Efficient Segmentation for Region-based Image Retrieval Using Edge Integrated Minimum Spanning Tree

Yang Liu*, Lei Huang*, Siqi Wang†, Xianglong Liu* and Bo Lang*
*State Key Laboratory of Software Development Environment
†School of Computer Science and Engineering
Beihang University, No.37 Xueyuan Rd., Beijing, China
Email: *{blonster, huanglei, xlliu, langbo}@nlsde.buaa.edu.cn, †wangsiqi@buaa.edu.cn

***Abstract*—Region-based Image Retrieval (RBIR), which bases itself on image segmentation rather than global features or key-point-based local features, is a branch of Content-based Image Retrieval. This paper proposes a novel RBIR-oriented image segmentation algorithm named Edge Integrated Minimum Spanning Tree (EI-MST). The difference between EI-MST and the traditional MST-based methods is that EI-MST generates MSTs over edge-maps rather than the original images, which achieved high retrieval performance cooperating with state-of-the-art matching strategies. In addition, by limiting the nodes in every MST with adaptive scale selection, EI-MST is efficient especially when processing high resolution images. The experiments on four popular public datasets proved that, EI-MST is capable of achieving higher retrieval accuracy over four widely used segmentation methods while only consuming moderate amount of time in both online and offline parts of RBIR systems.**

I. Introduction

As a branch of Content-based Image Retrieval (CBIR), Region-based Image Retrieval (RBIR) aims to solve the same problem, which is filling the gap between visual features and semantic meanings of images. Different from global feature or local feature based CBIR methods, RBIR compares two images by evaluating the similarity between homogenous (or semantically meaningful) regions. Intuitively, RBIR works in a scale half-way between global features like *Color and Edge Directivity Descriptor* (CEDD) [1] and key-point-based local features such as SIFT [2] and SURF [3].

There are mainly two types of RBIR systems: **a)** a few of them change the retrieval target from images to regions, i.e. instead of matching the query image to database images, these systems choose to match the query region(s) to regions, so these methods mostly need users to select the ROIs (region of interest) from query images; **b)** other systems were designed for the standard content-based image retrieval task, which makes them valid substitutes for any existing CBIR system. *This paper is focused on the second type.*

A RBIR system normally works through the following steps: **1)** segment the images into regions with a certain segmentation algorithm; **2)** extract visual features from each region; **3)** convert the similarities between regions into the similarities between images through certain matching strategy. Hence, there are three crucial parts consisting in a RBIR system, which are the *segmentation algorithm*, *visual feature* and *matching strategy*.

Segmentation algorithms, as an important part of a RBIR system, should have great influence on the efficiency and retrieval performance of the entire system. However, few papers paid much attention to it, while there were many papers proposing different matching strategies [4], [5] in the last few years. Most of the papers about RBIR chose existing segmentation algorithms as part of their systems without giving a reason.

The image segmentation method proposed here is named *Edge Integrated Minimum Spanning Tree* (EI-MST). EI-MST is a Minimum Spanning Tree (MST) based segmentation method inspired by *Recursive Shortest Spanning Tree* (RSST) [6] and *Local Variation segmentation* (LV) [7]. However, different from RSST and LV, EI-MST generates MSTs from edge maps rather than the original images, and meanwhile it scores a tree edge by collecting information from multiple nodes associated to it instead of its two ends. These differences lead to some special characteristics and help EI-MST achieving high stability in the experiments. Fig.1 shows a brief illustration of the segmentation process of EI-MST: 1) an edge map is generated from the original image; 2) a MST is built on the edge map; 3) the MST is split into subtrees, each of which represents an individual region.

In addition, EI-MST uses relatively large scale grid to divide the original images into cells, and then convert these cells into vertices of MSTs. In this way, the number of vertices in a MST is limited, and so is the time cost of segmentation process.

In order to evaluate the performance of EI-MST, it was compared to a few popular color image segmentation algorithms in a series of experiments: for evaluating retrieval performance, different combinations of segmentation algorithm, visual feature and matching strategy are tested on four widely used public datasets; for evaluating efficiency, time costs of competing segmentation algorithms are compared. Results of these experiments will be presented and discussed in Sec.IV.

