SI-Cut: Structural Inconsistency Analysis for Image Foreground Extraction

I-Chen Lin, *Member, IEEE,* Yu-Chien Lan, and Po-Wen Cheng

Abstract—This paper presents a novel approach for extracting foreground objects from an image. Existing methods involve separating the foreground and background mainly according to their color distributions and neighbor similarities. This paper proposes using a more discriminative strategy, structural inconsistency analysis, in which the localities of color and texture are considered. Given an indicated rectangle, the proposed system iteratively maximizes the consensus regions between the original image and predicted structures from the known background. The object contour can then be extracted according to inconsistency in the predicted background and foreground structures. The proposed method includes an efficient image completion technique for structural prediction. The results of experiments showed that the extraction accuracy of the proposed method is higher than that of related methods for structural scenes, and is also comparable to that of related methods for less structural situations.

Index Terms—segmentation, picture/image generation, scene analysis

1 Introduction

Image segmentation is an essential technique in computer graphics and vision. It involves assigning a label to every pixel of an image such that pixels that have similar visual characteristics have the same labels. The most useful labels are foreground and background labels. After the foreground is extracted, the discrete pixels can be grouped as regions or objects, and they are useful in processes such as image editing and synthesis, and object analysis.

Nevertheless, automatically segmenting precise foreground regions from an image remains challenging. For specific types of subjects, such as people, the difficulty of extraction can be alleviated by using analyzed prior information [1]. By contrast, general-purpose foreground extraction usually requires additional cues from users or other information. For instance, Rother et al. [2] used cues from additional images containing an object identical to that in the target image.

Among various strategies, interactive but intelligent segmentation is regarded as one of the most feasible because it requires only a single image and few user indications. Boykov and Jolly [3] presented the interactive graph cut technique, in which users indicate the foreground and background by drawing sparse strokes. Each pixel in the image is treated as a node in a graph. The contour can be shaped by minimizing energy with respect to the color distributions from indicated strokes [4]. GrabCut, which was proposed by Rother et al. [5], further lessens user intervention and requires only an input bounding rectangle for indicating the foreground. This method is an iterative graph-cut procedure. In every iteration, Gaussian mixture model (GMM) is employed in approximating current foreground and background color distributions to evaluate the contour. This procedure gradually shrinks the foreground region until it satisfies the termination criteria or is stopped by users.

Most existing methods entail labeling pixels according to their color distributions and intensity similarities between neighbors. They perform effectively in separating objects with distinctive colors from the background. However, they are less accurate when the foreground and background color distributions intermix. For instance, in Figs.1 and 2, the color distributions of the leopard and the background rock are similar. GrabCut [5] segmented the scene substantially according to the strength of neighbor links in the graph, and, therefore, the tails and detail contours were difficult to preserve, as shown in Fig. 2(a). Lempitsky et al. [6] assumed that the bounding box of the foreground is close to the indicated rectangle. Their method involves expanding the GrabCut contour when the distances between the bounding box and the indicated rectangle are large. This expansion reduces the excessive-shrinking problem that occurs in GrabCut, as shown in Fig. 2(b).

The purpose of our proposed method is to extract foreground objects from an image by using only an indicated rectangle. In addition to lessening user interaction, in automatic analysis and synthesis applications, a detecting system is easier to indicate a bounding box than to draw precise strokes within a target. Observation indicated that a region can be classified as a foreground according to two criteria: object identification at the semantic level and structural in-

• *The authors are with the Department of Computer Science and Institute of Multimedia Engineering, National Chiao Tung University, Hsinchu City, Taiwan.*
E-mail: ichenlin@cs.nctu.edu.tw

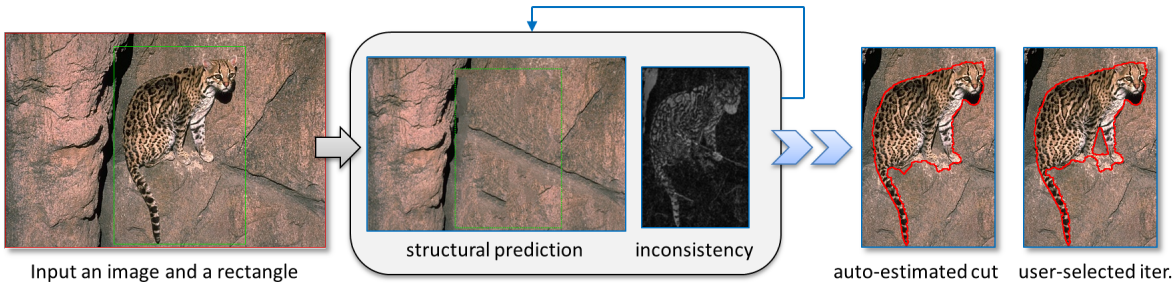


Fig. 1. Segmentation based on structural inconsistency analysis. Given an input image and rectangle, the proposed system iteratively predicts the background structures and evaluates the inconsistent regions. The final cut can be evaluated automatically or according to the iteration determined by users.

consistency (SI) with the background. Using the first criterion requires high-level machine learning and is beyond the scope of our study. This study focused on the second criterion, SI. The term "structure" is defined as the expected variations in color and texture according to spatial variations. For instance, a user can detect a colorful balloon in front of a wall with rainbow ribbons even when the color distributions of the balloon and ribbons are similar. In other words, a foreground pixel is distinct from the background not only in the color spectrum but also in color-spatial space.

However, background structures vary among images, causing the problem examined here to be highly difficult. This paper proposes using an image completion technique to predict the background structure within an indicated rectangle and using the inconsistency to separate the possible background and foreground regions. The detected background can be used to refine structural predictions and segmentation. Fig. 1 illustrates the proposed concept.

The proposed system enables automatically estimating the most likely foreground contour from a sequence of iterations or evaluating the contour from a user-designated iteration. The proposed method was compared with state-of-the-art approaches applied to two datasets from public sources. The results of experiments showed that the contours for structural scenes extracted using the proposed method were more accurate than those extracted using the related approaches, and the contours for less structural scenes were comparable. Fig. 2 shows results of segmentation executed using different methods. The results in (a) and (b) were extracted from [6], and that in (c) was extracted from [7]. Fig. 2(d) and (e) show the results by the proposed method with auto-estimated (AE) and user-assigned (UA) iterations. (The details of iteration determination are described in Section 5.1).

The major contributions of this paper are: *insight into using structural inconsistency* in foreground extraction, *a novel framework* for foreground extraction, and an efficient method for *background structural prediction*.

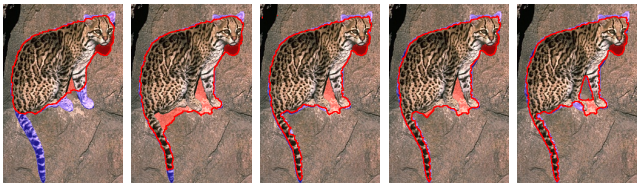


Fig. 2. Segmentation using different methods. (a) Result of GrabCut [5]. (b) Result of Box-prior [6]. (c) Result of One-cut [7]. (e) Result of the proposed method (AE). (e) Result of the proposed method (UA). The red curves represent the extracted foreground contours. For visual comparison, the red channels of incorrectly retrieved pixels and the blue channels of missing pixels were enhanced.

2 Related Work

This section first introduces state-of-the-art techniques applied in interactive image segmentation and related fields. Since the background structure is predicted by an image completion technique in the proposed method, several articles on that topic are introduced as well.

Interactive segmentation and related topics - Several studies have considered features other than color distributions for image segmentation. Price et al. [8] incorporated the geodesic distances from users' strokes and their confidence weights into graph-cut optimization. To improve stroke-based segmentation, Zhou et al. [9] considered color and a texture descriptor, representing the intensity change rate in a local area, as the primary features. Neighbor links in the graph are weighted according to the structure tensor. An active contour is applied at the postprocessing stage of the graph-cut procedure.

Nieuwenhuis et al. [10] combined the color and spatial distributions of user-provided strokes. Their method enables reducing the number of strokes for images with overlapping distributions of foreground and background colors. The awareness of color-spatial distributions is similar to the proposed concept of structure. However, when the color-spatial distribu-

tions were directly applied to the GMM in GrabCut, the approximation of a high-dimensional GMM became sensitive to the number of Gaussian components.

