Hybrid Superpixel Segmentation

Yuan Liu, Shangqi Lai, Tianyi Du, Yizhou Yu
Department of Computer Science, The University of Hong Kong, Hong Kong
Email: {yliu3, aquas, obiwandu}@connect.hku.hk,
yzyu@cs.hku.hk

***Abstract*—Superpixel over-segment image into meaningful clusters so that pixels in each cluster belong to one object. Many state-of-art superpixel algorithms have to make trade-offs between different concerns. As a result, algorithms that can produce good result in some situations fail in another. In order to take advantage of different algorithms and at the same time avoid their limitation, we propose a new fusion approach based on an efficient lazy greedy optimization. It incorporates two different superpixel algorithms as its ancestors and produces a hybrid result. The result is then refined based on a novel energy function that consists of two terms. The region term uses histogram diffusion distance and enforces intra-region similarity from an overall perspective; the boundary term models inter-region dissimilarity from a local perspective. In experiments, the result of proposed algorithm matches the best superpixel algorithm and shows outstanding performance over its ancestor algorithms in all the standard evaluation metrics.**

I. Introduction

Superpixel algorithm clusters pixels into over-segmented regions, which focuses on intra-region similarity. The goal of superpixel is to generate over-segmented regions that adhere well to image boundaries and have good compactness. Superpixel is a successful approach to compress the information. It has become a basis for many computer vision tasks, such as body model estimation [1] [2], segmentation [3] and image labeling [4].

There is a large amount of approaches about generating superpixel. Each method makes its own trade-offs. As a result, different algorithms have different natural advantages and inherent limitation. Broadly superpixel algorithms can be categorized into three families, seed-based, graph-based and grid-based. We will evaluate the advantage and disadvantage of each family.

Seed-based family. The essential idea of seed-based superpixel is to grow superpixel from some pre-assigned centroid. Since the initial centroid location is pre-defined, the structure relationship between neighboring superpixels is naturally good, few scatted tiny superpixel. Turbopixel [5] grows region using geometric flow techniques. SLIC [6], one of he most famous superpixel algorithms, adopts a k-means clustering approach.

Graph-based family. These algorithms take pixels as vertex and the similarity between pixels as the edge weight. The graph is then partitioned into several sub-graph to form final superpixel. Efficient graph based method (FH) [7] merges vertices in a dynamic programming manner such that each region is minimum spanning tree. Entropy rate superpixel (ERS) [8] gradually adds edge between two vertices to maximize the energy function that incorporates the entropy rate of the sub-graph and uniform size balancing term.

Grid-based family. Michael Van den et al. [9] try another way. They first segment an image in to uniformly distributed square box and then the refine it by exchanging pixels on the boundary between two superpixel region. The refinement process is based on an energy function that encourages homogeneous color distribution and smooth superpixel boundary.

Fig. 1 shows superpixel results of representative algorithms from three families. Graph-based algorithm depends on the feature of local information. As a result, even if inconspicuous object boundary can be segmented. However, graph-based algorithm always produces irregular structure relationship between superpixels. See the FH result in Fig. 1.

Seed-based and grid-based algorithms start from an initial state and continue refining the result.The advantage is that initial state can form good structure relationship constraints but some edges can not be segmented. In Fig. 1, there is no way to segment the balcony pillars if there is only one seed or grid placed in that image region.

Intuitively, if we can combine the advantages of graph-based and seed-based method, we will get superpixels that have good boundary adherence and also structure relationship. In this paper, we proposed a new superpixel algorithm by combining two former superpixel algorithm with a merging process. The proposed hybrid superpixel successfully segments balcony pillars in Fig. 1.

The paper is organized as follows. The preliminary knowledge of merging process is introduced in Section II. The algorithm and corresponding optimization is in Section III. Section IV presents the performance comparison in benchmark and practical use between proposed algorithm and the state-of-art. Section V is the conclusion.

II. Preliminary

A. Regions Adjacency Graph

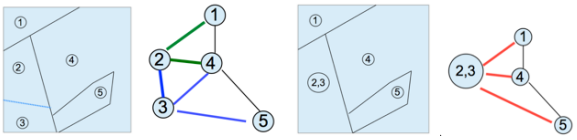
The regions adjacency graph (RAG) uses region as graph vertex. The edge represents the neighbor relationship of regions [10]. Formally, let $G = (V, E)$ be an undirected graph, the vertices $v \in V$ are the regions $R \in RAG$ in the image. In the following part, RAG vertices and image regions are used interchangeably.

