UNSUPERVISED OBJECT EXTRACTION BY CONTOUR DELINEATION AND TEXTURE-BASED DISCRIMINATION

Litian Sun and Tadashi Shibata

Department of Electrical Engineering and Information Systems, The University of Tokyo

ABSTRACT

An unsupervised object extraction system capable of separating animal images from background in natural scenes has been developed. Edges are detected from original images at multiple reduced resolutions and used to identify object contours. In order to determine the inside and the outside of an object, texture information is compactly represented using oriented edges and analyzed. The contour information and the texture information are integrated so that they complement each other. As a result, the object region has been successfully extracted as is from the scene with a fairly-well defined boundary line. Simulation experiments were carried out on several natural images and promising performance has been demonstrated.

Index Terms— Object extraction, Contour, Texture, Oriented edge, Natural images

1. INTRODUCTION

Development of intelligent image understanding systems is essential in a variety of applications such as robot vision, semantic indexing and description, image representation and compression, and so forth. Object extraction, in particular, plays an underlying role in implementing intermediate and high-level image processing tasks like object recognition and retrieval.

A lot of works have been done on supervised object extraction that requires various user inputs [1, 2]. It is still a challenging task to automatically distinguish an object from the background, although it is an easy task for human beings. The unsupervised object extraction algorithm in [3] mainly relies on multi-scale Canny edge detection [4] and can neither deal with ambiguous object boundary nor handle complex background with texture. Contour delineation is proposed to give better object boundary mimicking the Gestalt phenomenon of human eyes [5]. It delineates the most salient features in a scene as an object, but often undesirable results are obtained for images with salient background texture, because it produces false contours from the background. Gabor filters are utilized to analyze the texture difference between objects and background [6] of rice and bacteria images. Some other works apply center-surround divergence feature statistics [7] to detect the salient region or develop scale-based connected coherence tree[8] to extract the object region from natural scenes. Still, these methods neither produce exact object boundary, nor well handle complex background.

In this work, an unsupervised object extraction system capable of separating animal images from various backgrounds in natural scenes has been developed. Oriented edges are detected from original images at multiple reduced resolutions, which are used to delineate object contours. In order to discriminate an object from the background, we have used the texture information. Local images densely sampled from an input image are represented also using oriented edges based on the projected principal edge distribution (PPED) algorithm [9], then analyzed by K-means clustering. Since high-level information is indispensable to carry out object extraction, we simply assumed that the object is located centrally in the scene as in [10]. The delineated contours thus successfully identify boundary lines of the object, but several extra lines are delineated as well from the salient features existing in the background. On the other hand, the texture-based discrimination well separates the object area from the background area, while their boundaries are too fuzzy to determine the boundary lines correctly. In this work, the two cues, the contour and texture, are integrated complementarily to yield correct boundary lines and extract an object from the scene. Simulation experiments are carried out on several natural images and promising performance has been demonstrated.

2. OBJECT EXTRACTION ALGORITHM

In this work, it is assumed that only one object is depicted in the scene and that the object is almost entirely included. This assumption allows us to regard the peripheral region of the image as the background.

2.1. Contour delineation

We first locate salient features by repeatedly reducing the image resolution to filter out less salient edge fragments, then add in boundary details by restoring the resolution.

Thus, our algorithm extracts the most salient features from the object image to form contours.

A pyramid of reduced resolution images is first obtained by repeatedly shrinking the image to a quarter of its original size. To preserve the intensity contrast while suppressing textures simultaneously, a bilateral filter [11] is integrated into the "shrinking" process. Bilateral filter could enforce both geometric and photometric locality by combining domain and range filtering. Fig. 1(a) illustrates the implementation of "shrinking to a quarter". The pink 5×5 matrix is the domain of the bilateral filter. Blue pixels are the central points for bilateral filter calculation, which is equivalent to sampling one pixel out of each 2×2 region. Because we calculate the filter in a region of 5×5, we set the range sigma of the bilateral filter as (5+1)/6=1, and the intensity domain sigma of the bilateral filter as (256+1)/6=42.8. An image pyramid example is shown in Fig. 1(b).

