Scene Segmentation and Interpretation

Image Segmentation - Region Growing

Arash Akbarinia
akbarinia.arash@gmail.com

Igor Jovancevic
igorjovan@gmail.com

University of Girona, Image Processing. Girona, Catalonia

1 Objective

The objective of this report is to present the results of two different algorithms – Region growing and K-means – on segmenting four images. We first implemented Region growing algorithm and compared its results with the *kmeans* function of Matlab. Results of each algorithm are discussed in one section and in the last section we conclude and compare the results.

2 Region Growing

Region growing is a simple region-based image segmentation method. It is also classified as a pixel-based image segmentation method since it involves the selection of initial seed points. [Wikipedia]

In this section we applied the Region growing algorithm on the four provided images, 'coins', 'colour', 'crane' and 'woman'. Result of each image is studied separately in the following four subsections. At the end we discuss some of our observations.

For each image we tried different tolerances and two adjacency strategies, four and eight neighbourhoods. At the end we applied median filter on the segmented results to remove the noisy regions. Median filter turned out to be very useful and it improved the results significantly.

Note

One of the greatest difficulties of analysing the results of Region growing algorithm was meaningful representation of different regions. Because, when number of regions is high, two or more different regions are assigned to very similar RGB values, which are visually very hard to distinguish for humans. We used *label2rgb* function of Matlab for this purpose.

1 Coins

In this picture the background is clearly darker and the coins range between white and gray. The optimum result will be to have eleven clear regions, one for the background and ten coins. However as some parts of some coins have different brightness than the mean of region, number of regions are more than eleven. But we could obtain very good results by choosing correct tolerances and applying number of median filters. The results of segmented image with tolerance 65 can be observed in figures 1 and 2. Results of segmented image with tolerance 70 are illustrated in figures 3 and 4.

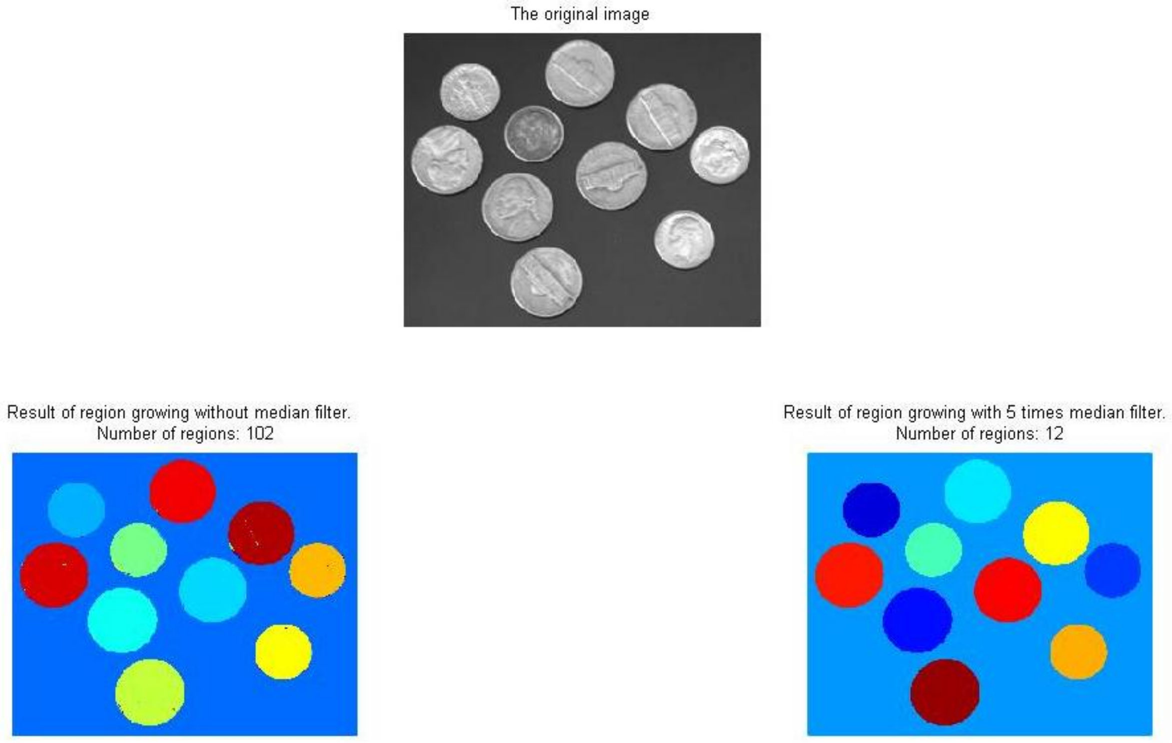

Figure 1: Result of Region growing on 'coins' image with tolerance **65** and using **4** neighbourhood.

As it can be observed in figure 1, pure regional growing with tolerance 65 generates fairly good results, with human eye we can detect the big eleven regions. However the real number of regions are 102, this is due to the fact that many small dots are identified as one region. We show that by applying median filter we reach the almost optimum result, twelve regions. The greatest difficulty is with the small coin on top of the image which has a very dark colour inside the coin and therefore its gray level is lower.

Applying the same tolerance on the same image with eight neighbourhood adjacency generates the result which is displayed in figure 2. In this case also by applying median filter we reach a perfect result. Same problem exists as mentioned above with the darker coin.