The edges $e(v_i, v_j) \in E$ represent pairs of neighboring region. It shows the location relationship between two regions. Removing one edge in RAG represents merging two vertices (regions) connected by this edge.

We adopt RAG to represent our image as it has some interesting characteristic about the neighboring relationship changes during the merging process. As shown in Fig. 2, after

(a) SLIC (b) FH (c) SEEDS (d) Hybrid

Fig. 1: Results of representative algorithm from three families: (a) SLIC [6] from seeds-based familiy;(b) FH [7] from graph-based family; (c) SEEDS [9] from graph-based family; (d) Hybrid superpixel proposed in this paper.



(a) before merging vertex 2 and 3 (b) after merging vertex 2 and 3

Fig. 2: Simplification of RAG after merging two vertices

merging vertex 2 and 3, the original five edges are reduced to three edges but the neighborhood relationship remains consistent. To set theory models, this process using following formula:

$$D = (A \cup B) \setminus C \tag{1}$$

D denotes the edge set after merging. A and B are the edge sets of the first vertex and the second vertex. C represents the edges which connect the first vertex and second vertex.

B. Energy Function

The proposed algorithm merges the image regions by following the Eq. (1), therefore, we determine the merging order via the energy between vertices in RAG. The energy is distributed in the edges of RAG. Each edge $e(v_i, v_j) \in E$ has a weight $W(e)$, which is the dissimilarity between two neighboring vertices v_i and v_j. We model weight $W(e)$ from two levels. The first term is region term $R(e)$, which is based on the color distribution of each region. From an overall perspective, it describes the dissimilarity between regions. The second term is boundary term $B(e)$. It focuses on local gradient change along the boundary between regions and models the dissimilarity from a local perspective. The weight value is the product of two terms:

$$W(e) = R(e)B(e) \tag{2}$$

And the total energy of one RAG is:

$$H(E) = \sum_{e \in E} W(e) \tag{3}$$

Our goal is to maximize the energy function of RAG $G = (V, E)$ by merging a subset of vertices, while the resulting RAG $G' = (V', E')$ remains no more than N vertices, Eq. (4) is the mathematical expression of two constraints:

$$\begin{aligned} &H(E') \ is \ maximized \\ &|V'| \leq N \end{aligned} \tag{4}$$

Where $|V'|$ denotes the total number of vertices in the RAG. This constraint enforces merging process to produce exactly N superpixel.

C. Region Term

Region Term is used to describe the color dissimilarity of each vertex in RAG. Comparing with other Distance methods, such as L_2 norm, use bin-to-bin manner are very sensitive to histogram distortion, Histogram is an effective method to capture information of local region. Thus, the similarity between two regions is defined by histogram diffusion distance [11]. It models the difference between two histogram as a diffusion process. The histogram diffusion distance allows cross-bin comparison and is robust to noise and distortion. To the best of our knowledge, histogram diffusion distance has never been used as metric in superpixel.

We exploit histogram diffusion distance $K(h_1, h_2)$ to define the similarity between two second-order superpixel regions:

$$K(h_1, h_2) = \int_0^{\bar{t}} |d(x,t)| \mathrm{d}t \tag{5}$$

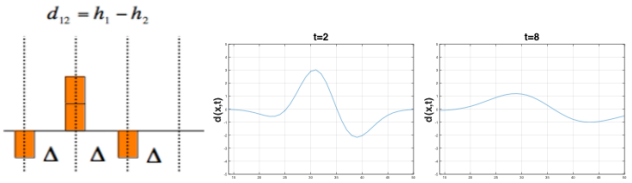
$$d(x,t) = [h_1(x) - h_2(x)] * \phi(x,t) \tag{6}$$

$$\phi(x,t) = \frac{1}{t(2\pi)^{\frac{1}{2}}} \exp(-\frac{x^2}{2t^2}) \tag{7}$$

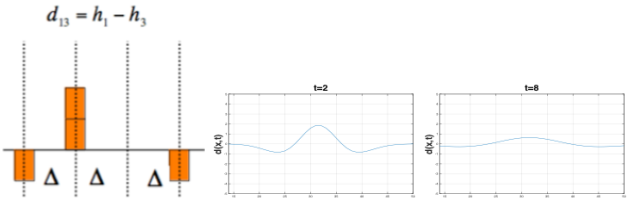
Where x is the RGB value of pixel, $\phi(x,t)$ is the Gaussian filter, $d(x,t)$ is the convolution of Gaussian filter and histogram difference, which models the diffusion process as time t increase. The histogram diffusion distance is the integration of convolution result, where $\bar{t}$ is the upper bound of integration.

