オーディオビジュアル複合情報処理 27－1
(1999. 12. 16)

動的輪郭モデルパラメータの設定について

鐘 文　　長谷山 美紀　　北島 秀夫

北海道大学大学院 工学研究科
〒060-8628 札幌市北区北13条西8丁目
北海道大学大学院 工学研究科 電子情報工学専攻
情報メディア工学講座 像情報工学分野
Tel: (011)706-6078
E-mail: zhongw@media.eng.hokudai.ac.jp

あらまし　動的輪郭モデルは輪郭抽出の有効な手法である。この手法においては、試行誤差によって適当なパラメータを設定する必要があり、これが具体的応用への障害である。本論文では、強化学習を用いて輪郭モデルパラメータを設定する方法を提案する。本提案方式においては既存の強化学習手法の流れを修正して、ロバストではないという欠点を改善する。また、ヒューリスティックルールを加えて、探索時間を従来法より大幅に減少させる。動的輪郭モデルパラメータを設定する実験結果から、提案手法は従来法より有効であることを示す。

キーワード　動的輪郭モデル、輪郭抽出、パラメータ設定、強化学習、ヒューリスティックルール

Determination of Parameters of Active Contour Models

Wen ZHONG　　Miki HASEYAMA　　Hideo KITAJIMA

School of Engineering, Hokkaido University
Kita-13 Nisi-8,kita-ku, Sapporo-shi, 060-8628, Japan.
Tel (011)706-6078
E-mail: zhongw@media.eng.hokudai.ac.jp

Abstract　Active contour models are effective in contour extraction. One difficulty with models the technologies is that much work is needed for reliable parameter acquisition, which is conventionally done by trial-and-error methods. This paper introduces reinforcement learning to determine parameters for specific applications. By modifying the learning stream, our system overcomes a drawback exisiting in the reinforcement learning method that it was not robust . Exploration time is shortened by adding heuristic rules. Experimental results show that our systems work effectively.

key words　active contour model, contour extraction, parameter determination, reinforcement learning, heuristic rule

1 Introduction

Active contour models, known as Snakes, being effective technologies for computer vision, have drawn a substantial amount of attention recently[2, 3, 4]. The original snake models are introduced by Kass *et al*[1]. A snake is an energy-minimizing spline. It is guided by internal forces coming from the curve itself. It is also influenced by image forces which pull it toward the features of interest such as the contour of an object. Some issues must be settled before the original snake algorithm is put to use. Initially it must be very close to the target object in order that important true features can be captured. The snake is very sensitive to noise.

To address the problems, we have presented a new active contour model called the shape-constraint-based active contour model (SC-ACM)[7] to extract the contour of an object whose approximate shape is known *a priori*. In the reference[7], like in other active contour models, all parameters used in the SC-ACMs are get by trial-and-error method. There is need to find a way to automatically to set the optimal parameters for specific applications.

Reinforcement learning (RL) is an approach of machine intelligence. A RL agent learns a mapping from situations to actions so as to maximize a scalar reward or reinforcement signal. The characteristic, trial-and-error searching, is one of the most important distinguished features of reinforcement learning. RL appeals to many researchers because of its generalities. It has been successfully used for computer vision application[6].

In the reference[6], Peng *et al* used a RL system to learn the parameters for image segmentation algorithms. If we can establish a reward rule for the contour extraction system, similar methods can be used for automatically setting parameters for active contour models though some drawbacks still exist in the Peng method.

In this paper, we try to overcome the drawbacks existing in the Peng method and use some heuristic methods to improve its exploration policy; therefore to increase the behavior of the RL.