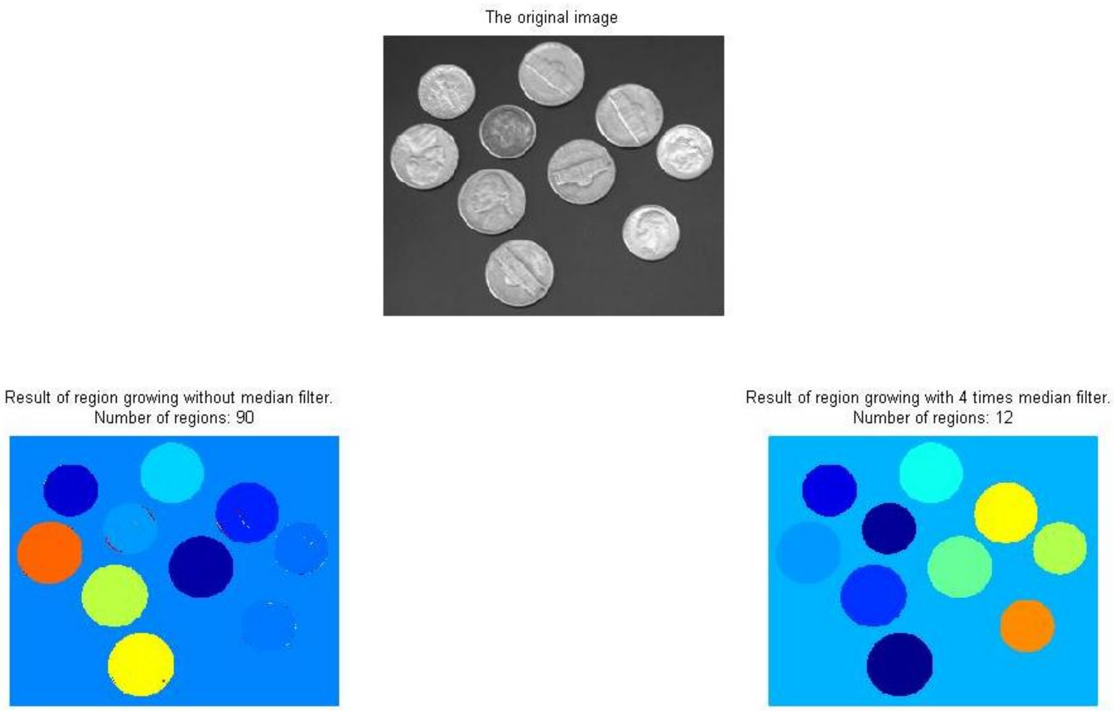

Figure 2: Result of Region growing on 'coins' image with tolerance **65** and using **8** neighbourhood.

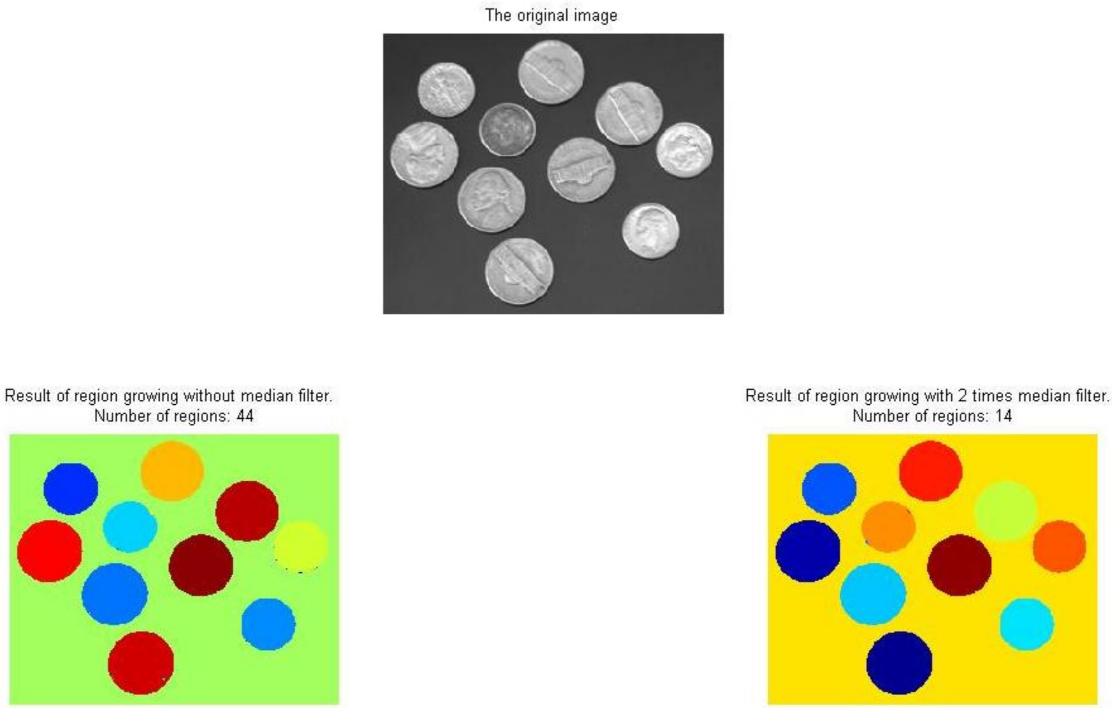

Figure 3: Result of Region growing on ‘coins’ image with tolerance **70** and using **4** neighbourhood.

Result of Region growing algorithm with tolerance 70 can be observed in figure 3. It is close to the results with tolerance with 65. However for the eight neighbourhood adjacency, as it can be observed in figure 4, the problematic coin is under-segmented and therefore after applying median filter we might lose its borders. Hence, a simple conclusion can be its better to use lower tolerance and over-segment the image and then polish the results by applying post-processing techniques. Because in case under-segmentation, we permanently lose information about some objects.

