Answers to questions in
Lab 3: Image segmentation

Name: Marcus Nystad Öhman, Jacob Malmberg Program: CINEK, MAIG, TIEMM1

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:

We initialized using the Forgy method, with a slight modification. That is, we chose k pixels randomly and used the colors of these as initial means. We then made sure that the randomly chosen pixels were not too close to each other, so that the initial means would not be too close too each other. We believe that this is a good method because if the picture is 70% orange, 20% blue and 10% white, we have a 70% chance of getting an orange pixel as an initial mean. Thus, it should somewhat equal the method of manually setting the initial means to the color distribution of the image, only now we don't have to do it manually.

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?

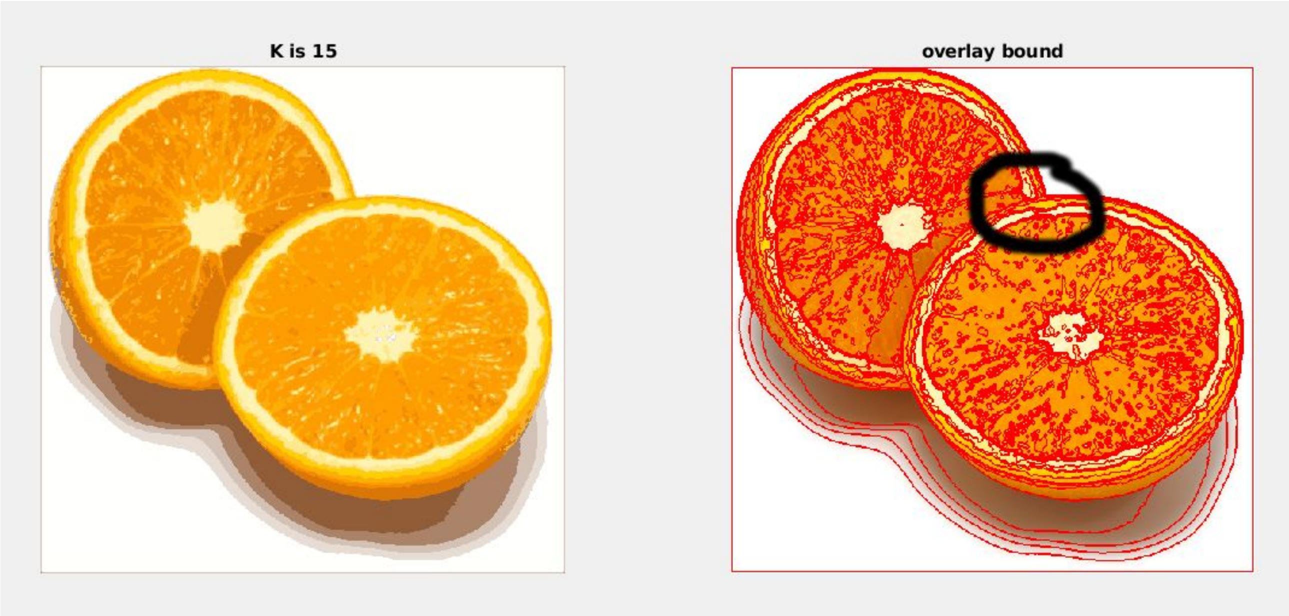
Answers:

