Transductive Object Cutout *

Jingyu Cui[1], Qiong Yang[3], Fang Wen[2], Qiying Wu[4], Changshui Zhang[1], Luc Van Gool[3], and Xiaoou Tang[2]

[1]Tsinghua University, Beijing, China
cui-jy@mails.thu.edu.cn, zcs@mail.thu.edu.cn

[2]Microsoft Research Asia, Beijing, China
{fangwen, xitang}@microsoft.com

[3]ESAT/PSI-IBBT, K.U.Leuven, Belgium
{Qiong.Yang, Luc.VanGool}@esat.kuleuven.be

[4]Xiamen University, Fujian, China
wuqiying@fj.chinamobile.com

Abstract

In this paper, we address the issue of transducing the object cutout model from an example image to novel image instances. We observe that although object and background are very likely to contain similar colors in natural images, it is much less probable that they share similar color configurations. Motivated by this observation, we propose a local color pattern model to characterize the color configuration in a robust way. Additionally, we propose an edge profile model to modulate the contrast of the image, which enhances edges along object boundaries and attenuates edges inside object or background. The local color pattern model and edge model are integrated in a graph-cut framework. Higher accuracy and improved robustness of the proposed method are demonstrated through experimental comparison with state-of-the-art algorithms.

1. Introduction

Object cutout is a fundamental task in computer vision. However, automatic object cutout in arbitrary settings remains a formidable challenge [4, 11, 14, 10, 9, 15].

User interaction can be of great help, but one may need to cut out objects from a large number of images. Existing interactive object cutout tools [15, 4, 9, 11, 10] focus on one image, and therefore require manual interaction for each image, which is prohibitively time consuming. Automatic cutout in every image seems to be quite impossible still, as it would require generic rules.

One possible solution is to start from just one or a few typical image examples with interactive tools, and then transduce the cutout model to other related images for automated object cutout there. In this case, the key issue is how to perform such transduction. This is what we address in this paper. Thus, although there exist several algorithms [7, 8, 1, 2] which used inductive learning for object segmentation, we propose to use transductive learning for cutting out objects from groups of images. We borrow the term "transductive" from the machine learning area here to contrast it with these inductive learning methods. That is, we opt for a scheme which transduces information directly from example to example, instead of trying to learn the general rule as an intermediate step.

1.1. Related Work

Existing object cutout algorithms [4, 9, 11, 12, 13] are generally formulated in a graph-cut framework, and solved by minimizing a Markov Random Field energy through a min-cut/max-flow algorithm [3]. They combine two aspects: the likelihood term to model appearance of foreground/background, and the smoothness term to impose the prior that adjacent pixels with similar color tend to have the same label.

For the likelihood term, most existing algorithms [4, 9, 11] represent foreground and background by the color values of their pixels. However, it is very likely in natural images that object and background contain similar colors, which causes ambiguity in separating foreground from background. For example, in Fig. 3(b)[1], the color of the face in the foreground is similar to that of the bricks in the background. This gives the facial region a low probability of being foreground as shown in Fig. 3(c).

Rother *et al*. [12] represent the whole foreground with a histogram, and look in two images for regions whose histograms are most closely matched. Since a histogram is only a coarse representation, and lacks discriminability, it is very likely that the histogram of an object might change only slightly even if the segmentation result has changed vastly, and thereby cannot give an accurate result. That is

*This work was done when Jingyu Cui, Qiong Yang, and Qiying Wu were visiting/working at Microsoft Research Asia.

[1]All the figures are best viewed under color.

978-1-4244-2243-2/08/$25.00 ©2008 IEEE

why the bricks beside the arm are wrongly labeled to be foreground in row 3 of Fig. 4(e).

Spatial color correlation is used in CBIR to encode the spatial configuration based on histograms. One of the typical tools is the correlogram [6], which measures the concurrence probability of pixel pairs at a predefined distance. This feature requires a scale prior to indicate the distance, thus cannot be easily used in object cutout problems where the scale changes severely.

Patch-based representations can discriminate very subtle differences between patches, but lack robustness. Schnitman *et al*. [13] select template patches from the example image, and patches from a novel input image are compared with the templates in a pixelwise manner. This representation requires that the two images are highly similar in illumination, resolution, scale and scene. For instance, in row 3 of Fig. 4(d), with a slight change of lighting on the face, this method fails by mismatching the face to the sunlit bricks in the training image (a), and therefore gives a wrong segmentation around the face region.

