Principal-channels for One-sided Object Cutout

Lior Gavish, Lior Shapira, Lior Wolf and Daniel Cohen-Or
School of Computer-Science
Tel Aviv University, Israel

Abstract

We introduce principal-channels for cutting out objects from an image by one-sided scribbles. We demonstrate that few scribbles, all from within the object of interest, are sufficient to mark it out. One-sided scribbles provide significantly less information than two-sided ones. Thus, it is required to maximize the use of image-information. Our approach is to first analyze the image with a large filter bank and generate a high-dimensional feature space. We then extract a set of principal-channels that discern one object from another. We show that by applying an iterative graph-cut optimization over the principal-channels, we can cut out the object of interest.

1. Introduction

One inherent difficulty in automatic image segmentation is that different user interpretations of the image lead to different results. It therefore seems that some level of user guidance is unavoidable. A natural question to ask is what is the minimal level of such user provided information.

This question is not merely a theoretical question. A reduction in required user intervention reduces the users' efforts and simplifies their experience. In addition, it can lead to automatic image segmentation of pre-specified object classes using today's often crude and unreliable object-recognition methods. These methods may mark reliably some pixels inside an object, but requiring the marking of outside pixels, makes the problem almost twice as hard.

We motivate our work by what we dub "the axiom of choice". Namely, that each object in the image can be selected without ambiguity from a few scribbles taken only from inside the object. This one-sided approach provides much less information than is provided to other segmentation techniques. For example, Lazy Snapping [1] requires scribbles both inside and outside the object. GrabCut [2], another popular segmentation method, requires the use of a bounding box, which segments the entire image approximately to foreground and background. Naturally, a reduced amount of user supervision makes the segmentation problem significantly harder.

A new method is suggested in order to deal with this decreased amount of supervision, while not sacrificing the segmentation quality. The method is tapping into sources of information that are not fully used in today's competing methods. One such source is texture information via the filter bank approach and a simple patch-based color representation.

In the texture representation above, a large number of filters and color descriptors are used. For a specific texture-edge, only a few of these are relevant. The rest may hinder the total edge response. An optimal combination of texture and color is also a non-trivial problem. We deal with these issues by employing Principal Component Analysis to infer a reliable visual similarity metric among image pixels. The result is a small set of principal-channels, which highlight different aspects of local appearance (Figure 1). We use the principal-channels to define an energy function which should correspond to a proper segmentation. An efficient graph-cuts approach [3] is then applied to optimize the energy function.

Lastly, the whole process is iterated in order to employ the information available at the end of one segmentation iteration to obtain a better-yet segmentation. This iterative scheme, which is similar to the one in GrabCut, improves the results significantly. Taken together, the newly combined foreground extraction method provides a solid solution to minimal intervention image segmentation.

2. Previous Work

Graph-cuts segmentation methods. The graph-cuts technique in the context of image segmentation is first introduced in [4], where it is used to integrate region information with boundary cues. The technique has since proven to be useful and efficient in optimizing a large class of cost functions.

Our work is especially related to Lazy Snapping [1] and GrabCut [2], both providing the user with an interactive UI for foreground extraction. In both systems, the user's input is used as a cue for modeling the foreground and background colors. A Graph-cuts optimization then finds an optimal labeling with respect to the computed models. While Lazy Snapping allows the user to draw scribbles on the image, marking the locations of the foreground and the background, GrabCut users draw a rectangle around the foreground object. In the GrabCut system, iterative graph-cuts are used to derive growingly accurate foreground models, compensating for the lack of specificity in the user's marking.

Figure 1. A small set of principal channels (p-channels) highlight different aspects of local appearance.

Texture analysis.. Traditional methods for texture analysis are often grouped into three major categories: statistical, structural and spectral. In the statistical approach, texture statistics (e.g., moments of the gray-value histogram, or co-occurrence matrices) serve as texture descriptors. In structural approaches, the structure is analyzed by constructing a set of rules that generates the texture. Spectral approaches analyze the frequency domain.