The last few years saw many attentions been given to *Semantic Segmentation* [8], [9]. It works a lot like RBIR

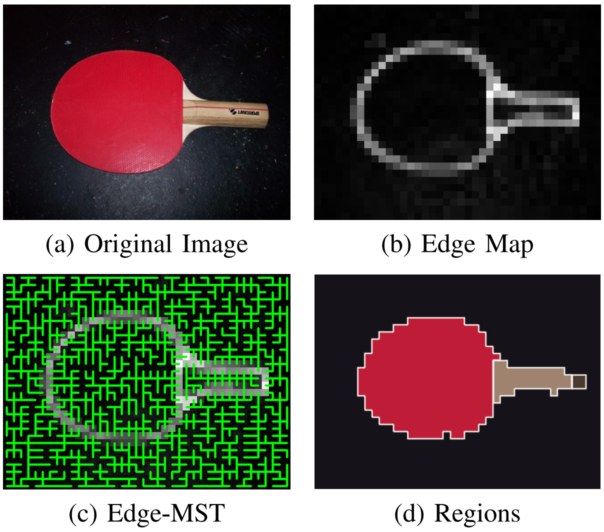
(a) Original Image
(b) Edge Map
(c) Edge-MST
(d) Regions

Fig. 1. Image segmentation process of EI-MST.

systems of the first type, because they both try to select a group of regions from images according to some given conditions, which for semantic segmentation could be models of a specific object category. However, for a practical RBIR system, the classification process of semantic segmentation could prove too costly and not necessary for standard image retrieval task, and it is nearly impossible to train enough models for a vast amount of arbitrary images. In contrast, traditional unsupervised segmentation methods [7], [10], [11], [12], though did not see much development recently, are much lighter and more general, which makes them more suitable for RBIR systems than semantic segmentation algorithms. Therefore, the competing methods we chose are widely used unsupervised segmentation algorithms. They may be a little old but their effectiveness has been proven by existing systems.

The main *contribution of this paper* lies in two parts:

- a novel way of generating Minimum Spanning Trees from images which makes use of edge detection results;
- an efficient MST-based segmentation algorithm specially designed for region-based image retrieval, named *Edge Integration Minimum Spanning Tree* (EI-MST).

The rest of this paper is structured as follows. Before proposing EI-MST, the generation process and characteristics of edge-MST are introduced in Sec.II. Then, Sec.III gives a fully detailed description of EI-MST. In order to evaluate the performance of EI-MST, several experiments have been conducted, and their results will be presented and discussed in Sec.IV. Finally, Sec.V concludes our work.

II. MST BASED ON EDGE MAP

To avoid ambiguity, from now on, the phrase "**edge**" stands for curves that separate two regions, and phrases "**graph edge**" and "**tree edge**" mean edges in the graph theory definition.

As the name suggests, EI-MST is a MST-based segmentation method. Classic MST-based image segmentation methods normally work in the following steps:

1) the image is divided into fix-sized cells (patches) with a grid, and certain features are extracted from each cell;
2) an adjacency graph $G = (V, E)$ is constructed by regarding cells as vertices and connecting each cell to its 4 or 8 neighbors with undirected edges, and then a Minimum Spanning Tree $T = (V, E_t)$ is generated from G, i.e. $E_t \subset E$;
3) tree edges of T are scored, and then edges with high scores are cut to split the original tree into a forest of subtrees, in which each subtree represents a homogeneous region of the target image.

According to the above steps, the construction of MSTs forms the basis, so the characteristics of MSTs are crucial to the segmentation processes. Step 3 shows that all MST-based methods are based on the assumption that *each homogeneous region in the image can be represented by a subtree of the MST*, which may be called ***Correspondence Assumption***. When the correspondence assumption does not hold, no MST-based method can have good segmentation performance.

The rest of this section will present edge-MST. Compared to traditional MSTs, edge-MSTs are built over edge maps rather than original images, and Sec.II-B will show that they are more likely to satisfy the correspondence assumption.

