Lab 3: Image segmentation

Carina Wickström CINTE17
Saman Rassam CDATE17

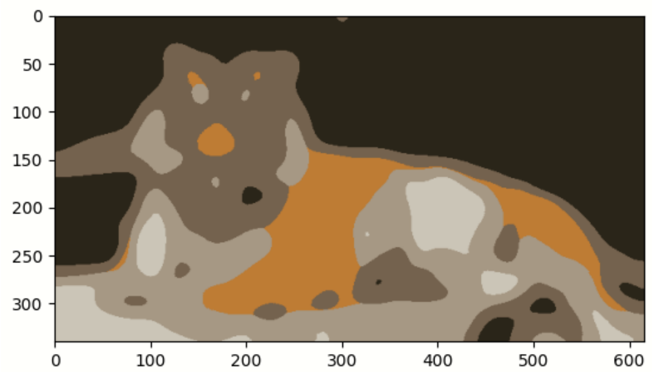
K-means clustering

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

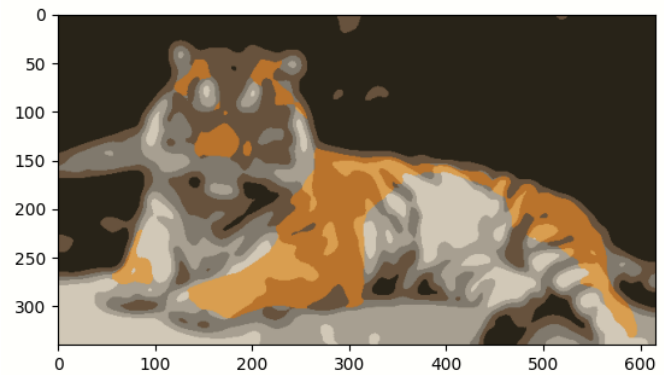
We tried two different methods. One where we sampled K pixels from the original image and one where we randomly initialized K RGB vectors. We could not notice any significant difference between the two methods. We believe that it would not suffice to initialize the super pixels to static values as it is possible that only one of the pixels get changed and the other become dead units. For instance, if the image is very light and we initialise all super pixels to be black, then it is possible that only one super pixel gets updated.

Try modifying the K and L parameters, the scale factor and the amount of pre-blurring, to see how the changes affect the results

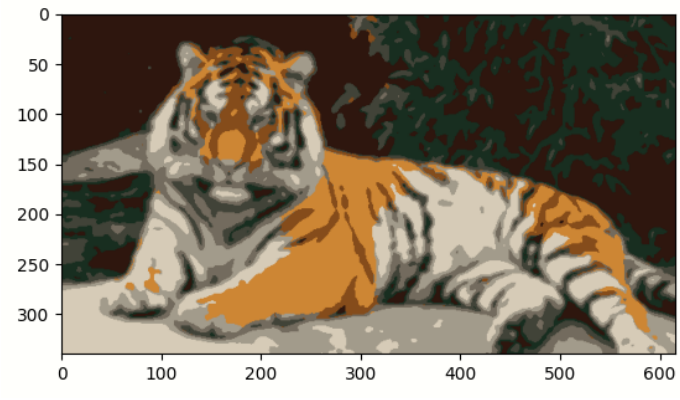
We try different image_sigma GaussianBlurs for the tiger with k=10 and L=5 below. It removes the noise but it also removes the details.



