PAPER

The Improvement of the Processes of a Class of Graph-Cut-Based Image Segmentation Algorithms

Shengxiao NIU[†], ***Nonmember and*** **Gengsheng CHEN**[†a)], ***Member***

SUMMARY In this paper, an analysis of the basic process of a class of interactive-graph-cut-based image segmentation algorithms indicates that it is unnecessary to construct n-links for all adjacent pixel nodes of an image before calculating the maximum flow and the minimal cuts. There are many pixel nodes for which it is not necessary to construct n-links at all. Based on this, we propose a new algorithm for the dynamic construction of all necessary n-links that connect the pixel nodes explored by the maximum flow algorithm. These n-links are constructed dynamically and without redundancy during the process of calculating the maximum flow. The Berkeley segmentation dataset benchmark is used to prove that this method can reduce the average running time of segmentation algorithms on the premise of correct segmentation results. This improvement can also be applied to any segmentation algorithm based on graph cuts.

key words: *graph cut, image segmentation, energy function, maximum flow*

1. Introduction

Interactive image segmentation is a means of quickly separating objects from the background of an image using simple user inputs. This technology has been widely used in many computer vision tasks, such as image editing, image analysis, and image recognition. In this technology, the most widely used method is the graph-cut-based segmentation algorithm.

The first interactive-graph-cut-based image segmentation algorithm was presented in 2001 [1]. Over the past fifteen years, several upgraded versions based on it have appeared in academia. These algorithms realize many improvements in many different ways. Some make progress by modifying the energy function, such as adding shape priors [2]–[5] or texture priors [6]–[8] to the energy function to render segmentation more accurate. Some help decrease the runtime of the algorithm. For example, the lazy snapping algorithm causes segmentation with super pixels provided by watershed segmentation instead of image pixels to reduce the computing time required to solve for maximum flow [9], and the multilevel banded segmentation algorithm speeds up the runtime by separating a smaller version of the original image first and then the original version using previous segmentation results [10]. Some use different forms of user input to simplify the interactive complexity, such as Grabcut algorithm [11], which separates the foreground from the background using only a bounding box. Some methods improve the maximum flow algorithm [12], [13] and greatly increase the speed of solving for maximum flow in vision applications.

All of the above-mentioned graph-cut-based algorithms can be summed up in the following basic steps: designing an energy function, constructing t-links for every image pixel (or super pixel) node and constructing n-links for every pair of adjacent pixel nodes based on user input and the energy function to build a graph, and using a maximum flow algorithm to solve for maximum flow and minimal cuts to map to the final segmentation results [1].

These basic steps were carefully researched, and the time required to construct all the n-links before calculating the maximum flow when the energy function is complicated was found to be excessive, and none of the graph-cut-based segmentation algorithms can skip this step. However, an analysis of the maximum flow algorithm in the last step shows that only a few of these n-links are actually used in the maximum flow algorithm, so is not necessary to construct a large number of n-links. Our Improvement lies in delaying the n-link construction operation, which is implemented dynamically for the pixel nodes explored by the maximum flow algorithm during the process of solving for the maximum flow. This improvement not only reduces the runtime but is also relatively flexible. It can be applied to any graph-cut-based segmentation algorithm.

2. Analysis of the Maximum Flow Algorithm Used in Graph-Cut-Based Segmentation Algorithms

So far, the most common maximum flow algorithm used in graph-cut-based segmentation algorithms has been the BK algorithm [12]. However, in 2011, Goldberg presented the IBFS algorithm [13], improving upon the BK algorithm. These two algorithms are based on augmenting path and will be analyzed as examples in the following section.

Augmenting path can be defined as follows: During the process of calculating the maximum flow, in each iteration, the algorithm tries to find an s-t path that consists of unsaturated edges in a graph. If a path is found, then the algorithm pushes the enough flow which saturates one edge of the path at least through the path. This process can be called augmenting path. Each augmentation process increases the total flow from the source node to the sink node. The maximum total flow is reached when no s-t path that consists of unsaturated edges can be found in the graph [12].

