

CHOOSING THE NUMBER OF LABELS IN IMAGE SEGMENTATION

Y. Liu, J.H. Draper, A.P. Gay, C.N. Howarth

R.R. Martin

Aberystwyth University, SY23 3DA, UK

Cardiff University, CF24 3AA, UK

ABSTRACT

Image segmentation is a fundamental yet challenging step during image analysis. In this paper we propose a novel method for choosing the number of labels during automatic image segmentation. It minimizes an objective function based on the number of labels, the segmentation errors, and consistency of labels between neighboring pixels. An experimental study on representative data shows encouraging results.

1. INTRODUCTION

Image segmentation usually uses intensity and color information to partition a given image into a limited, and preferably small, number of regions with meaningful structures. It is a fundamental step in image analysis and understanding, since it (1) identifies the number of objects, (2) localizes the objects in the image, and (3) provides information on the shape, appearance and other properties of these objects. However, it is a challenging problem, since the same structural parts can have different intensities and colors, and the same intensities and colors can represent different structural parts. These phenomena are caused by such factors as differences in reflectance of structural parts, non-uniform illumination, low contrast between foreground and background, and imaging noise. While some algorithms determine the number of segments as they proceed, others assume that the number of segments is given, or computed in some other way. Deciding *how many* segments there should be is just as challenging as deciding which pixels belong to each segment.

1.1. Previous work

Image segmentation is a tricky problem with a long history, but still remains a topic of significant current research. It has received intense attention from various communities such as those pursuing image processing, computer vision, pattern recognition, medical imaging, remote sensing, and biological imaging. Consequently, a large number of techniques have been proposed. They can be broadly classified into three main categories, or some combination of them: top-down, bottom up, and pixel relabeling methods:

- *Bottom up methods* typically start by considering pixels and labeling them, and then merge labelled regions gradually, using some criterion of consistency between these labels. The *watershed* method [9] is a popular example of a bottom up method. It first finds local minima of pixel values, and then keeps adding pixels to regions starting with seed pixels to progressively fill up different catchment basins, resulting in the image being divided into different segments. The watershed method usually over-segments images. In [3], a weighted graph is constructed that represents the local topology of pixels. After sorting the edges in ascending order according to their weights, they are then considered in turn. If the vertices at the ends of an edge currently belong to different segments, but are similar enough, they are merged into the same segment.
- *Top down methods* typically treat the whole image as a single segment and then split it into further segments recursively. A novel top-down region dividing (TDRD) approach is given in [10] which iteratively divides sub-regions if the size of a sub-region is larger than a predefined threshold or the homogeneity of a sub-region is less than a predefined threshold. The TDRD method produces good results at low computational complexity.
- *Pixel relabeling methods* consider consistency between a pixel of interest and a reference, unlike the previous two kinds of method which typically consider consistency between neighboring pixels during segmentation. The k -means method is probably the most widely used pixel relabeling method. However, it requires initialization of k means, which typically must be done in a data dependent way, and is thus a challenging problem in its own right. Consequently, it usually converges to a local minimum. For example, in [2], the k -means method is used to segment color images in HSV space after initialization using the maxmin algorithm [7]. The majority rule is then used to post-process the segmentation results. Recently, a new method has been proposed in [6] to initialize k -means, which calculates and sorts the means of the pixels of different frame sizes in ascending order. If two

means are similar, one of them is removed. The number of means left represents the number of clusters and these means are used as the centroids of these clusters. An alternative *iterated conditional modes* segmentation approach is given in [1], based on Markov random fields. It considers both data fitting errors and a local inconsistency cost. After initialization, it iteratively re-labels pixels by considering the benefit of replacing old labels by new ones.

1.2. Our approach

We now explain our approach to solving the problem of determining the number of segments. We formulate image segmentation as the problem of minimizing weighted average of errors produced by candidate segmentations. Our algorithm is inspired by the work in [5] on the automatic registration of overlapping range images represented as 3D point clouds. It has the advantage of simultaneously estimating the size of the overlap between two range images being registered, and finding a rigid 3D transformation that brings them into alignment.