In contrast to the wealth of approaches suggested in the past, the last decade has been dominated by the filter bank approach [5], where the image is convolved with a large collection of filters tuned to various orientations and scales. Of the many studies that employ banks of filters, the most common set of filters seems to be the Gabor filters [6]–[8], which we use here as well.

3. The proposed cutout algorithm

Figure 2 shows an overview of our system. Below, we portray its building blocks. We motivate each of the components separately, and describe their integration.

Texture Feature Extraction . Our system utilizes a set of color and texture features to capture local image appearance. For each pixel, we calculate three types of raw features. (1) the Lab color values (a 3-vector); (2) A vector of length 30 depicting the color histogram of a local patch; and (3) A set of 24 Gabor filter outputs. We then process these inputs and use (1) for the smoothness term and the combination of (2) and (3) as the texture descriptor used in the data term.

More specifically, the color distribution of a local 5×5 neighborhood is estimated channel-by-channel by employing histograms. Each histogram contains 10 bins, and we have three such histograms corresponding to the R, G and B channels. To obtain local spatial structure descriptors, we use a battery of Gabor filters in 4 different scales and in 6 orientations. For each image pixel the vector of length 24 of filter outputs forms the frequency encoding part of the texture descriptor.

The Principal-channels. We apply PCA to the concatenated Gabor and histogram vectors (54 dimensions) to extract the three most prominent texture-based channels. The transformation between the normalized texture space and the new representation is simply given as a 3×52 matrix with orthogonal rows.

Interestingly, in contrast to our original hypothesis supervised learning does not perform better for this task than PCA, even if many reliable foreground/background labels are given as training. We tried several classical and less classical learning techniques such as LDA, Fisher criterion score, Pearson coefficients, and the Kolmogorov-Smirnov test (the last three are feature selection techniques), SVM, and NCA [9]. We speculate that the reason of the lack of success with such techniques is that the most relevant discriminative information is at the texture boundaries. We cannot expect, however, to have that information available, until the segmentation problem is solved.

The Lab 3-vectors are also transformed via PCA. Here, we do not reduce the dimensionality of the data. However, the PCA transform allows us to observe visually the dominant edge pattern in the image. We coin the combination of the three texture PCA channels and the three color PCA channels "principal-channels", or *p*-channels for short. We present such channels as images in Figure 3. As can be seen, prominent image edges often manifest themselves clearly in at least one of those channels.

Graph-Cut iterations. To solve the segmentation problem, we adopt an energy minimization approach. We let each pixel i in the image have a label $x_i \in \{F, B\}$, indicating whether it belongs to the foreground or the background. We define the following energy term

$$E(x_1, \ldots, x_N) \quad = \quad \sum_i E_1(i, x_i) + \alpha \cdot \sum_{i,j \in Nei(i)} E_2(i, j, x_i, x_j),$$

Figure 2. Overview of the segmentation process. We extract a high-dimensional feature space (b) from the input image (a). Features are used to derive the *p*-channels (c). We then iteratively estimate the probability of a pixel to belong to the foreground (d) and segment the image (e).