Considering two one-Dimension histogram h_1, h_2 and their bin-to-bin distance d. As we can see in the Fig. 3, bin-to-bin distance does not take histogram distribution into consideration. However, during the diffusion process, the histogram distribution also plays essential roles. Fig. 3 shows the diffusion process $d(x,t)$ when the time t equals 2 and 8. We can find the left bin of h_3 is decrease to zero faster than right

bin of h_3 since the left bin of h_3 is only Δ away from h_1. Consequently, $|d_{12}(x,t)| < |d_{13}(x,t)|$ and it is reasonable that $K(h_1,h_2) < K(h_1,h_3)$.



(a) Bin-to-bin distance and diffusion process of d_{12}



(b) Bin-to-bin distance and diffusion process of d_{13}

Fig. 3: Different Bin-to-bin distance and its diffusion process

D. Boundary Term

Histogram is an effective method to capture the information in the local region of image. However, histogram suffers from the quantization effect [11].

We derive a new boundary term to counteract the quantization effect. Firstly in pixel level, we convert original image into an undirected graph $G_p = (V_p, E_p)$. Vertices represent the pixels in the image and each edge has a weight, which is the dissimilarity between neighboring vertex v_{p_i} and v_{p_j} (see Eq. (8)).

$$\begin{aligned} w(e_p) =& w(v_{p_i}, v_{p_j}) \\ =& \sqrt{(r^2_{v_{p_i}} - r^2_{v_{p_j}}) + (g^2_{v_{p_i}} - g^2_{v_{p_j}}) \\ & +(b^2_{v_{p_i}} - b^2_{v_{p_j}})} \end{aligned} \tag{8}$$

The RAG is partition of V_p into components such that each RAG vertex R is one second-order superpixel region and represents a sub-graph $G'_p = (V'_p, E'_p)$. We define the intra-difference of region R to be the average weight of the sub-graph:

$$Intra(R) = \frac{\sum_{e'_p \in (E'_p)} w(e'_p)}{|E'_p|} \tag{9}$$

where $|E'_p|$ denotes the total number of edge in the sub-graph $G'_p = (V'_p, E'_p)$. We define the inter-difference between two second-order superpixel region R_1, R_2 to be the average weight along the boundary between R_1, R_2:

$$Inter(R_1, R_2) = \frac{\sum_{v_{p_i} \in R_1, v_{p_j} \in R_2} w(v_{p_i}, v_{p_j})}{|B(R_1, R_2)|} \tag{10}$$

where $B(R_1, R_2)$ represents boundary between R_1, R_2, that is the set of edge $w(v_{p_i}, v_{p_j})$ crossing two superpixel region R_1, R_2. Second-order superpixel regions segmented by sharp

(a) Original (b) First-order (c) Second-order (d) Final result

Fig. 4: The procedure of SLIC-FH hybrid algorithm: Process (a) original image using SLIC to get (b) first-order superpixel; Use FH to further split each first-order superpixel region and result in (c) second-order superpixel; Merge second-order superpixel to pre-defined number to get (d) final result.

change of color usually have large inter-difference. Then the final boundary difference between R_1, R_2 is:

$$dif(R_1, R_2) = \frac{2Inter(R_1, R_2)}{Intra(R_1) + Intra(R_2)} \tag{11}$$

III. Hybrid Superpixel Algorithm

Hybrid superpixel algorithm is a fusion framework that incorporates any two different superpixel algorithms as its ancestor and produces a hybrid result. The most interesting part of hybrid superpixel is that it can keep the strength but discard the inherent limitation of its ancestor. Fig. 4 shows an overview of SLIC-FH Hybrid superpixel algorithm. The final result is sensitive to image boundary and maintains good structure relationship.

A. Algorithm

The image is first processed by SLIC, which is seed-based method and produce good neighboring relationship. We call the result first-order superpixel. Then we use the first-order superpixel as mask and perform FH, graph-based method, in each region of first order superpixel to get better boundary adherence. The result is called second-order superpixel.