We first construct a *dissimilarity* map that describes the difference in intensity between a pixel of interest and its neighbors and a *histogram* of the intensities of the pixels in the image. We then detect the bins belonging to the left (rising) slope of each peak in the histogram. Such bins are represented by their centers and used as tentative labels. These bins are sorted in descending order according to their frequencies. Then the k bins with greatest frequency are used as labels as a basis for segmentation of the image; we discuss how k is determined automatically shortly. All other pixels are assigned that candidate label to which it is closest in intensity amongst the tentative labels, leading to a *label map* as well as an *error map*. The error map describes the difference between the intensity of a pixel of interest and that of its candidate label. It is next processed by considering the consistency of labels of pixels and labels of their neighbors. If the label of a pixel is consistent with that of its neighbors, its error is decreased by its dissimilarity, otherwise, its error is increased by its dissimilarity. The total error over all pixels in the image can be computed as a function of k . We choose the number k of labels that produce a minimum error.

To analyse the performance of our new segmentation algorithm, we have compared it against an efficient graph based method [3], the iterated conditional mode (ICM) algorithm [1] and the k -means clustering method [2]. These methods belong to two of the three categories defined above; despite their age, they are representative. The graph based method is a typical bottom-up method with the advantages of computational efficiency and ease of implementation. The ICM algorithm has been subject to intense attention by the image processing and computation vision com-

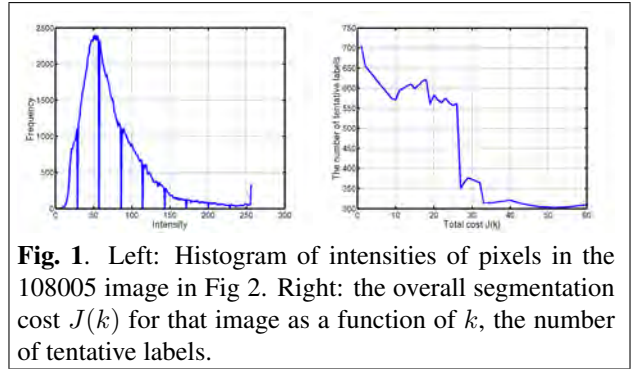


Fig. 1. Left: Histogram of intensities of pixels in the 108005 image in Fig 2. Right: the overall segmentation cost $J(k)$ for that image as a function of k , the number of tentative labels.

munities; it is probably one of the earliest methods based on Markov random fields. The selected algorithms were compared using the publicly accessible Berkeley Image Segmentation Database and data from [2].

Section 2 gives details of our new algorithm, Section 3 presents experimental results, and Section 4 draws some conclusions.

2. A NOVEL ALGORITHM

In this section, we give details of our algorithm which automatically determines the number of labels for image segmentation. The key ideas are the dissimilarity map, tentative segmentation, inconsistency penalization, and segmentation optimization, which we now discuss.

For simplicity, when we talk about intensity, we assume a grey-level image, which we derive from the color input images by RGB intensities. A more sophisticated approach considering the colour channels separately is clearly possible.

2.1. Dissimilarity map

We initially compute a dissimilarity map of the image. It essentially measures how dissimilar a given pixel is to its neighbors within a window; in this paper we use a 3×3 window centered at the given pixel. The dissimilarity map provides information about how consistent the final labels of the pixels are likely to be. For a given pixel (i, j) , we extract all its neighbors $N(i, j)$ within the window. The dissimilarity $d(i, j)$ of pixel (i, j) is then given by

$$d(i, j) = \frac{1}{|N(i, j)|} \sum_{(k, l) \in N(i, j)} |I(i, j) - I(k, l)|^\beta$$

where $|N(i, j)|$ is the number of neighboring pixels, $I(i, j)$ is the intensity of pixel (i, j) , and β is a positive parameter that controls the extent to which inconsistency in the final labels should be penalized.

2.2. Tentative segmentation

We next explain the tentative segmentation of a given image, which is done using the following steps.

