Computer Vision

Assignment 1: Segmentation

Kathrin Hartmann

1 Mean-shift Algorithm

The mean-shift algorithm is a popular algorithm for image segmentation. In this assignment, first a basic version of the algorithm was implemented. After that, two speedups were added in order to improve the runtime of the algorithm. Experiments are performed on the given test data and on three images from the Berkley Segmentation Dataset.

1.1 Implementation

The implementation follows the scheme given by the task description. The function mean_shift calls for each data point the function find_peaks that takes a data point as input and returns its corresponding peak. mean_shift then checks if the found peak already exists or if it is close to an already found peak and gives a label to currently examined the data point accordingly. The result of the basic mean_shift algorithm on the provided test data points is shown in figure 1.

The outcome on the provided test data points is the same when using the two speedup methods, find_peak_opt and mean_shift_opt. Overall, the code is written as efficient as possible. Within the methods find_peak_opt and mean_shift_opt, no for-loops are used. It is taken the advantage of numpy array functions.

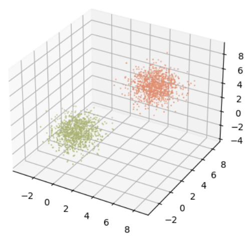


Figure 1: Test of mean-shift algorithm: The two obtained clusters with r = 2 and t = 0.01

1.2 Runtime improvement with opt-methods

When comparing the performance of the simple meanshift and the meanshift algorithm with speedups on the test data points, it is noticeable that the algorithm with speedups outperforms the simple one: The simple meanshift algorithm runs for 1.1052 seconds whereas the meanshift algorithm with speedups only for 0.2945 seconds. The algorithm with the speedups is almost 4 times faster than the simple meanshift algorithm, which is very beneficial when using the algorithm for image segmentation where a large space has to be searched.

2 Image Preprocessing

Before segmenting images with the mean-shift algorithm, several preprocessing steps were applied. As a first step the image, which was read with cv2, is converted from the BRG colour space to the RGB colour space. After that the image is resized by dividing its height and its width by a certain parameter. In order to make segmentation easier, a 5x5 filter is applied to blur the image. As a next step the image is converted to the CIELAB colour space. Now, the shape of the image array is adjusted: For 3D feature vectors with only CIELAB colours as features, the dimensions of the input array are transformed from the original image shape which is X x Y x 3 to the final shape 3 x (X x Y). When using 5D vectors that have the CIELAB colours and the position of the pixels as features, the positional information is appended to the 3D final shape, which results in the shape 5 x (X x Y).

3 Image Segmentation and Experiments

With the aforementioned shape adjustments for 3D of 5D feature space, the implemented mean-shift algorithm can segment images. In the following part, experiments are performed on three images from the Berkley Segmentation Dataset.

3.1 Experiment: parameters r and c

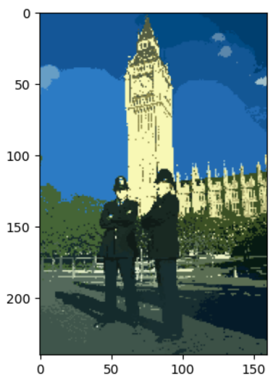
The mean-shift algorithm is mainly directed by two different parameters: r is the size of the basin of attraction around a peak and c defines to what extend pixels that come across the moving mean on the way to the peak should be considered of the same cluster as the peak. It is of interest to examine the effect of the size of these two parameters when performing image segmentation. It has to be mentioned that all images have been resized by half size and are blurred during preprocessing. The parameter t in find_peak_opt is set in all experiments to 0.01.

Firstly, the effect of r is investigated. On all 3 images, configurations of the algorithm with r = 5, 10, 20 and 30 are tested. c stays constant on the value 10. It is tested for 3D feature space as well as for 5D feature space. The results can be found in the Appendix in the figures 3, 5 and 7.
With an increasing value of r, the computational time increases. Also, for both the 3D and 5D feature space the number of segments in the image decreases. The best parameter of r depends from image to image. Whereas r = 20 leads to a good balance, meaning objects are still recognizable but details disappear for the first image, r = 30 is the better choice for the second and third image as r = 20 still leaves too many details for them.

