Real-Time Salient Object Detection with a Minimum Spanning Tree

Wei-Chih Tu[1], Shengfeng He[2], Qingxiong Yang[3]*, Shao-Yi Chien[1]
[1]Graduate Institute of Electronics Engineering, National Taiwan University
[2]Department of Computer Science, City University of Hong Kong
[3]School of Information Science and Technology, University of Science and Technology of China

Abstract

In this paper, we present a real-time salient object detection system based on the minimum spanning tree. Due to the fact that background regions are typically connected to the image boundaries, salient objects can be extracted by computing the distances to the boundaries. However, measuring the image boundary connectivity efficiently is a challenging problem. Existing methods either rely on superpixel representation to reduce the processing units or approximate the distance transform. Instead, we propose an exact and iteration free solution on a minimum spanning tree. The minimum spanning tree representation of an image inherently reveals the object geometry information in a scene. Meanwhile, it largely reduces the search space of shortest paths, resulting an efficient and high quality distance transform algorithm. We further introduce a boundary dissimilarity measure to compliment the shortage of distance transform for salient object detection. Extensive evaluations show that the proposed algorithm achieves the leading performance compared to the state-of-the-art methods in terms of efficiency and accuracy.

1. Introduction

The goal of salient object detection is to identify the most important objects in a scene. Recently, it has attracted much attention [15, 19, 5, 10, 25, 11, 4] for its wide range of applications such as image retargeting [16, 7], object recognition [22] and image segmentation [9], to name a few. As a preprocessing step, a desirable saliency detection algorithm should be computational efficient for practical usage.

In general, previous works can be categorized into bottom-up or top-down methods. Top-down methods [8, 29] are task-driven and usually require supervised learning with high-level information. Bottom-up methods [1, 26, 5, 28, 12] are usually based on low-level features such as color, gradient or contrast. Most saliency detection methods are bottom-up models owing to the task-free nature and superior computational efficiency over top-down approach.

The background and connectivity priors proposed by Wei *et al*. [26] have been shown to be effective in bottom-up salient object detection. The background prior assumes the image boundaries are mostly background. The connectivity prior describes the fact that background regions are usually large and homogeneous so that pixels in the background can be easily connected to each other. In [26], the boundary patches are set as the background seeds and the saliency of a patch is defined as the shortest-path geodesic distance towards the background seeds.

Many recent approaches [28, 12, 35, 21, 34] achieve state-of-the-art results by extending the background and connectivity priors. In [35], Zhu *et al*. also utilize the geodesic distance and take the spatial layout of image patches into consideration for a more robust boundary measurement. Yang *et al*. [28] compute the saliency value of a region according to its relevance to boundary patches by manifold ranking. Jiang *et al*. [12] formulate the problem as absorbing Markov chain on an image graph model. The boundary nodes are chosen as the absorbing nodes in a Markov chain and the absorbed time is used to measure saliency. In [21], they propose a saliency optimization approach based on Cellular Automata. They also leverage the background prior to compute a global color distinction map as well as a spatial distance map. These two maps are integrated into a simple background-based map for initialization. To achieve feasible complexity, all the above methods rely on superpixel abstraction to reduce the processing units. However, the over-segmentation step usually becomes the processing bottleneck and restricted this type of methods from real-time applications. Lately, Zhang *et al*. [34] use the minimum barrier distance (MBD) [24, 6] for salient object detection. They show that minimum barrier distance is superior to geodesic distance in several numerical evaluation. Instead of processing with superpixels, they demonstrate an efficient pixel-wise raster-scanning algorithm to compute the approximated MBD transform.

We also tackle the problem from the distance trans-

*Correspondence author.

form perspective. Unlike previous works [26, 34] which compute the distance on an image, we instead measure the distance between pixels on a minimum spanning tree (MST) [3, 30, 31]. This tree representation of an image offers us two main advantages over measuring distance in pixel/superpixel level: (i) We show that the minimum spanning tree representation largely reduce the candidate paths from any starting node to the target seeds so that it can effectively benefit the computational efficiency of the distance transform procedure. Specifically, when the minimum barrier distance is used as the distance metric, it takes only 6 comparisons and 2 subtractions per pixel regardless of the number of candidate paths and the number of target seeds. (ii) The structure-aware property of the minimum spanning tree allows the distance measure on a tree able to distinguish objects in a scene. The proposed salient object detection method achieves superior results to the state-of-the-art methods. In the meanwhile, the proposed method takes only 24.6 ms using a single thread CPU for a 300×400 image (including the construction of MST) which offers an ideal choice for real-time oriented applications.

2. Minimum Spanning Tree for an Image

The effectiveness of minimum spanning tree (MST) based image representation has been demonstrated in cost aggregation for stereo matching [30, 31] and edge-preserving filtering [3]. In this section, we briefly review the MST representation as a preliminary knowledge to the proposed method.

By treating an image I as a standard 4-connected, undirected graph (planar graph) with nodes being all the image pixels, and the edges between adjacent pixels being weighted by color/intensity differences (absolute gradients), a minimum spanning tree can be constructed by sequentially removing edges with large weights (Kruskal's algorithm [13]), leaving the remaining edges connecting through all pixels as a tree. More specifically, let s and r be a pair of adjacent pixels, the weight between s and r is

$$\omega(s, r) = \omega(r, s) = |I(s) - I(r)|.^{1} \tag{1}$$

Figure 1 shows a toy example of 4×5 pixels. During the tree construction, if two adjacent pixels have large gradient, the edge between these two pixels is likely to be removed and the path to traverse one pixel to another will be long as shown in Figure 1(c). As a result, the distance on the MST will be large for these two pixels. On the contrary, if two pixels are similar in appearance, they are likely to be connected on the MST and the distance will be short.

