Interactive Image Segmentation Based on Level Sets of Probabilities

Yugang Liu and Yizhou Yu

Abstract—In this paper, we present a robust and accurate algorithm for interactive image segmentation. The level set method is clearly advantageous for image objects with a complex topology and fragmented appearance. Our method integrates discriminative classification models and distance transforms with the level set method to avoid local minima and better snap to true object boundaries. The level set function approximates a transformed version of pixelwise posterior probabilities of being part of a target object. The evolution of its zero level set is driven by three force terms, region force, edge field force, and curvature force. These forces are based on a probabilistic classifier and an unsigned distance transform of salient edges. We further propose a technique that improves the performance of both the probabilistic classifier and the level set method over multiple passes. It makes the final object segmentation less sensitive to user interactions. Experiments and comparisons demonstrate the effectiveness of our method.

Index Terms—Image segmentation, level set method, statistical classification, distance transform, curvature.

1 Introduction

Object cutout from images have proven to be a vital technology with many applications in computational photography, image synthesis as well as special visual effects for film making. These applications typically require pixel-level or even subpixel accuracy. With todays image processing and computer vision methods, computers still need human assistance in successfully performing this task at such a high level of accuracy. Hence, there has been much work on interactive object cutout.

Object cutout is essentially image segmentation that exploits region statistics and/or edge responses. Pure region-based segmentation makes use of region statistics and can produce more semantically meaningful results, but typically suffers from poor localization of region boundaries. On the other hand, pure edge-based methods offer accurate boundary localization, but need an extra propagation step to obtain completely closed region boundaries and usually do not have sufficient global knowledge to perform the task well.

An ideal solution would perform global object segmentation with integrated region and edge information so that both region statistics and local edge responses can be utilized. The result would still be regions but with accurate boundary localization at pixels where edge detection operators produce strong responses. The rest of the region boundaries can also be viewed as a viable solution to contour completion based on region information.

- *Y. Liu is with the Department of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China.*
- *Y. Yu is with the Department of Computer Science, University of Illinois at Urbana-Champaign, 201 North Goodwin Ave, Urbana, IL 61801. E-mail: yyz@uiuc.edu.*

Manuscript received 9 June 2010; revised 15 Feb. 2011; accepted 2 Mar. 2011; published online 12 Apr. 2011.
Recommended for acceptance by G. Drettakis.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number TVCG-2010-06-0117.
Digital Object Identifier no. 10.1109/TVCG.2011.77.

In this paper, we adopt the level set method for interactive object cutout that integrates region statistics and edge responses. Performing object cutout using the level set method has obvious advantages. First, the zero level set can represent the boundaries of an object with an arbitrary topology. It is also very convenient to evolve the topology of the zero level set during the solution process. Thus, the level set method can be effectively used for extracting a foreground layer with fragmented appearances, such as leaves. Second, it is possible to make the zero level set snap to relatively distant salient edges by devising a force based on the location of these edges. We achieve this goal using a specially designed edge distance field.

Nevertheless, a serious limitation of existing level set algorithms for image segmentation is that the final result is sensitive to the location of the initialization. This is because level set evolution is typically driven by forces computed from local image data. We overcome this problem by integrating the level set method with a probabilistic pixel classifier. This classifier encodes characteristic statistics of pixels belonging to a target object. Given the attributes of a pixel, it outputs an estimated likelihood of the pixel being part of the target object. To integrate the classifier with the level set method, the level set function is defined to approximate the posterior probabilities of the pixels. We use the estimated likelihoods from the classifier as an initialization, which is further improved by the level set method. Since an accurate classifier does not exist at the beginning, we alternate classifier training and the level set method to improve the performance of both.

In summary, the key contributions of this paper are as follows:

- A supervised image segmentation method that integrates the level set method with a per-pixel probabilistic discriminative classifier. Because of such an integration, the level set function is defined to approximate posterior probabilities of pixels being part of a target object. The level set function

is initialized with the likelihoods produced by the per-pixel classifier, making the level set method less vulnerable to local minima. Comparisons with generative models for classification, such as Gaussian mixture models, have confirmed that the discriminative model we use is a more effective per-pixel classifier, and makes the level set algorithm achieve higher quality results.

- A specially designed edge field to make the zero level set in our method snap to distant object boundaries. This is achieved with a technique for suppressing spurious image edges inside texture regions and an unsigned distance transform of remaining salient edges which typically align with true object boundaries. Experiments have shown that this edge field can effectively guide the zero level set toward object boundaries. Although a signed distance transform of the zero level set has traditionally been used as the level set function itself, the use of such an edge field with spurious edge suppression is novel in the context of the level set method.
- A method that improves the performance of both the per-pixel classifier and the level set method over multiple passes. The per-pixel classifier is retrained at the end of every pass, and the training data are obtained from the latest segmentation result generated by the level set method. The retrained classifier helps the level set method obtain an improved segmentation result in a subsequent pass. Although such an EM approach has been employed in other image segmentation frameworks, such as graph cut based segmentation, its adoption in the level set method is novel. It makes final object segmentation less sensitive to random factors in user interaction, such as the exact location where a box is drawn.

2 Background and Related Work

2.1 Interactive Segmentation

Many interactive techniques have been developed for object cutout from still images [1], [2], [3], [4], [5], [6]. Gaussian mixture models have been frequently used for color distributions. Final results in recent work have been achieved with the graph cut algorithm [7]. Graph cut is a popular method and can be employed to solve a variety of image segmentation problems. One of its limitations is that its energy function can only model pairwise interactions between pixels or patches. Thus it is hard for graph cut algorithms to consider global statistics and higher order geometric properties, such as the curvature of object boundaries.

An important inspiration of our technique came from user-defined scribbles or bounding boxes which have been extensively exploited among aforementioned interactive segmentation techniques. For example, a user drags a bounding box around a target object in [3] and [6]. However, we use user-drawn boxes in a different way from these methods. We do not require a user-drawn box to be a bounding box of the target object, but instead use a histogram computed from pixels inside the box to initialize a foreground region, as described in Section 6.1.

2.2 Levelset-Based Segmentation

An introduction to the level set method and its applications can be found in [8], [9], and [10]. The level set method has been applied to supervised image segmentation in [11] where a user can provide hints by drawing a convex hull of the foreground object. A coupled variational framework was proposed to integrate boundary, region, and texture information. However, this work only adopts a relatively simple statistical model and does not consider distant interactions with edge pixels. As a result, region boundaries are often trapped in local minima, and do not snap to true object boundaries.

The majority of previous work on level set based image segmentation is unsupervised. An active contour model based on the Mumford-Shah functional and the level set method was developed in [12]. This method can detect object boundaries in blurry or noisy images. The relationships between the boundary and region functionals in level set based segmentation were analyzed in [13]. On the other hand, methods for incorporating shape, intensity and curvature prior information into level set based segmentation have been developed in [14] and [15]. At each step of level set surface evolution, they estimate the maximum a posteriori position and shape of the object in the image. However, these methods require much prior information to start with, and are not appropriate for general interactive segmentation where it is desired to have little user interaction. Variational frameworks with shape priors have been proposed in [16], [17], and [18]. In particular, Rousson and Paragios [17] integrate a shape prior with geodesic active regions [11], and the method in [18] introduces a PCA-based statistical shape prior for active contours. Our method in this paper does not rely on any shape priors.

Recently, methods [19], [20] for unsupervised texture segmentation based on active contours have been introduced. The method in [19] integrates active contours with Gabor filters, while [20] proposes an intrinsic texture descriptor based on the Beltrami representation, semilocal image information and the metric tensor. It chooses to maximize the Kullback-Leibler distance between the probability density functions of the foreground and background. Because of their unsupervised nature, results presented in these papers do not have very accurate object boundaries.

2.3 Segmentation Based on Supervised Classification

Supervised classification has been recently integrated with image segmentation in [21], [22], [23], [24], [25], and [26]. Boosting [27] has been adopted to improve the performance of a supervised classifier in [22], [23], and [26] to learn the statistics of an object region. Among them, Shotton et al. [23] detail automatic visual understanding and segmentation based on Joint Boost [28]. It uses a conditional random field (CRF) model to learn a conditional distribution over the class labels given an image. The gentleboost classifier was adopted in [24] to generate an initial solution for edge-aware interpolation of local image adjustments. Classical AdaBoost was used in [26]. Instead of boosting, an ensemble of overlapping local classifiers was adopted in [25] to achieve accurate pixel classification for video object cutout.

