A SUPERPIXEL SEGMENTATION ALGORITHM BASED ON DIFFERENTIAL EVOLUTION

Yue-Jiao Gong[1,2], *Yicong Zhou*[1,*], *Xinglin Zhang*[3]

[1]Department of Computer and Information Science, University of Macau, Macau, China
[2]School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China
[3]School of Computer Science and Engineering, South China University of Technology, Guangzhou, China
[*]Corresponding author: yicongzhou@umac.mo

ABSTRACT

This paper deals with the superpixel segmentation problem using a powerful global optimization technique: Differential Evolution. The algorithm mimics the process of nature evolution to realize efficient optimization, and it poses no restrictions on the form of objective functions. This way, we develop a novel and comprehensive objective function considering both local and global costs in the segmentation, including within-superpixel error, boundary gradient, a regularization term. The proposed method can produce superpixels in a computational time linear to the image size. Experimental results validate the competitive performance of our algorithm in terms of boundary adherence and segmentation capability.

Index Terms— Superpixel, segmentation, differential evolution, evolutionary computation, clustering

1. INTRODUCTION

Superpixel is a crucial image preprocessing technique that receives increasing attention in many multimedia and computer vision fields, such as salient object detection [1], image segmentation [2], and visual tracking [3]. The technique targets at oversegments an image into a number of perceptually meaningful regions (see Fig. 1). Compared with the rigid pixel representation of images, superpixel representation contains less redundancy and agrees well with human vision. As a preprocessing step, superpixel can provide a substantial speedup for the subsequent operation, since it greatly reduces the number of elements to be processed.

Since the concept of superpixel segmentation has been proposed [4], a lot of research efforts have been reported in this area [5, 6, 7, 8]. Each of these methods may favor a particular application, but generally the following three properties are desirable for a good superpixel algorithm. First, the boundaries of superpixels should stand on natural image boundaries such that each superpixel overlaps with a single natural object. Second, to enable the resulting superpixels be friendly to human vision or some subsequent feature extraction process, we prefer superpixels with relatively regular shapes and similar sizes. Third, since the algorithm plays as a preprocessing role, the computational complexity becomes a critical issue to be considered.

Fig. 1. Example DES results with different number of segmentations. From left to right, up to bottom: 100, 200, 300, 400, 500, and 600 superpixels, respectively.

To address these issues, this paper proposes a novel superpixel segmentation algorithm termed DES. Different from the existing work that optimizes the boundary adherence via minimizing local color variance [6, 7], DES optimizes this global property in a more direct way by means of global optimization. We design a *Boundary Gradient* term to evaluate the adherence of superpixel boundaries to the high gradient components in the image. Further, to enforcing generating homogenous superpixels, we introduce a *Regularizer* to measure the global variance of superpixel sizes. The two global terms, as well as the traditionally used local cost named *Within-Superpixel Error*, is considered in the objective function. The optimization is then accomplished by Differential Evolution (DE), a powerful stochastic global optimization algorithm mimicking the nature evolution process [9, 10]. Owing to the low complexity of DE, the proposed algorithm satisfies the computational restriction that it can produce promising superpixels with a linear computational complexity.

This work was partially supported by the Macau Science and Technology Development Fund under Grant FDCT/106/2013/A3, by the Research Committee at University of Macau under Grants MYRG2014-00003-FST and MRG017/ZYC/2014/FST, and by the National Natural Science Foundation of China (NSFC) under Grant 61502542 and 61502178.

In the experiments undertaken, DES is tested on the Berkeley segmentation benchmark [11], with standard performance indices evaluated. Compared with a number of state-of-the-art superpixel segmentation algorithms [5, 6, 7, 8, 12], DES exhibits equally well or better performance.

2. RELATED WORKS

Algorithms for superpixel segmentation can be generally categorized as either seeding-based or graph cutting-based. In this section, we review representative algorithms in these two categories. Here it is to be noticed that superpixel segmentation is a different area from traditional image segmentation. The area focuses on changing the representation of images for the subsequent image processing tasks, and it puts tighter limits on the time complexity.

