CONTENT-SENSITIVE COLLECTION SNAPPING

Yanwei Fu[1], *Yanwen Guo*[1,2] *

[1]National Key Lab for Novel Software Technology, Nanjing University, China
[2]Jiangyin Information Technology Research Institute of Nanjing University, China
ztwztq2006@gmail.com, ywguo@nju.edu.cn

ABSTRACT

Interactive segmentation methods have greatly simplified the task of object cutout from an image. However, segmenting a large number of images in a collection is still a tedious task. In this paper, we present a content-sensitive group segmentation method that iteratively segments the images in a collection and incrementally refines the results. With our method, a user only provides a small number of strokes to segment and refine a few sample images. For each of the rest of images, our method finds relevant sample images and applies the corresponding appearance models to guide the segmentation. To improve the segmentation results using the user strokes on a few images with unsatisfactory segmentation results, our method calculates the relevance map that measures the probability that a stroke can be appropriately applied at each pixel/region of an image, and applies it accordingly. Our experiments show that our method can effectively segment an image collection with a wide variety of image content and significantly reduce user input.

Index Terms— Interactive image segmentation, Collection segmentation, Cosegmentation

1. INTRODUCTION

The recent advance in interactive image segmentation techniques has greatly simplified the task of object segmentation (c.f. [1, 2, 3, 4, 5, 6]). These interactive methods, like lazy snapping [2], grabcut [3], and geodesic image segmentation [4], enable people to achieve high-quality object cutout with only a couple of strokes. However, it is still tedious to segment images in a large photo collection one by one.

Cosegmentation methods can automatically segment the common object from a group of images simultaneously [7, 8, 9, 10]. However, they are based on the assumption that the group of images have a common object and different background, which is often not valid. The most recent work of *iCoseg* provides a convenient way to interactively segment a group of photos [11]; however, they assume that the group of the images are related to each other. The cosegmentation methods, especially the *iCoseg* method, inspired our work.

*This work is supported in part by the National Science Foundation of China under Grants 61073098, 60723003, and 61021062, the Jiangsu Science Foundation (BK2009081) and the National Fundamental Research Program of China (2010CB327903).

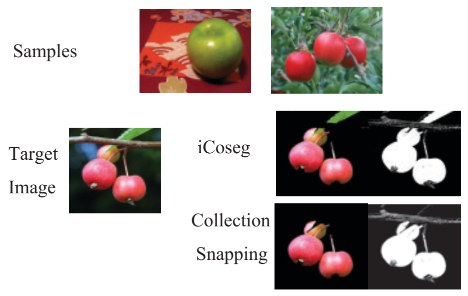

Fig. 1. Comparison between our collection snapping and iCoseg [11]. For this example, a user first segments the apples as foreground from two sample images. Then, iCoseg builds a global foreground model using the foreground content of both of the two sample images. A background model is built similarly. Since the foreground and background of the two sample images share similar color distribution, they cannot discriminate the foreground and background of the target images. As we can see, iCoseg combines the red apples and the background green leaf in the target image as foreground objects. In contrast, our collection snapping succeeds as our method discriminatively applies the color models learnt from the relevant image (right) to the segmentation of target image.

We aim to develop a collection segmentation method that iteratively segments objects of interest from a collection of images with a wide variety of content. Like the cosegmentation methods, our method is based on the observation that images in a collection or its sub-collections share commonness. The commonness can be used to automate or semi-automate the segmentation. However, it could be often misleading if we do not use them cautiously. As shown in Figure 1, we have two samples images that a user has already segmented the objects of interest, the apples. The left image has a green apple with a red background; while the right one has red apples with a green background. Then, if we build a color model for the background using the background information in the two sample images like [11, 12], and similarly a color model for the foreground, these two models have a quite similar color distribution. These two models are misleading for segmentation as shown in Figure 1. We need to discriminatively apply