The aforementioned methods involve estimating the foreground and background appearance models according to user-indicated strokes. By contrast, segmentation using an indicated rectangle can be a difficult *NP-hard* problem, since a system must concurrently estimate the appearance models and boundaries of the foreground and background. Classic methods [5], [6] involve using iterative procedures to overcome this difficulty. Tang et al. [7] assumed that the two color distributions are distinct and proposed a distribution overlap penalty in the segmentation energy function. This assumption enables solving the problem in one cut.

Other research used different approaches for user indication. Liu and Yu [11] required users to draw one or more rectangles that predominantly contained foreground objects. To increase discriminability in segmentation, they proposed a multipass level-set method in which edges, gradients, and color histograms are considered. Lazy Snapping [12] entails performing image cutout in two steps. In the first step, the object of interest is specified using a few marking strokes. The second step enables the user to edit the object boundary by dragging polygon vertices. Torsney-Weir et al. [13] developed a visual interface that experts can use to tune parameters for medical image segmentation.

By contrast, saliency detection methods involve extracting objects according to the distinctness in given appearance models. Goferman et al. [14] detected salient regions according to the occurrence frequency of patches in various sizes and a visual perceptual model. Shen and Wu [15] combined low-level segmentation with high-level priors, such as faces and warm colors. Cheng et al. [16] scored the saliency map by the weighted sums of color and spatial distances between regions. Perazzi et al. [17] took geodesic-distance-based elements as bases for regional saliency evaluation, and they retrieved the per-pixel saliency by up-sampling. Yan et al. [18] addressed a scale problem and fused saliency cues from multiple layers.

Image completion, which is also called inpainting, is a technique for filling holes or replacing unwanted objects in an image. Bertalmio et al. [19] smoothly propagated colors from surrounding areas in the isophote directions, but they did not reproduce textures. Criminisi et al. [20] employed an exemplar-based texture synthesis technique to propagate both a linear structure and texture into the target region. The mentioned methods involve considering only the local surroundings, and the lines across the target holes may not be connected correctly.

Shift-map editing, which was proposed by Pritch et al. [21], involves regarding object removal or replacement as a region shifting problem. Each pixel in a target hole has an unknown shift vector, and the goal of Shift-map is to estimate the optimal shifting vectors that fit for boundary and smoothness constraints. Darabi et al. presented the image melding method [22], in which a general patch-based optimization is employed. They enriched the patch search space by applying additional geometric and photometric transformations. These two optimization methods generate impressive results and facilitate unknown region estimation. For efficiency, the shift-map concept is adopted and extended in structural prediction in the proposed method.

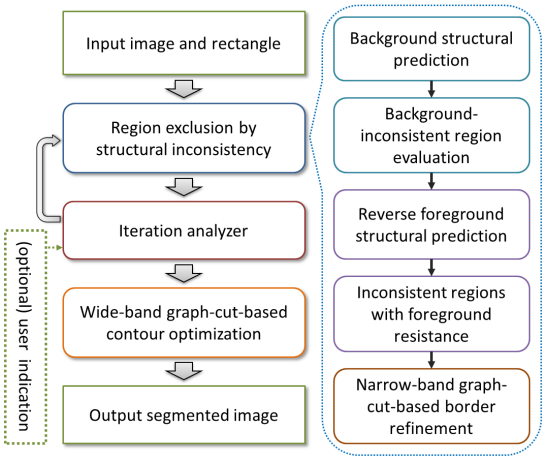


Fig. 3. Flowchart of the proposed SI-Cut framework.

3 Overview

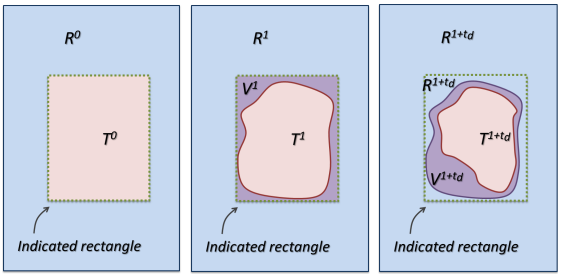
3.1 Proposed Framework

The proposed segmentation framework, which is abbreviated as *SI-Cut*, is based on SI analysis. This framework consists of two primary components: *region exclusion according to SI* and *iteration analyzer*. These two components are iteratively applied until the results satisfy the stop criteria. The iteration analyzer outputs the contours from one or a range of iterations. The *contour optimizer* component then uses a graph-cut-based method to estimate the final foreground contour. Users can optionally indicate to the iteration analyzer when to stop the iterations. Fig. 3 depicts a flowchart of SI-Cut.

Subsection 3.2 introduces an essential subcomponent, structural prediction. Subsection 3.3 describes the efficient implementation of this subcomponent; readers who are not interested in the details on the subcomponent may skip that subsection. Section 4 presents the SI analysis. The iteration analyzer and contour optimizer are presented in Section 5.

3.2 Formulation of Structural Prediction

The function of structural prediction, similar to that of image inpainting, is to predict (fill) certain target regions according to the reference regions. The input is a



(a) 0^{th} iter. (b) 1^{st} iter. (c) $(1+t_d)^{th}$ iter.

Fig. 4. Iteratively updating the regions T, R, and V. The excluded background in the i^{th} iteration first becomes the visible but unreferable region V^{i+1} and then becomes R after t_d iterations.

color image I_{in} and a corresponding indication map. The indication map indicates four types of regions: T, R, V, and M. The region T is the target region to be predicted; R is the reference source region, and the proposed system can learn textures from R; V is the visible but unreferable region and is used for evaluating smoothness at the target borders, but the proposed system does not learn textures from V; M is the masked region and can be skipped.

This prediction procedure is invoked for both background and foreground structures, and the indication maps are varied among iterations. For example, in Fig. 4, the regions inside and outside the user-indicated rectangle are initialized as the target and reference regions, respectively. The proposed system iteratively excludes regions from T. A region excluded in iteration i becomes the region V^{i+1} for t_d iterations (t_d is 2 in the proposed system). Without loss of generality, the example shown in Fig. 4(c) is used in the following explanation.

The proposed prediction approach extends the concept of shift vectors described in [21]. Given an indication map, each pixel p in the T has an unknown shift vector $s(p)$ directed toward a certain reference point $r(p) = p + s(p)$ in R. The result intensity at p is copied from $r(p)$. An optimal prediction fulfils three criteria. The first criterion is *border consistency*. Around the border between T and V or R, the border transited texture should be consistent with a certain texture in the reference region R. The texture of a pixel p is considered a window $W_b(p)$ surrounding the pixel. If any pixel $u \subset W_b(p)$ overlaps with $V \cup R$, then its attributes (color intensities and gradients) should be close to the attributes of a pixel $u' = u + s(p)$ that has an identical relative position to $r(p)$. Fig. 5(a) illustrates the concept and notation.

The criterion is formulated as an objective term E_{brd}:

$$E_{brd} = \sum_{p}\sum_{u}\Big(\lambda_c|I_{in}(u) - I_{in}\big(u + s(p)\big)| + \lambda_g|\nabla I_{in}(u) - \nabla I_{in}\big(u + s(p)\big)|\Big), \quad (1)$$

where $u \subset (V \cup R) \cap W_b(p))$ and $(p \subset T)$; I_{in} is the input color image; ∇I_{in} is the two-channel gradient image of I_{in}; λ_c and λ_g are the normalized weights for channel numbers. The CIE Lab color space was adopted in this study, but it can be replaced by other color spaces.

