Hierarchical Layered Mean Shift Methods

Milan Šurkala, Karel Mozdřeň, Radovan Fusek, and Eduard Sojka

Faculty of Electrical Engieneering and Informatics,
17. listopadu 15, 708 33 Ostrava-Poruba, Czech Republic
{milan.surkala,karel.mozdren,radovan.fusek,eduard.sojka}@vsb.cz

Abstract. Many image processing tasks exist and segmentation is one of them. We are focused on the mean-shift segmentation method. Our goal is to improve its speed and reduce the over-segmentation problem that occurs with small spatial bandwidths. We propose new mean-shift method called Hierarchical Layered Mean Shift. It uses hierarchical pre-processing stage and stacking hierarchical segmentation outputs together to minimise the over-segmentation problem.

Keywords: layer, segmentation, image, mean shift, hierarchical.

1 Introduction

Segmentation is one of the constantly developing image processing tasks. The goal is to improve not only the accuracy and the segmentation quality but also the speed of algorithms. We are focused on the one of the most popular segmentation methods, the Mean Shift. It was released in 1975 [7], but it started to be developed more 20 years later, in 1995 [2]. The most important papers about this method are, for example, [3], [5], and [4].

The main idea of the mean-shift method is in iterative motion of data points to the position of their highest density. We are segmenting images, therefore, these data points are image pixels. For each pixel, a kernel density estimate is computed and it is shifted according to this estimate. It is repeatedly computed until the point converges to an attractor, the place of the highest density of pixels. In general MS [7], we need two datasets. The first one is the original dataset that is used to compute the density estimate of the data points and the second one holds the shifted values (the actual data points for which the density estimate is computed). If we use Blurring MS [1], only one dataset is needed. The source dataset is replaced by the computed values after each iteration. The BMS method also has the smaller number of iterations per data point.

The hierarchical approaches [11] [10], [6] showed to be a very fast way to accelerate the algorithms with a very small influence on the segmentation quality. They use more stages of the algorithm with different bandwidths and they use the output from the first stage as the input for the following one. A small bandwidth in the first stage ensures fast computation as well as the smaller input for the next one.

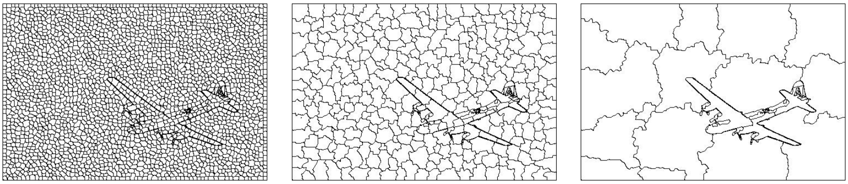
The layered approaches [9], on the other hand, run several MS computation and stack them together. If the pair of pixels belongs to the same segment

J. Blanc-Talon et al. (Eds.): ACIVS 2013, LNCS 8192, pp. 538–545, 2013.

in more resultant segmentations (threshold has to be set, for example, in two segmentations of the three processed), they are merged together.

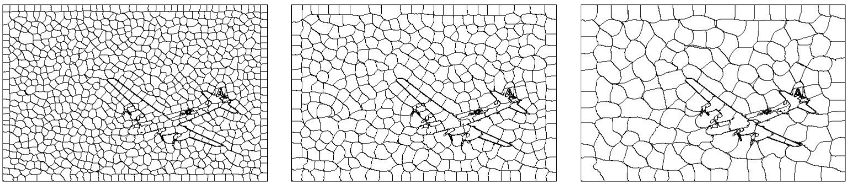
We present Hierarchical Layered Mean Shift (HLxMS) methods that use the hierarchical approach and stacking of segmentations to reduce the over-segmentation problem. In the next two sections, Mean Shift basics and our HLxMS approach will be discussed. In the Section 4, experiments will be carried out and the last section is devoted to the conclusion.

We are going to describe hierarchical layered mean-shift method that use the Blurring Mean Shift (BMS) method as its base. Therefore, we will denote it as HLBMS (the "B" standing for blurring is replacing the general "x" in HLxMS notation covering all hierarchical layered versions). We will also describe only the BMS method deeply, although the experiments will be also carried out with general MS [2] (HLMS) and Evolving Mean Shift [12] (HLEMS) method too.