For the smoothness term, traditional methods [4, 9, 11, 13, 12] calculate it only according to color contrast, and tend to cut along strong edges. If there is a strong edge inside the object, or a weak edge along the boundary, segmentation errors will occur.

Background cut [14] solves this problem for static backgrounds, by attenuating the contrast where the color of a pixel is similar to the color of the corresponding pixel in the calibrated background image. However, this algorithm cannot be directly used in transductive object cutout, since it relies strongly on the static background.

1.2. Observations and Our Method

Traditional methods are not capable of transducing object cutout from an example image to novel image instances, mainly because of lacking the ability to discriminate similar colors in foreground and background, and lacking robustness in handling possible changes of object scale, rotation and view point. So in this paper, we aim at designing an algorithm which is both more discriminative and more robust.

To enhance the discriminability, we propose to exploit the *color configuration* information, since it is much less likely that foreground and background share similar color configurations, thus ambiguity is greatly reduced. Here *color configuration* means what distinctive colors surround a certain color. For example, two images containing the same object but with different positions and scales are shown in Fig. 1. We label the doll in (a) as the foreground object, and want to cut out the same doll in (b). However, since regions 1 and 3 are both blue, it is ambiguous. Considering the color configuration, they can be easily discriminated since 1 is surrounded mainly by white color; while 3 has lots of yellow around it. This color configuration does not rely on the scale of the object. E.g. although the doll in Fig. 1(b) is much smaller than that in Fig. 1(a), the color configuration surrounding region 3 does not change. So the robustness is embodied in design of color configuration extraction.

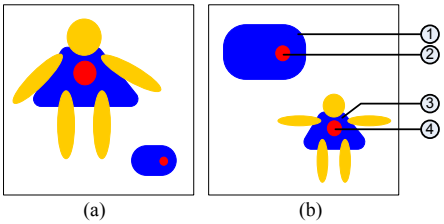

Figure 1. Toy sketch showing our key observations. Color ambiguity can be greatly reduced by considering color configuration (e.g. region 1 and 3). Edge information can help in places where color configuration fails (e.g. region 2 and 4).

For regions which even color configuration cannot distinguish, e.g. regions 2 and 4, edge information can help. We know from the example image (a) that a blue-red-paired edge only appears inside objects, so we are confident to attenuate the edge around region 2 and 4 in (b), thereby avoiding the cut along this edge.

Based on the observations above, we develop a color pattern model and an edge profile model for our transductive object cutout algorithm.

1. Local Color Pattern (LCP) Model

   We extract the local color pattern characterizing the spatial color configuration by searching for the nearest distinctive colors along certain directions. This is more invariant to the scale of color patches than correlogram features. It is also much more discriminative than color or histogram features, since contextual color information is included.

2. Edge Profile Model

   We extract the edge profile features in the direction normal to the edge, and use them to discriminate foreground silhouettes (boundary edges) from edges inside foreground or background (interior edges). The features are invariant to the rotation of objects. By enhancing possible boundary edges and attenuating possible interior edges, the cutout path is driven to follow the true object boundary.

We integrate both LCP model and edge profile model in a graph-cut framework, and get higher accuracy in the cutout results due to their complementary contributions.

2. Algorithm

2.1. Problem Formulation

We formulate the object cutout as a binary labeling problem, and solve it by minimizing the Gibbs energy $E(X)$ of a Markov Random Field (MRF) on a graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$:

$$E(X) = \sum_{i \in \mathcal{V}} E_1(y_i) + \lambda \sum_{(i,j) \in \mathcal{E}} E_2(y_i, y_j) \qquad (1)$$

where $\mathcal{V}$ is the set of all pixels and $\mathcal{E}$ is the set of all arcs connecting adjacent pixels. $y_i \in \{0, 1\}$ is the label for each pixel $p_i \in \mathcal{V}$, where $y_i = 1$ for pixel p_i belonging to the foreground, while $y_i = 0$ for the background. $E_1(y_i)$ is the likelihood energy denoting the cost of labeling pixel p_i with y_i, and $E_2(y_i, y_j)$ is the smoothness energy penalising differently labelled adjacent pixels. λ is a parameter to balance the two terms.