2.1. Seeding-based superpixel segmentation

Algorithms falling in this category have a common method that they identify a number of seeds (or centers) and grow superpixels from the seeds. A very popular algorithm to realize seeding is the K-means clustering, which are adopted by the Simple Linear Iterative Clustering (SLIC) [7] and Linear Spectral Clustering (LSC) [13]. The K-means-based algorithms produce superpixels that minimize local color variance. Compared with SLIC, LSC makes an improvement that it uses a kernal function to realize nomarlized cuts effect of the segmentation.

The Quick Shift (QS) [14] algorithm introduces a quick medoid-shift algorithm for shifting the seeds towards high density areas in the image. However, this algorithm cannot offer explicit control over the number and size of superpixels. Another popular work in this category is the Turbopixel (TP) method [6]. By gradually dilates a set of regularly distributed seeds using geometric flows, TP generates highly regular superpixels, but, at the same time, the adherence to boundaries is relatively low. Besides, the waterpixel (WP) [15] algorithm first selects a number of markers and then perform watershed transformation based on the makers and the image gradient.

2.2. Graph cutting-based superpixel segmentation

Many superpixel segmentation algorithms are based on gradually adding cuts to the graph representation of images. The most classical algorithm in this category is the Normalized Cuts (NCuts) [4], which is also considered as a pioneer of superpixel oversegmentation. However, the high computational overhead limits the wide applicability of NCuts. In comparison, Felzenszwalb and Huttenlocher [5] propose a highly efficient graph-based oversegmentation algorithm (GS) by performing agglomerative clustering. GS arrives at a good boundary adherence performance, but the resulting superpixels have very irregular shapes and variable sizes.

Considering more state-of-the-art, the Entropy Rate Superpixels (ERS) [16] designs a new objective, namely, the entropy rate of the graph, and optimizes it together with a balancing term. However, as reported in [16], the algorithm costs about 2.5s to segment a 481×321 image so the time overhead is relatively high. The Superpixels Extracted via Energy-Driven Sampling (SEEDS) [17] cuts images by hill-climbing. It possesses very promising computational efficiency in that only pixels near the superpixel boundaries are evaluated in the optimization. Nevertheless, SEEDS poses strict constraint on the number of segmentations, and it also suffers from shape irregularity of superpixels. Besides, Veksler *et al*. [8] formulate the superpixel segmentation problem in an energy optimization framework and develop two algorithms (EOpt0 and EOpt1) to generate compact and constant-intensity superpixels, respectively. The two algorithms provide good shape regularity at the cost of large time overhead.

3. THE PROPOSED ALGORITHM

Compared with the above reviewed algorithms, the proposed DES algorithm has the following distinctions: 1) compared with the algorithms that cluster superpixels by local color variance [7, 13, 6], DES directly embeds the global segmentation properties in the objective function; 2) compared with the algorithms based on global optimization [4, 8], the computational cost of DES is significantly reduced by using a highly efficient optimizer; 3) we also pay attention to the homogeneity of superpixels and avoid producing superpixels with various sizes [5, 14, 16, 17]. Next, the framework and implementations of DES are detailed.

3.1. Framework overview

Fig. 2 depicts the segmentation pipeline of DES. Given an input image I, we first blur the image by a mean filter to obtain an image I'. In this way, each pixel $I'(x, y)$ strores the mean pixel values in the local neighborhood of $I(x, y)$. The algorithm iteratively seeds K points in $I'(x, y)$, where K is the number of required superpixels. All the pixels in the input image I are assigned to their nearest neighboring seeds in I'. After the assignment, we evaluate the segmentation by three measures: *Within-Superpixel Error*, *Boundary Gradient*, and a *Regularization* term, which are then aggregated into a comprehensive objective function. The aggregated objective value of current assignment is propagated into the DE algorithm consisting of mutation, crossover, and selection to evolve even better seeds. The seed evolution, label assignment, and objective evaluation steps are performed iteratively for a number of generations. Finally, a postprocessing step is conducted on the label matrix, which reassigns disjoint pixels to their neighboring superpixels to enforce connectivity.