- Construct the histogram H of the intensities of the pixels in the input image. Assume that the histogram has a scale of s ($s = 256$ here) and each bin is represented as (b_i, f_i) where b_i is the location of the bin, represented by its centre: $b_i = (i + 0.5)b$, where b is the bin size, and f_i is the frequency of that bin.
- Smooth the histogram: $f_i \leftarrow (f_{i-1} + f_i + f_{i+1})/3$.
- Extract all bins (c_m, g_m) ($m = 1, \dots, M$) in the left slope of each peak: $f_{i-1} < f_i < f_{i+1}$, and those at the two ends of the histogram: $f_0 > f_1$ and $f_{s-1} > f_{s-2}$.
- Sort c_m in descending order according to g_m .
- For each candidate number k of segments, consider the k highest frequency extracted bins. For $m = 1, \dots, k$, assign each pixel (i, j) the label l which minimizes the difference between its intensity and bin c_l , where $l = \operatorname{argmin}_{m \in [1, k]} |I(i, j) - c_m|$. Set the error map value for pixel (i, j) to $e(i, j) = |I(i, j) - c_l|^\beta$.

2.3. Inconsistency penalization

We now improve on the basic segmentation provided by the previous section by considering neighbors: we add a constraint so that neighboring pixels tend to have the same labels. We thus penalize all pixels which have different labels to their neighbors. We consider the neighbors $N(i, j)$ of pixel (i, j) . If the pixel has the same label as at least three of its neighbors, its error is reduced by its dissimilarity: $e(i, j) = e(i, j) - d(i, j)$; otherwise, its error is increased by its dissimilarity: $e(i, j) = e(i, j) + d(i, j)$. This penalization is dynamic and depends on the basic segmentation.

2.4. Segmentation optimization

For each label $l(i, j)$, we compute an associated error (cost) over the whole image as: $C_l = \sum_{(i, j), l(i, j)=l} e(i, j)$. Then we sort the costs in ascending order. Let the number of pixels with label l be n_l . The final total labeling cost for a given set L of tentative labels is computed as $J(k) = \sum_{l=0}^k (n_l/n_p)^\gamma C_l/n_l$ which clearly depends on k . The parameter γ controls the relative contribution of different labels to the final objective function. n_p is the total number of pixels in the image. Finally, we choose the labeling L which minimizes $J(k)$.

Table 1. The numbers M and k of initial and final labels, precision, recall, and times of different methods for the construction of the histogram when segmenting the test 108005 image.

Slope	M	k	Precision	Recall	Time
Left	60	51	4.95%	48.29%	18.50s
Right	101	57	4.05%	43.16%	32.19s

3. EXPERIMENTAL RESULTS

To test the performance of the proposed segmentation algorithm, the graph based method [3], iterated conditional mode (ICM) method [1], and the k -means clustering method [2] were chosen for a comparative study based on images 219090, 126007, 108005, 208001, and 175032 downloaded from the Berkeley image segmentation database, and the *cap* image from [2]. The ICM method was initialized using all the peaks in the histogram of intensities of pixels in an image with a scale of 256. All the images from the Berkeley image segmentation database have ground truth for the boundaries between different segments. The performance of the algorithms was compared using three quantities: precision, recall, and processing time in seconds. Precision and recall [4] are defined as: Precision= $p_c/p_s \times 100\%$ and Recall= $p_c/p_g \times 100\%$ where p_c is the number of common pixels on the boundaries in the segmented and ground truth boundary images, p_s is the number of boundary pixels in the segmented image, and p_g is the number of boundary pixels in the ground truth boundary image. The precision measures the extent to which the segmented boundaries are expected, while the recall measures the extent to which the expected boundaries have been identified. To aid visualization of the experimental results, we also show the resulting segmentation boundaries. See Figures 2–7 and Tables 1–3.

3.1. Left or right slope

While we used the bins in the left slope of each peak for tentative segmentation, it seems that this was an ad hoc choice. Thus, in this section, we compare the results with those obtained from the bins in the right slope of each peak. To this end, the 108005 image and its ground truth segment boundary image were selected. The experimental results are presented in Figures 1–2 and Table 1.