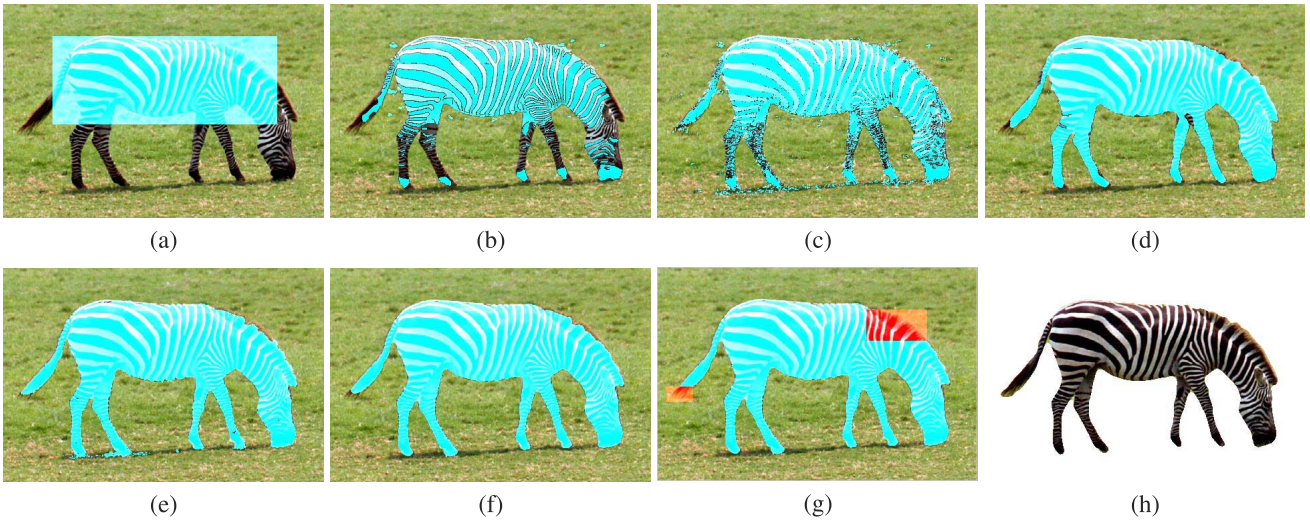

Fig. 1. The pipeline of our method. (a) initial user-drawn rectangular region, (b) initial foreground pixels selected by the global color histogram, (c) the output from the classifier in the first pass, (d) the output from the level set method at the end of the first pass, (e) the output from the classifier in the second pass, (f) the output from the level set method at the end of the second pass, (g) two rectangular regions with their own local classifiers in the refinement step (optional), and (h) the final segmentation result.

3 Overview

Our method consists of three stages: preprocessing, multipass level set method, and refinement. In the preprocessing stage, we perform various low-level image processing operations, including edge detection, Gabor filtering, and color histogram computation. Results from the preprocessing stage benefit later stages. For instance, detected edges will be used for boundary localization and Gabor filter responses will be used for discriminating different texture regions. The preprocessing stage has linear complexity because it consists of a majority of filtering operations, which has constant complexity per pixel.

The multipass level set method is a vital part of our algorithm. During this stage, the estimation of the posterior probability of every pixel being part of the target object is iteratively improved. The user initiates the process by drawing one or few rectangular regions over the target object. These rectangles do not need to be bounding boxes completely enclosing the object. We compare a histogram of the pixels within the boxes with a histogram of the entire image to initialize the likelihood of every histogram bin. Such likelihoods are then used for training a supervised probabilistic classifier, which in turn is used to estimate a likelihood for every pixel. As we know, independent pixelwise classifications are typically noisy and lack spatial coherence (Fig. 1c). Thus, we use pixelwise likelihood generated from the classifier to seed initial object boundaries (the zero level set), which are then iteratively evolved to improve both boundary localization and boundary smoothness using the level set method until convergence. Here both spatial coherence and boundary localization act as priors, and the converged result from the level set method is considered as the pixelwise posterior probabilities. We have designed three force terms for the level set method to address boundary localization, boundary smoothness as well as consistency with classifier results.

During each pass, we run the level set method until convergence. At the end of each pass, the pixelwise posterior probabilities from the level set method represent an intermediate segmentation result, which is used for training a new probabilistic classifier with improved accuracy. This new classifier will be used for generating pixelwise likelihood at the beginning of a subsequent pass, where the level set method is called again to compute improved pixelwise posterior probabilities. We thus alternate classifier training and the level set method in multiple passes to reach the final segmentation result. Typically 2-3 passes are sufficient to generate good results. Fig. 1 illustrates this entire process.

The segmentation from the aforementioned process may still have minor inaccuracies and edge misalignments. We run a final refinement stage to resolve them. In the refinement stage, the level set method is performed with larger boundary localization and smoothness force terms. Since some of the inaccuracies may be caused by misclassifications, before running the adjusted level set method, the user may choose to train local classifiers in restricted regions along the object boundary and use such local classifiers to correct segmentation in such regions. Fig. 1h shows a refined result.

4 Preprocessing

In the preprocessing stage, we perform various low-level image processing operations, including edge detection, edge field computation, Gabor filtering, and projected color histogram computation.

4.1 Weighted Canny Edges

The Canny edge detector (including its variants) is still a state-of-the-art edge detector [29]. Canny edge detection obtains more precise results than the Sobel operator because of its nonmaximum suppression. Unless in an unusual setup, it is hard to find an edge detector that performs significantly better than the Canny edge detector.

Traditional Canny edge detection [30] works on gray scale images and produces binary edge detection results. For color images, we first compute weighted Canny edges on each color channel independently. This is followed by a step to have such independent results merged. The strength

of an edge pixel is set to the magnitude of the gradient at that pixel. We merge the edge strengths from three color channels by collecting the strengths from all channels into a 3-vector and computing the L^2 norm of this vector. Fig. 3b shows the weighted edges of Fig. 3a.

To make region boundaries (the zero level set) quickly snap to salient edges without being trapped in local minima, we need a global mechanism to facilitate distant interactions between detected edges and level sets. We achieve this goal by computing an unsigned distance transform of the edges using a revised version of the Fast Marching Method in [9]. Since edges are not closed curves, we cannot compute a signed distance transform. We further clamp and normalize the distances using a prescribed maximal distance, d_{max}.[1] The result is called an *edge field*

$$\Psi(\mathbf{x}_i) = \begin{cases} 0 & \text{for} \quad \mathbf{x}_i \in S_e, \\ \frac{\min(d_{max}, \min_{\mathbf{x}_j \in S_e} d(\mathbf{x}_i, \mathbf{x}_j))}{d_{max}} & \text{for} \quad \mathbf{x}_i \notin S_e, \end{cases} \quad (1)$$

where S_e is the set of edge pixels, and $d(\mathbf{x}_i, \mathbf{x}_j)$ is the euclidean distance between two pixels $\mathbf{x}_i$ and $\mathbf{x}_j$.

4.2 Gabor Filtering

Texture descriptors based on Gabor filter responses are built in this step. Oriented filter banks have proven to be an effective method to characterize textures [31], [32]. We primarily use local statistics of oriented filter responses to differentiate textures. We apply 24 Gabor filters [31] at six orientations and four scales at every pixel, and such filtering is performed for each of the three color channels separately. Thus, every pixel has a 72-component filter response vector.

4.3 PCA-Based Color Histograms

Image color histograms represent a rich source of information [33]. However, directly building a three-dimensional color histogram would not be space efficient. On the other hand, computing three independent histograms for three color channels, respectively, would not be able to capture the correlation among the color channels. We propose a PCA-based color histogram. We first sample a number of pixels from an image and represent the color of every sampled pixel as a three dimensional vector. Principal component analysis is then applied to this set of vectors to extract the first principal direction. We use this principal direction to compute a one-dimensional color histogram. This is achieved by projecting all the pixel colors in the image onto this principal direction and then depositing the scalar projections into their corresponding histogram bins.

We build two types of color histograms using the extracted principal direction. To build a global histogram for an entire image, we choose to set up a large number of bins, typically 1,000 to 2,000. This global histogram is primarily used for distinguishing the target foreground object from the background. We also build a local color histogram for the neighborhood surrounding every pixel. The number of bins for such histograms is much smaller, and typically set to be the same as the number of components in the Gabor filter response vector. Both the filter response vector and the color histogram are concatenated together to form the *texture descriptor* of a pixel. In our experiments, the neighborhood size is typically set to 9×9.

1. d_{max} is set to (Image Width + Image Height)/8.