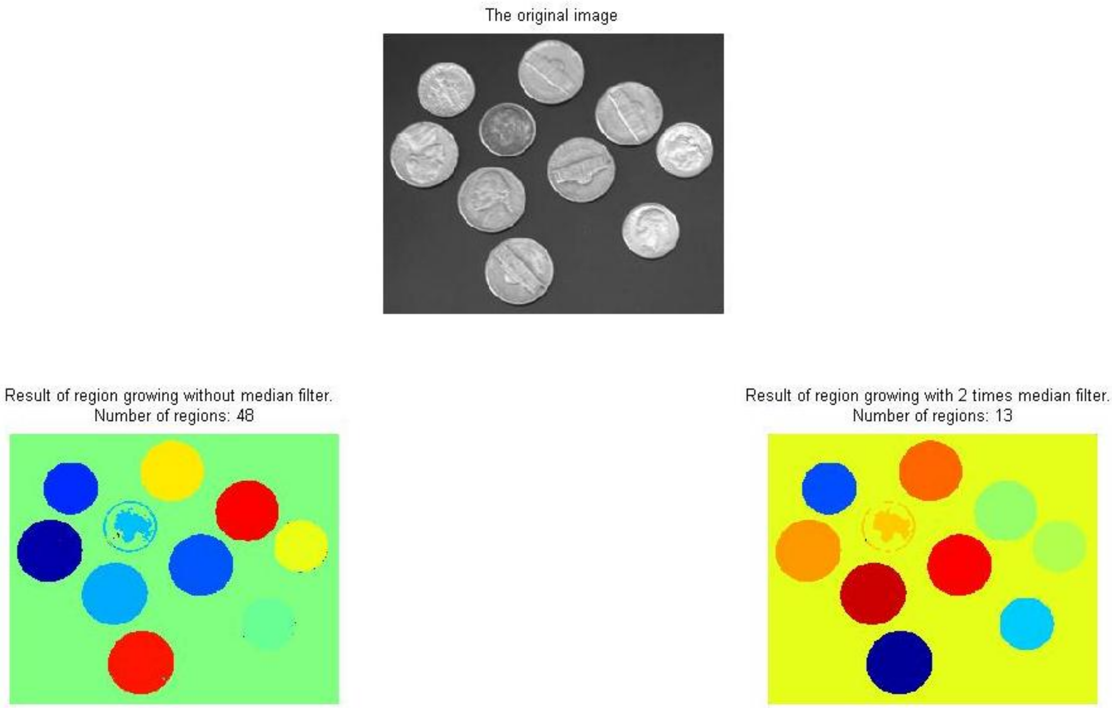

Figure 4: Result of Region growing on ‘coins’ image with tolerance **70** and using **8** neighbourhood.

The time consumed for all the above created results with an average speed laptop was satisfactory, around three to four seconds.

2 Colours

This image contains four different distinguishable colour segments. The differences between colours are very big, therefore with big different tolerance levels we obtained the same perfect results. The computation time for different tolerance levels was almost the same. However same as ‘coins’ image, eight neighbourhood adjacency consumes more computation time. One sample result is illustrated in figure 5. As it can be seen median filter was not necessary to be applied for this image and in fact with pure Region growing algorithm we can obtain the perfect solution.

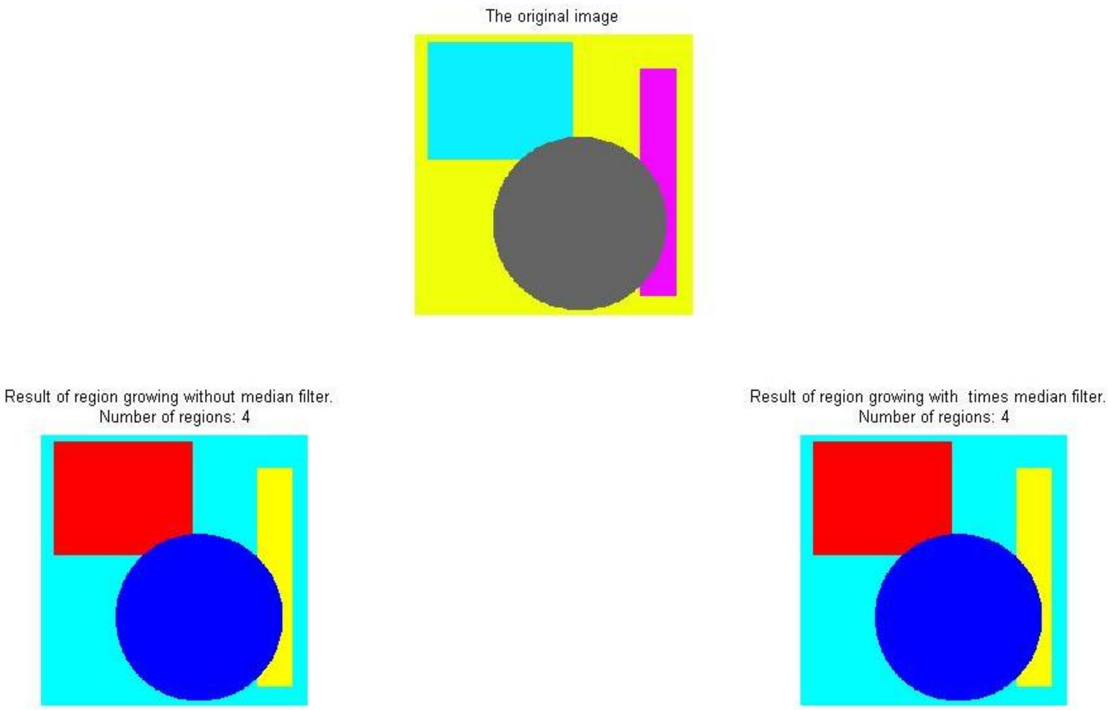

Figure 5: Result of Region growing on ‘colour’ image with tolerance **10** and using **4** neighbourhood.

Since the result of different tolerance levels was identical on this image we did not include those images in this report. All the results considered, it can be concluded that since this image is synthetic, segmentation is easy and perfect results can be obtained. However, same results are not easy to obtain with natural images. Firstly because in natural images many more regions exist. And secondly because the change of colours are not always this big and deciding for pixels on the edges might be difficult.

3 Crane

Segmenting this image with Region growing algorithm is very difficult, because there are many different small regions in this image. It is hard to exactly say how many regions is the optimum outcome, but as it can be observed in figures 6, 7, 8 and 9, good results can be observed, however applying post-processing algorithms is unavoidable and essential.

With tolerance 95 and four neighbourhood adjacency, we can get a good segmented image, as it can be observed in figure 6, but too many regions are identified, 813. Even small screws are recognised as regions. By applying a few times median filter on the image, number of regions are dramatically reduced to 130. Therefore we conclude choosing a correct post-processing algorithm on this image is very important.

