IMAGE SEGMENTATION

A report on implementing the mean-shift algorithm.

Richard Polzin

22.05.2017
Maastricht University - Computer Vision - Stelios Asteriadis

INTRODUCTION

The following report will provide information about the accompanying implementation of the mean-shift algorithm. To keep everything concise no introduction to the mean-shift algorithm is provided (It can be found on the Internet and a basic knowledge of the functioning of the algorithm is assumed). Furthermore the documentation on the usage of the code was moved to the *README.md* file.
In the Experiments chapter results of the experiments conducted on three images from the Berkeley Segmentation Dataset [1] will be shown and discussed. The relevance of the different parameters as well as the changes when including positional information in the algorithm will be detailed.
The chapter Performance discusses the runtime of the algorithm and provides statistics on it's performance.
In the last chapter Conclusion & Future Work the results are concluded and some ideas for future work are presented.

EXPERIMENTS

This chapter details the results of the experiments conducted on the three images from the Berkeley Segmentation Dataset. Some images are shown here, the rest can be found in the data accompanying this report. This chapter will show a subset of those images to clarify the influence of different parameters. All experiments in this section used the optimized meanshift algorithm. It runs significantly faster, as shown in the chapter Performance.

Parameter r defines the distance that is used to threshold neighbouring pixels in the algorithm. Therefore it is directly related to the peaks found. If r is close to 0 the algorithm will not find many neighbors and will quickly converge. This will result in a large amount of peaks and little segmentation. If r is large and many neighbors are found more points will converge to the same peak resulting in larger segments. Two assumptions have to be made to back this theory up. The points are spread out in the search space and the c parameter does influence the result. We assume that the conversion from RGB to LAB spreads the points. The effect of c will be discussed later.

[1] The dataset is available at https://www2.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/BSDS300/html/dataset/images/color/ . Images 181091, 55075 and 368078 have been used and are included in the project.

Image 1: From left to right: Original, r = 5, 10, 15, 20 with c = 1 and *3D feature vector*.

Segments found:	362	41	8	8
Runtime:	10.0s	11.9s	13.9s	9.6s

Image 2: From left to right: Original, r = 5, 10, 15, 20 with c = 1 and *5D feature vector*.

Segments found:	367	42	8	8
Runtime:	10.6s	12.4s	10.8s	10.9s

Image 1 shows that the intuition was correct. As *r* increases the amount of peaks is reduced and the segments contain larger parts of the image. In the last images of the series with r=20 gray-green fragments are visible in the sky where originally clouds have been. The images with r=5 and r=10 fill these spots with the same color as the brighter part of the sky. With r=15 the sky is in a single color.

As the algorithm works in the three-dimensional LAB space without spatial information, the mean-shift algorithm does not consider of the position of the pixels. To my understanding the inclusion of spatial features should remove the green clouds, but as can be seen in Image 2 this is not the case. Also the amount of found clusters are very close in Image 1 and 2. This could be related to an issue with the implementation or a lack of focus on the positional data. Within the scope of this project I did not find an extensive explanation of this behavior.

In comparison to the non-optimized version of meanshift the speed up through the optimization linked to the parameter c is significant. On a small image [2] the speedup was almost of factor 10. How the value of c influences the results and the runtime can be seen in Image 3:

Image 3: From left to right: Original, c = 1, 5, 10 with r = 10 and *3D feature vector*.

Segments found:	22	30	32
Runtime:	0:52 min	4:41 min	11:23 min

In the optimized mean-shift algorithm the points on the path to the peak that are closer than r/c to the current center of gravity are added to the same segment as the starting point. The effect of c is maximized at almost 0 and c is irrelevant [3] at infinity. The image shows that large values of c lead to larger clusters and longer runtimes, which fits the intuition as r/c is minimized and the benefit of the speedup is small while the overhead is large.

[2] Exact parameters: Image a, scale 0.5, r=5, c=4 Runtime with speedup: 37.7s without: 372.5s
[3] Technically not irrelevant. While it does not add points, the neighbours are still calculated. This increases the runtime even when the optimization does not add any benefit.

Image 4 shows how c is influenced by adding positional information:

Image 4: From left to right: Original, c = 1, 5, 10 with r = 10 and *5D feature vector*.