5 Level Sets of Probabilities

Given an input image I, the goal of our level set method is to obtain a posterior probability for every pixel. This posterior probability defines the likelihood of a pixel being part of a target object given the evidences from the entire image. We use the level set function, $\Phi(\mathbf{x}, t)$,[2] to achieve an increasingly better approximation of such posterior probabilities over time. Since probabilities fall into $[0, 1]$ while the values of our level set function belong to $[-1, 1]$ with positive values falling outside the zero level set, probabilities and level sets have the following relationship:

$$\Phi(\mathbf{x}) = -2(P(l(\mathbf{x}) = 1|I) - 0.5), \quad (2)$$

where $l(\mathbf{x})$ denotes the label of pixel $\mathbf{x}$, and $P(l(\mathbf{x}) = 1|I)$ represents the posterior probability of pixel $\mathbf{x}$ being part of the foreground target object. Once we have obtained the final solution of the level set function, the final object segmentation is defined accordingly as follows:

$$l(\mathbf{x}) = \begin{cases} 1, & \Phi(\mathbf{x}) \leq 0, \\ 0, & \Phi(\mathbf{x}) > 0, \end{cases} \quad (3)$$

where 1 represents foreground pixels and 0 represents background pixels.

5.1 Region Term

Very often, the information available from a local image neighborhood is already sufficient to perform foreground/background classification on a pixel. Such information includes pixel colors, local histograms, and local Gabor filter responses. To fully utilize local pixel classification results in global image segmentation, our first goal is to make the global posterior probabilities close to the likelihoods estimated by a local pixel classifier. We use the following energy term E_R to measure the degree of inconsistency between the two and then try to minimize this energy

$$E_R = \int\int_I (\Phi(\mathbf{x}) + 2(P(l(\mathbf{x}) = 1|N(\mathbf{x})) - 0.5))^2 d\mathbf{x}, \quad (4)$$

where $P(l(\mathbf{x}) = 1|N(\mathbf{x}))$ denotes the likelihood of pixel $\mathbf{x}$ being part of the target object according to a local discriminative probabilistic classifier, which only gathers evidences available from a local neighborhood $N(\mathbf{x})$.

Given initial user-supplied scribbles or boxes, such a probabilistic pixel classifier can be trained. The output of this classifier should be a continuous likelihood value between 0 and 1. Logistic regression can satisfy this requirement. In practice, we train probabilistic boosting trees [34] as the discriminative classifier using pixelwise texture descriptors defined at the end of Section 4. Suppose this classifier is reasonably accurate, its likelihood estimation should be large than 0.5 over a majority of pixels inside the regions defined by the target object. This is indeed the case. Fig. 2 demonstrates that the probabilistic classifier we use achieves higher classification accuracy than Gaussian mixture models.

2. In the rest of the paper, when all quantities in an equation refer to the same time step, we drop the temporal variable t in the notations.

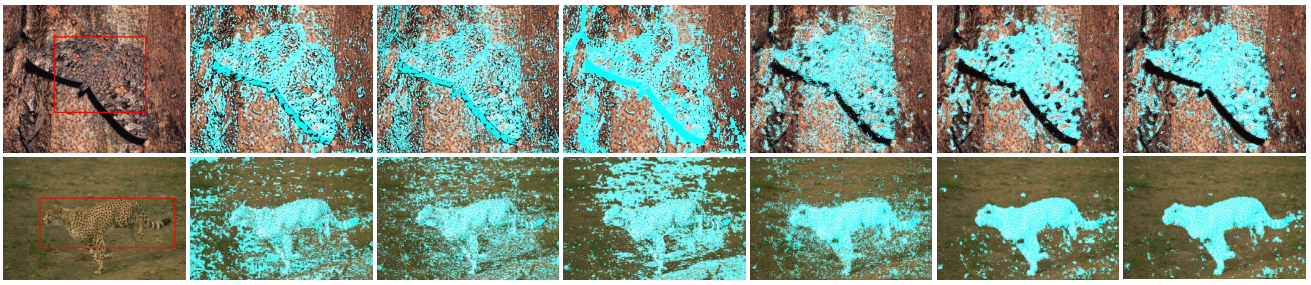

Fig. 2. Comparisons of classification accuracy between a probabilistic discriminative classifier and the Gaussian Mixture Model (GMM). (a) initial user-drawn box, (b) classification results using a GMM obtained from one iteration of Expectation Maximization (EM), (c) classification results using a GMM obtained from 3 iterations of EM, (d) classification results using a GMM obtained from 100 iterations of EM, (e) classification results using a probabilistic discriminative classifier [34] obtained from one iteration of EM, (f) classification results using a classifier of the same type but with two iterations of EM, (g) classification results from a classifier of the same type but with 3 iterations of EM. Because of the limited discriminative power of a GMM, even a large number of EM iterations are unable to improve its classification accuracy while the classification accuracy of the discriminative classifier we use can be significantly improved over a small number of EM iterations.

We design the following force term to reduce the energy defined in (4):

$$R(\mathbf{x})\frac{\nabla\Phi}{\|\nabla\Phi\|} = (\Phi(\mathbf{x}) + 2(P(l(\mathbf{x}) = 1|N(\mathbf{x})) - 0.5))\frac{\nabla\Phi}{\|\nabla\Phi\|}, \tag{5}$$

where $\frac{\nabla\Phi}{\|\nabla\Phi\|}$ represents the unit outward normal vector of the zero level set. Suppose $\mathbf{x}$ is a point on the zero level set. $P(l(\mathbf{x}) = 1|N(\mathbf{x})) > 0.5$ means $\mathbf{x}$ is considered inside the target object by the local classifier. To make the posterior probability consistent with the local classifier, the zero level set should be expanded along the outward normal direction, which is consistent with the force term prescribed in (5). On the other hand, $P(l(\mathbf{x}) = 1|N(\mathbf{x})) < 0.5$ means $\mathbf{x}$ is considered outside the target object by the local classifier. To make the posterior probability consistent with the local classifier, the zero level set should be shrunk along the reversed normal direction, which is also consistent with the force term prescribed in (5).

5.2 Edge Field Term

Our second goal is to make the zero level set snap to salient edges in the image because salient edges are likely to lie on the boundary of the target object. We rely on the edge field computed in the previous section to facilitate distant interactions between edges and level sets. Since the edge field reaches its minimal value at edge pixels and has a magnitude increasing monotonically with the distance from edge pixels, we design the following energy term E_B to measure the overall proximity between the zero level set and the set of detected edge pixels:

$$E_B(\Gamma) = \oint_{\Gamma} \Psi(\Gamma(s))\|\dot{\Gamma}(s)\|ds, \tag{6}$$

where $\Psi(\mathbf{x})$ is the edge field defined in (1), $\Gamma(s)$ is a 1D parametric representation of the zero level set and $s \in [0, 1]$, $\|\dot{\Gamma}(s)\|$ represents the derivative of the arc length of the zero level set with respect to the parameter. To minimize the energy E_B, we apply the Euler-Lagrange equations to (6), and obtain the following equation for optimizing the position of points on the zero level set:

$$\frac{d\mathbf{x}}{dt} = -(\Psi(\mathbf{x}(s))\kappa(\mathbf{x}(s)) + \nabla\Psi(\mathbf{x}(s)) \cdot \mathbf{N}(\mathbf{x}(s)))\mathbf{N}(\mathbf{x}(s)), \tag{7}$$

where $\kappa(\mathbf{x})$ is the curvature of the zero level set at $\mathbf{x}$ and $\mathbf{N}(\mathbf{x})$ is the outward normal of the zero level set at $\mathbf{x}$. The first term in (7) tries to make the zero level set as straight as possible while the second term tries to move the zero level set toward relatively distant edges along the negative gradient of the edge field. Both terms can reduce the energy term defined in (6).

The first term in (7) concerns the smoothness of the detected object boundary, which we will be addressed in the next section. Therefore, here we only focus on the second term, which improves boundary localization. Since the normal of the zero level set can be formulated using the gradient of the level set function, we can replace the unit normal vector $\mathbf{N}(\mathbf{x})$ in (7) with the normalized gradient, $\frac{\nabla\Phi(\mathbf{x})}{\|\nabla\Phi(\mathbf{x})\|}$. Thus, we design the second force term for boundary localization as follows:

$$B(\mathbf{x})\frac{\nabla\Phi}{\|\nabla\Phi\|} = -\left(\nabla\Psi(\mathbf{x}) \cdot \frac{\nabla\Phi(\mathbf{x})}{\|\nabla\Phi(\mathbf{x})\|}\right)\frac{\nabla\Phi}{\|\nabla\Phi\|}, \tag{8}$$

Although edge pixels are very useful in shaping the boundary, in the presence of textures in the interior of the target object, there may exist too many spurious edges in the texture regions, as shown in Fig. 3b. Such spurious edges in the middle of an object region may severely interfere with boundary localization, making the zero level set converge to certain edges inside the object region instead of edges on the object boundary. We use the pixel classifier trained for the region term to identify edges lying on the true object boundary and suppress edges inside texture regions. The basic idea is to look at the likelihoods returned by the classifier within a neighborhood of every edge pixel. If an edge pixel lies on the true object boundary, its neighborhood has a significant percentage of pixels both inside and outside the object region, and their likelihoods exhibit two extreme values, giving rise to a large variance. On the other hand, if the edge pixel lies inside a texture region, its neighborhood mostly consists of either foreground or background pixels, and their likelihoods would be similar, giving rise to a small variance. Thus, we suppress edges inside texture regions according to centered and normalized standard deviation of the likelihoods in their neighborhoods. The detailed steps of our edge suppression technique can be found in Algorithm 1. Figs. 3c and 3d show a