A. Edge map generation

For simplicity and efficiency, image gradient is used in EI-MST for edge detection, i.e. $|v(x,y)| = \sqrt{\|I(x+1,y) - I(x,y)\|^2 + \|I(x,y+1) - I(x,y)\|^2}$, where $v(x,y) \in V$ represents the cell (x,y) with $|v(x,y)|$ being its edge strength, and $I(x,y)$ is the average color of pixels in cell (x,y). The reason of making this choice lies in the fact that this paper is supposed to be focused on two things: the generation of MST and segmentation process based on MST. Moreover, a robust algorithm should be able to tolerate minor instability of the edge detection.

The only problem left is how to determine the scale of edge maps. The word "**scale**" here is referred to as the size of grid cells. More specifically, assuming that each cell in a patch consists of $s \times s$ pixels, s is the parameter representing scale.

Obviously, parameter s determines how fine or coarse will the edge detection and the segmentation work, so it may have strong influence on the performance and efficiency. Simply speaking, smaller s could lead to finer (not always better) segmentation and larger time cost, while larger s will probably make the algorithm run faster but the segmentation result may be a little sketch-like.

Normally, we can choose a fixed s for all the images, and it works fine if the resolutions of images are mostly similar, e.g. images from the same public dataset. For images varying much in size, however, it might be better to choose s for each individual image according to its size. The experiments presented in Sec.IV are conducted on four different datasets with images from around 100,000 pixels to 6,000,000 pixels, so for these experiments we propose ***Adaptive Scale Selection***:

$$s = \left[\sqrt{H \cdot W / F} + 1\right], \tag{1}$$

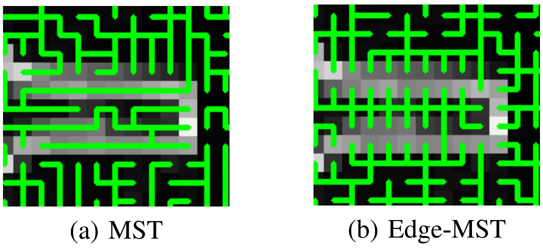

Fig. 2. Difference between a traditional MST and an edge-MST.

where H and W are the height and width of the image, $[c]$ is the closest integer of c, and F is an integer constant representing how many cells should be in an edge map.

Due to the purpose of RBIR, the segmentation method should be fast enough so that it doesn't slow down the whole system, and textures are probably better to be ignored during segmentation. Therefore, large s or small F might be more suitable for RBIR-oriented applications.

B. Construction and characteristics of edge-MSTs

Traditional MST-base methods [6], [7] construct adjacency graphs and MSTs directly on the original images, i.e. the weight of $\{a, b\} \in E$ is $\|I(a) - I(b)\|$. In this way, two cells connected by a tree edge are supposed to be similar.

EI-MST generates adjacency graphs and their associated MSTs on edge maps, i.e. the weights of graph edges are assigned according to edge strengths instead of original feature vectors. More specifically,

$$\omega(\{a, b\}) = |a|^2 + |b|^2, \{a, b\} \in E, a \in V, b \in V, \quad (2)$$

where $\omega(\{a, b\})$ is the weight of graph edge $\{a, b\}$. Obviously, instead of trying to connect similar cells, the tree edges of an edge-MST tend to avoid cells with high edge strengths, which makes these cells have very low degrees (mostly 1).

Fig.2 shows the comparison between a MST and an edge-MST based on the same zoom area of Fig.1(a). Intuitively, MST vertices with high edge strengths can have degrees of any number, and branches could go along edges without difficulty. In contrast, almost all the edge-MST vertices with high edge strengths are leaves, and the branches tend to cross the edges. Thus, *the correspondence assumption is more likely to hold for edge-MSTs than traditional MSTs*.