For tree construction, Bao *et al*. [3] present a linear time approach tailored for 8-bit images (which may have multiple channels) based on Prim's algorithm [20]. It takes about 0.07 seconds to build a MST using single thread on a modern CPU for a 1-megapixel image. We adopt the same algorithm and it takes in average 5.86 ms in MST construction for a 300×400 image.

[1]For color images, the maximum value computed separately from three channels is used as the weight.

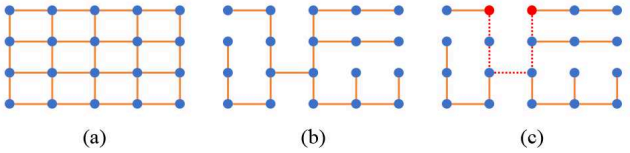


Figure 1: A toy example showing the construction of a minimum spanning tree for an image. (a) A planar graph in which nodes are image pixels and edges are weighted by color/intensity differences between adjacent pixels. (b) The MST is constructed by sequentially removing edges with large weights. (c) The distance of the two red pixels is defined on the tree path (the dashed line).

3. Distance Transform with a Minimum Spanning Tree

Previously, the geodesic distance (GD) and the barrier distance (BD) are used for salient object detection. In this section, we present a novel distance transform based on the MST representation of an image. The proposed algorithm is able to apply to both kinds of distance metrics.

3.1. Basic Definition

The goal of distance transform is to compute a distance map D with respect to a set of seed nodes[2] S. Again, the input image I is treated as a planar graph. Let $\pi = \{\pi(0), ..., \pi(k)\}$ denote a path on the graph. $\pi(i)$ and $\pi(i+1)$ are adjacent pixels on image I. Given a distance metric $f(\pi)$, the distance transform of a node v is

$$D(v) = \min_{\pi \in \Pi_{S,v}} f(\pi), \tag{2}$$

where $\Pi_{S,v}$ denotes the set of all paths connecting v and a seed in S. In general, the distance measure of two nodes u and v has the following properties

$$f(u \to v) = f(v \to u) \tag{3}$$

$$f(u \to v) >= 0 \tag{4}$$

In [26, 35], the geodesic distance is used for salient object detection. The geodesic distance accumulates the distance of all traversed pixel pairs on a path. Formally, the geodesic distance metric $f_{GD}(\pi)$ is defined as

$$f_{GD}(\pi) = \sum_{i=0}^{k-1} |I(\pi(i+1)) - I(\pi(i))|. \tag{5}$$

[2]We use node and pixel interchangeably when we discuss the graph or the image. They are the same in this paper since the proposed algorithm is a pixel-wise algorithm.

In [34], the minimum barrier distance [24, 6] is used for salient object detection. The barrier distance metric $f_{BD}(\pi)$ is defined as

$$f_{BD}(\pi) = \max_{i=0}^{k} I(\pi(i)) - \min_{i=0}^{k} I(\pi(i)). \quad (6)$$

The barrier distance is shown to be more robust than geodesic distance for salient object detection [34]. However, finding the exact minimum barrier distance (MBD) has high complexity of $O(mn \log n)$ [6], where n is the number of pixels in the image and m is the number of distinct values the image contains. Zhang *et al.* [34] proposed an iterative raster scanning approach to approximate the MBD transform instead of exhaustive search by Dijkstra-like algorithms.

3.2. The Proposed Distance Transform

Instead of searching the shortest distance on the image, we propose to find the shortest path on a minimum spanning tree. The proposed MST-based distance transform is an exact solution for distance defined on the tree. It consists of two passes: bottom-up traversal and top-down traversal. Figure 2 gives a simple illustration using geodesic distance. The counterpart using barrier distance is similar. Given a set of seed nodes S, we initialize the distance values of seed nodes to 0 and all other nodes to ∞ as shown in Figure 2(a). Figure 2(b) and (c) demonstrate the bottom-up traversal and the top-down traversal respectively.

For the bottom-up pass, we start from the leaf nodes and update the distance values of their parent nodes by

$$D(p) = \min\{D(p), f(\pi_v \cup p)\}, \quad (7)$$

where p denotes the parent node of v, π_v denotes the current optimal path connecting v to its nearest seed node and $\pi_v \cup p$ denotes the same path plus one step further to reach its parent p. As the nearest seed node to p can come from its bottom or from its top, Eq. 7 tests the possible solution from the bottom. If p has multiple child nodes, Eq. 7 will be evaluated for each child node and the minimum distance is stored. One example is shown in the root node of Figure 2(b). In short, the updating process is conducted in bread first search (BFS) order in each node. The bottom-up traversal continues until it reaches the root node.

The top-down pass is similar while starting from the root node. For each node, we visit its child nodes and update their distance values by

$$D(v) = \min\{D(v), f(\pi_p \cup v)\}. \quad (8)$$

Note that Eq. 8 not only tests for the possible solution from the top, but also propagate potentially better solutions from other branches. One can see the illustration in Figure 2(b). After the bottom-up pass, many nodes still have ∞ distance from seed nodes. The nearest seed node to a certain node may locate at the top or at the branches splitting from the nodes above it. After the bottom-up pass, a splitting node is expected to record the optimal distance value from one of its branches. And in the top-down pass, the optimal solution will be propagated down to other branches as shown in Figure 2(c). This is why we conduct the bottom-up pass first.