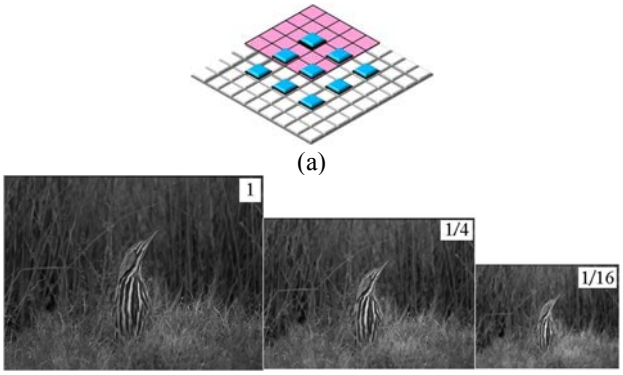

Fig. 1. (a) Scheme of "shrinking". The sampled pixels (blue) are convolved with the bilateral filter in its 5×5 neighborhood pixels (pink). (b) Image pyramid of original, 1/4 and 1/16 resolutions. To improve the visibility, the pictures shown above are not correctly scaled.

Then edge pyramid is calculated from the image pyramid employing the globally determined threshold [12]. Gradient is calculated at every pixel location as the summation of the magnitude of gradient values along horizontal and vertical directions. The gradient values are sorted, and a certain number of pixels of larger gradient values than others are marked as edges. The number of pixels to be left as edges is specified by their percentage to the total number of pixels in the entire image (12.5% is used throughout this work). In the edge pyramid, edge maps (EMs) from smaller images contain less background or texture information. Besides, fragmental lines in larger EMs tend to merge into continuous lines in smaller EMs, which reflect the connectivity between adjacent fragments. (See the top string in Fig. 2.)

Object contour restoration is carried out by merging two adjacent scaled EMs in the edge pyramid (Fig. 2). The EM of 1/16 and the EM of 1/4 are utilized to produce the restored EM of 1/4, the detailed procedure of which is explained in Fig. 3. The same procedure is repeated to produce the restored EM of full resolution (×1) from the restored EM of 1/4 and the full scale EM (×1), which yields the most salient features of the image.

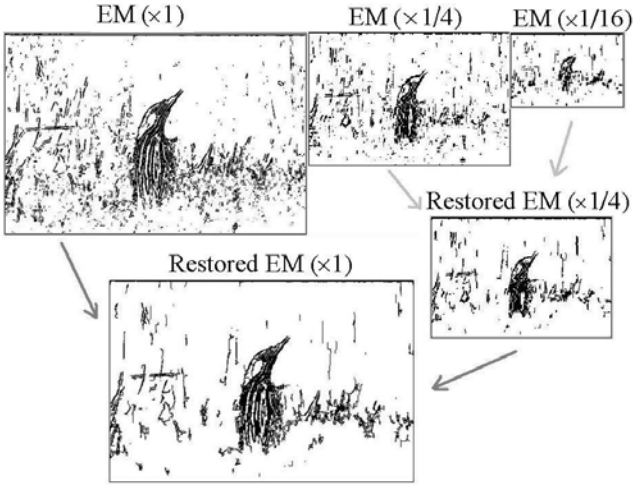

Fig. 2. The edge pyramid and restoration process.

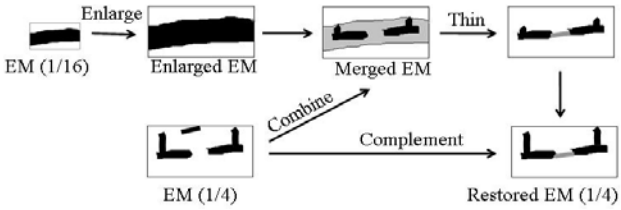

Fig. 3. Contour restoration process from EM of 1/16 and EM of 1/4.