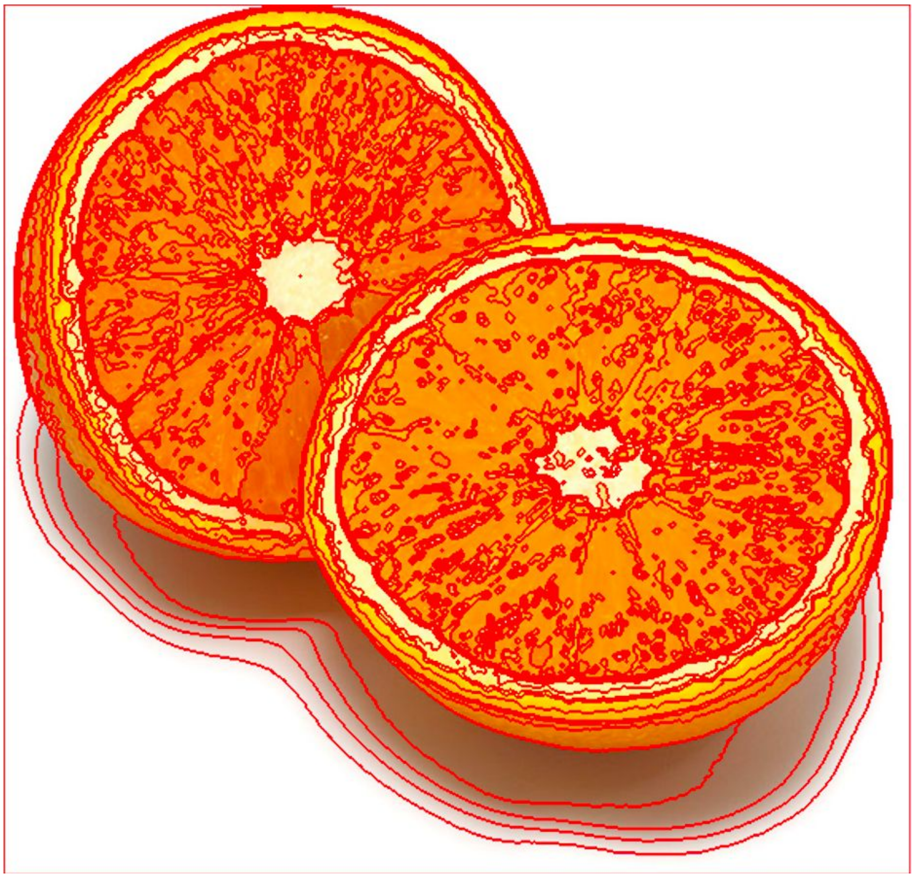
k	orange.jpg	tiger1.jpg	tiger2.jpg	tiger3.jpg
2	7	11	6	4
6	16	29	106	107
8	38	73	136	141

To check convergence, we check that the cluster center that changed the most moved less than some threshold. Generally, larger k gives longer time to convergence.

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:





K = 15. Otherwise, we get a superpixel that covers both parts of halves. It happens in the black circled area above.

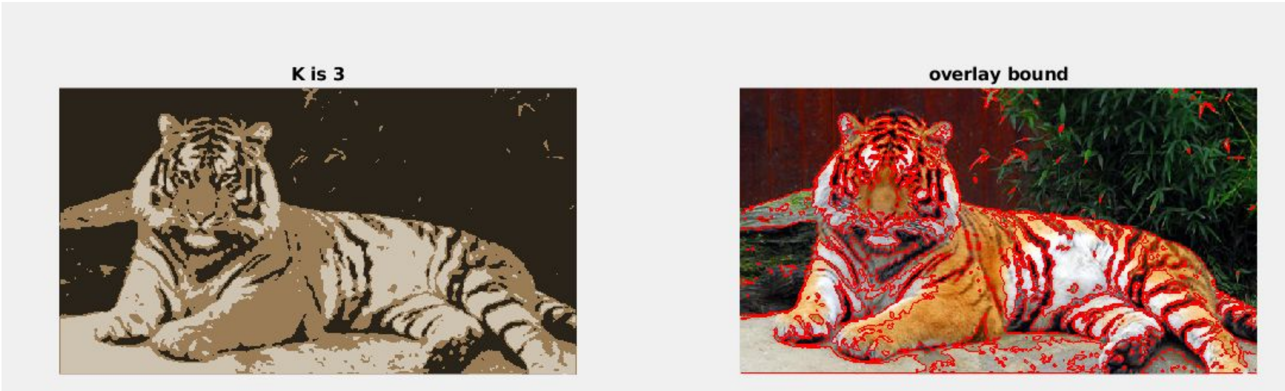
If we use K = 15 for tiger1.jpg, we get far too many segments.