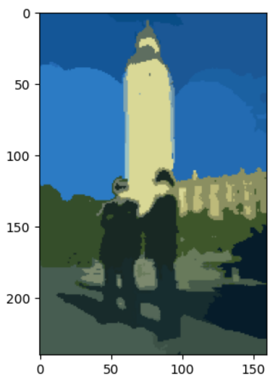
Secondly, different values for c are tested: c = 5, 10, 20, 30. r is set to 10 for these experiments. The results for 3D and 5D feature space can be found in the Appendix in the figures 4, 6 and 8.
When c increases, the computational time decreases. Apart from that, the two feature spaces behave differently with the increasing value of c: In the 5D feature space, increasing values of c have almost no effect on the number of segments in the image and how the image looks like, whereas in the 3D feature space the number of segments decreases and many segments are connected wrongly. This shows very well how the 5D feature space benefits from the positional encoding and does not recognize pixels with the same colour values from different areas in the image as one cluster.

3.2 Experiment: performance with and without blur

In this section a comparison is made between images that were preprocessed with and without blurring. Figure 2a shows the segmented image result for the third test image without blurring in preprocessing. There is some separation of the trees in the background and the sky noticeable, however the building in the background is still separated in multiple parts and not well separated from the trees. In figure 2b where blurring was used during preprocessing, the sky, the trees, the tower and the people are well separated from each other. It can therefore be concluded that a blur step during preprocessing is beneficial for image segmentation as blurring reduces the noise in the image and the level of detail. This makes it easier for the algorithm to find coherent segments in the image.



(a) Segmented image without blurring in preprocessing



(b) Segmented image with blurring in preprocessing

Figure 2: Experiments with blurring in preprocessing for parameter r = 30, c = 10, t = 0.01 and 5D feature space