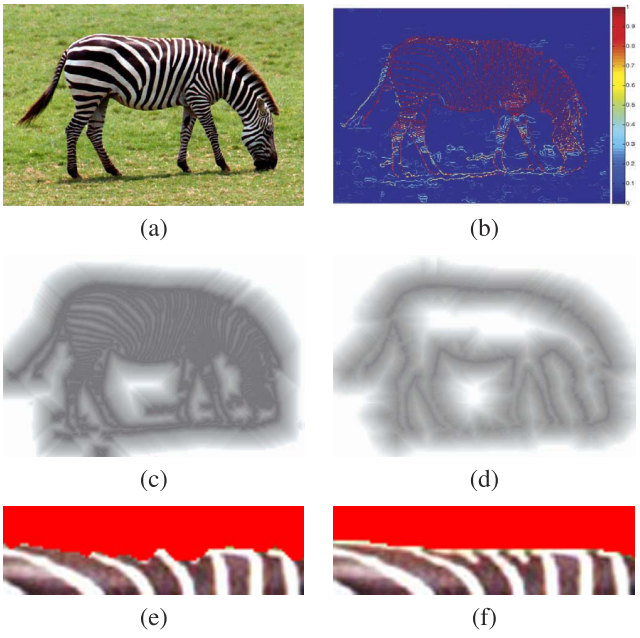

Fig. 3. Edge suppression. (a) Original image, (b) weighted Canny edges, (c) edge field computed directly from (b), (d) edge field computed after edge suppression. In (b)-(d), white represents high values; black represents low values, (e) segmentation boundary using an edge field without spurious edge suppression, (f) segmentation boundary using our edge field with spurious edge suppression.

comparison of edge fields computed from all edges and edges surviving suppression, respectively.

Algorithm 1. Edge Suppression

Declarations: C (weighted Canny edges), P (pixelwise foreground probability), S (standard deviation of probability), θ_e (a threshold), $\mathbf{x}_i$ (pixel).
procedure EDGE_SUPPRESSION(C, P, θ_e)
  Initialize all $S(\mathbf{x}_i) \leftarrow 0$, $\Psi(\mathbf{x}_i) \leftarrow 0$
  for $C(\mathbf{x}_i) > 0$ **do**
    $S(\mathbf{x}_i) \leftarrow$ standard deviation of $P(\mathbf{x}_i)$ within a 9×9 neighborhood of pixel $\mathbf{x}_i$
  end for
  $m \leftarrow$ mean value of all $S(\mathbf{x}_i) > 0$
  $d \leftarrow$ standard deviation of all $S(\mathbf{x}_i) > 0$
  for all $S(\mathbf{x}_i)$ **do**
    $S(\mathbf{x}_i) \leftarrow (S(\mathbf{x}_i) - m)/d$
  end for
  for $C(\mathbf{x}_i) > 0$ **do**
    if $(C(\mathbf{x}_i) + S(\mathbf{x}_i)) > \theta_e$ **then**
      Add pixel $\mathbf{x}_i$ to the set of edge pixels S_e
    end if
  end for
  $\Psi \leftarrow$ edge field of Ψ
end procedure
Result: Ψ is the edge field after edge suppression.

5.3 Curvature term

The curvature term is a standard force term in level set methods. It is primarily used for improving boundary smoothness. However, it is unnecessary to enforce boundary smoothness unconditionally, especially when the true object boundary has rough details. Our curvature term tries to provide a trade-off between boundary smoothness and boundary faithfulness. That means in the neighborhood of unsuppressed edge pixels, boundary localization is still a more important goal than boundary smoothness. But in the absence of edge pixels, boundary smoothness serves as an effective prior to determine the shape and position of the local object boundary. Thus, we define the curvature force term as follows:

$$C(\mathbf{x})\frac{\nabla\Phi}{\|\nabla\Phi\|} = -(\mu\kappa(\mathbf{x}) + (1-\mu)\Psi(\mathbf{x})\kappa(\mathbf{x}))\frac{\nabla\Phi}{\|\nabla\Phi\|}, \quad (9)$$

where $\kappa(\mathbf{x})$ denotes the curvature of the level set passing through $\mathbf{x}$, and $0 \le \mu \le 1$. Locally convex regions have $\kappa > 0$ on the boundary while locally concave regions have $\kappa < 0$ on the boundary. The second term in (9) modulates curvature with the edge field to weaken the curvature term in the neighborhood of edges. Nevertheless, the first term guarantees that the curvature term does not disappear completely as long as μ remains positive.

The curvature of a level set in a two dimensional level set method can be computed using the following equation, which has been proved in [15]:

$$\kappa = \frac{\Phi_{xx}^{sd}\Phi_{y}^{sd2} - 2\Phi_{x}^{sd}\Phi_{y}^{sd}\Phi_{xy}^{sd} + \Phi_{yy}^{sd}\Phi_{x}^{sd2}}{\left(\Phi_{x}^{sd2} + \Phi_{y}^{sd2}\right)^{\frac{3}{2}}}, \quad (10)$$

where Φ^{sd} represents a signed distance transform of the zero level set in our method. Since it is extremely unlikely to have a circle with a radius smaller than the size of a pixel, we clamp any computed curvature to $[-\frac{1}{\Delta x}, \frac{1}{\Delta x}]$, where Δx represents the size of a pixel [10]. Since the default level set function in our method is defined using probabilities, the signed distance transform of the zero level set needs to be recomputed during every time step.

5.4 Overall Scheme

Our level set method embeds a propagating front $\Gamma(s,t)$ as the zero level set of a level set function $\Phi(\mathbf{x},t)$ (i.e., $\Gamma(s,t) = \{\mathbf{x}|\Phi(\mathbf{x},t) = 0\}$). At any given time t, this level set function tries to approximate a transformed version of the pixelwise posterior probabilities of being part of a target object while its zero level set tries to snap to the true boundary of the target object. The evolution of the zero level set is driven by three force terms discussed in the previous sections, the *region force*, *edge field force*, and *curvature force* terms. The level set method tracks the evolution of the front by numerically solving the following differential equation and extracting the zero level set of the solution:

$$\frac{\partial\Phi}{\partial t} = -\left[\underbrace{\alpha \cdot R(\mathbf{x},t)}_{\text{region}} + \underbrace{\beta \cdot B(\mathbf{x},t)}_{\text{boundary}} + \underbrace{\gamma \cdot C(\mathbf{x},t)}_{\text{curvature}}\right]\|\nabla\Phi\|, \quad (11)$$

where $\mathbf{x}$ denotes pixel coordinates in the image and t denotes the time of advection. The computation of $\nabla\Phi$ is based on *upwind differencing* in [10]. The initial level set function is defined according to the probabilistic classifier. That means, instead of initializing the zero level set to be the borders of the user-drawn boxes, we train a probabilistic

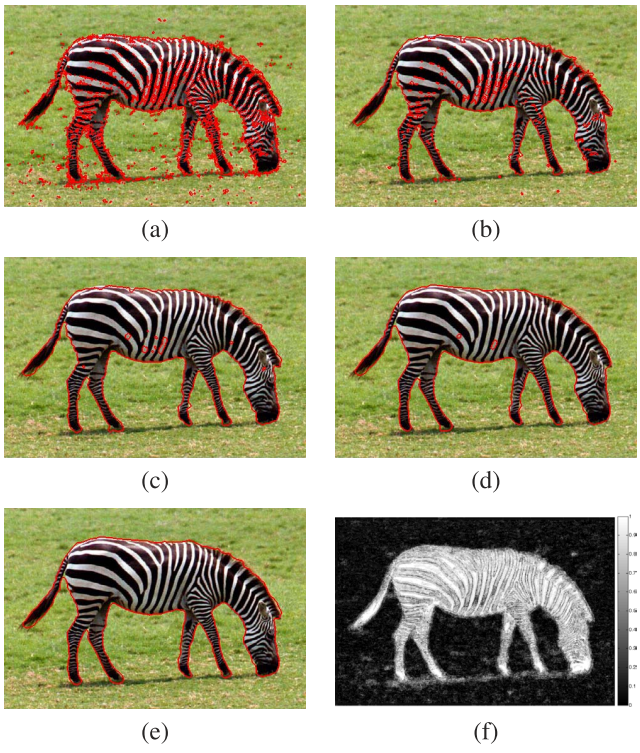

Fig. 4. The evolution of the zero level set of probabilities. (a) an initial zero level set initialized by the probabilistic classifier, (b) an updated zero level set at time step 1, (c) an updated zero level set at time step 2, (d) an updated zero level set at time step 3, (e) the final zero level set, and (f) final pixelwise posterior probabilities.

classifier according to the user-given hints (Section 6.1) and use the pixelwise likelihoods produced by this classifier as the initial level set function.