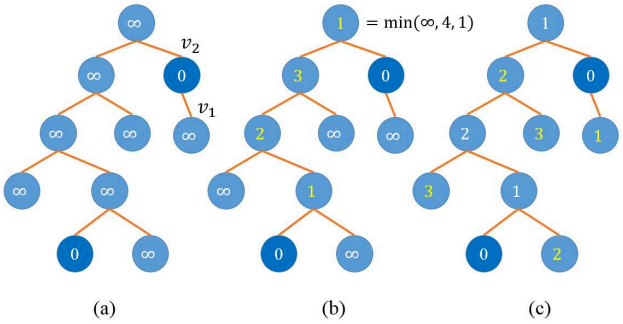


Figure 2: A simple illustration of the proposed two-pass distance transform algorithm. For simplicity, let all edge weights be 1 and assume the geodesic distance is used. (a) Initially, the distance values of seed nodes are set to 0 and ∞ for others. (b) Bottom-up updating. The updated distance is labeled in yellow. (c) Top-down updating.

We can learn more rules with this simple illustration. First, the distance of node v_1 is uniquely determined by seed node v_2. This is based on the fact that traveling across the seed node has the chance to increase the distance for both GD and BD. Therefore v_2 is the optimal seed node for v_1. For GD, traveling one step further add one more absolute gradient term to its distance computation so the updated distance is non-decreasing. For BD, we track the maximum and minimum values for each node. Traversing one more node may update the maximum or minimum values and thus increase the barrier value, so the final distance is also non-decreasing. It implies another observation that travelling across seed nodes or updating the distance of seed nodes would not give better results and thus is unnecessary.

Finally, we summarize the rules as follows:
1. Performing the bottom-up traversal and then the top-down traversal results in the optimal distance transform.
2. If a seed node is the only seed node and is located at the root node of a sub-tree, then the distance transform of the nodes on the sub-tree is uniquely determined by this seed node. The corresponding distance transform will be obtained in the top-down pass.
3. During traversal, we can ignore the updating steps in Eq. 7 and Eq. 8 for seed nodes.

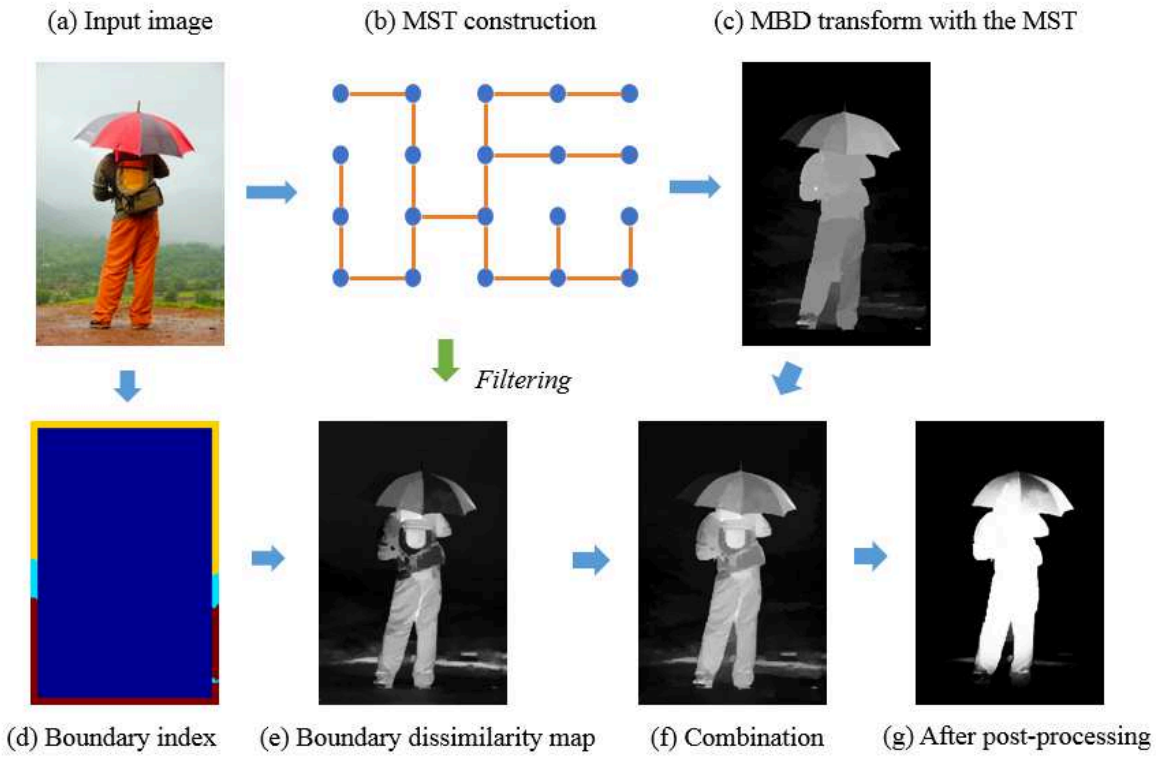


Figure 3: An overview of our MST-based saliency detection framework.

3.3. Complexity Analysis

We estimate the operation count by considering the case that every node performs bottom-up updating in Eq. 7 and top-down updating in Eq. 8 once. In practice, the root node has no parent and the leaf nodes have no child. Moreover, we can ignore the seed nodes in the updating steps, so the estimated operation count is in fact a loose upper bound.

