*Electronic Letters on Computer Vision and Image Analysis 6(3):13-28, 2007*

# SnakeCut: An Integrated Approach Based on Active Contour and GrabCut for Automatic Foreground Object Segmentation

Surya Prakash, R. Abhilash and Sukhendu Das

*Visualization and Perception Lab,*
*Department of Computer Science and Engineering,*
*Indian Institute of Technology Madras, Chennai-600 036, India.*

Received $28^{th}$ April 2007; Accepted $5^{th}$ November 2007

**Abstract**

Interactive techniques for extracting the foreground object from an image have been the interest of research in computer vision for a long time. This paper addresses the problem of an efficient, semi-interactive extraction of a foreground object from an image. Snake (also known as Active contour) and GrabCut are two popular techniques, extensively used for this task. Active contour is a deformable contour, which segments the object using boundary discontinuities by minimizing the energy function associated with the contour. GrabCut provides a convenient way to encode color features as segmentation cues to obtain foreground segmentation from local pixel similarities using modified iterated graph-cuts. This paper first presents a comparative study of these two segmentation techniques, and illustrates conditions under which either or both of them fail. We then propose a novel formulation for integrating these two complimentary techniques to obtain an automatic foreground object segmentation. We call our proposed integrated approach as "SnakeCut", which is based on a probabilistic framework. To validate our approach, we show results both on simulated and natural images.

*Key Words*: Snake, Active Contour, GrabCut, SnakeCut, Object segmentation, Distance transform.

## 1 Introduction

Interactive techniques for extracting the foreground object from an image have been the interest of research in computer vision for long time. Snake (Active contour) [9] and GrabCut [14] are two popular semi-automatic techniques, extensively used for foreground object segmentation. Active contour is a deformable contour, which segments the object using boundary discontinuities by minimizing the energy function associated with the contour. Deformation in contour is caused because of internal and external forces acting on it. Internal force is derived from the contour itself and external force is invoked from the image. The internal and external forces are defined so that the snake will conform to object boundary or other desired features within the image. Snakes are widely used in many applications such as segmentation [11, 16], shape modeling [17], edge detection

Correspondence to: <sdas@iitm.ac.in>

Recommended for acceptance by <U.Pal and P.Nagabhushan>
ELCVIA ISSN:1577-5097
Published by Computer Vision Center / Universitat Autònoma de Barcelona, Barcelona, Spain

[9], motion tracking [18] etc. Active contours can be classified as either *parametric active contours* [5, 9] or *geometric active contours* [3, 4], according to their representation and implementation. In this work, we focus on using parametric active contours, which synthesize parametric curves within the image domain and allow them to move towards the desired image features under the influence of internal and external forces. The internal force serves to impose piecewise continuity and smoothness constraint, whereas external force pushes the snake towards salient image features like edges, lines and subjective contours.

GrabCut [14] is an interactive tool based on iterative graph-cut for foreground object segmentation in still images. GrabCut provides a convenient way to encode color features as segmentation cues to obtain foreground segmentation from local pixel similarities and global color distribution using modified iterated graph-cuts. GrabCut extends graph-cut to color images and to incomplete trimaps. GrabCut has been applied in many applications for the foreground extraction [6, 8, 12].

Since Active Contour uses gradient information (boundary discontinuities) present in the image to estimate the object boundary, it can detect the object boundary efficiently but cannot penetrate inside the object boundary. It cannot remove any pixel present inside the object boundary which does not belong to a foreground object. Example of such case is the segmentation of an object with holes. On the other hand, GrabCut works on the basis of pixel color (intensity) distribution and so it has the ability to remove interior pixels which are not the part of the object. Major problem with the GrabCut is: if some part of the foreground object has color distribution similar to the image background, that part will also be removed in GrabCut segmentation. In the GrabCut algorithm [14], missing foreground data is recovered by user interaction. This paper first presents a comparative study of these two segmentation techniques. We then present a semi-automatic technique based on the integration of Active Contour and GrabCut which can produce correct segmentation in cases where both Snake and GrabCut fail. We call our technique as "SnakeCut", which is based on integrating the outputs of Snake and GrabCut using a probabilistic framework. In SnakeCut, user needs to only specify a rectangle (or polygon) enclosing the foreground object. No post corrective editing is required in our approach. Proposed technique is used to segment a single object from an image.

Rest of the paper is organized as follows. In section 2, we briefly present Active Contour and GrabCut techniques which provides the theoretical basis for the paper. Section 3 compares the two techniques and discusses the limitations of both. In section 4, we present the SnakeCut algorithm, our proposed segmentation technique for foreground object segmentation. Section 5 presents some results on simulated and natural images. We conclude the paper in section 6.

## 2 Preliminaries

### 2.1 Active Contour (Snake) Model

A traditional active contour is defined as a parametric curve $\mathbf{v}(s) = [x(s), y(s)]$, $s \in [0, 1]$, which minimizes the following energy functional

$$E_{snake} = \int_0^1 \frac{1}{2}(\eta_1 |\mathbf{v}^{'}(s)|^2 + \eta_2 |\mathbf{v}^{''}(s)|^2) + E_{ext}(\mathbf{v}(s))ds \tag{1}$$

where, $\eta_1$ and $\eta_2$ are weighting constants to control the relative importance of the elastic and bending ability of snake respectively. $\mathbf{v}^{'}(s)$ and $\mathbf{v}^{''}(s)$ are the first and second order derivatives of $\mathbf{v}(s)$, and $E_{ext}$ is derived from the image so that it takes smaller values at the feature of interest such as edges, object boundaries etc. For an image $I(x, y)$, where $(x, y)$ are spatial co-ordinates, typical external energy is defined as follows to lead snake towards step edges [9]:

$$E_{ext} = -|\nabla I(x, y)|^2 \tag{2}$$

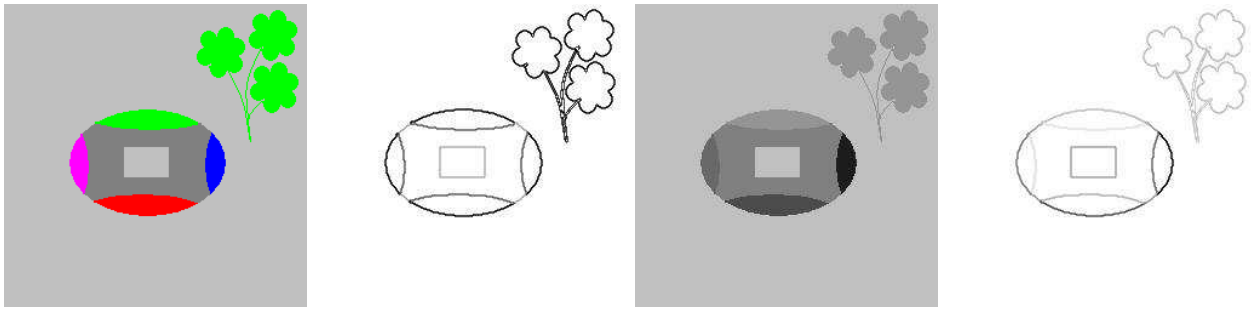
where, $\nabla$ is gradient operator. For color images, we estimate the intensity gradient which takes the maximum of the gradients of $R$, $G$ and $B$ bands at every pixel, using:

