Image Segmentation using Combined User Interactions

Jonathan-Lee Jones, Xianghua Xie*, and Ehab Essa
Department of Computer Science, Swansea University, Swansea, UK
*x.xie@swansea.ac.uk
http://csvision.swan.ac.uk

Abstract

We present an approach for user assisted segmentation of objects of interest, as either a closed object or an open curve. The proposed method combines point based soft constraint on object boundary and stroke based regional constraint. The user points act as attraction points and are treated as soft constraints, rather than hard constraints that the segmented boundary has to pass through the user specified control points. User can also use strokes to specify region of interest. The probabilities of region of interest for each pixel are then calculated and their discontinuity is used to indicate object boundary. This combined approach is formulated as an energy minimization problem on a multilayered graph and is solved using a shortest path search algorithm. We show that this combined approach allows efficient and effective interactive segmentation, which can be used in open or closed curves to segment a variety of images in different ways, and the method is compared against several other techniques, qualitatively.

1. Introduction

Effectively separating objects of interest from background in images is of vital importance to many applications. Automated techniques are appealing in terms of efficiency. More often than not, prior knowledge about object appearance and/or shape is necessary to achieve meaningful results. However, it is not always practical or even possible to obtain comprehensive prior information and sufficiently robust learning algorithm to deal with large and sometimes unpredictable variations in real world images. An alternative approach to automated segmentation is to allow and encourage user input and provide interactive segmentation results to suffice user demand. Often, one dilemma is to balance user involvement and interaction flexibility, particularly given the ubiquitous imaging device and ever increasing amount of images in modern age. Effectively and efficiently capture user intension is vitally important.

The user interaction is conventionally made by simple mouse click or drag operations on the region of interest or on the object boundary. Intelligent paint [16] is a simple interactive method that allows the user to identify all the regions inside an object. The object region is interactively expanded by simple click and drag operation. Homogeneous area that has the same intensity profile is selected. Intelligent Scissors [15], and Live Wire [6] are among early methods to perform on the fly segmentation by allowing the user to follow the object boundary instead of region through a few mouse click. These methods are based on well-known shortest path algorithms, such as Dijkstra's method, to find the optimal shortest path between two user points. Shortest path methods have an advantage of segmenting both open and close end objects. However, often only edge-based features are used to find the shortest path, and more importantly usually those user points are treated as anchor points that the segmented path has to go through.

With the help of powerful optimization techniques, user interaction has been expanded to, for example, adding object/background strokes, at the same time simplify user involvement compared to painstakingly tracing the object boundary [1, 17, 19, 20, 2, 6, 24]. For example, the user can simply draw multiple strokes inside and outside the object then the segmentation model can learn the distribution of pixel intensities for both object and background. These techniques usually is more suited for segmenting closed objects, but not for open curve segmentation.

Graph cut algorithms are widely used to find optimal solution in interactive segmentation. It is usually for segmenting closed objects. Boykov and Jolly [1] introduce a graph cut based method by defining unary and pairwise costs of each pixels. The unary cost is inversely proportional with the probability of each pixel

Figure 1. From left: Graph cut [1], Seeded Star Graph Cut [20], GrabCut [17], and proposed method. Red curve shows the segmentation result, blue for the background strokes, green for background strokes and yellow for the star point and the initial window of the Grabcut.

to be in the object or in the background while the pairwise cost is based on the intensity difference between two neighboring pixels. Many methods have been introduced to extended this method, such as Grab Cut [17] and Lazy Snapping [11]. In the Grab Cut, the authors proposed to use a Gaussian mixture model to build a local color model to enhance the unary cost. It reduces the user intervention by allowing the user to define a rectangular window surrounding the object. Lazy Snapping is also based on graph cut over a pre-segmented image using a watershed algorithm. K-means is used to cluster the foreground/background colors and assign each pixel to the nearest cluster. The method also has a boundary editing tool to refine the result. However, these methods usually needs a multiple user intervention to correctly cut out the object, due to the simplicity in cost function. Shortest path is another optimization technique that has been used in interactive segmentation, e.g. [15, 6, 24]. Those methods emphasize on boundary based features; edge based features are used to define the cost between pixels. The user interactively identify a starting point of the path and iteratively add more seeds around the outline of the object. On the other hand, Intelligent paint method [16] allows the user to identify regions inside the object instead of the boundary. The region is interactively expanded by simple click and dragging operations. Incorporating shape prior into graph based segmentation has also shown improving the segmentation result, e.g. [7, 12, 22, 20]. Veksler [20] introduced a star shape prior to graph cut, also through user interaction. User is required to specify the center of ROI as the star point, and hence all boundary points of ROI lie on the radial spikes from the star point. Additional points, specifying foreground and background, are often necessary. A ballooning term is also used to discourage bias towards small segment. However, the method can only segment the convex shapes. Gulshan *et al.* [9] have extended the method to multiple stars by using Geodesic paths instead of Euclidean rays. Other interactive segmentation methods such as a transductive framework of Laplacian graph regularizer [5] has been also introduced.