The equation in (11) can be efficiently solved using the Narrow Band Method in [9] and [10] and the Fast Local Level Set Method in [35]. They restrict most computation to a narrow band of active pixels immediately surrounding the zero level set. In general, the narrow band is the following set of pixels, $\{\mathbf{x} : |\Phi^{sd}(\mathbf{x})| < \theta_b\}$ where θ_b is a prescribed threshold (typically set to 10), and Φ^{sd} is the signed distance transform of the zero level set. Note that we only need to update the signed distance transform within the narrow band during every time step.

The evolution of the zero level set driven by the force in (11) is illustrated in Fig. 4. The initial zero level set is fragmented because of the noisy pixelwise likelihoods from the classifier. Note that although being fragmented, these contours spread over the entire foreground object, making the level set method less likely to be stuck in local minima. These initial contours are evolved by the level set method. They gradually merge with each other and also move toward true object boundaries to improve their localization and spatial coherence. Thus, our segmentation algorithm can also be viewed as an advanced version of region-split-and-merge algorithms.

6 Multipass Level Set Method

Our level set method heavily relies on the accuracy of the pixel classifier because the region force term is defined with reference to the probability returned by this classifier and the edge field relies on this classifier to suppress spurious edges. Nevertheless, the accuracy of a classifier depends on the quality of its training data and we do not have high-quality training data at the beginning because we only require the user to draw one or few rectangular boxes that roughly cover the target object. However, we believe the posterior probabilities returned by the level set method is more accurate than the initial likelihoods provided by the classifier because the posterior probabilities have improved spatial coherence and conform better to the edge field. We can potentially obtain a better classifier if we use the posterior probabilities as training data. In this way, we cannot only incorporate a classifier to achieve better results in the level set method, but also use higher quality training data provided by the level set method to obtain an improved classifier. Therefore, we decide to alternate classifier training and the level set method multiple times to achieve increasingly better results. The steps of this multipass level set method is illustrated in Algorithm 2. During every pass, we update the edge field according to Algorithm 1 by suppressing spurious edges using pixelwise probabilities provided by the most recent classifier.

Algorithm 2. Multipass Level Set Method

Declarations: C (weighted Canny edges), P (pixelwise foreground probability), Φ (*level set function*), Ψ (edge field), n (number of passes).
procedure MULTIPASS_LEVEL_SET
  while $n \geq 0$ **do**
    $n \leftarrow n - 1$
    Data Sampling:
    Construct a set M of randomly sampled pixels
    if $\Phi(\mathbf{x}_i) \leq 0$ and $i \subset M$ **then**
      add pixel $\mathbf{x}_i$ to positive training set F
    else
      add pixel $\mathbf{x}_i$ to negative training set B
    end if

    Classifier Updating:
    Train a new probabilistic classifier with F and B
    Recompute P with the new classifier
    $\Psi = \textbf{Edge\_Suppression}\{C, P, \theta_e\}$

    Level Set Method:
    Re-initialize the level set function Φ using P
    Run the level set method using P and Ψ
      until convergence
  end while
end procedure
Result: The label of all pixels computed from the final level set function Φ

Our multipass scheme is insensitive to initial user interactions and capable of gradually improving the result. Some intermediate results of our multipass level set method are shown in Fig. 5. It is evident that the summation of the region and edge field energy terms drops quickly over

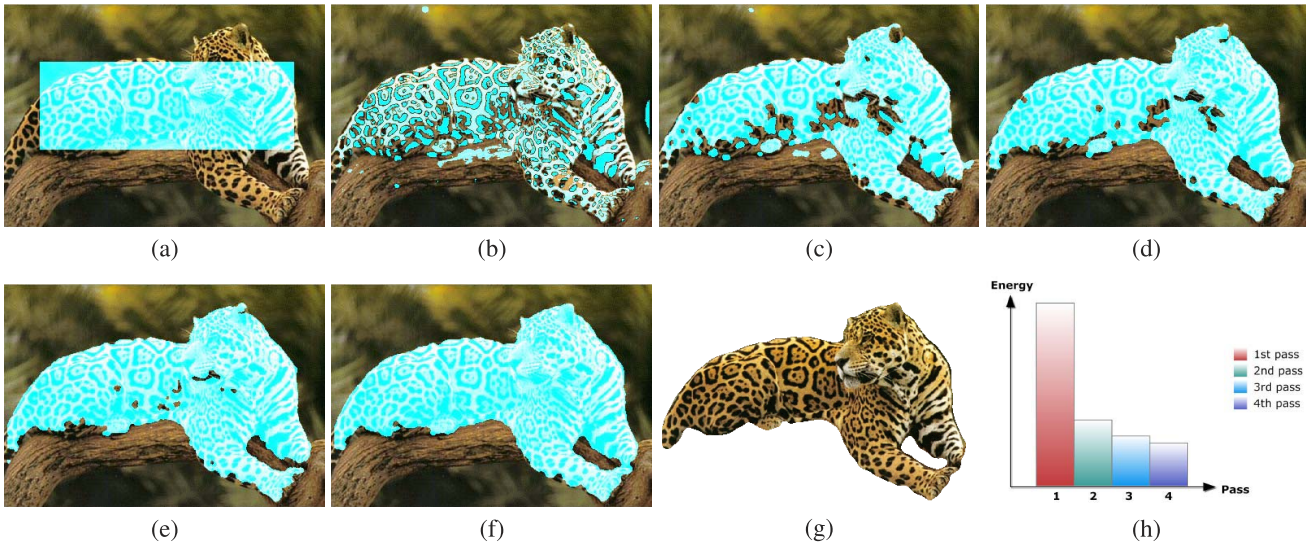

Fig. 5. Multipass level set method. (a) Initial user-drawn region, (b) foreground pixels selected by the global color histogram, (c) foreground segmentation after the first pass, (d) the second pass, (e) the third pass, (f) the fourth pass, (g) the final segmentation result, and (h) region and edge field energy after each pass.

multiple passes. Although there is no theoretical proof, our multipass scheme always converged in all our experiments.

6.1 Initial Training Data

Since we adopt a probabilistic classifier, we need probability values in the initial training data. Such probability values are not readily available from the user interactions. Therefore, we develop the following scheme based on the global histogram to generate the initial probabilities. Let $\{C_1, C_2, \ldots, C_n\}$ be the number of pixels in the entire image that fall into the global histogram bins, and $\{B_1, B_2, \ldots, B_n\}$ be the number of pixels in the user-drawn boxes that fall into the global histogram bins. Then it is straightforward to compute the a priori probability of global histogram bin h_i as $P(h_i) = \frac{B_i}{C_i}$. Thus, every pixel in the image can be assigned a probability according to which histogram bin it falls into. Such pixelwise probabilities are spatially smoothed using a Gaussian filter. The mean of these smoothed probabilities is used as a threshold to classify all pixels into foreground and background pixels. The set of foreground pixels is used to update $\{B_1, B_2, \ldots, B_n\}$. This process is repeated 3-5 times to obtain a set of initial foreground pixels and their associated probabilities as positive training data. The rest of the pixels are used as negative training data. The Gaussian filter we use incorporates an edge mask. That is, if two pixels are divided by an edge, they do not influence each other. Fig. 1b shows an example of initial training data obtained in this way.

6.2 Refinement

We provide two schemes to refine the segmentation result obtained from our multipass level set method, local classifiers, and boundary refinement. For the first scheme, we provide an interface for a user to draw a rectangular region where the result needs further improvement. The latest foreground/background labels as well as pixelwise probabilities are used as training data to obtain a local classifier within the rectangular region. Finally, we update the foreground/background labels in the region using the local classifier. This refinement scheme is shown in Figs. 1g and 1h.

In the second scheme, we further improve boundary localization and smoothness by running the level set method in (11) with larger values for β and γ with respect to α. This scheme makes the zero level set cling to edges and removes high-curvature artifacts along the segmentation boundary. A refinement result is shown in Fig. 6.