$$|\nabla I| = \max(|\nabla R|, |\nabla G|, |\nabla B|) \tag{3}$$

Figure 1(b) shows an example of intensity gradient estimation using the Eq. 3 for the image shown in Figure 1(a). Figure 1(d) shows the intensity gradient for the same input image estimated from its gray scale image (Figure 1(c)). The gradient obtained using Eq. 3 gives better edge information. A snake that minimizes $E_{snake}$ must satisfy the following Euler equation [7]

$$\eta_1 \mathbf{v}''(s) - \eta_2 \mathbf{v}''''(s) - \nabla E_{ext} = \mathbf{0} \tag{4}$$

where, $\mathbf{v}''(s)$ and $\mathbf{v}''''(s)$ are the second and fourth order derivatives of $\mathbf{v}(s)$. Eq. 4 can also be viewed as a force balancing equation, $\mathbf{F}_{int} + \mathbf{F}_{ext} = \mathbf{0}$ where, $\mathbf{F}_{int} = \eta_1 \mathbf{v}''(s) - \eta_2 \mathbf{v}''''(s)$ and $\mathbf{F}_{ext} = -\nabla E_{ext}$. $\mathbf{F}_{int}$, the internal force, is responsible for stretching and bending and $\mathbf{F}_{ext}$, the external force, attracts the snake towards the desired features in the image.



(a) (b) (c) (d)

Figure 1: Gradient in color and gray scale images: (a) Input image, (b) Gradient image of (a) estimated using Eq. 3, (c) Gray scale image of the input image (a), (d) Gradient image of (c).

To find the object boundary, Active Contour deforms so it can be represented as a time varying curve $\mathbf{v}(s,t) = [\mathbf{x}(s,t), \mathbf{y}(s,t)]$ where $s \in [0,1]$ is arc length and $t \in R^+$ is time. Dynamics of the contour in presence of external and internal forces can be governed by the following equation

$$\xi \mathbf{v}_t = \mathbf{F}_{int} + \mathbf{F}_{ext} \tag{5}$$

where, $\mathbf{v}_t$ is the partial derivative of $\mathbf{v}$ $w.r.t.$ $t$ and $\xi$ being an arbitrary non-negative constant. The contour comes to rest when the net effect of the internal and external forces reaches zero, which eventually happens when deforming contour reaches the object boundary.

## 2.2 GrabCut

GrabCut [14] is an interactive tool based on iterative graph-cut for foreground extraction in still images. To segment a foreground object using GrabCut, user has to select an area of interest (AOI) with a rectangle to obtain the desired result. GrabCut extends the graph-cut based segmentation technique, introduced by Boykov and Jolly [1], using color information. In this section, we briefly discuss about the GrabCut. For more details readers are advised to see [14].

Consider image $I$ as an array $\mathbf{z} = (z_1, ..., z_n, ..., z_N)$ of pixels, indexed by the single index n, where $z_n$ is in RGB space. Segmentation of the image is expressed as an array of "opacity" values $\underline{\alpha} = (\alpha_1, ..., \alpha_n, ..., \alpha_N)$ at each pixel. Generally $0 \leq \alpha_n \leq 1$, but for hard segmentation, $\alpha_n \in \{0, 1\}$ with 0 for background and 1 for foreground. For the purpose of segmentation, GrabCut constructs two separate Gaussian mixture models (GMMs) to express the color distributions for the background and foreground. Each GMM, one for foreground and one for background, is taken to be a full-covariance Gaussian mixture with K components. In order to deal

with the GMM tractability in an optimization framework, an additional vector $\mathbf{k} = (k_1, ..., k_n, ..., k_N)$ is taken, with $k_n \in \{1, ..., K\}$, assigning to each pixel a unique GMM component, which is either from the foreground or background according to $\alpha_n = 0$ or 1.

GrabCut defines an energy function $\mathbf{E}$ such that its minimum should correspond to a good segmentation, in the sense that it is guided both by the observed foreground and background GMMs and that the opacity is "coherent". This is captured by "Gibbs" energy in the following form:

$$\mathbf{E}(\underline{\alpha}, \mathbf{k}, \underline{\theta}, \mathbf{z}) = U(\underline{\alpha}, \mathbf{k}, \underline{\theta}, \mathbf{z}) + V(\underline{\alpha}, \mathbf{z}) \tag{6}$$

The data term $U$ evaluates the fit of the opacity distribution $\underline{\alpha}$ to the data $\mathbf{z}$. It takes into account the color GMM models, defined as

$$U(\underline{\alpha}, \mathbf{k}, \underline{\theta}, \mathbf{z}) = \sum_n D(\alpha_n, k_n, \underline{\theta}, z_n) \tag{7}$$

where,

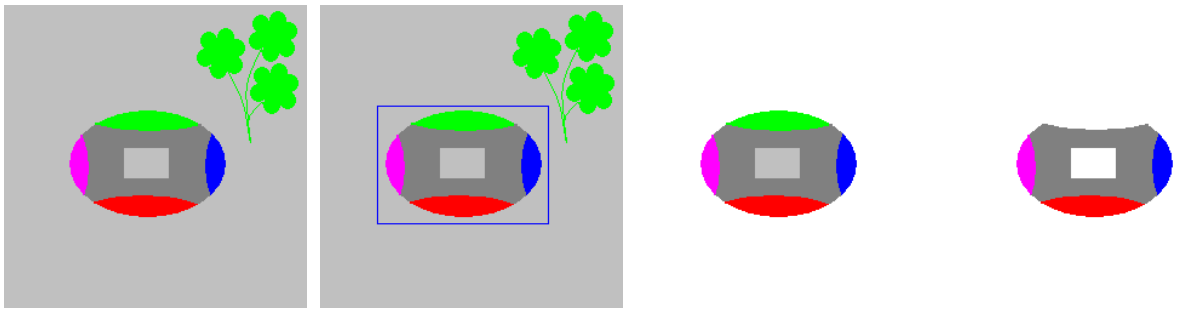
$$D(\alpha_n, k_n, \underline{\theta}, z_n) = -\log p(z_n|\alpha_n, k_n, \theta_n) - \log \pi(\alpha_n, k_n) \tag{8}$$