If the geodesic distance is adopted, both Eq. 7 and Eq. 8 require one comparison operation and one addition operation. In total, 2 addition operations and 2 comparison operations are required.

If the barrier distance is adopted, we track the maximum and the minimum values for each node. Each time when a new node is visited, 3 comparison operations are required for bottom-up or top-down pass, including one comparison for the maximum, one for the minimum and one for comparing the optimal distance. Extra subtraction operation is required to compute the barrier distance. As a result, in total 6 comparisons and 2 subtraction operations are required for the minimum barrier distance.

As a result, the distance transform with a MST has constant complexity for each pixel regardless of the distance metric used. With the linear time construction algorithm described in [3], the overall distance transform is also linear in the number of pixels.

4. Salient Object Detection

We describe our salient object detection system in this section. Despite of the distance transform presented in previous section, we introduce another simple yet useful auxiliary map based on appearance similarity measure to compliment the shortage of measuring the boundary connectivity. We further utilize the off-the-shelf MST to apply tree filtering [3] to smooth the map. Finally, we also describe the post-processing in this section. The overall salient object detection system is summarized in Figure 3.

4.1. Measuring the Boundary Connectivity

We set all pixels along the image boundary as a set of seed nodes to exploit the background and connectivity priors for salient object detection. We have tested our MST-based distance transform using both GD and BD. When computing the GD transform, we also account for the internal edge weight clipping step similar to [26]. We compute the average edge weight of all remaining edges on the MST as the clipping threshold. The barrier distance is not based on accumulation so it does not contain this step.

Example results of our MST-based distance transform using GD or BD are shown in Figure 4. As one can see, the BD transform is more robust to texture and has the ability to capture the geometry information better. Thus the BD

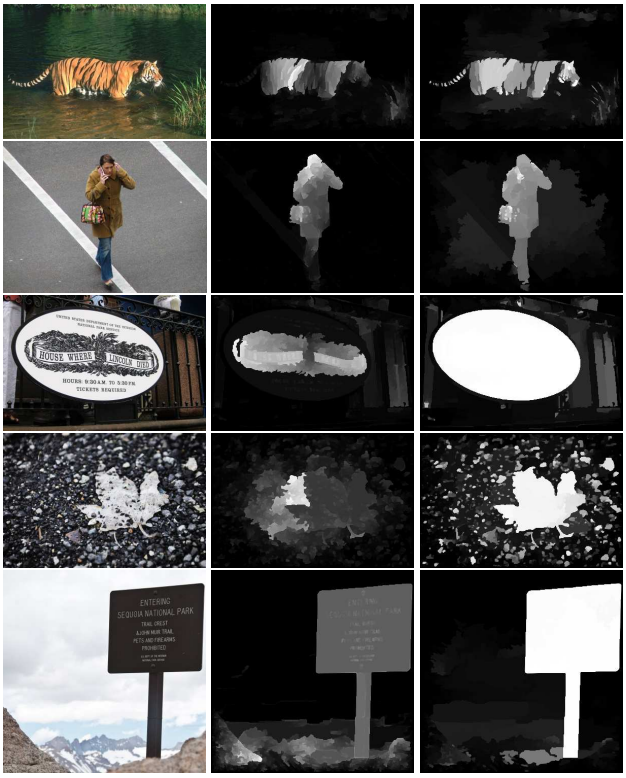


Figure 4: Distance transform result. Boundary pixels are set as seed nodes. From left to right: input images, distance transform using GD, distance transform using BD. The results using BD is usually more robust than GD.

transform is more favoured in salient object detection and we adopt the BD transform in our final experimental results.

4.2. Boundary Color Dissimilarity Measure

The distance map alone sometimes can not produce satisfying results when the background has many cluttered and isolated regions or the salient object touches the image boundary. To compliment the shortage of distance transform, we introduce another auxiliary map computed by pixel-wise color similarity measure to improve the saliency detection quality.

Again, we assume most image boundary pixels are background. First of all, boundary pixels are clustered into K groups by their color values in Lab color space. We have found that 3 clusters is enough and set $K = 3$ for all evaluation. The boundary size between $10 - 30$ pixels produces similar results. We simply set boundary size = 10 pixels. Let n_k be the number of pixels in each group after clustering. For each group, we compute the mean color μ_k and covariance matrix C_k. Then, the pixel-wise color dissimilarity map of boundary group k is computed using the Mahalanobis distance:

$$S_k(i) = \sqrt{(I(i) - \mu_k)C_k^{-1}(I(i) - \mu_k)^T}. \quad (9)$$

S_k is normalized and the final boundary color dissimilarity map S is obtained by weighted sum of S_k maps.

$$S = \frac{\sum_{k=1}^{K} n_k S_k}{\sum_{k=1}^{K} n_k}. \quad (10)$$

Since the map is computed in pixel-wise fashion, sometimes it appears noisy due to image noise or image compression artifact. We utilize the off-the-shelf MST data structure previously built for distance transform. The map is smoothed using tree filtering [3]. Note that the MST data structure is built once and can be used multiple times for distance transform as well as tree filtering. It is our advantage over other distance transform based methods.

Example results are shown in Figure 5. Note that even when the foreground object touches the image boundary, the proposed boundary color dissimilarity measure is able to distinct the object from the background. This is due to the boundary clustering step and weighted sum scheme. Even the foreground touches the image border, usually the majority part of boundary pixels are still background. The preliminary clustering step separates boundary pixels that have distinctive color appearance. The potential foreground pixel group is usually small, so after the weighted summation step, the final boundary dissimilarity map can still reveal the foregroundness of a scene.