2 Active Contour Models

An original snake[1] is a deformable curve. A snake curve has two energy function

$$\begin{aligned} E_{snake} &= \sum_{i=1}^{n} E_{int}(v_i) + E_{ext}(v_i) \\ E_{int}(v_i) &= (\alpha||v_i - v_{i-1}||^2 \\ &\quad +\beta||v_{i-1} - 2v_i + v_{i+1}||^2)/2 \\ E_{ext}(v_i) &= w_e E_e(v_i) = -w_e|\nabla I(v_i)|^2 \end{aligned} \quad (1)$$

where α,β and w_e are weighting parameters; n, v_i and $I(v_i)$ are the number of snake points, the coordinate of the ith snake point and the intensity value on position v_i respectively.

A snake that minimizes energy E_{snake} must satisfy the Euler equation[1]

$$\nabla E_{int}(v_i) + \nabla E_{ext}(v_i) = 0. \quad (2)$$

This can be viewed as a force-balance equation

$$\mathbb{F}_{int}(v_i) + \mathbb{F}_{ext}(v_i) = 0 \quad (3)$$

$$\mathbb{F}_{int}(v_i) = \nabla E_{int}(v_i) \quad (4)$$

$$\mathbb{F}_{ext}(v_i) = \nabla E_{ext}(v_i) = w_e \mathbb{F}_e(v_i) = w_e \nabla E_e(v_i) \quad (5)$$

where $\mathbb{F}_{int}(v_i)$ is the internal force and $\mathbb{F}_{ext}(v_i)$ is the external force. This is to say that a snake moves under the influence of the forces till the forces reach a balance.

To solve the problems existing in the original snake that it is very sensitive noise and its initial position must be very close to the target, we have presented a new active contour model called the shape-constraint-based active contour model (SC-ACM)[7] to extract the contour of an object whose approximate shape is known *a priori*. A SC-ACM adds a shape force $\mathbb{F}_s$ to the $\mathbb{F}_{ext}$ to insure its shape against departure from the reference too far as the snake moves. The external force of the SC-ACM is modified as follows:

$$\mathbb{F}_{ext}(v_i) = w_e \mathbb{F}_e(v_i) + w_s \mathbb{F}_s(v_i) \quad (6)$$

where $\mathbb{F}_e(v_i)$ is the edge force shown in formulation (5) and $\mathbb{F}_s(v_i)$ denotes the shape-constraint force for each snake point v_i. w_s is the weighting parameter of shape force. The shape-constraint force is defined as

$$\mathbb{F}_s(v_i) = \begin{cases} d_i \mathbb{f}_i & \text{if } |d_i| > d_{th} \\ 0 & \text{otherwise.} \end{cases} \quad (7)$$

where d_i is the distance between v_i and the reference, which is determined according to the shape information, and $\mathbb{f}_i$ is the force unit vector which is normal to the snake at the point v_i and d_{th} is a threshold for the d_i.

In the original snake, parameter setting is a difficult work. A final result depends on the relationship of the weighting parameters of the forces. The difficulty of parameter setting increases with the shape force added in the SC-ACM.

3 Reinforcement Learning

Reinforcement learning (RL) is an approach of machine intelligence. A RL agent learns to perform actions that will maximize the sum of the reinforcements received when starting from some initial state and proceeding to a terminal state. The RL has been used for many applications such as computer vision[6]. In the Peng method[6], the connectionist REINFORCE algorithm is used to learn parameters for image segmentation algorithms. The RL uses a team architecture network (Figure 1), which is called team of Bernoulli quasilinear units, where units in the network are *Bernoulli quasilinear units*, in that the output of such a unit is either 0 or 1, determined stochastically using the Bernoulli distribution with parameter $p = f(s)$ where f is the logistic function

$$f(s) = 1/(1 + exp(-s)) \quad (8)$$

and $s = \sum_i w_i x_i$ is the usual weighted summation of input values to that unit. For a unit, p represents its probability of choosing 1 as its output value. For an