7 Experimental Results

We have implemented our multipass level set method on an Intel Core 2 Duo 3.0 GHz processor with 2 GB RAM and successfully applied it to a large number of images (Figs. 1, 5, 8, 9, 10, and 11). A user only needs to draw one or at most two rectangular boxes over a target object in every image, and our technique will take that as a hint and iteratively obtain more and more accurate segmentation results. User interactions typically take less than 5 seconds. The location and size of the rectangular region do not need to be accurate as our method is insensitive to initialization. The time complexity of our algorithm is also reasonable since we only need to update the level set function within a narrow band of the zero level set.

There exist a few parameters in our method. In all our experiments, we use the same set of parameter settings. The size of a pixel neighborhood is always set to 9×9. In Algorithm 1, $\theta_e = 1.4$. In the level set speed (11), $\alpha = 0.5$, $\beta = 0.5$, and $\gamma = 0.25$ during a regular pass. However, during the refinement stage, $\alpha = 0.2$, $\beta = 0.8$, and $\gamma = 0.4$ because we would like to emphasize boundary localization and smoothness. In (9), we always set $\mu = 0.5$.

We have compared our method with three representative techniques, including geodesic active regions [11], Grabcut [3] and, lazy texture selection [26]. The first technique is based on the level set method while the other two are based on graph cuts [36], [7]. Visual comparison results can be found in Figs. 8, 9, and 10. It is evident that our method produces the best results. It accurately

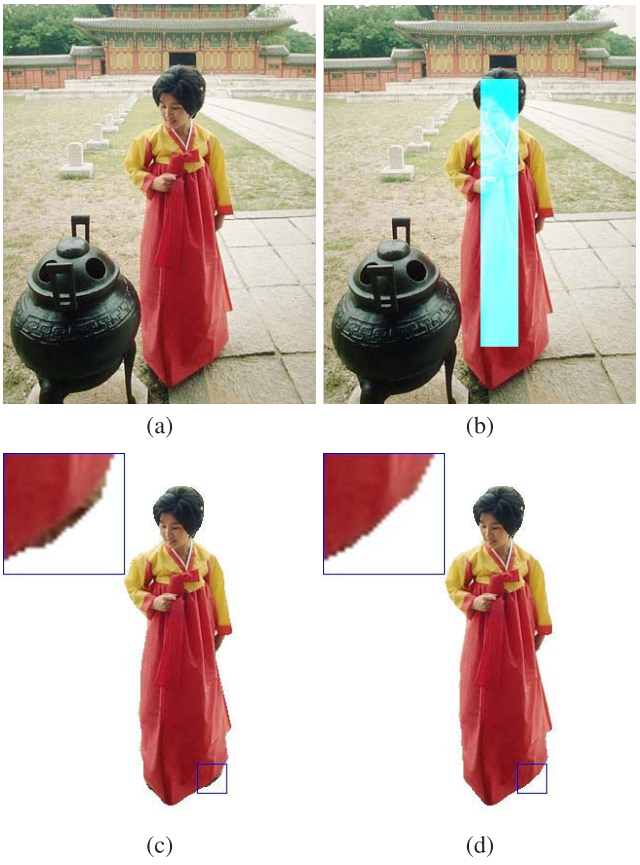

Fig. 6. Refinement. (a) Original image, (b) user-drawn rectangle, (c) result without boundary refinement (shadow pixels on the ground are misclassified as foreground pixels), and (d) final result after boundary refinement (second scheme).

segments out the fragmented leaves in Fig. 9 and our segmentations are more accurate along concave portions of the object boundaries in Figs. 8 and 10. Such results confirm the effectiveness of the edge field and the discriminative classifier employed in our method.

To provide a fair numerical comparison, we further compute the percentage of misclassifications for all methods using the ground truth data provided by the authors of [3] (also shown in Fig. 7). The numerical results shown in

Fig. 7. Ground truth data (first row shows original images, and the second row shows ground truth segmentation). (a) Cat, (b) Flowers, (c) Fish, and (d) Boy.

TABLE 1
A Comparison of the Percentage of Misclassified Pixels in Supervised Foreground/Background Segmentation among Four Methods

Image	Geodesic Active Regions	Grabcut	Lazy Texture Selection	Our method
Cat	2.45%	2.04%	0.76%	0.51%
Flowers	2.05%	0.89%	1.09%	0.67%
Fish	2.20%	1.50%	1.20%	0.63%
Boy	1.41%	0.16%	0.52%	0.13%

The original images and ground truth segmentation are shown in Fig. 7.

Table 1 clearly demonstrates that our method produces more numerically accurate segmentations than existing techniques.

7.1 Comparison with Geodesic Active Regions

Both our method and the method in [11] have a similar flavor. However, the method in [11] employs Gaussian mixture models as the statistical models discriminating foreground from the background. Gaussian mixture models are generative models while the probabilistic classifier in our method is a discriminative one. It is well known in machine learning that discriminative models in general can achieve better classification performance than generative models. The comparison shown in Fig. 2 confirms this conclusion. In addition, our method incorporates an edge field which permits distant interactions between salient edges and the zero level set. As a result, our method can achieve better foreground/background discrimination and more accurate boundary localization.

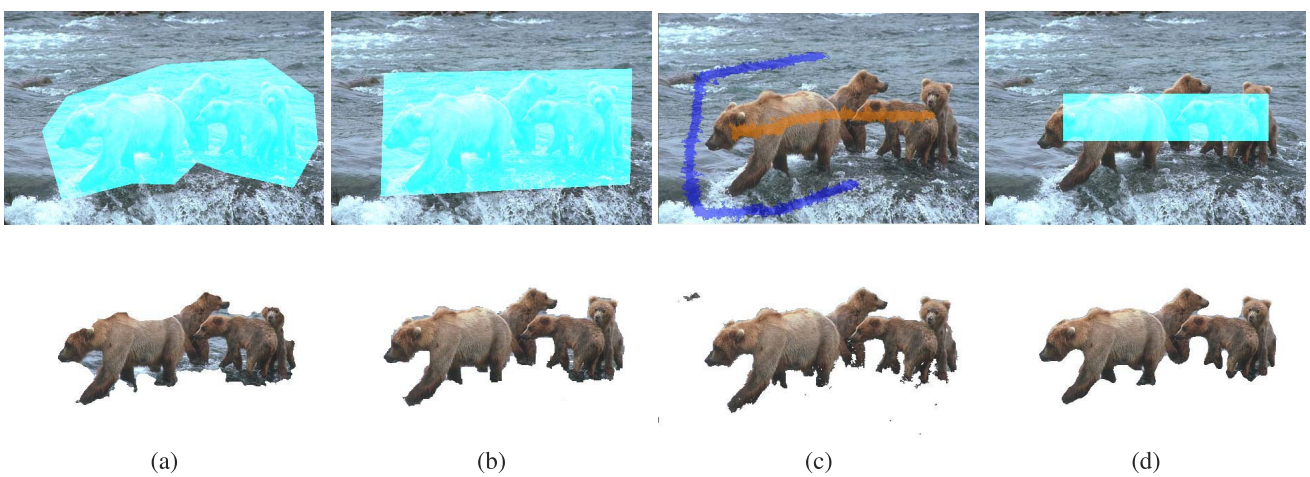

Fig. 8. First comparison. (a) Geodesic Active Regions, (b) Grabcut, (c) Lazy Texture Selection, and (d) our method.

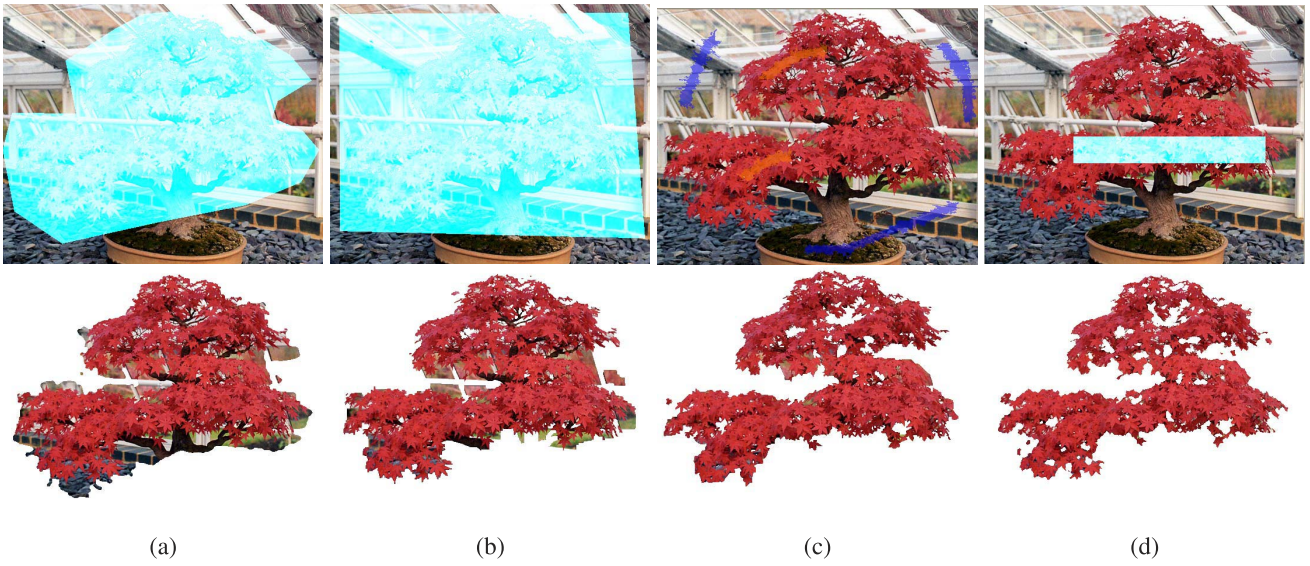

Fig. 9. Second comparison. (a) Geodesic Active Regions, (b) Grabcut, (c) Lazy Texture Selection, and (d) our method.