The second criterion is *neighbor consistency*. For a pixel $p \subset T$ and its adjacent neighbor $q \subset T$, even though their shift vectors $s(p)$ and $s(q)$ may be different, the attribute transit from p to q should be similar to the transit around their reference points and vice versa. Fig. 5(b) illustrates the concept, which is formulated as an objective term E_{nb}:

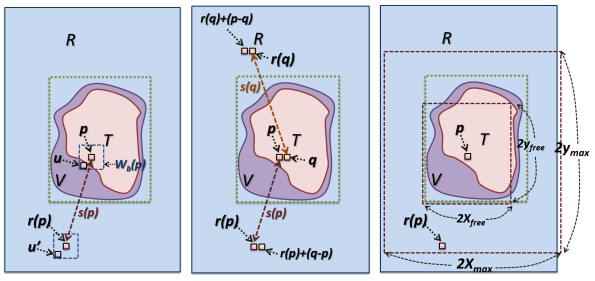
$$E_{nb} = \sum_{(p,q\subset T)\&(|p-q|=1)} e_{nb}(p,q), \quad (2)$$

$$\begin{aligned} e_{nb}(p,q) = {} & \lambda_c|I_{in}(r(p)) - I_{in}\big(r(q) + (p-q)\big)| \\ & + \lambda_c|I_{in}\big(r(p) + (q-p)\big) - I_{in}(r(q))| \\ & + \lambda_g|\nabla I_{in}(r(p)) - \nabla I_{in}\big(r(q) + (p-q)\big)| \\ & + \lambda_g|\nabla I_{in}\big(r(p) + (q-p)\big) - \nabla I_{in}(r(q))|, \end{aligned} \quad (3)$$

where the notation is identical to that used in (1).

The third criterion is the *location penalty*. When there are multiple qualified reference candidates, keeping the shift vector $r(p)$ close to p is usually preferable. This locality criterion is mainly designed for foreground structural prediction described later. Because the behavior of foreground prediction is highly similar to extrapolation, this term can help target pixels refer to nearby reference regions if there is no other appropriate choice. It is also applicable to background prediction, which is a user option. These constraints are formulated as a piecewise linear function approximating the sigmoid shape.

When x and y of the shift vector $s(p)$ are less than the constraints x_{free} and y_{free}, they do not



(a) (b) (c)

Fig. 5. Conceptual diagram for three prediction criteria. (a) Border consistency. (b) Neighbor consistency. (c) Location penalty.

receive any penalty. When they are greater than the constraint x_{max} and y_{max}, they receive high constant penalties c_{x_max} and c_{y_max}. Otherwise, the penalties are linearly interpolated according to their distances. The objective term E_{loc} becomes

$$E_{loc} = \sum_{p \subset T} F_{xloc}(p) + F_{yloc}(p), \tag{4}$$

$$F_{xloc}(p) = \begin{cases} c_{x\_max} & \text{, if } |x(s(p))| \geq x_{max}; \\ 0 & \text{, if } |x(s(p))| \leq x_{free}; \\ \frac{c_{x\_max} \cdot (|x(s(p))| - x_{free})}{x_{max} - x_{free}} & \text{, otherwise.} \end{cases} \tag{5}$$

, where, $x(s)$ represents the x extraction function for a vector s. Formulation of $F_{yloc}(p)$ is similar to (5) but x is substituted by y. Fig. 5(c) illustrates the concept and its notation. The default values of x_{free} and x_{max} are 25% and 50% of the width of the indicated rectangle; those of y_{free} and y_{max} are 15% and 45% of the height of the indicated rectangle. A lower tolerance in the y-direction is used, because it is observed that homogeneous structures frequently occur at a similar height.

The summary objective function is E_{pred}:

$$E_{pred} = w_{brd}E_{brd} + w_{nb}E_{nb} + w_{loc}E_{loc}, \tag{6}$$

where w_{brd}, w_{nb}, and w_{loc} are the weights for objective terms. The default values of the first two terms are 0.5 and 0.1875, and that of the final term can be either 0 or 1. Compared with ShiftMap [21], in which only directly connected borders and neighbors are considered, the proposed prediction approach involves considering a window for border consistency and restricting the reference locations. The proposed approach enables preserving more structures from the reference regions and is applicable to structure extrapolation.

3.3 Multilevel Prediction with Moves and Leaps

The aforementioned prediction approach is a graph-based multilabeling problem. Equation 6 can be evaluated using the alpha-expansion method [4]. However, the alpha-expansion method entails decomposing a multilabel graph-cut into multiple binary graph-cut problems. Its complexity is approximately *(label number)· O(binary graph-cut)*. Depending on the implementation, *O(binary graph-cut)* is approximately $O(N_{node} \cdot N_{edge} \cdot \log(N_{node}^2/N_{edge}))$, where N_{node} and N_{edge} are the numbers of nodes and edges in the graph.

For the configuration in Subsection 3.2, the label number is approximately equal to the reference pixel numbers $|R|$; the node number is equal to the target pixel number $|T|$; the edge number is nearly two times of the node number. An input image for segmentation usually contains several hundred thousand pixels.

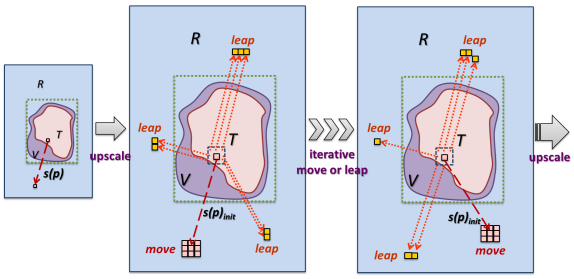


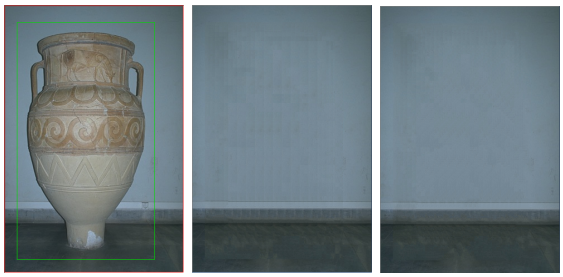
Fig. 6. Shift vector upscaling with local moves and neighbor leaps for multilevel prediction.

Therefore, directly solving (6) requires a substantial amount of computation time. Since all terms in the big O are polynomially related to image pixel numbers, it is reasonable to estimate (6) primarily at a downscaled level and then to propagate the results (shift vectors) back to the higher-resolution levels.

Given an image and its indication map, the proposed system first iteratively downscales their widths and heights by half (a quarter in size) until the target pixel number $|T|$ is lower than 600. At the lowest-scale level, all reference regions can be considered reference points, and the optimized shift vector of each target pixel can be evaluated in less than 5 seconds.

To propagate a shift vector $s(p)$ to the upper level, the double of $s(p)$ can be the initial vector of the adjacent upper level. At the upper level, for efficiency in solving the alpha-expansion, a compact label space instead of the entire reference regions must be selected. The first possibility is to set the upscaled vector $s(p)$ and its eight neighbors $NB(s(p))$ as a label space for pixel p. This label set is called *local move*, as shown in the middle of Fig. 6, where the dashed arrow indicates the double of the lower-level $s(p)$. This label set was used in [21], and these labels facilitate reducing zigzag artifacts during upscaled optimization. Nevertheless, considering only the local-move labels causes the results in the upper levels to be biased by the decision at the lowest level. Hence, additional labels called *neighbor leap* are considered. These labels are the shift vectors of four or eight neighbors of p, $NB(p)$, and are represented by the dotted arrows in the middle of Fig. 6. They are especially useful for the regions where shift vectors are diverse and ambiguous at the lower level. While alpha-expansion is performed with the combined *move-and-leap* (M&L) label sets, the shift vectors can be upscaled and adjusted level by level to the original image resolution.

Moreover, at a certain level, the initial shift vector from the lower level is not retained; instead the estimated shift vectors are used as the new initial guesses, and alpha-expansion is iteratively performed multiple times by using the M&L label sets. Thus, it increases the likelihood that the algorithm does not become trapped in the local minimum (e.g., attempting multiple neighbor leaps). Consider Fig. 7 as an



(a) (b) (c)

Fig. 7. Inpainting results obtained using different label sets. (a) Input image and indicated rectangle. (b) Result obtained using the *local move* label set. (c) Result obtained using the *move-and-leap* label set.