III. Edge Integrated Minimum Spanning Tree

As shown in Fig.1, the process of EI-MST is almost the same as those of most MST-based methods with only one difference, i.e. the generation of edge maps. Therefore, after edge-MSTs are obtained, the work left for EI-MST is to split the original tree into a forest of its subtrees. EI-MST has two major differences from the other MST-based methods:

- each tree edge is scored by collecting edge strength information from the two subtrees connected by it, instead of its two ends (Sec.III-A);
- after cutting a tree edge, scores of other edges are updated through simple set operations instead of rerunning costly scoring processes (Sec.III-B).

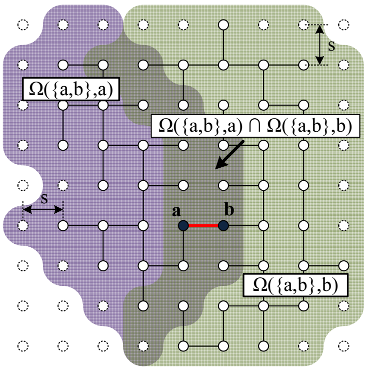

Fig. 3. Intersection set of a tree edge $\{a, b\}$.

A. Scores of tree edges

Let $\Lambda(e)$ denote the score of a tree edge $e \in E_t$. Due to the purpose of segmentation, $\Lambda(e)$ should reflect the likelihood that, after cutting e the original tree will split into two subtrees each representing a homogeneous region. However, this assumption is hardly true during the early iteration cycles, e.g. after cutting the first edge both of the two temporary regions are most likely unions of many smaller regions rather than two large homogeneous regions.

Most of the MST-based methods choose to let $\Lambda(e) \equiv \omega(e)$. In this way, tree edges crossing region edges are more likely to be cut first, and after a few cuts the subtrees should correspond to homogeneous regions. This idea is inspired, but when applied to large scale patches it is usually unstable and tends to get numerous small regions around rough edges. Therefore, we extended this idea into a more stable form in EI-MST.

For each tree edge $\{a, b\} \in E_t$, two sets of nodes are generated, which are denoted by $\Omega(\{a, b\}, a)$ and $\Omega(\{a, b\}, b)$ respectively, where $\Omega(\{a, b\}, v)$ consists of all the nodes that are either within or in possession of at least one neighbor in the subtree attached to $v \in \{a, b\}$, as shown in Fig.3. After the Ω sets are generated, we can easily get the **intersection set** $I(\{a, b\})$ through the following equation:

$$I(\{a, b\}) = \Omega(\{a, b\}, a) \cap \Omega(\{a, b\}, b). \quad (3)$$

According to the definition of Ω sets, $I(\{a, b\})$ should consist of all nodes around the edge separating the two subtrees connected by $\{a, b\}$. Apparently, by taking into consideration all elements in $I(\{a, b\})$ instead of merely $\{a, b\}$, random errors are less likely to affect the segmentation results.

The intersection set is supposed to include all nodes around the edge, so it seems logical to score a tree edge according to its intersection set. In summary, the scoring equation of EI-MST is as follows:

$$\Lambda(\{a, b\}) = \frac{\sum\limits_{u \in I(\{a,b\})} |u|}{|I(\{a, b\})|^q}, \{a, b\} \in E_t, q \in \mathbb{R}^+. \quad (4)$$

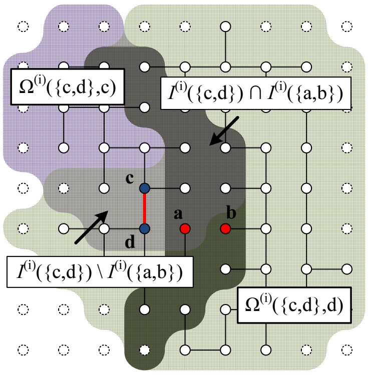

Fig. 4. Intersection set of $\{c, d\}$ after cutting $\{a, b\}$.

According to (4), the scoring strategy of EI-MST is based on the integration of edge strengths of cells in the edge map. $|I(\{a, b\})|$ denotes the number of elements within $I(\{a, b\})$. q is a constant which needs tuning. Generally speaking, a model with small q tends to segment images into a few big regions, while one with large q usually yields numerous small regions.