7.2 Comparison with Grabcut

Our multipass method for achieving progressively better results is inspired by Grabcut [3]. However, Grabcut as well as other related techniques, such as [2], still make use of Gaussian mixture models. Therefore, they have a similar discriminative power as [11]. From its segmentation results, it can be observed that Grabcut is not very effective for a foreground layer with a complex topology and fragmented appearances, such as the leaves in Fig. 9b, and misses the tail of the cat in Fig. 10b. Another limitation of GrabCut is that the user-drawn rectangle needs to completely encloses the foreground object, while our method only requires that the rectangular regions partially cover the foreground object. Note that the method in [6] can produce better results than Grabcut, but still share similar limitations with Grabcut.

7.3 Comparison with Lazy Texture Selection

Both our method and the method in [26] adopt discriminative models for pixel classification. Nevertheless, our method requires less user interaction but achieves better results. The method in [26] requires user scribbles over both image foreground and background, and it requires additional user inputs during the query-by-boosting step. Its results are also relatively more sensitive to the location of user scribbles. If the initial scribbles missed certain textures in the foreground, it would be hard to correct this later on. The results from lazy texture selection also exhibit less boundary smoothness and spatial coherence.

8 Conclusions

We have presented a robust and accurate level set method for interactive image segmentation. The level set method is advantageous for image objects with a complex topology and fragmented appearance. Our method integrates statistical classification and distance transforms to avoid local minima and better snap to true object boundaries. We have further proposed a computational framework that improves the performance of both pixelwise classification and the level set method over multiple passes. Most of the running time is spent on the computation of pixelwise likelihood using the probabilistic classifier. Since such computation is performed independently over every pixel, the overall performance of

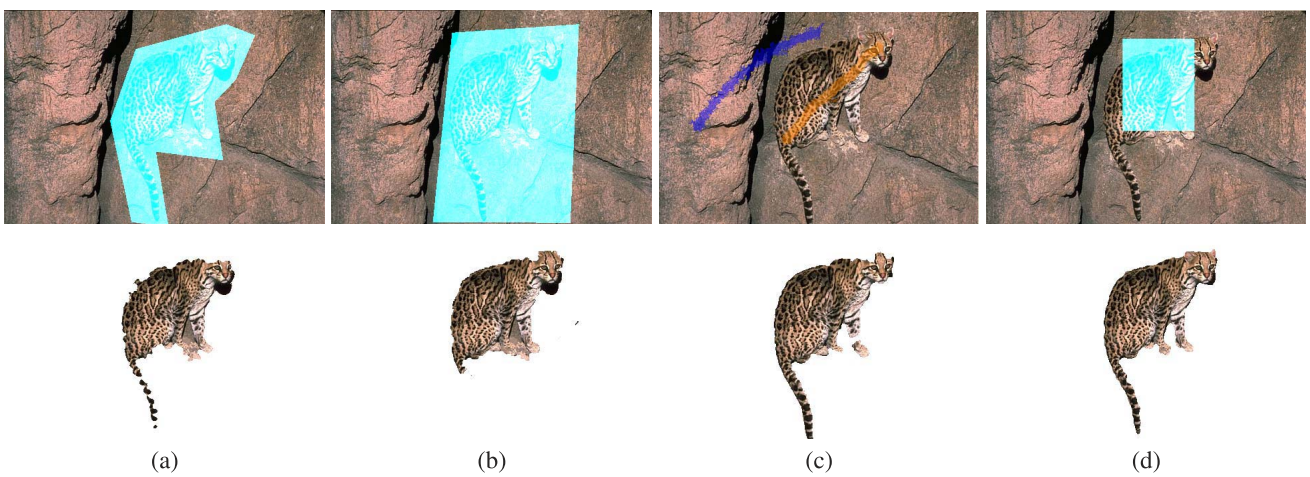

Fig. 10. Third comparison. (a) Geodesic Active Regions, (b) Grabcut, (c) Lazy Texture Selection, and (d) our method.

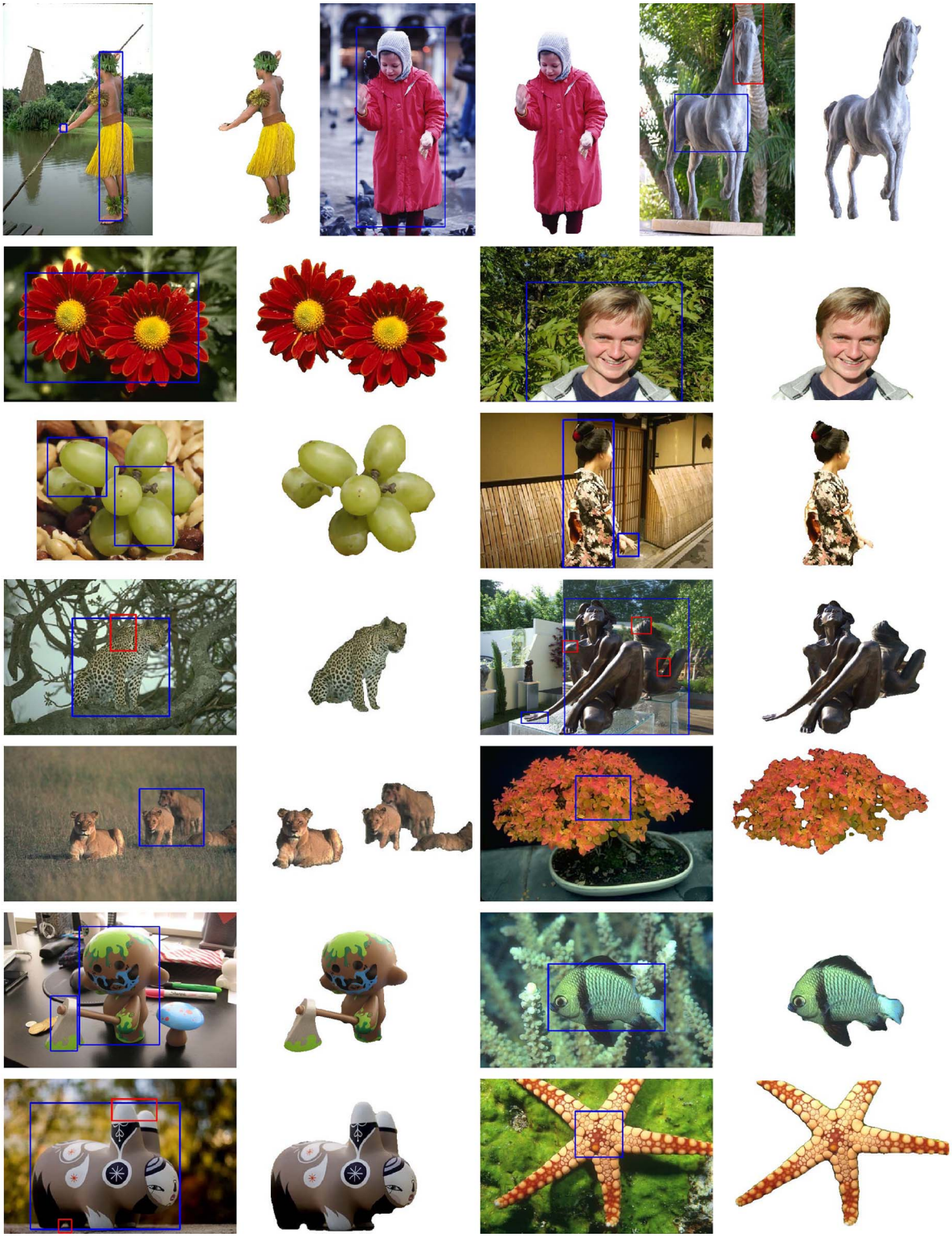

Fig. 11. A gallery of segmentation results. Blue rectangles are user inputs during initialization, red rectangles are user inputs during refinement.

our method can be significantly improved by parallelization on multicore CPUs. Experiments and comparisons have demonstrated the effectiveness of our method.