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

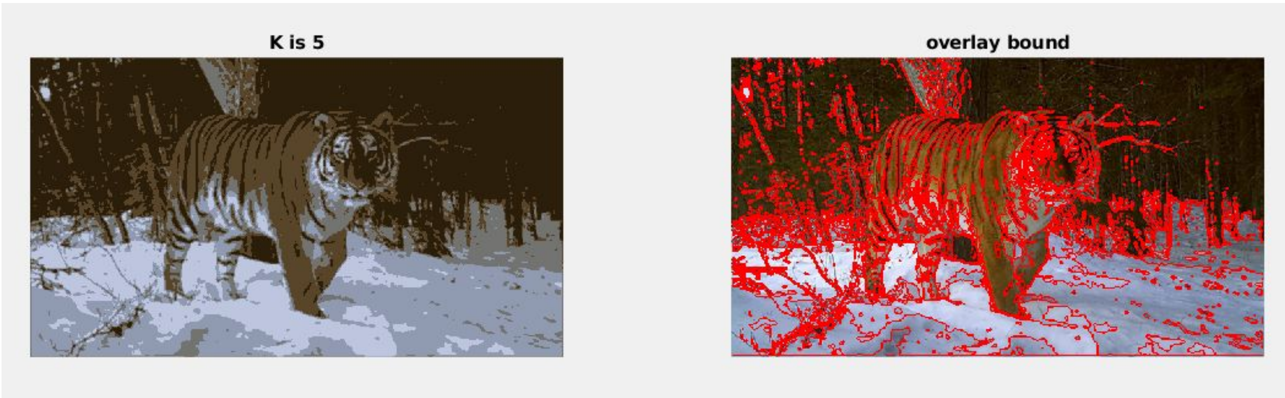
Answers:

Since the tiger1 image have more distinct colors and features compared to the oranges, we should lower K and L to get a good separation of the tiger from the background.

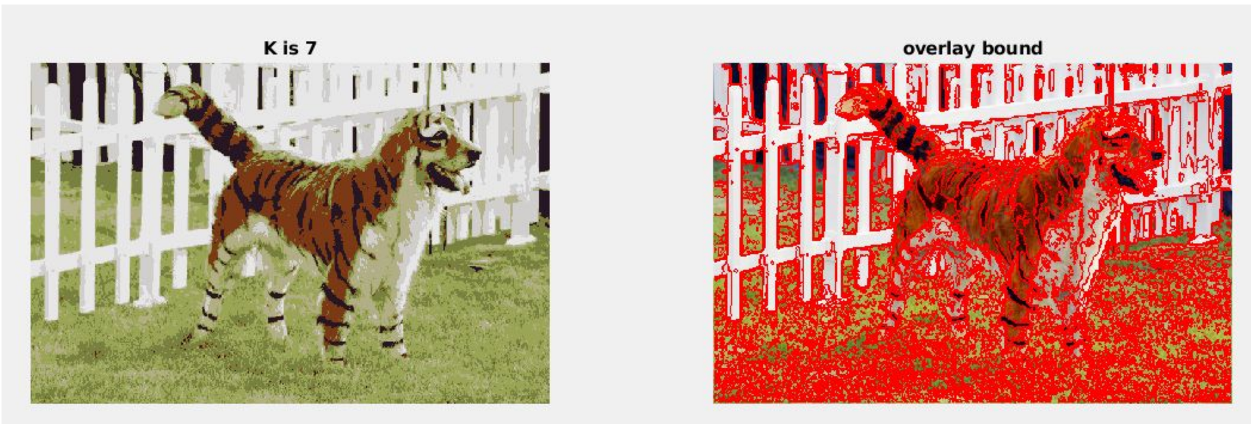
For k = 3, we get a pretty good superpixel for the tiger1:



Tiger2.jpg has less pronounced colors compared to tiger1.jpg, so K and L was increased slightly compared to tiger1.jpg.