(a) 1st stage (b) 2nd stage (c) 3rd stage

Fig. 1. Stages of the HBMS method



(a) 1st iteration (b) 2nd iteration (c) 3rd iteration

Fig. 2. Iterations of the LBMS method

2 Mean Shift Methods

If $X = \{x_i\}_{i=1}^{n} \subset R^d$ is a dataset of n points in the d-dimensional space, the *kernel density estimator* for BMS method is given by the following equation

$$p(x) = \frac{1}{n\sigma^d} \sum_{i=1}^{n} K\left(\frac{x - x_i}{\sigma}\right), \tag{1}$$

where the first fraction is a normalisation constant and σ is the bandwidth. It sets the diameter of the searching window (kernel function $K(x)$). In digital images, we use two types of bandwidths. The spatial bandwidth σ_s is limiting the neighbourhood of the processed pixel in x and y axis (usually, it has the same values for both axes). The range bandwidth σ_r indicates the maximum possible luminance difference between the processed pixel and the pixels in its neighbourhood. MS can use broad kernels (for example, the *Gaussian*) that cover all the data points and σ parameters change only the shape of kernel as all pixels are involved in computation. If we use truncated kernels (for example, the *Epanechnikov*, *uniform*), the bandwidth parameters really limit the size of kernel. In our algorithm, we use only truncated kernels, because it is based on using the small kernels that improve the speed of the algorithm. The Epanechnikov kernel is given by the equation

$$K(x) = \begin{cases} 1 - x^2, & \text{if } \|x\| \le 1 \\ 0, & \text{otherwise} \end{cases} . \tag{2}$$

The *mean-shift vector* that is iteratively needed for each pixel in the processed image, is given by

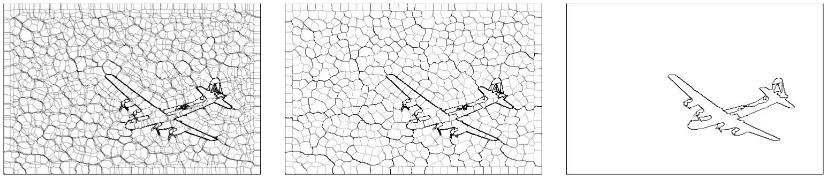
$$m_{\sigma,k}(x) = \frac{\sum_{i-1}^{n} x_i k\left(\left\|\frac{x-x_i}{\sigma}\right\|^2\right)}{\sum_{i-1}^{n} k\left(\left\|\frac{x-x_i}{\sigma}\right\|^2\right)} - x\,, \tag{3}$$

where the function $k(x)$ is a derivative of the kernel $K(x)$. This equation indicates the difference between the former position of the processed pixel x (on the right-hand side of the equation) and a new position of the processed pixel x (estimate of the position with the highest density of data points). In each iteration, the point is moved to the new computed position until the movement is zero or close to zero (smaller than a preset threshold). Each iteration consists of moving all the points according to their mean-shift vectors, then the output of this iteration is converted to the input for the next one.

3 Hierarchical Layered Mean Shift

Our new method called Hierarchical Layered Mean Shift (HLxMS in general) combines the hierarchical [11] [10], [6] and layered approach [9]. The hierarchical approach divides the segmentation of the image into several stages. The first stage use very small spatial bandwidth and, therefore, it is carried out very fast. It creates a large number of small segments. We consider the segment as one data point with the weight proportional to the number of points it contains. This output is used as the input for the next stage with a larger spatial bandwidth. Because this input is smaller (due to preprocessing in the first stage), the next stage can be carried out very fast too even if the larger spatial bandwidth is used (the dataset is smaller). The number of stages is not limited, mostly two or three stages are enough (it depends on the size of the processed image). HBMS results can be seen in Fig. 1.

The layered approach uses a larger number of stages on the same dataset (not for outputs from the previous stages). We carry out computation with several



(a) stacked LBMS segmentation

(b) stacked HLBMS segmentation

(c) merged HLBMS segmentation

Fig. 3. Stacked LBMS and HLBMS (hierarchical LBMS) image and the result after merging the HLBMS segments

small, but different bandwidths. Each result has the same boundaries around the significant objects but it has variously shaped segments in the flat areas of the image. If we stack the segmentation results, the most significant boundaries would be clearly seen (Fig. 3(a)). The *merging algorithm* is straightforward. If two random pixels are, for example, twice in the same segment in two different segmentations of three processed segmentations, they are assigned to one bigger segment. The number of segmentations does not necessarily need to be 3. In practice, some images could be processed with three segmentations very well, but mostly four segmentations are better.

