Answers to questions in Lab 3: Image segmentation

Name: ________________ *de Gibert, Xavi* AND *Laso, Pablo* ______________________________
Program: ______ *DD2423 CVHT21 HT21-1 Image Analysis and Computer Vision* ________

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!

2. Means clustering

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

Answers:

By using `centers = randi(255, K, 3);`, we initialize each cluster center between 0 and 255, i.e., the range of pixel values in our image. We did so in order to obtain a random, yet valid, pixel value within our 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:

We set a maximum number of possible iterations to 500 (we chose a high number so as not to fall short). However, should the absolute value of the difference between the previous center and the newly-defined center be any higher than 0.001, the loop will break without fully reaching convergence.

Depending upon the image complexity (e.g. wider color range), convergence shall be harder or simpler to attain. Likewise, a higher number of clusters will stall algorithm convergence. Contrarily, a higher sigma (more blurring) may very well decrease the complexity of the problem, leading to a lower number of loops until the algorithm is eventually able to converge. That said, the results obtained for each image file are as follows[1]:

image filename	# loops until convergence
tiger1.jpg	47
tiger2.jpg	83
tiger3.jpg	120
orange.jpg	73

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:

It is a difficult area to segment. We can observe that over-segmentation occurs at k=8. Below that, it is over-simplified, i.e, some superpixels actually cover both halves of the image.

[1] Results may vary with different definitions of the threshold for early convergence. In this case, it was set to automatically converge if the updated center differ no more than 0.001 with respect to its previous value. Sigma (blurring effect) was set to 0.5.

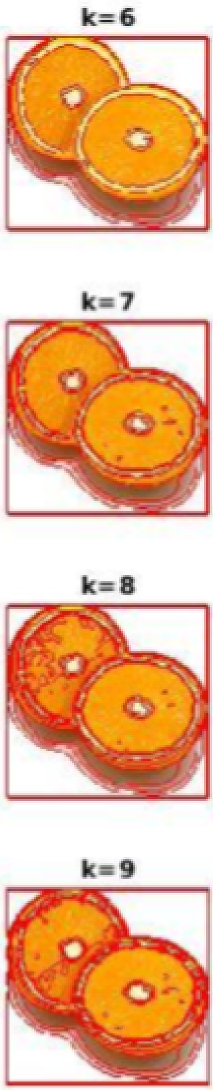
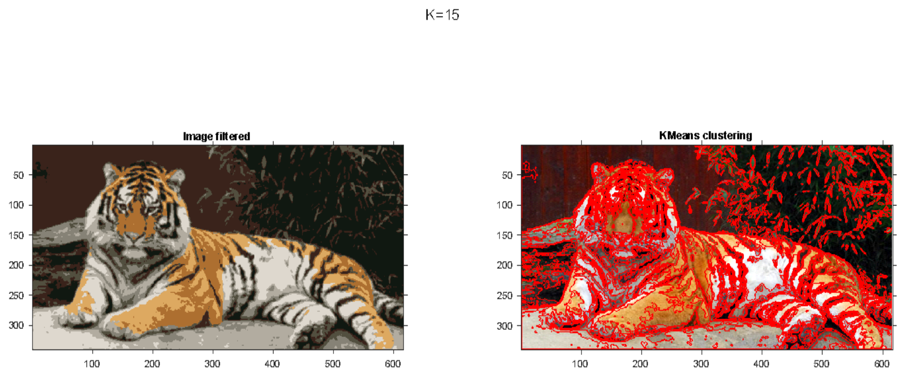


Figure 1. “k” parameter influence on segmentation.

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

Answers:

If we use the same parameters for the tiger image that we used for the orange one, we observe that we don’t have successful results. The image is way more complex, so we need more clusters and more iterations to make it converge.