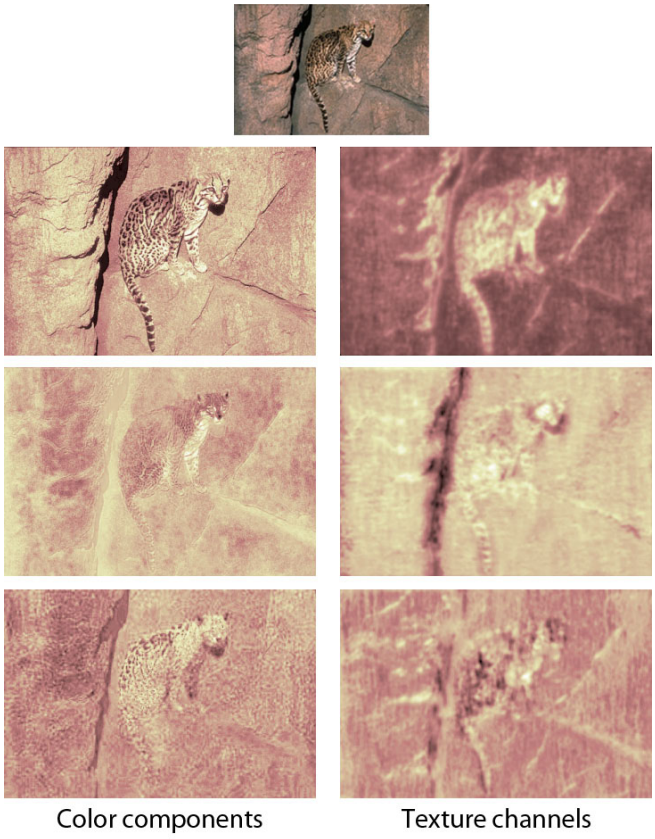

Figure 3. The *p*-channels that were extracted from the input image. Color components are shown on the left column, texture on the right. Notice how the topmost texture channel captures the cat.

which accounts for foreground/background modeling via the E_1 term, and for the local smoothness and coherence via the E_2 term. E_1 describes the likelihood of each pixel in the image to belong to one of the classes: the foreground or the background. Specifically,

$$E_1(i,x_i) \quad = \quad -\log\left(Px_i(a(i))\right),$$

where $P_{x_i}(a(i))$ is the probability that a pixel with a *p*-channel descriptor $a(i)$ belongs to class x_i. We choose to model the distribution of each class using a Gaussian Mixture Model (GMM) over the 6D *p*-channels, where each mixture contains five Gaussians with general covariance matrices.

The parameters of the foreground model are estimated using the points on the user specified scribbles as the training set. This is done using the EM algorithm. We do not, however, have training points for the background data. We found that initializing the background GMM using all of the points in the image as training works well. First, in many situations the background dominates the foreground with respect to the number of pixels, and second, it is the difference in the likelihoods between the classes that determines much of the segmentation. That is, it is sufficient at first that for many foreground pixels, the foreground likelihood would be larger than the background likelihood.

Our smoothness term E_2 relies solely on the three color components of the *p*-channels. We do not use the texture information here, since some of our texture features are derived using filters with significant supports, and therefore show rather crude edges. The combination of color-based edge term with color/texture-based region term allows us to simultaneously capture the appearance of the foreground object and maintain accuracy near the edges.

Specifically, we use the term

$$E_2(i,j,x_i,x_j) \quad = \quad \begin{cases} \exp\left(-\|a^c(i)-a^c(j)\|^2/\beta\right) & x_i \neq x_j \\ 0 & x_i = x_j \end{cases}$$

where $a^c(i)$ is the color components vector of pixel i's *p*-channels response, and $\beta = 10\frac{1}{4N}\sum\|a^c(i)-a^c(j)\|^2$. This term penalizes foreground/background boundaries where there is no clear edge in the image, thus encouraging coherent labeling of spatially-related uniformly-colored pixels. The energy term E is optimized using an iterative α-expansions graph-cuts technique [3]. The optimization results in a labeling $x_1,\ldots,x_N$, assigning each pixel to the foreground or the background.

Repeated iterations. Our system faces a situation which is not unlike the one faced by GrabCut: Statistical models that are used for segmentation are being estimated from limited user input.

The solution used in GrabCut is appropriate here as well. We use an iterative scheme where the background/foreground solution of one iteration is fed to the

next. Given a segmentation from a preceding iteration, we re-estimate the parameters of the background and foreground models by fitting the foreground and background labels of that iteration. Provided the new, more accurate GMM models, we re-estimate the probabilities $P_F(i), P_B(i)$, which govern the region term E_1. We then apply the graph-cuts method to obtain a new, better segmentation. The process is demonstrated in Figure 4.

We repeat this process until changes in $E(x_1, \ldots, x_N)$ are negligible. Our iterations are guaranteed to converge to a local minima of $E(x_1, \ldots, x_N)$, since every step we make does not increase the energy. This can be shown as follows. While graph-cuts segmentation optimizes $E(x_1, \ldots, x_N)$ with respect to the labels $x_1, \ldots, x_N$, the re-estimation of the GMMs optimizes $E(x_1, \ldots, x_N)$ with respect to the model parameters. Since both steps cannot increase the energy, the process is bound to converge.