Here, $p(.)$ is a Gaussian probability distribution, and $\pi(.)$ are mixture weighting coefficients. Therefore, the parameters of the model are now $\underline{\theta} = \{\pi(\alpha, k), \mu(\alpha, k), \Sigma(\alpha, k); \alpha = 0, 1; k = 1..K\}$, where $\pi$, $\mu$ and $\Sigma$'s represent the weights, means and covariances of the $2K$ Gaussian components for the background and the foreground distributions. In Equation 6, the term $V$ is called the smoothness term and is given as follows:

$$V(\underline{\alpha}, \mathbf{z}) = \gamma \sum_{(m,n)\in R} \frac{1}{dist(m,n)} [\alpha_n \neq \alpha_m] exp(-\beta(\|z_m - z_n\|^2)) \tag{9}$$

where, $[\phi]$ denotes the indicator function taking values 0, 1 for a predicate $\phi$, $\gamma$ is a constant, $R$ is the set of neighboring pixels, and $dist(.)$ is the Euclidian distance of neighboring pixels. This energy encourages coherence in the regions of similar color distribution.

Once the energy model is defined, segmentation can be estimated as a global minimum: $\widehat{\underline{\alpha}} = \arg\min_{\underline{\alpha}} \mathbf{E}(\underline{\alpha}, \underline{\theta})$. Energy minimization in GrabCut is done by using standard minimum cut algorithm [1]. Minimization follows an iterative procedure that alternates between estimation and parameter learning.



(a) (b) (c) (d)

Figure 2: (a) Input image, elliptical object present in the image contains a rectangular hole at the center, (b) foreground initialization by user, (c) Active Contour segmentation result and (d) GrabCut segmentation result.

## 3 Comparison of Active Contour and GrabCut Methods

Active contour relies on the presence of intensity gradient (boundary discontinuities) in the image. So it is a good tool for the estimation of the object boundaries. But, since it cannot penetrate inside the object boundary, it is not able to remove the undesired parts, say holes, present inside the object boundary. If an object has a hole in it, Active Contour will detect the hole as a part of the object. Figure 2(c) shows one such segmentation example

of Active Contour for a synthetic image shown in Figure 2(a). Input image (Figure 2(a)) contains a foreground object with rectangular hole at the center, through which gray color background is visible. Segmentation result for this image (shown in Figure 2(c)), contains the hole included as a part of the detected object which is incorrect. Since Snake could not go inside, it has converted the outer background into white but retained the hole as gray. Similar erroneous segmentation result of Active Contour for a real image (shown in Figure 3(a)) is shown in Figure 3(b). One can see that segmentation output contains a part of the background region (e.g. grass patch between legs) along with the foreground object. Figure 4(b) shows one more erroneous Active Contour segmentation result for the image shown in Figure 4(a). Segmentation output contains some pixels in the interior part of the foreground object from the background texture region.

(a) (b) (c)

Figure 3: (a) Soldier Image, (b) segmentation result of Active Contour, (c) segmentation result of GrabCut.


(a) (b) (c)

Figure 4: (a) Image containing wheel, (b) segmentation result of Active Contour, (c) segmentation result of GrabCut.

On the other hand, GrabCut considers global color distribution (with local pixels similarities) of the background and foreground pixels for segmentation. So it has the ability to remove interior pixels which are not a part of object. To segment the object using GrabCut, user draws a rectangle enclosing the foreground object. Pixels outside the rectangle are considered as background pixel and pixels inside the rectangle are considered as unknown. GrabCut estimates the color distribution for the background and the unknown region using separate GMMs. Then, it iteratively removes the pixels from the unknown region which belong to background. Major problem with the GrabCut is as follows. If some part of the object has color distribution similar to the image background then that part of foreground object is also removed in the GrabCut segmentation output. So

GrabCut is not intelligent enough to distinguish between the desired and unnecessary pixels, while eliminating some of the pixels from the unknown region. Figure 2(d) shows one such segmentation result of GrabCut for the image shown in Figure 2(a), where the objective is to segment the object with a hole present in the image. Segmentation result does not produce the upper part of the object (shown in Green color in Figure 2(a)) near the boundary. This occurs because, in the original input image (Figure 2(a)), a few pixels with Green color were present as a part of the background region. Figure 3(c) presents a GrabCut segmentation result for a real world image shown in Figure 3(a). The objective in this case is to crop the soldier from the input image. GrabCut segmentation result for this input image does not produce the soldier's hat and the legs. In another real world image example in Figure 4(a), where the user targets to crop the wheel present in the image, GrabCut segmentation output (Figure 4(c)) does not produce the wheel's grayish green rubber part. This happened because of the presence of some objects with similar color in the background.

In GrabCut [14] algorithm, missing data of the foreground object is often recovered by user interaction. User has to mark the missing object parts as compulsory foreground. We present in this paper, an automatic foreground object segmentation technique based on the integration of Active Contour and GrabCut, which can produce accurate segmentation in situations where both or either of these techniques fail. We call our proposed technique as "SnakeCut". We present it in the next section.

## 4 SnakeCut: Integration of Active Contour and GrabCut

Active Contour works on the principle of intensity gradient, where the user initializes a contour around or inside the object for it to detect the boundary of the object easily. GrabCut, on the other hand, works on the basis of the pixel's color distribution and considers global cues for segmentation. Hence it can easily remove the unwanted part (parts from the background) present inside the object boundary. These two segmentation techniques use complementary information (edge and region based) for segmentation. In SnakeCut, we combine these complementary techniques and present an integrated method for superior object segmentation. Figure 5 presents the overall flow chart of our proposed segmentation technique. In SnakeCut, input image is segmented using the Active Contour and GrabCut separately. These two segmentation results are provided to the probabilistic framework of SnakeCut. This integrates the two segmentation results based on a probabilistic criterion and produces the final segmentation result.