Tiger3.jpg. To reasonably be able to distinguish the “tiger” from the fence, K was set to 7 and L was increased.



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.

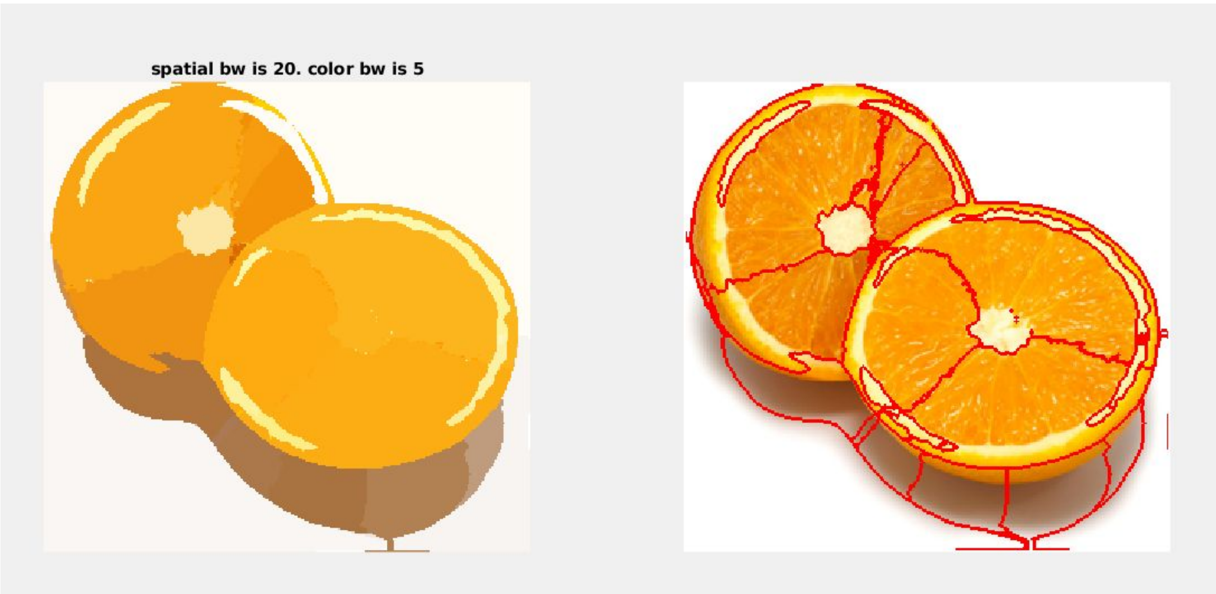
Answers:

By increasing the colour or the spatial bandwidth, we get fewer segments. Increasing the colour bandwidth over 50 for tiger1.jpg does not decrease the amount of segments. This may be because the colours in this image do not have a large span.

For tiger1.jpg we prefer a spatial bw of 10 and a color bw of 50.
For tiger2.jpg we prefer a spatial bw of 10 and a color bw of 5
For tiger3.jpg we prefer a spatial bw of 50 and a color bw of 2.

Theoretical musings: If the picture is segregated by color, like tiger3, we want a lower color bw to segment. For tiger1.jpg since the tiger has a very different color than the bg, we can have a higher color bw without risking the tiger being segmented with the bg.

For orange.jpg we get pretty good results. For the orange, we want a small color bw because the colors are so similar to better distinguish between the halves.



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

Answers:

K-means as we programmed it in this lab only takes color into account when segmenting. Mean shift takes both color and spatial information into account. For an image where both segments have the same color, as in orange.jpg, we therefore get better results with mean shift, since the segments are segregated spatially.