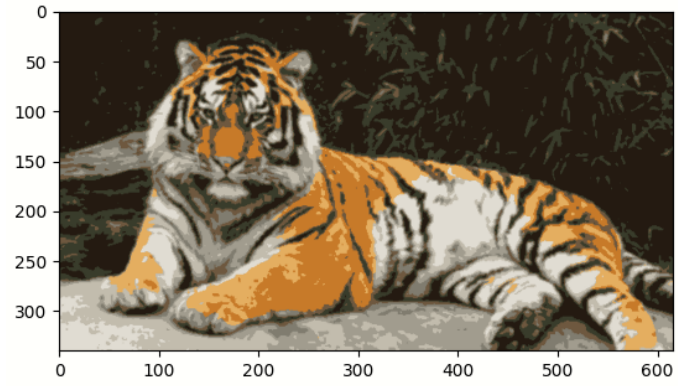
image_sigma = 10



image_sigma = 5

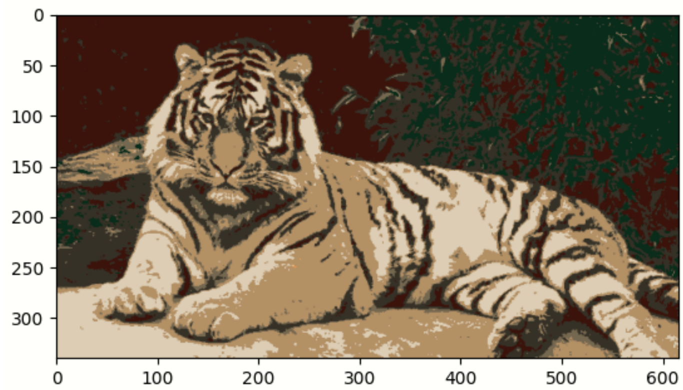


image_sigma = 2

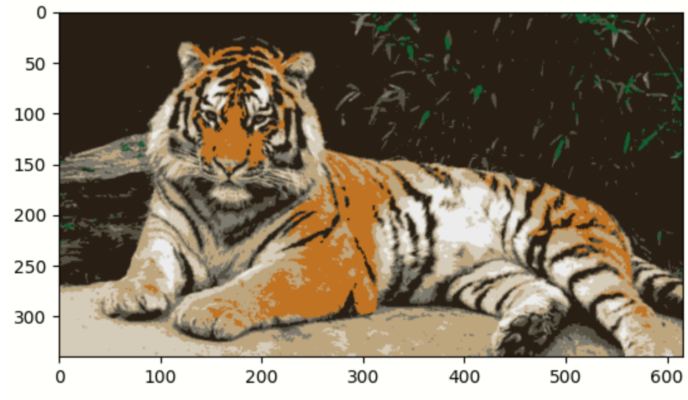


image_sigma = 1

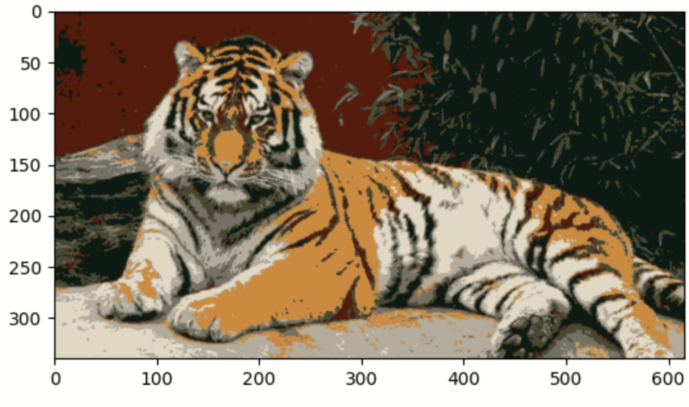
We further want to investigate how the number of iterations affect the resulting colors and number of segments. Now we consider different number of iterations L for image_sigma=0.5 and K=10



L=1



L=2



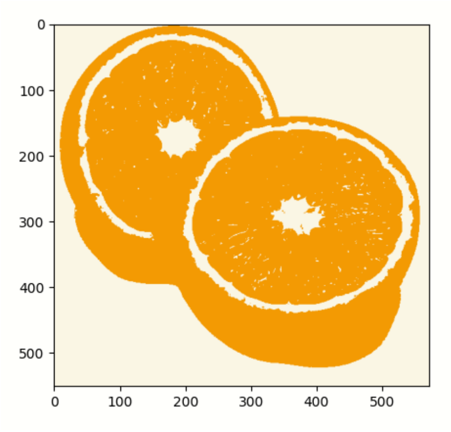
L=3

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?