Main steps of the SnakeCut algorithm are provided in Algorithm 1. The probabilistic framework used to integrate the two outputs is as follows. Inside the object boundary $C_0$ (detected by the Active Contour), every pixel $z_i$ is assigned two probabilities: $P_c(z_i)$ and $P_s(z_i)$. $P_c(z_i)$ provides information about the pixel's nearness to the boundary, and $P_s(z_i)$ indicates how similar the pixel is to the background. Large value of $P_c(z_i)$ indicates that pixel $z_i$ is far from the boundary and a large value of $P_s(z_i)$ specifies that the pixel is more similar to the background. To take the decision about a pixel belonging to foreground or background, we evaluate a decision function $p$ as follows:

$$p(z_i) = \rho P_c(z_i) + (1 - \rho) P_s(z_i) \tag{10}$$

where, $\rho$ is the weight which controls the relative importance of the two techniques, and is learnt empirically. Probability $P_c$ is computed from the distance transform (DT) [2] of the object boundary $C_0$. DT has been used in many computer vision applications [13, 19, 15, 10]. It is given by the following equation:

$$I_d(z_i) = \begin{cases} 0, & \text{if } z_i \text{ lies on contour } C_0 \\ d, & \text{otherwise} \end{cases} \tag{11}$$

where, $d$ is the Euclidian distance of pixel $z_i$ to the nearest contour point. Figure 6(b) shows an example of DT image for the contour image shown in Figure 6(a). Distance transform values are first normalized in the range $[0, 1]$, before they are used for the estimation of $P_c$. Let, $I_n$ be the normalized distance transform image of $I_d$ and $d_n$ be the DT value of a pixel $z_i$ in $I_n$ (i.e. $d_n = I_n(z_i)$). Probability $P_c$ of $z_i$ is estimated using the

following fuzzy distribution function:

$$P_c(z_i) = \begin{cases} 0, & 0 \leq d_n < a; \\ 2\left(\frac{d_n - a}{b-a}\right)^2, & a \leq d_n < \frac{a+b}{2}; \\ 1 - 2\left(\frac{b-d_n}{b-a}\right)^2, & \frac{a+b}{2} \leq d_n < b; \\ 1, & b \leq d_n \leq 1. \end{cases} \tag{12}$$

where, $a$ and $b$ are constants and $a < b$. When $a \geq b$ this becomes a step function with transition at $(a+b)/2$ from 0 to 1. Probability distribution function (Eq. 12) has been chosen in such way that the probability value $P_c$ is small near the contour $C_0$ and large for points farther away. In this fuzzy function, $a$ and $b$ dictate the non-linear behavior of the function. The parameters $a$ and $b$ control the extents (distance from the boundary) to which the output response is considered from Snake and then onwards from that of GrabCut respectively. The extent of the points considered near the contour can be suitably controlled by choosing appropriate values of $a$ and $b$. The value of $P_c$ is zero (0) when the distance of the pixel from the boundary is in the range $[0..a]$, and one (1) in the range $[b..1]$ (all values normalized). For the values between $[a..b]$, we empirically found the smooth, non-linear S-shaped function to provide the best result. Figure 7 shows the effect of the interval $[a, b]$ on the distribution function.

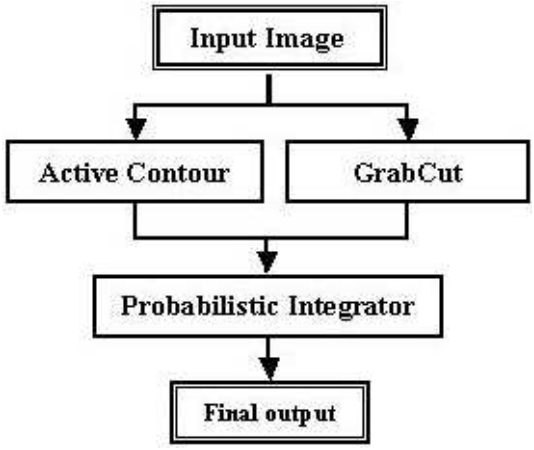
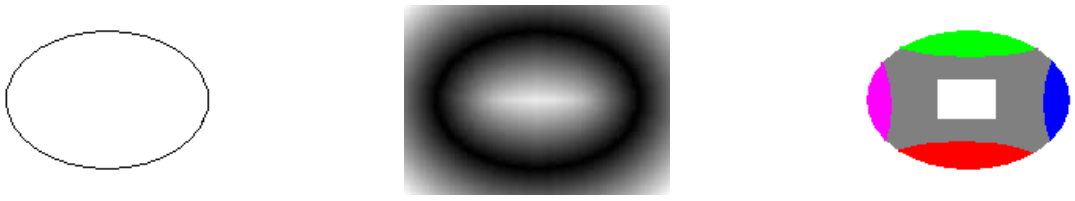



Figure 5: Flow chart of proposed SnakeCut technique.



(a) (b) (c)

Figure 6: Segmentation of image shown in Figure 2(a) using SnakeCut: (a) object boundary produced by Active Contour, (b) distance transform for the boundary contour shown in (a); (c) SnakeCut segmentation result.

Probability value $P_s$ is obtained from the GrabCut segmentation process. GrabCut assigns likelihood values to each pixel in the image using the GMMs constructed for the foreground and background, which represent how likely a pixel belongs to the foreground or background. In our approach, after the segmentation of the object using GrabCut, the final background GMMs are used to estimate $P_s$. For each pixel $z_i$ inside $C_0$, $D(z_i)$

is computed using Eq. 8 considering background GMMs. Normalized values of $D$ between $0$ and $1$, for all the pixels inside $C_0$, define the probability $P_s$.

Using the decision function $p(z_i)$ estimated in Eq. 10 and an empirically estimated threshold $T$, GrabCut and Active Contour results are integrated using the SnakeCut algorithm (refer Algorithm 1). In the integration process of the SnakeCut algorithm, segmentation output for a pixel is taken from the GrabCut result if $p > T$, otherwise it is taken from the Active Contour result. In our experiments, we empirically found $\rho = 0.5$ to give the best result, and $T = 0.7$, $a = 0.15$ and $b = 0.2$.

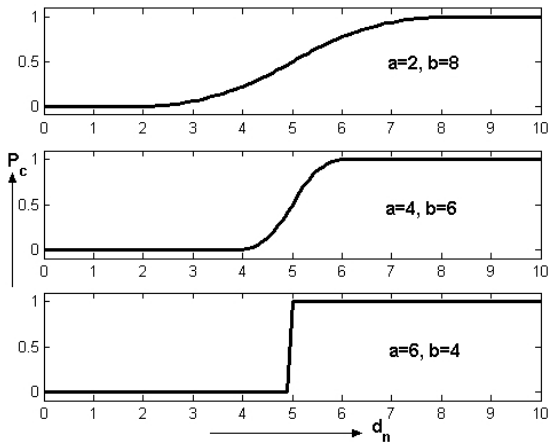



Figure 7: Effect of interval $[a, b]$ on the non-linearity of the fuzzy distribution function (Eq. 12). When $a < b$, transition from $0$ (at $a$) to $1$ (at $b$) is smooth. When $a \geq b$, we have a step function with the transition at $(a + b)/2$.