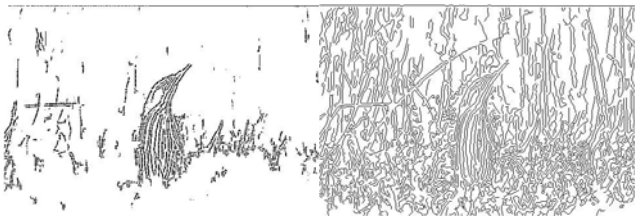

Fig. 4. Contour line candidates obtained by our algorithm (left), and those obtained by Canny detector [4] (right).

As shown in Fig. 3, the EM of 1/16 is enlarged to the size of the1/4 EM, and they are merged together. The edge flags in the 1/4 EM that appear in the edge flag region of the enlarged EM are preserved (indicated in black in the merged EM), while isolated fragmental lines are eliminated. Then the edge flag region in the merged EM (indicated in gray) is thinned by erosion to restore the connectivity among the fragmental contour lines.

The candidates for contour lines obtained in this algorithm are much less compared to the edge lines calculated

by a Canny detector [4] as shown in Fig. 4. The selection to determine the final contour line is carried out by merging the candidates with the texture information, which will be explained in § 2.2.

2.2. Texture analysis

The texture analysis is carried out only for full scale images, and reduced resolution images are not used in this process. The texture of a local image (64×64 pixels) is also represented with oriented edges. However, we employ locally determined threshold rather than the globally determined threshold used in the contour delineation. The local threshold is determined by the rank-ordering threshold of a small region of 5×5 pixels.

An edge flag is detected at each pixel if its gradient value is larger the 10th largest gradient value in its 5×5 neighborhood. This threshold process is expressed as follows:

$$Edge\,flag(p) = \begin{cases} 1 & \text{if } p > q_{10}, \{q_1 >, \ldots, > q_{25}, q \in S\}, \\ 0 & \text{else} \end{cases} \quad (1)$$

where S is a 5×5 matrix centered with pixel p.

The spatial structure of texture is compactly represented by a PPED (projected principal edge distribution) vector [9] (Fig. 5). An edge map is first split into four edge flag maps according to the principal orientations at each pixel. The texture information in this 64×64 region is then summarized into a 64 dimension vector by projection and concatenation (the bottom right in Fig. 5). We name the PPED vectors thus obtained as "texture vectors". Differences among the texture vectors are calculated using Manhattan distance. Sampling of texture vectors is carried out for every 16 pixels in both horizontal and vertical directions in the entire image.

Gradient values theoretically vary from 0 to 1275, and typical values fall in the range from 20 to 1000. Edge flags with gradient values less than 20 are considered as unnoticeable. If the number of unnoticeable edge flags exceed 80% of the total number of edge flags in the 64×64-pixel region of a sample point, this sample point is taken as non-texture. Such sample points are grouped as a non-texture cluster and excluded from the following segmentation analysis.

The texture vectors generated from an input scene are analyzed with K-means clustering. Sample points of such texture vectors are indicated by dots in the texture map (Fig. 6). We found that increasing the cluster number either leads to new clusters around the boundary of different textures (like the light blue cluster in the left of the figure) or subdivision of ambiguous region (like the pink cluster in the left of the figure), which is not preferable. Although not shown, we could expect that cluster number larger than 5 would severely impair the semantics of the segmentation result. For simplicity and generality, we set the default cluster number as 5.

Note that the K-means clustering carried out in the feature space of the texture vector does not include any location (coordinate) information about where the vectors are sampled. But as shown in Fig. 6, clusters are very well segregated and separated in the spatial coordinate space. This fact indicates that the texture vector representation based on the PPED algorithm very well describes the texture feature of local images.