In this paper, we propose a novel local color pattern based appearance model for the likelihood term, and learn an edge profile model to modulate the smoothness term.

2.2. Likelihood via Local Color Pattern Model

2.2.1 Local Color Pattern Extraction

As discussed in Sec. 1.1, we define the Local Color Pattern (LCP) as the color configuration which reflects the spatial distribution of distinctive colors, e.g. skin color of the face surrounded by black color of the hair in Fig. 2.

Note that this color configuration is not a correlogram, which is defined for a specific spatial distance [6]. Also, it is different from colors of neighboring regions obtained by oversegmentation methods such as MeanShift [5]: in case of oversegmentation, one color homogeneous region may be divided into many pieces (such as the face of the girl in Fig. 2), and the neighboring pieces in the same homogeneous region cannot provide configuration information; on the other hand, increasing the color radius parameter may merge regions with different colors together, which is also undesirable.

Now the key issues in the LCP extraction are: 1) find what colors are distinctive; 2) avoid the introduction of a scaling factor. Here, we design a soft threshold scheme to find the distinctive colors to let it adapt to different images, and search along predefined directions until it reaches a region with a different distinctive color to make it scale-invariant.

Gaussian Mixture Model (GMM) clustering of pixel colors is carried out to estimate how many *color modes* exist in the image and what they are. In this way, the color space of all pixels $\mathcal{C}$ is divided into several non-overlapping color modes: $\mathcal{C} = \bigcup_n \mathcal{C}_n$. This division gives a general and robust view of what colors in the image are close (in the same mode) or different (in different modes). We use the HSV color space.

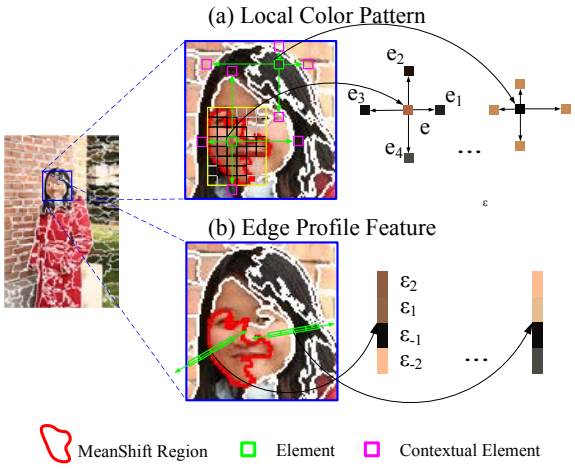

Figure 2. Local Color Pattern and Edge Profile Feature Extraction.

After that, we oversegment the image with MeanShift [5]. Considering the large variation of the shapes and sizes of MeanShift regions, we divide each region R into *elements*, then extract the LCP feature based on the estimated color modes of the elements. Take Fig. 2(a) as an example. For the MeanShift region with a red boundary (partial face of the girl), its bounding box (yellow rectangle) is divided into a grid, forming *elements*. For each *element* e, we search along D predefined directions for *contextual elements*, denoted by $e_1, ..., e_D$. The *contextual element* is defined as the nearest *element* that belongs to a *color mode* different from that of e in the predefined direction, thus the search can reach beyond the MeanShift region boundary, and get to the real distinctive color to form a color pattern. E.g. in Fig. 2(a), $D = 4$, contextual elements $e_1, ..., e_4$ are obtained for element e. Searching for e_1 reaches beyond the region boundary, and gets to the region of hair, which is the true contextual color to form the color pattern.

Finally, we form the *local color pattern* $\mathbf{p}$ for element e as: $\mathbf{p}(e) = [c_0, c_1, ...c_D]^{\mathrm{T}}$, where c_0 is the mean color of element e, and $c_1, ...c_D$ are mean colors of its contextual elements $e_1, ..., e_D$.