In this work, we propose an approach to combine these two different types of user interactions, i.e. boundary based interaction (utilising the user inputted control points) and region based stroke interaction, to segment the image. The user control points, however, are treated as soft constraint, instead of hard constraint in most interactive segmentation methods. We show that this soft user constraint allows effective combination of boundary and region based features. The user points give the user control over the segmentation process, allowing errors in segmentation to be easily prevented and a more desirable result to be obtained Fig. 1.

The rest of the paper is organized as follows. Section 2 presents the proposed method, including user input, superpixel segmentation, and multilayered graph segmentation. Experimental results from segmenting natural images with groundtruth are presented in Section 3. Section 4 concludes the paper.

2. Proposed Method

The proposed method involves the user selecting a series of user control points on the image. These represent the start and the end point for the segmentation, and the user selected points act as the attraction points in the shortest path search which results in the segmentation. These user selected points act in a fashion similar to an elastic band, pulling the segmentation

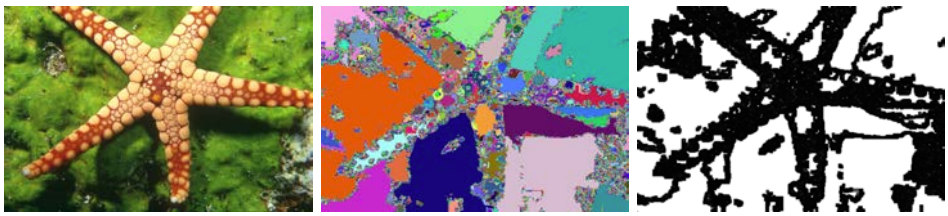

Figure 2. Stages of the process. From left to right, initial image, super-pixel segmentation, binary representation of graph nodes

towards them. In this way, it is possible for the user to influence the segmentation process so as to select features that they want. In order to enhance the image segmentation, the user can also select areas for foreground using strokes.

An energy functional is then formulated based on the combination of the attraction force that is computed using distance transform and the discontinuity in foreground probability. By assuming the user points are in a sequential order, we construct a multi-layer graph with each layer encapsulating a single individual user point. The segmentation problem is then transformed into searching the shortest path in this layered graph. This layered approach allows the segmentation to be carried out in polynomial time, instead of an NP-hard optimization problem, at the same time achieving global minima.

The combination of these user selected points and regions are used to calculate the energy function. We use an approach where we create multiple identical layers, all made up of duplicates of the image, for each user point added. The resulting segmentation is obtained through searching a minimum path in this stack of layers in a manner similar to a 3D object. In this way we can ensure that the shortest path results in a global minimum, thus avoiding local minima that can be a problem with other techniques.