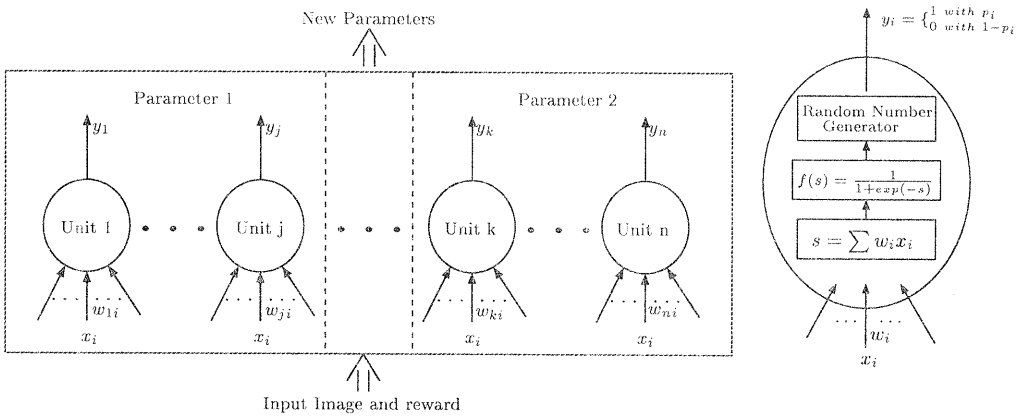

Figure 1: (a) Team of Bernoulli quasilinear units architecture. (b) Bernoulli quasilinear unit

LOOP

1. $rr = 0$ (rr: average reward)
2. For each image i in the training set do
   (a) Segment image i using current parameters
   (b) Get reward r
   (c) $rr = rr + r$
   (d) Obtain new parameters using r as reward for the reinforcement learning algorithm

UNTIL number of iterations is equal to N or $rr/n \geq R_{th}$

Figure 2: Main steps of the Peng method

input, the network generates an output and the environment responds by providing a reward r as its evaluation of the output, which is then used to drive the weight changes. At the tth time step, the weight increment is

$$\begin{aligned} \Delta w_{ij}(t) &= \alpha_{RL}(r(t) - \bar{r}(t-1))(y_i(t) - \bar{y}_i(t-1))x_j \\ &\quad -\delta w_{ij}(t) \qquad (9) \\ \bar{r}(t) &= \gamma_{RL}\bar{r}(t-1) + (1-\gamma_{RL})r(t) \\ \bar{y}_i(t) &= \gamma_{RL}\bar{y}_i(t-1) + (1-\gamma_{RL})y_i(t) \end{aligned}$$

where α_{RL} is a positive learning rate, x_j is the input of each unit, $y_i(t)$ and $r(t)$ are the output of the ith unit and the received reward at the tth time step respectively, δ is the weight decay rate, γ_{RL} is the *trace* parameter[6], $0 < \gamma_{RL} < 1$, $\bar{r}(0) = 0$ and $\bar{y}_i(0) = 0$. Figure 2 shows the main steps of the algorithm of the Peng method. In theory, the algorithm is guaranteed to statistically converge to a local optimum[5, 6]. In real applications, the outputs are determined stochastically by Bernoulli distribution. Each set of the parameters corresponds to one image in the training set and yields one reward. In the worst case, one set of the parameters yields a high reward for the final image in the training set, but the set of the parameters can not be guaranteed to yield high rewards for the other images. Therefore, the algorithm of the Peng method is not robust. One more drawback of the Peng method is that its learning time is long when R_{th} is set to be a big value; on the other hand it is very easy trapped in a local solution when R_{th} is set to a not big one. The latter drawback is particularly distinct for the problems which are very sensitive to parameters.

4 Improving Approaches

In the Peng method, the RL is used to learn parameters for image segmentation algorithms. Similar approaches also can be considered to be used for learning parameters for the SC-ACMs if the drawbacks existing in it can be overcomed.