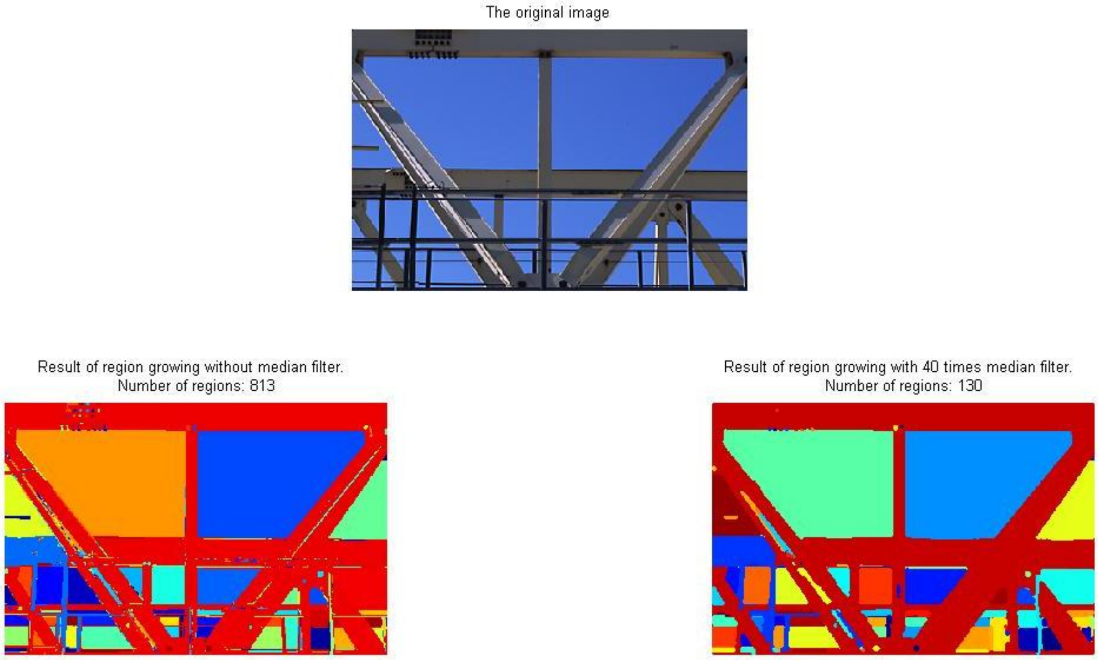

Figure 6: Result of Region growing on ‘crane’ image with tolerance **95** and using **4** neighbourhood.

Applying the same tolerance level – 95 – with eight neighbourhood adjacency generates fewer number of regions, 590 and 106 with and without median filter respectively. It is hard to judge the results, but one hypothesis worth more investigation can be that for images that have many number of isolated same regions – in this picture different parts of sky and iron bar – using eight neighbourhood adjacency might be better. Nevertheless it must be pointed out that probably Region growing algorithm is not the best solution for this types of images.

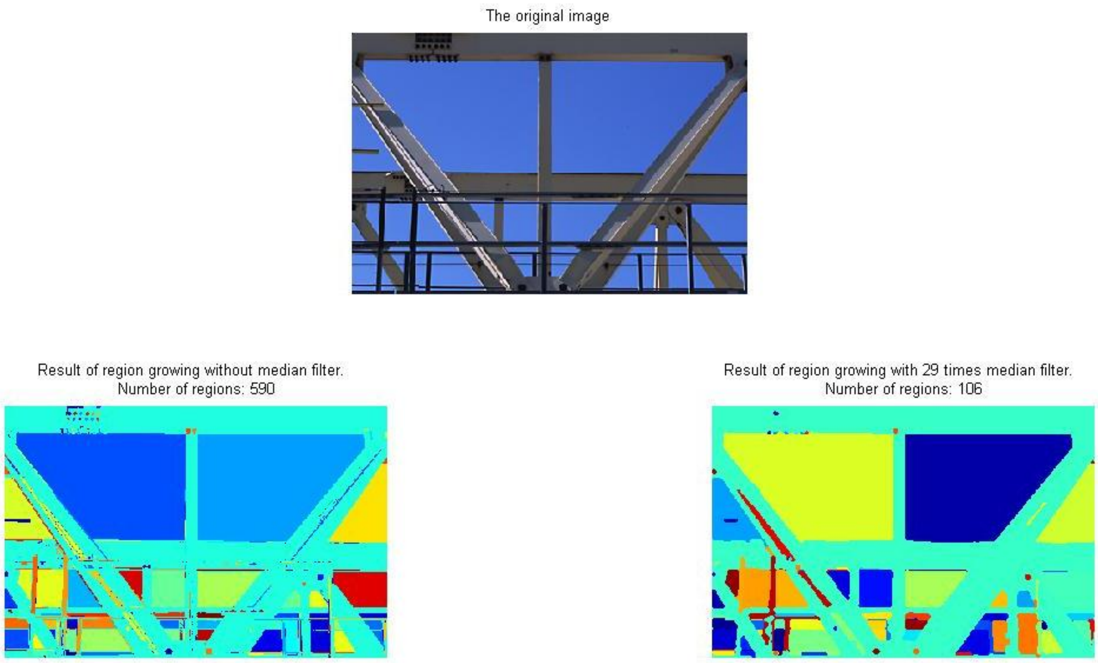

Figure 7: Result of Region growing on ‘crane’ image with tolerance **95** and using **8** neighbourhood.

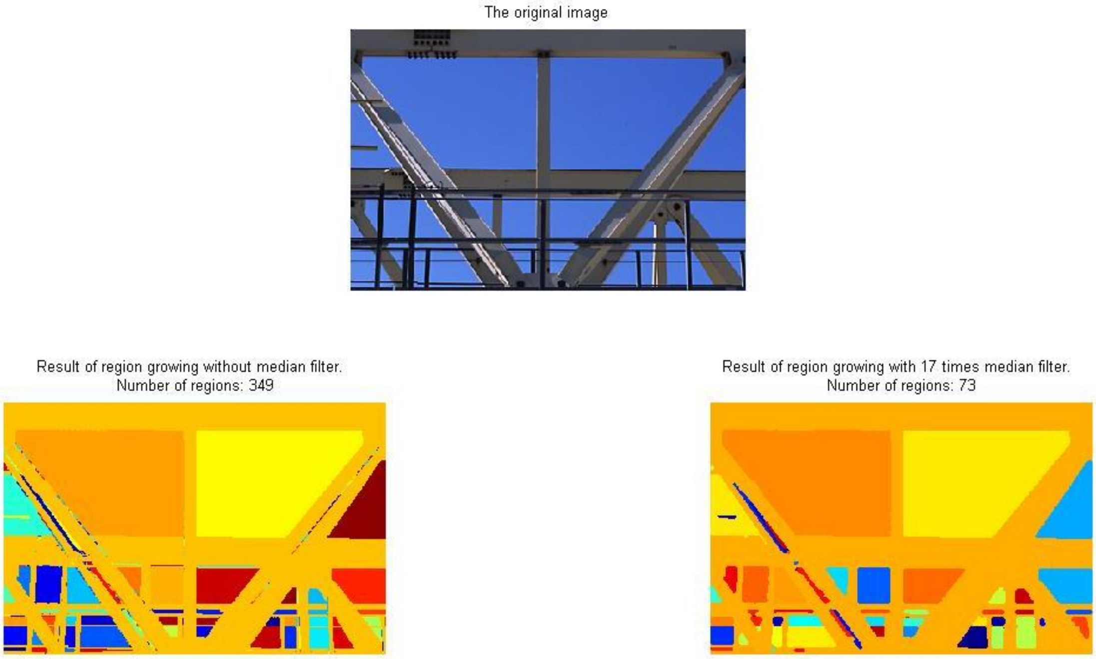

Figure 8: Result of Region growing on 'crane' image with tolerance **125** and using **4** neighbourhood.