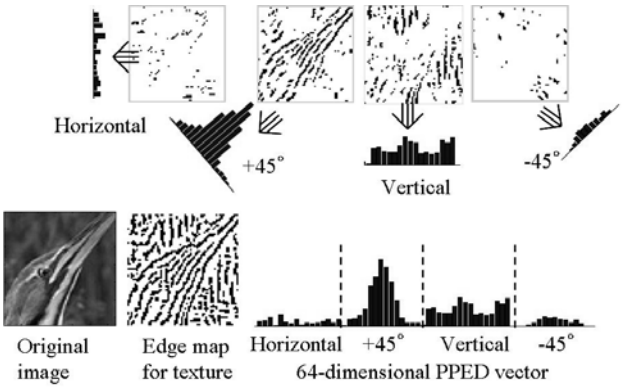

Fig. 5. A 64-dimension PPED vector is generated from four direction edge maps as distribution histograms.

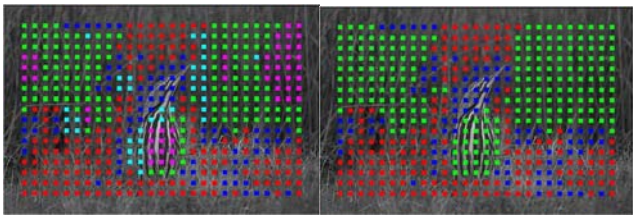

Fig. 6. Texture map showing the sample points after K-means clustering of PPED vectors. The cluster number is set as K=5 (left) and K=3 (right).

2.3. Cue integration

In this part, it is explained how the contour and texture information are integrated to give out the object region.

Notice that the contour information is line-based, while texture and non-texture information are region-based. To combine them, the contour information is first converted into a region-based form by the watershed technique to a fake gray scale image generated from the contour information. The pixels on the contour candidates and the periphery of the whole image are assigned with the highest intensity value, and then the image is dilated with descending intensities, hence the farthest pixels would form valleys and the contour and periphery pixels form peaks. Then watershed is carried out to divide the fake gray scale image into a number of regions as shown in Fig. 7. The contour candidates extracted in § 2.1 are preserved. Besides, unde-

sirable breaks are eliminated and object boundaries are integrated through watershed. Notice that the common over-segmentation problem of conventional watershed [13] is greatly relieved even without any post merging process.

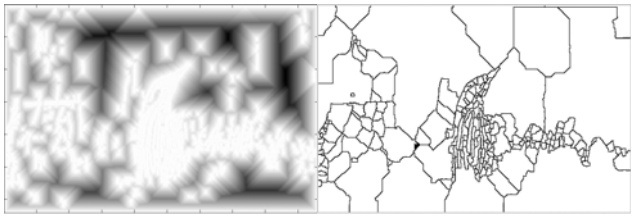

Fig. 7. Fake gray scale image produced from the contour information (left) and waterlines produced by watershed (right).

Here we explain how the texture map like the ones in Fig. 6 is utilized to extract the object from a scene. A rough object region is determined firstly. Sample points of the same feature cluster but spatially separated in the texture map (the left of Fig. 6) are regarded as different clusters. Since the object is supposed to be entirely included in the scene, we define background as the region of the two rows at the top and bottom and the two columns at the right end and left end. Then the clusters that include sample points falling in the peripheral region are erased as background. Afterwards, those clusters adjoining to each other are merged into one cluster and the largest cluster is extracted as the object or a part of the object. Via dilation and erosion, the sample points in the extracted cluster are fused into a unified region. Thus a rough object region is determined.

The texture-based discrimination well separates the object area from the background area, while their boundaries are too fuzzy to determine the boundary lines correctly.

In each basin (a segmented region) of the watershed image, the number of pixels belonging to the rough object region is counted. If the number exceeds 80% of all pixels in the basin, the basin is regarded as belonging to the object. Since the waterlines contains all salient features extracted in § 2.1, the boundary lines are correctly retrieved.

3. EXPERIMENTS

Simulation experiments are carried out on several natural scenes, some from the Berkeley segmentation dataset [14], and some from RuG natural image dataset [5], with varying object sizes, shapes, and various backgrounds. Due to space limitation only a few representative results are shown as examples (Fig. 8). As for the bird image on the top, our method extract the object region with clear and almost exact boundaries. There are still some imperfect fragments of background mistaken as a part of object, which mainly comes from the salient background. Such noise in the background is also difficult for human eyes in an instant view, and we mainly deal with such ambiguous boundary relying on our knowledge that things usually appear with smooth boundaries.