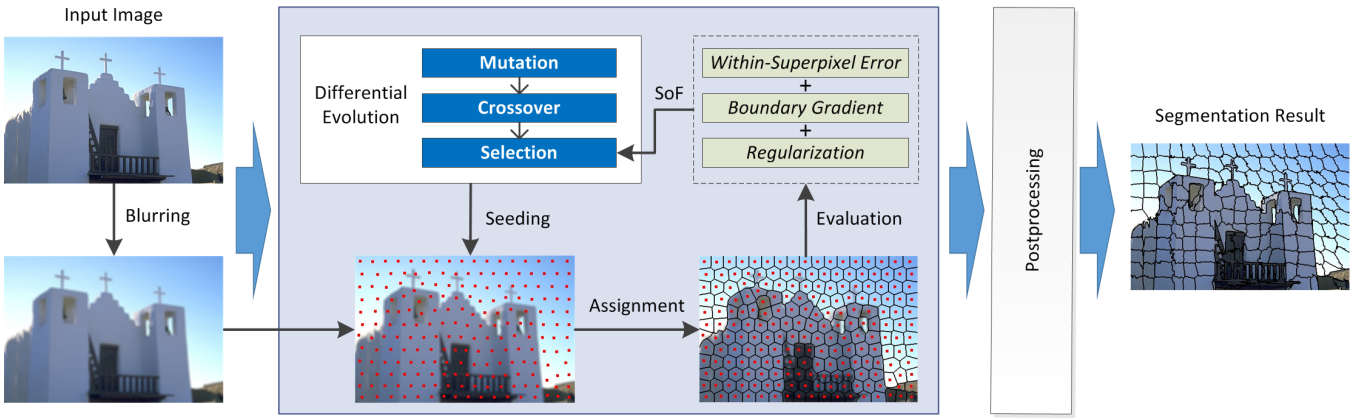


Fig. 2. The DES segmentation pipeline.

3.2. Objective function

Given a set of K seeds $\boldsymbol{s} = [\boldsymbol{s}_1, \boldsymbol{s}_2, \cdots, \boldsymbol{s}_K]$, each is a feature vector representing the superpixel center, the objective function $F(\boldsymbol{s})$ to be optimized is defined as

$$F(\boldsymbol{s}) = f_{wse}(\boldsymbol{s}) + \lambda(-f_{bg}(\boldsymbol{s})) + \gamma f_r(\boldsymbol{s}) \tag{1}$$

where $f_{wse}(\boldsymbol{s})$, $f_{bg}(\boldsymbol{s})$, and $f_r(\boldsymbol{s})$ stand for the within-superpixel error, boundary gradient, and regularization terms, respectively. Since we want to minimize $f_{wse}(\boldsymbol{s})$ and $f_r(\boldsymbol{s})$ while maximizing $f_{bg}(\boldsymbol{s})$, in the aggregation, $f_{bg}(\boldsymbol{s})$ is prefixed with a negative sign. Parameters λ and γ control the relative influence of the three terms, which are fixed to constant values. In the experiments of this paper, we empirically use $[\lambda = 0.2, \gamma = 0.8]$.