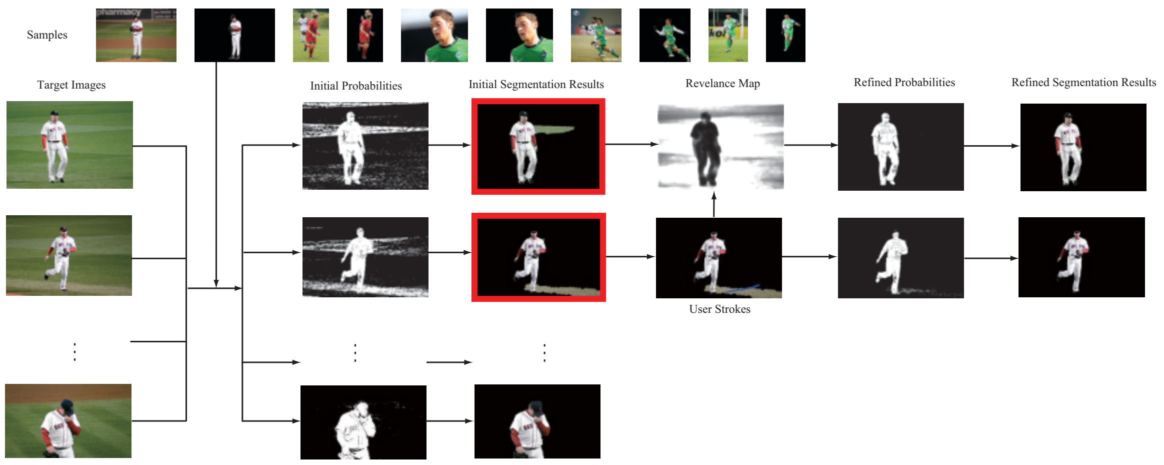

Fig. 2. Collection snapping overview. Given an image collection, we first segment a small number of samples. The rest target images are then automatically segmented guided by probabilities transferred from the color appearance models of relevant samples. When the initial segmentation results are unsatisfactory, the user can supply one representative image with more strokes for updating its foreground and background probabilities which are further propagated to the relevant images to refine segmentation results.

the appearance model from those relevant sample images to guide the segmentation, instead of all the sample images.

1.1. Our work

Based on this observation, this paper presents a content-sensitive collection segmentation method. Our method iteratively segments the images in a collection and incrementally refines the results, as illustrated in Figure 2. Specifically, a user first interactively segments a small number of sample images, typically one or two. Then our method automatically segments the rest of images by finding relevant sample images and propagating the appearance characteristics from the relevant sample images to guide the segmentation (Section 2). Appearance is often insufficient to achieve a good segmentation result. A user can supply more strokes to refine the background and foreground. Then our method estimates a relevance map for each of the rest of images, and uses the stroke information accordingly to refine the segmentation results (Section 3). Our experiments show that our method can efficiently segment an image collection with a wide variety of image content and significantly reduce user input (Section 4).

2. COLLECTION SNAPPING

Given a collection of images, our method first allows a user to use an interactive image segmentation technique, specifically lazy snapping [2] to cut out foreground objects from a few sample images, which are the representative images of the image collection. Our method then learns the appearance models for the background and foreground from these sample images and uses them to guide the segmentation of the rest of images in the collection, as shown in Figure 2.

We adopt the interactive graph cut-based segmentation framework proposed by Boykov and Jolly [1], and formulate image segmentation as the Gibbs energy minimization problem. Specifically, the Gibbs energy for the segmentation of an image $\mathbf{I}$ which can be viewed as a graph $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t)$ is,

$$E(X_t) = \sum_{i \in \mathcal{V}} E_1(x_i) + \sum_{(i,j) \in \mathcal{E}} E_2(x_i, x_j) \tag{1}$$

where $x_i \in X_t$ indicates the label of node i with $x_i = 1$ for foreground, and 0 for background. $E_1(x_i)$ is the likelihood energy term encoding the cost labeling node i with x_i, and $E_2(x_i, x_j)$ is the smoothness term penalizing the different labels of adjacent nodes.