Manuscript received August 15, 2016.
Manuscript publicized September 14, 2016.
[†]The authors are with State Key Lab of ASIC and System, Fudan University, Shanghai 201203, China.
a) E-mail: gschen@fudan.edu.cn
DOI: 10.1587/transinf.2016EDP7347

2.1 Analysis of the Initial Paths in the Maximum Flow Algorithm

All maximum flow algorithms based on augmenting path have the same pattern: they first search for paths and then augment them. According to previous works, in a graph built with a 2D image, every pixel node has two t-links, one connecting to the source terminal node, and the other to the sink terminal node, and the weight of each t-link is calculated using the regional term of the energy function in the Appendix. The paths to be augmented first are between the source terminal node and the sink terminal node through the pixel nodes whose t-links connect two of them, and these are also initially the shortest paths for the maximum flow algorithm to search for. The length of these paths is 2 and the number of paths is the same as the number of pixel nodes [1]. The realization detail of the maximum flow algorithms corresponding to the process of augmenting these initial paths involves calculating the difference of two t-link weights from every pixel node and recording it, then temporarily classifying every pixel node into three categories: source class nodes have differences being greater than 0, sink class nodes have difference less than 0, and undefined class nodes have differences equal to 0.

2.2 Analysis of the Search Start Nodes of Path Search in the Maximum Flow Algorithm

After calculating the difference of two t-link weights from every pixel node, in the BK algorithm, the pixel nodes whose t-link difference is not equal to 0 are stored in an active queue, and, in the IBFS algorithm, two kinds of pixel nodes whose t-link differences are greater than 0 and less than 0 are stored in the source queue and sink queue, respectively. The pixel nodes in active queues serve as start nodes from which remaining augmenting paths are searched for by both algorithms. To prevent overlooking a path to be augmented, these two algorithms take all pixel nodes whose t-link difference is not equal to 0 as search start nodes. In most cases, in a graph, the weights of edges are calculated using the energy function, and the t-link difference of every pixel node is not equal to 0. So the number of these start nodes remains mostly the same as that of the pixel nodes.

Referring to the BK algorithm, we divide all pixel nodes into two categories, source nodes, and sink nodes. The pixel nodes are classified as source nodes and sink nodes according to whether or not their t-link difference is greater than 0 or less than 0, respectively. If the t-link differences of the pixel nodes are equal to 0, we classify them as source nodes by default. Both the BK and IBFS algorithms use initial search strategies such that each path search start from every pixel node. However, this process is very inefficient, which may be attributable to the fact that, at the beginning of the path search process, many searches are found to be invalid because they produce no path to augmentation owing to the large number of pixel nodes in the same category as their adjacent nodes. As demonstrated in the following section, we can prove that all paths to augment must cross the border between the area of source nodes and the area of sink nodes (hereinafter referred to as the ST border) in a graph. The path search process begins with only those pixel nodes located at the ST border and can also finish the calculation of the maximum flow without omitting any path to augment. This substantially reduces the number of invalid searches.

Theorem 1: In graph-cut-based image segmentation algorithms, when the maximum flow is calculated, all paths to augment must cross the ST border.

Proof: Any path to augment must include at least one source node and one sink node to ensure that flow would move from the source terminal through the t-link of a source node, other edges in the path, and the t-link of a sink node to the sink terminal. The paths to augment meeting the conditions given above must cross the ST border.

Theorem 2: In graph-cut-based image segmentation algorithms, any augmentation-path-based maximum flow algorithm whose path search begins with those pixel nodes located at ST border can also be used to solve for maximum flow correctly.

Proof: According to Theorem 1, all paths to augment cross the ST border. Path searches beginning with those pixel nodes located at the ST border by breadth first search (BFS) or depth first search (DFS) can also find all paths to augment node-by-node, and the maximum flow algorithms based on Ford-Fulkerson method [15] can ensure that the maximum flow can be solved correctly so long as all paths to augment are found. Some experiments were performed to prove Theorem 2.

3. Details of Implementation

Theorem 2 is the foundation of the new algorithm proposed in this paper. If only those pixel nodes located at the ST border are taken as search start nodes, there is no need to construct n-links for every pixel node in advance, as in previous works [1]. Instead, only the necessary n-links are constructed, dynamically, during the process of path search, and others are never constructed. In other words, it is totally unnecessary to construct n-links connecting pixel nodes not explored during path search. So the maximum flow algorithm does not need to calculate the boundary terms of the energy function for pixel nodes not explored during path search, and can avoid the cost of the boundary term calculation. The following experiments prove there to be a large number of pixel nodes for which it is unnecessary to construct n-links.

Our proposed improvements on the process of conventional graph-cut-based image segmentation algorithms are described in detail as follows.

