# A Self-Adjusting Mechanism for Active Contour Models

Jiankang Wang and Xiaobo Li
Department of Computing Science
University of Alberta
Edmonton, Alberta, Canada T6G 2H1

March 29, 2000

## Abstract

In this paper we propose a self-adjusting mechanism for the balloon model, an improved version of the famous active contour models, in an effort to solve the problem we call the "peeling effect" we observed in some applications. The balloon model can expand to search for edges by using a normal pushing force to overcome the shrinking force from the original internal energy model. However, choosing an appropriate balloon force is a tough task, especially for boundaries with non-constant magnitudes. Once part of a balloon has overrun a weak boundary, it can easily pull its neighboring parts away from their equilibrium locations. We devise a self-adjusting mechanism to cope with this "peeling effect" problem. The method implements the idea of a "helping" snake: The snake parts that have located strong edges can direct the movement of their neighboring parts, thus offsetting the "peeling effect". Experimental results show that the new model reduces the difficulty of choosing parameters and is rather robust.

## 1 Introduction

Active Contour Models [5], or snakes, are considered an effective and robust way for contour extraction because they extract edges based on information not only from a specific pixel but also from the spatial distribution. They differ from traditional edge detecting methods by using smoothness and other constraints from high level modules or user input. However, the original snake model suffers from a few shortcomings. First, a snake model in the original form is myopic, *i.e.* it is not able to locate a far away boundary. This requires a close initialization from user or other modules, thus restricting its use. Second, the original snake could be easily trapped by local minima if used with noisy images.

Cohen and Cohen [3] proposed a *balloon* model which uses an inflating force to make the contour expand until it meets an object boundary. The balloon model partially solves the initialization problem. However, as pointed out by Xu *et al.* [11] a snake tends to shrink due to its internal forces and even worse, the internal forces are not homogeneous along the boundary, being large at points with high curvature. This makes it difficult to choose a constant inflating force for the balloon model.

We observed a phenomenon, which we call a "peeling effect", in using the balloon model with some noisy ultrasound images. Boundaries in these images are typically incomplete, being missing or weak at some places. Moreover, there are a large number of erroneous edges which are comparable in magnitude to those boundary edges. To apply the balloon model, we have to choose an inflating force large enough to overcome the erroneous edges. However, this would also cause some parts of the balloon to overrun some weak boundaries. Then these parts would pull their neighboring contour parts away from their otherwise equilibrium locations.

In this paper, we propose a self-adjusting mechanism which uses an iterative process to dynamically identify snakes parts that have successfully located some strong boundaries. Once a few parts have been identified, an influence force is propagated from these parts to their neighbors to counter the "peeling" force.

The outline of this paper is as follows. In Section 2, the "peeling effect" is described in more detail. In Section 3 and 4, we present our dynamic boundary verifying procedure and use information acquired to manipulate the balloon evolution. Experimental results and conclusions are reported in Section 5 and 6.

## 2 Problems with the Balloon Model

Choosing an appropriate inflating force for a balloon model is not an easy task. For boundaries with weak or missing parts, the "peeling effect" problem described in Section 1 could occur. To further understand the problem some illustrations are provided. Fig. 1 (a) shows a synthetic boundary with one gap on it. The boundary magnitude changes gradually from small to normal from the gap. Suppose we choose an inflating force which can overcome an average magnitude edge. A probable intermediate state of the balloon would be

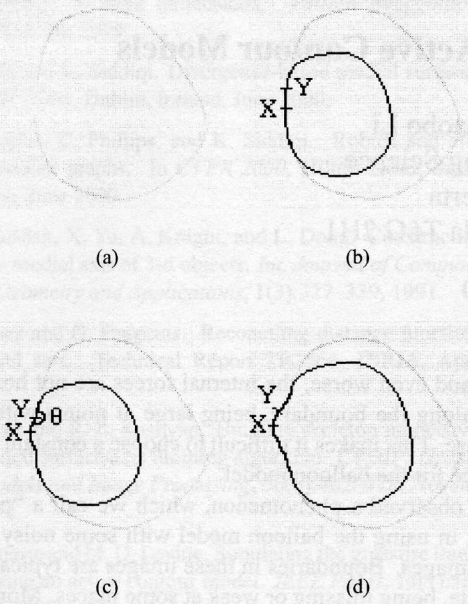



Figure 1: An illustration of the "peeling effect": (a) A synthetic boundary; (b) A probable state of the balloon; (c) Snaxel X keeps moving outward; d) Snaxel Y is pulled away from its equilibrium location.