With the interactive cutout results of sample images, we define the energy term E_1 and the smoothness term E_2 for each image $\mathbf{I}$, according to the appearance models learned from the samples.

We assume that there are K sample images $\{I_{Ek} | k = 1, ..., K\}$ that have been segmented interactively. The *iCoseg* method estimates a global background and foreground model from all the samples [11]. As shown in Figure 1, a global appearance model learned from all the sample images is problematic when the background and foreground aggregated from all the sample images have a similar color distribution. Instead, we train a separate appearance model for each sample image I_{Ek}. Specifically, for each image I_{Ek}, we train a Gaussian Mixture Model $\text{GMM}^{\mathcal{F}}_{\mathbf{I}_{Ek}}$ for the foreground and a $\text{GMM}^{\mathcal{B}}_{\mathbf{I}_{Ek}}$ for the background.

Then, we rely more on the sample images that are similar to the target images segmented to guide the segmentation than those less similar sample images. Specifically, the probability that a pixel of $\mathbf{I}$ labeled as foreground or background is determined more by similar sample images.

We calculate the probability of a pixel v_i (node i) in the target image $\mathbf{I}$ as foreground w.r.t an sample image $\mathbf{I}_{Ek}$ by using the maximum probability component in GMMs,

$$p^{\mathcal{F}}_{\mathbf{I}_{Ek}}(v_i) = \frac{p^{\mathcal{F}}_{\mathbf{I}_{Ek}{}^{\mathcal{F}}}(v_i)}{p^{\mathcal{F}}_{\mathbf{I}_{Ek}{}^{\mathcal{F}}}(v_i) + p^{\mathcal{B}}_{\mathbf{I}_{Ek}{}^{\mathcal{B}}}(v_i)} \quad (2)$$

where $p^{\mathcal{F}}_{\mathbf{I}_{Ek}{}^{\mathcal{F}}}(v_i)$ and $p^{\mathcal{B}}_{\mathbf{I}_{Ek}{}^{\mathcal{B}}}(v_i)$ are the initial probabilities of v_i being foreground and background computed with respect to the foreground and background appearance models of the sample image $\mathbf{I}_{Ek}$.

Combining the probabilities computed according to all the sample images, the overall probability is defined as,

$$p^{\mathcal{F}}(v_i) \propto \sum_k p^{\mathcal{F}}_{\mathbf{I}_{Ek}}(v_i) \cdot S(\mathbf{I}_{Ek}, \mathbf{I}) \quad (3)$$

where $S(\mathbf{I}_{Ek}, \mathbf{I})$ measures the similarity between the sample image $\mathbf{I}_{Ek}$ and the target image $\mathbf{I}$, as described in Section 2.1. Sample images similar to the target have higher influence on calculating this probability. We compute $p^{\mathcal{B}}(v_i)$ similarly.

We then define the likelihood energy term $E_1(x_i)$ that encodes the cost labeling v_i (node i) with x_i as,

$$\begin{aligned} E_1(x_i = 1) &= \frac{\log[p^{\mathcal{F}}(v_i)]}{\log[p^{\mathcal{F}}(v_i)] + \log[p^{\mathcal{B}}(v_i)]}, \\ E_1(x_i = 0) &= 1 - E_1(x_i = 1). \end{aligned} \quad (4)$$

We define the smoothness term $E_2(x_i, x_j)$ in the same way as existing graph cut-based segmentation methods,

$$E_2(x_i, x_j) = \mid x_i - x_j \mid \exp(-\beta d_{ij}) \quad (5)$$

where $d_{ij} = ||c_i - c_j||$ is the color distance between nodes i and j of the target image $\mathbf{I}$. β is a scale parameter usually set to $[2\langle d_{ij} \rangle]^{-1}$, where $\langle \cdot \rangle$ is the expectation operator.

We finally employ the graph cut segmentation algorithm [1] to obtain the segmentation results.