Clusters found:	22	30	32
Runtime:	0:46 min	5:42 min	22:29 min

Without changes in the amount of clusters, nor in the optical quality of the clusters the runtime at c=10 is almost twice the runtime when not using positional information. Due to the additional dimensionality the complexity for this speedup is increased and the penalty when choosing high values is significant. Regarding low values of c [4] the runtime does not change much.

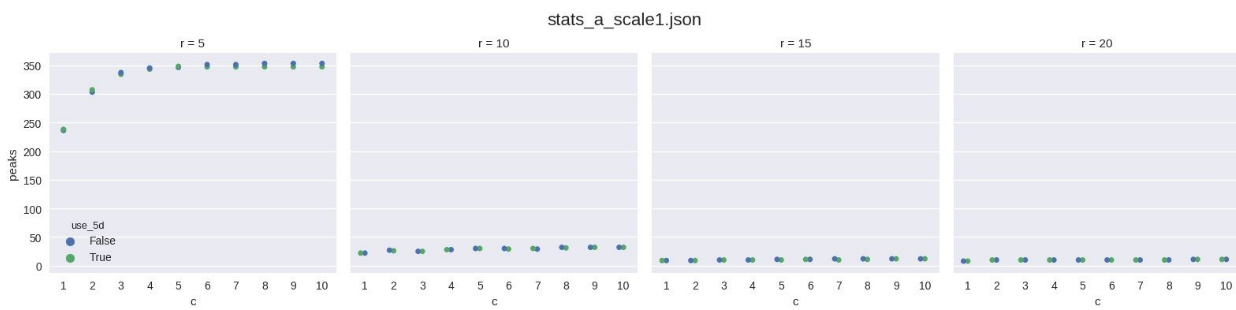
In general the effect of c does not affect the clusters a lot. But using c values < 1 can lead to large and possibly wrong clusters, as seen in the image on the right (11 Clusters found in 42 seconds, c=0.4, r=10, 3D information). When using small r values and large c values the runtime quickly increases up to 30 minutes even though the images used only had about 150,000 pixels.

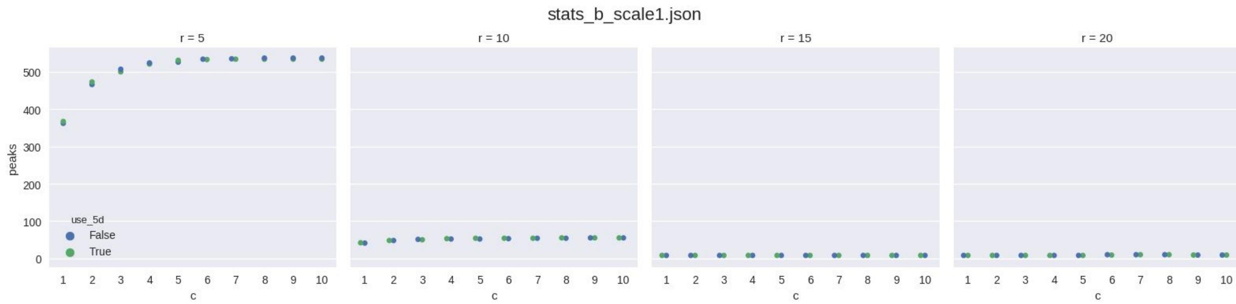
[4] Good results were achieved within the range of roughly 2 to 6.