Fig. 1 shows the total cost of segmentation of the 108005 image as a function of a given set of tentative labels. It is not smooth, due to the large amount of detail in the image. It reaches a global minimum at $k = 51$. Fig. 2 shows that the left slope method clearly identifies the main parts of the tiger and tree. In contrast, the right slope method over seg-

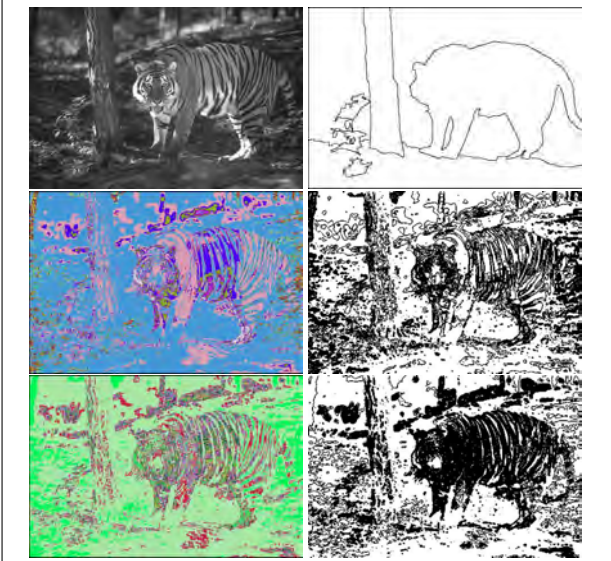


Fig. 2. Segmentation results using different methods for tentative segmentation applied to the 108005 image. Top row: original image. Middle row: left slope method. Bottom row: right slope method. Left column: segmented image. Right column: segment boundaries.

mented the stripes of the tiger and the ground above it.

Table 1 shows that the left slope method is more accurate and efficient than the right slope method. While the latter finds 12% more labels than the former, the former is actually more accurate than the latter in the sense of either precision or recall of the boundaries of the segments. The former increases the precision and recall of the latter by 22% and 12% respectively, and reduces the computational time by 43%. These results show that the left slope method performs better than the right slope method, thus justifying our choice.

3.2. Parameter β

Parameter β characterizes the extent to which neighboring pixels are considered dissimilar and thus, plays an important role in optimization of labels during image segmentation. In general it is difficult to optimize its value in advance due to data dependence, but here we experimentally investigate suitable choices for β . To this end, the 208001 image and its ground truth segment boundary image were used; the experimental results are presented in Figure 3 and Table 2.

Figure 3 and Table 2 shown that if β is made too small, e.g. $\beta = 0.5$, the segmentation results are visually unsatisfactory. The algorithm identifies only a few leaves on the ground. Indeed, when $\beta = 0$, the dissimilarity $d(i, j)$ is a constant, and so fails to characterize the dissimilarities between neighboring pixels. However, when $\beta = 1.5$, the proposed method correctly identifies the mushroom and tree

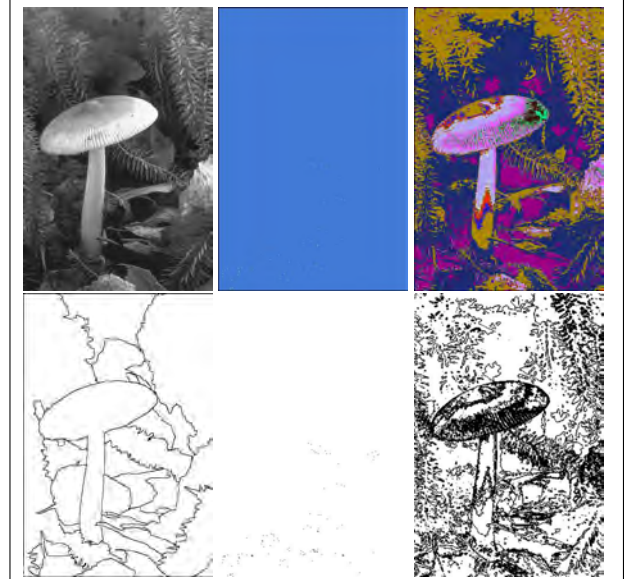


Fig. 3. Segmentation results for our method with different values for β , applied to the 208001 image. Left column: original image; Middle column: $\beta = 0.5$; Right column: $\beta = 1.5$. Top row: segmented image; Bottom row: segment boundaries.

Table 2. The numbers M and k of initial and final labels, precision, recall, and time of the proposed method for parameters β and γ taking different values when segmenting the test 208001 and 175032 images respectively.