2.1. Similarity computation

We calculate the similarity between a sample image $\mathbf{I}_{Ek}$ and a target image $\mathbf{I}$ using two metrics: global color distribution based similarity and region-based similarity as follows:

$$S(\mathbf{I}_{Ek}, \mathbf{I}) = \lambda_g \cdot S_g(\mathbf{I}_{Ek}, \mathbf{I}) + \lambda_l \cdot S_l(\mathbf{I}^{\mathcal{F}}_{Ek}, \mathbf{I}). \quad (6)$$

where $S(\mathbf{I}_{Ek}, \mathbf{I})$ is the similarity between $\mathbf{I}_{Ek}$ and $\mathbf{I}$, $S_g(\mathbf{I}_{Ek}, \mathbf{I})$ is the global color distribution based similarity and $S_l(\mathbf{I}^{\mathcal{F}}_{Ek}, \mathbf{I})$ is the region-based similarity. λ_g and λ_l are two weights.

We model the color distribution in an image using a color histogram in HSV color space, and use the histogram quadratic distance to compute $S_g(\mathbf{I}_{Ek}, \mathbf{I})$, the global color distribution-based similarity between two images [13] .

We compute the region-based similarity metric $S_l(\mathbf{I}^{\mathcal{F}}_{Ek}, \mathbf{I})$ using the integrated region matching (IRM) [14]. In detail, we pre-segment each of the two images using the watershed algorithm [15]. Then, IRM builds correspondence between regions. IRM is consistent with human perception, and is especially suitable for our application in that it is designed to highlight a pair of images with similar objects. For more details about IRM, please refer to [14].

3. ITERATIVE REFINEMENT

Our collection snapping method can usually produce good segmentation results on most images in a collection. However, there are often some images with poor segmentation results, when the sample images are not representative enough. Besides, segmentation is known to be difficult when the contrast around object boundaries is low. Thus, our method supports a user to iteratively refine segmentation results by drawing strokes on the unsatisfied result to indicate either the foreground or background as shown in Figure 2. Once a user refines the segmentation of one image, our method can then propagate his input to refine other results.

Specifically, our method builds a GMM color model in the region around a user-input stroke. We find that directly applying this appearance model to other images is often problematic. For instance, if a background region in an image has similar color to the region around the user-input foreground stroke, the background region will have a high probability to be foreground, and thus will be mis-segmented into a foreground object. Therefore, we should restrict the influence of the newly drawn stroke to be applied to relevant regions in an image. We devise a scheme for estimating a relevance map for an image so that the stroke information can be applied appropriately in a content-sensitive way.

Assume that the local foreground and background appearance models extracted from strokes $\mathbf{S}$ are $\mathbf{S}^{\mathcal{F}}$ and $\mathbf{S}^{\mathcal{B}}$ separately. We add to the foreground probability $p^{\mathcal{F}}(v_i)$ the updated foreground probability $p^{\mathcal{F}}_u(v_i)$ for a pixel v_i in an image $\mathbf{I}_u$ with an unsatisfactory segmentation result as follows:

$$p^{\mathcal{F}}_u(v_i) \propto p^{\mathcal{F}}_{\mathbf{S}^{\mathcal{F}}}(v_i) \cdot \mathbf{M}_{\mathbf{I}_u}(v_i) \quad (7)$$

where $p^{\mathcal{F}}_{\mathbf{S}^{\mathcal{F}}}(v_i)$ is the foreground probability directly computed using the maximum probability component of foreground appearance models. $\mathbf{M}_{\mathbf{I}_u}$ is the relevance map for the unsatisfied image $\mathbf{I}_u$ evaluated with respect to the strokes $\mathbf{S}$. $\mathbf{M}_{\mathbf{I}_u}(v_i)$ measures the probability that strokes $\mathbf{S}$ can be applied appropriately to each pixel/region v_i in the unsatisfied image $\mathbf{I}_u$. The background probability $p^{\mathcal{B}}(v_i)$ is updated similarly.