2.1. User Input

The proposed method allows two different types of user input: attraction points to indicate the edge of the desired object and strokes to indicate region of interest. Fig. 2 provides an example of segmentation using the proposed method. Conventionally, user input to segmentation is focused on foreground and background specification [1, 17, 19, 20]. For example, in [17], the user interaction consists of dragging a rectangle around the object of interest and in doing so the user specifies a region of background that is modeled in separating the foreground object. Several other methods require user to specify points on the object boundaries instead [2, 6, 24]. However, more often than not, these boundary based user points are treated as anchor points and the segmentation path has to go through them. This kind of hard constraint is not always desirable. It does not allow imprecise user input, and it can lead to difficulties in combining region based and boundary based approaches as discrepancy between different object descriptions is generally expected. Notably, in [24] the authors introduced soft constraint user point by embedding the user constraint in distance functions. The segmentation result is considered to be the shortest path to loosely connect the user points. However, it is known to be a NP-hard problem. Hence, it is assumed that the user points are placed in a sequential order and such a constraint reduced the computational complexity to polynomial time. This user input constraint can be seen to be generally acceptable as it is intuitive to follow the outline of an object, rather than skipping around. In this work, we follow this approach to treat boundary based user points. However, we also allow user to place region based strokes. These strokes are used to model foreground probability, and the discontinuity in foreground probability indicates the presence of object boundary. We combine these two types user input with image features in an energy functional which is then optimized using graph partitioning through finding the shortest path from the first to last user points. Moreover, we apply a superpixel segmentation in order to generate a much coarser, but irregular, multilayer graph so that the computational cost is drastically reduced. It also provides a regional support at a low level for the shortest path search in the graph.

2.2. Superpixel Segmentation

Efficient search for the shortest path, for instance, using Dijkstra's Algorithm on a multidimensional graph is not a trivial task. Many researchers attempted to speed up the Dijkstra's Algorithm by e.g. using multilevel scaling [10] or restricting the search space [23] by deciding whether or not the edge will be considered during the searching process. In this paper, we speed up the Dijkstra's Algorithm by using the mean shifting method to over-segment the image. This over segmented image is then used to create the graph, by only

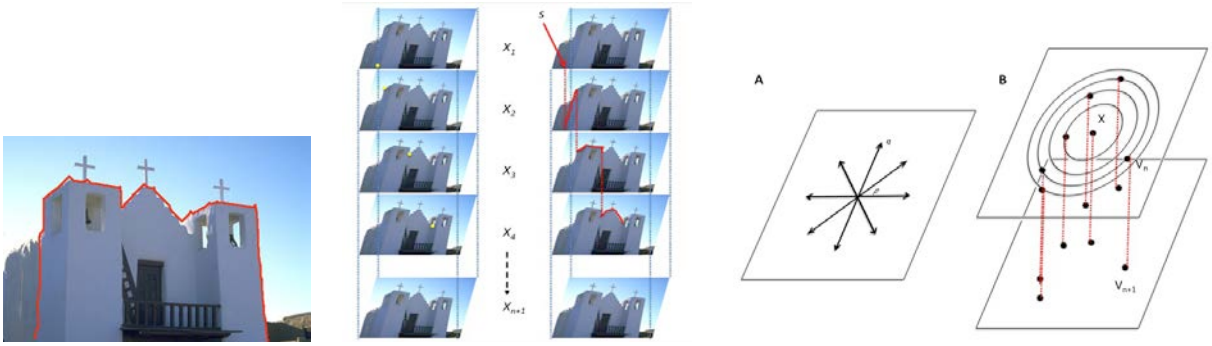

Figure 3. Layered Graph Construction. The stack of images in the middle and right show how the graph is constructed out of a number of layers corresponding to the number of user points $n+1$. The final result of the segmentation is shown on the left. The diagrams on the right illustrate the internal layer edges (A) and the edges between neighboring layers (B).

considering the boundary of the superpixel regions as a potential paths that can be used to find the shortest path between two points, the whole process is thus far more efficient. Additionally, this superpixel segmentation provides low level regional information to the graph search which relies significantly on edge information.

Superpixel segmentation method is grouping a set of homogeneses neighboring pixels together to reduce the complexity of solving further image processing such as segmentation [11, 16] and object localization [8]. Superpixel segmentation algorithms vary from graph based [18, 14] to gradient descent methods [3, 21].

Mean shift algorithm [3] is a non-parametric gradient descent method that iteratively shifting the mean of the region toward the local maxima of the density for a given set of samples. Mean shifting method is suitable for clustering any real data without any assumption of the cluster shape. It has been widely used in many applications, such as clustering [3] and tracking [4]. Given n data points (pixels) of x_i in the d-dimensional space R^d, the non-parametric probability function is defined by kernel density estimator (KDE) as the following:

$$f(x) = \frac{1}{nh^d} \sum_{i=1}^{n} K(\frac{x - x_i}{h}) \qquad (1)$$

where h is the bandwidth parameter and K is the radially symmetric kernel such as Gaussian kernel $K(x) = (2\pi)^{-d/2} \exp(-\frac{1}{2} \|x\|^2)$. The local maxima of density is located among the zeros of the gradient $\nabla f(x) = 0$. So the mean shift can be derived as the following:

$$m_{h,G(x)}(x) = \frac{\sum_{i=1}^{n} x_i G(\left\| \frac{x-x_i}{h} \right\|^2)}{\sum_{i=1}^{n} G(\left\| \frac{x-x_i}{h} \right\|^2)} - x \qquad (2)$$

where $G(x) = -K^{'}(x)$ and $m_{h,G(x)}(x)$ is the difference between the weighted mean, using kernel G, and x, the center of the kernel. The mean shift vector points toward the maximum increase of the density and it converges at a nearby point where the density estimate has zero gradient.

Figure 2 shows an example of the Mean Shift segmentation. Mean shift is preserving the edge features in the image. The black region, shown in the second row of the figure, represents areas on or close to edges in the superpixel segmentation, and are used to construct the graph as it is discussed in the next section.

2.3. Layered Graph Construction

In order to impose soft constraint for user point, we follow the approach proposed in [24] to construct a layered graph so that given a set of attraction points we fit a curve to follow the edges in the image and pass through the vicinity of the given points. The user points are assumed to be placed in a sequential order, which is acceptable in our application. The computational complexity, however, is reduced to polynomial time.

For each user point, $X_i, i \in \{1, 2, ..., k\}$, we create a new layer of directed graph. This is a copy of the image layer, with the same edge based weighting. In that way we have a series of layers equal to the number of user points n, plus an additional layer (in order for the last user point's weighting to be used), as shown in Fig. 3. This results in a multi-layer directed graph, $G = (V, E)$, where V is the set of vertices, and E the set of weighted edges. For each pixel p, there exits an edge e to each of its neighboring pixels (up to 8) on the same layer, providing that they are on the boundaries of the super-pixels. Therefore, a pair of neighboring pixels $(p, q) \in V$ with a corresponding edge $e = (v_p, v_q)$ also have an edge to the corresponding point on the superseding layer $e = (v_{p_i}, v_{p_{i+1}})$, where i represents the current layer of the image. For each edge, we assign a weight w to build a weighted graph (V, E). These

weights are calculated based on whether the edge is internal to a layer (w_i) or trans-layer (w_x). By creating the graph in this way, an order is established with the user points. Edges of zero weight are added from the start node s to each pixel in the first layer, and from the terminal node t to the last layer $k+1$. If P is the set of pixels in the image, and p_i and q_i are pixels in layer i, we can define the set of nodes V as

$$V = \{s, t\} \cup \{p \in P \wedge 1 \leq i \leq k+1\} \tag{3}$$

and thusly the set of edges as,

$$E = \begin{cases} (s, v_{p_1}) | p \in P & \cup \\ (v_{p_{k+1}}, t) | p \in P & \cup \\ (v_{p_i}, v_{q_i}) | (p, q) \in N \wedge 1 \leq i \leq k+1 & \cup \\ (v_{p_i}, v_{p_{i+1}}) | p \in P \wedge 1 \leq i \leq k+1. \end{cases} \tag{4}$$

The segmentation is thus to find the shortest path from the start point s to the end point t, see Fig. 3.

The edges on the directed layered graph are categorized as internal edges w_i within individual layers and interlayer edges w_x. The weighting for these two types edges is assigned differently.

The internal edges are assigned with two types of weights, i.e. boundary based edge weights and region based edge weights. The boundary based edge weights are calculated based on the magnitude of image gradients, i.e. using an edge detection function $g = 1/(1 + \nabla I)$ where I denotes the image or its smoothed version using, for instance, Gaussian. Hence, for any given edge between neighboring pixels (v_p, v_q) we assign a weight (w_e) according to

$$w_e((v_p, v_q)) := 1/2||p - q||(g(p) + g(q)). \tag{5}$$

The region based edge weights are computed from foreground probabilities. The user strokes placed in the foreground provide an estimation for foreground intensity distribution, which is then used to evaluate each pixel in the image. The discontinuity in this generated probability map is then used to compute the region based edge weight in the similar fashion to image intensity, i.e.

$$w_f((v_p, v_q)) := 1/2||p - q||(g_f(p) + g_f(q)) \tag{6}$$

where g_f is the edge detection function based on probability values. The internal edge weight is thus the linear combination of the boundary based weight and region based weight: $w_i = w_e + w_f$. The attraction force imposed by user points is materialized through the interlayer edge weights w_x. We apply distance transform to the user points in each layer of the graph, and the interlayer edge weight is assigned as $w_x = d(v_{p_i}, v_{p_j})$ where d denotes the distance transform function. This distance weighting produces isolinear bands of weight around the user point, with increasing weight to go through to the next layer as the distance from the user point increases.