B. Splitting the tree

After the edges are scored, it should be easy to cut the edge with the highest score and split the whole tree into two subtrees each representing a temporary region. However, after the first cut a problem arises: all the edge scores were calculated on the original tree and cannot be used directly in the following iteration cycles, because cutting them will be splitting the subtrees instead of the original one. Therefore, their scores should be updated before the next iteration cycle.

If $e_0 \in E_t$ is cut during the ith iteration cycle and e_0 belongs to subtree $T' = (V', E')$, then for $\forall e \in E'$ and $\forall v \in e$, let

$$\Omega^{(i+1)}(e, v) = \Omega^{(i)}(e, v) \setminus I^{(i)}(e_0). \qquad (5)$$

By doing so, $I^{(i)}(e)$ becomes $I^{(i+1)}(e) = I^{(i)}(e) \setminus I^{(i)}(e_0)$, and then $\Lambda^{(i+1)}(e)$ can be recalculated according to $I^{(i+1)}(e)$. Fig.4 illustrates the effect of updating Ω sets of tree edge $\{c, d\}$ after cutting $\{a, b\}$ in the previous iteration.

As shown in Fig.4, $I^{(i)}(\{c, d\})$ contains many nodes of $I^{(i)}(\{a, b\})$ since they are based on the same subtree. However, after cutting $\{a, b\}$, the subtree attached to d shrank, so most of the nodes in $I^{(i)}(\{a, b\})$ should not be in $I^{(i+1)}(\{c, d\})$ any more. Of course, by setting $I^{(i+1)}(\{c, d\}) = I^{(i)}(\{c, d\}) \setminus I^{(i)}(\{a, b\})$, we may lose a few nodes such as a, because all nodes in $I^{(i)}(\{a, b\})$ are excluded arbitrarily. However, this loss is tolerable when compared to the cost of recalculating all the Ω sets.

In summary, the process of segmentation is as follows: the tree edge with the highest score is cut at the beginning of each iteration cycle, and then scores of the tree edges belonging to the same subtree as the cut one are updated before the next iteration cycle. This process is done over and over again until the highest score among the rest tree edges is under a threshold t, and then the survived tree edges form a forest with each tree in it representing a homogeneous region. An example of segmentation result of EI-MST is given by Fig.1(d).

IV. Evaluation

To evaluate the efficiency and retrieval accuracy of EI-MST, we compared EI-MST to four widely used color image segmentation algorithms in a series of experiments:

1) different combinations of segmentation algorithm, visual feature and matching strategy are tested on public datasets, and the retrieval performances are compared;
2) time costs of the competing segmentation algorithms are compared in order to evaluate their efficiency.

In type 1 experiments, segmentation algorithms are first applied to convert images into sets of regions with each region described by a specific visual feature. Then, segmented images are matched via Integrated Region Matching [4] (IRM) or Bag-of-Regions [5] (BOR). More specifically,

- **for IRM**, each region is given a weight equal to the proportion of its area to the whole image, and similarities between region sets (images) are calculated directly through the matching strategy of IRM;
- **for BOR**, a codebook will first be generated for each dataset by using K-means clustering. Then, region sets are converted into BOR vectors by weighting each region with a saliency measure proposed in [13], and the vectors are compared with Euclidean Distance.

The retrieval performances are evaluated with *Mean Average Precision* (MAP), and the results will be discussed in Sec.IV-A and Sec.IV-B.

In type 2 experiments, only the time costs of segmentation processes are compared, i.e. the time consumed by image loading, feature extraction and matching is not contained in the final results. In practical RBIR systems, the efficiency of segmentation methods will mostly affects the offline time cost, while the online efficiency are usually determined by *Average Region Count* (ARC), which is the average number of regions generated for one image. Therefore, as well as MAP, ARC has also been recorded in type 1 experiments, and the results will be discussed in Sec.IV-C

Specifically, the setup of the experiments is as follows:

- **Datasets**: datasets used in the following experiments are INRIA Holiday [14] (1491 pictures), ZuBuD [15] (1005 pictures), UCID [16] (1338 pictures with 200 queries) and ukbench [17] (10200 pictures);
- **Segmentation Algorithms**: competing algorithms other than EI-MST are *Local Variation segmentation* (LV) [7], JSEG [10], *Mean Shift segmentation* (MS) [11] and *Color Watershed Adjacency Graph Merge* (CWAGM) [12];[1]
- **Visual Features**: CEDD [1], AlexNet [18] and R-CNN [19] are used to describe images and regions.[2]

[1] MS and CWAGM are implemented by LTI-LIB project.