2.2.2 Infer the Likelihood Energy

Modeling Local Color Pattern

For the labeled example image, we get LCP features for all foreground and background elements, and use GMM to fit the LCP likelihood model of foreground $l_F(\mathbf{p}) = p(\mathbf{p}|\, y_e = 1)$ and background $l_B(\mathbf{p}) = p(\mathbf{p}|\, y_e = 0)$. Here y_e denotes the label of element e. Taking a non-informative prior on foreground and background, we get a posterior proportional to the likelihood by the same constant, i.e. $p_{F,B}(\mathbf{p}) \propto l_{F,B}(\mathbf{p})$.

(a) (b) (c) (d) (e) (f) (g)

Figure 3. Intermediate results. (a) is the example image with manually labeled object boundaries represented by red curves, and is used to cut out foreground in (b). Our algorithm gets the final result (g). (c) and (d) are probability maps using single color and LCP, respectively. Brighter pixels have higher probability of being foreground. (e) and (f) are contrast map and modulated contrast map, respectively. Higher intensity means larger contrast with smaller penalty in cut. In both maps, the contrast values are normalized for visualization.

Under the assumption that colors of contextual elements $c_i, i = 1, ..., D$ are conditionally independent given the center color c_0, approximations are made to more easily fit the high dimensional model $l(\mathbf{p})$ by decomposing it into the multiplication of lower dimensional models:

$$l(\mathbf{p}) = p(c_0, c_1, ..., c_n) \approx p(c_0) \prod_{i=1}^{D} p(c_i | c_0) \quad (2)$$

Optionally, the fitting can be further simplified according to the color modes we obtained:

$$l(\mathbf{p}) = \sum_j \left[p(c_0 \in \mathcal{C}_j) \prod_{i=1}^{D} p(c_i | c_0 \in \mathcal{C}_j) \right] \quad (3)$$

Inferring Likelihood Energy for Novel Images

For a novel image, we extract LCP for each element, and calculate $p_{F,B}(\mathbf{p})$ from $l_{F,B}(\mathbf{p})$ using Eq.(3).

When calculating the probability of pixel p_i belonging to foreground/background, we carry out the median process inside each region R to increase robustness, since a small change does not affect the result on the whole region:

$$\underset{\forall p_i \in R}{p_{F,B}}(p_i) \leftarrow \underset{\forall \mathbf{p} \in R}{median} [p_{F,B}(\mathbf{p})] \quad (4)$$

The likelihood energy is finally obtained by normalization:

$$E_1(y_i = 1) = \frac{\log [p_F(p_i)]}{\log [p_F(p_i)] + \log \{p_B(p_i)\}} \quad (5)$$
$$E_1(y_i = 0) = 1 - E_1(y_i = 1)$$

Fig. 3(d) shows the probability map of our method. It is obvious that the ambiguity in regions with similar colors (e.g. face and wall) is reduced compared with Fig. 3(c), which uses the color of pixels as the feature.

Note that LCP provides a more eleborate representation than color histograms [12] and color values of pixels [4, 9, 11], and thereby it gives a more accurate result than these methods. There is also no scaling parameter involved, so it outperforms correlograms [6] when the scale changes largely. Additionally, the conditional independence assumption between the colors of contextual elements (see Eq.(2)) not only simplifies the learning process, but also makes it more robust to rotation. Only the distinctive colors are counted in the extraction of LCP, thereby it is robust to small color changes. The median process inside each region in Eq.(4) further increases its robustness to spatial shifts between object and background, and viewpoint changes of the object.

2.3. Contrast Modulation by Edge Profile Model

The smoothness term used in traditional graph-cut based segmentation methods is based on image contrast [11], i.e.

$$E_2(y_i, y_j) = |y_i - y_j| \exp(-\beta d_{ij}) \quad (6)$$

where β is a parameter to weight the color distance d_{ij}. Here $d_{ij} = \|I_i - I_j\|$, and $\beta = [2\langle d_{ij} \rangle]^{-1}$ with the expectation operator $\langle \cdot \rangle$ (The expectation is calculated as the average over lots of randomly sampled data). I_i and I_j are colors of p_i and p_j.

This term forces the segmentation to follow strong edges. However, when there exists a strong interior edge inside the foreground or background, or a weak boundary edge, undesirable segmentation occurs. For instance, in row 4 of Fig. 4, there is a strong edge between the clothes and legs of the boy; Methods that use the traditional smoothness term (Fig. 4 column (f)) fail by cutting along this edge.