4 Appendix

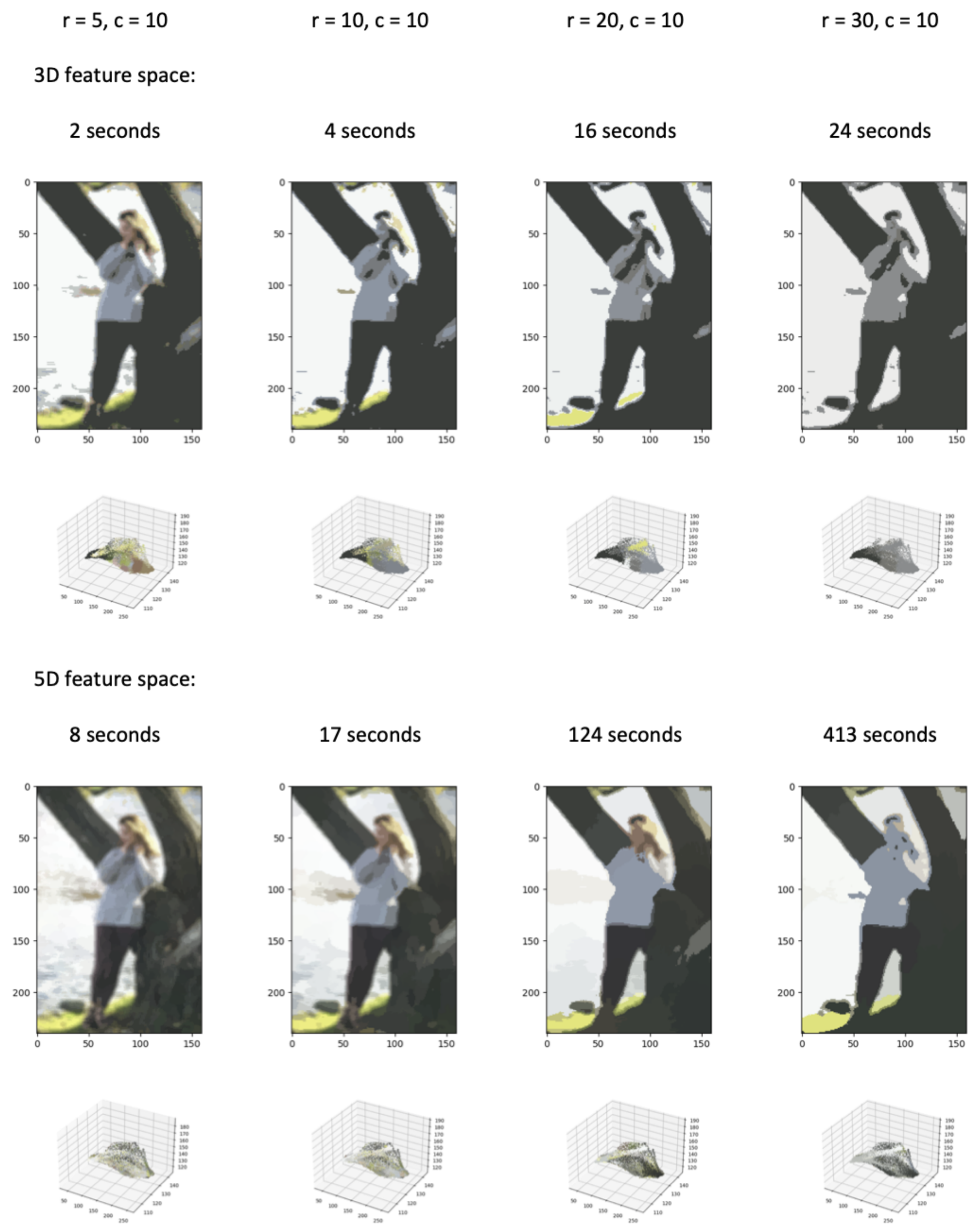


Figure 3: Experiments with different numbers of r for the first image. The time in seconds indicates the runtime of the experiment