Para.	Value	M	k	Precision	Recall	Time
β	0.5	57	9	28.81%	0.33%	19.51s
	1.5	57	37	8.09%	41.53%	16.70s
γ	0.5	83	5	3.84%	27.75%	31.45s
	1.5	83	76	5.42%	91.29%	31.58s

branches and leaves. In fact, as long as β is large enough, the algorithm is not too sensitive to the choice of β . This robustness of the algorithm is desirable in that it leads to stable behavior in segmenting various images. In other experiments, $\beta = 1.5$ was used.

3.3. Parameter γ

Parameter γ balances the contribution of dissimilarities of pixels within the same and different classes. Again, we experimentally investigate a suitable choice of value. To this end, the 175032 image and its ground truth segment boundary image were selected. The experimental results are presented in Figure 4 and Table 2.

From Figure 4 and Table 2, it can be seen that a large $\gamma = 1.5$ significantly reduces the effect of the number of pixels in different classes on the defined objective func-

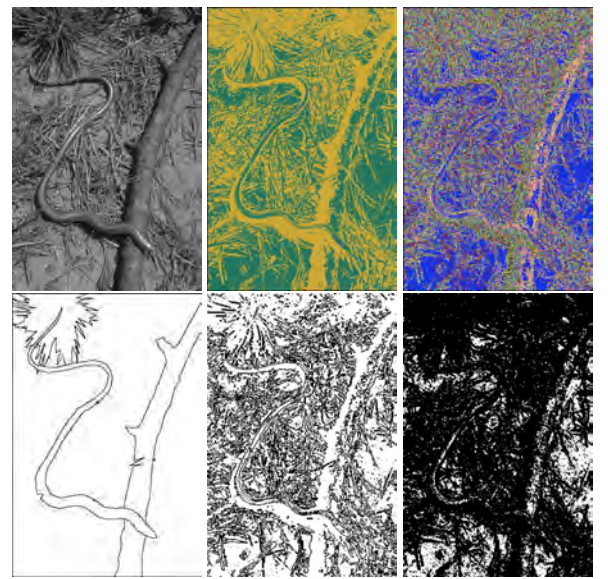


Fig. 4. Segmentation results for our method with different values of γ , applied to the 175032 image. Left column: original image; Middle column: $\gamma = 0.5$; Right column: $\gamma = 1.5$. Top row: segmented image; Bottom row: segment boundaries.

Table 3. The number k of labels (Ls) or regions (Rs), precision, recall, and time for different methods when segmenting test images.

Image	Method	k	Precision	Recall	Time
219090	New	36(Ls)	5.85%	41.84%	30.87s
	Graph	162(Rs)	4.45%	17.00%	2.27s
	ICM	30(Ls)	4.60%	62.26%	17.61s
126007	New	79(Ls)	8.35%	39.46%	26.23s
	Graph	129(Rs)	12.04%	28.17%	5.23s
	ICM	49(Ls)	7.73%	73.32%	20.28s

tion, leading to over-segmented results. Even though the resulting precision and recall rates are higher than those obtained for a smaller $\gamma = 0.5$, the structure of the snake and the woodlog have been largely destroyed and broken into pieces, rendering subsequent processing and image understanding more challenging. Thus, $\gamma = 0.5$ is used for other experiments in the rest of the paper.

3.4. Comparative study

In this section, we compare the graph based method, ICM, and the k -means method using three images: 219090, 126007, and cap. The experimental results are presented in Figs. 5–7 and Table 3.

Fig 5 shows that the proposed method clearly identifies the main parts of the sky, mountain, water, and boat. In