We can carry out much more segmentations and we need to set the threshold t lower than this number to denote the number of segmentations, where two random pixels need to be in the same segment in order to assign them to the same final segment. We have to check all the pairs of pixels that are in the distance equal or lower than the maximal used spatial bandwidth (we do not need to check all the pairs of pixels in the processed image). It greatly reduces the computational cost of the algorithm with no influence to the segmentation result.

This approach needs to compute several very fast segmentations (because of small σ_s) and almost completely reduces the problem of over-segmentation. The flat areas are merged together. It can be used with BMS and EMS, but the character of MS (general Mean Shift) segments is not very suitable for this approach - much larger initial bandwidth is necessary.

HLxMS combines both approaches together. It carries out one very fast initial segmentation with a very small bandwidth. Its output is used as the input for two or more layered segmentations. That is the difference. General layered approach uses the original image as an input, whereas the hierarchical layered approach uses the first preprocessing stage for creating its input as in the hierarchical versions of mean-shift methods. Simply, the hierarchical approach makes one over-segmented image and the layered segmentation post-processing will decide which segments should be grouped and which boundaries should be preserved. Of course, more hierarchical stages can be executed in order to minimize the computational time if we process very large images.

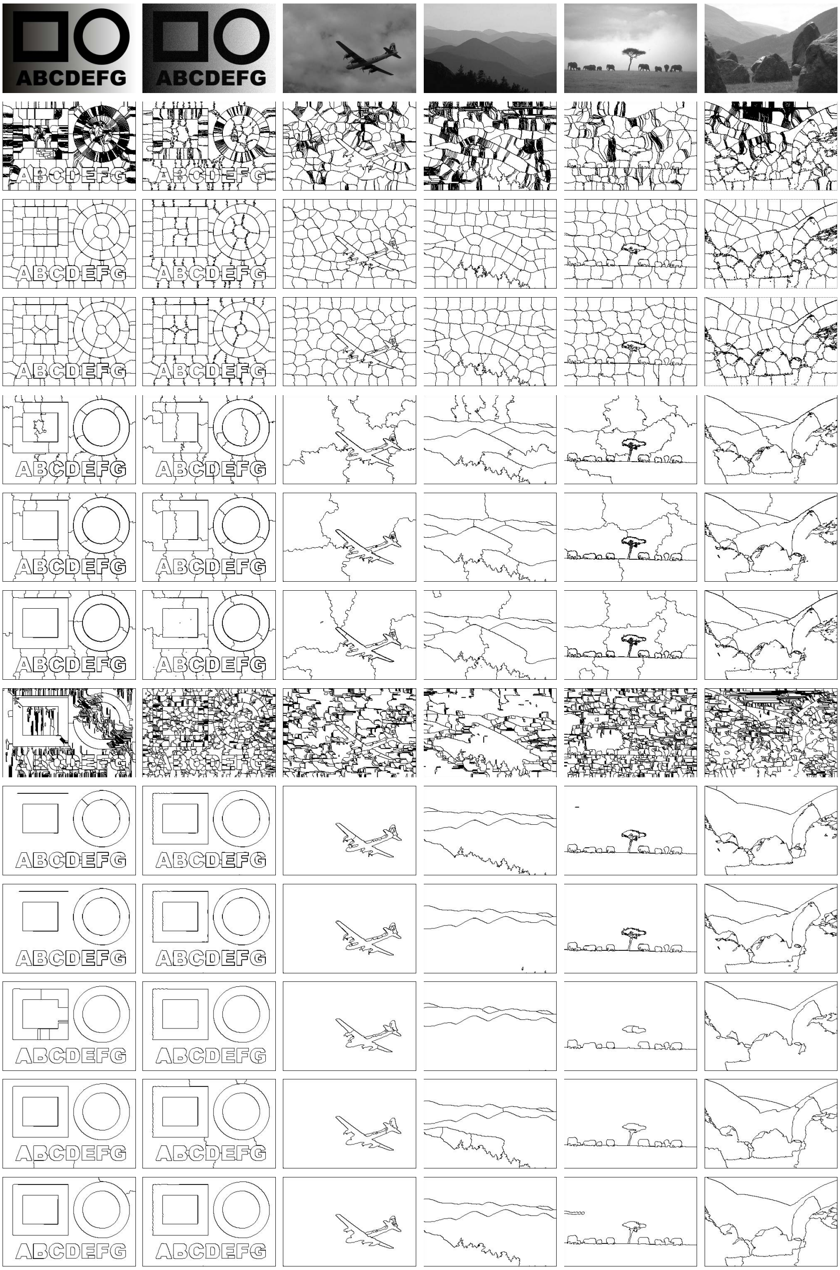


Fig. 4. Rows 1: the original image; 2/3/4: MS/BMS/EMS (spatial bandiwdth $\sigma_s = 15$); 5/6/7: HMS/HBMS/HEMS ($\sigma_s = 4/16/64$); 8/9/10: LMS/LBMS/LEMS ($\sigma_s = 4$, multiplier of the bandwidth $mul = 1.4$); 11/12/13: HLMS/HLBMS/HLEMS ($\sigma_s = 3.5$, $mul = 1.35$).