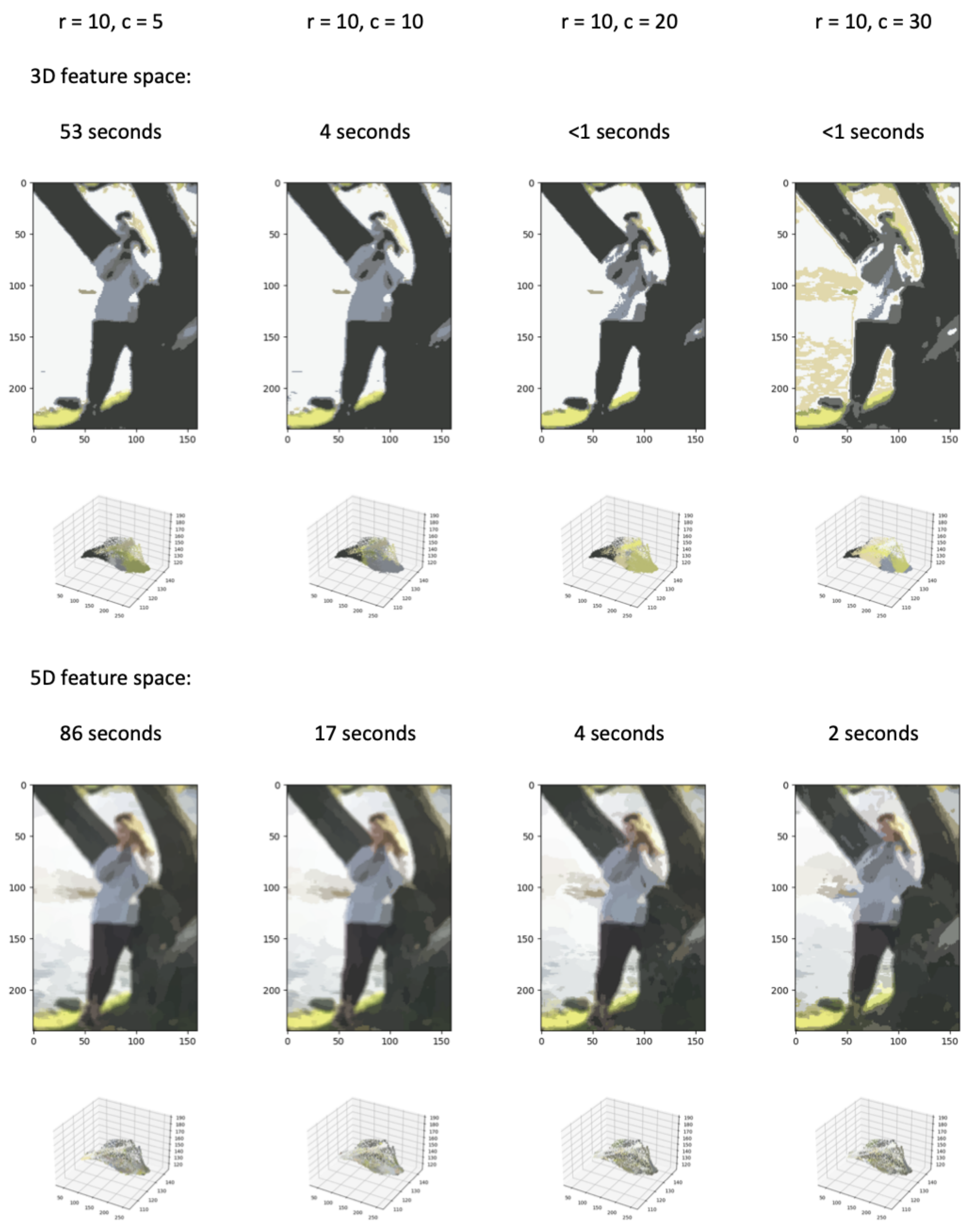


Figure 4: Experiments with different numbers of c for the first image. The time in seconds indicates the runtime of the experiment