4.1 Modification of algorithm stream

To address the first problem existing in the Peng method, we modify the stream of algorithm as Figure 3. The R'_{th} is a constant which is a little smaller than the average reward threshold R_{th}. In all our applications, we set $R'_{th} = R_{th} - 0.05$. As shown in Figure 3, the final result is aimed at all images in the training set instead of only one image in the Peng method. Thus the result can be guaranteed to yield a high average reward over all images in the training set.

4.2 Using Heuristic Rules

For a team architecture, one parameter is encorded by the outputs of some units. In the Peng method, a parameter is represented by five-bit Gray code. In Gray

(A) $count = 0$

(B) Obtain default parameters randomly

LOOP

For each image i in the training set do

(a) Extract the contour for image i using current parameters

(b) Get reward r

(c) $count = count + 1$

(d) If $r \geq R'_{th}$

(1) $rr = r$ (rr: average reward)

(2) For each image k in the training set except image i do

(i) Extract the contour for image k using current parameters

(ii) Get reward r

(iii) $rr = rr + r$

(iv) $count = count + 1$

(3) if $rr/n \geq R_{th}$ then terminate

(e) Obtain new parameters using r as reward for the reinforcement learning algorithm

UNTIL $count \geq N$

Figure 3: Main steps of Our methods

code, if a parameter p_i is encoded by $(y_4, y_3, y_2, y_1, y_0)$, we know that if $(y_4, y_3, y_2, y_1, y_0)$ changes from (00000) to (10000) or from (10000) to (00000), the value of p_i will change 15; on the other hand, if the value changes from (00000) to (00001) or from (00001) to (00000), the p_i will only change 1. For p_i, the influences come from y_0 and y_4 are different. In the Peng method did not consider the difference, all units are treated equally. Experimentally, we wish the possibility for a parameter changing becomes higher when the reward descents and become smaller when increases. To simulating the human experience, we add some heuristic rules to the algorithm. We use the simple binary code instead of the Gray code of the Peng method and add some hueristic rules to the R-L system. To simplify discuss our problems, we assume that one output parameter is encorded by n units and outputs of the n units are $(y_{n-1}, y_{n-2}, \ldots, y_1, y_0)$ and its value is equal to $\sum_{i=0}^{n-1} y_i 2^i$.

4.2.1 Hueristic Rule 1

One factor influencing the output of one unit is the weights of the unit. The bigger the change of the weights is, the bigger the possibility for the output changing is. We add a heuristic rule to adjust the weights increment. In the weights increment formulation (9), we add a unit factor k_i to it.

$$\begin{aligned}\Delta w_{ij}(t) &= \alpha_{RL} k_i (r(t) - \bar{r}(t-1))(y_i(t) - \bar{y}_i(t-1))x_j \\ &\quad -\delta w_{ij}(t) \end{aligned} \tag{10}$$

For a parameter, the unit factors are defined as formulation (11)

$$\{k_{n-1}(t), \ldots, k_0(t)\} = \begin{cases} \{k_{i+1}(t) \geq k_i(t) | 0 \leq i < n-1\} & r(t) < r(t-1) \\ \{k_{i+1}(t) \leq k_i(t) | 0 \leq i < n-1\} & r(t) > r(t-1) \end{cases} \tag{11}$$

where $k_i(t)$ is the unit factor of ith unit at tth time step. The parameter may change in a big step if the reward descents, on contrary, the parameter may change in a small step if the reward increases. By this way we can adjust the weight changing amplitude according to the received reward. Using this heuristic rule, the exploring behavior of the system, in some sense, likes human exploring behavior more. This can help the system save its exploration time.

4.2.2 Hueristic Rule 2

The output of one unit is not only influenced by the inputs controlled by the weights, but also influenced by the logistic function (8). For the same reason as the heuristic rule 1, we can also add a unit factor to the logistic function to adjust the unit output.

Instead formulation (8), we use the following formulation as its logistic function.

$$f'(s) = 1/(1 + exp(-s/k_i)) \tag{12}$$

where k_i is a unit factor, which is definited as the same as formulation (11).