We estimate the relevance map of a stroke to an image by computing the similarity of a local region around each pixel in $\mathbf{I}_u$ and a stroke $\mathbf{S}$. Specifically, for each pixel v_i in $\mathbf{I}_u$, we compute the summed squared distances (SSD) between its neighborhood and the sampled areas with the same size in the stroke $\mathbf{S}$. Denote $d(v_i)$ as the minimum distance between them. The relevance map value on v_i, i.e. $\mathbf{M}_{\mathbf{I}_u}(v_i)$ is then calculated as $\frac{1}{d(v_i)+\varepsilon}$. ε is set to $1e-6$ empirically for numerical stability. We accelerate the computation of SSD using the

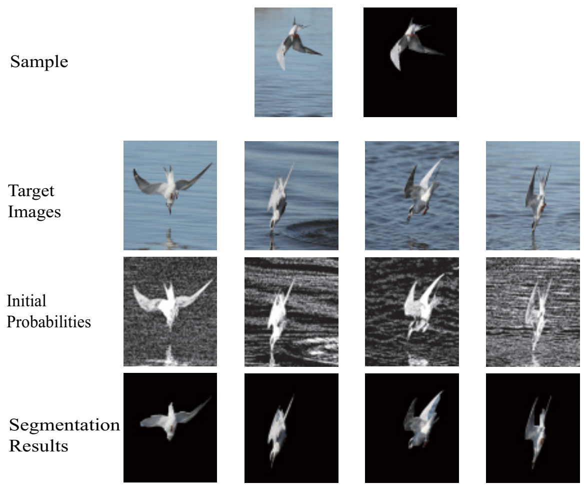

Fig. 3. Collection snapping results of seagulls.

fast Fourier Transform. We set the local neighborhood of v_i to a window of size 11×11 centered at v_i.

With the updated foreground and background probabilities, we update the Gibbs energy function for an image with an unsatisfactory segmentation result accordingly and refine the segmentation result.

4. EXPERIMENTS

We experimented with our algorithm on a variety of image collections, which come from the CMU-Cornell iCoseg Dataset [11], and miscellaneous sources, e.g. *Flickr*[1] and the dataset of HP Challenge High Impact Visual Communication in ACM Multimedia 2010. Some representative results are shown in Figures 3, 4, 5, and 6.

In Figure 3, the first row is the sample image that is interactively segmented by a user. The second, third and fourth rows are target images, the foreground probabilities predicted from the sample image, and segmentation results, respectively. Although the foreground probabilities are noisy in the background, our collection snapping method still produces satisfactory segmentation results. Figure 4 shows an image collection with butterflies and sculptures as foreground objects. The samples, target images, initial probabilities, initial segmentation results and refined segmentation results are shown in each row separately. After initial segmentation, green leafs and some stones in the third and fourth target image are mis-segmented as foreground. Good results are produced by adding more strokes. The first row in Figure 5 also lists the sample images segmented by a user, while the second row and third row are the target images and segmentation results separately. Although the background patterns are cluttered with grasses or leaves in most target images, the collection snapping results are of good quality. Note that, we produce the final segmentation results for the collections shown in Figures 3 and 5 with only initial snapping, and no further user strokes are needed.

[1]http://www.flickr.com

	Seagull	Landscape	Flower	Sports
Ours	0.4	1.0	1.8	2.0
Li's [2]	2.0	3.0	5.3	6.75

Table 1. Statistical data of usability study. The average numbers of strokes applied to per image in each photo collection used by our collection snapping and lazy snapping are shown.

The collection in Figure 6 consists of sports photos. The photos in this dataset have very complex and diverse background and foreground appearance. For example, the color background in images with baseball players is green, while the color of foreground in images of man football players is green. The first row in Figure 6 lists the sample images that are interactively segmented by a user, and the following rows list the whole process of segmenting the rest images by automatically propagating the appearance characteristics from the relevant sample images. The next every four rows are target images, initial foreground probabilities, initial segmentation results, and the refined segmentation results individually. Although the initial segmentation results for many images are good, there are still some results that need to be improved. The ambiguous and low contrast object boundaries also cause unsatisfactory segmentation. For example, the black hats in the third and fourth images of baseball players, and the shoes in the eighth image of a woman player are missing in the initial segmentation results. We add user strokes to refine these results.

