HIERARCHICAL EVOLVING MEAN-SHIFT

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

VŠB-Technical University of Ostrava, FEECS
17. listopadu 15, 708 33 Ostrava-Poruba, Czech Republic
{milan.surkala, karel.mozdren, radovan.fusek.st, eduard.sojka}@vsb.cz

ABSTRACT

Segmentation and filtration are widely discussed problems in image processing. Mean shift and its variants belong to the most popular methods in this area. In this paper, we propose a new variant of relatively new evolving mean shift that is based on the idea of minimization of dataset energy given by the sum of sizes of the mean-shift vectors. Our hierarchical EMS is focused on a significant reduction of computational time due to hierarchical evolution of the size of the kernel. We also present acceleration of precise point selection and vector recalculation, which can be applied also to original EMS.

Index Terms— evolving, hierarchy, image, mean-shift, segmentation

1. INTRODUCTION

For the purpose of segmentation and filtration of data, many methods exist. The mean-shift idea (nowadays known as *blurring mean shift*) appeared in 1975 [1]. Different version was later presented by Cheng in 1995 [2] and Comaniciu and Meer [3], [4]. This method seeks for the positions with the highest densities of data points. Positions of point convergence are called attractors, which are representatives of all points that converged to them and mark them as cluster members. Many variations of the MS method have been proposed. They improve the segmentation results, speed or both. One of the most popular methods is *Gaussian Blurring MS* (GBMS) that was presented in 2006 [5]. This method is based on the method [1] and slightly changes handling the dataset in comparison with MS because it replaces the source data for each following iteration with the output from the previous one. It makes it possible to achieve a lower number of iterations per data point and it makes BMS faster than MS.

Among many others, *Evolving MS* (EMS) is one of the latest variations of the MS method [6]. The MS vector influences an energy between the data points and EMS is formulated as an energy minimization problem. In each iteration, only one data point is shifted and the energies are recomputed. EMS needs a smaller number of iterations per point than BMS, but it suffers from a higher overhead because of the need to search for the maximum energy and recalculation.

The MS methods are described in Section 2. In Section 3, we propose our Hierarchical EMS (HEMS) method in detail and we present some efficient ways of algorithm acceleration. Section 4 is devoted to experiments carried out with our method; we discuss our results there too.

2. MEAN-SHIFT METHODS

Consider $X = \{x_n\}_{n=1}^{N} \subset R^d$ as a dataset of N points in the d-dimensional space. We use the image pixels as data points. We define a *kernel density estimator* with a kernel $K(x)$ as

$$p(x) = \frac{1}{N} \sum_{n=1}^{N} K\left(\left\| \frac{x - x_n}{\sigma} \right\|^2 \right), \tag{1}$$

where N is the number of pixels (data points), x is a processed pixel and x_n are the pixels in the neighbourhood of the pixel x. The term σ is a bandwidth, which limits the size of the neighbourhood. We can distinguish between two types of bandwidths in images. Spatial limit (in x and y axis) is denoted by σ_s and range limit (luminance) is denoted by σ_r.

The kernel function $K(x)$ is used as a weighting function and many various types can be used. The simplest one is a *uniform* kernel which always takes values of 1. The *Epanechnikov* kernel is probably the most popular kernel in the MS method because of very good speed, filtration and segmentation results. Both kernels are truncated. The kernel size is given by the σ_s parameter. The Epanechnikov kernel is described by the following equation

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

The last popular kernel is the *Gaussian*. It has very good filtration and segmentation results, but it is computationally very demanding. Slow speed is given by the fact that the Gaussian kernel uses all data points for computation (kernel is not truncated). The bandwidth only changes the shape of the Gaussian curve. For a better speed, the kernel is sometimes truncated, e.g., in a distance of 3σ.

In the classical MS method, the previously mentioned kernels are used and the MS vector is represented by the equation

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

In this equation, the term x is the former position of the processed pixel. The first term on its right-hand side is a new position of this pixel computed as a weighted mean of "similar" pixels (pixels that belong to the searching window). MS is an iterative process and we can stop the process if the shifts are small or zero. In such a case, the pixel converged to its attractor. MS and BMS are shifting the points until convergence, but MS uses original data as an input for all iterations, whereas BMS uses filtered data from the previous one.

EMS is a new method that was published in 2009 [6]. The size of MS vector influences the energy to be minimized. We search for the pixel with the highest energy in each iteration and we shift it according to the vector. All adjacent vectors have to be recomputed. The speed of EMS is highly dependent on the speed of searching for the maximum energy and recalculation. Therefore, the truncated kernels are better because the recalculation does not involve the whole dataset.