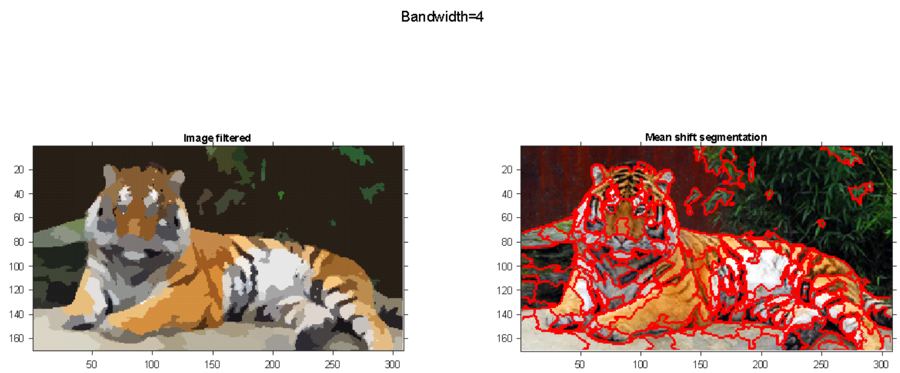


3. Mean-shift segmentation

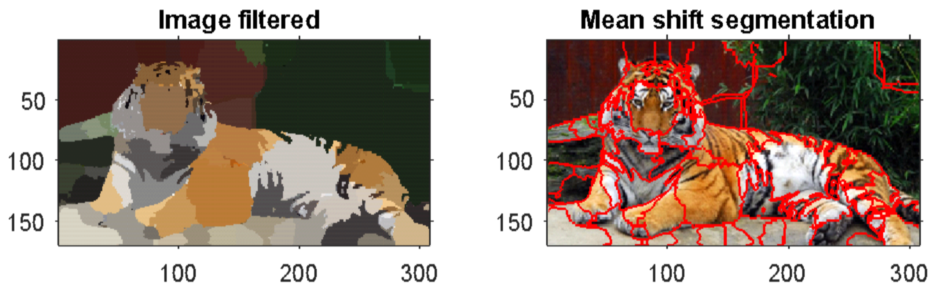
Question 5: How do the results change depending on the bandwidths? What settings did you prefer for the different images? Illustrate with an example image with the parameter that you think are suitable for that image.

Answers:

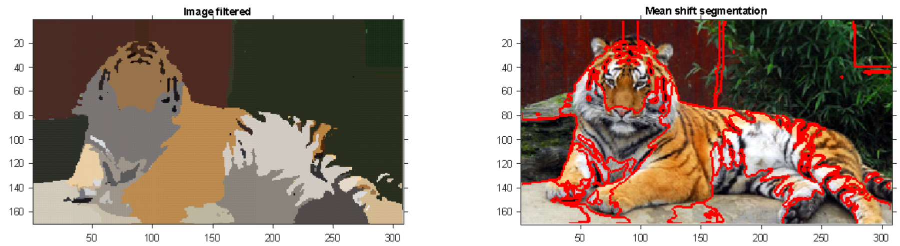
Spatial bandwidth refers to the range of the density function around the pixels. If we increase the bandwidth, the Gaussian also increases, and it is easier to mix with different pixels. So, we will get more pixels in the same mode, and then we would have fewer modes, also the area of the segmentations will increase, we will not have full detailed segments. Contrarily, if the Gaussian is narrow, we will not get different pixels mixed together.
On the other hand, when we increase the **color bandwidth**, the number of modes should decrease because we would mix up more colors together.



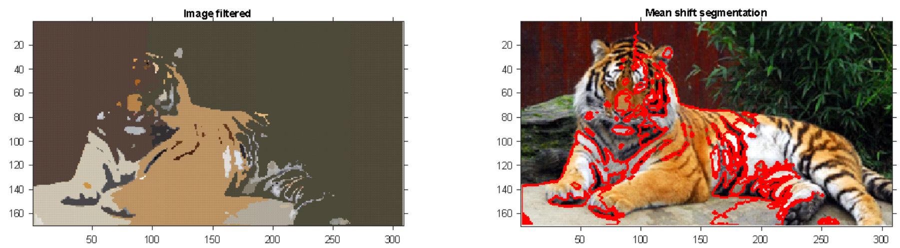
Bandwidth=10



Bandwidth=20



Bandwidth=50



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

Answers:

Differences:
- K-means determines superpixels based solely on the color, while mean-shifty also considers the position of those, i.e., color and spatial information.
- K-means needs a pre-specified number of clusters, i.e., “k”; whereas mean-shift will find a number of modes, but needs pre-specified bandwidth (Gaussian).
- K-means has a high sensitivity to outliers, whereas mean-shift is less affected by them.

Similarities:
- Both determine superpixels based upon their color.
- Both methods are iterative in their nature.
  - K-means will update its cluster center, according to the mean color.
  - Mean-shift will update its position, according to where the maximum of local density is located.
- Both methods are used for segmenting images, and treat color and pixels as samples from a probability distribution.

4. 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 with the parameters you prefer for that image.