Acknowledgments

We would like to thank the reviewers, whose comments have been valuable in improving our manuscript. This work was partially supported by US National Science Foundation (NSF) (IIS 09-14631).

References

[1] E. Mortensen and W. Barrett, "Intelligent Scissors for Image Composition," *Proc. SIGGRAPH '95* , pp. 191-198, 1995.
[2] Y. Li, J. Sun, C.-K. Tang, and H.-Y. Shum, "Lazy Snapping," *ACM Trans. Graphics*, vol. 23, no. 3, pp. 303-308, 2004.

[3] C. Rother, A. Blake, and V. Kolmogorov, "Grabcut-Interactive Foreground Extraction Using Iterated Graph Cuts," *ACM Trans. Graphics,* vol. 23, no. 3, pp. 309-314, 2004.
[4] A. Protiere and G. Sapiro, "Interactive Image Segmentation via Adaptive Weighted Distances," *IEEE Trans. Image Processing,* vol. 16, no. 4, pp. 1046-1057, Apr. 2007.
[5] J. Liu, J. Sun, and H.-Y. Shum, "Paint Selection," *ACM Trans. Graphics,* vol. 28, no. 3, pp. 1-7, 2009.
[6] V. Lempitsky, P. Kohli, C. Rother, and T. Sharp, "Image Segmentation with a Bounding Box Prior," *Proc. Int'l Conf. Computer Vision,* 2009.
[7] Y. Boykov and M.-P. Jolly, "Interactive Graph Cuts for Optimal Boundary and Region Segmentation of Objects in n-d Images," *Proc. Int'l Conf. Computer Vision,* pp. 105-112, 2001.
[8] S. Osher and J. Sethian, "Fronts Propagating with Curvature Dependent Speed: Algorithms Based on Hamilton-Jacobi Formulations," *Computational Physics,* vol. 79, pp. 12-49, 1988.
[9] J. Sethian, *Level Set Methods and Fast Marching Methods.* Cambridge Univ. Press, 1999.
[10] S.J. Osher and R.P. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces,* first ed. Springer-Verlag, 2003.
[11] N. Paragios and R. Deriche, "Geodesic Active Regions for Supervised Texture Segmentation," *Proc. Int'l Conf. Computer Vision,* pp. 926-932, 1999.
[12] T.F. Chan and L.A. Vese, "Active Contours without Edges," *IEEE Trans. Image Processing,* vol. 10, no. 2, pp. 266-277, Feb. 2001.
[13] G. Aubert, M. Barlaudi, O. Faugeras, S. Jehan-Besson, and Stephanie, "Image Segmentation Using Active Contours: Calculus of Variations or Shape Gradients?," *SIAM J. Applied Math.,* vol. 63, no. 6, pp. 2128-2154, Aug. 2003.
[14] M.E. Leventon, W.E.L. Grimson, and O. Faugeras, "Statistical Shape Influence in Geodesic Active Contours," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 316-323, 2000.
[15] M.E. Leventon, W.E.L. Grimson, O. Faugeras, and W.M.W. III, "Level Set Based Segmentation with Intensity and Curvature Priors," *Proc. IEEE Workshop Math. Methods in Biomedical Image Analysis,* pp. 4-11, 2000.
[16] Y. Chen, H.D. Tagare, S. Thiruvenkadam, F. Huang, D. Wilson, K.S. Gopinath, R.W. Briggs, and E.A. Geiser, "Using Prior Shapes in Geometric Active Contours in a Variational Framework," *Int'l J. Computer Vision,* vol. 50, no. 3, pp. 315-328, Dec. 2002.
[17] M. Rousson and N. Paragios, "Shape Priors for Level Set Representations," *Proc. European Conf. Computer Vision,* pp. 78-92, 2002.
[18] X. Bresson, P. Vandergheynst, and J.-P. Thiran, "A Variational Model for Object Segmentation Using Boundary Information and Shape Prior Driven by the Mumford-Shah Functional," *Int'l J. Computer Vision,* vol. 68, no. 2, pp. 145-162, June 2006.
[19] C. Sagiv, N.A. Sochen, and Y.Y. Zeevi, "Integrated Active Contours for Texture Segmentation," *IEEE Trans. Image Processing,* vol. 15, no. 6, pp. 1633-1646, June 2006.
[20] N. Houhou, J.-P. Thiran, and X. Bresson, "Fast Texture Segmentation Based on Semi-Local Region Descriptor and Active Contour," *Numerical Math.: Theory, Methods and Applications,* vol. 2, no. 4, pp. 445-468, Nov. 2009.
[21] D. Martin, C. Fowlkes, and J. Malik, "Learning to Detect Natural Image Boundaries Using Brightness and Texture," *Proc. Neural Information Processing Systems(NIPS),* 2002.
[22] S. Avidan, "Spatialboost: Adding Spatial Reasoning to Adaboost," *Proc. European Conf. Computer Vision,* 2006.
[23] J. Shotton, J. Winn, C. Rother, and A. Criminisi, "Textonboost: Joint Appearance, Shape and Context Modeling for Multi-Class Object Recognition and Segmentation," *Proc. European Conf. Computer Vision,* 2006.
[24] Y. Li, E. Adelson, and A. Agarwala, "Scribbleboost: Adding Classification to Edge-Aware Interpolation of Local Image and Video Adjustments," *Computer Graphics Forum,* vol. 27, no. 4, pp. 1255-1264, 2008.
[25] B. Xue, W. Jue, S. David, and S. Guillermo, "Video Snapcut: Robust Video Object Cutout Using Localized Classifiers," *ACM Trans. Graphics,* vol. 28, no. 3, pp. 1-11, 2009.
[26] T. Xia, Q. Wu, C. Chen, and Y. Yu, "Lazy Texture Selection Based on Active Learning," *The Visual Computer,* vol. 26, no. 3, pp. 157-169, Mar. 2010.
[27] R.E. Schapire, "The Boosting Approach to Machine Learning: An Overview," *Proc. MSRI Workshop Nonlinear Estimation and Classification,* pp. 149-171, 2002.
[28] A. Torralba, K.P. Murphy, and W.T. Freeman, "Sharing Visual Features for Multiclass and Multiview Object Detection," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 29, no. 3, pp. 854-869, May 2007.
[29] L.G. Shapiro and G.C. Stockman, *Computer Vision,* first ed. Prentice Hall, 2001.
[30] J.F. Canny, "A Computational Approach to Edge Detection," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 8, no. 6, pp. 679-698, Nov. 1986.
[31] B. Manjunath and W. Ma, "Texture Features for Browsing and Retrieval of Image Data," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 18, no. 8, pp. 837-842, Aug. 1996.
[32] T. Leung and J. Malik, "Representing and Recognizing the Visual Appearance of Materials Using Three-Dimensional Textons," *Int'l J. Computer Vision,* vol. 43, no. 1, pp. 29-44, 2001.
[33] C. Novak and S. Shafer, "Anatomy of a Color Histogram," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 599-605, 1992.
[34] Z. Tu, "Probabilistic Boosting-Tree: Learning Discriminative Models for Classification, Recognition, and Clustering," *Proc. Int'l Conf. Computer Vision,* pp. 1589-1596, 2005.
[35] D. Peng, B. Merriman, S. Osher, H. Zhao, and M. Kang, "A PDE-Based Fast Local Level Set Method," *J. Computational Physics,* vol. 155, no. 2, pp. 410-438, 1999.
[36] Y. Boykov, O. Veksler, and R. Zabih, "Efficient Approximate Energy Minimization via Graph Cuts," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 20, no. 12, pp. 1222-1239, Nov. 2001.

Yugang Liu received the BS degree in computer science from Chongqing University of Post and Telecommunications, China, in 2005. He is currently working toward the PhD degree in computer science at the University of Electronic Science and Technology of China. He was a visiting student in the graphics lab at the University of Illinois at Urbana-Champaign from September 2008 to October 2010. His research interests include computer graphics, image processing, and computer vision.

Yizhou Yu received the BS degree in computer science and the MS degree in applied mathematics from Zhejiang University, China, in 1992 and 1994, respectively, and the PhD degree in computer science from the University of California at Berkeley in 2000. He is currently an associate professor in the Department of Computer Science at University of Illinois at Urbana-Champaign. He is a recipient of the best paper award at the 2005 and 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, 2002 National Science Foundation CAREER Award and 1998 Microsoft Graduate Fellowship. He is on the editorial board of the *Visual Computer Journal* and *International Journal of Software and Informatics*. His current research interests include computer animation, geometry processing, image processing, and visual analytics.