We demonstrate the integrated approach to the process of foreground segmentation with the help of a simulated example. Figure 6 shows the details of the SnakeCut technique for the segmentation of a foreground object present in the simulated image shown in Figure 2(a). Intermediate segmentation outputs produced by Active Contour and GrabCut for this image have been shown in Figures 2(c) & 2(d). These outputs are integrated by the SnakeCut algorithm. Figure 6(a) shows the object boundary obtained by Active Contour for the object shown in Figure 2(a). Active Contour boundary is used to estimate the distance transform, shown in Figure 6(b), using Eq. 11. Probability values $P_c$ and $P_s$ are estimated for all the pixels inside the object boundary obtained by Active Contour as described above. SnakeCut algorithm is then used to integrate the outputs of Active Contour and GrabCut. Figure 6(c) shows the segmentation result of SnakeCut after integration of intermediate outputs (Figure 2(c) & 2(d)) obtained using Active Contour and GrabCut algorithms. Our proposed method is able to retain a part of the object which appears similar to background color and simultaneously eliminate the hole within the object.

To demonstrate the impact of the probability values $P_c$ and $P_s$, and its impact on the decision making in SnakeCut algorithm, we use the soldier image (Figure 3(a)). We compute $P_c$, $P_s$ and $p$ values for a few points marked in the soldier image (Figure 8(a)) and then use SnakeCut algorithm to obtain the final segmentation decision. Values obtained for $P_c$, $P_s$ and $p$ are shown in Figure 8(b). Last column of the table shows the final decision taken by SnakeCut based on the estimated value of $p$.

## 5 SnakeCut Segmentation Results

To extract a foreground object using SnakeCut, user draws a rectangle (or polygon) surrounding the object. This rectangle is used in the segmentation process of Active Contour as well as GrabCut. Active Contour considers the rectangle as an initial contour and deforms it to converge on the object boundary. GrabCut uses the rectangle

**Algorithm 1** Steps of SnakeCut

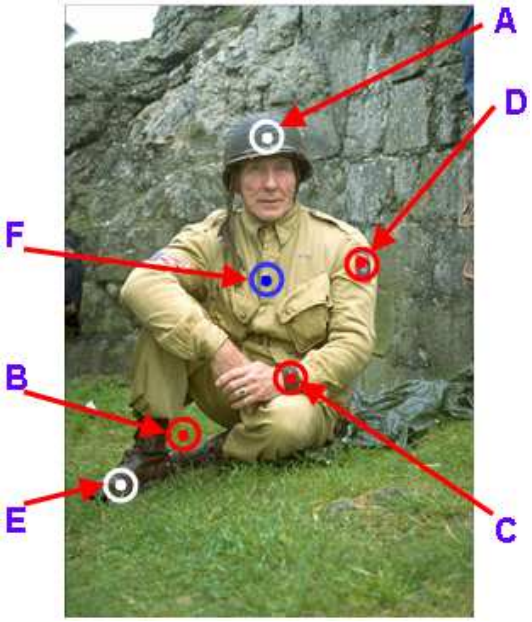
- Input $I$ and output $I_{sc}$.
- All pixels of $I_{sc}$ are initialized to zero.

**A. Initial Segmentation**

1. Segment desired object in $I$ using Active Contour. Say, object boundary identified by the Active Contour is $C_0$ and segmentation output of Active Contour is $I_{ac}$.
2. Segment desired object in $I$ using GrabCut. Say, segmentation output is $I_{gc}$.

**B. Integration using SnakeCut**

1. Find set of pixels $Z$ in image $I$, which lie inside contour $C_0$.
2. For each pixel $z_i \in Z$,
   (a) Compute $p(z_i)$ using Eq. 10.
   (b) **if** $p(z_i) \leq T$ **then**
   $I_{sc}(z_i) = I_{ac}(z_i)$
   **else**
   $I_{sc}(z_i) = I_{gc}(z_i)$
   **end if**

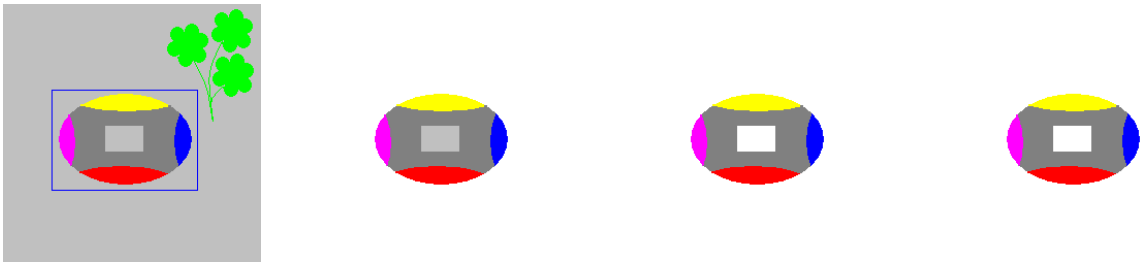



(a)

| Point | $Pc$ | $Ps$ | $p$ | Output taken from |
|---|---|---|---|---|
| **A** | **0.0026** | **0.6827** | **0.3426** | **Snake** |
| **B** | **1.0000** | **0.6601** | **0.8300** | **GrabCut** |
| **C** | **1.0000** | **0.6366** | **0.8183** | **GrabCut** |
| **D** | **0.0000** | **0.7300** | **0.3650** | **Snake** |
| **E** | **0.0000** | **0.5840** | **0.2922** | **Snake** |
| **F** | **1.0000** | **0.0000** | **0.5000** | **Snake** |

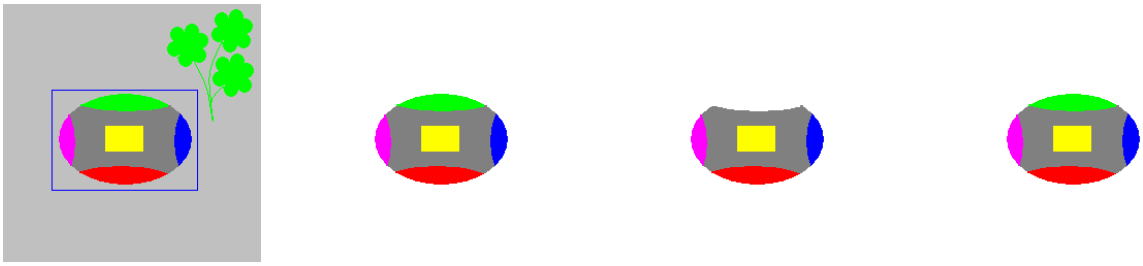
(b)

Figure 8: Demonstration of the impact of $P_c$ and $P_s$ values on the decision making in Algorithm 1: (a) soldier image with a few points marked on it, (b) values of $P_c$, $P_s$ and p, and the decision obtained using Algorithm 1. Values used for $\rho$ and $T$ are $0.5$ and $0.7$ respectively.