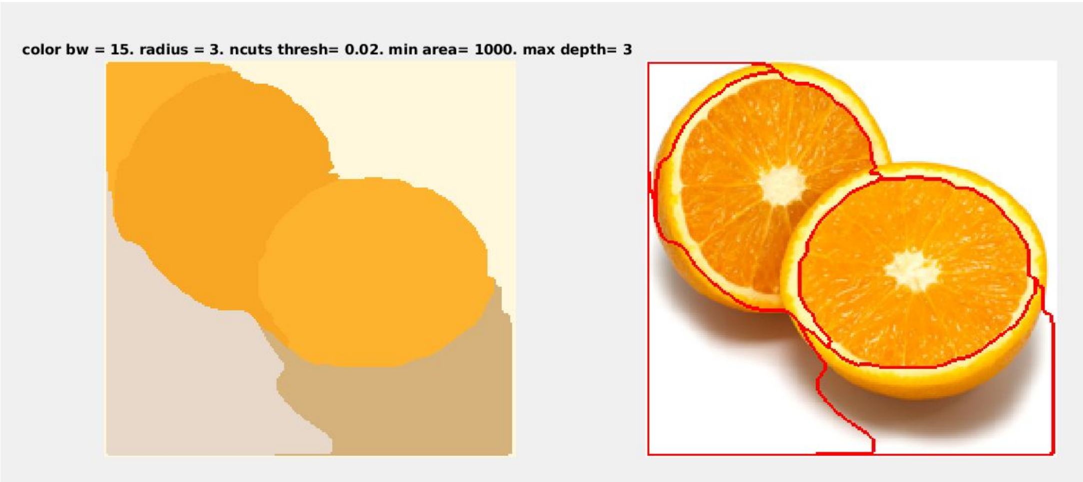
Mean shift segmentation is computationally slower than k-means. Also, k-means iterates over all pixels in a batch, while mean shift iterates over each pixel separately. K-means predefines the number of clusters, while mean shift does not. Mean shift more robust to outliers. Kmeans is sensitive to initialization, mean shift is not senitive to initialization.

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:

radius: If we decrease radius, the neighbourhoods get smaller.
ncuts_thresh: If the Ncut(A,B) > threshold, we do not cut. So, if we cannot find a cut small enough in the graph, we do not cut.
min_area: Controls the minmum size of a segment
max_depth: limits the depth of recursion.
color_bandwidth: Affects the weight/affinity matrix. With high color bw, the relative difference in affinity is smaller between pixels of different colors. High bw = cares less about color.

Yes, the ideal parameters vary depending on the image. If we have large segments, we will want a large min area to avoid subdivision. Lower color banfdwidth -> carers more about color. Segments with relation to color better.



If want one segment for each orange half, we want a high min area. If we want to include the white centers of the halves we need a high color bw.

The tigers are multicolored. With a constant radius, we cannot have just one segment for the tiger because it is multicolored. Instead, we want a low minimum area with low color bw so we get many different segments for the different colors on the tiger. That means we should allow more segments, we should increase max depth.

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

Answers:

For the orange, increasing the min area so that one area covers an entire orange half. Otherwise, we found that decreasing the max_depth and ncuts_thresh works well for reducing subdivision, which makes sense intuitively.

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

Answers:

The equation of Ncut is

$$Ncut = \frac{cut(A, B)}{assoc(A, V)} + \frac{cut(A, B)}{assoc(A, V)}$$

We want to find the extreme points of Ncut(A,B). We therefore differentiate Ncut with relation to assoc(A,V).

Let n be the total number of edges in the graph. This gives

$$n = assoc(A,V) + assoc(B,V) - cut(A,B)$$

$$\Rightarrow assoc(B,V) = n - assoc(A,V) + cut(A,B) \qquad (*)$$

The expression for Ncut(A,B) then becomes

$$Ncut(A, B) = \frac{cut(A, B)}{assoc(A, V)} + \frac{cut(A, B)}{assoc(B, V)} = \frac{cut(A, B)}{assoc(A, V)} + \frac{cut(A, B)}{n - assoc(A, V) + cut(A, B)}$$

Now, we differentiate this with relation to assoc(A,V).