Generally, the bigger the inputs are, the bigger the possibility for the output becoming 1 is. Even if the inputs are positive and are very big, the possibility for the output being 0 rather exist; the bigger the k_i is the higher the possibility is. Thus the RL can control the behavior of exploration and save its exploration time by adjusting the value of k_i according to reward received from the system.

5 Experimental Results

An experiments waw performed to evaluate the improving algorithms on the SC-ACMs which are used to extract the contours of some synthetic images. Before giving the experiment, we would like to show the features of the parameters of the SC-ACMs and establish a simple reward rule for the learning system.

5.1 Parameters of the SC-ACMs

In a 2-D application, the force balance equation (3) can be written in matrix form[1] as

$$\begin{aligned}\mathbf{A}\mathbf{x} + \mathbf{f}_x(\mathbf{x}, \mathbf{y}) &= 0 \\ \mathbf{A}\mathbf{y} + \mathbf{f}_y(\mathbf{x}, \mathbf{y}) &= 0 \end{aligned} \tag{13}$$

where $\mathbf{f}_x(\mathbf{x}, \mathbf{y})$ and $\mathbf{f}_y(\mathbf{x}, \mathbf{y})$ are components of force $\mathbf{F}_{ext}$ in x and y axis directions respectively, $\mathbf{A}$ is a pentadiagonal matrix whose band is a function of α and β. The following equation are iteratively to find a solution to (13):

$$\begin{aligned}\mathbf{A}\mathbf{x}_t + \mathbf{f}_x(\mathbf{x}_{t-1}, \mathbf{y}_{t-1}) &= -\gamma(\mathbf{x}_t - \mathbf{x}_{t-1}) \\ \mathbf{A}\mathbf{y}_t + \mathbf{f}_y(\mathbf{y}_{t-1}, \mathbf{y}_{t-1}) &= -\gamma(\mathbf{y}_t - \mathbf{y}_{t-1}) \end{aligned} \tag{14}$$

where γ is a step size.

Table 1: The Comparison Result

Method	Parameters					Training image	Reward	Number of passing images	average reward on testing data
		β	w_e	w_s	γ				
Peng		0.5000	2.3715	0.3536	13.4543	frame 1	0.8339	348	0.5874
		0.4204	2.3715	5.6569	11.3137	frame 2	0.8278		
	*	0.5000	3.0855	5.6569	4.0000	frame 3	0.8995		
						frame 1	0.5146		
						frame 2	0.4404		
Heuristic Rule 1	*	0.7071	2.8305	5.1878	4.0000	frame 1	0.9112	177	0.8047
						frame 2	0.8368		
						frame 3	0.9075		
Heuristic Rule 2	*	1.0000	2.3715	1.4142	3.6680	frame 1	0.8504	136	0.8019
						frame 2	0.9071		
						frame 3	0.8953		

The parameters which should be determinated are α, β, w_e, w_s and γ. Each parameter measures one aspect of a snake. In formulation (14), if we double all these parameters, The results are the same. Therefore, if we set one of the parameters as the base, we only need to find the relative values for the other parameters. In all our applications, we set α as the base and let $\alpha = 1$.

5.2 Reward Rule

We defined the following formulation as the reward derived from system:

$$r = 1/(1 + MSE/RB). \tag{15}$$

Here, MSE is the difference between the detected contour and the expected contour[7]. RB is a constant measuring the size of target which is experimentally set to near the square root of area of a rectangular that circumscribes the expected contour. Since the SC-ACMs are not move in one way, they can expand and contrast. During the learning procedure, the models may move out of the work region[7] or the image. Once a model move out of the work region or the image, the parameters are regarded not suit the model. Then the model stops moving, and the reward is set equal to 0. This strategy can help the system do not waste time to explore in vain.