3.1 Graph Initialization without N-Links

In conventional graph-cut-based image segmentation

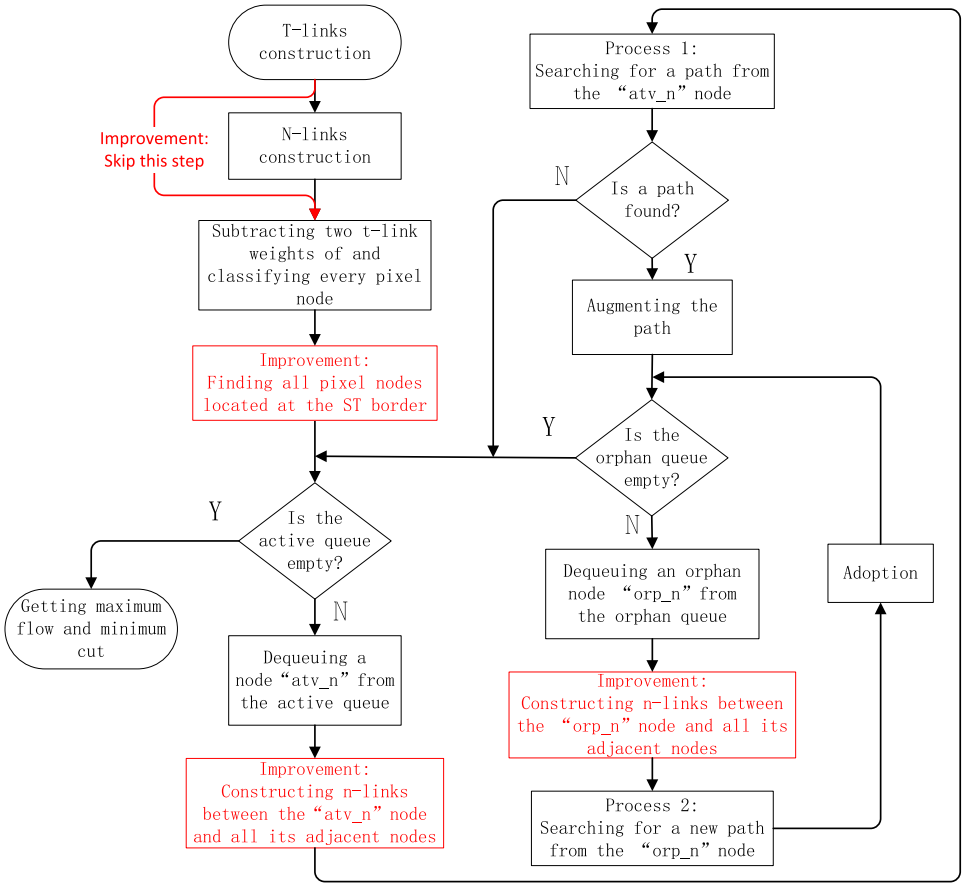

Fig. 1 The BK algorithm flow chart. The steps containing the key word "Improvement" in the chart are our improvements to the original BK algorithm.

algorithms, two t-links from every pixel node are constructed using the regional term of the energy function and two n-links from every pair of adjacent pixel nodes are constructed using the boundary term of the energy function. However, in the improved algorithm, as shown in Fig. 1, only t-links are constructed without constructing any n-links before the calculation of the maximum flow.

3.2 ST-Border Pixel Search

In conventional graph-cut-based image segmentation algorithms, at the beginning of the calculation of the maximum flow, the two t-link weights from every pixel node are subtracted and the difference is saved. Then the pixel nodes whose t-link difference is greater than 0 are classified as source nodes and those with differences less than 0 are classified to sink nodes, and the pixel nodes whose t-link differences are unequal to 0 serve as the search start nodes. However, in our algorithm, after classification (the pixel nodes whose t-link difference is equal to 0 are classified as source nodes by default), an extra graph-rescanning operation is launched to identify all pixel nodes located at the ST border, which are exclusively employed as search start nodes. Theorem 2 is found to ensure that this modification can also find all paths to augment and solve for maximum flow correctly.

As shown in Fig. 1, the implementation detail is that after t-link weights are subtracted and all pixel nodes are classified, all adjacent nodes from every pixel node are searched, and, if any one adjacent node is not in the same category as the center pixel node, the center pixel node will be considered to be located at the ST border, and all pixel nodes found in this way will be pushed into active queues. These pixel nodes are the search start nodes in the improved algorithm and the number of them is far smaller than that of all pixel nodes.