4. Results

We have compared the quality of the foreground extraction provided by our method to the one provided by Lazy Snapping and GrabCut. Lazy Snapping requires background scribbles, however, even with this extra information it does not perform as good as our algorithm. The main reason is its limited texture analysis capabilities. See Figure 5 for one such comparison. GrabCut, like our algorithm is motivated by the need of having minimal user intervention. However, a rectangle marking the outside of an object if often insufficient for accurate segmentation, as demonstrated in Figure 6. The authors of GrabCut propose to correct the result by letting the user add scribbles. Effective use of GrabCut, therefore, might require more user interaction than our method.

In another set of experiments, we have applied our algorithm to the output of automatic object-detection systems.

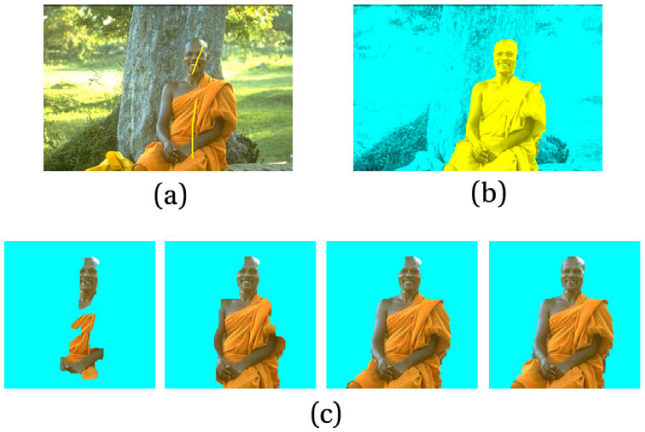

Figure 4. We show how the input image (a) is iteratively segmented (c) to arrive at the result (b).

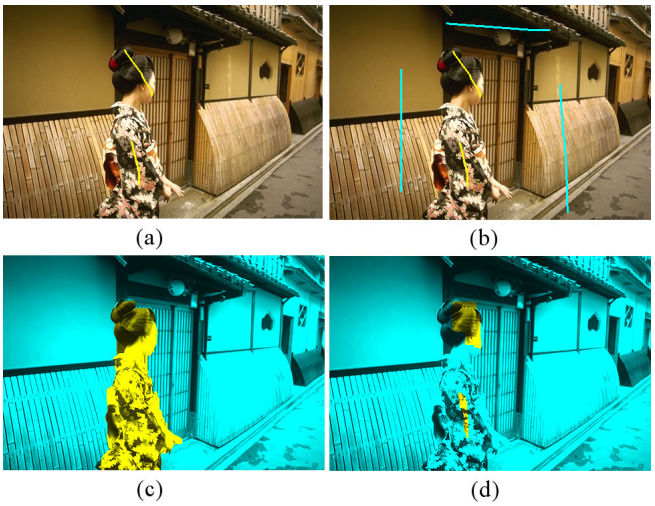

Figure 5. We compare our method with Lazy Snapping. Though we use only the foreground scribbles (a) we derive a better result (c) as compared to Lazy Snapping (d) with two-sided scribbles (b).

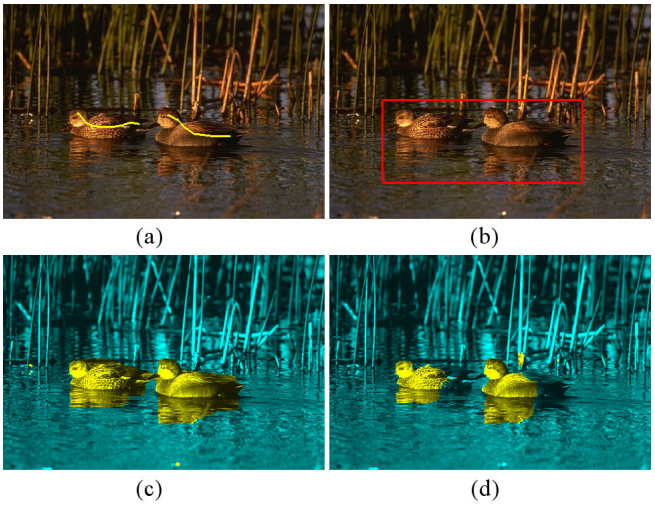

Figure 6. We compare our method with GrabCut. Using our scribbles (a) we derive a better result (c) as compared to GrabCut (d) using a surrounding box (b). Though GrabCut allows the user to manually correct the segmentation, this would considerably add to the amount of required user input.