For the sake of simplicity, in the equations below we have used
assoc(A,V) = x and cut (A,B) =x

$$\frac{d}{dx}\left(\frac{c}{x}+\frac{c}{n+c-x}\right)=0 \quad = \frac{c}{(c+n-x)^2}-\frac{c}{x^2}$$

$$= c\left(\frac{1}{(c+n-x)^2}-\frac{1}{x^2}\right)$$

$$= c\left(\frac{x^2-(c+n-x)^2}{(c+n-x)^2x^2}\right)$$

$$= c\left(\frac{x^2-(c^2+2cn+2cx+n^2-2nx+x^2)}{(c+n-x)^2x^2}\right)$$

$$= -c\left(\frac{c^2+2cn+2cx+n^2-2nx}{(c+n-x)^2x^2}\right)$$

$$= -c\left(\frac{(c+n)(c+n-2x)}{x^2(c+n-x)^2}\right) \quad = 0$$

Either $-c$ or $(c+n)=0$. These are not interesting cases. This leaves

$$c+n-2x=0$$

$$\Rightarrow x = \frac{n+c}{2}$$

With $x = assoc(A,V)$ and $c = cut(A,B)$ this becomes

$$assoc(A,V) = \frac{n+cut(A,B)}{2} \qquad (\#)$$

Putting (#) into (*) gives

$$assoc(B,V) = n - \frac{n+cut(A,B)}{2} + cut(A,B)$$

$$= \frac{n+cut(A,B)}{2} = assoc(A,V)$$

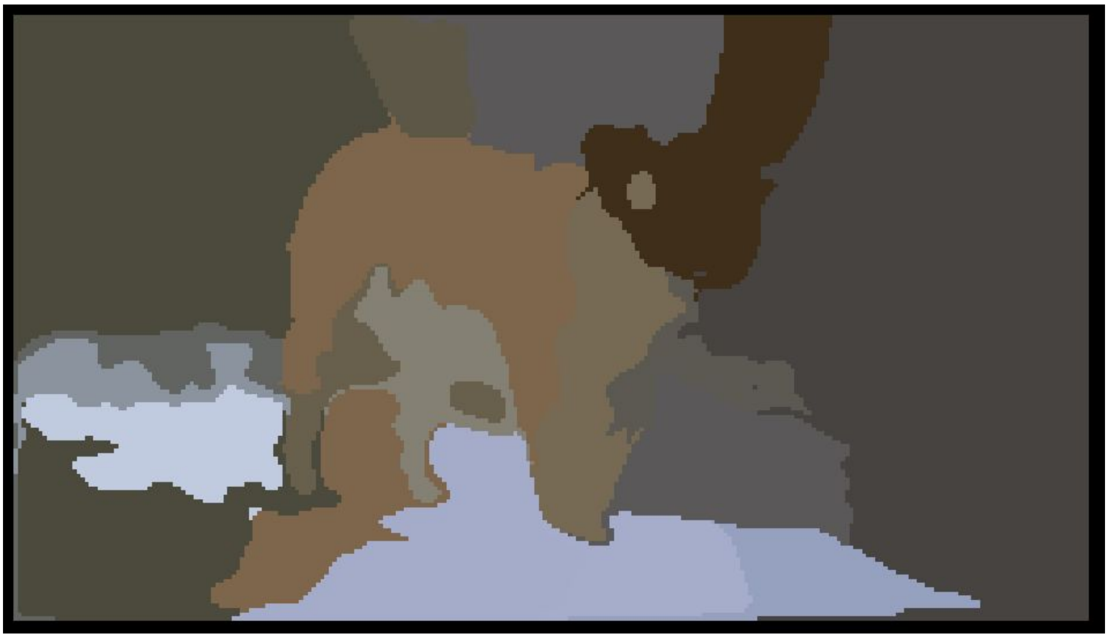
Since assoc(B,V) = assoc(A,V), we have that Ncut(A,B) tries to minimize the cut by finding cuts of equal size. In our experiments, this did not happen exactly, but somewhat.