Within-Superpixel Error. This term is defined as the summarization of errors assigning each pixel to its nearest seed in $\boldsymbol{s}$. We treat the color image in the CIELAB space and represent each pixel by a five-dimentional feature vector (l, α, β, x, y), where l, α, and β stand for light and color components, and x and y denote the pixel coordinates. Without loss of generality, the five components are linearly normalized into range $[0, 1]$. Then, we multiply the l, α, and β components by a coefficient $\eta = 30$ to enhance their relative importance in the feature vector. The L_2 norm is used to compute the distance between two feature vectors. Namely, given two points $\boldsymbol{a}$ and $\boldsymbol{b}$, the distance is

$$\begin{aligned} ||\boldsymbol{a}, \boldsymbol{b}||_2 =& ((l_{\boldsymbol{a}} - l_{\boldsymbol{b}})^2 + (\alpha_{\boldsymbol{a}} - \alpha_{\boldsymbol{b}})^2 + (\beta_{\boldsymbol{a}} - \beta_{\boldsymbol{b}})^2 \\ &+ (x_{\boldsymbol{a}} - x_{\boldsymbol{b}})^2 + (y_{\boldsymbol{a}} - y_{\boldsymbol{b}})^2)^{1/2} \end{aligned} \tag{2}$$

Following [7], in the assignment step, we associate the seeds with a limited window, i.e., only pixels located in the small window can be assigned to the seed. This restriction significantly reduces the computational overhead for distance calculations and also facilitates producing compact superpixels. In this work, the window length is set to $3/2S$, where $S = \sqrt{N/K}$ is the grid step of uniformly partitioning the image with N pixels into K grids. It is also noted here that, in the nearest seed assignment, each pixel uses the feature vector in the input image I, while each seed uses the local mean vector in the blurred image I'. Finally, after the assignment, the within-superpixel error is calculated as

$$f_{wse}(\boldsymbol{s}) = \sum_{(x,y)} ||\boldsymbol{p}_{x,y} \in I, \boldsymbol{s}_k \in I'||_2 \tag{3}$$

where $\boldsymbol{p}_{x,y}$ is the feature vector of pixel (x, y) and $\boldsymbol{s}_k$ is that of its nearest seed.

Boundary Gradient. As introduced above, improving the boundary adherence is a primary goal of superpixel segmentation. However, we could not know the natural image boundaries in prior, so the boundary gradient is used instead, since it provides a signal of the boundary strength. Before starting the algorithm, a gradient map of the input image is calculated by performing the Roberts mask and is denoted as ΔI. We evaluate the boundary gradient based on ΔI and the current superpixel assignment as

$$f_{bg}(\boldsymbol{s}) = \frac{1}{|B(\boldsymbol{s})|\Delta_{\max}} \sum_{(x,y)\in B(\boldsymbol{s})} \Delta I(x, y) \tag{4}$$

where $B(\boldsymbol{s})$ is the obtained set of boundary pixels by assigning pixels to the $\boldsymbol{s}$, $\Delta_{\max}$ is the maximum gradient value for the purpose of normalization.

Regularization. Since we partition the image with N pixels into K superpixels, the expected number of pixels assigned to each superpixel is $c_m = N/K$. The regularizer punishes segmenting superpixels with sizes far away from c_m, which is defined as

$$f_r(\boldsymbol{s}) = \frac{1}{NK} ||c(\boldsymbol{s}) - c_m||_2 \tag{5}$$

where $c(\boldsymbol{s}) = [c(\boldsymbol{s}_1), c(\boldsymbol{s}_2), \cdots, c(\boldsymbol{s}_K)]$ denotes the number of pixels assigned to each seed, N and K are divided for the purpose of normalization.

3.3. Seeding via Differential Evolution

Our goal is to find a seed set $\boldsymbol{s}^*$ to minimize the objective function:

$$\boldsymbol{s}^* = \arg\min_{\boldsymbol{s}\in\boldsymbol{S}} F(\boldsymbol{s}) \tag{6}$$

where $\boldsymbol{S}$ is the set of all feasible combinations of seeds. The DE algorithm is introduced to efficiently finish this optimization task. DE is a powerful stochastic global optimization algorithm specialized in solving nondeterministic polynomial-time hard problems. Mimicking the process of nature evolution, the algorithm maintains a population of individuals to represent a set of solutions and evolves the individuals (solutions) via mutation, crossover, and selection at each generation. The objective function to be optimized acts as the role of environment, which evaluates the fitness of individuals. Mutation and crossover breed new and possibly more competitive offspring solutions. After evaluating the fitness, a rule named "Survival of Fitness (SoF)" is used in the selection to update the population. Finally, after a number of generations, the individual with the best fitness is outputted as the final solution.