In this section, we solve this problem by modulating d_{ij} based on a rotation invariant edge profile feature. The modulation reduces d_{ij} at interior edges and increases it at boundary edges, thereby guiding the cutout along the boundary edge.

(a) (b) (c) (d) (e) (f)

Figure 4. Comparison with state-of-the-art algorithms. (a) is the example image with manually labeled object boundaries represented by red curves, and is used to cut out (b). Our algorithm gets the final result (c). (d), (e) and (f) are the results with S-Induct [13], Co-Seg [12] and GrabCut [11], respectively. Green rectangles in Column (f) indicate the interaction by the user used in GrabCut [11].

2.3.1 Edge Profile Feature Extraction

We take shared borders of MeanShift regions as edge segments, and extract profiles along them to describe the color appearance normal to the edge. This profile is rotation-invariant. For example, in Fig. 2(b), starting from a pixel pair on the edge between two adjacent regions, N distinctive colors are found in the normal direction of the edge (green arrows). In total $2N$ colors are collected to form the profile feature: $\boldsymbol{\varepsilon} = [\varepsilon_{-N}, ..., \varepsilon_{-1}, \varepsilon_1, ..., \varepsilon_N]^{\mathrm{T}}$. In Fig. 2, $N = 2$. Note that the distinctive colors are also based on the color modes of the regions lying on the normal line, similar to Sec. **??**. This means that it searches along the normal line until it reaches the region which has a different color mode. Thereby, it is also scale-invariant.

2.3.2 Modulate the Smoothness Energy

Similar to the method used in Sec. 2.2.2, likelihood models for boundary edges and interior edges are fitted as follows:

$$l(\boldsymbol{\varepsilon}) = p(\varepsilon_1)\, p(\varepsilon_{-1}|\varepsilon_1) \prod_{i=1}^{N-1} p(\varepsilon_{i+1}|\varepsilon_i) \prod_{i=1}^{N-1} p(\varepsilon_{-i-1}|\varepsilon_{-i}) \tag{7}$$

either for $l_B(\boldsymbol{\varepsilon})$ (boundary edge) or $l_I(\boldsymbol{\varepsilon})$ (interior edge).

This simplification is based on the approximation that only adjacent colors in edge profiles are dependent. Our feature is extracted in two directions in a symmetric manner, thus the first two terms in the equation can be equivalently changed to $p(\varepsilon_{-1})\, p(\varepsilon_1|\varepsilon_{-1})$.

In the novel image, for any adjacent pixel pair (p_i, p_j) at edge ε between regions R_1 and R_2, we can get $p_B(\boldsymbol{\varepsilon})$ and $p_I(\boldsymbol{\varepsilon})$ by $p_{I,B}(\boldsymbol{\varepsilon}) \propto l_{I,B}(\boldsymbol{\varepsilon})$.

The final posterior of the pixel pairs (p_i, p_j) at the sharing borders of regions R_1 and R_2 are obtained through a