3.3 Flow Augmentation with Dynamic N-Link Construction

Once a pixel node is searched for or used by the algorithm such as a node dequeued from a queue, it is considered "explored", and others are considered "unexplored." The only time that an n-link needed to be used is when the pixel node connected by the n-link is explored. We use BK algorithm as example. Figure 1 summarizes the workflow of BK algorithm. In each flow augmentation iteration, there are two processes where all adjacent nodes of an explored pixel node must be searched and corresponding n-links must be used. One is "path search" process where a new node is dequeued

and taken as a start node from the active queue and BK algorithm searches a path for augmentation from the new node. The other is "orphan adoption" process where isolated subtrees caused by edge saturation of flow augmentation are connected to valid search trees. Figure 1 highlights the process 1 and process 2. In only these two processes, BK algorithm explores any unexplored pixel node and the n-link constructed between the explored and unexplored pixel nodes is used.

The improvements are made by constructing n-links for every explored pixel node before the algorithm searches any unexplored pixel node adjacent to the explored pixel node in those two processes. Every n-link only can be constructed once and used immediately after construction, and the algorithm will determine whether an n-link has been constructed before constructing a new n-link between two pixel nodes. This way can ensure that an n-link connecting to an unexplored pixel node has already been constructed when the algorithm explores the unexplored pixel node and needs to use the n-link, and no n-link connecting any unexplored pixel node is constructed if the algorithm does not explore the unexplored pixel node. As shown in Fig. 1, in BK algorithm flow, the n-links between the explored node "atv_n" dequeued from the active queue and all its adjacent unexplored nodes are constructed before Step "Process 1" and the n-links between the explored node "orp_n" dequeued from the orphan queue and all its adjacent unexplored nodes are constructed before Step "Process 2".

The above-mentioned technique of "two processes" can apply not only to BK algorithm, but also to all BK-style maximum flow algorithms with "path search" and "orphan adoption" processes such as IBFS algorithm. In addition, if another style maximum flow algorithm has to be used, those key processes where n-links need to be used similar to the "two processes" in BK algorithm should be redefined according to the specific case.

4. Experiments

In this paper, all algorithms were realized using the C++ programming language and tested using the following platform: Ubuntu14.04 operation system, Intel core i5-2400 cpu. To prove that the improved algorithm can be used with any kind of graph-cut-based image segmentation algorithm, three combinatorial algorithms were designed and the improved algorithm was embedded into these three algorithms for testing. These three algorithms use different interactive methods (brush and bounding rectangle), different energy functions (lazy snapping [9] and Grabcut [11]), and different maximum flow algorithms (BK and IBFS). In the maximum flow algorithms, the image pixel was used directly instead of the super pixel.

- Combination 1: Brush interactive method + lazy snapping energy function (λ = 50) + BK algorithm
- Combination 2: Brush interactive method + lazy snapping energy function (λ = 50) + IBFS algorithm
- Combination 3: Bounding rectangle interactive method + Grabcut energy function (iteration once) + BK algorithm

Here, 500 images from the Berkeley Segmentation Data Set (BSDS500) are used as the test cases [14], and foreground strokes and bounding rectangles were drawn on these 500 images with a mouse. Four border lines per image served as the background strokes.

Table 1 Pixel-by-pixel comparison of the segmentation results of the original and improved algorithms

Test image	Combinational algorithm	Comparison of results
BSDS 500	Combination 1	Same in pixel (500 results)
	Combination 2	Same in pixel (500 results)
	Combination 3	Same in pixel (500 results)

Table 2 Statistics of the proportion of the pixel nodes connected by constructed n-links in the improved combinational algorithms

Test image	Combinational algorithm	n/P (avg)
BSDS 500	Combination 1	0.4132
	Combination 2	0.4115
	Combination 3	0.2583

4.1 Correctness Proof

To further prove the correctness of Theorem 2 and the segmentation results of the improved algorithm, the original versions of the three combinational algorithms and the improved versions containing the improved algorithm were tested, and the segmentation results (corresponding to the minimal cuts) were compared pixel by pixel to confirm whether the improved algorithms produced the same segmentations as the original ones.

As shown in Table 1, in all experiments, the improved versions provided equivalent results to those of the original versions, which also proves the correctness of Theorem 2.