[2] Models of AlexNet and R-CNN are trained by Caffe project [20].

TABLE I
RETRIEVAL PERFORMANCE OF IRM+CEDD IN MAP (%). ($F = 8000$, $q = 0.9$, $t = 0.2$)

	RBIR										CBIR
	LV		**JSEG**		**MS**		**CWAGM**		**EI-MST**		**CEDD**
	MAP	ARC	MAP	ARC	MAP	ARC	MAP	ARC	MAP	ARC	MAP
ZuBuD	85.83	241	62.99	68	81.89	37	**88.57**	536	87.62	136	79.12
UCID	71.93	136	36.46	72	70.73	36	74.54	390	**75.00**	170	67.41
ukbench	72.91	149	52.06	54	76.16	34	76.94	372	**78.61**	65	70.26
Holiday	73.96	255	55.37	186	71.54	36	73.45	4854	**74.19**	107	69.82

TABLE II
RETRIEVAL PERFORMANCE OF BOR+CEDD IN MAP (%). ($F = 8000$, $q = 1.0$, $t = 0.05$)

	Codebook	RBIR					CBIR
		LV	**JSEG**	**MS**	**CWAGM**	**EI-MST**	**CEDD**
ZuBuD	200	78.87	60.95	53.05	84.34	**85.83**	79.12
	2000	76.36	56.84	53.50	81.90	**84.23**	
UCID	200	60.77	58.26	52.55	**69.00**	**69.01**	67.41
	2000	57.78	47.28	50.46	60.17	**64.63**	
ukbench	200	59.86	60.48	47.68	68.50	68.92	**70.26**
	2000	55.27	56.92	43.50	61.33	63.09	
Holiday	200	57.55	55.87	50.36	67.14	66.68	**69.82**
	2000	56.68	56.01	47.30	63.61	63.45	

A. *Working with CEDD*

CEDD [1] is one the most popular global features for CBIR. It integrates color features and textural features into fixed-sized vectors, which makes it highly efficient and capable of reflect a series of different image characteristics. So far, CEDD has been used in many CBIR systems.

The retrieval performance of segmentation methods working with CEDD is evaluated by cooperating with IRM or BOR, i.e. the results of different segmentation methods cooperating with IRM are compared to each other (Table I) and the same goes for BOR (Table II). The columns marked CEDD give the MAPs of pure CEDD as a baseline, and the same goes for R-CNN and AlexNet in Table III.

With both IRM and BOR, EI-MST achieved the best performance among all the competing methods, which proved the effectiveness of EI-MST as a part of RBIR system. However, BOR only beat pure CEDD on two datasets (ZuBud and UCID), while on the other two the segmentation process seems a waste of time since all the results of RBIR systems are even worse than those of pure CEDD.

Compared to the other 3 methods, EI-MST and CWAGM performed significantly better. This might be because these two methods are both edge sensitive, which means that the segmentation results of EI-MST and CWAGM are more dependent on the clarity of edges than the uniformity of regions.

B. *Working with CNN*

Along with the flourish of Deep Learning, the last few years saw *Convolutional Neural Network* (CNN) [18], [19] draw much attention in the area of image analysis, especially retrieval and recognition. By training models with vast amount of data, CNN-based methods achieved amazingly high performances on a series of contests and benchmarks. However, compared to traditional manually designed features such as CEDD, CNN requires an additional complicated and time consuming training process, which makes it a little difficult to use in many practical systems. Therefore, a few researchers chose to use pre-trained CNN models in their system as substitutes of traditional visual features, and their results are encouraging. We adopted this idea and designed the following experiments to evaluate EI-MST from a different angle.

