larger K) to cluster the different parts of the image and more iterations (larger L) to reach the convergence for the center of these cluster centers.

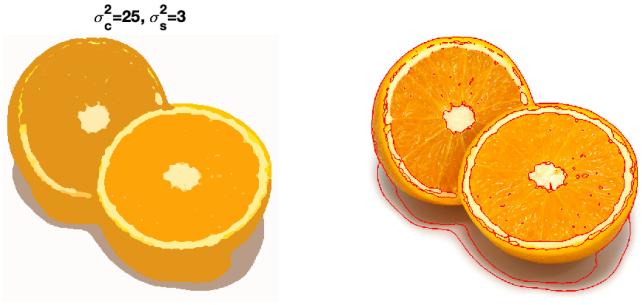
Question 5: How do the results change depending on the bandwidths? What settings did you prefer for the different images? Illustrate with an example image with the parameter that you think are suitable for that image.

Answers:
Spatial bandwidth: Higher spatial bandwidth (variance) will result in larger unicolor areas. An increased spatial bandwidth will result in a decrease in the number of modes. This is because more pixels are included in the mean calculations, due to the fact that our region on interest is larger.

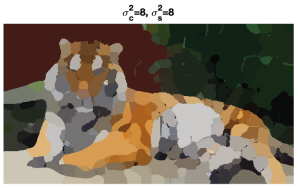
- Due to the fact that the number of modes becomes larger when spatial bandwidth (region of interest) increases, it can be said that the bandwidth affects the density function in five dimensional space.
    - Small bandwidth → more pointy peaks, larger gradient ascent → more accurate assignment of pixels to modes (position wise).
    - Large bandwidth → Flatter Gaussian, neighboring pixels blends more together → easier for assignment of pixels to modes that are more spread out.

Color bandwidth: Higher color bandwidth/variance makes the image smoother to a higher degree. Radius of color space is therefor determined by the bandwidth.

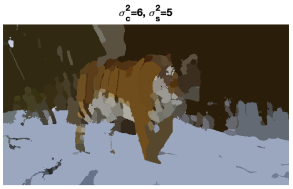


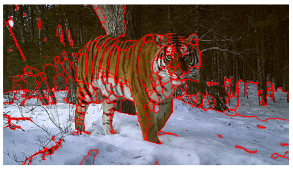


σ_c^2=8, σ_s^2=8

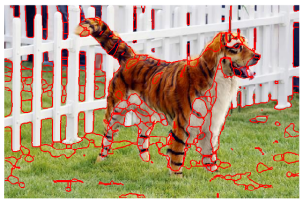


σ_c^2=6, σ_s^2=5





σ_c^2=6, σ_s^2=5



Question 6: What kind of similarities and differences do you see between K-means and mean-shift segmentation?

Answers:

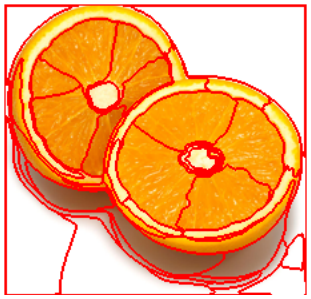
- Differences:
  - K-means doesn't consider spatial information, only information about the colors, which means that the position of the pixels. Mean shift does take this into consideration though.
  - K-means needs a pre-specified number of clusters, while mean-shift finds the number of modes, but it requires pre-defined bandwidths.
  - K-means has a high sensitivity to outliers, while mean-shift is not as sensitive to them.

- Similarities: Both are
  - Both are iterative methods
    - K-means updates its cluster centers, according to its mean colors.
    - Mean shift updates its position with respect to where the maximum of local density is located.
  - Both K-means and mean shift are used in segmenting images, treating the colors and pixels from a probability distribution.

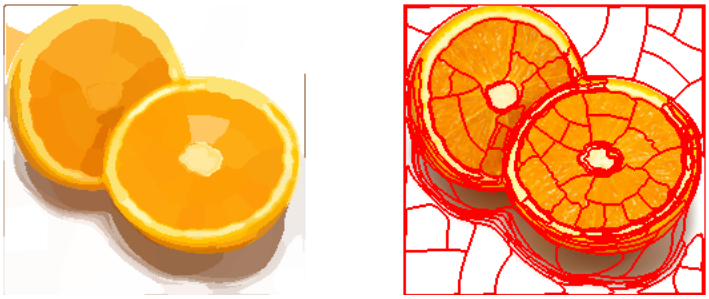
Question 7: Does the ideal parameter setting vary depending on the images? If you look at the images, can you see a reason why the ideal settings might differ? Illustrate with an example image with the parameters you prefer for that image.