2.4. Energy Minimization

The energy function for any curve C in our method is a combination of three terms, i.e.

$$\begin{aligned} \mathcal{E}(C, s_1, ..., s_k) =& \alpha \sum_{i=1}^{k} ||C(s_i) - X_i|| \\ &+ \beta \int_0^{L(C)} g(C(s))ds \\ &+ \int_0^{L(C)} g_f(C(s))ds, \\ & s.t. s_i < s_j, \forall i < j. \end{aligned} \tag{7}$$

The first term is used to enforce the soft constraint by the user points, and it penalizes the path further away from the user points. The second term is the boundary based data term that prefers the path passing through strong edges, whileas the last term is the region based data term which prefers path traveling through abrupt changes in foreground probability. By using the layered graph construction, the minimization of the energy functional is achieved by finding the shortest path from the start point s to the end point t. The Dijkstra's algorithm is used to calculate the shortest path in the layered directed graph. Note, the interlayer edges are unidirectional so that the path can not travel back to previously visited layers.

The Dijkstra's algorithm is working on a directed graph $G(V, E)$ to find the shortest path between two defined nodes, the algorithm divided the nodes of the graph to two sets; visited and unvisited nodes. Once the node is marked as visited node, it will not be checked again. The algorithm starts searching from the starting node s, assigns an initial tentative distance of zero to the staring node and infinity to all other nodes, and then calculates the tentative distances for all neighboring nodes, these tentative distance is defined as the summation of the edge weight w_i and the current distance of the beginning node of that edge. The edge weight must be nonnegative value. The algorithm will mark the node that has the minimum distance as a visited node. The algorithm will repeat the process by calculating the tentative distance for all neighboring nodes for all visited nodes and only mark the node has

the minimum distance as a visited node until reach the terminal node t. The running time of Dijkstra's algorithm is $O(|E| + |V|log|V|)$ where E is the number of edges and V is the number of nodes.

3. Experimental Results

To show the effectiveness of the proposed method, we compare our method to others by using it to segment general images, e.g. natural scenes and wide life animals. The proposed method was evaluated using the Berkely Image Database [13]. This dataset contains images of various types. The methods were used to perform a selection/segmentation based on features in the image that would be a realistic segmentation to be carried out (for example, object selection, horizon selection etc.) The results from the proposed method were then compared to other available methods, namely $s-t$ graph cut [1], seeded star graph cut [20], GrabCut [17], and [24]. A selection of open and closed curves (a feature of the proposed method that other segmentation methods do no always have) were used to demonstrate and compare the results.

The proposed method showed a very favorable segmentation performance compared to the methods we tested it against. The combination of a background/foreground separation, combined with the edge based approach gave the method a very robust segmentation, being able to segment an object of interest from an image where other (single methodology based) techniques found difficult to handle, for example if colors were closely related to background, or if there were many conflicting edges. The advantage in being able to perform with open or closed curves is shown in some of the images, for example dividing the image on the horizon or segmenting figures that extend to the edge of the image. Fig. 5 provide several comparative results in segmenting images of natural scenes or wide life animals. Fig. 6 shows examples results of segmenting human from complex background.

Fig. 4 demonstrates the advantages of the inclusion of regional data into the algorithm, in comparison to [24]. In some cases, where there are edges (other than those required), without the regional selection, the segmentation can lose accuracy. By selecting the region specificity, in most cases these edges can be ignored without the requirement for more user points, which would increase the complexity of the graph.

4. Conclusion

We presented an interactive segmentation technique which combines boundary based and region based object representations. We used this method to segment both natural images, and a specific application for segmentation. We adopted layered graph representation to simplify computation, and a super-pixel method to improve segmentation speed and efficiency. The proposed method was compared against a recent methods, and the standard graph cut techniques, showing improved versatility and results. Where other methods had difficulty with certain image types, the combined approach was able to segment the desired information.