(a) (b) (c) (d)

Figure 9: Demonstration of a SnakeCut result on a synthetic image, where Snake fails and GrabCut works: (a) input image with foreground initialized by the user (object contains a rectangular hole at the center), (b) Snake segmentation result (incorrect, output contains the hole as a part of the object), (c) GrabCut segmentation result (correct, hole is removed), and (d) SnakeCut segmentation result (correct, hole is removed).



(a) (b) (c) (d)

Figure 10: Demonstration of a SnakeCut result on a synthetic image, where Snake works and GrabCut fails: (a) input image with foreground initialized by the user, (b) Snake segmentation result (correct), (c) GrabCut segmentation result (incorrect, upper green part of the object is removed), and (d) correct segmentation result produced by SnakeCut.

(a) (b) (c) (d)

Figure 11: Segmentation of real pot image: (a) input real image, (b) Active Contour segmentation result (incorrect), (c) GrabCut segmentation result (correct), and (d) SnakeCut segmentation result (correct, background pixels visible through the handles of the pot are removed).

(a) (b)

Figure 12: SnakeCut segmentation results of (a) soldier (for image in Figure 3(a)); and (b) wheel (for image in Figure 4(a)).

to define the background and unknown regions. Pixels outside the rectangle are taken as known background and those inside as unknown. GrabCut algorithm (using GMM based modeling and minimal cost graph-cut) iterates and converges to a minimum energy level producing the final segmentation output. Segmentation outputs of Active Contour and GrabCut are integrated using SnakeCut algorithm to obtain the final segmentation result. First, we present a few results of segmentation using SnakeCut on synthetic and natural images, where either Snake or GrabCut fails to work. This is followed by a few examples where both Snake and GrabCut techniques fail to perform correct segmentation, whereas integration of the outputs of these techniques using SnakeCut algorithm gives correct segmentation results.

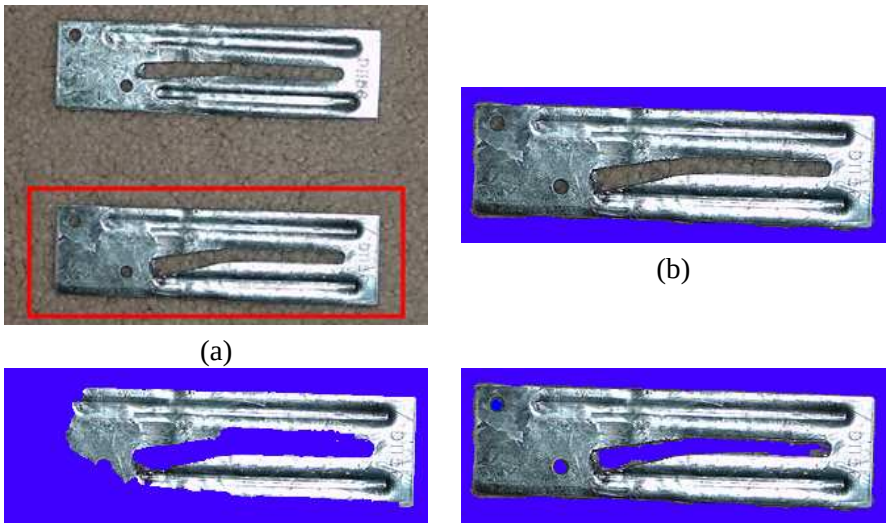
Figure 9 shows a result on a synthetic image where Active Contour fails but GrabCut works, and their integration ($i.e.$ SnakeCut) also produces the correct segmentation. Figure 9(a) shows an image where the object to be segmented has a rectangular hole (at the center) in it through which gray background is visible. Segmentation result produced by Active Contour (Figure 9(b)) shows the hole as a part of the segmented object which is incorrect. In this case, GrabCut performs correct segmentation (Figure 9(c)) of the object. Figure 9(d) shows the correct segmentation result produced by SnakeCut for this image. Figure 10 shows a result on another synthetic image where Active Contour works but GrabCut fails, and their integration ($i.e.$ SnakeCut) produces the correct segmentation. Figure 10(a) shows an image where the object to be segmented has a part (upper green region) similar to the background (green flowers). Active contour, in this example, produces correct segmentation (Figure 10(b)) while GrabCut fails (Figure 10(c)). Figure 10(d) shows the correct segmentation result produced by SnakeCut for this image. Figure 11 presents a SnakeCut segmentation result on a real image. In this example, Active Contour fails but GrabCut performs correct segmentation. We see in Figure 11(b) that Active Contour segmentation result contains the part of the background (visible through the handles) which is incorrect. SnakeCut algorithm produces correct segmentation result which is shown in Figure 11(d).

In the examples presented so far, we have seen that only one among the two (Snake and GrabCut) techniques fail to perform correct segmentation. In these examples, either Snake is unable to remove holes from the foreground object or GrabCut is unable to retain the parts of the object which are similar to the background. SnakeCut performs well in all such situations. We now present a few results on synthetic and real images, where SnakeCut performs well even when both Snake and GrabCut techniques fail to perform correct segmentation. Figure 12 presents two such SnakeCut results on real world images. Figure 12(a) shows the segmentation result produced by SnakeCut for the soldier image shown in Figure 3(a). This result is obtained by integrating the Active Contour and GrabCut outputs shown in Figures 3(b) and 3(c), without user interaction. Figure 12(b) shows the wheel segmentation result produced by SnakeCut, for the image shown in Figure 4(a). Intermediate Active Contour and GrabCut segmentation results for the wheel are shown in Figure 4(b) and 4(c).

Two more SnakeCut segmentation results are presented in Figures 13 and 14 for cup and webcam bracket images, where both Snake and GrabCut techniques fail to perform correct segmentation. The objective in the cup example (Figure 13(a)) is to segment the cup in the image. Cup's handle has some blue color spots similar

(a) (b) (c) (d)