4.3. Post-processing

The post-processing is similar to [34]. The distance map D and the boundary dissimilarity map S are added together to form an intermediate map M. This map accounts for the object geometry information from the distance transform as well as the global distinctiveness of appearance from the boundary color dissimilarity map.

We further apply a pixel location dependent masking to emphasize the photographic bias that photographers tend to locate important objects ta the center of the image. Let the image resolution be $H \times W$. This mask is simply a two-dimensional Gaussian fall-off with variance $\frac{H}{3}$ and $\frac{W}{3}$ respectively. The masked intermediate map is further normalized so that the maximum value is 1.

Finally, we apply an adaptive contrast enhancement with the following sigmoid mapping:

$$g(x) = \frac{1}{1 + exp(-\gamma(x - \tau))}, \quad (11)$$

where τ is an adaptive threshold obtained using Otsu's binary threshold method [18] and γ controls the overall sharpness. We set $\gamma = 20$ in our experiments.

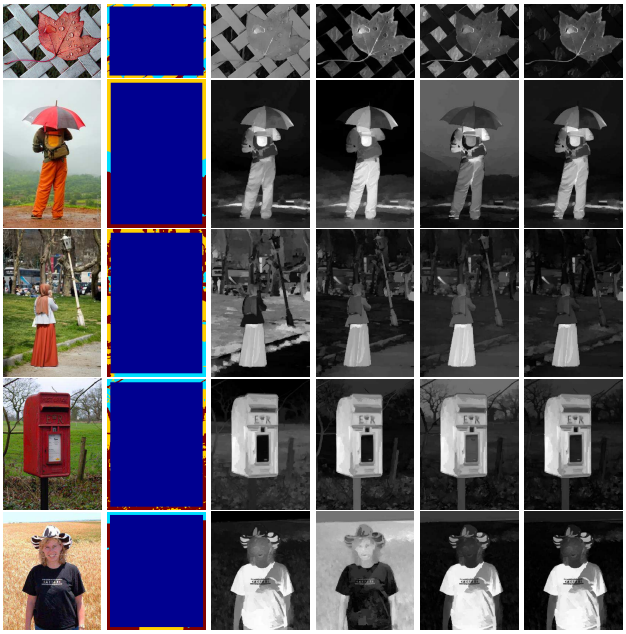


Figure 5: Examples of color dissimilarity map. From left to right: input images, boundary clustering results, color dissimilarity maps against boundary group $1-3$, and final boundary dissimilarity maps.

5. Experimental Results

We evaluate the proposed method on three datasets. The first one is the MSRA-1000 dataset [1], which contains 1000 images. The second one is the ECSSD dataset [23], which also contains 1000 images. Moreover, it has many semantically meaningful yet structurally complicated images. The last one is the PASCAL-S [14] dataset, which contains 850 natural images with complex background. The PASCAL-S dataset is designed to avoid the dataset design bias and is the most challenging among these three datasets. These datasets all have accurate human-labelled masks for salient objects.

We compare our method with twelve classic or state-of-the-art saliency detection algorithms. They are BL [25], MB+ [34], SO [35], BMS [32], HS [27], RC [4], GC [5], AMC [12], MR [28], GS [26], SF [19] and FT [1] methods. Among them, GS, SO, MB+ use distance transform and BMS is closely related to the minimum barrier distance [33]. The saliency maps of different methods are provided by authors or obtained from available software. Note that we use the implementation of SF provided by the author of [35].

5.1. Processing Time Evaluation

We evaluate the processing time using a 3.6 GHz Core i7-4790 CPU with 8GB memory to process color images

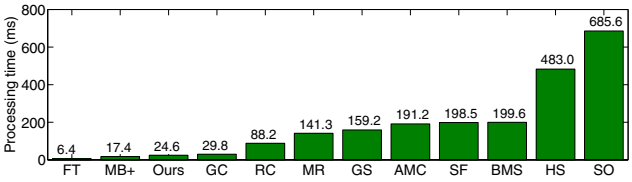


Figure 6: The average time to process a 300×400 color image.

of 300×400 resolution. FT, RC, GC and our method is evaluated with C code. MB+ is evaluated using the software provided by the author. And the rest use MATLAB and C. For those methods using superpixel segmentation as preprocessing, we use the default settings in the provided code. They all use SLIC superpixel [2] in the implementation. The processing time of different methods is sorted and shown in Figure 6. Note that we remove the processing time of BL from the plot due to its higher order. It takes 11.784 seconds to process an image.

Our method achieves real-time detection (more than 30 FPS). It takes in average 24.6 ms to process an image. The MST construction, tree filtering and distance transform presented in this paper all have complexity linear in the number of pixels. As we discussed earlier, the MST is constructed once and can be used for different purposes many times. For more detail, it takes 5.86 ms for MST construction using Prim's algorithm, 1.12 ms for filtering a single channel image and 1.75 ms for single channel MBD transform. With low complexity, the proposed MST-based distance transform along with other MST-based algorithms provides an ideal tool for applications with speed requirement.