With tolerance 125 number of regions are reduced but still neither of four and eight neighbourhood adjacencies generates the optimum result, as it is displayed in figures 8 and 9.

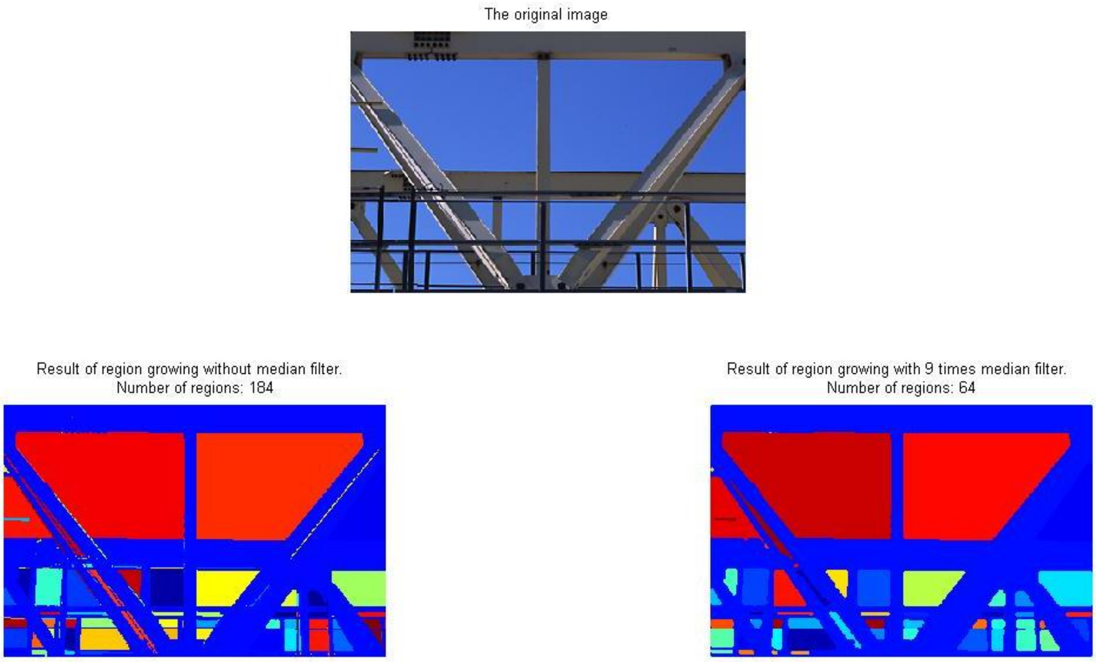

Figure 9: Result of Region growing on 'crane' image with tolerance **125** and using **8** neighbourhood.

One of the disadvantages of Region growing algorithm is working with shades. In fact this method may not distinguish the shading of the real images. This affect can be observed in segmentation of 'crane' image. Some parts of the iron bar that are darker due to the shade are identified as a separate region.

The computational time was greater for this image comparing with all other images, due to the fact that there are many regions in this image. However still the speed is acceptable, on an average laptop with Matlab code it took between six to seven seconds to segment the image.

4 Woman

Segmenting ‘woman’ image can be tricky, mainly because it is hard to decide which parts we want to take as a region, for instance one might want to have small details like eye as a region and in one application you might only want to segment the image based on background, body, and different cloths. Different results of segmentation on this image are illustrated in figures 10, 11, 12 and 13.

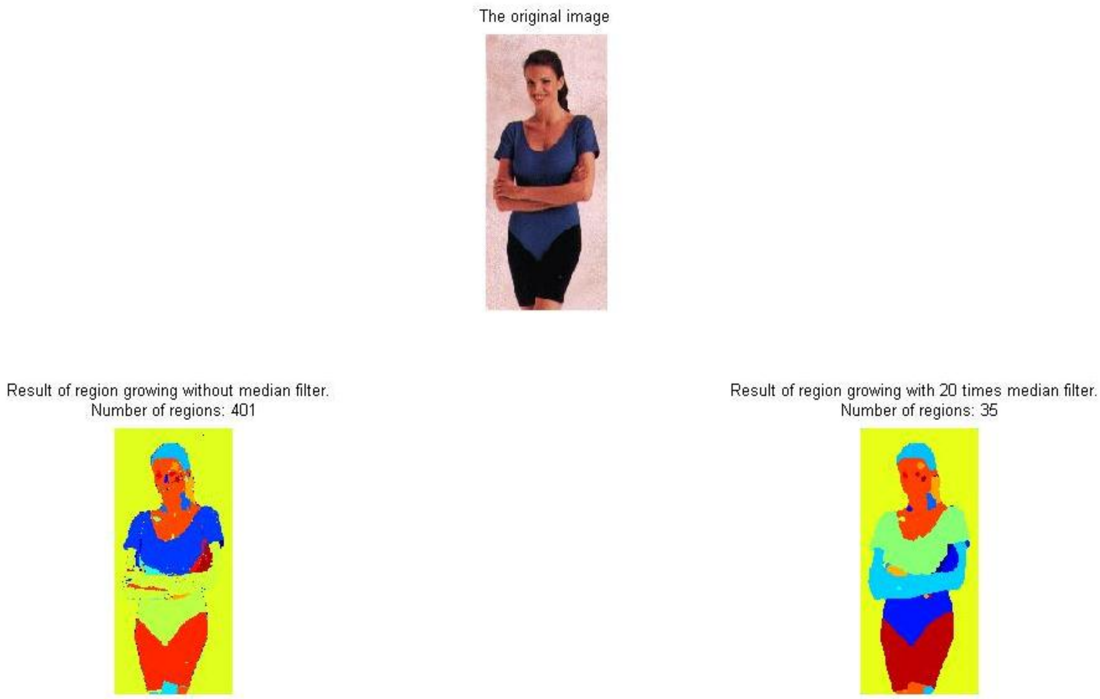

Figure 10: Result of Region growing on ‘woman’ image with tolerance **65** and using **4** neighbourhood.