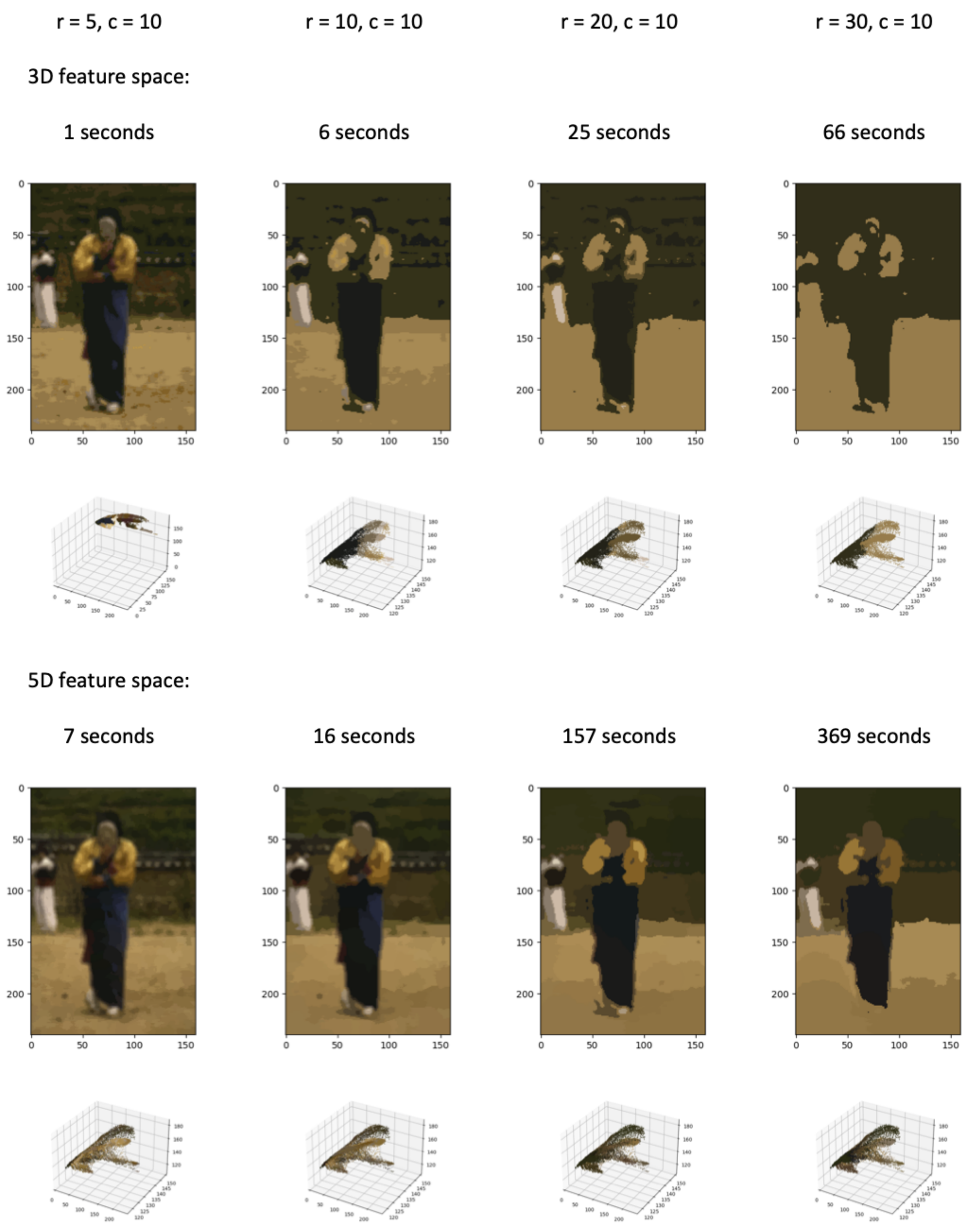


Figure 5: Experiments with different numbers of r for the second image. The time in seconds indicates the runtime of the experiment

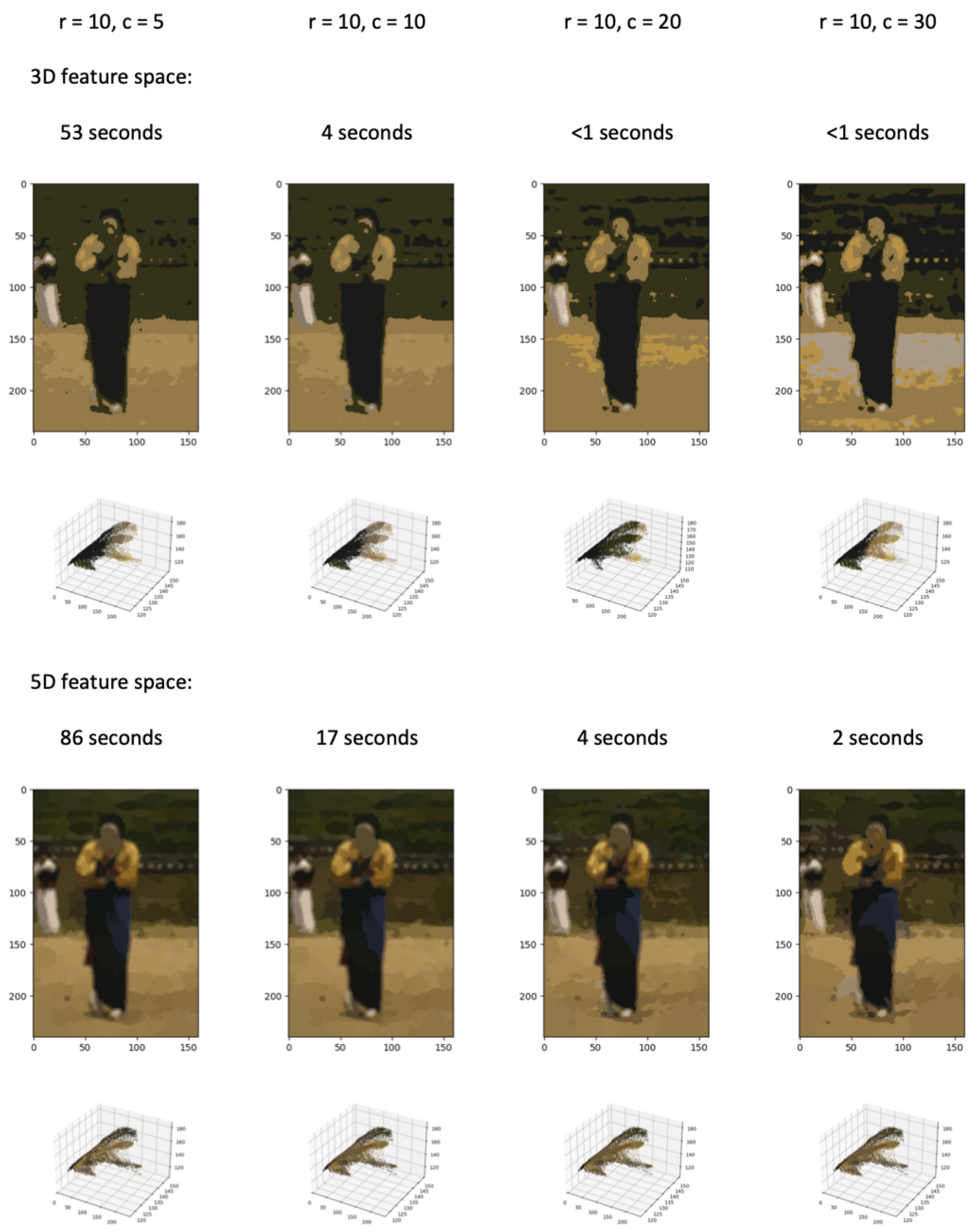


Figure 6: Experiments with different numbers of c for the second image. The time in seconds indicates the runtime of the experiment