4.2 Statistics of N-Link Construction in the Improved Algorithms in Calculating the Maximum Flow

We determined the percentage of the pixel nodes connected by constructed n-links for the three improved combinational algorithms. The number of pixel nodes connected by constructed n-links is here expressed in terms of "n" and the number of all pixel nodes in terms of "P." Three average values of the variable n/P were determined in the experiments concerning the 500 images using the three improved combinational algorithms.

As shown in Table 2, an average of only 40% of the pixel nodes were connected by constructed n-links in every image of BSDS500 in the experiments involving Combination 1 and Combination 2, and there was an average of 25% pixel nodes for Combination 3. And, as shown in Fig. 2, in the experiments involving Combination 1 and Combination 2, there are one third of the 500 images whose n/P value

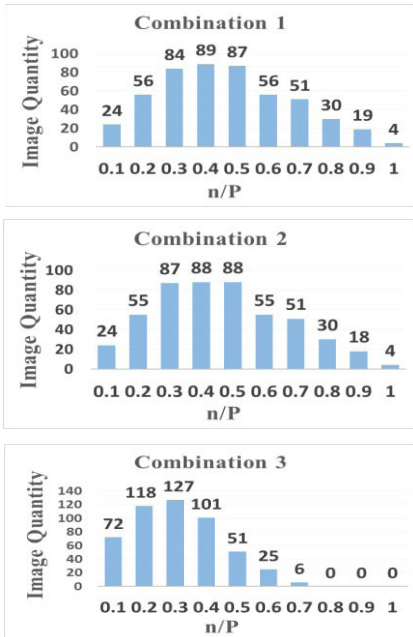

Fig. 2 Statistics of the number of images of different n/P values in the three improved combinational algorithms. Each column shows for the quantity of the images whose n/P value is in the range of two n/P values of the current and previous column.

Table 3 Statistics of the average speedups of the running time of the three combinational algorithms before and after improvement

Test image	Combinational algorithm	Speedup (avg/all images)	Speedup (avg/images of n/P<0.3)	Speedup (avg/images of n/P<0.2)
BSDS 500	Combination 1	1.147	1.252	1.307
	Combination 2	1.151	1.382	1.473
	Combination 3	1.356	1.486	1.680

is less than 0.3, and in the experiment involving Combination 3, there are more than half of the 500 images meeting the same condition. This indicates that many of the n-links are not needed.

4.3 Comparison of the Speedups

The runtime of the three combinational algorithms was tested before and after improvement and all speedups were recorded. As shown in Table 3, the average speedup of each combinational algorithm is greater than one. This indicates that the proposed improved algorithm can be applied to many different interactive image segmentation algorithms and decrease their runtime. As shown in Fig. 3, the lower the n/P value, the more pronounced the increase in speed in each experiment. In this way, the proposed improved algorithm is more applicable to situation where the value of n/P is low.

4.4 Other Features of the Proposed Improved Algorithm

In addition, two extra experiments for Combination 1 algorithm were designed: the first involves repeating the experiments involving Combination 1 in Sect. 4.3 with the 500 images of BSDS500 expanded by a factor of four using bilinear interpolation method, but with no change in the relative position of user marks, and the results are listed in line 3 of Table 4. The other experiment involves repeating that process while substituting the expression containing an exponential function $\lambda \cdot |x_i - x_j| \cdot e^{-\beta\|C(i)-C(j)\|^2}$ [11] for the original boundary term of the energy function of the lazy snapping algorithm in Combination 1 because the expression has showed greater computational cost than the original, and similar segmentation results were obtained with $\lambda = 10$ and $\beta = 0.03$, and the results are given in line 4 of Table 4. The parameter λ is the same as the one in the Appendix. $|x_i - x_j|$ is the difference of the binary labels of