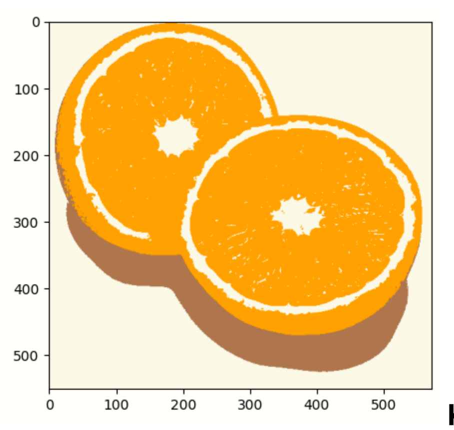
We have tried different values for L and K to experimentally determine when the image converges. The convergence happens when no pixel changes segmentation.

When we increase K, the number of required iterations go up.
For K=2 we converged at L=7
For K=4 we converged at L=21

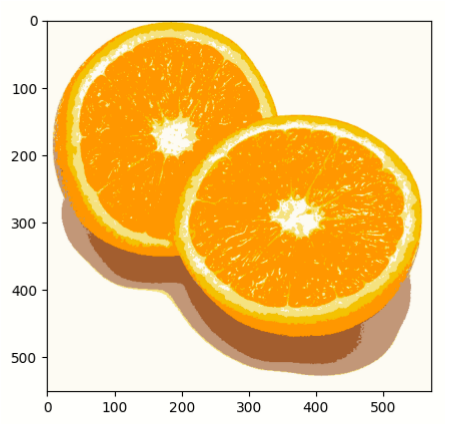
Question 3: What is the minimum value for K that you can use and still get no superpixel that covers parts from both halves of the orange? Illustrate with a figure. Try using parameters suitable for orange.jpg and see how these affect the tiger images.