In the proposed DES, the population maintains 5 individuals. Each individual X_i is encoded by a $2K$-dimensional seed index vector that represents the coordinates of K seeds, by which we retrieve K feature vectors from the blurred color image. The search range of each seed is restricted in an $S \times S$ grid in the image. As $S = \sqrt{N/K}$, the K seeds will be sampled from the K uniformly partitioned regions, respectively. In the initialization, the individuals are randomly generated within the search space, with fitness evaluated according to Eq. (1). Then, the following three operators are performed to update the individuals.

Mutation. The mutation is performed on each individual X_i to create a mutant vector V_i:

$$V_i = \lfloor X_{best} + 0.5 \cdot (X_{r_1} - X_{r_2}) \rfloor \tag{7}$$

where X_{best} is the best fitted individual, r_1 and r_2 are two randomly generated indices, and the three individuals are kept distinct.

Crossover. In crossover, each individual X_i randomly exchanges some components from the mutant vector V_i with 90% probability in order to breed a trial vector U_i:

$$U_{i,j} = \begin{cases} V_{i,j}, & \text{if } rand(0,1) < 0.9 \\ X_{i,j}, & \text{otherwise} \end{cases} \tag{8}$$

where $rand(0,1)$ a uniform random number generator, and $j = 1, 2, ..., 2K$.

Selection. The fitness of the new trial vector is evaluated according to Eq.(1). The trial vector will replace the individual once it has an equal or better objective value.

$$X_i = \begin{cases} U_i, & \text{if } F(U_i) \leq F(X_i) \\ X_i, & \text{otherwise} \end{cases} \tag{9}$$

In this way, individuals (solutions) that are more fitted to the environment will be preserved in the population, a.k.a., SoF.

The above operators iterates for G generations in order to output a satisfactory solution. The setting of G balances the segmentation performance and computational overhead of DES. We found that $G = 10$ is enough for producing promising superpixel segmentation, and meanwhile the required time is relatively short.

3.4. Postprocessing

In the above superpixel segmentation procedures, we do not explicitly enforce the connectivity of superpixels. Therefore, there may possibly exist some "orphaned" pixels that are assigned with labels different from the sounding pixels. Following the other algorithms [5, 7], we perform a postprocessing step named "enforce connectivity" on the assigned label matrix to correct the lables of these isolated pixels and obtain the final segmentation results.