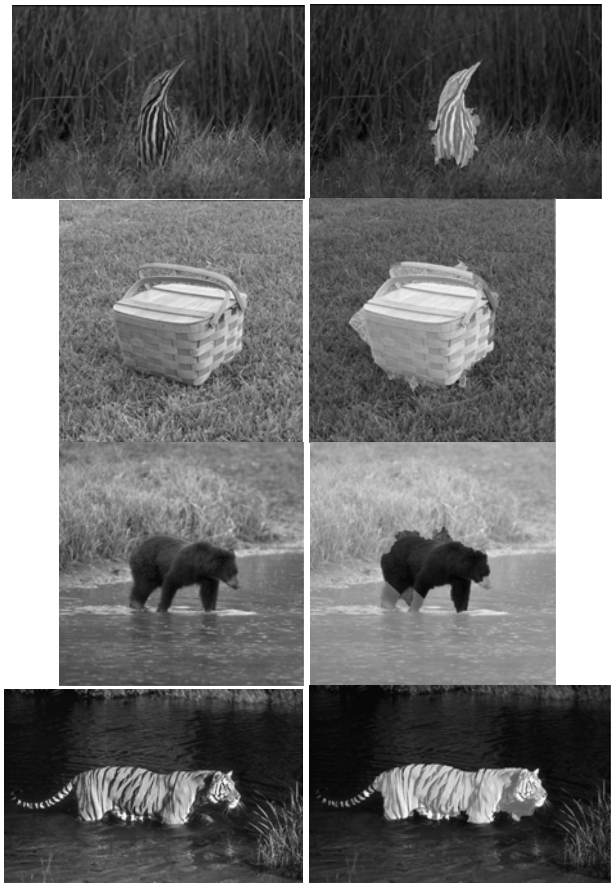

Fig. 8. The original image (left) and object region detected (right).

We have the human segmentation of the datasets modified to serve our object extraction task. And use the F-measure, which is the harmonic mean of precision and recall measures of the ground truth pixels. As for the above four images, we report an average F-measure score of 0.89, which is promising compared with 0.86 reported in [15].

In the tiger image at the bottom, the tail is lost due to the large window size in texture analysis step. To solve this problem, some of the discarded basins of small size (less than 300 pixels) and adjoining to the object region are recovered. Some results are shown in Fig. 9. The tail of the tiger missing in Fig. 8 was also restored.

Another point worth mentioning is that, all the techniques employed in this work are pixel-based, which provides a possibility of highly-parallel hardware implementation. In addition, we could expected that the present method would yield much better results as compared to other methods employing GMM instead of K-means, Euclidean distance or f-divergence instead of Manhattan distance, and so forth. Future work will focus on improving the conversion

of contour into region information and hardware implement of the system.