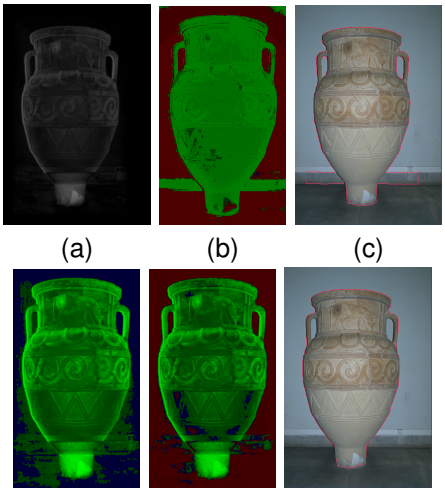
example. Fig. 7(a) is an input image in which the green rectangle indicates the target region T. Multiple iterations of alpha-expansion methods were performed by using *move* and iterative M&L label sets for 10 seconds, respectively. Fig. 7(b) shows the result obtained using the *move* labels. The result was acceptable but exhibited visible stripes. Alpha-expansion using move labels became trapped in the local minimum at approximately the third second. By contrast, performing optimization by using the proposed M&L labels continually lowers the objective value. The objective value by using the M&L labels is two thirds of that achieved by using *move* labels. The result achieved by using iterative M&L labels is shown in Fig. 7(c). When the naive single-level optimization was applied, 15 hours and 41 minutes were required to predict the target region, which comprised 106,144 pixels.

4 Structural Inconsistency Analysis

This section presents the SI feature, which is derived from the structural prediction mentioned in Subsections 3.2 and 3.3. This section illustrates why the feature is effective and describes how it can be used in foreground estimation.

4.1 Background Structural Inconsistency as a Classification Feature

Before the novel feature is introduced, Fig. 7(a) is again used as an example, and the limitation of classification using the color GMM is illustrated. The initial procedure of GrabCut [5] was applied to train the foreground and background GMM according to the pixels inside and outside the rectangle, respectively. Fig. 8(b) shows the per-pixel classification results (without neighbor connectivities). Pixels with higher background probabilities are shown in red, pixels with higher foreground probabilities are represented in green, and pixels of which the probability differences are less than 1% are represented in blue. The gray stripe belonging to the wall was classified as the foreground, and a triangular region at the base of the



(a) (b) (c)

(d) (e) (f)

Fig. 8. Visualizing the discriminability when the color GMM and structural inconsistency were applied. (Classified background in red, foreground in green, and neutral in blue.) (a) Inconsistency values v_{bp} between Fig. 7(a) and 7(c). (b) Per-pixel classification using the color GMM. (c) Graph cut with probabilities in (b). (d) Per-pixel classification by applying a threshold at vtc_{20}. (e) Per-pixel classification by applying a threshold at vtc_{40}. (f) Graph cut with probabilities in (e).

vase was classified as the background because it is white. When the probabilities were applied to the data term in the graph cut, the segmentation shown in Fig. 8(c) was obtained. In this case, the GrabCut may still be applicable when iterative color model refinement is performed and strong neighbor links are present across the rectangle border.

To enhance the discriminability of the per-pixel classification and tailor classification to users' expectations, this study proposes estimating the inconsistent regions between the predicted background I_{bp} and the input image I_{in}. The inconsistency value v_{bp} at pixel p is defined as the weighted summary of differences between I_{bp} and I_{in} in a window W surrounding p:

$$v_{bp}(p) = \frac{\sum_{q \subset W(p)} g(|q-p|) \cdot |I_{bp}(q) - I_{in}(q)|}{\sum_{q \subset W(p)} g(|q-p|)}, \quad (7)$$

where $g()$ is a Gaussian function for weighting. The inconsistency values between Figs. 7(c) and 7(a) are illustrated in Fig. 8(a).

The first approach developed involved mapping the v_{bp} value to the background probability by using a zero-mean Gaussian function and using a certain v_{bp} value as the threshold for foreground candidates. This threshold can be identified in a histogram of v_{bp} values. The top k% of the background consistency value (the smallest k% of v_{bp}) is abbreviated as vtc_k. Figs. 8(d) and 8(e) illustrate the per-pixel classification results obtained by applying thresholds

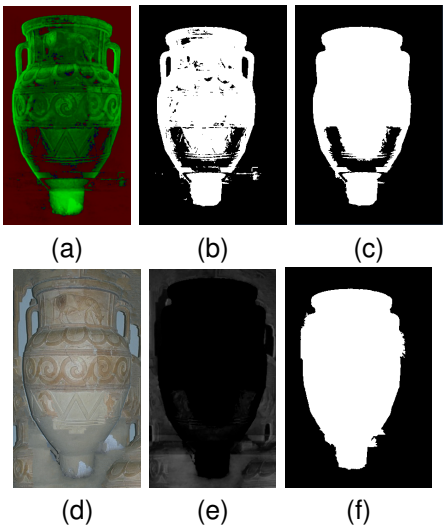


Fig. 9. Resisting excessive background exclusion through reverse structural prediction. (a) Per-pixel classification by applying a threshold at vtc_{50}. (b) Binarizing the per-pixel classification by vtc_{50}. (c) Removing small and isolated connected components. (d) Reverse predicted structure. (e) Inconsistency values obtained by applying reverse structural prediction. (f) Marked map of foreground candidates.

at vtc_{20} and vtc_{40}, respectively. Applying a tighter consistency threshold (e.g. top 20%) caused all true foreground pixels to be retained, but only a few background regions were excluded. By contrast, applying a wider consistency threshold (e.g. top 40%) caused more background regions to be excluded and a few foreground pixels to be included.

In this case, the per-pixel result obtained by applying a threshold at vtc_{40} is close to the ground truth and can directly become a classification feature. For instance, when the probabilities were applied to the data term in the graph cut, the preliminary segmentation was satisfactory as shown in Fig. 8(f). Nevertheless, the predicted background structure typically contains imperfections that hamper the inconsistency evaluation. In the following subsections, the per-pixel SI classification result is used in an improved procedure for adapting the analysis to various situations.

4.2 Reverse Structural Prediction as Resistance

When the aforementioned procedure is used, parts of the true foreground can be considered the background if the threshold is too loose. For example, Figs. 9(a) and (b) show the per-pixel probabilities and their binarized map when an overly aggressive threshold vtc_{50} was applied, respectively. Although small and isolated connected components were filtered out, as shown in Fig. 9(c), parts of the vase were still excluded from the foreground candidate. This problem is called the excessive exclusion problem.

To address this problem, this study proposes examining not only the consistency of background structures but also the foreground consistency. The indication map was changed for reverse prediction. The remaining foreground candidates in Fig. 9(c) then became the reference R for structural prediction. The background region within the rectangle became the unknown T to be predicted. The region outside the rectangle was masked as M. The reverse predicted structure image I_{fp} is shown in Fig. 9(d). Similar to the v_{bp}, the inconsistency value of reverse (foreground structural) prediction at pixel p is defined in (8). The inconsistency values are shown in Fig. 9(e).

$$v_{fp}(p) = \frac{\sum_{q \subset W(p)} g(|q-p|) \cdot |I_{fp}(q) - I_{in}(q)|}{\sum_{q \subset W(p)} g(|q-p|)} \quad (8)$$

Instead of applying a threshold at a certain percentage of the background inconsistency value vtc_k, this stage involves determining whether a pixel is retained in the foreground candidates according to its similarity to predicted background and foreground structures. According to (9), a pixel p remains in the current foreground candidates only if it has a lower inconsistency value with respect to the predicted foreground structure, and it is assigned a nonzero marked value m_{rf}:

$$m_{rf}(p) = \begin{cases} 1 & \text{, if } v_{fp}(p) \le v_{bp}(p); \\ 0 & \text{, otherwise.} \end{cases} \quad (9)$$

Comparison with reverse prediction can be regarded as the resistance to background exclusion. Fig. 9(f) shows the pixels marked according to (9) after isolated components were filtered out.

4.3 Refining Inconsistent Regions

As mentioned previously, small and isolated components are filtered out by executing two operations. The first operation is *region-size filtering*. After connected component labeling (CCL) is applied, the proposed system identifies the largest nonzero regions and then removes regions of which the pixel numbers are lower than 20% of the largest number. The second operation is *reference-reachable filtering*. The border of the indicated rectangle is used as the roots in applying a depth-first-search to the background (zero) pixels. The unreachable background pixels are transformed into the nonzero pixels.