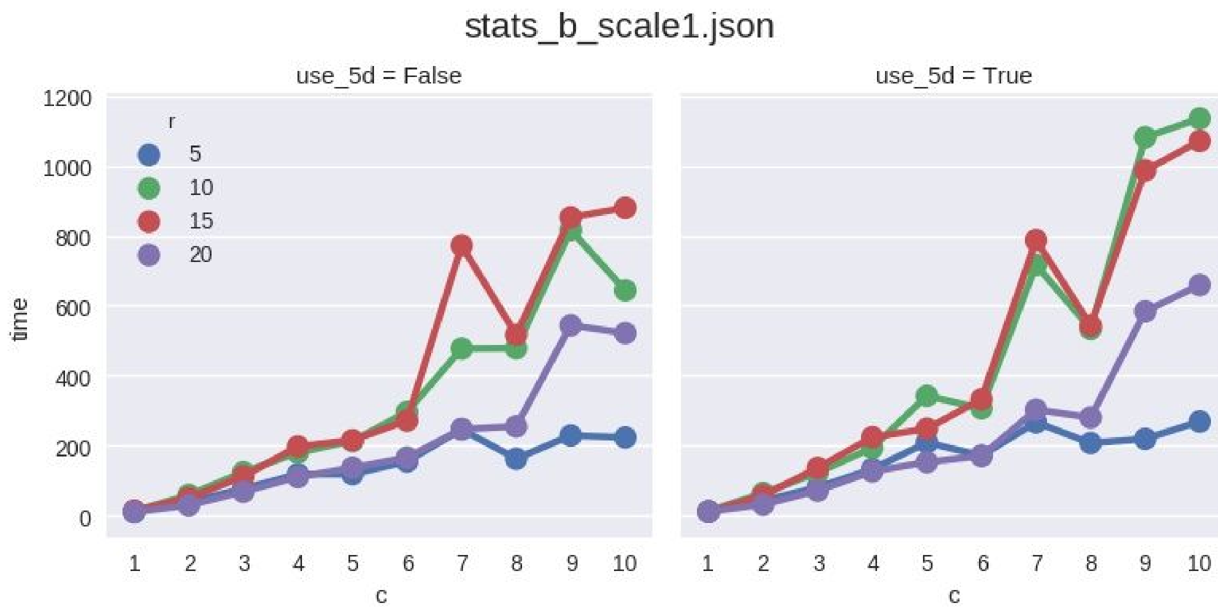
PERFORMANCE

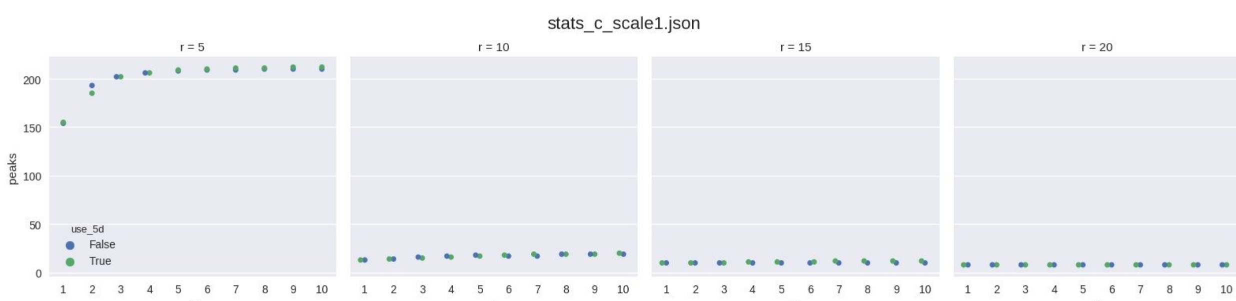
As mentioned in the previous chapter the speedups offered significant improvements on the runtime. Therefore only a small set of tests was executed using the non-optimized algorithms and due to lack of statistical significance the results are not included in the following chapter.

The following images show six plots per used image. In the first four the peaks found for different r and c values are shown. The time required to converge for different r and c values is shown in the next two. In general the runtime increased as *r* decreased and *c* increased. The increase was amplified when positional information was included. On the other hand the amount of found peaks rarely differs when 5D features are used.




stats_b_scale1.json
r = 5
r = 10
r = 15
r = 20
peaks
use_5d
False
True
c


stats_b_scale1.json
use_5d = False
use_5d = True
time
r
5
10
15
20
c


stats_c_scale1.json
r = 5
r = 10
r = 15
r = 20
peaks
use_5d
False
True
c

For higher resolution variants of the images see the accompanying files.

Conclusion & Future Work

This work on the mean-shift algorithm offers insight into several different aspects of the algorithm. For once the significance of the optimizations is shown. Furthermore the effect of the parameters r and c, as well as the effect of the positional data is discussed. Finally the performance of the algorithm is analyzed and used to back up the observations made in the previous chapters.

Parameter r strongly affects the generation of clusters. Larger r values result in larger clusters and lower computation times. If the parameter c is 'small' it does not affect the results but speeds up the performance significantly. The usage of positional information provided a slight boost on the clustering while not slowing the program down for sensible values.

A general shortcoming of this work is that the inclusion of positional information into the algorithm did not yield the expected results. Furthermore some of the outcomes [5] could not be explained. Future Work should extend on those issues and investigate the reasons to offer a deeper understanding of the algorithm.

[5] Especially the 'green clouds' in Image 2