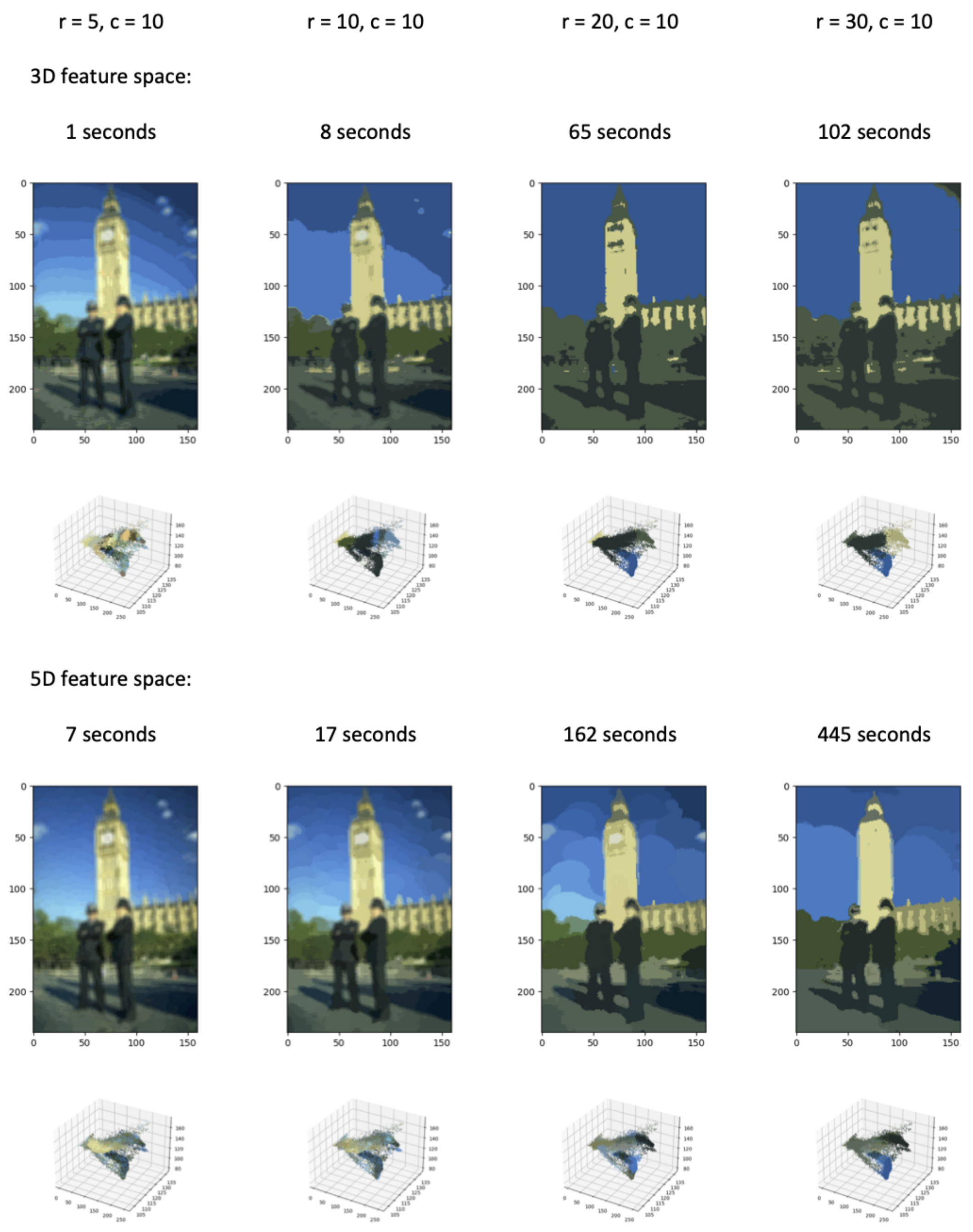


Figure 7: Experiments with different numbers of r for the third image. The time in seconds indicates the runtime of the experiment