We ran the pedestrian detector of Dalal and Triggs [10] on sample images. This detector excels in its task, but more often than not marks rectangular detections that are either too large or too small. For that reason, running GrabCut on the detector's output fails in extracting the boundaries of the pedestrian.

We ran our algorithm where as a scribble, we took a stripe in the middle of each detection result (a rectangle) which has a height of 1/5 of the rectangle and a width of 1/20. As can be seen in Figure 8 the resulting segmentation can handle detections that are out of scale in both directions.

Our algorithm can sometimes display behavior which is unexpected by most users. For example, it can mark regions that are not connected to the marked foreground object (see Figure 7). This can be fixed by simple means, if the application so requires. Additional segmentation results can be seen in Figure 9.

5. Conclusions

We have explored minimal forms of interaction which are suitable for many graphics editing tasks. We have demonstrated that it is possible to reduce the amount of user guidance and in the same time improve the results of a foreground extraction task. This result opens new avenues for other graphical editing tasks in which scribbles are used, since the number of the employed scribbles can be reduced as well as their variety. Moreover, robotic "users" with limited intelligence, so to speak, such as the current automatic object recognition systems, can also benefit from minimal user guidance systems.

We attribute the success of our system to two factors. The first factor is the use of carefully chosen texture features that capture local structures as well as their color. The other contributing factor is a carefully engineered process in which these features are combined into one overall system. By making the design choices presented here we deliver a foreground extraction system which requires minimal user indication and works well for a variety of images.

References

[1] Y. Li, J. Sun, C.-K. Tang, and H.-Y. Shum, "Lazy snapping," in *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers*. New York, NY, USA: ACM Press, 2004, pp. 303–308.

[2] 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, 2004.

Figure 7. The hands of people at the back are marked as foreground, since they have visual similarity to the swimmer.

Figure 8. (a) The original images with pedestrian detections marked in red. (b) Pedestrians extracted using our method. (c) Pedestrians extracted using GrabCut.

[3] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," in *ICCV (1)*, 1999, pp. 377–384. [Online]. Available: citeseer.ist.psu.edu/boykov99fast.html

[4] Y. Boykov and M.-P. Jolly, "Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images." in *ICCV*, 2001, pp. 105–112.

[5] I. Fogel and D. Sagi, "Gabor filters as texture discriminator," *Biological Cybernetics*, vol. 61, pp. 103–113, 1989.

[6] J. Malik and P. Perona., "Preattentive texture discrimination with early vision mechanisms," *J. Optical Soc. Am.*, vol. 7, no. 2, pp. 923–932, 1990.

[7] T. Hofmann, J. Puzicha, and J. Buhmann, "Unsupervised segmentation of textured images by pairwise data clustering." in *The IEEE International Conference on Image Processing*, 1996.

[8] C. Fowlkes, D. Martin, and J.Malik, "Learning affinity functions for image segmentation: Combining patch-based and gradient-based approaches." in *IEEE Conference on Computer Vision and Pattern Recognition*, 2003.

[9] J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov, "Neighbourhood components analysis," in *Advances in Neural Information Processing Systems 17*, 2005, pp. 513–520.