Our method propagates the user strokes based on relevance maps. The relevance maps are used to guarantee that the stroke in an image will be appropriately applied to another one. For example, the initial segmentation result of the first and second images of player photos in Figure 6 both have some green land mis-segmented into foreground. The user strokes in the second image have positive influence on the first image via the relevance map. This greatly reduces the overall strokes needed in the refinement process.

4.1. Usability Study

We compare the performance of our collection snapping method with our implementation of lazy snapping [2], a representative single-image interactive segmentation method.

In our study, we did not take the length of strokes as a measure, because it does not cost much more effort when a user draws a longer stroke than a short one. We count the average number of the strokes user labeled on an image. We asked three subjects who are Photoshop experts, thus familiar with the stroke interface, and are familiar with the Lazy snapping cutout tool to segment the same collection. We report the average number of strokes for each image using our method

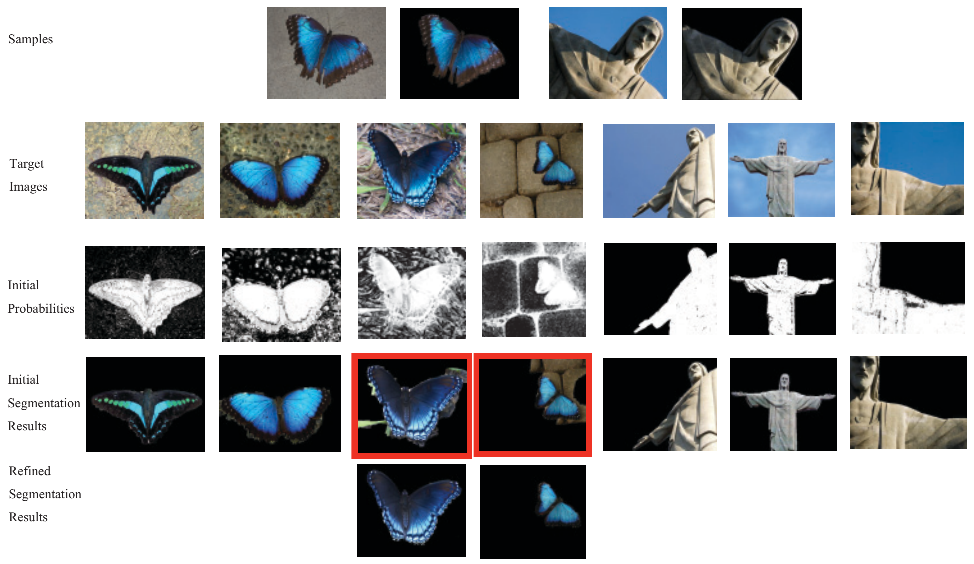

Fig. 4. Collection snapping results of landscape.

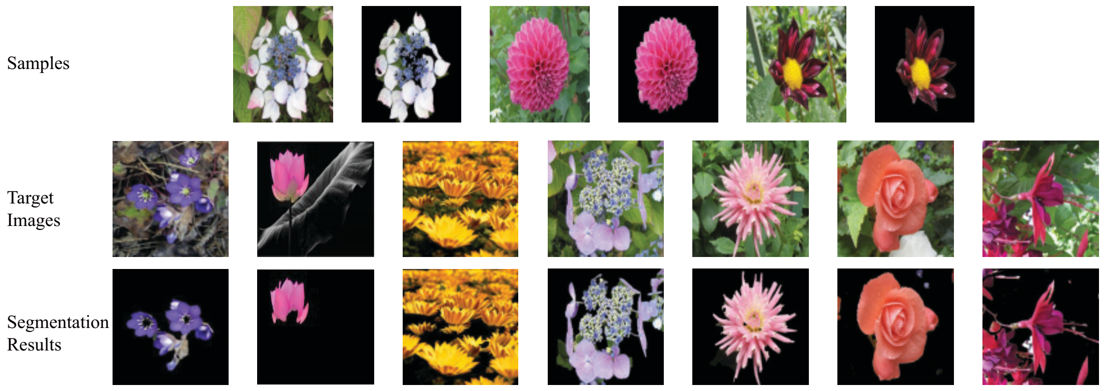

Fig. 5. Collection snapping results of flowers.