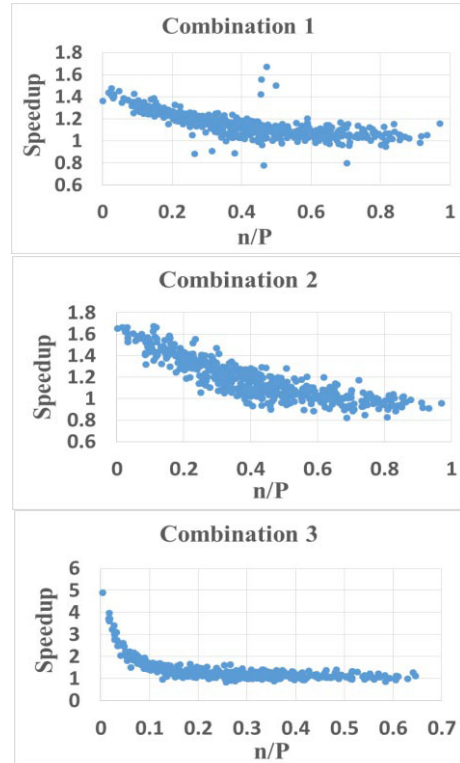

Fig. 3 The speedups of the running time of the three combinational algorithms before and after improvement versus the n/P values of each image in the experiments with the three improved combinational algorithms. Every point in each graph represents the result of one test of one image from BSDS500.

Table 4 Statistics of the average speedups under conditions of different ranges of the n/P value in the extra experiments of combination 1

Test image	Speedup n/P<0.2	Speedup n/P<0.4	Speedup n/P<0.8	Speedup n/P≤1
BSDS500	1.307	1.211	1.152	1.147
BSDS500 (4x)	1.287	1.209	1.156	1.151
BSDS500 (B term change)	1.479	1.306	1.198	1.190

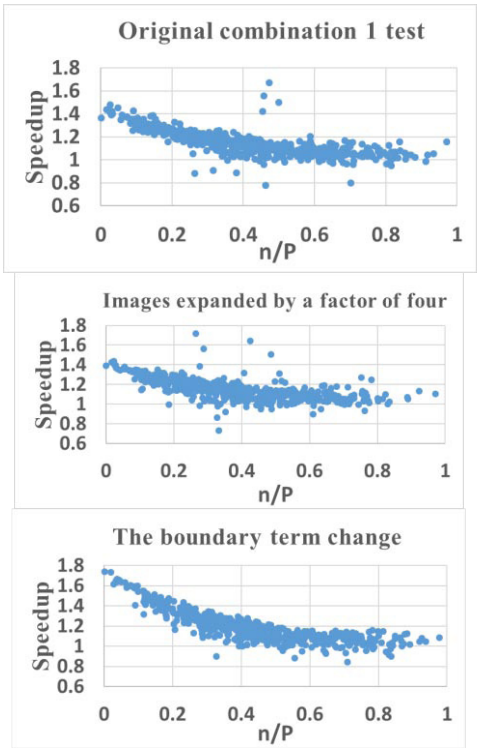

Fig. 4 The speedups of the running time of the algorithms before and after improvement in the three extra experiments of Combination 1 versus the n/P values of each image in the experiments with the improved combination 1 algorithm. Every point in each graph represents the result of one test of one image from BSDS500.

two adjacent pixels i and j. $\|C(i) - C(j)\|^2$ is the Euclidean distance of two adjacent pixels i and j in color space and the parameter β is an empirical constant [11]. The results of the original Combination 1 experiment in Section C are recorded in line 2 of Table 4.

As shown in Table 4 and Fig. 4, although the resolution of the test image was increased by a factor of four, the speedup of the improved Combination 1 algorithm was nearly unaffected. This indicates that the proposed improved algorithm is not affected by the image resolution. Besides, when the boundary term of the energy function was replaced with an expression containing a higher computational cost function, the speedup of the improved algorithm increased. It can be concluded that the more complicated the energy function, the more pronounced the increase in speed of our proposed algorithm. At present, many other improved graph-cut-based image segmentation algorithms have more complicated energy functions where many other priors are added, and that can render our proposed improved algorithm more useful.

4.5 Time Complexity

The construction cost of n-links in the graph-cut-based segmentation algorithm is O(n) where n is the pixel number of the whole image. What we improved in our proposed algorithm is reducing the construction cost of n-links. But this improvement does not change the worst-case running time complexity for the construction of n-links. So the construction cost of n-links is still O(n) after improvement.

5. Conclusion

After an analysis of the basic process of conventional graph-cut-based image segmentation algorithms, many of the n-links constructed in conventional algorithms were found to be redundant. Delaying n-link-construction until the process of calculating the maximum flow and constructing all necessary n-links dynamically was found to eliminate many of the redundant n-links constructed and reduce the runtime of the segmentation algorithms.

This improvement not only reduces the runtime of the algorithm but can also be applied to any graph-cut-based image segmentation algorithm without the influence of image resolution, and, particularly, to those segmentation algorithms whose energy function is complicated and has high computational costs. Meanwhile it is also easy to replace the energy function with another using some programming skills such as function templates.