The second-order superpixel is converted into RAG. Each graph vertex represents one second-order superpixel region. Then vertices are merged until number of vertices reaches user-defined number. Merging process is based on a novel energy function that consists of two terms. The region term enforces intra-region similarity from overall perspective while boundary term models inter-region dissimilarity from local perspective.

Merging process is fast because number of second-order superpixel has the same order of magnitude of user-specified final superpixel. The final result is a RAG, in which each vertex representing one superpixel region.

Algorithm 1 Hybrid Superpixel Segmentation

Input: An image I in RGB space.
Output: A RAG G represents the different regions of input image.
Run SLIC superpixel algorithm to get I'
Run FH superpixel algorithm to get I''
Get the RAG $G_{I''}$ of I''
Run Merging process on $G_{I''}$ to get G

B. Merging Process

The merging process changes the structure and edge weight of RAG. It is computationally expensive to update all edges weight after merging two vertices. However, a key observation is that the edge weight can never decrease after merging two vertices.

Proposition 1: Let h_1, h_2 and h_3 denote the histogram of three neighboring region R_1, R_2 and R_3. After merging h_1 and h_2, the histogram diffusion distance will increase as the following equation:

$$K(h_1, h_3) \leq K(h_1 + h_2, h_3) \tag{12}$$

Proof: The 1-norm of a histogram is greater than or equal to zero and then:

$\because$ *Merging process increases the dissimilarity between adjacent vertices*

$\therefore |h_1 - h_3| \leq |h_1 + h_2 - h_3|$

$\therefore \int_0^t |[h_1 - h_3] * \phi(x,t)| \mathrm{d}t \leq \int_0^t |[h_1 + h_2 - h_3] * \phi(x,t)| \mathrm{d}t$

$\therefore K(h_1, h_3) \leq K(h_1 + h_2, h_3)$ ■

Proposition 2: Boundary term remains nearly constant during optimization.

Proof: There are two cases of boundary change:

1) vertex i and vertex j have no shared boundaries.The boundary term of edge is not affected by the merging process.
2) vertex i and vertex j will merge only if they have similar histogram, which means $Intra(v_i) \approx Intra(v_j)$.. Assuming that it has a vertex k adjacent to v_j, after merging process, we get new vertex v_{ij}, It is easy to know taht the $Inter(v_j, v_k)$ will keep constant, and $Intra(v_{ij}) \approx Intra(v_i) \approx Intra(v_j)$. Therefore, $dif(v_{ij}, v_k) \approx dif(v_j, v_k)$

■

Proposition 3: The edge weight will not decrease during the merging process.

By exploiting these observations, we can achieve an efficient implementation, called lazy optimization. Supposing N superpixel need to be produced, we set max superpixel size to be $2 * S/N$, where S is the total pixel number in the image. A pseudocode is given as follow:

The RAG initialization takes $O(|V|)$ to scan all second-order superpixels. The optimization algorithm loop $O(|E|)$ times to merge vertices connected by the RAG edge. In each iteration, we find minimum edge using heap data structure. Therefore, the complexity of the algorithm is $O(|E| \log |E|)$. Since we only construct one hop edge in RAG, the complexity is then $O(|V| \log |V|)$. $|V|$ is not the number of pixel in image but the number of second-order superpixel, which is the same order of magnitude of final superpixel, usually 200 times less than image pixel number in the experiment.

Algorithm 2 Merging Process

Input: An RAG $G_{I''}$ of second-order superpixel I''
Output: A RAG G represents the different regions of input RAG.

```
Build min heap using RAG G_{I''} edge weight
while vertex number n > preset number N do
  if Edge in heap top is merged then
    Pop heap
    Remove the popped element
  end if
  Pop edge e(v_i, v_j) with minimum weight
  if size(v_i)+size(v_j) < 2 * S/N then
    Merge two vertices by using Eq. (1)
    Update weight and terms related to the new vertex
    Update RAG G_{I''} structure
    Update the min heap
  end if
end while
```

IV. Comparision

In this section, We present the comparison result on the BSDS500 contour detection benchmark [12]. BSDS500 dataset have 500 images and their ground truth segmentation. For each image, we run different algorithms on 8 different scale levels, from 50 to 400 superpixels and compute the average performance over all images. We compare Hybrid superpixel with other state-of-art superpixel approaches and use standard metrics that are commonly used for evaluating the quality of superpixels: under-segmentation error, boundary recall and achievable segmentation accuracy. These metrics are exactly used in [7] and [8].