The energy of dataset is a sum of energies between all pixels and can be represented by the equation

$$E(X) = \sum_{i=1}^{N} \sum_{j=1, j\neq i}^{N} \left(E_{x_i x_j} + E_{x_j x_i}\right). \tag{4}$$

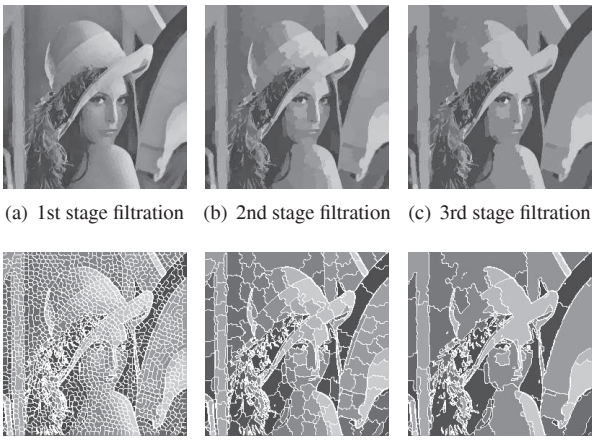
$E(X)$ is the sum of all energies. The energy in each pixel is described as the sum of all energies between this pixel and all pixels in its neighbourhood. The energy contributed by the pixel x_i to the pixel x_j is given by the equation

$$E_{x_i x_j} = k(0) - k\left(\frac{x_i - x_j}{\sigma}\right). \tag{5}$$

Hierarchical MS (HMS) was presented in 2009 [7]. It utilizes multiple MS steps (stages), each with different kernel size. The benefits of hierarchical approach will be described in Section 3 devoted to our HEMS method.

3. HIERARCHICAL EVOLVING MEAN SHIFT

Our new method called *hierarchical EMS* (HEMS) is focused on a significant reduction of computational time. We presented the hierarchical approach with introduction of HBMS in 2011 [8]. EMS is highly dependent on the size of searching window (kernel); it is very useful to keep it as small as possible. In such a case, we obtain a segmentation with a high number of small segments in a fraction of original time. The number of segments is lower than the number of source points in the first iteration. We consider these segments as source points for the next stage. The points are weighted according to the number of pixels that the segment contains (in the first iteration, data points are handled as one-pixel segments). In Fig. 1, an evolution of segments in each stage is shown.



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

(d) 1st stage segm. (e) 2nd stage segm. (f) 3rd stage segm.

Fig. 1. HEMS filtration and segmentation after each stage

In the following stages, the number of data points is reduced and, therefore, faster computation is achieved. Two stages can be sufficient for a small image. Because of a high dependence on the kernel size, it is appropriate to use a higher number of stages with larger images.

Algorithm 1 HIERARCHICAL EVOLVING MEAN SHIFT

Input: Dataset X_l^k, where k is an iteration index and l is a stage index.

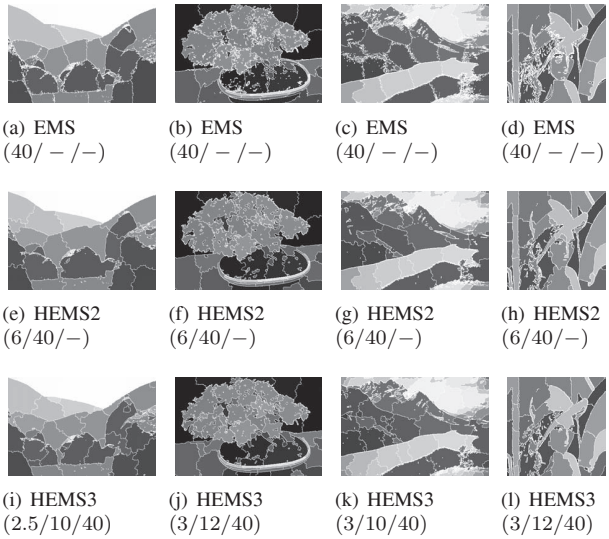
Output: A clustered dataset X_{EMS_l}

1: **repeat**
2: **repeat**
3: Select data point $x_i^k \in X_l^k$ with the highest energy and move x_i^k according to EMS vector. Compute $x_i^{k+1} = x_i^k + EMS_x^k$
4: Update all EMS vectors that are outdated because of the shift of x_i^k data point.
5: Set $k \leftarrow k + 1$.
6: **until** X_l^k satisfies the stopping criterion (energy of dataset is lower than a selected target energy value)
7: If l is not the final stage, enlarge the searching window (σ_s) set the dataset $X_{l+1}^k = EMS_l$.
8: **until** Last stage is performed. EMS_l is a result.