and the lazy snapping method in Table 1. We can see that our method can significantly reduce user interaction.

5. CONCLUSIONS

We have presented a collection snapping method that iteratively segments an image collection with a wide variety of image content. Our method starts from a small number of sample images that are segmented by a user. Then, we use the segmentation results from appropriate sample images to guide the segmentation of the rest of the images in the collection. In order to further improve the segmentation results using user-provided strokes, our method calculates the relevance map that measures the probability that a stroke can be appropriately applied at each pixel/region, and applies the stroke accordingly. Our experiments show that our method can efficiently segment an image collection with a variety of image content and significantly reduce user efforts.

The initial segmentation results are sensitive to similarities between the target images and samples. We compute the image similarities using a region matching method which only employs low level information. We intend to incorporate more sophisticated high level features based method into our framework.

6. ACKNOWLEDGEMENT

We would like to thank the anonymous reviewers for their helpful comments and suggestions which help us improve the quality of the paper.

7. REFERENCES

[1] Y. Y. Boykov and M.-P. Jolly, "Interactive graph cuts for optimal boundary & region segmentation of objects in n-d images,"

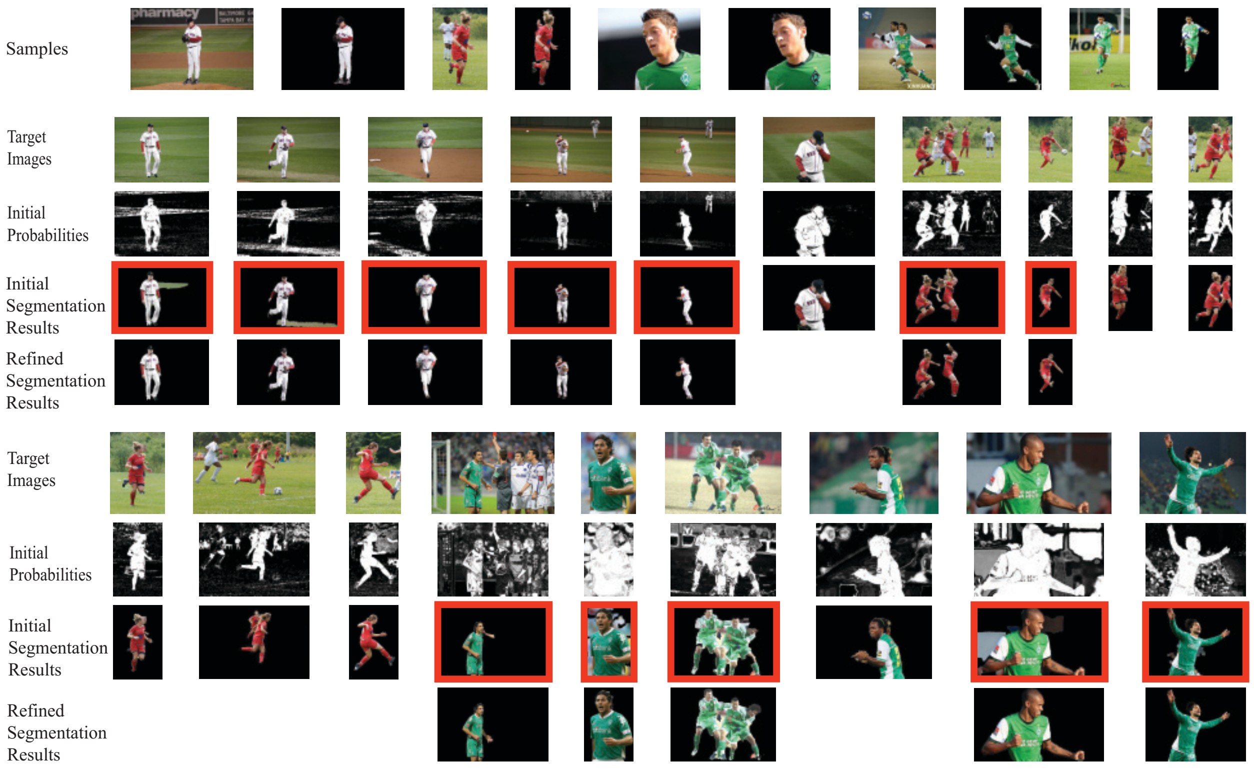
Samples
Target Images
Initial Probabilities
Initial Segmentation Results
Refined Segmentation Results
Target Images
Initial Probabilities
Initial Segmentation Results
Refined Segmentation Results