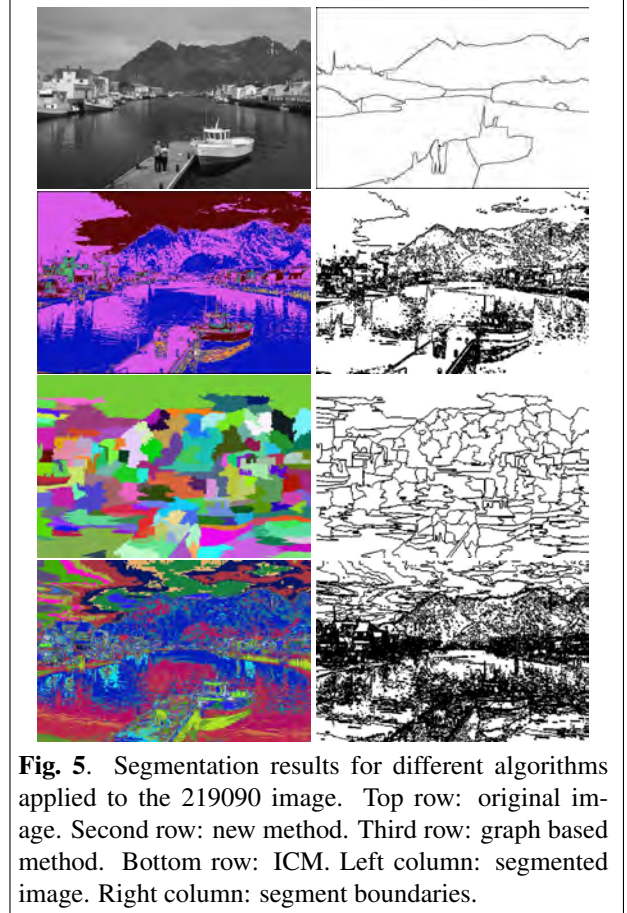


Fig. 5. Segmentation results for different algorithms applied to the 219090 image. Top row: original image. Second row: new method. Third row: graph based method. Bottom row: ICM. Left column: segmented image. Right column: segment boundaries.

contrast, the graph based method under-segments the image, leaving the very little recognizable, while the ICM algorithm over-segments the image, especially in the sky region, and blurs the boundary between the sky and the mountain.

The graph based method misses the cloud in the sky in the 126007 image, while the ICM method over-segments the top-left part of the sky. Our method clearly segments the cloud and sky but over-segments the tree canopy at the top-right. This is partly due to blocking effects caused by JPEG compression of the input image. Table 3 shows that our method has detected more accurate boundaries and the ICM method has detected a large number of false boundaries, while the graph based method clearly produces the worst results. The reason why the precision of all methods is low is because the ground truth boundaries are provided by human subjects, who identified only the main boundaries and omitted all detailed and smaller ones. Even worse, different subjects usually have different opinions on the definition of the main boundaries. This shows that image segmentation is still a challenging problem.

Compared to the k -means method, our method produces very good results even though it does not use color informa-

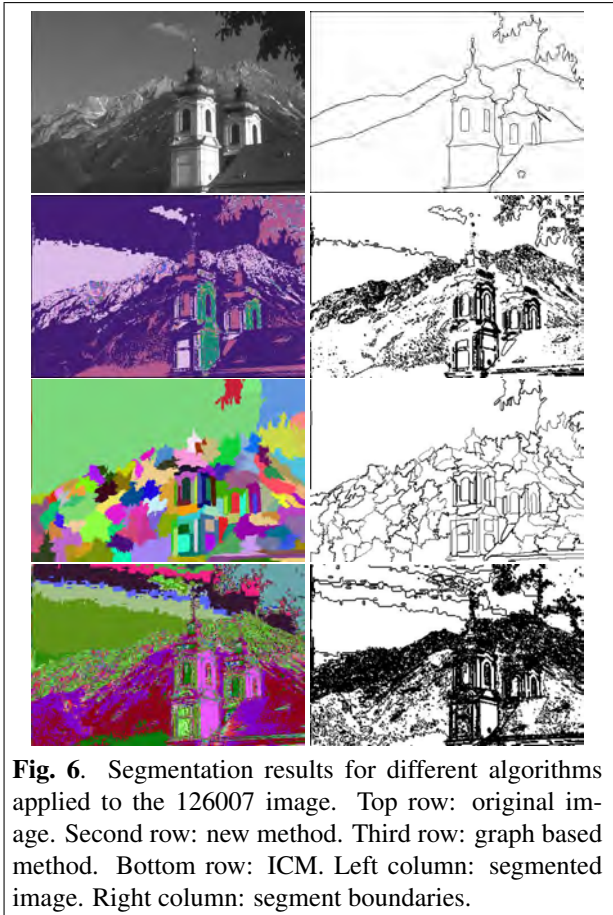


Fig. 6. Segmentation results for different algorithms applied to the 126007 image. Top row: original image. Second row: new method. Third row: graph based method. Bottom row: ICM. Left column: segmented image. Right column: segment boundaries.

tion. While the former under-segments the image, our approach picks up a lot of details in the sky, wall and shadow. Over-segmentation is typically more useful to downstream applications than under-segmentation: the latter may mislead end users. A careful analysis of Figures 5–7 also shows that the boundaries between different segments usually correspond to sharp changes in detail in the input images.