Two big problems emerge in EMS algorithm. If we process a large image (especially in the first stage), searching for the maximum in such a large amount of energies can be computationally very exhaustive (n steps in each EMS iteration). We propose a precise searching method that is used in the first stage in which the contribution should be the most significant. We use two buffers in hierarchy to store the energy values. In the first step, we compute an average and maximum energy of the dataset (in *n* steps). We adjust a threshold energy between the average and the maximum energy and we pick up the en-

ergies higher than this threshold (also in n steps) into the first buffer. After that, the values are sorted very fast because of their low number. We pick a more limited number of energy values to the smaller second buffer, where we will be searching for the maximum. Because of a small number of values, it can be done very fast. The ideal size of buffer is mostly between $\frac{1}{2}\sqrt{n}$ and $\sqrt{n}$. The energy value of the shifted pixel is removed and if an energy higher than the threshold emerges during recalculation of EMS vectors, it is added to the buffer.

If this buffer is empty, we repeat the selection process. We use the threshold values that select approx. 4% of data points. Therefore, the sorting algorithm has approx. complexity $0.04n \cdot \log(0.04n)$. It is performed after more than $\sqrt{n}$ iterations (if $\sqrt{n}$ is the buffer size). The searching for the maximum in the buffer takes $\frac{1}{2}\sqrt{n}$ on average instead of n.



(a) EMS $(40/-/-)$ (b) EMS $(40/-/-)$ (c) EMS $(40/-/-)$ (d) EMS $(40/-/-)$

(e) HEMS2 $(6/40/-)$ (f) HEMS2 $(6/40/-)$ (g) HEMS2 $(6/40/-)$ (h) HEMS2 $(6/40/-)$

(i) HEMS3 $(2.5/10/40)$ (j) HEMS3 $(3/12/40)$ (k) HEMS3 $(3/10/40)$ (l) HEMS3 $(3/12/40)$

Fig. 2. Segmentations produced by EMS, HEMS2 and HEMS3. The parameters σ_{s_1} / σ_{s_2} / σ_{s_3} are shown in the brackets, σ_r was set to 32 in all cases

Recalculation of the EMS vector is an another problem. In the next stages, we use a method that recalculates EMS vectors for all pixels possibly affected by the pixel shift. This area is specified by the range from $\min(oldX, newX) - \sigma_s$ to $\max(oldX, newX) + \sigma_s$ on the x-axis and similarly on the y-axis. Because of a low number of pixels in the next stages, it is quite fast even though the complexity is m^2, where m is the number of pixels in the neighbourhood of shifted pixel and $m \ll \pi\sigma_s^2$ (the size of neighbourhood in the first stage).

Even this approach is very slow in the first stage because $m = \pi\sigma_s^2$. We use an extra buffer in our approach. It stores the numbers of pixels (weighted by the kernel function) that were used to compute the EMS vector for each pixel. The recalculation is done by multiplying the EMS vector by the number of pixels contributing in computation of the EMS vector. We obtain the sum of energies of the pixel x_i. We subtract the contribution of the shifted pixel to the affected pixel from the sum and we divide it by a reduced number of pixels.

$$ES_{x_i}^{\text{temp}} = \frac{EMS_{x_i} \cdot \sum_{i-1}^{N} k(x, x_i, \sigma) - k(x_j, x_i, \sigma)}{\sum_{i-1, i\neq j}^{N} k(x, x_i, \sigma)} \quad (6)$$

In (6), $ES_{x_i}^{\text{temp}}$ is a temporary sum of energies of the pixel x_i with subtracted influence of the pixel x_j. This sum equals to the sum of energies that were used to compute the EMS vector of the pixel x_i. EMS_{x_i} is the energy of the pixel x_i and $k(x, x_i, \sigma)$ is a result of the kernel function between the pixel x_i and x with the bandwidth σ. The reverse step is processed in the area where the new position of shifted pixel is. This algorithm needs only $2\pi\sigma_s^2$ steps per pixel instead of σ_s^4.

4. EXPERIMENTS

This section presents the results of our implementation of EMS and HEMS. All algorithms use the accelerated searching for the maximal energy and adjusting the EMS vectors in the first stage. Therefore, EMS is fully accelerated too. We examine the speed of algorithms, quality of filtration and segmentation, although this can be quite subjective. We use the images from The Berkeley Segmentation Dataset [9]. As a testing computer, a laptop with 2,4 GHz Intel Core i5 processor was used. We used a single-core configuration.

In Fig. 2, the bandwidth was set experimentally to the values that led to the fastest computation. We use σ_s=40 for the final stage of all algorithms. Note that the higher values of σ_s should lead to a lower number of segments and it is the main parameter to control the result. The index behind the term σ_s denotes the stage number. 2-stage and 3-stage HEMS are referred to as HEMS2 or HEMS3, respectively.

All results are presented in Table 1. A segmentation looks visually good, although this is subjectively evaluated. The corresponding images are shown in Fig. 2. HEMS2 is up to $70\times$ faster than EMS and HEMS3 is even faster. The drawback is the increased number of segments and visually not so good shaped clusters. The hierarchical approaches create more segments and a slightly larger searching window should be used to achieve a smaller number of segments. HEMS was approx. $45 - 180\times$ faster for all the images we tested. Small, mostly circular, segments are created in the first stage and they are grouped in the later stages. The jagged lines are reduced. Significant lines are always segmented properly.