4 Experiments

For comparison, we present the achieved results with various mean-shift methods, especially general MS, Blurring MS and Evolving MS as our method is aimed as an improvement to Mean Shift. Therefore, we will compare it only with several mean-shift methods. All their hierarchical (HxMS) and layered versions (LxMS) are presented too. Our hierarchical layered versions are denoted by HLxMS. As it was already said, in all cases, the "x" letter stands for arbitrary MS method (general, blurring or evolving). Because each type of algorithms has another properties, they will be used with different spatial bandwidths that will be mentioned and deeply described later. Testing images are presented in the first row of Fig. 4, we use the images from Berkeley Image Dataset [8] and our synthetic image in noise-free and noisy version. All images were downscaled to the resolution 320 × 214 pixels.

Table 1. The comparison of speed depending on algorithm

	synthetic	syn. noise	airplane	hills	savana	bird
MS	1948.3 s	983.6 s	1199.3 s	1923.6 s	1564.7 s	1411.1 s
HMS	127.1 s	37.3 s	37.8 s	36.6 s	39.2 s	42.1 s
LMS	3472.9 s	498.6 s	1139.4 s	2148.9 s	782.4 s	1307.9 s
HLMS	261.8 s	46.5 s	57.1 s	55.5 s	56.6 s	62.7 s
BMS	31.8 s	37.5 s	34.7 s	40.2 s	34.7 s	52.4 s
HBMS	8.7 s	10.1 s	9.6 s	8.9 s	8.2 s	9.9 s
LBMS	42.2 s	43.4 s	46.1 s	44.3 s	41.1 s	45.7 s
HLBMS	6.4 s	5.6 s	6.1 s	6.4 s	6.1 s	6.9 s
EMS	812.2 s	603.8 s	168.3 s	198.9 s	160.9 s	184.9 s
HEMS	29.7 s	33.8 s	18.8 s	17.3 s	20.4 s	20.6 s
LEMS	561.5 s	440.2 s	149.2 s	156.2 s	153.2 s	158.7 s
HLEMS	20.3 s	20.7 s	16.8 s	17.0 s	17.8 s	17.3 s

In Fig. 4, the results of all mentioned methods are clearly visible. The computational times are listed in Table 1. All algorithms used the range bandwidth $\sigma_r = 24$, the spatial bandwidths are clearly described in the caption of Fig. 4. We will justify our choice of the bandwidth in the following paragraphs.

Original methods (MS, BMS, and EMS) used the spatial bandwidth of only $\sigma_s = 15$. It is obvious that the result suffers from a heavy over-segmentation and the computational times are very long even with such a small bandwidth. Hierarchical approaches (HMS, HBMS, and HEMS) started with $\sigma_{s_1} = 4$ and ended with $\sigma_{s_3} = 64$, fast three-staged versions of the algorithm was used (2-staged versions are usually slower). Because of the better speed of the algorithm, we could afford to enlarge the final spatial bandwidth to 64 in order to lower the over-segmentation problem.

Layered algorithms (LMS, LBMS, and LEMS) used the initial bandwidth $\sigma_s = 4$ and each following stage used enlarged bandwidth by the multiplier of 1.4. All algorithms were run in 4-staged mode, where pixels that were three times in the same segmentation, were merged. Even though 3-staged type of the algorithm can be successfully used too, we chose the 4-staged version because of its lower sensitivity to the parameter settings and greater stability.

Hierarchical layered version (our new presented algorithm) used even smaller initial bandwidth $\sigma_s = 3.5$ and the slightly smaller multiplier 1.35. This multiplier is applied once more for the first layered stage. Therefore, the initial stage used $\sigma_s = 3.5$, the first layered stage ran with $\sigma_s = 3.5 \times 1.35 \times 1.35 = 6.4$, the second layered stage used $\sigma_s = 6.4 \times 1.35 = 8.6$ and the last one ran with the bandwidth $\sigma_s = 8.6 \times 1.35 = 11.6$ in our tests. Note that only three layered segmentations are sufficient in the hierarchical layered approach.