5.3 RL System Used in the experiment

We use the same architecture shown in Figure 1 as ours. In the experiment, $\alpha_{RL} = 0.2$, $\delta = 0.01$ and $\gamma_{RL} = 0.9$. Each parameter is encoded by six units. Each unit has a total 9 input weights. The unit factors of formulation (11) for one output parameter is definited as follows:

$$\{k_5, \ldots, k_0\} = \begin{cases} \{\mu, \sqrt{\mu}, 1, 1, 1/\sqrt{\mu}, 1/\mu\} & r(t) < r(t-1) \\ \{1/\mu, 1/\sqrt{\mu}, 1, 1, \sqrt{\mu}, \mu\} & r(t) > r(t-1) \end{cases} \tag{16}$$

where μ is a scale factor and $\mu > 1$.

For that each parameter is encoded by six units, the range of each output parameter p_o of the RL system is between $[0, 63]$. The correspondence between the output parameter (p_o) and the parameter used in the SC-ACM (p_m) can be established by the following formulation.

$$p_m = 2^{(p_o - 31)/8} \tag{17}$$

The learning range of each parameter of the SC-ACM is between $2^{-31/8}$ and 16.

5.4 Application for Synthetic Images

We apply the improved RL methods to learning the parameters for the AC-ACMs[7] to extract contours for synthetic images.

Three 150x120 images, each consisting of a target object with a difference wave-like shape and six circles, three over the target and three under it, against backgrounds are generated at first. Then 45 images further are generated from the three images. For each image, the background and the target are generated by two Gaussian distributions with μ_b and μ_t, and standard deviations $\sigma_b = \sigma_t = \sigma$ respectively, and a Gaussian random noise $N(0, \sigma_n)$ is added. The six circles have the same distribution and standard deviation as the target. In the 45 images, $\mu_b = 125$, μ_t varied from 140 to 160, $\sigma = 5$ and σ_n varied from 15 to 20. We divide the 45 images into three group. The images which have the same wave-like shape belong to one group. Figure 4 (a)-(c) show one example of each of the three group respectively.

We randomly select three images (Group 1, $\mu_t = 150$, $\sigma = 20$; Group 2, $\mu_t = 150$, $\sigma = 15$ and Group 2, $\mu_t = 160$, $\sigma = 10$) from the 45 as training set and the rest 42 images are testing data. Each unit of the network takes average gray value of input on a 50x40 neighborhood on the input image. The average is normalized to lie between 0 and 1. R_{th} is set equal to 0.85, μ is set to be 1.5 and RB is equal to 60 . Table 1 shows the comparison experimental results of the Peng method and our presented two methods. These experiments all learning begin with a same set of default parameters. From Table 1, we can see that for each image in the training set, the Peng method get a set of parameters to yield a high reward. But if the final result (marked with * in Table 1) is used for the other two training images, the rewards are only 0.5146 and 0.4404 respectively and the achieved average reward on the test data is only 0.5874.