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

Answers:

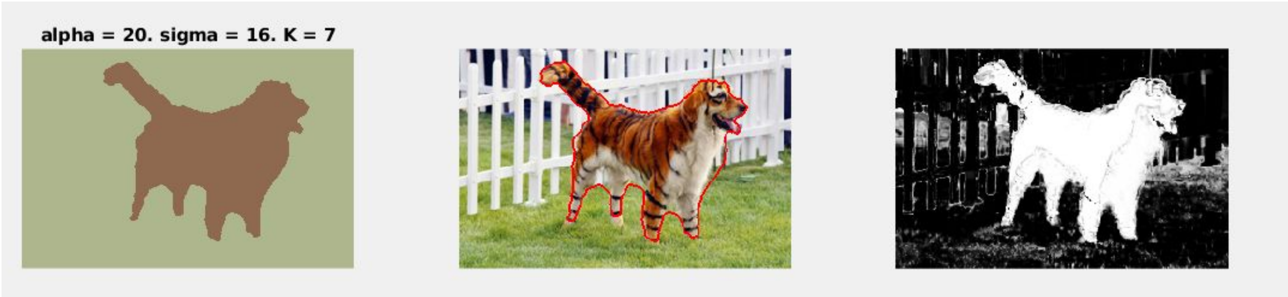
Yes. The computational time increased greatly. With radius = 20, it was very slow. However, by increasing radius increased the neighbourhood to include pixels that are a bit further away from each other. The orange back/body of the tiger became one segment instead of being cut up into multiple segments.

We think that we should use a big radius if we want large segments, since we better can see similarities between pixels far away. Then the probability that we segment them to the same segment should then be higher.



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:



For the orange, a=12 and sigma = 16 gives a nice segmentation. For tiger3.jpg, we prefer alpha = 20 and sigma = 16. If we increase alpha, we generally get better edges. We want neighboring colors to have more impact. For orange, we can use a pretty low sigma, say sigma=4, which for the tiger gives bad results. It seems that if we have very clear edges, we can use a lower sigma. It is generally hard to isolate just these variables, it depends on K too.

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

Answers:

K is the number of gaussians. We can think of the number of gaussians as the number of colors we can model and mix into one color for a pixel.

For tiger3.jpg , this means that K =5 gives considerably worse results.

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:

The results with Graph Cut are much better than with the other methods in this lab. The other results are not really useful, while the segmentation with GC is really good.

If we have multiple objects, and not just foreground/background, then one of the earlier methods may be better, since we really only get foreground/background with GC.

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:

Graph Cut works well when the object of interest has another color distribution compared to the background. If the object of interest has similar color distribution as the background it may not work well.

Ncut tries to segment into equally large parts, which kmeans/meanshift/graphcut does not.

Both segmentation metods and energy-based methods are similar in the sense that they try to segment an image into different clusters/segments.

GC is far more advanced and needs more memory than the other methods. In GC the user also has to predefine a foreground/background. In K-means the user predefines the number of clusters, which is different from predefining the foreground/background. The segmentation metods can divide into more than foreground/background (more clusters). GC and Ncut looks at the segmentation from a graph perspective, which Kmeans and Mean-shift does not.

K-means as we programmed it in this lab only takes color into account when segmenting. Mean shift takes both color and spatial information into account. For an image where both segments have the same color, as in orange.jpg, we therefore get better results with mean shift, since the segments are segregated spatially. Mean shift also allows for different bandwidths for different parameters, which allows it to have clusters that are not circles but rather ellipses etc.

Mean shift segmentation is computationally slower than k-means. Also, k-means iterates over all pixels in a batch, while mean shift iterates over each pixel separately. K-means predefines the number of clusters, while mean shift does not.