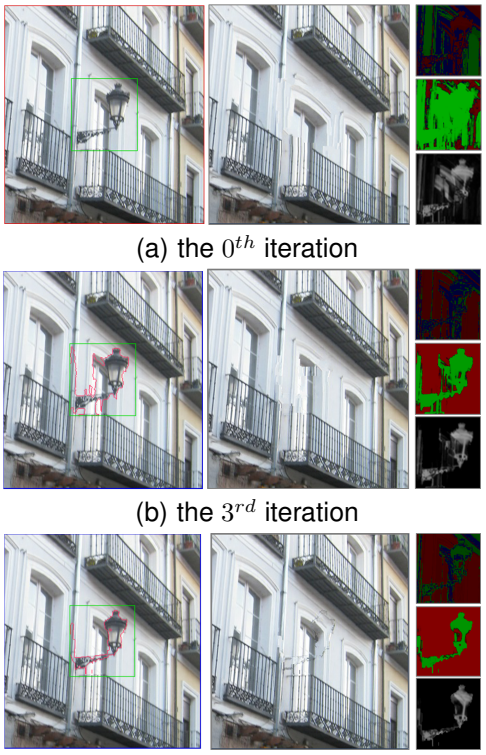
The proposed system subsequently performs an additional operation, *narrow-band graph-cut refinement*. Graph-cut optimization is used to smooth and slightly refine the contour to fit local colors. Given foreground regions F, this binary mask is eroded and dilated as E and D by using a 3×3 block kernel. Pixel $p \subset E$ is set as the definite foreground, pixel $p \subset (D-E)$ is set as the neutral pixel, pixel $p \subset \overline{D}$ is set as the definite background pixels, and graph cut is performed. This

operation facilitates clarifying the cases in which the foreground and background predictions are equivalent in (9). This local tuning operation is different from general graph-cut since the variable (neutral) pixels are restricted in a narrow band along the given contour.

4.4 Iterative Region Exclusion

As shown on the right-hand side of Fig. 3, a complete stage of background region exclusion consists of background-structure-based exclusion, foreground-structure-based resistance and border refinement. To apply this process in extracting various foregrounds, this subsection presents an iterative framework in a conservative strategy. In one iteration, only the regions that are highly consistent (lower k in vtc_k) and adjacent to the background are excluded. During the iterations, the true background regions are gradually extracted from the foreground candidates, and the newly included background regions expand the reference R and increase the accuracy of subsequent prediction.

Fig. 10 illustrates the iterative exclusion of high-consistency regions, showing only the zeroth, third, and ninth iterations. The left column shows the input image with the rectangle, and the red curves represent the temporarily assumed foreground contour before exclusion. At the right-hand side of each row, three small figures show (from the top to the bottom) the classification achieved by using the color GMM, the classification achieved by using vtc_{20}, and the inconsistency value image. In this test, the initial classfication achieved by using the GMM was mostly incorrect, and the GrabCut implemented in the OpenCV library [23] did not classify any pixel as the foreground. The middle column shows the structures predicted by the proposed method. Although the prediction in the zeroth iteration was not perfect, as more background regions were extracted, the prediction in the ninth iteration was quite similar to the true background structure. Furthermore, the temporary foreground and background regions were employed in color GMM training. Using the updated regions enhanced the discriminability of the GMM classifier, as shown in the ninth iteration.



(a) the 0^{th} iteration

(b) the 3^{rd} iteration

(c) the 9^{th} iteration

Fig. 10. Iteratively excluding consistent regions from the foreground candidates. The red curves in the left column show the foreground candidates before iteration t begins (extracted at t-1); the middle column shows the predicted background; the upper-right image in each row shows the classification achieved by using the color GMM (notation is explained in Fig. 8); the middle-right image in each row shows the classification achieved by using vtc_{20}; the lower-right image in each row is the inconsistency value image.

(a) 0^{th} iter. (b) 1^{st} iter. (c) 2^{nd} iter. (d) 3^{rd} iter. (e) 4^{th} iter.

Fig. 11. Five of the six iteration results in region exclusion of a sheep example. (Only the pixels within the indicated rectangles are shown.) The corresponding vtc_{20} values are shown in Fig. 12(a).

5 Optimized Contour from Iterations

This section describes the stop criteria of the iterative framework and how to identify the optimal foreground contour from a sequence of iterations.

5.1 Iteration Analyzer

The most common criteria require that a procedure stop at a maximum iteration number or at a minimum variation threshold. These two criteria are included in the essential stop criteria for the proposed method. Another essential criterion concerns the bounding box of the target region T, called $BBox(T)$. The region T can also represent the region of the remaining foreground candidates. When any side of the $BBox(T)$ is too far from the indicated rectangle, the target regions have been excessively shrunk. The system can terminate the iterations to avoid superfluous computations.

Even when the mentioned stop criteria are applied,

the contour extracted in the final (stop) iteration may not always be close to the true foreground. Fig. 11 shows an example, in which the red curves indicate the contour of T. The orthogonal distances between $BBox(T)$ and the input rectangle reached the threshold in the fourth iteration, but the contour has shrunk excessively. Conversely, the contour in the second iteration is nearly identical to the ground truth. To estimate the optimal iterations automatically, we required users to identify the iterations they preferred the most and analyzed the correlations between these iterations and various factors. Several factors, such as the fitness of contours and image gradients and discrimination between color distributions inside and outside the contour, were examined. One of the primary coefficients in SI evaluation, namely the top k% consistent values, vtc_k, had an interesting characteristic in various cases.

The vtc_{20} versus iteration number of examples is plotted in Fig. 12, where the iterations preferred by users are indicated by dotted circles. The shapes of the vtc_{20} curves resemble a sigmoid function. These curves consist of three main periods in the following order: low-level, rising, and high-level periods. The preferred iterations occur at the transition points between the rising and the high-level periods.

During background structural prediction, the high-consistency regions are typically close to the contour of the initial target T. Therefore, the magnitude of vtc_k is related to the structural predictability of the target regions near the current foreground-candidate contour. This characteristics is discussed by using the sheep example shown in Figs. 11 and 12(a). In the zeroth and first iterations shown in Fig. 11, the initial T regions are the input rectangle and the zeroth contour, respectively. The regions near these two contours T mainly consisted of grass. Therefore, predictions regarding these regions were highly consistent, resulting in low vtc_{20} values. However, after exclusion in the first iteration, only the grass regions near the sheep's legs were predictable according to the reference R; hence, the vtc_{20} in the second iteration markedly increased. After the second iteration, no background region was located inside T, and the vtc_{20} in the third iteration further increased slightly. Subsequently, excessive shrinking occurred. Parts of the true foreground were included in the reference R. The vtc_{20} slightly decreased.

As shown in Fig. 12, the iterations preferred by users occurred at two points:

1) The point with a negative $\frac{d^2(vtc_k)}{dt^2}$ after a sequence of points with a high $\frac{d(vtc_k)}{dt}$. (t represents the iteration id)
2) The point in front of points with the highest values.

Applying these two rules typically results in identical or adjacent iterations. In a few cases, the transition points in the curve were ambiguous, but they still remain within the point interval defined by these two rules. Fig. 13(a) shows a vtc_{20} curve of an image of a woman. The optimal interval of iterations rather than the optimal single iteration was determined. The process is described as follows. After applying a bilateral filter to the vtc_k curve, the proposed system first identifies the iteration with the highest value, t_h. The t_h and its derivative are used to form a tangent line l_h. The system then backward searches the first iteration of which the distance to l_h is lower than a threshold and sets the iteration t_{bb}. Besides, the system forward searches the first iteration of which the second-order derivative on the curve is negative and lower than a threshold. The iteration is set as t_{fb}. Figs. 13(b) and 13(c) show the optimal forward and backward iterations estimated using the iteration analyzer. Please refer to the pseudo code listed in the supplementary document for details.

5.2 Graph-based Contour Optimization

The next procedure is to estimate the single optimal foreground contour F_{gco}. Subsection 4.3 introduces *narrow-band border refinement*, in which the contour ∂F is refined within the band between the erosion and dilation of the given region F. The operation here is similar, but it is applied over a wider interval and can be called *wide-band contour optimization*. The foreground regions of iterations t_{fb} and t_{bb} are F_{fb} and F_{bb}. Since the SI-Cut framework is based on region exclusion, $F_{bb} \subseteq F_{fb}$, and F_{bb} should be the inner bound of the final region F_{gco}. To alleviate the effects of prediction error and other types of noise, F_{bb} is eroded by k_e iterations as E_{bb}, and its contour ∂E_{bb} is indicated as the definite foreground. By contrast, F_{fb} should be the outer bound of F_{gco}; likewise, F_{fb} is dilated by k_d iterations as D_{fb}, and its contour ∂D_{fb} is indicated as the definite background. The terms k_e and k_d are proportional to the minimum side and perimeter of the input indicated rectangle. In addition, the number of k_e is restricted to avoid losing the strip-shaped regions (e.g., the leopard's tail).