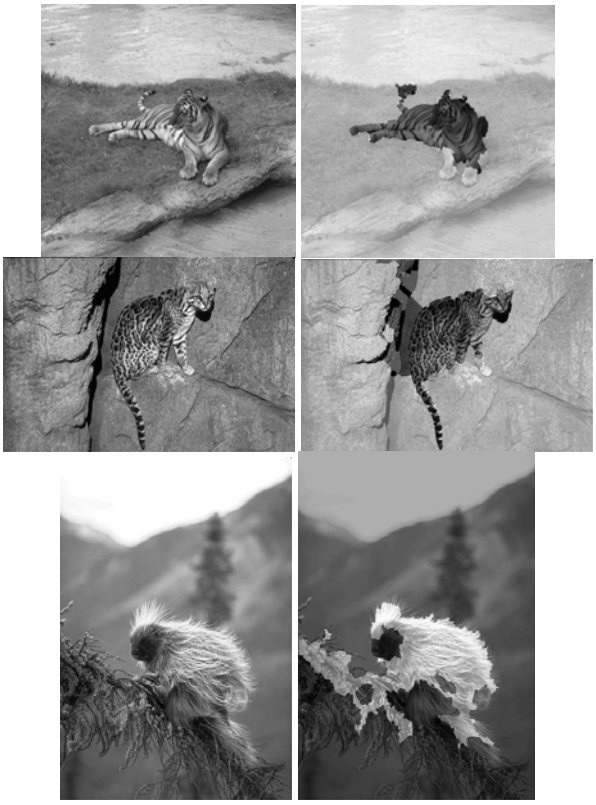

Fig. 9. The original image (left) and object region detected (right).

4. SUMMARY

We present an unsupervised object extraction system capable of separating animal images from background in natural scenes. Edges are detected from original images at multiple reduced-resolution images and used to identify object contours. A rough object region is determined utilizing texture information based on oriented edges. Contour is converted into a region-based form via watershed technique to complement the exact object boundaries. The method has been further applied to various natural scenes and has shown promising results.

5. REFERENCES

[1] N. Xu, R. Bansal, and N. Ahuja, "Object segmentation using graph cuts based active contours", Computer Vision and Pattern Recognition. IEEE, 2003, vol. 2, pp. II–46.

[2] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut: Interactive foreground extraction using iterated graph cuts," in ACM Transactions on Graphics (TOG). ACM, 2004, vol. 23, pp. 309–314.

[3] S. Kiranyaz, M. Ferreira, and M. Gabbouj, "Automatic object extraction over multi-scale edge field for multimedia retrieval," Image Processing, IEEE Transactions on, vol. 15, no. 12, pp. 3759–3772, 2006.

[4] J. Canny, "A computational approach to edge detection," PAMI 1986, IEEE Transactions on, no. 6, pp. 679–698.

[5] G. Papari and N. Petkov, "Adaptive pseudo dilation for gestalt edge grouping and contour detection," Image Processing, IEEE Transactions on, vol. 17, no. 10, pp. 1950–1962, 2008.

[6] Y. Ji, K.H. Chang, and C.C. Hung, "Efficient edge detection and object segmentation using gabor filters," in Proceedings of the 42nd annual Southeast regional conference. ACM, 2004, pp. 454–459.

[7] D.A. Klein and S. Frintrop, "Center-surround divergence of feature statistics for salient object detection," ICCV 2011, pp. 2214-2219.

[8] J. Ding, R. Ma, and S. Chen, "A scale-based connected coherence tree algorithm for image segmentation," Image Processing, IEEE Transactions on, vol. 17, no. 2, pp. 204–216, 2008.

[9] M. Yagi and T. Shibata, "An image representation algorithm compatible with neural-associative-processor based hardware recognition systems," Neural Networks, IEEE Transactions on, vol. 14, no. 5, pp. 1144–1161, 2003.

[10] S. Kim, S. Park, and M. Kim, "Central object extraction for object-based image retrieval," Image and Video Retrieval, pp. 523–528, 2003.

[11] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in Computer Vision, 1998. Sixth International Conference on. IEEE, 1998, pp. 839–846.

[12] H. Zhu and T. Shibata, "A real-time image recognition system using a global directional-edge-feature extraction vlsi processor," ESSCIRC'09. Proceedings of. IEEE, 2009, pp. 248–251.

[13] V. Osma-Ruiz, J.I. Godino-Llorente, N. S´aenz-Lech´on, and P. G´omez-Vilda, "An improved watershed algorithm based on efficient computation of shortest paths," Pattern Recognition, vol. 40, no. 3, pp. 1078–1090, 2007.

[14] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," ICCV 2001. IEEE, 2001, vol. 2, pp. 416–423.

[15] Alpert, S. and Galun, M. and Basri, R. and Brandt, A., " Image segmentation by probabilistic bottom-up aggregation and cue integration," CVPR 2007, pp. 1-8.