like the one shown in Fig. 1 (b). Now we check the status of two neighboring snaxels, X and Y, with X on the gap and Y on the brink of the gap. Each of them is under the influences of three forces, namely, a gradient force, an internal smoothing force, and an inflating force. Suppose that snaxel Y is now in equilibrium under the influences of these three forces. For X, the gradient force is small compared with the inflating force, so it will continue to move outward (Fig. 1 (c)). This will cause a change in the internal force exerted on Y since the internal angle $\theta$ is getting bigger. Eventually this force would pull Y away from its equilibrium state, and reach a state as shown in Fig. 1 (d). This is what we call a "peeling effect".

The reason behind this problem is two-fold. For those snaxels in equilibrium, a small extra force could disturb the balance. For snaxels without any gradient force, the inflating force is enough to defeat the internal smoothing force; otherwise it would not be able to make the snake expand in the first place.

# 3 Dynamic Boundary Verification

We now try to devise a method to dynamically identify snake parts that have successfully located some strong boundaries. For each snaxel, an appropriate measure should be defined

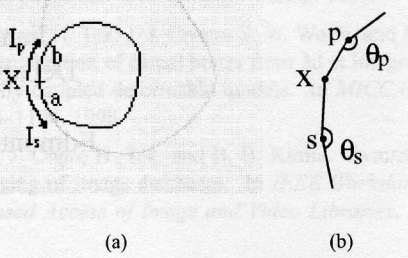



Figure 2: Measuring influences: (a) For X, two influence measures are computed, $I_p^x$ and $I_s^x$; (b) Angles used in influence measures computation.

and computed. Naturally, the measure for a snaxel should be large if it lies on a strong boundary. The task is similar to identifying salient structures in images [8, 4], except for two differences. First, the latter is two dimensional, *i.e.* for each pixel, a number of directions should be checked for possible connections with salient structures. However, for a snaxel, we only need to check two directions, its predecessor[1] and successor on a snake. Second, a snake is always evolving, so the computation should be dynamic as well. Nonetheless, the guidelines presented in these two papers could still be used.

Since our final objective is to use the measures of some snaxels to direct the movement of their neighbors, we represent them in terms of influence factors. Since the influence factors are bi-directional, two measures should be defined for each snaxel. Let $I_p^X$ stand for influence for snaxel, X, on its predecessor from its successor's side, and $I_s^X$ the opposite one. For snaxel X in Fig. 2 (a), $I_p^X$ should be larger than $I_s^X$ because part $a$ lies on a strong boundary while part $b$ does not. Some qualitative guidelines in computing these two measures are listed as follows:

- Strong support from image features—Usually gradient information is used.
- Strong connections—Since snaxels are separated by some distance, image features along the connecting route between them are an important contributing factor to the measures.
- Smoothness constraint—Smooth curves are preferred over non-smooth ones.

A simple way to compute the measures is to calculate an integral of image features, along with other factors, for a

[1] Snakes discussed in this paper are closed and in anti-clockwise order.

fixed length $l$, starting from each snaxel. However, the complexity of the algorithm is $O(n*l)$, where $n$ is the number of snaxels. We propose to use a simpler algorithm with complexity $O(n)$. The algorithm is used in an iterative way so that influence measures are propagated along the snake, thus having the same effect as computing an integral.