Answers:
The image will definitely be affected by the ideal parameters.

- **ncuts_thresh**: sets a maximum cutting cost that we accept when deciding to make cut on vertices between edges. Larger n_cuts_thresh → more similar areas are OK to seprate apart.
- **min_area**: Minimum limit on how small (area of) segments are accepted.
- **Max_depth**: Number of recursive call that are made, i.e. amount of cuts that will be made. Increase max_depth → more segments.
- **Color bandwidth**: Affects the weight between similar pixels:
  - Low Bandwidth → Larger weigths for similar pixels and low weights for unsimilar pixels.
  - High Bandwidth → Larger variance in Gaussain – Decreases the weights of similar pixels and increase weight for unsimilar pixels → Affects cost of cut and segmentation.
- **Radius**: Decides size of the neighboring area of each pixel. Larger radius → include pixels even further away into account, will give fewer segments → higher time complexity.

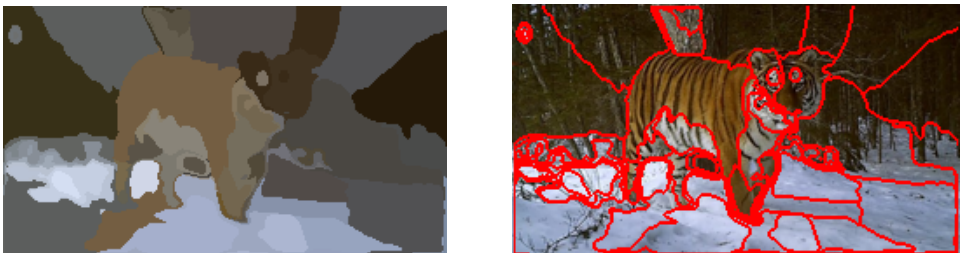



Orange: Color bandwidth=20, radius = 15, ncuts_thresh=0.5, min area 60, max_depth = 6

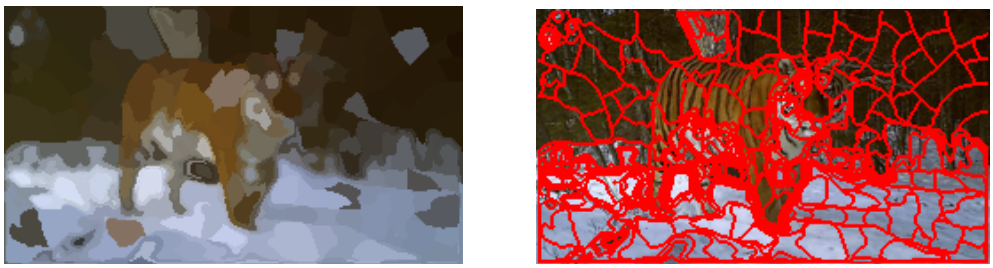


Orange: Color bandwidth=20, radius = 15, ncuts_thresh=0.5, min area 60, max_depth = 10

Tiger1: Color bandwidth=15, radius = 10, ncuts_thresh=0.5, min area 18, max_depth = 6



Tiger2: Color bandwidth=15, radius = 10, ncuts_thresh=0.5, min area 18, max_depth = 6



Tiger2: Color bandwidth=15, radius = 10, ncuts_thresh=0.5, min area 18, max_depth = 12

Tiger3: Color bandwidth=11, radius = 10, ncuts_thresh=0.5, min area 8, max_depth = 10.

Question 8: Which parameter(s) was most effective for reducing the subdivision and still result in a satisfactory segmentation?

Answers:
The parameters that has the most effective impact for reducing the subdivisions and in a satisfactory result was:

- max_depth
- ncut_thresh
- min_area

Question 9: Why does Normalized Cut prefer cuts of approximately equal size? Does this happen in practice?

Answers:
When we use the depth (recursive call) we cut the pieces into even smaller pieces and they will therefor be cut approximately in the middle IF we only take the recursive call into account. But when we use the other parameters, the proportionality will vary. So in practice not all areas will be equal.

Theoretical explanation for this can be shown with
assoc(V) = assoc(A,V) + assoc(B,V) – cut(A,B) (i)

Ncut(A,B) = cut(A,B)/assoc(A,V) + cut(A,B)/assoc(B,V) (ii)

Derive (ii) with respect to Assoc(V) → -2*accoc(A,V) + assoc(V) + cut(A,B) = 0 →

2*assoc(A,V) = assoc(V) + cut(A,B) → assoc(A,V) = 0,5*(assoc(V) + cut(A,B))

{Substituting (i) into this} → assoc(A,V) = 0,5*(assoc(A,V) + assoc(B,V)) →
0,5*assoc(A,V) = 0,5*assoc(B,V)

Question 10: Did you manage to increase *radius* and how did it affect the results?