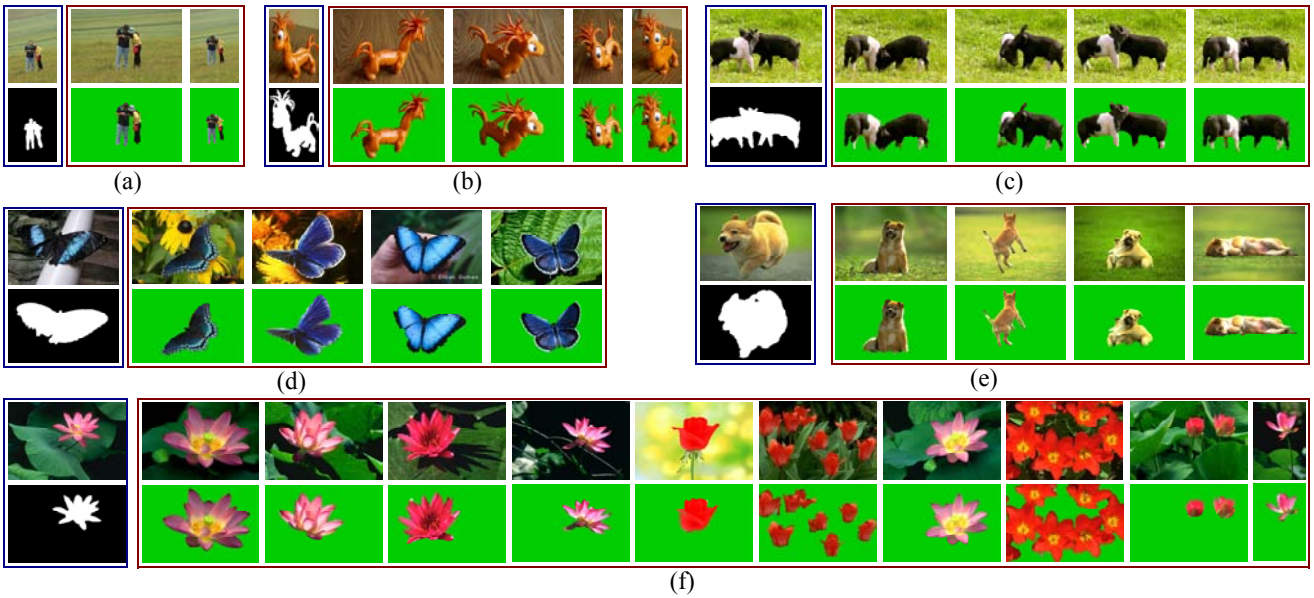

Figure 5. Robustness Test. Given the example image (first column in each group), our algorithm shows robustness when camera zooms or scale change (a)(e), view point change (b)(d)(e), multi-object occlusion (c), and some foreground/background color change (d)(e)(f).

robust voting over all pairs at the sharing border between R_1 and R_2:

$$\underset{\forall p_i \in R_1, p_j \in R_2}{p_{I,B}} (p_i, p_j) \leftarrow \underset{\substack{\forall \varepsilon=(p,q) \\ p \in R_1, q \in R_2}}{median} [p_{I,B}(\boldsymbol{\varepsilon})] \quad (8)$$

We use $p_I(p_i, p_j)$ and $p_B(p_i, p_j)$ to modulate the contrast. When $p_I(p_i, p_j)$ is large enough, the edge between p_i and p_j is very likely to be an interior edge, and we reduce d_{ij}. When $p_B(p_i, p_j)$ is large enough, we are confident that the edge is a boundary edge, thus increase d_{ij}. Other ambiguous areas are kept unchanged:

$$d_{ij} = \begin{cases} \|I_i - I_j\| \left(\frac{p_B(p_i,p_j)}{p_I(p_i,p_j)} \right)^{\alpha}, & \left| \log \frac{p_B(p_i,p_j)}{p_I(p_i,p_j)} \right| > \delta, \\ \|I_i - I_j\|, & otherwise. \end{cases} \quad (9)$$

where $\alpha > 0$ controls the intensity of modulation, $\delta > 0$ is a confidence threshold for robustness.

Fig. 3(f) shows the contrast map of our method. After modulation of contrast, edges inside foreground and background are generally attenuated, while edges along foreground boundaries are enhanced compared with the original contrast map Fig. 3(e).

3. Experiments

3.1. Comparison with State-of-the-Art Methods

To verify the proposed approach, we compare with contemporary methods on a variety of images, which are real photos (traveling, friends gathering, and home activities). Three state-of-the-art algorithms are implemented for comparison, including two automatic methods: semantic induction (S-Induct) of Schnitman *et al.* [13], co-segmentation (Co-Seg) of Rother *et al.* [12], and one with user interaction: GrabCut [11]. The results are shown in Fig. 4.

Note that the three state-of-the-art algorithms are carried out under optimal conditions: For S-Induct, we use the whole sample set instead of choosing a subset of samples stated in [13], to remove the effect of insufficient sampling. For Co-Seg, the best parameter is given by providing the ground truth scale of the novel input image in the first iteration. For GrabCut, the rectangle is given to be nearby the bounding box of the foreground object.

In our algorithm, we use fixed experimental parameters for all images: $D = 4$, $N = 2$, $\lambda = 1$, $\alpha = 0.5$, $\varepsilon = \log(0.7/0.3) \approx 0.37$. For GMM fitting, we use a modified EM algorithm to start with a relatively big number of clusters, and conduct merging after clustering to adaptively determine the number of Gaussian components. For example, in our experiments to determine the color modes, the final number of clusters will typically converge to 15-20 after initially set to be 40.