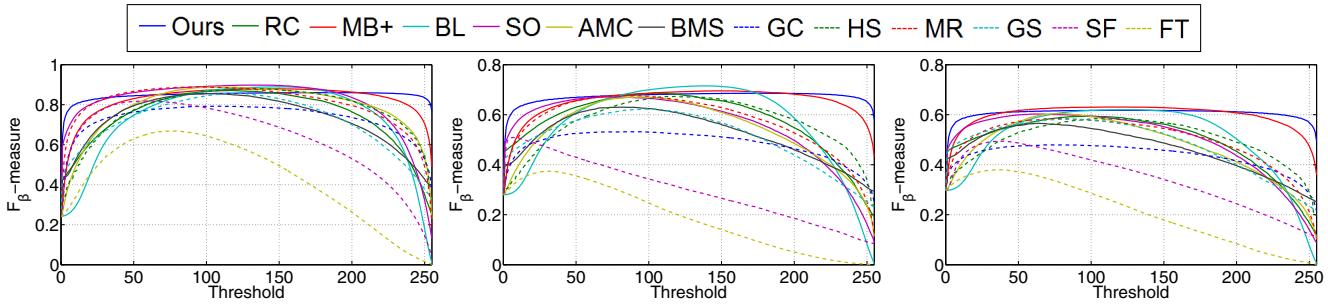
5.2. Quantitative and Qualitative Evaluations

We compute the precision and recall scores by binarizing the saliency maps with a threshold sliding from 0 to 255 and compare the binary maps with ground truth maps. Usually, precision and recall are both important and therefore F-measure is used as the overall performance measure:

$$F_\beta = \frac{(1+\beta^2) \cdot precision \cdot recall}{\beta^2 \cdot precision + recall}. \qquad (12)$$

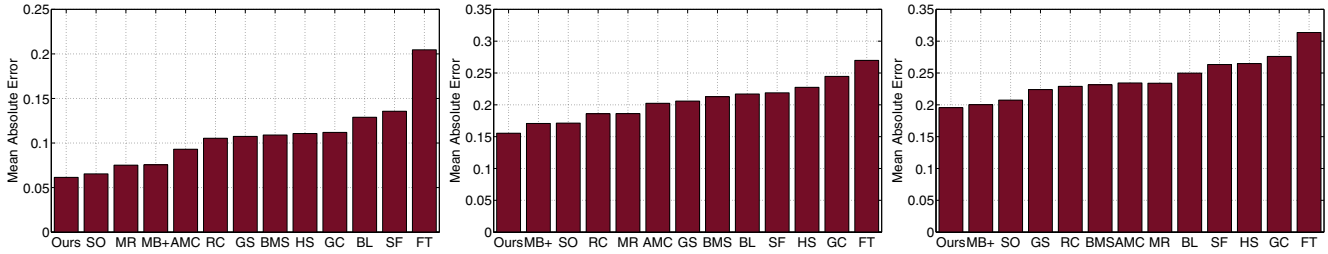
We set $\beta^2 = 0.3$ as suggested in [1] to emphasize the precision. Figure 7 shows the results in three datasets. In our evaluation, no single method outperforms others in all datasets. Nevertheless, one can still see that our method achieves comparable results to the leading methods over all thresholds.

As neither precision nor recall measure consider the true negative saliency assignments, the mean absolute error (MAE) is also introduced as a complementary. The MAE score calculates the average difference between the saliency



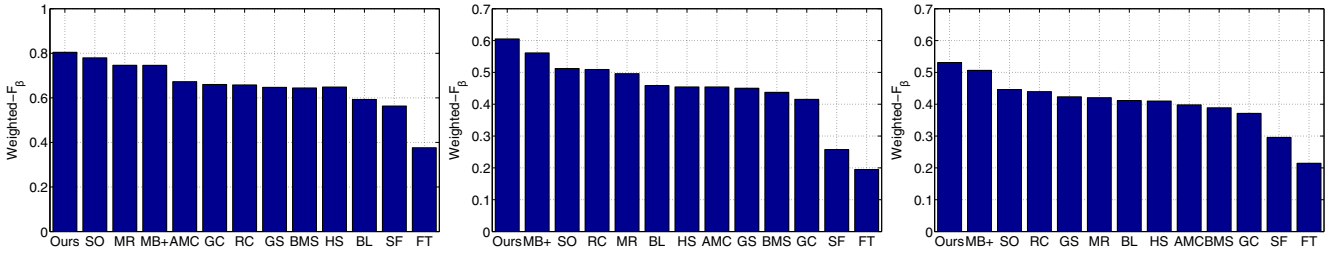
(a) MSRA-1000 (b) ECSSD (c) PASCAL-S

Figure 7: The F_β scores with sliding threshold.



(a) MSRA-1000 (b) ECSSD (c) PASCAL-S

Figure 8: The mean absolute error (MAE) scores.



(a) MSRA-1000 (b) ECSSD (c) PASCAL-S

Figure 9: The weighted-F_β scores.

map M and the ground truth GT:

$$MAE = \frac{1}{H \times W} \sum_{i=1}^{H \times W} |M(i) - GT(i)|. \qquad (13)$$

The MAE scores of compared methods are sorted and shown in Figure 8. Our method has the lowest MAE scores among all compared methods in all datasets.

We also evaluate all methods with the weighted F_β-measure proposed in [17]. The weighted F_β-measure serves as a more reliable metric than previously used metrics like area under the curve (AUC), average precision (AP) or F_β-measure to evaluate foreground maps. We use the code and default parameters provided by the author [17] to evaluate saliency maps. The sorted results are shown in Figure 9. Our method achieves the highest scores than all other competitors in all datasets.

From above evaluation, the proposed method achieves state-of-the-art in terms of efficiency and accuracy. Figure 10 shows some sample saliency maps from three datasets for reference. Our model is able to detect salient objects in the scene with complex or highly textured background.