We use the default parameter set in [6] and [7] in our pre-processing phase. In the optimization part of hybrid superpixel algorithm, we set histogram bin number to be 5 in each channel when computing histogram diffusion distance as the region term of edge weight. During merging, the max superpixel size is set to be $2 * S/N$. As result, no superpixel will be larger than this upper bound. All experiments are performed on a four-core CPU (2.5 GHz Intel Core i7). We will analyze the effect of these parameter in the latter.

A. Benchmark Test

In the first experiment, we compare our results with SLIC [6], FH [7], ESR [8] and SEED [9] using the evaluation metrics shown above. The result is shown in Fig. 5. The hybrid outperforms its ancestor algorithm FH and SLIC in under-segmentation error and achievable segmentation accuracy curve. With 200 superpixels, the under-segmentation error of hybrid is only 0.11 while SLIC is 0.12 and FH is 0.21. In achievable segmentation accuracy curve, hybrid is 0.93 while SLIC is 0.92 and FH is 0.89. Hybrid also matches the boundary recall of FH, ESR and SEEDS. When the amount of superpixel is larger than 200, one can barely differentiate the performance curves of Hybrid, SEEDS and ESR, where the latter two are regarded as the best superpixel algorithms so far.

In the second experiment, we analyze the parameter effect. The first one is max superpixel size which guides the merging process (See Fig. 6) and the second is the bin number in

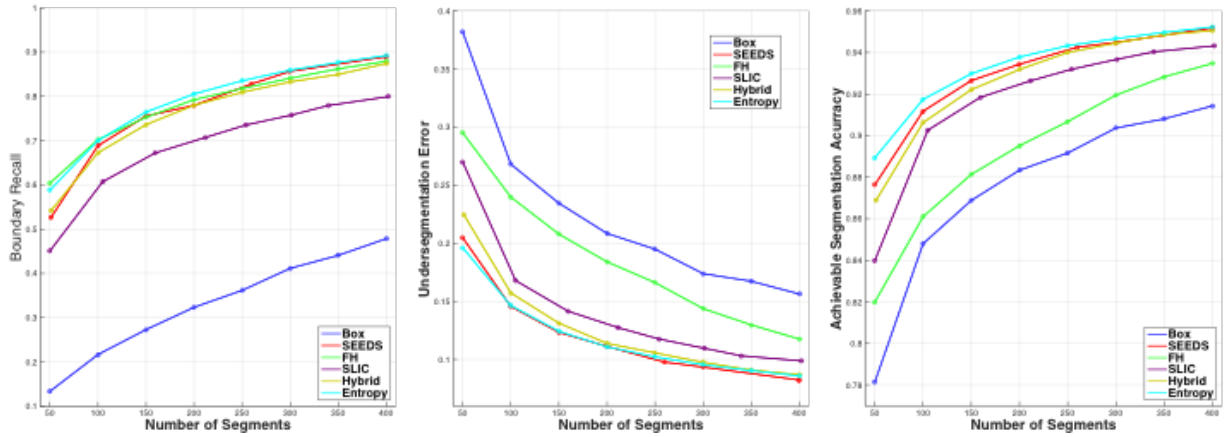


Fig. 5: Performance metrics: left: Boundary recall; middle: Under-segmentation error; right: achievable segmentation accuracy