References

[1] Y. Boykov and M.-P. Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in n-d images. In *ICCV*, volume 1, pages 105–112, 2001. 1, 2, 3, 6, 8, 9

[2] L. Cohen and R. Kimmel. Global minimum for active contour models: A minimal path approach. *IJCV*, 24(1):57–78, 1997. 1, 3

[3] D. Comaniciu and P. Meer. Mean shift: a robust approach toward feature space analysis. *T-PAMI*, 24(5):603–619, 2002. 4

[4] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. In *CVPR*, pages 142–149, 2000. 4

[5] O. Duchenne, J.-Y. Audibert, R. Keriven, J. Ponce, and F. Segonne. Segmentation by transduction. In *CVPR*, pages 1–8, 2008. 2

[6] A. X. Falco, J. K. Udupa, S. Samarasekera, S. Sharma, B. E. Hirsch, and R. de A. Lotufo. User-steered image segmentation paradigms: Live wire and live lane. *Graphical Models and Image Processing*, 60(4):233 – 260, 1998. 1, 2, 3

[7] D. Freedman and T. Zhang. Interactive graph cut based segmentation with shape priors. In *CVPR*, pages 755–762, 2005. 2

[8] B. Fulkerson, A. Vedaldi, and S. Soatto. Class segmentation and object localization with superpixel neighborhoods. In *ICCV*, pages 670–677, 2009. 4

[9] V. Gulshan, C. Rother, A. Criminisi, A. Blake, and A. Zisserman. Geodesic star convexity for interactive image segmentation. In *CVPR*, pages 3129–3136, 2010. 2

[10] M. Holzer, F. Schulz, and D. Wagner. Engineering multilevel overlay graphs for shortest-path queries. *J. Exp. Algorithmics*, 13:5:2.5–5:2.26, 2009. 3

[11] Y. Li, J. Sun, C.-K. Tang, and H.-Y. Shum. Lazy snapping. *ACM Transactions on Graphics*, 23(3):303–308, 2004. 2, 4

[12] J. Malcolm, Y. Rathi, and A. Tannenbaum. Graph cut segmentation with nonlinear shape priors. In *ICIP*, pages 365–368, 2007. 2

Figure 4. Comparison between Single Method, and the proposed double approach

[13] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. 8th Int'l Conf. Computer Vision*, volume 2, pages 416–423, 2001. 6

[14] A. Moore, S. J. D. Prince, and J. Warrell. Lattice cut - constructing superpixels using layer constraints. In *CVPR*, pages 2117–2124, 2010. 4

[15] E. N. Mortensen and W. A. Barrett. Interactive segmentation with intelligent scissors. *Graphical Models and Image Processing*, 60(5):349–384, 1998. 1, 2

[16] L. J. Reese. Intelligent paint: Region-based interactive image segmentation. *Masters Thesis, Department of Computer Science, Brigham Young University*, 1999. 1, 2, 4

[17] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics*, 23(3):309–314, 2004. 1, 2, 3, 6, 8, 9

[18] J. Shi and J. Malik. Normalized cuts and image segmentation. *T-PAMI*, 22(8):888–905, 2000. 4

[19] A. K. Sinop and L. Grady. A seeded image segmentation framework unifying graph cuts and random walker which yields a new algorithm. In *ICCV*, pages 1–8, 2007. 1, 3

[20] O. Veksler. Star shape prior for graph-cut image segmentation. In *ECCV*, pages 454–467, 2008. 1, 2, 3, 6, 8, 9

[21] L. Vincent and P. Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *T-PAMI*, 13(6):583–598, 1991. 4

[22] N. Vu and B. S. Manjunath. Shape prior segmentation of multiple objects with graph cuts. In *CVPR*, pages 1–8, 2008. 2

[23] D. Wagner and T. Willhalm. Geometric speed-up techniques for finding shortest paths in large sparse graphs. *European Symposiumm on Algorithms (ESA)*, 2832:776–787, 2003. 3

[24] T. Windheuser, T. Schoenemann, and D. Cremers. Beyond connecting the dots: A polynomial-time