5.3. Limitations

The background and connectivity priors work because of the photographic biases that photographers tend to locate important objects at the center of the image. If the salient objects touch the image boundary, the proposed method may produce bad results due to bad boundary connectivity measure. As shown in Fig. 11, the boundary color dissimilarity measure is helpful to enhance the final results.

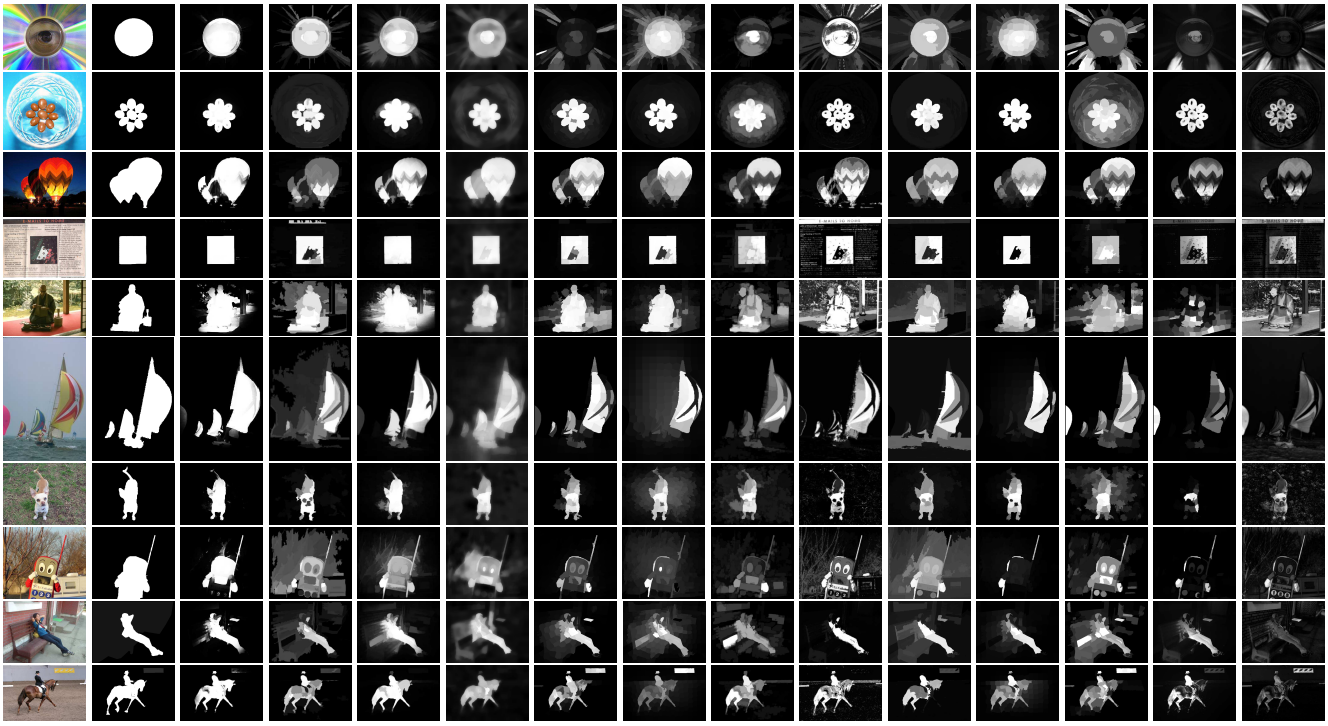


Figure 10: Comparison of our saliency maps with other classic or state-of-the-art methods.

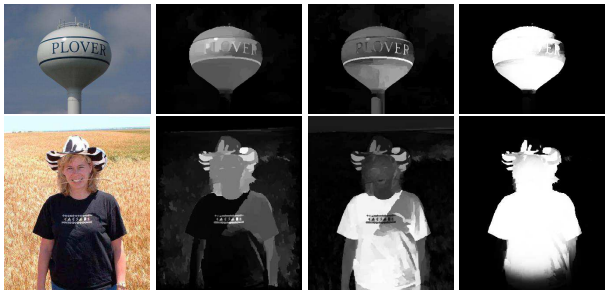


Figure 11: From left to right: input images, distance transform results, boundary color dissimilarity measure, final results.

However, it still cannot fully handle such situation. This limitation can be observed in other background prior based methods as well.

6. Conclusion

In this paper, we present a novel distance transform method using the minimum spanning tree representation of an image. Instead of finding the shortest paths on the image using Dijkstra-like algorithms, we compute distance on the tree paths. The MST structure largely reduces the search space of shortest paths. We show that the MST-based distance transform has constant complexity for each pixel using either geodesic distance or barrier distance (the overall complexity is linear in the number of pixels). Together with previously proposed linear time MST construction and tree filtering. The family of MST-based algorithms is an ideal choice for applications with real-time demand.

We apply the proposed distance transform to measure boundary connectivity for salient object detection. The proposed salient object detection is evaluated and compared with state-of-the-art algorithms and achieves comparable or better results in numerical evaluation. Moreover, the proposed method runs at over 30 FPS. In summary, the proposed method is an accurate and efficient algorithm with the built-in MST structure being our powerful advantage over other distance transform based algorithms.

Acknowledgements. This work was partially supported by the Early Career Scheme through the Research Grants Council of Hong Kong (Project Number: CityU 21201914) and a grant from Qualcomm Technologies, Inc. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Tesla K40 GPU used for this research. S.-Y Chien and W.-C Tu are supported in part by Himax Technologies, Inc. and the Ministry of Science and Technology of Taiwan under Grant MOST104-3115-E-002-002.