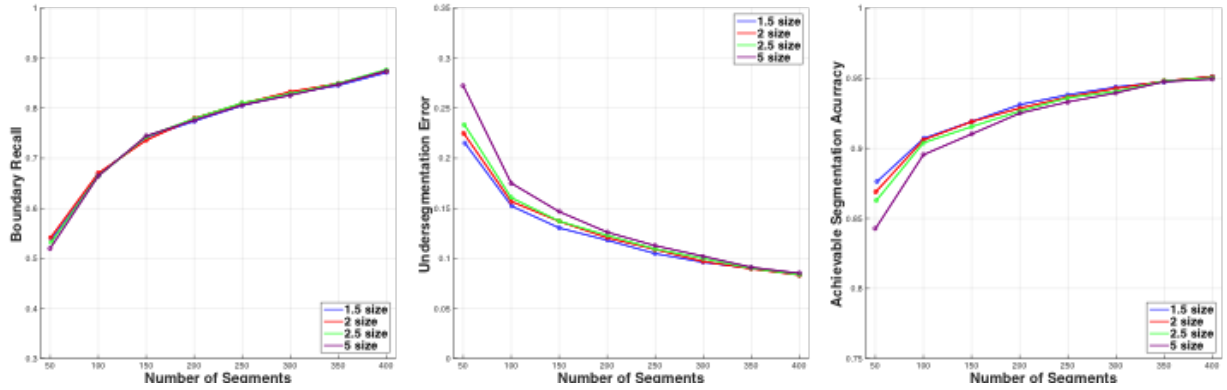


Fig. 6: Effect of max superpixel size on the performance metrics

histogram diffusion Distance (See Fig. 7). Competitive results are achieved within a wide range of parameter selection.

In the third experiment, we compare hybrid with SEEDS, ERS, SLIC and FH visually. Several examples are shown in Fig. 8. The image is segmented into 100 superpixel. Hybrid combines the boundary adherence of FH and the regular superpixel shape of SLIC. As result, its performance matches the best superpixel algorithms, ESR and SEEDS.

In Fig. 9, we further compare the details of hybrid and SLIC and show the reason that hybrid outperform SLIC in all three metric. SLIC fails to segment the cross on the church because of its inherent limitation, only one seed planted in that area. Hybrid does not have this limitation and successfully segment the cross. It is the same case with statue head segmentation.

The proposed algorithm has the same run-time with the state-of art method. The hybrid superpixel takes 0.58 seconds. For the ancestor algorithm of hybrid superpixel, SLIC runs 0.28 seconds and FH runs 0.21 seconds. The hybrid superpixel adds only small overhead to its ancestor algorithm but bring huge performance increase. Hybrid superpixel is also faster than entropy rate superpixel (1.3 seconds). The SEEDS is still the fast algorithm (0.12 seconds) however hybrid superpixel can also achieve this level of speed if it chooses SEEDS as its ancestor algorithm.

B. Application of Hybrid Superpixel

Superpixel can speed up many existing computer vision task, and even improve its result in some case [6]. We consider a typical vision task – multi-object class recognition in this paper.

We perform our task in MSRC dataset [13] which contains 21 different objects, and adopt the object class recognizing tool in STAIR [14]. It extracts features for each region class. Then, boosted classifiers are learned using these features. Finally, a Conditional Random Field (CRF) model is learned using the output of boosted classifier as its features. We also segment each image into about 200 superpixels as the input of classifier. As shown in Table. I. Hybrid superpixel has increased the accuracy of classification successfully.

TABLE I: Accuracy of multi-object class recognition for different superpixel methods (adapted from [15]).