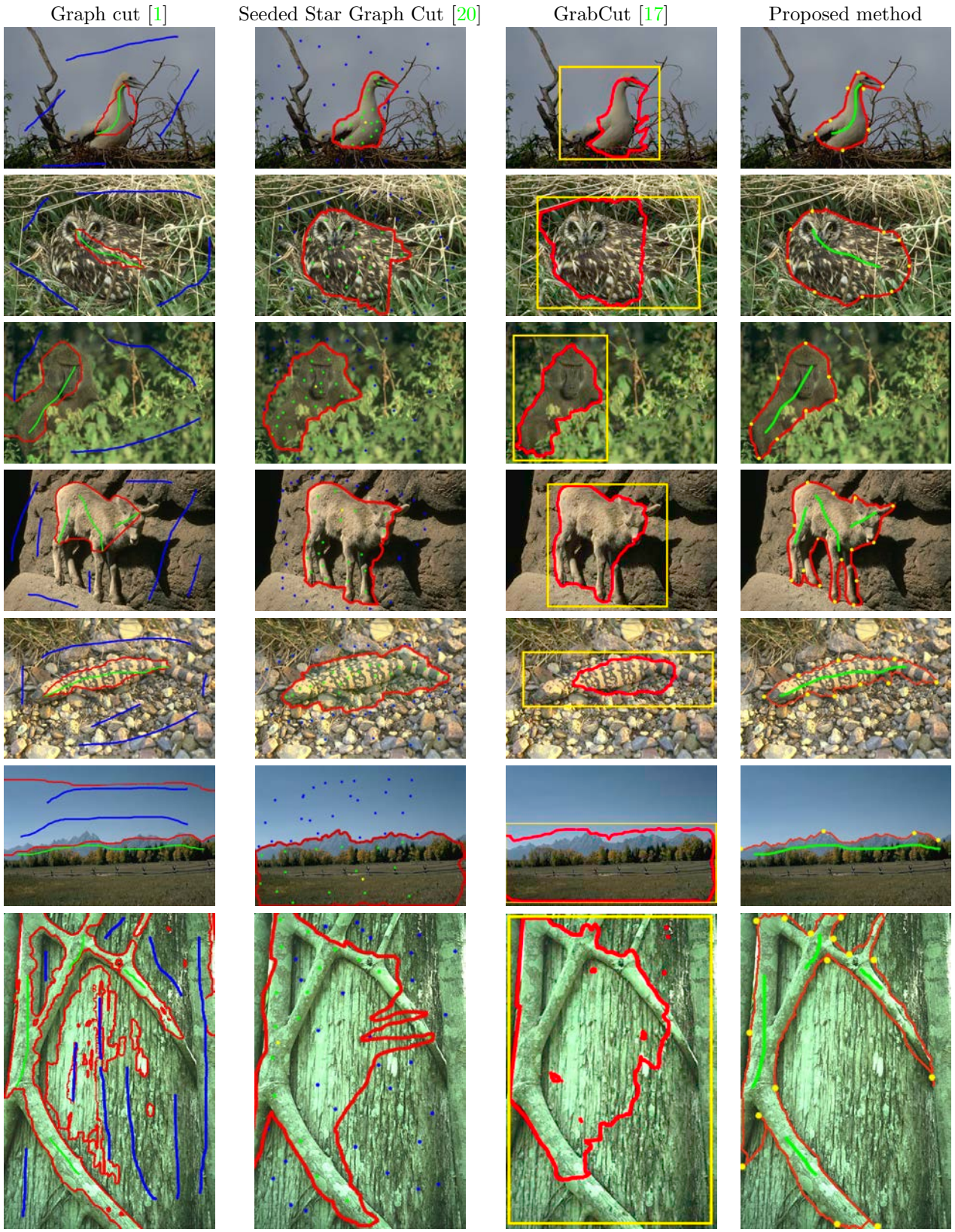

Figure 5. Segmenting images of wide lives and natural scenes. From left: Original Image, Graph cut [1], Seeded Star Graph Cut [20], GrabCut [17], and proposed method. Red curve shows the segmentation result, blue for the background strokes, green for foreground strokes, and yellow for star point and the initial window of the Grabcut.