In narrow-band refinement, the region inside ∂E is retained as definite foreground to ensure prediction stability. At this stage, a robust contour enables the interior pixels to join the graph-cut optimization. Here, the region inside ∂E_{bb} is indicated as the probable foreground and becomes variable nodes in graph cut. Fig. 13(d) is an indication map for graph-cut computation. Fig. 13(e) depicts the optimized contour, where the background regions around the elbows are unmarked. Users can choose whether to set the interior regions as the variable nodes.

5.3 Extended SI-Cut

The aforementioned primary SI-Cut procedure is effective in most situations. However, fewer than 5%

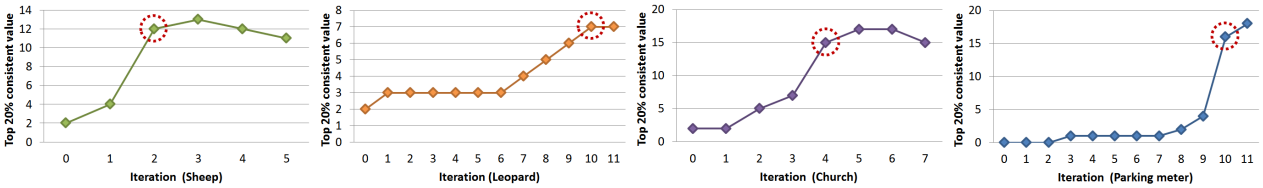


Fig. 12. Four examples of the top 20% consistent values (vtc_{20}) versus the iteration id. The red dotted circles show the iteration most preferred by users.

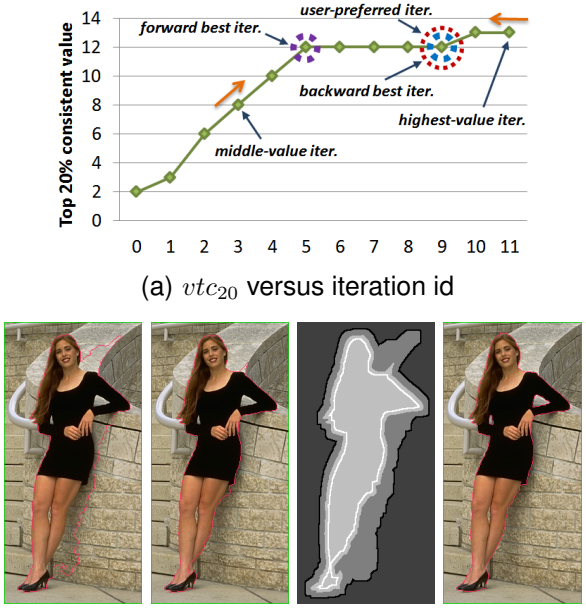


Fig. 13. Optimized contour from the auto-selected optimal forward and backward iterations. (d) Graph-cut indication map. (White: definite foreground; light gray: probable foreground; medium gray: neutral; dark gray: probable background; black: definite background.)

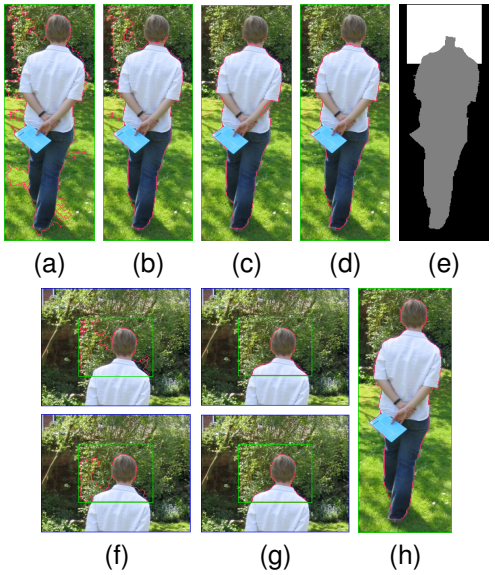


Fig. 14. Extended SI-Cut for an image of a person. (a), (b), and (c) are the contours of the 1^{st}, 2^{st} and 3^{rd} iterations in primary region exclusion. (d) Optimized contour of primary SI-Cut. (e) Indication map for extended SI-Cut (white: target region; gray: masked region; black: reference region). (f) The 1^{st} and 3^{rd} iterations in extended region exclusion. (g)The 5^{th} and 7^{th} iterations in extended region exclusion. (h) Optimized contour of extended SI-Cut.

of the samples did not directly fit this procedure. Fortunately, these images can be segmented by executing extended passes of the SI-Cut procedure with few modifications. Figs. 14(a), (b) and (c) show the first to third iterations of an example of region exclusion performed using the primary SI-Cut procedure. When either a tight or loose bounding-box distance constraint was applied, the person's head could not be retained in the foreground candidates. The optimized cut F_{gco} is shown in Fig. 14(d). This problem also occurs in GrabCut.

To extend the SI-Cut result, the proposed system can set a new indicated rectangle at the top region, when the $BBox(F_{gco})$ is far from the rectangle border. The F_{gco} is then indicated as a masked region. Fig. 14(e) shows the indication map for the extended SI-Cut procedure at the top region. In addition, an extended masked region called *sprout* is added to the top of F_{gco}. The sprout is a short rectangular mask emitting from the barycenter of the top masked region. Its direction is along the outward gradient of the top ∂F_{gco}. During reverse structural prediction, these masked regions become the references. The texture of the sprout can be propagated to the nearby reverse target region and hence, prevent the head from being excluded as a background region. Figs. 14(f) and (g) show four exemplary iterations of the extended region exclusion procedure. The final extended SI-Cut iteration is shown in Fig. 14(h). The proposed extended SI-Cut can be used for top, down, left, or right regions according to their distances to the rectangle. This extension computation is automatic. Since the requirement of an extended procedure is rare, using the extension is an option for users.

5.4 Utilities

This final subsection describes options that increase the efficiency and smoothness of the iterative procedure. First, since the computational complexity of predictions is related to the reference region R, the reference region can be restricted within a fixed scale window with respect to the indicated rectangle (e.g., 3 times in width and height). Second, the contours extracted in the first one or two iterations usually do not reach the foreground contour. The threshold vtc_k can be boosted (e.g., from 20% to 25% or 30%) in the early iterations. The early boosting utility can reduce the number of iterations. For images with only a few reference regions (e.g., the borders of input images and indicated rectangles are close), the v_{bp} of pixels near the reference regions can be lowered. This function can cause those pixels to become references for further structural prediction earlier and does not affect the high-inconsistency regions.

6 Experiment and Discussion

6.1 Comparisons

The proposed method was compared with three related state-of-the-art methods: *GrabCut* [5], the pinpoint method with a bounding box prior [6] (abbreviated as *Box-prior*), and the GrabCut in one cut [7] (abbreviated as *One-cut*). These three methods are designed for segmentation using an indicated rectangle. To evaluate the discrimination of each method, we followed the related articles and calculated the error rate of pixels within the indicated rectangle (originally unlabeled pixels).

6.1.1 Comparison Using a Structural Scene Dataset

A structural scene dataset containing forty images was collected from the public LabelMe dataset [24], in which the ground-truth masks are marked by users. Several images are challenging for segmentation because the distributions of the foreground and background colors overlap. For experimental comparison, we adopted the GrabCut procedure implemented in the OpenCV library [23], and the OpenCV GrabCut program was extended to replicate the Box-prior method. In addition, the One-cut method was reproduced. The error rates of the One-cut method varied according to the smoothness weight. Therefore, we evaluate the results of the One-cut method by two types, *One-cut (GW)* and *One-cut (IW)*, using different parameter settings. The *generally optimal-weight (GW)* type involved using an identical optimal weight for all images, and the *individually optimal-weight (IW)* type entailed using the individually adjusted weight for each test image.