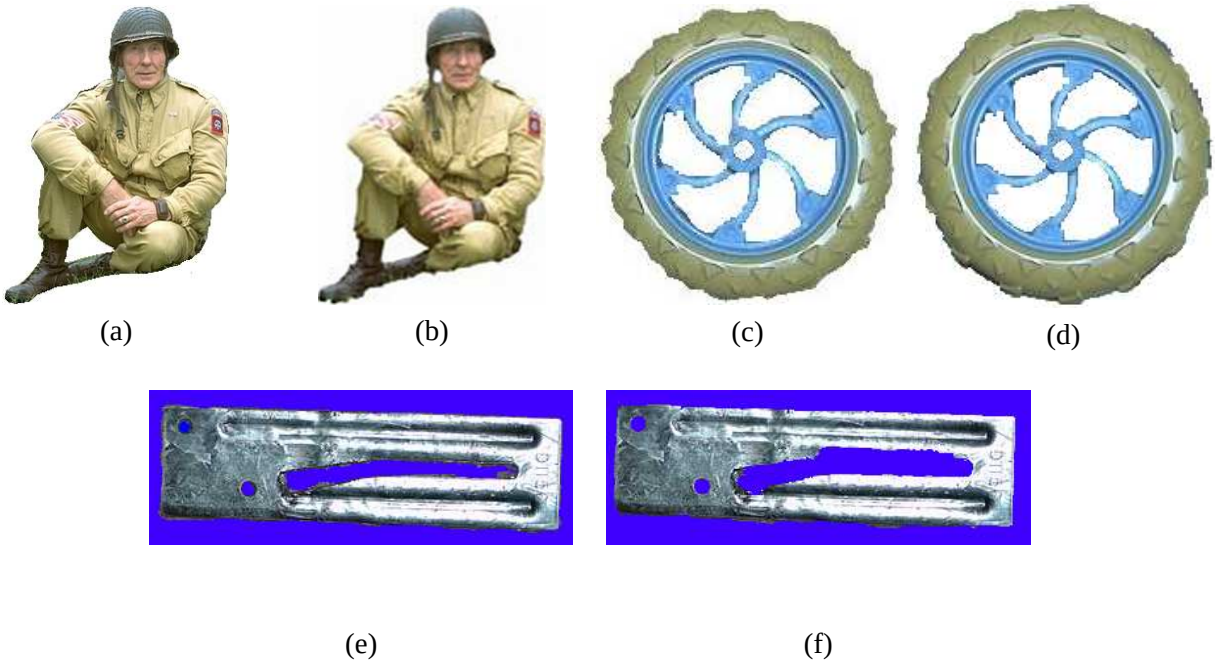
Figure 13: Segmentation of cup image: (a) input real image, (b) segmentation result produced by Snake (incorrect, as background pixels visible through the cup's handle are detected as a part of the object), (c) GrabCut segmentation result (incorrect, as spots present on the cup's handle are removed), and (d) correct segmentation result produced by SnakeCut.



(a) (b)

(c) (d)

Figure 14: Segmentation of webcam bracket: (a) input real image where the objective is to segment the lower bracket present in the image, (b) Snake segmentation result (incorrect, as background pixels visible through the holes present in the object are detected as part of the foreground object), (c) GrabCut segmentation result (incorrect, as large portions of the bracket are removed in the result), and (d) correct segmentation result produced by SnakeCut.



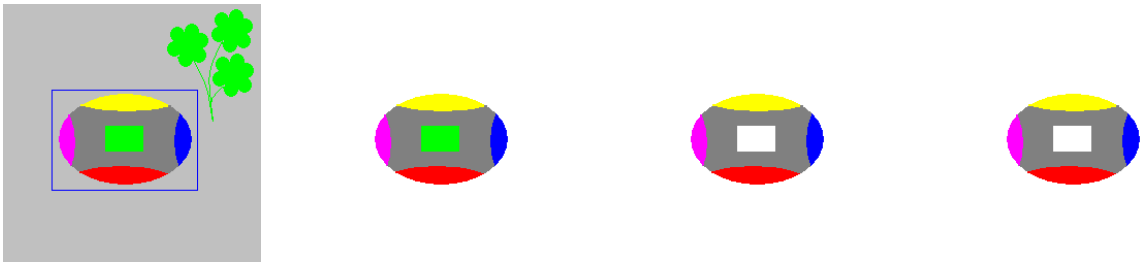
(a) (b) (c) (d)

(e) (f)

Figure 15: Comparison of the results: (a) SnakeCut result for soldier, (b) GrabCut Output of soldier with user interaction (reproduced from [14]), (c) SnakeCut result for wheel, (d) GrabCut Output of wheel with user interaction, (e) SnakeCut result for webcam bracket, (f) GrabCut Output of webcam bracket with user interaction.

to the background color. Snake and GrabCut results for this image are shown in Figure 13(b) and Figure 13(c) respectively. We can see that both these results are erroneous. Result obtained using Snake contains some part of the background which is visible through the handle. GrabCut has removed the spots in the handle since their color is similar to the background. Correct segmentation result produced by SnakeCut is shown in Figure 13(d). Objective in the webcam bracket example (Figure 14(a)) is to segment the lower bracket (inside the red contour initialized by the user) present in the image. Snake and GrabCut results for this image are shown in Figure 14(b) and Figure 14(c) respectively. We can see that both these results are erroneous. The result obtained using Snake contains some part of the background which is visible through the holes. GrabCut has removed large portions of the bracket. This is due to the similarity of the distribution of the metallic color of a part of another webcam bracket present in the background (it should be noted that the color distribution of the two webcam brackets are not exactly same due to different lighting effects). Correct segmentation result produced by SnakeCut is shown in Figure 14(d). We also observed a similar performance when the initialization was done around the upper bracket.

In Figure 15, we compare the automatic SnakeCut segmentation results of soldier (Figure 3(a)), wheel (Figure 4(a)) and webcam bracket (Figure 14(a)) images with the interactive GrabCut outputs. To obtain correct segmentation for these images with GrabCut, user interaction was necessary to obtain the results shown in Figures 15(b), 15(d) & 15(f). In case of soldier (Figure 15(b)), user marked the soldier's hat and legs as parts of the compulsory foreground. In case of wheel (Figure 15(d)) user marked the outer grayish green region of the wheel as a compulsory part of the foreground object and in case of webcam bracket (Figure 15(f)) user marked missing regions as compulsory parts of the foreground object. Segmentation results using SnakeCut were obtained without user interaction and are better than the results obtained by GrabCut with user's corrective editing. One can observe the smooth edges obtained at the legs of the soldier in Figure 15(a), unlike that in Figure 15(b). The same is true for Figure 15(c) and 15(e) (*w.r.t* Figures 15(d) and 15(f) respectively), which can be noticed after careful observation.



(a) (b) (c) (d)

Figure 16: Example where SnakeCut fails: (a) input image with foreground initialized by user, (b) Active Contour segmentation result (correct), (c) GrabCut segmentation result (incorrect), and (d) SnakeCut segmentation result (incorrect).

The presented approach takes advantage of Active Contour and GrabCut, and performs the correct segmentation in most cases where one or both of these techniques fail. However, the proposed technique (SnakeCut) was observed to have the following limitations:

1. Since the SnakeCut relies on Active contours for regions near the object boundary, it fails when holes of the object (through which the background is visible) lie very close to the boundary.