From the results, it is obvious that our algorithm outperforms both S-Induct [13] and Co-Seg [12]. Please note that all these images are rather difficult for the object cutout task, since many parts of the foreground and background share similar colors. For instance, in row 1 in Fig. 4, the face of the boy is very similar in color to the face of the girl, which belongs to the background. This is the reason why S-Induct and Co-Seg incorrectly label the boy's face to be

background. However, the local color patterns of the two faces are distinctively different, with a white region below the girl's face and a yellow region below the boy's. Using our LCP model, the correct result is obtained.

Although both S-Induct and Co-Seg use a smoothness term, neither of them can correct the mistake on the boy's face, since the edge between the face and the yellow clothes is strong. Our edge profile model can effectively attenuate this edge by giving high probability that it is an interior edge, thereby strongly constraining the label of the boy's face to be foreground.

We also conduct an experiment with GrabCut on these examples. Although GrabCut can incorporate user interaction, the strong edge inside the object (e.g. column (f), row 4 and 5) and the similar color shared by foreground and background (e.g. column (f), row 2 and row 3) cause unsatisfactory results. Our method can handle these situations well.

Note that here we are demonstrating the effectiveness of the proposed feature and algorithm, not comparing the whole framework. Even if comparing the framework, although Co-Seg and GrabCut have their own advantages, when the problem comes to segmenting a lot of similar images, such as the case in Figure 7, our method will be much more time saving.

3.2. Robustness Test

Our algorithm also shows robustness to camera zoom or scale change (e.g. Fig. 5(a)(e)), view point change (e.g. Fig. 5(b)(d)(e)), multi-object occlusion (e.g. Fig. 5(c)), and foreground/background color change to some extent (e.g. Fig. 5(d)(e)(f)) in our experiments. This owes to the following aspects.

First, the LCP feature is obtained by searching for the nearest distinctive colors, thus it is robust to the scale change. It is also robust to rotation, spatial shift and viewpoint change to some extent due to the voting(median) mechanism inside the regions (see Eq.(4)) and the conditional independence assumption (see Eq.(2)). Second, the edge profile feature is rotation invariant, since it is obtained along the normal direction of the local edge. (Please refer to our supplementary materials for more results.)

3.3. Failure Cases

The algorithm might fail when there is a big change in the context (see Group 1 in Fig. 6). Another case is shown in Group 2 in Fig. 6, where a novel color or edge pattern appears.

3.4. Applications

Object Cutout for Group of Images. As taking pictures becomes easy, we usually have many photos at hand. We first cluster these photos into several groups (e.g. in Fig. 7, by using correlogram feature [6], the images are clustered into 3 groups). For each group, we select one image to start with (the first image in each group), and use the interactive tool [4] to get their object cutout results. Then our approach automatically cuts out objects for other photos in the same group, which greatly reduces the manual labor.

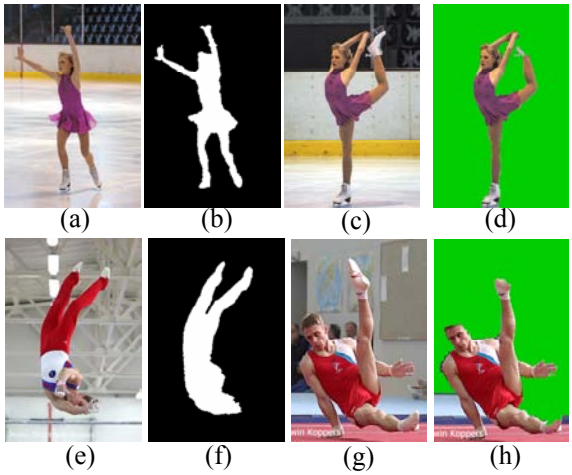

Figure 6. Failure Cases. Group 1: Left foot hold high in (c) was wrongly labeled as background because of the big change in context (shoe surrounded by black background in (c) instead of white ice in (a)). Edge model doesn't help because color pairs (skin color, white) exist in both interior edges and boundary edges. Group 2: Red and blue region in the background of (g) was incorrectly labeled as foreground because it does not exist in the background of the example image (e) but similar to its foreground.