With lower tolerance levels, as it is illustrated in figures 10 and 11, some small features e.g. eye, fingers, etc. can be extracted. However too many regions will be identified with lower tolerance levels, 401 and 329 with four and eight neighbourhood adjacency respectively. Those numbers can be naturally reduced by post-processing algorithms, as we did it with median filter to 35 and 33 regions in four and eight neighbourhood respectively.

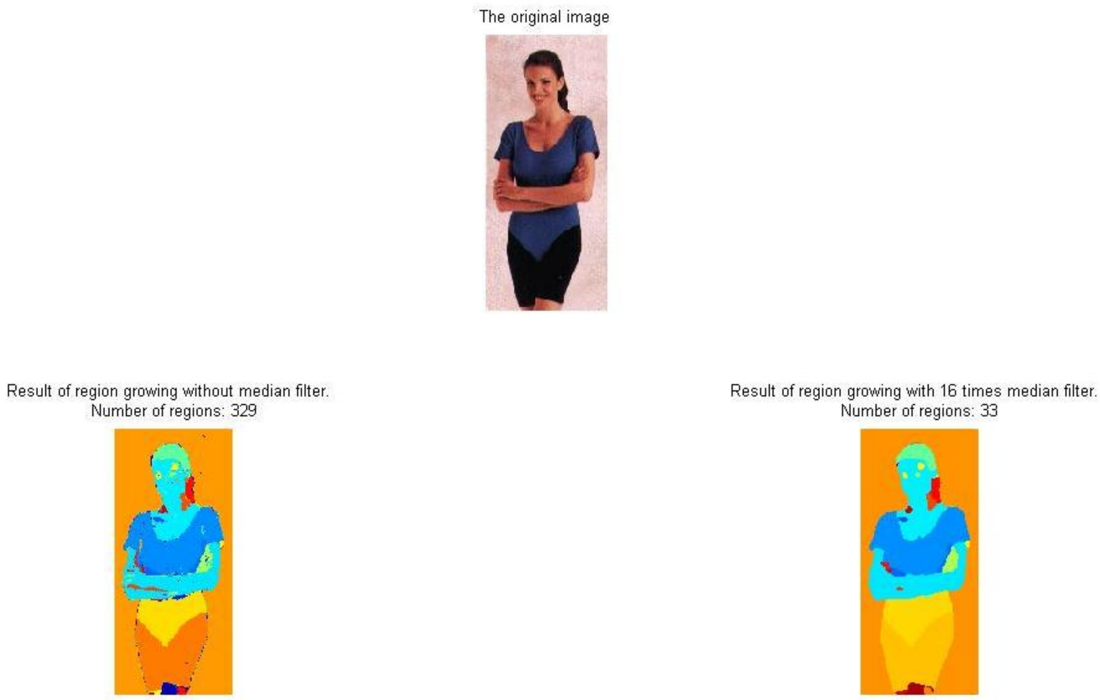

Figure 11: Result of Region growing on ‘woman’ image with tolerance **65** and using **8** neighbourhood.

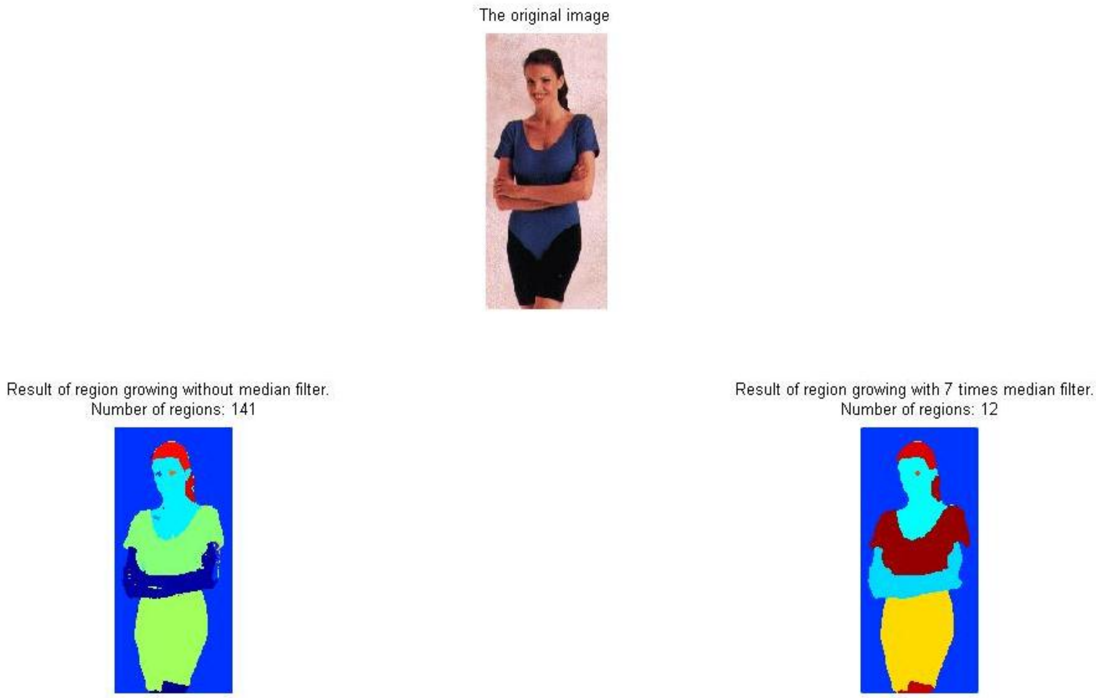

Figure 12: Result of Region growing on ‘woman’ image with tolerance **90** and using **4** neighbourhood.

With higher tolerance level, 90 for example, we can obtain a more clear image, reaching twelve regions after applying median filter. However this way all the face parts, or the leg parts will be identified as one unified region.