The algorithm is used along with a greedy heuristic search for minimizing the snake energy [10]. To avoid the side-effects that a sequential updating method may have on computing the two measures, we use a random order updating method. Each time a random number within $(1, n)$ is generated. Then the energy of the snaxel labelled with that number is minimized, and the two measures associated with it are updated using the following formulae:

$$I_p^X = I_{image} + \alpha_s I_p^s \cos(\theta_s - \theta_{avg}) \quad (1)$$

$$I_s^X = I_{image} + \alpha_p I_s^p \cos(\theta_p - \theta_{avg}) \quad (2)$$

where $I_{image}$ is the normalized magnitude of the gradient at X. $I_p^s$ is $I_p$ computed at its successor's position, and $I_s^p$ is $I_s$ from its predecessor. $\alpha_s$ is a measure of the connection between the current snaxel and its successor and is computed by normalizing the average gradient magnitude along the straight line that connects the two snaxels. $\alpha_p$ is computed similarly. Finally, $\theta_s$ is the internal angle formed by the current snaxel and its two successors (Fig. 2 (b)), $\theta_p$ is the internal angle formed by its two predecessors, and $\theta_{avg}$ is the average internal angle.

Eqns. 1 and 2 embody the guidelines we presented above. $\alpha$ introduces the connections between the current snaxel and its two immediate neighbors. $\cos(\theta - \theta_{avg})$ is designed to favor influence from smooth neighboring snaxels. $I_{image}$ is a contribution from the snaxel in question. By updating the measures iteratively, influences from one snaxel could propagate to far away snaxels, depending on how parameters are chosen. An example is shown in Fig. 3. Note that for almost all snaxels which are inside the boundary, both $I_p$ and $I_s$ are small. Moreover, the few with relatively large measures are not able to propagate this information to their neighbors because of poor connections between them and their neighbors. For snaxels that lie on a piece of strong boundary, there are some noticeable differences between $I_p$ and $I_s$. An example is with snaxel X, where $I_p$ is smaller than $I_s$ (Fig. 3 (b) and (c)).

# 4 The Self-Adjusting Mechanism

The self-adjusting mechanism's objective is to use a snaxel's influence measures to direct the search of its neighboring parts. Once a part of a snake has hit a strong boundary, it predicts some directions for its neighboring parts to search. To be consistent with snake models, "directing" is applied by adding one more kind of stiffness force. The forces are generated by snake parts that have located salient image features, and are applied to their neighboring parts. There are two problems that should be solved to make the algorithm work:

- First, how to judge if a snake part has hit a strong boundary? How large should the influence measures be to qualify as a strong boundary?
- Second, how to predict the directions and how to assign forces?

Our current solutions are relatively simple, and work with a discretized snake directly. For each snaxel, the influence measures from its predecessor and successor are compared. If one is larger than the other by a certain amount, then the side that the bigger influence is from, is considered a part of a strong boundary; influence forces are generated and applied to the other side. By using relative measures instead of absolute ones, we circumvent the problem of choosing a threshold for strong boundaries. Experiments show that choosing a difference threshold for two measures is relatively easy, and that one value works with most cases.

To generate influence forces, we need to decide the direction and magnitude of the forces. Fig. 4 is an example of how we achieve that. Suppose snaxel X is the one in question, and Y is its successor with a large influence measure. Our task is to use the shape, distance, *etc.*, of part a to predict a position for snaxel X. A natural way to do that is to extend the curve along the tangent (Fig. 4 (a)). Or we can extend the curve in such a way that the internal angle $\theta$, shown in Fig. 4 (b), equals to the average internal angle. The differences between the two methods are small. We choose to implement the latter, and use position information of two snaxels only. To estimate a more accurate tangent direction, a more complex method could be used. This is still being investigated.