References

[1] R. Achanta, S. Hemami, F. Estrada, and S. Susstrunk. Frequency-tuned salient region detection. In *CVPR*, 2009.

[2] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *TPAMI*, 34(11):2274–2282, 2012.

[3] L. Bao, Y. Song, Q. Yang, H. Yuan, and G. Wang. Tree filtering: Efficient structure-preserving smoothing with a minimum spanning tree. *TIP*, 23(2):555–569, 2014.

[4] M.-M. Cheng, N. J. Mitra, X. Huang, P. H. S. Torr, and S.-M. Hu. Global contrast based salient region detection. *TPAMI*, 37(3):569–582, 2015.

[5] M.-M. Cheng, J. Warrell, W.-Y. Lin, S. Zheng, V. Vineet, and N. Crook. Efficient salient region detection with soft image abstraction. In *ICCV*, 2013.

[6] K. C. Ciesielski, R. Strand, F. Malmberg, and P. K. Saha. Efficient algorithm for finding the exact minimum barrier distance. *CVIU*, 123:53–64, 2014.

[7] Y. Ding, J. Xiao, and J. Yu. Importance filtering for image retargeting. In *CVPR*, 2011.

[8] D. Gao, S. Han, and N. Vasconcelos. Discriminant saliency, the detection of suspicious coincidences, and applications to visual recognition. *TPAMI*, 31(6):989–1005, 2009.

[9] S. Goferman, L. Zelnik-Manor, and A. Tal. Context-aware saliency detection. *TPAMI*, 34(10):1915–1926, 2012.

[10] S. He and R. W. Lau. Saliency detection with flash and no-flash image pairs. In *ECCV*, 2014.

[11] S. He, R. W. Lau, W. Liu, Z. Huang, and Q. Yang. Supercnn: A superpixelwise convolutional neural network for salient object detection. *IJCV*, 115(3):330–344, 2015.

[12] B. Jiang, L. Zhang, H. Lu, C. Yang, and M.-H. Yang. Saliency detection via absorbing markov chain. In *ICCV*, 2013.

[13] J. B. Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical society*, 7(1):48–50, 1956.

[14] Y. Li, X. Hou, C. Koch, J. M. Rehg, and A. L. Yuille. The secrets of salient object segmentation. In *CVPR*, 2014.

[15] T. Liu, Z. Yuan, J. Sun, J. Wang, N. Zheng, X. Tang, and H.-Y. Shum. Learning to detect a salient object. *TPAMI*, 33(2):353–367, 2011.

[16] L. Marchesotti, C. Cifarelli, and G. Csurka. A framework for visual saliency detection with applications to image thumbnailing. In *ICCV*, 2009.

[17] R. Margolin, L. Zelnik-Manor, and A. Tal. How to evaluate foreground maps. In *CVPR*, 2014.

[18] N. Otsu. A threshold selection method from gray-level histograms. *Automatica*, 11(285-296):23–27, 1975.

[19] F. Perazzi, P. Krähenbühl, Y. Pritch, and A. Hornung. Saliency filters: Contrast based filtering for salient region detection. In *CVPR*, 2012.

[20] R. C. Prim. Shortest connection networks and some generalizations. *Bell system technical journal*, 36(6):1389–1401, 1957.

[21] Y. Qin, H. Lu, Y. Xu, and H. Wang. Saliency detection via cellular automata. In *CVPR*, 2015.

[22] U. Rutishauser, D. Walther, C. Koch, and P. Perona. Is bottom-up attention useful for object recognition? In *CVPR*, 2004.

[23] J. Shi, Q. Yan, L. Xu, and J. Jia. Hierarchical image saliency detection on extended cssd. *TPAMI*, 38(4):717–729, 2016.

[24] R. Strand, K. C. Ciesielski, F. Malmberg, and P. K. Saha. The minimum barrier distance. *CVIU*, 117(4):429–437, 2013.

[25] N. Tong, H. Lu, X. Ruan, and M.-H. Yang. Salient object detection via bootstrap learning. In *CVPR*, 2015.

[26] Y. Wei, F. Wen, W. Zhu, and J. Sun. Geodesic saliency using background priors. In *ECCV*. 2012.

[27] Q. Yan, L. Xu, J. Shi, and J. Jia. Hierarchical saliency detection. In *CVPR*, 2013.

[28] C. Yang, L. Zhang, H. Lu, X. Ruan, and M.-H. Yang. Saliency detection via graph-based manifold ranking. In *CVPR*, 2013.

[29] J. Yang and M.-H. Yang. Top-down visual saliency via joint crf and dictionary learning. In *CVPR*, 2012.

[30] Q. Yang. A non-local cost aggregation method for stereo matching. In *CVPR*, 2012.

[31] Q. Yang. Stereo matching using tree filtering. *TPAMI*, 37(4):834–846, 2015.

[32] J. Zhang and S. Sclaroff. saliency detection: a Boolean map approach. In *ICCV*, 2013.

[33] J. Zhang and S. Sclaroff. Exploiting surroundedness for saliency detection: a Boolean map approach. *TPAMI*, 2015.

[34] J. Zhang, S. Sclaroff, Z. Lin, X. Shen, B. Price, and R. Měch. Minimum barrier salient object detection at 80 fps. In *ICCV*, 2015.

[35] W. Zhu, S. Liang, Y. Wei, and J. Sun. Saliency optimization from robust background detection. In *CVPR*, 2014.