0
100
200
300
400
500
0
100
200
300
400
500

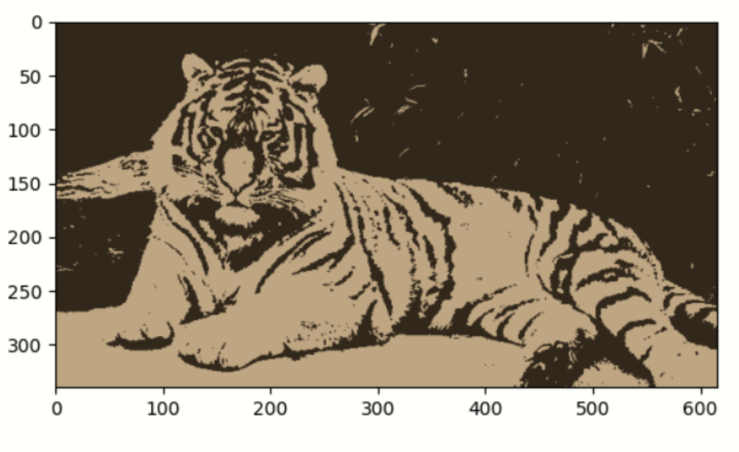
K=2


0
100
200
300
400
500
0
100
200
300
400
500

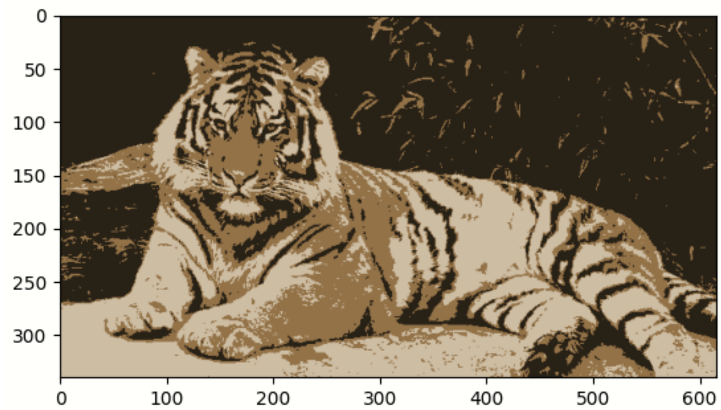
K = 3


0
100
200
300
400
500
0
100
200
300
400
500

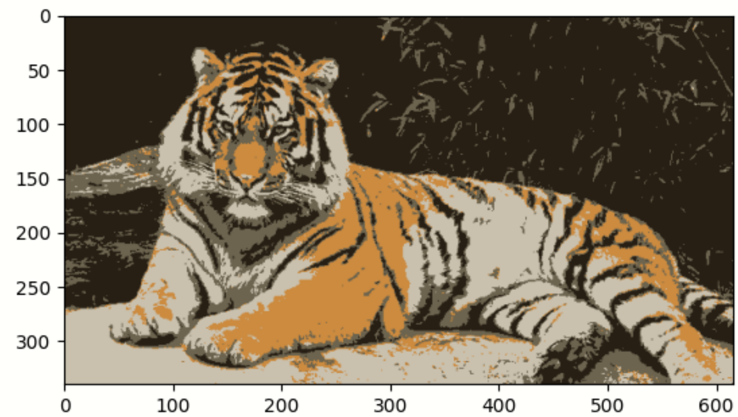
K=16


0
50
100
150
200
250
300
0
100
200
300
400
500
600

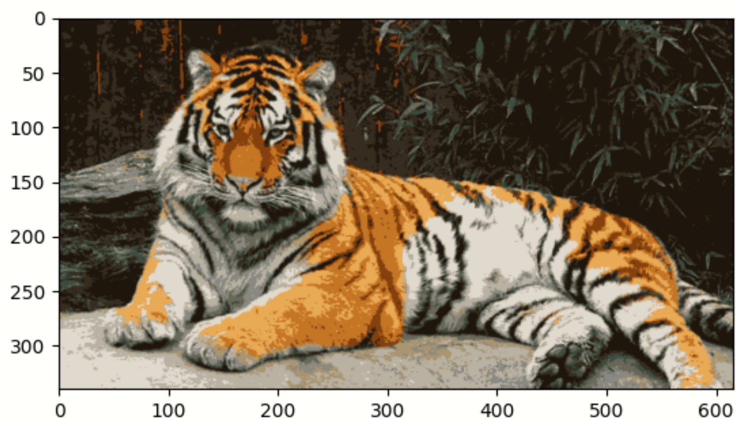
K=2


0
50
100
150
200
250
300
0
100
200
300
400
500
600

K=3


0
50
100
150
200
250
300
0
100
200
300
400
500
600

K=6


0
50
100
150
200
250
300
0
100
200
300
400
500
600

K=16

Case orange:
We can tell that, even for large K, the two orange halves still end up the same shade of orange. This is because the nuances of the two halves are very similar, and even if we have a large K, the halves are so similar that the same superpixel adopts both halves.

Case tiger:
In this case, the different colors in the image are very different from each other, we have about 6 dominant different colors (orange, green, red, gray, white, black). However, if we set k=6, it is possible that we get two similar superpixels and hence we don't usually get all the different colors. Therefore we need a bigger K anyway.

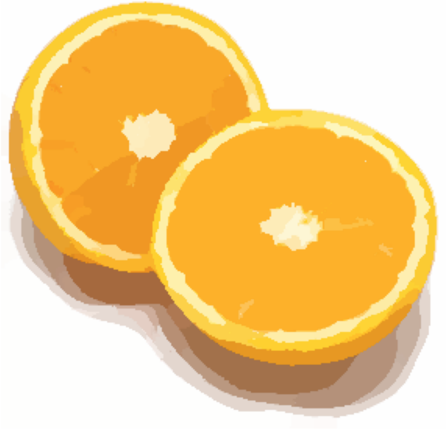
Question 4: What needs to be changed in the parameters to get suitable superpixels for the tiger images as well?

Because the tiger is a more complex structure with visibly more segments, we want a larger K because it represents the number of segments in the image. The tiger also contains more details. If we're interested in the general shape, and want to ignore the details, it also increases the need to pre-blur the image.

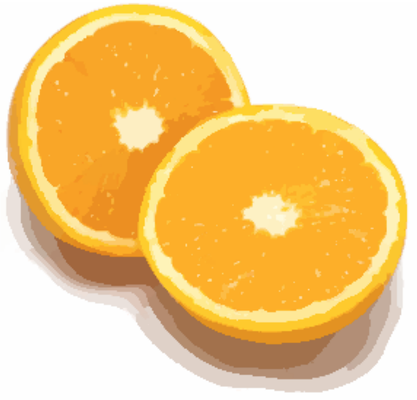
Mean-shift segmentation

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.