Algorithm	FH(GS04 in [15])	SLIC	Hybrid
Pixelwise accuracy	74.6%	76.9%	80.1%

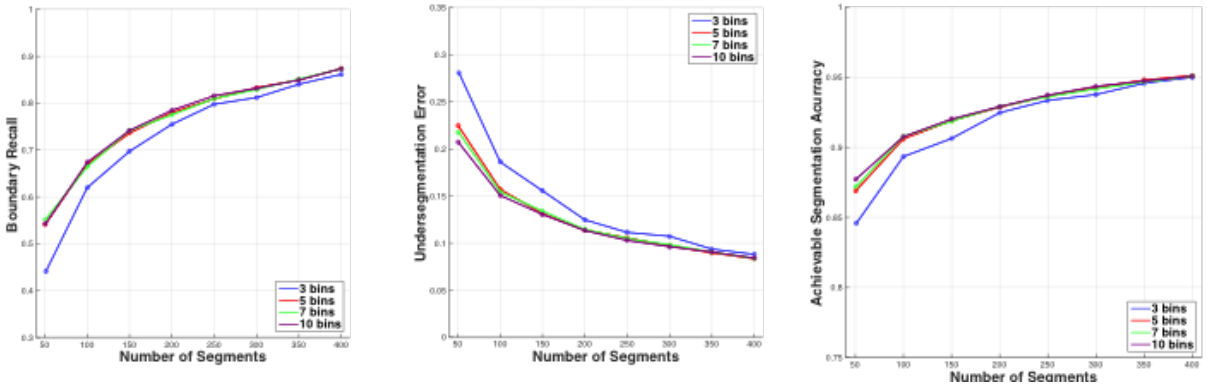


Fig. 7: Effect of bin number on the performance metrics

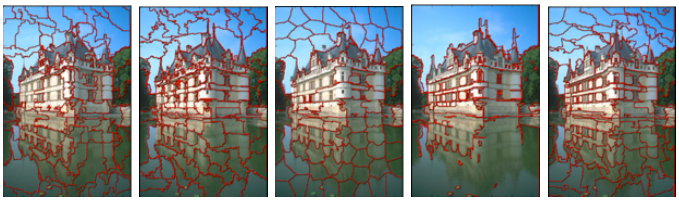
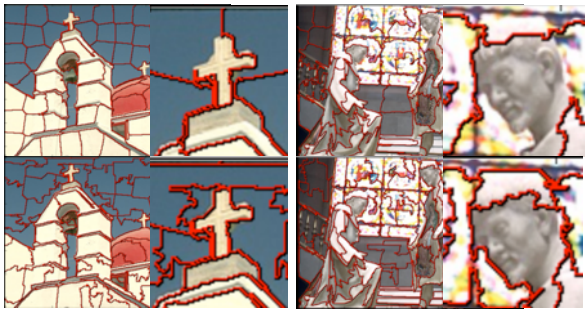


Fig. 8: Superpixel segmentation result. From left to right, SEEDS, Entropy, SLIC, FH and proposed hybrid



(a) (b) (c) (d)

Fig. 9: SLIC and Hybrid. In each pair SLIC is on the above and Hybrid on the bottom

V. Conclusion

In this paper, we propose a novel fusion algorithm, called Hybrid superpixel. It has the flexibility to incorporate any two different superpixels to get better result. We make further contribution in following aspects:

- We present a novel energy function on Regions Adjacency Graph (RAG) which consists of region term and boundary term for generating superpixels with good boundary adherence and structure relationship.
- We conduct the evaluation in our novel algorithm and the state-of-art, Hybrid approach shows better performance than its ancestor algorithms in all standard evaluation metrics and the one of the practical application.

References

[1] X. Ren and J. Malik, "Learning a classification model for segmentation," in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*. IEEE, 2003, pp. 10–17.

[2] Y. Li, J. Sun, C.-K. Tang, and H.-Y. Shum, "Lazy snapping," in *ACM Transactions on Graphics (ToG)*, vol. 23, no. 3. ACM, 2004, pp. 303–308.

[3] G. Mori, "Guiding model search using segmentation," in *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, vol. 2. IEEE, 2005, pp. 1417–1423.

[4] J. Tighe and S. Lazebnik, "Superparsing: scalable nonparametric image parsing with superpixels," in *Computer Vision–ECCV 2010*. Springer, 2010, pp. 352–365.

[5] A. Levinshtein, A. Stere, K. N. Kutulakos, D. J. Fleet, S. J. Dickinson, and K. Siddiqi, "Turbopixels: Fast superpixels using geometric flows," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, no. 12, pp. 2290–2297, 2009.

[6] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk, "Slic superpixels compared to state-of-the-art superpixel methods," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 11, pp. 2274–2282, 2012.

[7] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *International Journal of Computer Vision*, vol. 59, no. 2, pp. 167–181, 2004.

[8] M.-Y. Liu, O. Tuzel, S. Ramalingam, and R. Chellappa, "Entropy rate superpixel segmentation," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, 2011, pp. 2097–2104.

[9] M. Van den Bergh, X. Boix, G. Roig, B. de Capitani, and L. Van Gool, "Seeds: Superpixels extracted via energy-driven sampling," in *Computer Vision–ECCV 2012*. Springer, 2012, pp. 13–26.

[10] A. Trémeau and P. Colantoni, "Regions adjacency graph applied to color image segmentation," *Image Processing, IEEE Transactions on*, vol. 9, no. 4, pp. 735–744, 2000.

[11] H. Ling and K. Okada, "Diffusion distance for histogram comparison," in *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, vol. 1. IEEE, 2006, pp. 246–253.

[12] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 33, no. 5, pp. 898–916, 2011.

[13] J. Shotton, J. Winn, C. Rother, and A. Criminisi, "Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context," *International Journal of Computer Vision*, vol. 81, no. 1, pp. 2–23, 2009.

[14] S. Gould, I. Goodfellow, P. Baumstarck, A. Y. Ng, and D. Koller. (2010) The STAIR Vision Library (v2.4). Stanford AI Lab. [Online]. Available: http://ai.stanford.edu/~sgould/svl

[15] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC superpixels," EPFL, Tech. Rep., 2010.