We measure the segmentation quality as a combination of number of segments and Mean Square Error (MSE). MSE describes the difference between the images, not the filtration quality itself. It is desirable to achieve a smaller number of segments with a small value of MSE. If the number of segments is high, the image is oversegmented and if the MSE is high, the image is significantly damaged. The values of MSE are similar in EMS and HEMS.

Table 1. Comparison of the numbers of segments (seg), speed (t[s]) and number of iterations (iter) depending on algorithm

		1st stage			2nd stage			3rd stage				Total time
		seg	t[s]	iter	seg	t[s]	iter	seg	t[s]	iter	MSE	t[s]
Stones image	EMS	55	6089.9	5.93	-	-	-	-	-	-	107	6090.5
	HEMS2	1151	62.63	6.40	78	26.36	8.90	-	-	-	113	89.69
	HEMS3	6440	19.15	6.82	593	6.72	8.41	96	7.4	9.02	113	34.16
Tree image	EMS	59	3836.9	5.78	-	-	-	-	-	-	208	3837.9
	HEMS2	1925	48.51	6.33	111	66.01	8.04	-	-	-	205	115.25
	HEMS3	4758	22.87	6.58	614	7.23	8.5	122	8.73	8.6	202	39.72
Mountains image	EMS	84	4256.9	5.97	-	-	-	-	-	-	123	4257.5
	HEMS2	1636	65.73	6.47	161	50.46	8.54	-	-	-	137	117.0
	HEMS3	5728	24.70	6.63	1059	6.40	8.37	194	25.96	8.67	132	58.17
Lena image	EMS	36	726.1	5.37	-	-	-	-	-	-	192	726.4
	HEMS2	623	25.1	6.40	59	16.1	8.33	-	-	-	238	41.5
	HEMS3	2318	8.5	6.79	293	2.9	8.46	71	4.2	8.67	270	16.1

We present the acceleration of point selection and the EMS vector recalculation in Section 3. The Lena image was segmented in 3147.7 seconds without acceleration (only recalculation in the affected area was used, $\sigma_s = 10$). Selective back recalculation using Eq. (6) reduced time to 76.9 seconds. Hierarchical optimization in search for the maximal energy using two buffers reduced the time to 58.0 seconds.

5. CONCLUSION

Our HEMS method showed that the hierarchical approach to EMS algorithm can shorten the computational time significantly and it does not necessarily lead to much worse segmentation capabilities. We think that a remarkable speedup is a good compensation for a very small reduction of the segmentation quality. Moreover, we presented fully precise hierarchical acceleration for searching the maximal energy applicable not only for HEMS, but also for original EMS. We also presented an effective way of EMS vector recalculation. In the future, we are going to focus on the minimization of overheads, especially, the ability to use full acceleration in the higher stages.

Acknowledgements. This work was partially supported by the grant SP 2011/163 of VŠB-TU of Ostrava, FEECS.

6. REFERENCES

[1] K. Fukunaga and L. Hostetler, "The estimation of the gradient of a density function, with applications in pattern recognition," *IEEE Transactions on Information Theory*, vol. 21, no. 1, pp. 32–40, January 1975.

[2] Yizong Cheng, "Mean shift, mode seeking, and clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, pp. 790–799, 1995.

[3] D. Comaniciu and P. Meer, "Mean shift analysis and applications," *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, vol. 2, pp. 1197 –1203 vol.2, 1999.

[4] D. Comaniciu and P. Meer, "Mean shift: a robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603–619, 2002.

[5] M.A. Carreira-Perpiñán, "Fast nonparametric clustering with Gaussian blurring mean-shift," in *Proceedings of the 23rd international conference on Machine learning*, New York, NY, USA, 2006, ICML '06, pp. 153–160, ACM.

[6] Qi Zhao, Zhi Yang, Hai Tao, and Wentai Liu, "Evolving mean shift with adaptive bandwidth: A fast and noise robust approach," in *ACCV (1)*, 2009, pp. 258–268.

[7] Pavan Vatturi and Weng-Keen Wong, "Category detection using hierarchical mean shift," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, New York, NY, USA, 2009, KDD '09, pp. 847–856, ACM.

[8] Milan Šurkala, Karel Mozdřeň, Radovan Fusek, and Eduard Sojka, "Hierarchical blurring mean-shift," in *Proceedings of the 13th international conference on Advanced concepts for intelligent vision systems*, Berlin, Heidelberg, 2011, ACIVS'11, pp. 228–238, Springer-Verlag.

[9] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "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*, July 2001, vol. 2, pp. 416–423.