[10] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *International Conference on Computer Vision & Pattern Recognition*, C. Schmid, S. Soatto, and C. Tomasi, Eds., vol. 2, INRIA Rhône-Alpes, ZIRST-655, av. de l'Europe, Montbonnot-38334, June 2005, pp. 886–893. [Online]. Available: http://lear.inrialpes.fr/pubs/2005/DT05

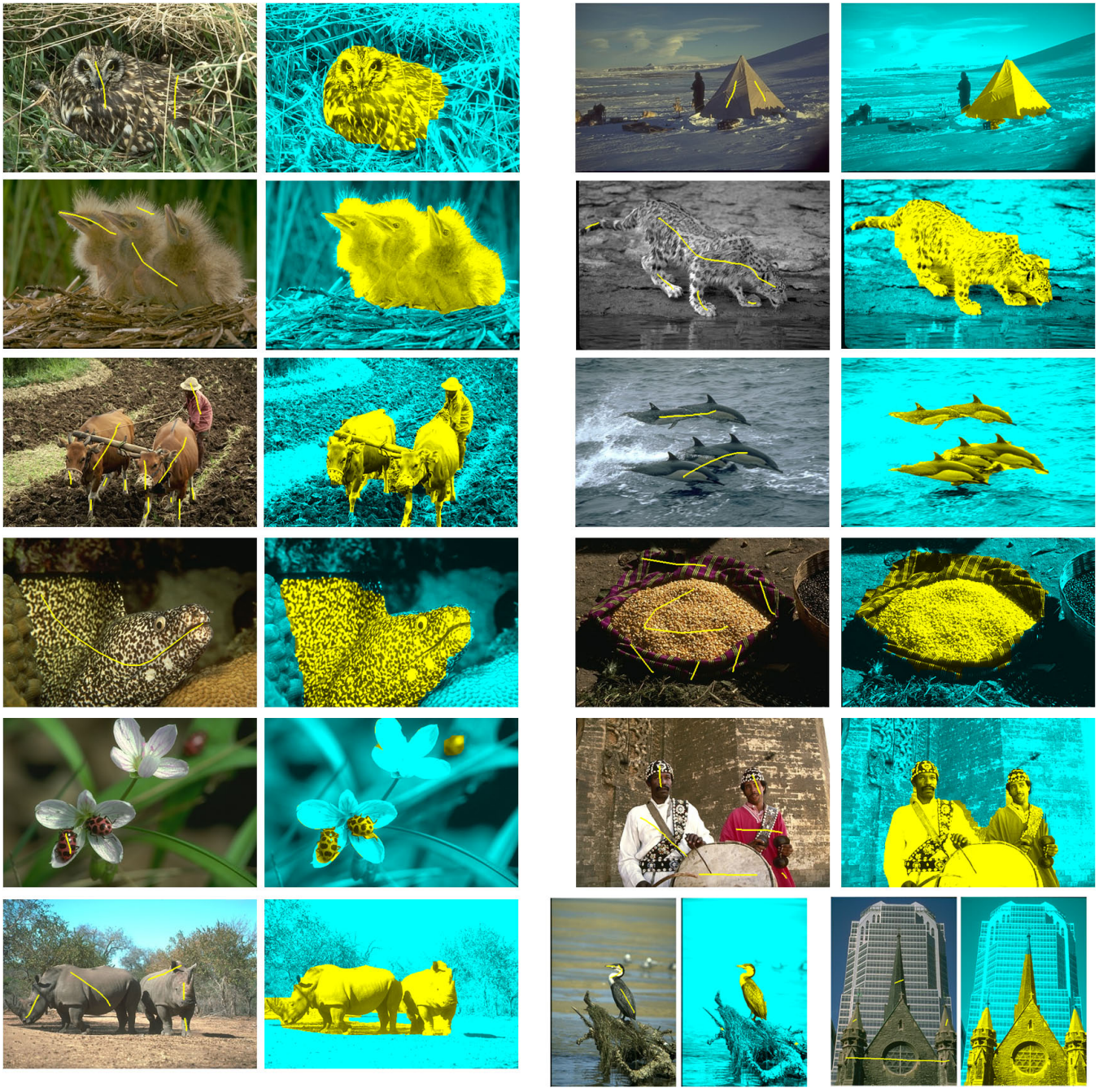

Figure 9. Additional segmentation results. In each pair the original image is shown on the left with one-sided scribbles, and the segmented result is shown to its right.