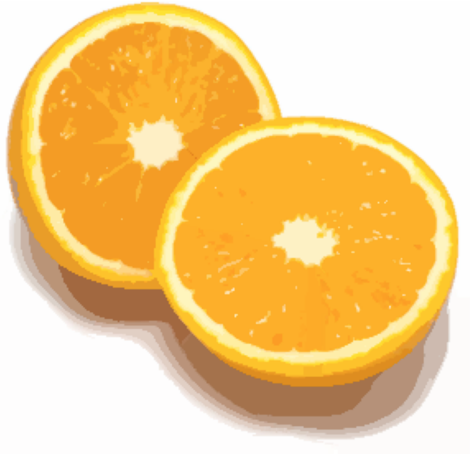
scale: 0.5, sigma:1.0, spatial: 5.0, color:20.0



scale: 0.5, sigma:1.0, spatial: 5.0, color:5.0



scale: 0.5, sigma:1.0, spatial: 20.0, color:5.0



The images above uses the default k value of k = 16

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

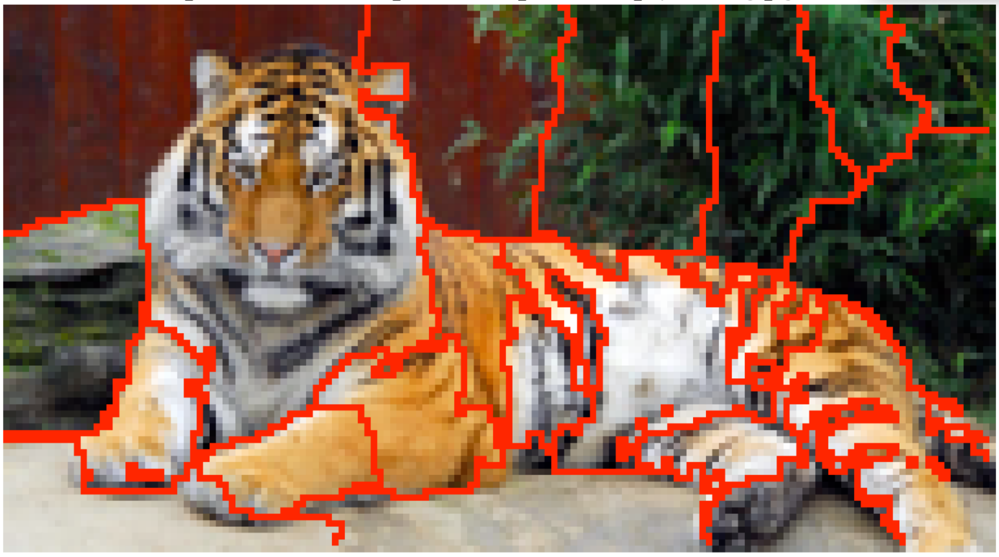
We see that both K-means and mean-shift segmentation does a good job to cluster the image. The K means algorithm expects us to know in advance the number of clusters/segments in the image. The mean-shift algorithm clusters the image from an unknown number of modes (cluster centers), and it takes into account the proximity of the segments. We can see from the results that there is a more clear difference between the orange halves in the mean-shift method compared to the k-means where both of the halves look very similar.