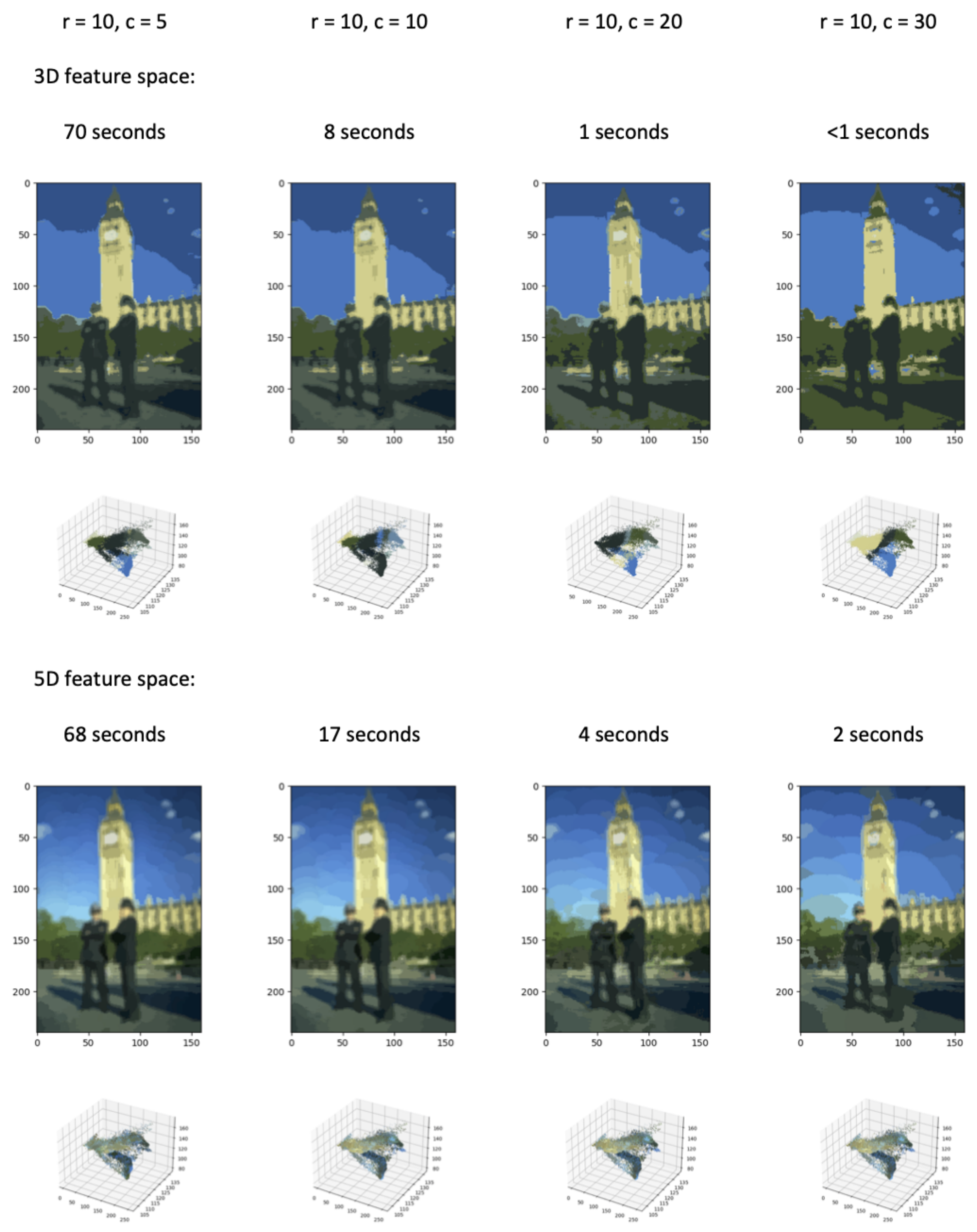


Figure 8: Experiments with different numbers of c for the third image. The time in seconds indicates the runtime of the experiment