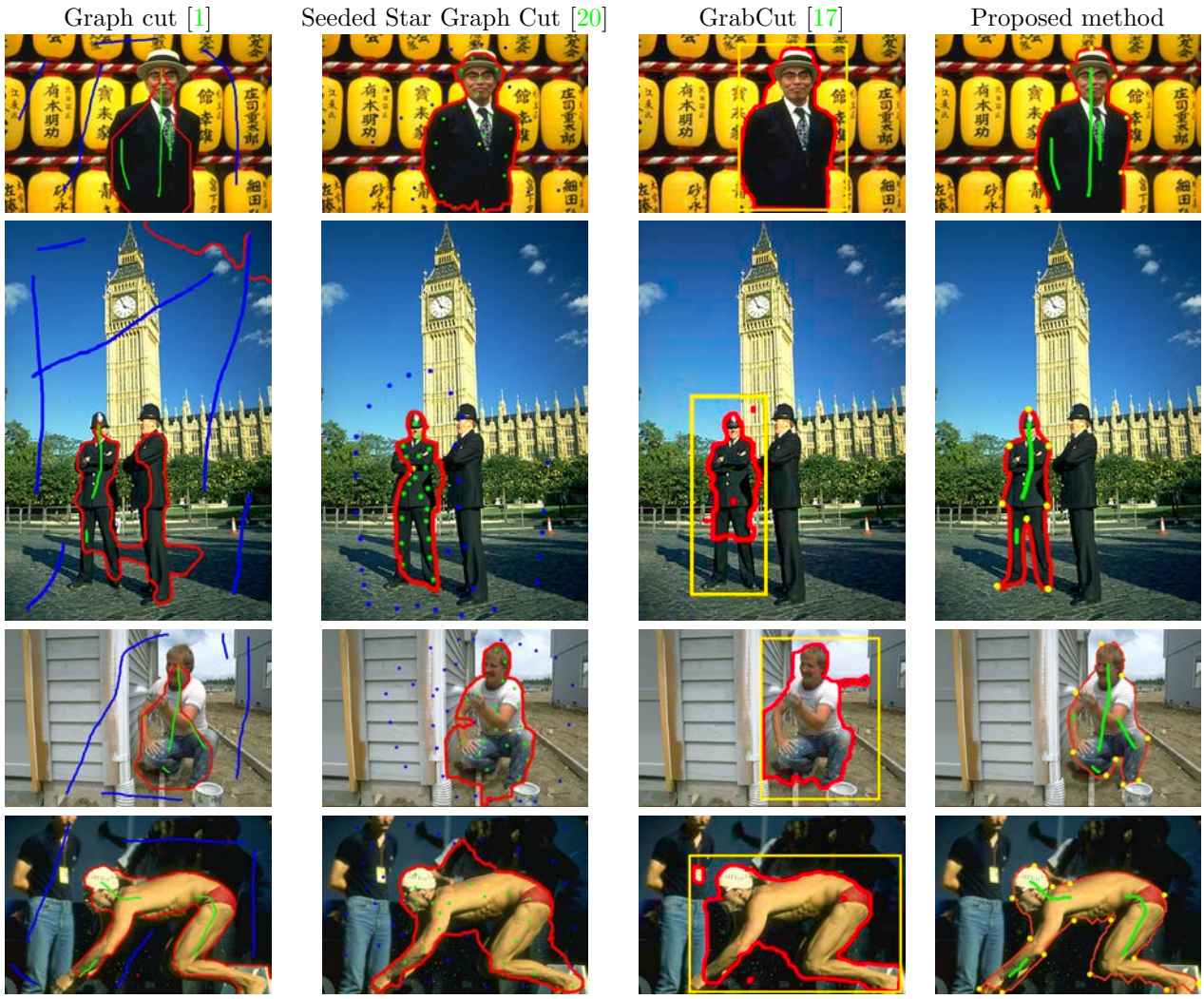

Figure 6. Segmenting human from complex scenes. From left: Original Image, Graph cut [1], Seeded Star Graph Cut [20], GrabCut [17], and proposed method. Red curve shows the segmentation result, blue for the background strokes, green for foreground strokes, and yellow for star point and the initial window of the Grabcut.

algorithm for segmentation and boundary estimation with imprecise user input. In *ICCV*, pages 717–722, 2009. 1, 2, 3, 4, 6