To clarify the capabilities of the proposed method in automatic and interactive use, the results of the proposed method were separated into two types, *the proposed (AE)* and *the proposed (UA)*, according to the input. In the *auto-estimation (AE)* type, the proposed system estimated the optimal iterations and contour as described in Subsections 5.1 and 5.2 with fixed $BBox$ constraints (11% for the long side length; 16% for the short side length). In the *user-assigned (UA)* type, four volunteers participated in the experiments and determined the iterations that they preferred. Each preferred iteration then became the optimal forward and optimal backward iterations for contour optimization. For each test image, the majority cut (iteration) selected by volunteers was added to the UA result pool. When the cuts were diverse, the median iteration from users' selections was added.

For each test image, the indicated rectangle was identical in all methods. Fig. 15 shows four examples from the structural scene dataset. In the statue and glass examples, GrabCut and One-cut results exhibited excessive shrinking. The Box-prior method complemented the deficiencies, and the proposed methods generated fitted contours. In the desk lamp example, GrabCut misclassifed the lamp stand and shaded wall. The lamp stand was not extracted in One-cut result because of the assumption of less color overlap. The proposed (AE) method yielded moderate results with few defects. In the proposed (UA) method, the base of the lamp stand was estimated by using the extended SI-Cut procedure. In the sofa example, the carpet and the sofa are similar in color but different in structure. GrabCut stopped at a contour far from the true foreground. Table 1 lists the error rates of the methods in segmenting the images from the structural dataset.

TABLE 1
Average error rates of methods for the structural scene dataset.

Method	**Error rate (%)**
The proposed (AE)	4.591
The proposed (UA)	3.906
Grabcut	12.332
Box-prior	10.418
One-cut (GW)	9.493
One-cut (IW)	7.292

6.1.2 Comparison Using a Related Dataset

To evaluate the performance of the proposed method when applied to scenes that are less structural or more color-dominated than those in the structural dataset, experiments were conducted using the GrabCut dataset [5]. In this comparison, the *AE* input type was subdivided into *unextended (unext.)* and *extended (ext.)* types. The unextended type was identical to the previous AE type, in which the system automatically estimated the iterations without executing extended procedures. For the extended type, users can turn off the bounding box distance constraints and can enable the extended SI-Cut procedure, but the optimal iterations were still estimated automatically.

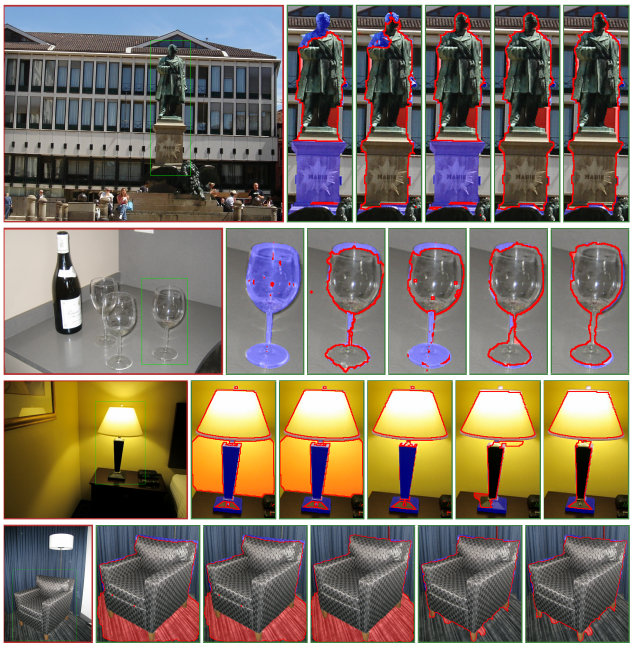


Fig. 15. Comparative results for the structural scene dataset. From left to right: *Input* images and rectangles, *GrabCut* results, *Box-prior* results, *One-cut (IW)* results, *the proposed (AE)* results, *the proposed (UA)* results.

Fig. 16. Comparative results for the GrabCut dataset. From left to right: *Input* images and rectangles, *GrabCut* results (from [6]), *Box-prior* results [6], *One-cut* results [7], *the proposed (AE)* results, *the proposed (UA)* results.

Fig. 16 shows three examples for the GrabCut dataset. Results used for comparison were extracted from the original or subsequent papers of the authors. Table 2 lists the error rates for the entire GrabCut database. As emphasized in [6], the error rates reported here are based on bounding-rectangle inputs. They are not appropriate for comparison with error rates of related methods based on the trimap or scribble inputs.

TABLE 2
Average error rates of methods for GrabCut dataset.

Method	Error rate (%)
The proposed (unext.)	4.693
The proposed (ext.)	4.250
The proposed (UA)	3.756
GrabCut (from [6])	7.2
Box-prior [6]	3.7
One-cut [7]	6.71
One-cut (IW)	5.346

In the unextended type of the proposed method, a bounding box distance constraint identical to that applied to the structural dataset was used, the method performed mostly satisfactory. However, for three test images listed in the supplementary document, the method stopped at inappropriate iterations. When extended SI-Cut was enabled in these cases, the performance of the proposed method was comparable to that of the Box-prior method. Please refer to the supplementary file which provides more comparisons of these methods using these two datasets.

6.2 Performance

The experiments were performed on a desktop computer with an Intel 3.4-GHz CPU and 8-GB memory. Structural prediction was formulated as a multi-labeling problem, and the system included an alpha-expansion library developed by Veksler et al. [4], [25], [26] for label optimization. For the subcomponents related to binary graph-cut problems, including narrow- and wide-band contour optimization, the graph-cut was modified from GrabCut in OpenCV [23]. Currently, the proposed system is implemented in a single thread and does not yet use the multi-core capability.

When SI-Cut was applied to the two aforementioned datasets, the average iteration number for one input image was 9.689. The average optimal backward iteration id (t_{bb}) was 6.189 and can be regarded as the final effective iteration id for the final contour optimization. The average time distribution for one iteration is listed in Table 3. Before exporting the final cut, the system required an average of 2.791 seconds for wide-band contour optimization.

Table 3 illustrates that background structural prediction accounted for approximately 58% of the computations. The reverse prediction required few computations because its reference and target regions were small. The remaining computation time was mainly used for narrow-band graph-cut refinement. It can be roughly thought that the multilevel predictions performed in the proposed method require a computation time 2.93 times of the general graph-cut computation time for one iteration. For an identical image, the iteration number of the proposed method is generally two to four times of the GrabCut iteration

TABLE 3
Average computation time per SI-Cut iteration.

(seconds)	Bg. pred.[a]	Fg. pred. [b]	GC. and others [c]
Time per iter.	6.324	1.712	2.742

a. Background structural prediction.
b. Reverse foreground structural prediction.
c. Other computations. The time is dominated by graph-cut refinement.

number. In the paper proposed Box-prior method [6], the authors reported that their recommended fast pinpoint method is one to fifteen (typical one to four) times slower than the conventional graph-cut method for an identical image. The computation time of One-cut method [7] is approximately one half to six times of the total GrabCut computation time.

6.3 Discussion

According to the results of the experiments, the advantages and limitations of the four methods are discussed as follows. GrabCut is among the most commonly used extraction tools and the first rectangular-input-based segmentation method. It is efficient and effective for images with distinctive GMMs. When the distributions are ambiguous, the regions can be excessively shrunk or the iterative procedures can stop at an early stage. The Box-prior method entails applying a bounding box prior to the GrabCut framework and solving the excessive shrinking problem. This method generates remarkable results for the GrabCut dataset, but the pinpoint strategy is mainly dependent on the color distributions.

The recently developed One-cut method involve penalizing overlap between foreground and background histograms. This new assumption makes rectangle-input-based segmentation determinable in one iteration. The results of the experiments showed that the One-cut method can generate several impressive results for both datasets with fine histogram bins, but the results varied according to the parameters. Therefore, in the reproduced One-cut (IW) method, we fixed the bin number at 128^3 or 256^3 and exhaustively searched the weights for individual images.