General MS is very good algorithm for filtration but its segmentation results are not great in digital images, especially in the flat areas with a small spatial bandwidth. Blurring MS and Evolving MS are much faster but suffer from the over-segmentation problem too. We can enlarge their spatial bandwidth but the computation time will enlarge too. Hierarchical approaches can use much larger bandwidths and achieved dramatically better computational times.

Layered MS is unusable with the general MS as its base method when using small kernel sizes as many variously shaped segments emerge. Layered BMS and Layered EMS give very nice segmentation results and their computational times are comparable with the original methods. Note that we used 4-staged algorithm but we can also use only 3 stages that would lead to much faster time. The drawback is in the need of careful setting of parameters in order to get properly segmented results (4 stages are more robust).

Hierarchical methods (HLxMS) are more robust and faster than the layered versions (LxMS). HLMS is slower than HMS, but it solves its over-segmentation problem. On the other hand, HLBMS and HLEMS are faster than original hierarchical methods and, moreover, also lower the problem of over-segmentation. We can say that our new method successfully suppresses the over-segmentation problem and also improves the speed when using Blurring MS or Evolving MS as its base method.

5 Conclusion

We have shown that the layered approaches can almost eliminate the over-segmentation problem in a reasonable time. The general layered versions are a little bit sensitive to parameter settings and the use of general MS is very problematic. Both BMS and EMS algorithms are useful in their layered versions. Hierarchical layered versions that we have presented in this paper do not suffer from the need of proper setting of parameters and they work very well with general MS even when the lower number of stages is used. It outperforms the hierarchical mean-shift methods in both areas of the speed and the segmentation quality. Even when the hierarchical versions use very large final bandwidth, they are not able to merge all segments in large flat areas.

References

1. Carreira-Perpiñán, M.: Fast nonparametric clustering with Gaussian blurring mean-shift. In: Proceedings of the 23rd International Conference on Machine Learning, ICML 2006, pp. 153–160. ACM, New York (2006)
2. Cheng, Y.: Mean shift, mode seeking, and clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence 17, 790–799 (1995)
3. Comaniciu, D., Meer, P.: Mean shift analysis and applications. In: The Proceedings of the Seventh IEEE International Conference on Computer Vision, vol. 2, pp. 1197–1203 (1999)
4. Comaniciu, D., Meer, P.: Mean shift: a robust approach toward feature space analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence 24(5), 603–619 (2002)
5. Comaniciu, D., Ramesh, V., Meer, P.: The variable bandwidth mean shift and data-driven scale selection. IEEE International Conference on Computer Vision 1, 438 (2001)
6. DeMenthon, D., Megret, R.: Spatio-Temporal Segmentation of Video by Hierarchical Mean Shift Analysis. Tech. Rep. LAMP-TR-090,CAR-TR-978,CS-TR-4388,UMIACS-TR-2002-68, University of Maryland, College Park (2002)
7. Fukunaga, K., Hostetler, L.: The estimation of the gradient of a density function, with applications in pattern recognition. IEEE Transactions on Information Theory 21(1), 32–40 (1975)
8. Martin, D., Fowlkes, C., Tal, D., Malik, J.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: Proc. 8th Int'l Conf. Computer Vision, vol. 2, pp. 416–423 (2001)
9. Šurkala, M., Mozdřeň, K., Fusek, R., Sojka, E.: Layered mean shift methods. In: Pack, T. (ed.) SSVM 2013. LNCS, vol. 7893, pp. 465–476. Springer, Heidelberg (2013)
10. Vatturi, P., Wong, W.-K.: Category detection using hierarchical mean shift. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2009, pp. 847–856. ACM, New York (2009)
11. Šurkala, M., Mozdřeň, K., Fusek, R., Sojka, E.: Hierarchical blurring mean-shift. In: Blanc-Talon, J., Kleihorst, R., Philips, W., Popescu, D., Scheunders, P. (eds.) ACIVS 2011. LNCS, vol. 6915, pp. 228–238. Springer, Heidelberg (2011), `http://dl.acm.org/citation.cfm?id=2034246.2034270`
12. Zhao, Q., Yang, Z., Tao, H., Liu, W.: Evolving mean shift with adaptive bandwidth: A fast and noise robust approach. In: Zha, H., Taniguchi, R.-i., Maybank, S. (eds.) ACCV 2009, Part I. LNCS, vol. 5994, pp. 258–268. Springer, Heidelberg (2010)