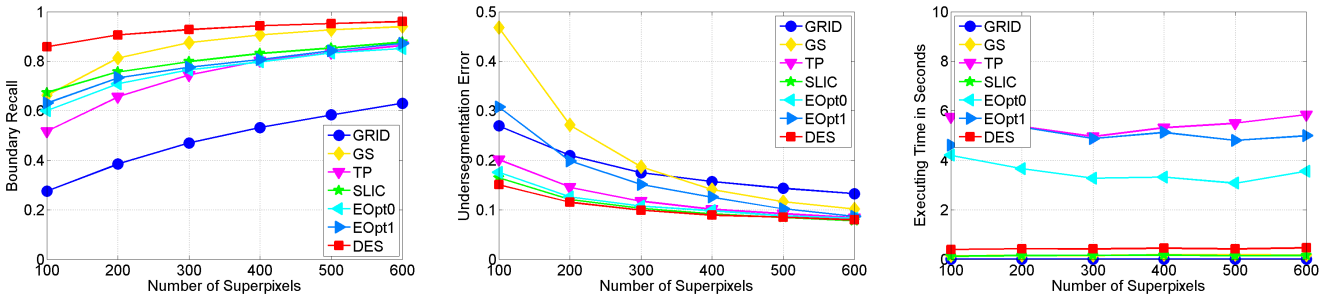
3.5. Complexity

Given an image of size N, the complexity of blurring the image and calculating the gradient map is $O(N)$. By restricting the covering range of seeds in a limited window, the complexity of label assignment is reduced to $O(N)$, as reported in [7]. The cost in calculating the objective value (i.e., the fitness of individuals in DE) is obviously $O(N)$. The DE operation has a linear complexity to the number of seeds and it is thus $O(K)$ complex. The seeding process iterates for a constant number of times, which is independent with N and K. This way, DES is $O(N + K)$ complex. Since $K \ll N$, the term $O(K)$ can be omitted. To conclude, the proposed DES algorithm has a linear complexity $O(N)$ to the image size.

4. EXPERIMENTS AND COMPARISONS

The proposed DES algorithm is compared with five state-of-the-art peer algorithms including GS [5], TP [6], SLIC [7], EOpt0 and EOpt1 [8], and a baseline algorithm GRID [12] that uniformly segments the image into K grids. These algorithms are implemented based on their public codes. We test the algorithms on the Berkeley segmentation benchmark, which consists of 300 images with ground truth labels [11]. All the algorithms are coded by C++, and executed on the same PC platform. Standard performance metrics are used to evaluate different algorithms, including Boundary Recall (BR), Undersegmentation Error (UE) [12], and the required processing time per image. BR measures the proportion of grand truth edges that fall within two pixels of the superpixel boundaries. UE matures the area of superpixels that slops over the ground truth segmentation borders.

The quantitative results obtained by the algorithms for segmenting 100-600 superpixels are compared in Fig. 3. To



(a) (b) (c)

Fig. 3. Performance comparison curves of different superpixel algorithms. (a) Boundary Recall. (b) Undersegmentation Error. (c) Executing Time per Image.

Table 1. Performance metrics obtained by different algorithms when $K = 300$

	GRID	GS	TP	SLIC	EOpt0	EOpt1	DES
Boundary Recall	0.4702	0.8752	0.7442	0.7988	0.7652	0.7754	**0.9278**
Undersegmentation Error	0.1749	0.1868	0.1175	0.1030	0.1075	0.1513	**0.0993**
Executing Time per Image	**3.60E-4**	0.1534	4.9589	0.1580	3.2751	4.8779	0.4214

make the comparison more clear, we list the numeric results with $K = 300$ specifically in Table 1. It can be observed that the proposed DES algorithm performs the best in terms of BR and UE. Particularly, when a small number of superpixels is required, the improvement brought by DES to the other algorithms becomes more significant. GS obtains promising BR values, but its performance on UE is relatively poor. Besides, SLIC and EOpt0 provides perceptually satisfactory solutions among the compared algorithms. On the other hand, considering the computational overhead, reported in Fig. 3(d), GRID is the fastest as we can expect. Then, GS, SLIC, and the proposed DES are among the fastest superpixel segmentation algorithm category, whereas TP, EOpt0, and EOpt1 are much more slower. Fig. 1 depicts visual results of the superpixels generated by the proposed DES, with an increasing value of K. Note that the algorithm not only provides good boundary adherence but also its output superpixels have relatively regular shapes and similar sizes.

The superpixel segmentation technique is commonly used as a preprocessing procedure in image segmentation and the related fields. Because of this, it is desired that, by using superpixel segmentation, the subsequent algorithm obtains not only substantial speedup but also promising segmentation performance. This can be investigated by assuming that an ideal classification algorithm is performed after the superpixel segmentation. Each superpixel is hence assigned with a label of ground truth segment within which the superpixel overlaps the most. Fig. 4 shows five examples of the achievable segmentation results based on DES, EOpt0, SLIC, and TP, respectively. We can see that, DES exhibits better preprocessing performance than the others, and it is more suitable to be applied in the image segmentation and related fields.