Answers:
Increasing the radius resulted in a more computations for the algorithm, (longer time to calculate), since we take into account more neighboring pixels in the calculations. This gives larger segments, but the color was slightly not correct in some places.

Question 11: Does the ideal choice of *alpha* and *sigma* vary a lot between different images? Illustrate with an example image with the parameters you prefer.

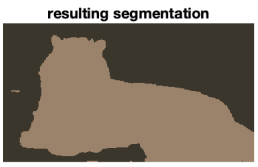
Answers:
Alpha: maximum cost of an edge

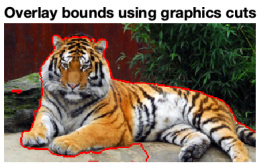
- Increase alpha will increase maximum cost of an edge → More difficult to cut between similar pixles or smooth surfaces. (because they have higher costs).

Sigma: Speed of decay for decreasing similarities between neighboring pixels.

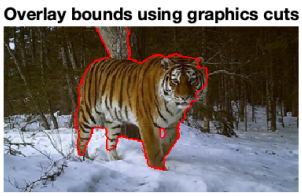
- Decrease sigma → Decrease in a pixel's similarities from neighboring pixels. In other words, the cost decays even more → Easier to do more simplified cuts along strong edges.

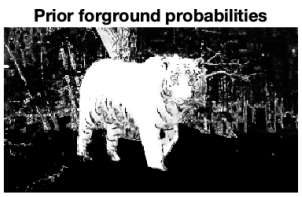
Decreasing both parameters makes the cut become more sensitive and decreased accuracy of segmentation. Because we are more okay with cutting similar pixels (high edge value/cost).



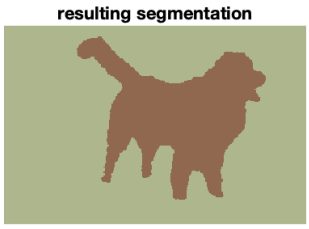


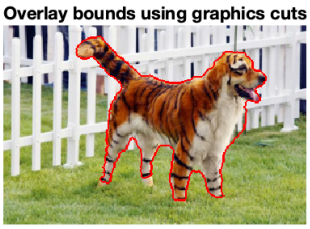
Tiger1: alpha = 15, sigma =10.





Tiger2: alpha = 6, sigma = 35.





Tiger 3: alpha = 15, sigma = 10.

Question 12: How much can you lower K until the results get considerably worse?

Answers:
K=3 (Gaussian components) still produces acceptable results. Less than 3 gives a much worse segmentation. K=2 means we only look at two segments/blocks in our gaussian variation smoothness, which does not make a good representation of the original mage.

Question 13: Unlike the earlier method Graph Cut segmentation relies on some input from a user for defining a rectangle. Is the benefit you get of this worth the effort? Motivate!

Answers:
Depending on the information in on image it may make a difference. If there for instance is some clear object that has a clear bounded area where we can set a rectangle to defines the area, then it helps. But there might be no specific object involved that we want to apply the algorithm on and we instead have a mixture of foreground and background. For example if we have a image of a landscape, or a sunset image, the bounding rectangle might not be so useful.

Question 14: What are the key differences and similarities between the segmentation methods (K-means, Mean-shift, Normalized Cut and energy-based segmentation with Graph Cuts) in this lab? Think carefully!!

Answers:
Similarities:

- They are all cluster based, meaning they group the information in the image to different clusters (mostly colors) that pixels might belong to.

- All methods separates the similar pixels from those that are not similar by lustering those that are similar in on cluster and everything else in other clusters.
- Graph cut and mean shift both uses Gaussian distribution in data modeling.
- Normalized cut and Graph cut are both graph based, meaning they model the image as a graph by setting pixels to vertices and edges as connections between pixels.

Differences:

- Graph cut is the only one that requires information prior to the algorithm, because it need to optimize the results based on the ration between foreground and background. Normalized cut does not on the other hand need such prior information regarding the image.
- K-means only look at the coloring dimension (RGB) and does not care about spatial information. The other methods (mean shift and the graph- based methods) also takes spatial information into account, prioritizing pixels closer to each other as well as colors.
    - K means can therefor give results that stretch colors across different objects.