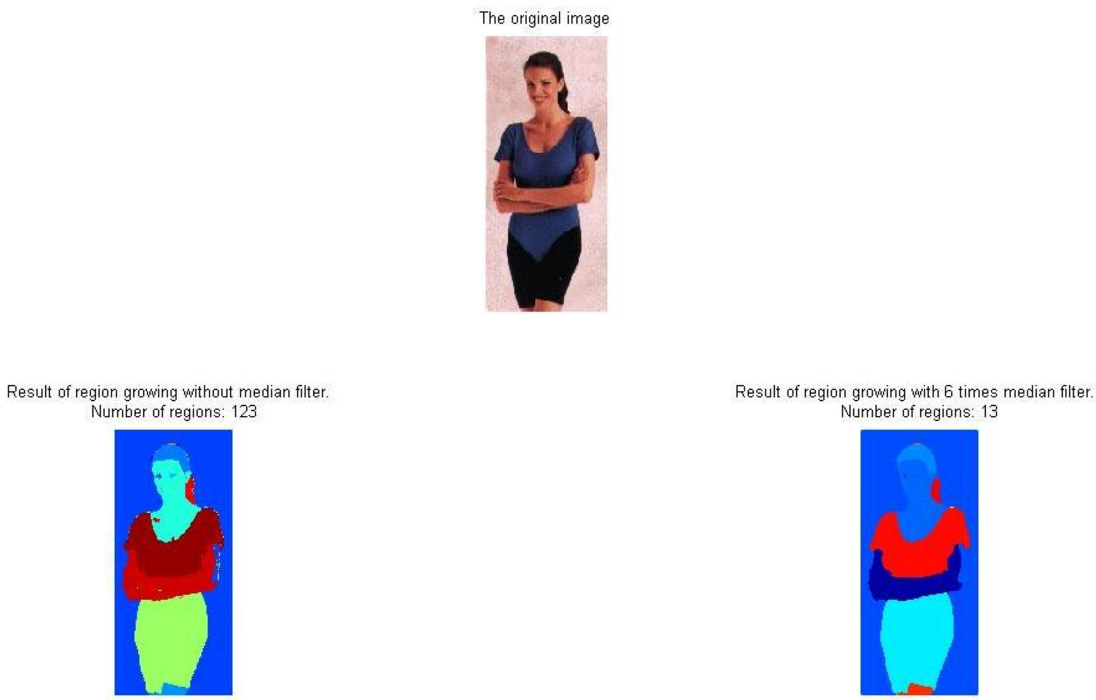

Figure 13: Result of Region growing on ‘woman’ image with tolerance **90** and using **8** neighbourhood.

5 Observations

In the table 1 we have summarised the results of Region growing algorithm on different images, providing different tolerance levels and neighbourhood systems.

Image	4N			8N		
	Threshold	**Time**	**Regions**	**Threshold**	**Time**	**Regions**
Coins	65	3.312538	102 (12)	65	4.029445	90 (12)
	70	3.322577	44 (14)	70	4.034269	48 (13)
Colour	10	4.678672	4 (4)	10	5.343206	4 (4)
	90	4.562824	4 (4)	90	5.350627	4 (4)
Crane	95	6.341729	813 (130)	95	7.674212	590 (106)
	125	6.323116	349 (73)	125	7.538770	184 (64)
Woman	65	3.312538	401 (35)	65	4.029445	329 (33)
	90	1.741426	141 (12)	90	2.139769	123 (13)

Table 1: Region growing on different images. Number of regions in parenthesis indicates number of regions after applying the median filter.

As it can be observed in table 1 using eight neighbourhood consumes more computational power, in comparison with four neighbourhood adjacency. Additionally, applying eight neighbourhood adjacency generates fewer number of regions, providing the same tolerance level.

Another observation on the table 1, shows that pure Region growing algorithm does not generate optimum result, due to the fact that many small unnecessary regions are created. Applying additional post-processing algorithms are required to polish the result. As it can be observed with the number in parenthesis in the ‘region’ column of table 1, after applying number of median filter we can enhance the results significantly.

Naturally, by applying better post-processing techniques we can boost the segmented results. We only tried median filter, which was computationally efficient. And it also reduced many unnecessary regions. Choosing how many times one must apply the median filter is also important. We automatically applied $n = numberofregions/20$ times median filter on each image. Therefore for the images with larger number of regions we applied the median filter more. And for those images with perfect number of regions – e.g. ‘colour’ – we did not apply median filter at all.

Another post-processing technique which can be applied is to eliminate the regions which have number of pixels lower than a chosen threshold, for example two per cent of total number of pixels. Those regions can be merged with their surrounding regions.

Comparing different tolerances on each image indicates the importance of choosing a correct tolerance on segmentation results. As it can be observed as threshold increased, number of regions decreases. However, it must be noted that most of the times it is better to over-segment the image than under-segment it. Afterwards, by applying post-processing techniques we can remove unnecessary regions or merge similar regions.

3 K-means

K-means is unsupervised clustering algorithm. K stands for number of clusters which is **user input** which disables this method of being automatic without additional “smart” predictions. It is iterative algorithm which tries to minimize distance between points within one cluster and mean (centroid) of that cluster. Being clustering method, it works in feature space and does not take into account any kind of spatial connectivity between points.

One of the drawbacks is **initialization of clusters (centroids of clusters)**. We used Matlab *kmeans()* function. It randomly initializes clusters. We experienced repetitive error *“Empty cluster created at iteration 1”*, because in some cases one of the clusters lose all its points after first iteration. We solved that by setting parameter *‘emptyaction’ = ‘drop’* (remove any clusters that become empty).

One of the advantages is **simple implementation**.

1 Coins

Since the aim is to imitate human brain in segmenting the scene, we could say that good segmentation result would have 11 regions (ten coins plus the background). However, from Figure 14 we can see that K-means gives bad result for eleven clusters while it gives quite good result for two clusters. The reason is that being clustering method, K-means did not take into account that regions are not spatially connected. In this case, Region growing is better choice.