5. CONCLUSION

We develop a novel superpixel segmentation algorithm, DES, which can produce superpixels with good boundary adherence efficiently. The promising performance of DES owes much to the comprehensive objective function that considers both local and global information in the segmentation. A regularizer is also aggregated in the objective, so as to generate homogenous superpixels with similar sizes and regular shapes. To optimize such a complex model, we use a state-of-the-art global DE algorithm. DES is efficient, whose computational complexity is $O(N)$. Qualitative and quantitative experimental results show that the proposed algorithm works well on the Berkeley segmentation benchmark in terms of boundary recall, undersegmentation error, and time overhead. In the future, we will further consider the noisy and cluttered environments and try to develop more robust DES to handle these situations.

6. REFERENCES

[1] R. Liu, J. Cao, Z. Lin, and S. Shan, "Adaptive partial differential equation learning for visual saliency detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2014, pp. 3866–3873.

[2] L. Zhang, Y. Gao, Y. Xia, K. Lu, J. Shen, and R. Ji, "Representative discovery of structure cues for weakly-supervised image segmentation," *IEEE Trans. Multimedia*, vol. 16, no. 2, pp. 470–479, 2014.

[3] Y. Wang and Q. Zhao, "Superpixel tracking via graph-based semi-supervised SVM and supervised saliency detection," in *IEEE Int. Conf. Multimedia & Expo*, 2015, pp. 1–6.