Fig. 6. Collection snapping for a sports photo collection. After initial collection snapping, the results of some images are of good quality, while those images with poor initial results are refined iteratively (denoted by the red rectangles) until all images are properly segmented.

in *Proceedings of IEEE International Conference on Computer Vision*, 2001, pp. 105–112.

[2] Y. Li, J. Sun, C.-K. Tang, and H.-Y. Shum, “Lazy snapping,” in *Proceedings of ACM SIGGRAPH*, 2004, pp. 303–308.

[3] C. Rother, V. Kolmogorov, and A. Blake, “Grabcut: Interactive foreground extraction using iterated graph cuts,” *ACM Transactions on Graphics*, vol. 23, pp. 309–314, 2004.

[4] X. Bai and G. Sapiro, “A geodesic framework for fast interactive image and video segmentation and matting,” in *Proceedings of IEEE International Conference on Computer Vision*, 2007, pp. 1–8.

[5] D. Freedman and T. Zhang, “Interactive graph cut based segmentation with shape priors,” in *Proceedings of ACM SIGGRAPH*, 2005, pp. 20–25.

[6] J. Liu, J. Sun, and H.-Y. Shum, “Paint selection,” *ACM Transactions on Graphics*, vol. 28, pp. 69:1–69:7, July 2009.

[7] C. Rother, V. Kolmogorov, T. Minka, and A. Blake, “Cosegmentation of image pairs by histogram matching - incorporating a global constraint into mrfs,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2006, pp. 993–1000.

[8] L. Mukherjee, V. Singh, and C. R. Dyer, “Half-integrality based algorithms for cosegmentation of images,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 2028–2035.

[9] J. Cech, J. Matas, and M. Perdoch, “Efficient sequential correspondence selection by cosegmentation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1568–1581, 2010.

[10] A. Joulin, F. Bach, and J. Ponce, “Discriminative clustering for image co-segmentation,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 1943–1950.

[11] D. Batra, A. Kowdle, D. Parikh, J. Luo, and T. Chen, “i-Coseg: Interactive co-segmentation with intelligent scribble guidance,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 3169–3176.

[12] D. Batra, A. Kowdle, D. Parikh, J. Luo, and T. Chen, “Interactively co-segmentating topically related images with intelligent scribble guidance,” *International Journal of Computer Vision (IJCV), To appear*, 2011.

[13] J. R. Smith and S.-F. Chang, “Visualseek: a fully automated content-based image query system,” in *Proceedings of ACM Multimedia*, 1996, pp. 87–98.

[14] J. Z. Wang, J. Li, and G. Wiederhold, “Simplicity: Semantics-sensitive integrated matching for picture libraries,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, pp. 947–963, 2001.

[15] H. T. Nguyen, M. Worring, and R. Van Den Boomgaard, “Watersnakes: Energy-driven watershed segmentation,” *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 25, pp. 330–342, 2003.