2. Since the Snake cannot penetrate inside the object boundary and detect holes, the proposed method of SnakeCut has to rely on the response of the GrabCut algorithm in such cases. This may result in a hazardous situation only when the GrabCut detects an interior part belonging to the object as a hole due to its high degree of similarity with the background. Since decision logic of SnakeCut relies on GrabCut response for interior parts of the object, it may fail in cases where GrabCut does not detect those parts of the object as foreground.

Figure 16 presents one such situation (using a simulated image) where SnakeCut fails to perform correct segmentation. Figure 16(a) shows a synthetic image where Active Contour works correctly (see Figure 16(b)) but GrabCut fails (see Figure 16(c)). GrabCut removes the central rectangular green part of the object in the segmented output, which may be perceived as a part of the object. We see in this case that SnakeCut also does not perform correct segmentation and removes the object's central rectangular green part from the segmentation result. SnakeCut thus fails when parts of the foreground object are far away from its boundary and very similar to the background.

The heuristic values of some of the parameters used in our algorithm, which were obtained empirically, were not so critical for accurate foreground object segmentation. The overall computational times required by SnakeCut on a P-IV, 3 GHz machine with 2 GB RAM, are given in Table 1 for some of the images.

## 6 Conclusion

In this paper, we have presented a novel object segmentation technique based on the integration of two complementary object segmentation techniques, namely Active Contour and GrabCut. Active Contour cannot remove the holes in the interior part of the object. GrabCut produces poor segmentation results in cases when the color distribution of some part of the foreground object is similar to background. Proposed segmentation technique, SnakeCut, based on a probabilistic framework, provides an automatic way of object segmentation, where the user has to only specify the rectangular boundary around the desired foreground object. Our proposed method is able to retain parts of the object which appears similar to background color and simultaneously eliminates holes with the object. We validate our technique with a few synthetic and natural images. Results obtained using SnakeCut are quite encouraging and promising. As an extension of this work, one can use geodesic Active

Table 1: Computational times for foreground object segmentation, required by Snake, GrabCut and SnakeCut for various images.

| Image Name | Image Size (in pixels) | Time required (in seconds) | | | |
|---|---|---|---|---|---|
| | | Snake (A) | GrabCut (B) | Integration time[a] (C) | SnakeCut (A+B+C) |
| Synthetic Image (Figure 2(a)) | $250 \times 250$ | 4 | 5 | 2 | 11 |
| Soldier Image (Figure 3(a)) | $321 \times 481$ | 8 | 10 | 3 | 21 |
| Wheel Image (Figure 4(a)) | $640 \times 480$ | 6 | 14 | 5 | 25 |
| Synthetic Image (Figure 9(a)) | $250 \times 250$ | 4 | 5 | 2 | 11 |
| Pot image (Figure 11(a)) | $296 \times 478$ | 6 | 7 | 4 | 17 |
| Cup image (Figure 13(a)) | $285 \times 274$ | 5 | 7 | 3 | 15 |
| Webcam bracket (Figure 14(a)) | $321 \times 481$ | 7 | 8 | 3 | 18 |

[a] time required to integrate Snake and GrabCut outputs using the probabilistic integrator.

Contour (which can intrinsically segment multiple objects) to make the technique suitable for the segmentation of multiple objects.

## References


[1] Y. Boykov and M.-P. Jolly. Interactive graph-cuts for optimal boundary and region segmentation of objects in N-D images. In *Proceedings of International Conference on Computer Vision, ICCV' 01*, volume 1, pages 105–112, 2001.

[2] H. Breu, J. Gil, D. Kirkpatrick, and M. Werman. Linear time euclidean distance algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(5):529–533, 1995.

[3] V. Caselles, F. Catte, and T. Coll. A geometric model for active contours. *Numerische Mathematik*, 66:1–31, 1993.

[4] V. Caselles, R. Kimmel, and G. Saprio. Geodesic active contours. *International Journal of Computer Vision*, 22(1):61–79, 1997.

[5] L. D. Cohen. On active contour models and balloons. *CVGIP: Image Understanding*, 53(2):211–218, 1991.

[6] P. Deepti, R. Abhilash, and S. Das. Integrating linear subspace analysis and interactive graphcuts for content-based video retrieval. In *Proceedings of International Conference on Advances in Pattern Recognition, ICAPR' 07, January 2-4, 2007*, pages 263–267, ISI Calcutta, India, 2007. World Scientific, Singapore.

[7] L. E. Elsgolc. *Calculus of Variations*. Pergamon Press, 1963.

[8] A. Haasch, N. Hofemann, J. Fritsch, and G. Sagerer. A multi-modal object attention system for a mobile robot. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2005*, pages 1499–1504, Edmonton, Alberta, Canada, 2005.

[9] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, January 1988.

[10] D.-J. Lee, J. Archibald, X. Xu, and P. Zhan. Using distance transform to solve real-time machine vision inspection problems. *Machine Vision and Applications*, 18(2):85–93, 2007.

[11] F. Leymarie and M. D. Levine. Tracking deformable objects in the plane using an active contour model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(6):617–634, 1993.

[12] B. Moller, S. Posch, A. Haasch, J. Fritsch, and G. Sagerer. Interactive object learning for robot companions using mosaic images. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2005, 2-6 August, 2005*, pages 2650– 2655, Edmonton, Alberta, Canada, 2005.

[13] D. W. Paglieroni, G. E. Ford, and E. M. Tsujimoto. The position-orientation masking approach to parametric search for template matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(7):740–747, 1994.

[14] C. Rother, V. Kolmogorov, and A. Blake. GrabCut: Interactive foreground extraction using iterated graph-cuts. *ACM Transactions on Graphics*, 23(3):309–34, 2004.

[15] M. Sanjay, S. Das, and B. Yegnanarayana. Robust template matching for noisy bitmap images invariant to translation and rotation. In *Indian Conference on Computer Vision, Graphics and Image Processing, December 21-23, 1998*, pages 82–88, New Delhi, INDIA, 1998.

[16] Surya Prakash and S. Das. External force modeling of snake using DWT for texture object segmentation. In *Proceedings of International Conference on Advances in Pattern Recognition, ICAPR' 07, January 2-4, 2007*, pages 215–219, ISI Calcutta, India, 2007. World Scientific, Singapore.

[17] D. Terzopoulos and K. Fleischer. Deformable models. *The Visual Computer*, 4(6):306–331, 1988.

[18] D. Terzopoulos and R. Szeliski. *Tracking with Kalman Snakes*. MIT Press, Cambridge, MA, 1992.

[19] P. Tsang, P. Yuen, and F. Lam. Classification of partially occluded objects using 3-point matching and distance transformation. *Pattern Recognition*, 27(1):27–40, 1994.