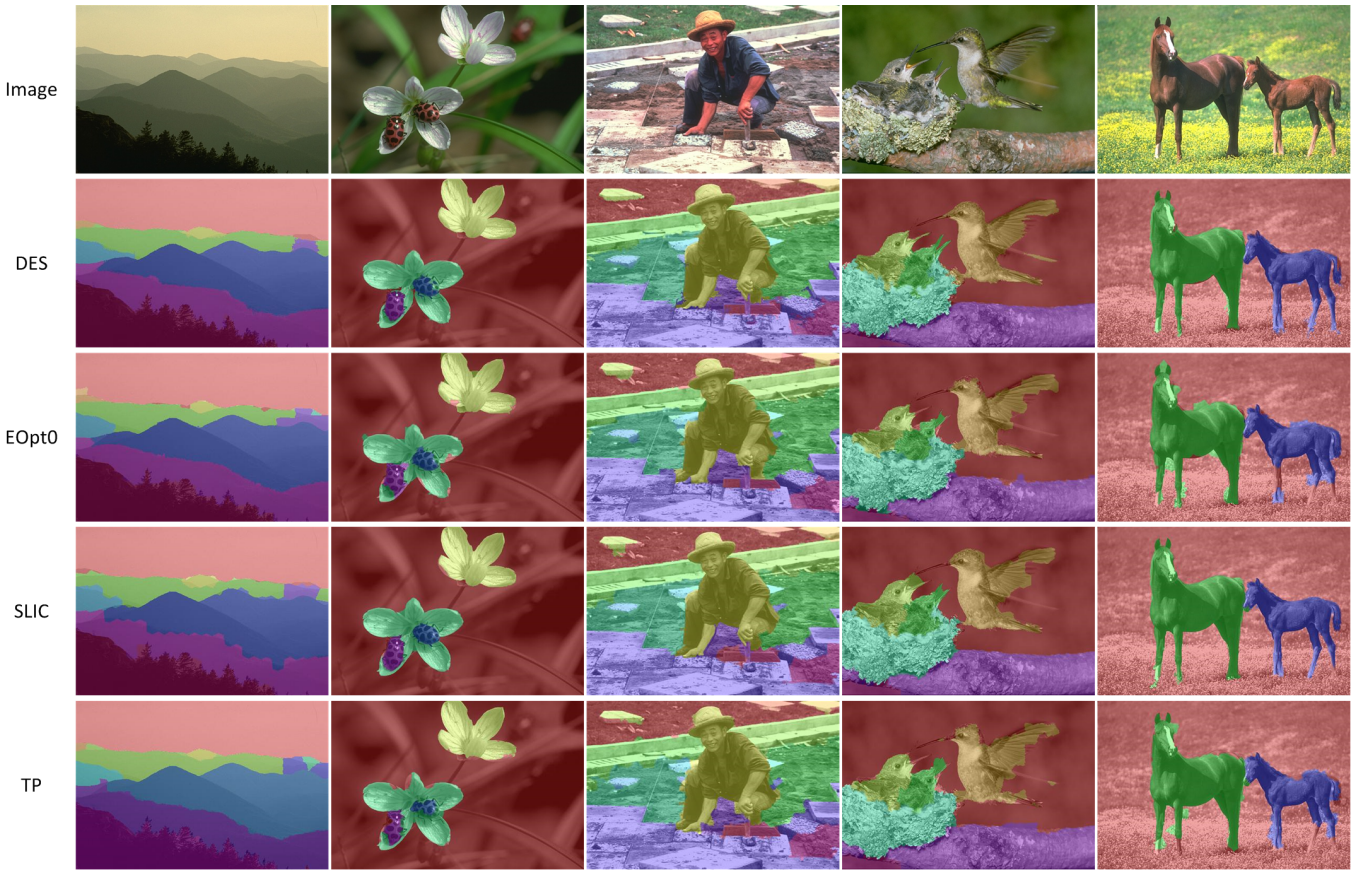


Fig. 4. Achievable segmentation results based on DES, EOpt0, SLIC, and TP.

[4] X. Ren and J. Malik, "Learning a classification model for segmentation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2003, pp. 10–17.

[5] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 59, no. 2, pp. 167–181, 2004.

[6] A. Levinshtein, A. Stere, K. N. Kutulakos, D. J. Fleet, S. J. Dickinson, and K. Siddiqi, "Turbopixels: Fast superpixels using geometric flows," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 12, pp. 2290–2297, 2009.

[7] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 11, pp. 2274–2282, 2012.

[8] O. Veksler, Y. Boykov, and P. Mehrani, "Superpixels and supervoxels in an energy optimization framework," in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 211–224.

[9] S. Das and P. N. Suganthan, "Differential evolution: a survey of the state-of-the-art," *IEEE Trans. Evolut. Comput.*, vol. 15, no. 1, pp. 4–31, 2011.

[10] A. Khan, M. A. Jaffar, and L. Shao, "A modified adaptive differential evolution algorithm for color image segmentation," *Knowl. Inf. Syst.*, vol. 43, no. 3, pp. 583–597, 2015.

[11] 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. Eur. Conf. Comput. Vis.*, vol. 2, July 2001, pp. 416–423.

[12] P. Neubert and P. Protzel, "Superpixel benchmark and comparison," in *Proc. Forum Bildverarbeitung*, 2012, pp. 1–12.

[13] Z. Li and J. Chen, "Superpixel segmentation using linear spectral clustering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2015.

[14] A. Vedaldi and S. Soatto, "Quick shift and kernel methods for mode seeking," in *Proc. Eur. Conf. Comput. Vis.*, 2008, pp. 705–718.

[15] V. Machairas, M. Faessel, D. Cárdenas-Pena, T. Chabardes, T. Walter, and E. Decencière, "Waterpixels," *IEEE Trans. Image Process.*, vol. 24, no. 11, pp. 3707–3716, 2015.

[16] M.-Y. Liu, O. Tuzel, S. Ramalingam, and R. Chellappa, "Entropy rate superpixel segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2011, pp. 2097–2104.

[17] M. Van den Bergh, X. Boix, G. Roig, B. de Capitani, and L. Van Gool, "SEEDS: Superpixels extracted via energy-driven sampling," in *Proc. Eur. Conf. Comput. Vis.*, 2012, pp. 13–26.