Table III shows the retrieval performances of competing segmentation methods cooperating with IRM and R-CNN (pre-trained model). We also conducted experiments with the combination of IRM+AlexNet, but the results of IRM+AlexNet are in general slightly worse than those of IRM+R-CNN. Therefore, we only discuss the results of IRM+R-CNN.

EI-MST outperformed all the competing methods here, and EI-MST+IRM+R-CNN shows improvement over pure R-CNN on all four datasets. However, when compared to AlexNet, EI-MST+IRM+R-CNN only won on two datasets (ZuBuD and Holiday), while on the other two the results of EI-MST+IRM+R-CNN are slightly worse than those of AlexNet.

On the whole, the superiority of RBIR is limited when compared to CNN models (R-CNN and AlexNet). The reason could be that there is much overlap between the information provided by segmentation and that provided by CNN models, so RBIR systems cannot get enough additional information to improve the retrieval performance.

C. *Efficiency Issue*

As shown in Table IV, EI-MST is significantly faster than the other algorithms, especially on INRIA Holiday dataset. This could be because images in INRIA Holiday are extremely big (around 3000 × 2000 pixels) compared to the other 3 datasets, and adaptive scale selection prevented edge-MSTs from growing overly large while processing high resolution pictures. Meanwhile, EI-MST worked very well with adaptive scale selection and achieved high retrieval performance.

TABLE III
RETRIEVAL PERFORMANCE OF IRM+R-CNN IN MAP (%). ($F = 4800$, $q = 0.7$, $t = 0.5$)

	RBIR										**CBIR**	
	LV		**JSEG**		**MS**		**CWAGM**		**EI-MST**		**R-CNN**	**AlexNet**
	MAP	ARC	MAP	ARC	MAP	ARC	MAP	ARC	MAP	ARC	MAP	MAP
ZuBuD	74.52	210	46.72	29	82.05	6	79.43	249	**86.44**	89	83.01	83.38
UCID	80.69	120	31.92	51	81.05	7	80.51	204	81.18	50	80.15	**82.88**
ukbench	82.67	152	6.01	19	**85.50**	7	84.42	216	84.00	25	83.53	84.95
Holiday	70.45	241	48.98	125	77.61	7	58.75	1697	**79.63**	58	76.60	77.43

TABLE IV
TOTAL TIME COSTS OF DIFFERENT SEGMENTATION ALGORITHMS IN SECONDS.

	LV	**JSEG**	**MS**	**CWAGM**	**EI-MST**
ZuBuD	721.18	1.23×10^5	201.46	209.25	**118.10**
UCID	**94.26**	8427.52	195.77	510.08	216.62
ukbench	1576.92	1.08×10^5	2024.81	2318.87	**1047.54**
Holiday	5682.20	4.50×10^6	4979.94	3657.06	**186.52**

The online efficiency of IRM mainly depends on ARC, i.e. higher ARC almost always means larger time cost. According to Table I and Table III, there seems to be a positive correlation between ARC and MAP for IRM+CEDD systems and a negative correlation for IRM+CNN systems, though the relationships are vague and weak. In both experiments, EI-MST achieved the highest accuracy among all competing methods with moderate ARCs, which proved its capability of optimizing the tradeoff between accuracy and efficiency. However, IRM is still a heavy-loaded matching strategy, so it may need to be combined with speed-up methods such as hashing [21], [22], [23] in order to form practical systems.

V. CONCLUSION

This paper proposed a novel image segmentation algorithm named EI-MST and evaluated its efficiency and retrieval performance by comparing it to a few widely used segmentation algorithms on four popular public datasets.

The experiments proved that EI-MST achieved the highest retrieval accuracy among all the competing methods. Meanwhile, EI-MST is highly efficient both online and offline, especially when processing high resolution images such as INRIA Holiday pictures. Therefore, EI-MST is suitable for processing pictures taken with high resolution cameras which are common in user generated media.