In the future work, besides researching how to reduce the redundant n-links construction, we will further research how to reduce the redundant t-links construction. This will make our algorithm more efficient.

References

[1] Y.Y. Boykov and M.-P. Jolly, "Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images," Proc. IEEE Int. Conf. Computer Vision, pp.105–112, 2001.
[2] D. Freedman and T. Zhang, "Interactive graph cut based segmentation with shape priors," Proc. IEEE Computer Society. Conf. Computer Vision and Pattern Recognition, vol.1, pp.755–762, 2005.
[3] O. Veksler, "Star shape prior for graph-cut image segmentation," Proc. European Conf. Computer Vision, pp.454–467, 2008.
[4] V. Gulshan, C. Rother, A. Criminisi, A. Blake, and A. Zisserman, "Geodesic star convexity for interactive image segmentation," Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), pp.3129–3136, 2010.
[5] L. Gorelick, O. Veksler, Y. Boykov, and C. Nieuwenhuis, "Convexity Shape Prior for Segmentation," Proc. European Conf. Computer Vision, pp.675–690, 2014.
[6] J. Malcolm, Y. Rathi, and A. Tannenbaum, "A graph cut approach to image segmentation in tensor space," Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), pp.1–8, 2007.
[7] S. Han, W. Tao, D. Wang, X.-C. Tai, and X. Wu, "Image segmentation based on GrabCut framework integrating multiscale nonlinear structure tensor," IEEE Trans. Image Process., vol.18, no.10, pp.2289–2302, 2009.
[8] H. Zhou, J. Zheng, and L. Wei, "Texture aware image segmentation using graph cuts and active contours," J. Pattern Recognition, vol.46, no.6, pp.1719–1733, 2013.
[9] Y. Li, J. Sun, C.-K. Tang, and H.-Y. Shum, "Lazy snapping," ACM Transactions. Graphics, vol.23, no.3, pp.303–308, 2004.
[10] H. Lombaert, Y. Sun, L. Grady, and C. Xu, "A multilevel banded graph cuts method for fast image segmentation," Proc. IEEE Int. Computer Vision, vol.1, pp.259–265, 2005.
[11] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut: Interactive foreground extraction using iterated graph cuts," ACM Transactions. Graphics, vol.23, no.3, pp.309–314, 2004.

[12] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," IEEE Trans. Pattern Anal. Mach. Intell., vol.26, no.9, pp.1124–1137, 2004.
[13] A.V. Goldberg, S. Hed, H. Kaplan, R.E. Tarjan, and R.F. Werneck, "Maximum flows by incremental breadth-first search," Proc. European Symposium. Conf. Algorithms, pp.457–468, 2011.
[14] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," IEEE Trans. Pattern Anal. Mach. Intell., vol.33, no.5, pp.898–916, 2011.
[15] L.R. Ford and D.R. Fulkerson, Flows in Networks, Princeton Univ. Press, Princeton NJ, 1962.

Appendix: Energy Function in Graph-Cut-Based Segmentation

$$E(A) = R(A) + \lambda \cdot B(A) \tag{A·1}$$

A is a binary vector with each element representing a state of one pixel in the whole image. *E(A)* is an energy function and a specific vector A obtained by minimizing the energy function defines a segmentation in the image.

The coefficient λ in (A·1) is a weight parameter which represents a relative importance of B(A) and R(A). The regional term R(A) describes the degrees which a pixel belongs to "object" and "background". The boundary term B(A) describes a penalty for a discontinuity of the state between two adjacent pixels [1].

Shengxiao Niu received M.S. degree in Microelectronics and Solid State Electronics from Fudan University, 2011. Now, he is in pursuit of Ph.D. in Computer Vision in the Fudan University. His research interests include region filling, image segmentation, 3d vision and graph cut.

Gengsheng Chen received his B.S. degree in electrical engineering from Shanghai Jiaotong University of China in 1991, his M.S. degree from Fudan University of China in 1994. He is a Senior Research Engineer in the Statekey Lab of ASIC and System, Fudan University, China. He served for Motorola Electronics, Nortel Networks, Siemens Technical Innovation Center and Zarlink Semiconductor between 1994 and 2005. His major research interests include image processing, embedded system, FPGA circuit and system reliability. He has over 30 papers published on international academic conference and journals and he holds three Chinese patents.