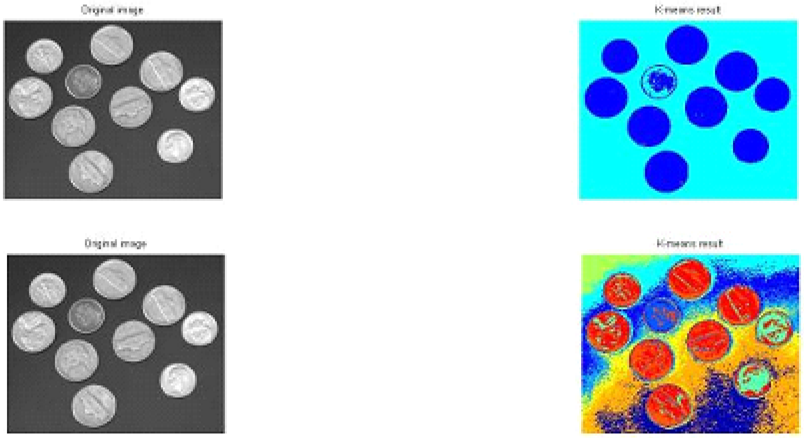

Figure 14: K-means on ‘coins’ image: results with **2** and **11** clusters, using Euclidean distance as metric.

2 Colour

Dependence on initialization of clusters can be observed in figure 15. We can clearly see how the result was getting better gradually with same number of clusters just with different random initializations.

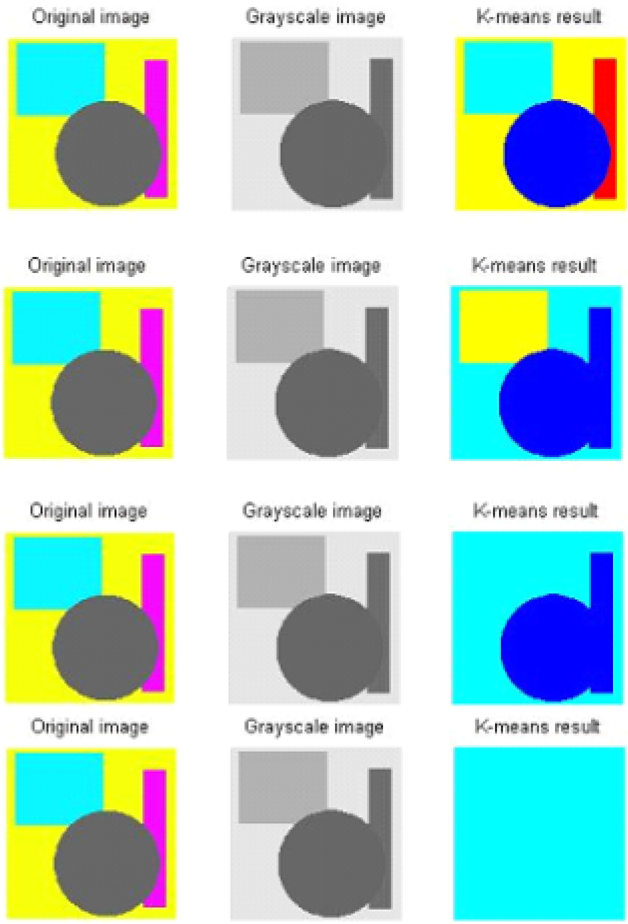

Figure 15: K-means on ‘colours’ image: four different results with same predefined number of clusters: **4** and using Euclidean distance as metric.

3 Crane

In figure 16 we successfully segmented the sky as one region. In this case, K-means method is better option since it neglects fact that parts of the sky are not spatially connected. That will never happen with region based methods such as Region growing.

Figure 16: K-means on 'crane' image: result with **4** clusters, using Euclidean distance as metric.

4 Woman

In figure 17, first we have ideally segmented background and then with the same number of clusters provided, we have very noisy background. Again result is highly dependant on cluster initialization.

In this case taking into account only feature space is advantage. For example in figure 17 (second case), whole blue shirt is detected as one region even though it is interrupted by hands.

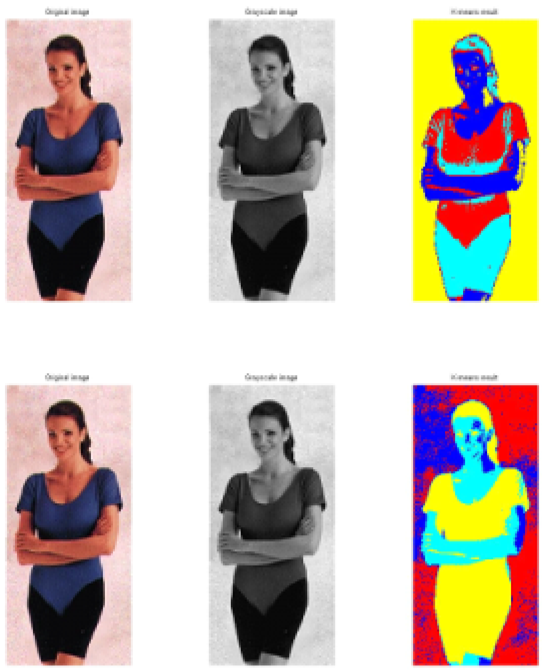

Figure 17: K-means on 'woman' image: two different results with same predefined number of clusters: **4** and using Euclidean distance as metric.

4 Conclusion

In this report we illustrated the results of two different segmentation techniques – Region growing and K-means – on four images. Each algorithm has its strengths and limitations. Therefore, we cannot conclude one algorithm is more efficient than the other one. Depending on the image information – i.e. distribution of objects, colours, edges, etc. – and application requirements one must choose the suitable algorithm.

We learnt that images with clear set of edges – e.g. the 'colour' image – can be successfully segmented by Region growing algorithm. Whereas in images with shades K-means performs better, for example the 'crane' image.

K-means is robust to regions spatially disconnected by another regions. For instance as it can be observed in figure 16 sky is detected as one uniform region. Whereas by applying Region growing, as it is illustrated in figure 6, sky is recognised as many separated regions.

The main drawback of K-means is its requirement to provide the number of clusters k. An inappropriate choice of k may yield poor results. That is why, when performing K-means, it is important to run diagnostic checks for determining the number of clusters in the data set. This issue is not a problem for Region growing, however this algorithm is dependant on choice of tolerance level and type of neighbourhood as well as distributions of seeds. Region growing method can be improved by some other ideas. For example by using contour information we could obtain smart initial seeds placement or make better decisions on classifying pixels.