ACKNOWLEDGMENT

This work is partially supported by the National Natural Science Foundation of China (Grant Nos: 61402026 and 61370125), the Innovation Foundation of BUAA for PhD Graduates and the Foundation of the State Key Laboratory of Software Development Environment (Grant No. SKLSDE-2015ZX-04).

REFERENCES

[1] S. A. Chatzichristofis and Y. S. Boutalis, “Cedd: color and edge directivity descriptor: a compact descriptor for image indexing and retrieval,” in *Computer vision systems*. Springer, 2008, pp. 312–322.

[2] D. Lowe, “Object recognition from local scale-invariant features,” in *ICCV*, vol. 2, 1999, pp. 1150–1157 vol.2.

[3] H. Bay, T. Tuytelaars, and L. V. Gool, “Surf: Speeded up robust features,” in *ECCV*, 2006, pp. 404–417.

[4] J. Li, J. Z. Wang, and G. Wiederhold, “Irm: integrated region matching for image retrieval,” in *ACM MM*. ACM, 2000, pp. 147–156.

[5] R. Vieux, J. Benois-Pineau, and J.-P. Domenger, *Advances in Multimedia Modeling: 18th International Conference, MMM 2012*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, ch. Content Based Image Retrieval Using Bag-Of-Regions, pp. 507–517.

[6] S. H. Kwok and A. G. Constantinides, “A fast recursive shortest spanning tree for image segmentation and edge detection,” *IEEE Transactions on Image Processing*, vol. 6, no. 2, pp. 328–332, 1997.

[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] P. Arbeláez, B. Hariharan, C. Gu, S. Gupta, L. Bourdev, and J. Malik, “Semantic segmentation using regions and parts,” in *CVPR*. IEEE, 2012, pp. 3378–3385.

[9] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, “Semantic image segmentation with deep convolutional nets and fully connected crfs,” *CoRR*, vol. abs/1412.7062, 2014.

[10] Y. Deng, B. S. Manjunath, and H. Shin, “Color image segmentation,” in *CVPR*, vol. 2. IEEE, 1999.

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

[12] J. P. Alvarado Moya, “Segmentation of color images for interactive 3d object retrieval,” Ph.D. dissertation, Bibliothek der RWTH Aachen, 2004.

[13] M. M. Cheng, J. Warrell, W. Y. Lin, S. Zheng, V. Vineet, and N. Crook, “Efficient salient region detection with soft image abstraction,” in *IEEE ICCV*, 2013, pp. 1529–1536.

[14] H. Jégou, M. Douze, and C. Schmid, “Hamming Embedding and Weak Geometry Consistency for Large Scale Image Search - extended version,” Research Report 6709, Oct. 2008.

[15] H. Shao, T. Svoboda, and L. Van Gool, “Zubud-zurich buildings database for image based recognition,” *Computer Vision Lab, Swiss Federal Institute of Technology, Switzerland, Tech. Rep*, vol. 260, 2003.

[16] G. Schaefer and M. Stich, “Ucid: an uncompressed color image database,” in *Electronic Imaging 2004*. International Society for Optics and Photonics, 2003, pp. 472–480.

[17] D. Nister and H. Stewenius, “Scalable recognition with a vocabulary tree,” in *CVPR*, vol. 2. IEEE, 2006, pp. 2161–2168.

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[19] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *CVPR*, 2014, pp. 580–587.

[20] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, “Caffe: Convolutional architecture for fast feature embedding,” *arXiv preprint arXiv:1408.5093*, 2014.

[21] X. Liu, J. He, and B. Lang, “Reciprocal hash tables for nearest neighbor search.” in *AAAI*, 2013.

[22] X. Liu, L. Huang, C. Deng, J. Lu, and B. Lang, “Multi-view complementary hash tables for nearest neighbor search,” in *Proceedings of ICCV*, 2015, pp. 1107–1115.

[23] X. Liu, X. Fan, C. Deng, Z. Li, H. Su, and D. Tao, “Multilinear hyperplane hashing,” in *Proceedings of CVPR*, 2016, pp. 5119–5127.