Normalized Cut

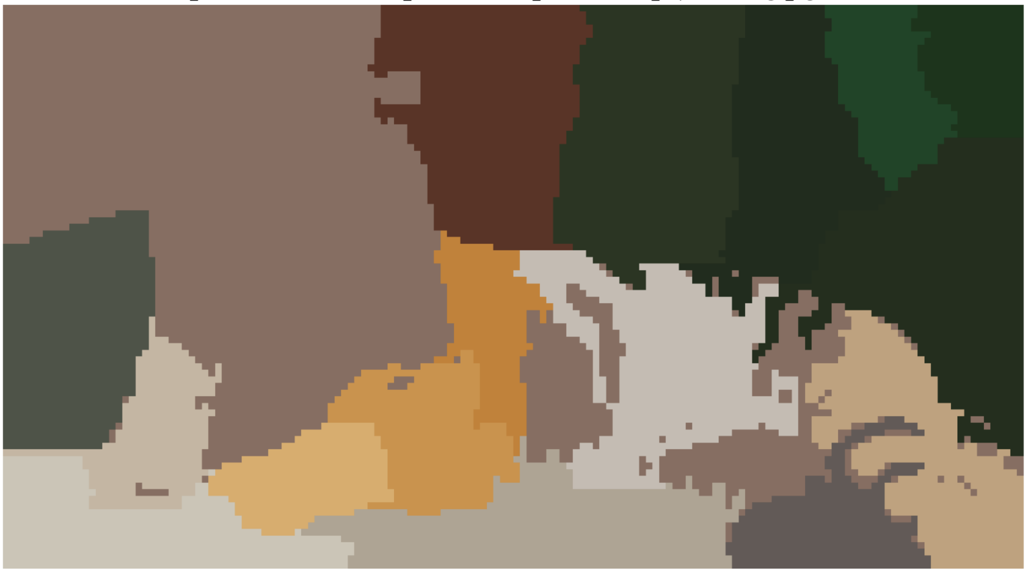
Question 7: Does the ideal parameter setting vary depending on the images? If you look at the images, can you see a reason why the ideal settings might differ? Illustrate with an example image using the parameters you prefer for that image.
As seen below for the tiger3 image compared to tiger1 the same parameters could not yield equally good results for both pictures.

Talking:

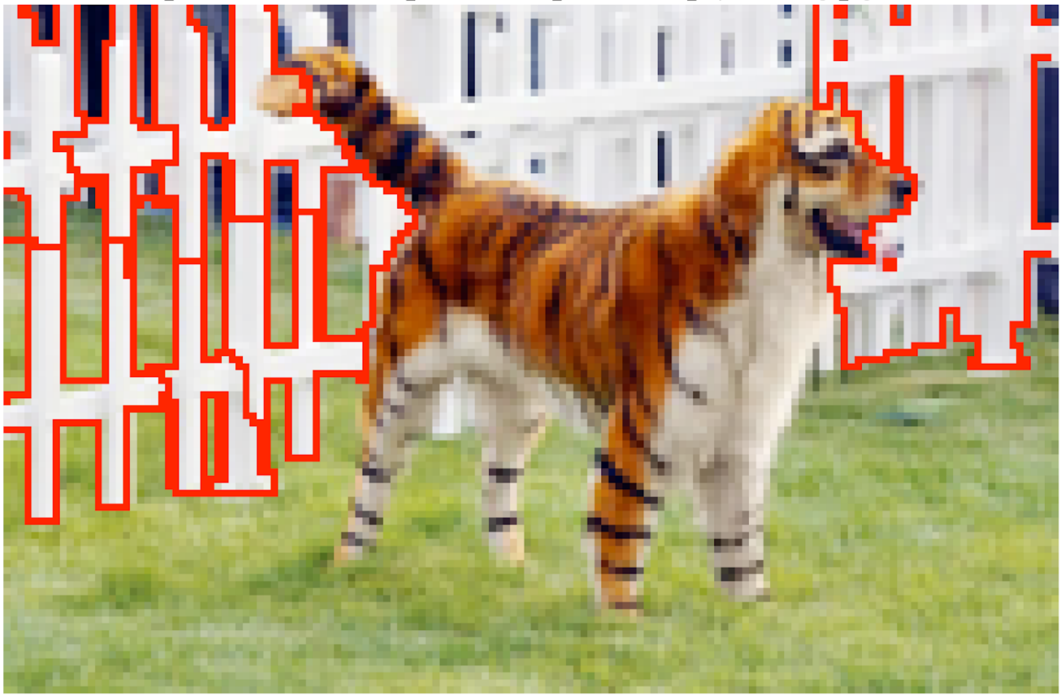
color_band: 20.0, radius:1, ncuts_thresh: 0.1, min_area:200, max_depth:12, image_sigma: 0.5