4. CONCLUSIONS

We have formulated automatic image segmentation as a problem of *simultaneously* optimizing the number of labels and the segmentation arising therefrom. Such a formulation provides a better characterisation of the aims of the image segmentation problem. It significantly reduces human invention needed to achieve better segmentation results. A comparative study based on various images shows that the proposed algorithm gives encouraging results. Further research will refine the definition of dissimilarities between neighboring pixels in an image and apply the proposed method for the analysis of the data captured by a stereo camera system for 3D oat phenotyping [8].

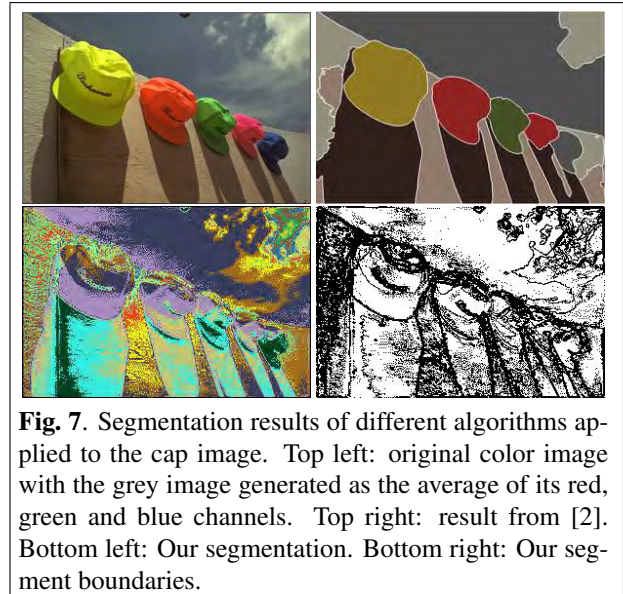


Fig. 7. Segmentation results of different algorithms applied to the cap image. Top left: original color image with the grey image generated as the average of its red, green and blue channels. Top right: result from [2]. Bottom left: Our segmentation. Bottom right: Our segment boundaries.

5. REFERENCES

- [1] J. Besag. On the statistical analysis of dirty pictures. *J. Royal Statistical Society: Series B* 48(1986) 259-302.
- [2] T.-W. Chen, Y.-L. Chen, S.-Y. Chien. Fast image segmentation based on k -means clustering with histograms in HSV color space. *Proc. IEEE 10th Workshop on Multimedia Signal Processing (MMSP)*, 2008, pp. 522-525.
- [3] P.F. Felzenszwalb, D.P. Huttenlocher. Efficient graph-based image segmentation. *IJCV* 59(2004) 167-181.
- [4] F.C. Monteiro and A.C. Campilho. Performance evaluation of image segmentation. *Proc. Int. Conf. Image Analysis and Recognition*, 2006, pp. 248-259.
- [5] J.M. Phillips, R. Liu, C. Tomasi. Outlier robust ICP for minimizing fractional RMSD. *Proc. 3DIM*, 2007, pp. 427-434.
- [6] A.S. Samma and R.A. Salam. Adaptation of k -means algorithm for image segmentation. *World Academy of Science, Engineering and Technology* 50(2009) 58-62.
- [7] V. Mezaris, I. Kompatsiaris, and M. G. Strintzis. Still image segmentation tools for content-based multimedia applications. *Int. J. Patt. Rec. Art. Int.*, 18(2004) 701-725.
- [8] R.N. Vankadavath, et al. Computer aided data acquisition tool for high-throughput phenotyping of plant populations. *Plant Methods*, 5(2009) 18
- [9] L. Vincent and P. Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE Trans. PAMI*, 13(1991) 583-598.
- [10] Y.-T. Wu, F.Y. Shih, J. Shi, Y.-T. Wu. A top-down region dividing approach for image segmentation. *Pattern Recognition* 41(2008) 1048-1060.