Unlike the related methods, the proposed method has a novel SI feature addressing the aforementioned situation in which colors are intermixed. The proposed framework performed satisfactorily in segmenting structural scenes. Since the proposed method does not rely heavily on strong neighbor links a graph, the method typically obtains more detailed contours. A notable advantage of segmentation by SI regions is that its results are insensitive to different rectangle sizes for scenes containing nearly homogeneous structures. As the officer and church examples shown in the supplementary file (sup. Fig. 8, 11), even applying indicated rectangles which are several times larger than the original ones, the segmented results are still satisfactory. By contrast, the One-cut method had to apply different weights to reach similar qualities, and the Box-prior method was not applied to these cases since they do not conform to the bounding box prior assumption. Moreover, since the proposed region exclusion is resisted by reverse (foreground) structural prediction, the proposed method can also tolerate slight contraction of rectangles (sup. Fig. 8 to 11). On the other hand, when an indicated rectangle is too large and covers other objects (e.g the white flower above the target in sup. Fig. 10) or a rectangle is too small (e.g. the Church facade in sup. Fig. 11), the proposed method still fits the contours for the high inconsistent regions.

The proposed method has a few limitations. The iteration analyzer tends to find the iterations of large SI value change without other prior knowledges. In most cases, the AE iterations were similar to the UA ones. However, for the boat case in sup. Fig. 7, the auto analyzer chose an more inconsistent contour and excluded the left sail. For this problem, it is possible to include additional criteria (e.g. region size penalty) or to generate multiple contour candidates for users or other automatic recognition systems to choose. Besides, the background structures of a few images are difficult to be accurately predicted from the references outside the rectangles, the exclusion progress can become inefficient. Notwithstanding, the results of the proposed method are still comparable to those of other methods.

There are several possible extensions of the proposed work. The SI-Cut framework is flexible and exhibits high potential to cooperate with other methods. Several methods and concepts in segmentation according to strokes or other inputs, such as the geodesic distance in [8], fitting structure tensors in [9], and the level set and edge field in [11], may be applicable in the local contour refinement. The proposed SI features can also complement other features in related frameworks. In addition, inconsistency estimation can be improved by employing more advanced or efficient structural prediction methods in the future. Another possible future work is for saliency detection. Tang et al. [7] combined their histogram overlap criterion with the smoothness and the saliency data term estimated by [17] for a salient object database [27][28]. By contrast, the proposed work currently requires a part of definitely background regions for structural prediction, and it cannot directly be applied to improve the saliency map. It is worthwhile studying how to apply the SI concept to saliency detection.

7 Conclusion

This paper proposes analyzing structural inconsistency in images to execute foreground extraction. This novel method integrates efficient image completion

and graph labeling techniques into a powerful framework. The method involves considering not only colors but also textures and their localities. It can be automatic or intervened by users. The results of the experiments showed that the proposed system can extract accurate contours from input images with an indicated rectangle. It is highly effective in segmenting structural scenes and is still comparable to related methods in segmenting less structural scenes.

ACKNOWLEDGMENTS

The authors appreciate the helpful comments from the anonymous reviewers. This paper was partially supported by the Ministry of Science and Technology, Taiwan under grant no. MOST 103-2221-E-009-143.

REFERENCES

[1] P. F. Felzenszwalb, D. A. McAllester, and D. Ramanan, "A discriminatively trained, multiscale, deformable part model," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2008, pp. 1–8.
[2] C. Rother, T. Minka, A. Blake, and V. Kolmogorov, "Cosegmentation of image pairs by histogram matching - incorporating a global constraint into MRFs," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2006, pp. 993–1000.
[3] Y. Boykov and M.-P. Jolly, "Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images," in *Proc. Intl. Conf. on Computer Vision*, vol. 1, 2001, pp. 105–112 vol.1.
[4] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts." *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 11, pp. 1222–1239, 2001.
[5] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut: Interactive foreground extraction using iterated graph cuts," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 309–314, Aug. 2004.
[6] V. S. Lempitsky, P. Kohli, C. Rother, and T. Sharp, "Image segmentation with a bounding box prior," in *Proc. Intl. Conf. Computer Vision*, 2009, pp. 277–284.
[7] M. Tang, L. Gorelick, O. Veksler, and Y. Boykov, "Grabcut in one cut," in *Proc. Intl. Conf. on Computer Vision*, 2013, pp. 1769–1776.
[8] B. L. Price, B. S. Morse, and S. Cohen, "Geodesic graph cut for interactive image segmentation," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2010, pp. 3161–3168.
[9] H. Zhou, J. Zheng, and L. Wei, "Texture aware image segmentation using graph cuts and active contours," *Pattern Recognition*, pp. 1719–1733, 2013.
[10] C. Nieuwenhuis and D. Cremers, "Spatially varying color distributions for interactive multilabel segmentation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 35, no. 5, pp. 1234–1247, 2013.
[11] Y. Liu and Y. Yu, "Interactive image segmentation based on level sets of probabilities," *IEEE Trans. Visualization and Computer Graphics*, vol. 18, no. 2, pp. 202–213, 2012.
[12] Y. Li, J. Sun, C.-K. Tang, and H.-Y. Shum, "Lazy snapping," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 303–308, Aug. 2004.
[13] T. Torsney-Weir, A. Saad, T. Moller, H.-C. Hege, B. Weber, and J.-M. Verbavatz, "Tuner: Principled parameter finding for image segmentation algorithms using visual response surface exploration," *IEEE Trans. Visualization and Computer Graphics*, vol. 17, no. 12, pp. 1892–1901, 2011.
[14] S. Goferman, L. Zelnik-Manor, and A. Tal, "Context-aware saliency detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 34, no. 10, pp. 1915–1926, 2012.
[15] X. Shen and Y. Wu, "A unified approach to salient object detection via low rank matrix recovery," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2012, pp. 853–860.
[16] M.-M. Cheng, G.-X. Zhang, N. J. Mitra, X. Huang, and S.-M. Hu, "Global contrast based salient region detection," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2011, pp. 409–416.
[17] F. Perazzi, P. Krahenbuhl, Y. Pritch, and A. Hornung, "Saliency filters: Contrast based filtering for salient region detection," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2012, pp. 733–740.
[18] Q. Yan, L. Xu, J. Shi, and J. Jia, "Hierarchical saliency detection," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2013, pp. 1155–1162.
[19] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting," in *Proc. ACM SIGGRAPH*, 2000, pp. 417–424.
[20] A. Criminisi, P. Perez, and K. Toyama, "Object removal by exemplar-based inpainting," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 2, 2003, pp. II–721–II–728.
[21] Y. Pritch, E. Kav-Venaki, and S. Peleg, "Shift-map image editing," in *Proc. Intl. Conf. Computer Vision*, 2009, pp. 151–158.
[22] S. Darabi, E. Shechtman, C. Barnes, D. B. Goldman, and P. Sen, "Image melding: Combining inconsistent images using patch-based synthesis," *ACM Trans. Graph.*, vol. 31, no. 4, p. 82, 2012.
[23] G. Bradski et al., "The OpenCV Library," http://opencv.org/.
[24] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman, "Labelme: A database and web-based tool for image annotation," *Intl. J. Comput. Vision*, vol. 77, no. 1-3, pp. 157–173, May 2008.
[25] V. Kolmogorov and R. Zabih, "What energy functions can be minimized via graph cuts," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 26, pp. 65–81, 2004.
[26] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision." *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 26, no. 9, pp. 1124–1137, 2004.
[27] R. Achanta, S. S. Hemami, F. J. Estrada, and S. Susstrunk, "Frequency-tuned salient region detection," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2009, pp. 1597–1604.
[28] T. Liu, J. Sun, N. Zheng, X. Tang, and H.-Y. Shum, "Learning to detect a salient object," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2007, pp. 1–8.

I-Chen Lin received the B.S. and Ph.D. degree in computer science from National Taiwan University, in 1998 and 2003, respectively. In 2005, he joined Dept. of Computer Science and Inst. of Multimedia Engineering, National Chiao Tung University. He is currently an associate professor. His research interests include computer graphics, vision, and interactive multimedia systems. He is a member of IEEE and ACM SIGGRAPH.

Yu-Chien Lan received the B.S. and M.S. degree in computer science and multimedia engineering from National Chiao Tung University, Taiwan, in 2011 and 2013, respectively. In 2014, she joined Taiwan Semiconductor Manufacturing Company, Limited. Her research interests include computer graphics and image synthesis.

Po-Wen Cheng received the B.S. degree in computer science from National Chiao Tung University, Taiwan, in 2014. He is now a graduate student in the Institute of Multimedia Engineering, National Chiao Tung University. His research interests include computer vision and image processing.