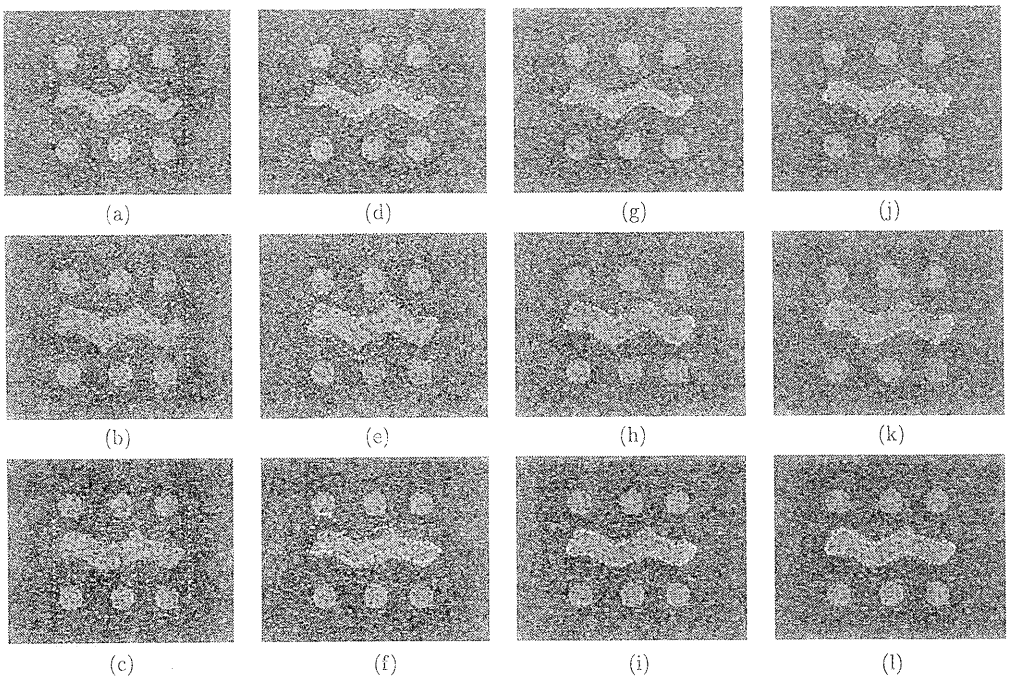

Figure 4: Extraction performance of the SC-ACMs with learned parameters (a)-(c) are initial images ($\mu_t = 160$, $\sigma_n = 15$). (d)-(f) are from the Peng method. (g)-(i) are from the Heuristic Rule 1. (j)-(l) are from the Heuristic Rule 2.

On the other hand, our method can avoid the drawback existing in the Peng method. Their exploration time is quite short than the Peng method too. Figure 4 (d)-(l) show the extraction performance of the SC-ACMs with learned parameters from the three RL methods applied to the images shown in Figure 4(a)-(c).

6 Conclusion

We try to use the reinforcement learning to automatically determine parameters for the SC-ACMs in this paper. By modifying the learning stream, our proposed methods can overcome the drawback existing in the presented method that it was not robust. With the help of two heuristic rules, the RL systems can short much exploration time. The experimental results demonstrated our methods work well than the presented method. Although our methods were only used for determining parameters for the shape-constraint-based active contour models in this paper, we believe that it can be used for automatically determining parameters for other active contour models and similar automatic parameter determination problems too.

References

[1] M. Kass, A. Witkin and D. Terzopoulos, *"Snakes: Active contour models,"* International Journal of Computer Vision, pp. 321-331, 1988.

[2] Lawrence H. Staib and James S.Duncan, *"Boundary finding with parametrically deformable models,"* IEEE Trans. Pattern Anal. Machine Intell., Vol. 14, No. 11, pp. 1061-1075, 1992.

[3] Laurent D. Cohen and Isaac Cohen, *"Finite-element methods for active contour models and balloons for 2-D and 3-D images,"* IEEE Trans. Pattern Anal. Machine Intell. , Vol. 15, No. 11, pp. 1131-1147, 1993.

[4] Steve R. Gunn and Mark S. Nixon, *"A robust snake implementation; a dual active contour,"* IEEE Trans. Pattern Anal. Machine Intell. , Vol. 19, No. 1, pp. 63-68, 1997.

[5] Ronald J.Williams, *"Simple statistical gradient-following algorithms for connectionist reinforcement learning,"* Machine Learning, Vol. 8, pp. 229-256,1992

[6] Jing Peng and Bir Bhanu, *"Closed-loop object recognition using reinforcement learning,"* IEEE Trans. Pattern Anal. Machine Intell., Vol. 20, No. 2, pp. 139-154, 1998.

[7] W. Zhong, M. Haseyama and H. Kitajima, *"A shape-constraint-based active contour model,"* The Journal of the Institute of Image Information and Television Engineers, Vol.53, No.10, pp. 1421-1429, 1999.