Answers:

- **radius** is the neighboring system that this method uses (decreasing this will result in the algorithm considering fewer pixels, i.e., lower computational cost, and possibly more segmentations.).
- **ncuts_thresh** that controls the maximum allowed value for Ncut(A, B) for a cut to take place (increasing this will allow for more cuts, i.e., more segmentations. This might be needed for more complex images).
- **min_area** that controls the minimum size of a segment (decreasing this will result in lower areas permitted for segmentations, possibly giving rise to a higher number of them).
- **max_depth** that limits the depth of recursion (increasing this will result in more segmentations. This might be needed for more complex images)
- The weights corresponding to the similarities between pixels are computed with a function similar to K(ˆx) that was used for mean-shift segmentation (see previous section), where you can control the color bandwidth σ_c through the parameter **colour_bandwidth** (decreasing this will result in higher weights for similar pixels, i.e., stricter).


```
colour_bandwidth = 15.0
radius = 10
ncuts_thresh = 0.5
min_area = 10
max_depth = 10;
```

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

Answers:

Max depth, min_area, and ncut. Max_depth, for instance, when set to 2.5 resulted in less overlaying bounds that adjusted better to the dog -instead of many segments, such as we can see now on the right image of the Figure. However, the segmentation image was oversimplified, and many dog areas were not completely recognized.

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

Answers:

Any cut that separates out individual nodes will have a lower cut value than a cut that partitions the nodes into equal halves without normalization. To avoid this (normalized cut), normalization was introduced: Nout(A, B) = cut(A,B) / assoc(A,V) + cut(A,B) / assoc(B,V). Because the cut value will almost probably be a high percentage (if not all) of the total connection from that tiny set to all other nodes under this definition, the cut that partitions out small isolated points will no longer have a small Neut value. Because of the problem's intricacy, it doesn't arise very often in practice.

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

Answers:

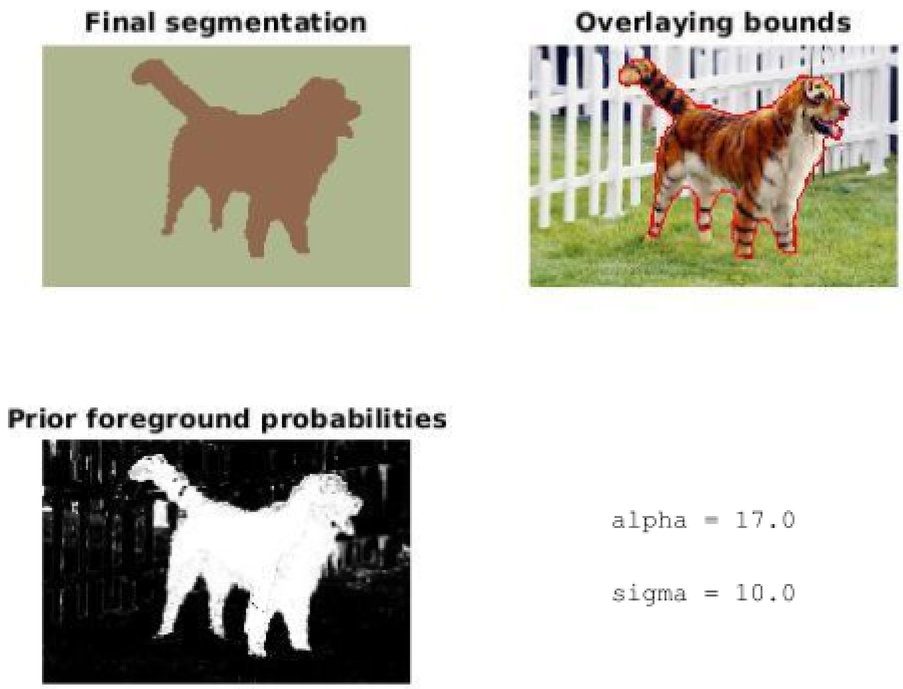
Increasing the radius will make the algorithm consider more neighboring pixels, and so, the computational time increases, while the number of segmentations is reduced.

5. Segmentation using graph cuts

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**: the maximum cost of an edge (increasing this will result in fewer edges, that is, fewer segmentations).
- **sigma**: how much the edge cost decays for decreasing similarity between neighboring pixels (decreasing this will result in more segmentations, since the cost of similar pixels is lower).



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

Answers:

In the case before, we chose k=20. Selecting a low value for k will result in fewer segmentations. A k value below 4 or 3 had a poor performance in our images.

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:

It depends on the problem. If the image shows an object that is wrapped around the background, then choosing an input will help. However, if it is distributed throughout the whole image with no clear boundaries, it will be of little help.
Additionally, the need for an input makes it highly dependent on the user, and cannot be automated straightforward.

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:

K-means	**Mean-shift**	**Normalized Cut**	**Graph Cuts**
use **color pixel info**rmation for segmentation			
based on **clustering**			
color only	color and space	vertices	Gaussian and prior