Video Object Cutout. Without a temporal coherence assumption, our algorithm can be useful when there is sharp movement, viewpoint change or articulation in the object, as long as the color model keeps consistent, as shown in Fig. 8.

Side-view Face detection. Our algorithm might also be helpful for other tasks such as side-view face detection, which is quite challenging for existing face detection algorithms using only facial features. Given that spatial configuration between the body and the face is consistent, we can infer the location of the face from the object cutout result. Fig. 9 gives an example.

4. Future Work

In the future, we would like to integrate our algorithm into an active learning framework. The first application in Sec. 3.4 is just initial work. Following that, we will model the image set as a graph, where each image is modeled as a node and the affinity between two images is modeled as the edge connecting the two nodes. First, we study how to cluster the image set into subsets and how to select the

Figure 7. Object cutout for group of images. The images are first clustered into 3 groups. Then in each group one image is selected as an example (marked by the red rectangle). We manually cut out the object from the example image, using the tool [4]. The objects in the other images are automatically cut out as shown.

Figure 8. Video object cutout. The first frame is manually labeled with boundary marked by blue curve. Cutout results of typical succeeding frames are marked by red curves.

(a) (b) (c)

Figure 9. Object cutout for side view face detection. Using the image (a), its predefined object boundary (red curve) and the face detection result (blue rectangle), we get both the cutout result and face detection result of image (b) in (c).

most typical image in each subset to start with. Second, we study the transduction on the graph level, not just from one example to another; third, we study the online learning of transductive model when the user refines the results and new typical images are given. In this way, we expect to cut out objects from a large number of images with as few manual interactions as possible. Other future work is to incorporate texture and shape information despite of the lower speed.

5. Acknowledgement

The authors sincerely thank Chao Wang for preliminary experimental implementation. Additionally, thanks to Jan-Hendrik Becker for the constructive discussion and proof-reading.

References

[1] L. Bastian, L. Alex, and B. Schiele. Combined object categorization and segmentation with an implicite shape model. *ECCV workshop on SLCV*, 1, 2004.

[2] E. Borenstein and S. Ullman. Class-specific, top-down seg- mentation. In *ECCV*, pages 109–124, 2002.

[3] M.-Y. Boykov and M.-V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans. on PAMI*, 26:1124–1137, 2004.

[4] Y. Y. Boykov and M.-P. Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in n-d images. In *Proceedings of ICCV '01*, 2001.

[5] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. on PAMI*, 24(5):603–619, 2002.

[6] J. Huang, S. R. Kumar, M. Mitra, W.-J. Zhu, and R. Zabih. Image indexing using color correlograms. In *Proceedings of CVPR '97*, page 762, 1997.

[7] M. P. Kumar, P. H. S. Torr, and A. Zisserman. OBJ CUT. In *Proceedings of CVPR '05*, 2005.

[8] A. Levin and Y. Weiss. Learning to combine bottom-up and top-down segmentation. In *ECCV '06*, 2006.

[9] Y. Li, J. Sun, C.-K. Tang, and H.-Y. Shum. Lazy snapping. In *ACM SIGGRAPH '04*, pages 303–308, 2004.

[10] E. N. Mortensen and W. A. Barrett. Interactive segmenta- tion with intelligent scissors. In *Graphical Models and Image Processing*, pages 349–384, 1998.

[11] C. Rother, V. Kolmogorov, and A. Blake. "GrabCut": interactive foreground extraction using iterated graph cuts. In *ACM SIGGRAPH '04*, pages 309–314, 2004.

[12] C. Rother, T. Minka, A. Blake, and V. Kolmogorov. Cosegmentation of image pairs by histogram matching - incorporating a global constraint into mrfs. In *Proceedings of CVPR '06*, pages 993–1000, 2006.

[13] Y. Schnitman, Y. Caspi, D. Cohen-Or, and D. Lischinski. Inducing semantic segmentation from an example. In *Proceedings of ACCV '06*, 2006.

[14] J. Sun, W. Zhang, X. Tang, and H. Shum. Background cut. In *Proceedings of ECCV '06*, 2006.

[15] V. Vezhnevets and V. Konouchine. "GrowCut" - Interactive Multi-Label N-D Image Segmentation By Cellular Automata. In *Graphicon*, 2005.