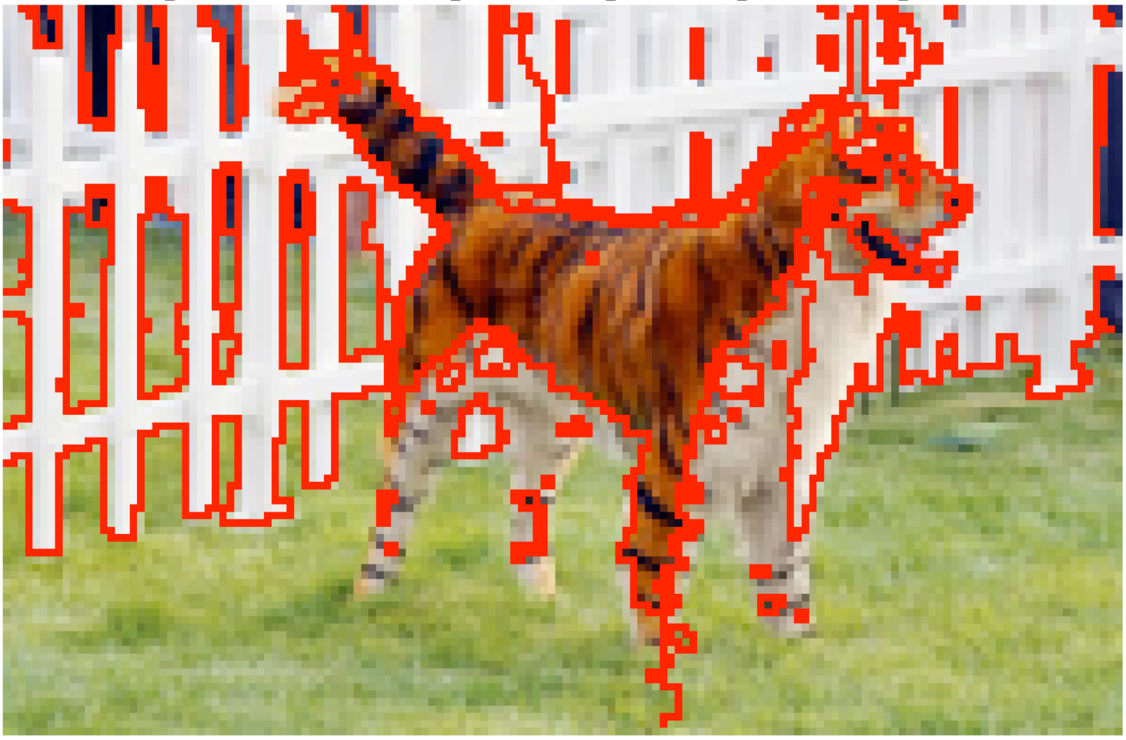
color_band: 20.0, radius:1, ncuts_thresh: 0.1, min_area:200, max_depth:12, image_sigma: 0.5



color_band: 20.0, radius:1, ncuts_thresh: 0.1, min_area:200, max_depth:12, image_sigma: 0.5



color_band: 60.0, radius:40.0, ncuts_thresh: 0.5, min_area:60, max_depth:12, image_sigma: 0.5



Question 8: Which parameter(s) was most effective for reducing the subdivision and still result in a satisfactory segmentation?
Changing the min_area, max_depth, color_bandwidth seems to give smaller subdivisions and still yield good results.

After we experimented with the parameters we found that color bandwidth and radius made the biggest difference in finding a good overlay.

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

We recursively split the segments into smaller segments. If the image contains equal amounts of details in all different places on the image, we want equal sizes of the cuts in order to catch details of equal sizes.

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

Yes, it takes significantly longer to run which is of course because we include more pixels to calculate and we can therefore get larger segments (segments with larger radius).

Experiments

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.

Theory what is alpha and sigma, related to the specific image.

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!

The human is very good at approximating where the object is, so it is very little effort for the human in order to effectively determine the mask for where the object is. Therefore I think it's worth the effort, but it depends on the use case.

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!!

They all look at each pixel and try to assign it to a more general value that the pixel matches. However, the rules for the assignment vary between the methods. K-means is of course the most basic and the energy-based segmentation is the most advanced. Furthermore, k-means, mean-shift and normalized cut can all be deterministic while the energy-based cut is stochastic.