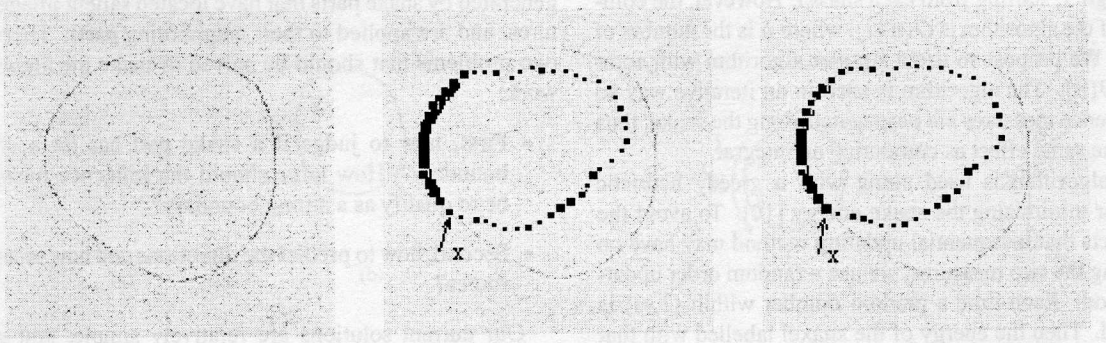
Once the curve extending method is decided, the predicted position for the current snaxel is calculated simply by extending the curve by a length of $l$, which is the distance between snaxels X and Y. The resulting positions are shown in Fig. 4 as Z and Z'. In Fig. 4 (c), positions predicted from both sides of snaxel X are shown. Which one to use depends on which of the measures is bigger. Let $\mathbf{v}$ be the influence force vector, we set

$$\mathbf{v} = Z - X. \quad (3)$$

The magnitude of the vector $\mathbf{v}$ is subject to some scaling factor.

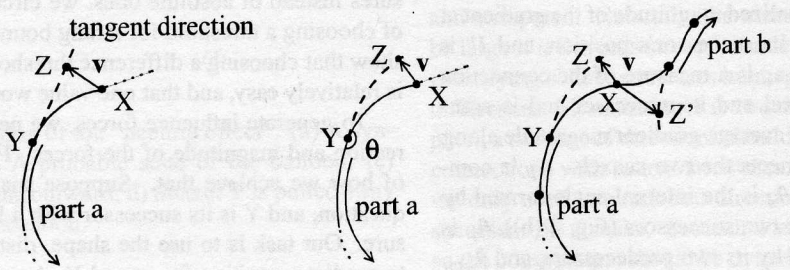
# 5 Implementation and Experiments

We evaluate our method by comparing results generated by a balloon model, both equipped with and without the self-




(a) (b) (c)

Figure 3: An example of influence measures calculation: (a) A synthetic boundary with noise; (b) An example of $I_p$, where the size of the snaxels indicates magnitude of the measure; (c) The corresponding $I_s$.




(a) (b) (c)

Figure 4: An example of generating influence forces: (a) Part $a$ is extended in the tangent direction; (b) Extended by maintaining the internal angle; (c) Extending is done from both directions.

adjusting mechanism. Testing images include some synthetic line-drawings with different noise levels and some noisy real images.

First, some implementation details are introduced. For the sake of simplicity, the fast greedy heuristic algorithm [10] is used to minimize the snake energy. However, this algorithm may produce worse results than other methods do. We also implemented the more accurate dynamic programming method [1] for the original balloon model and used the best results for comparison. Each snake starts with a very rough initialization inside the boundary we are looking for. Additional snaxels are added at places where the distance between consecutive snaxels exceeds a predefined length[2]. In the experiments shown below, the distance between snaxels are maintained at 4 pixels.

The balloon forces and the influence forces produced by the self-adjusting mechanism are converted into a potential field so that they can be used along with other energy functions. Suppose $\mathbf{v}$ is the influence force vector for snaxel X (Fig. 5). For each of X's 8 neighbors, the potential energies are assigned in the following way:

$$E_{influ} = -\frac{1}{2}\kappa(\bar{\mathbf{v}} \cdot \bar{\mathbf{w}}) \left(\frac{|\mathbf{v}|}{h}\right)^2 \quad (4)$$

where $\kappa$ is a scaling factor. Barred symbols are unit vectors, and $\bar{\mathbf{w}}$'s direction is from X to each of its neighbors. $h$ is the average distance between snaxels and is used to normalize the energy. The operator $\cdot$ is the usual vector dot product.

A snake may oscillate under the influence of all these forces [6]. A simple termination criterion, such as to see if any of snaxels is moving, is usually unsuccessful. We devised a more robust criterion: For each snaxel, a list of 5 latest history positions are recorded. Whenever a snaxel moves, its new position is entered into the list and compared. If it is at least two pixels away from any of the history posi-

[2] In [3, 6], extra snaxels are introduced by resampling the snake. Either way would compromise the purpose of the internal energy. This is discussed in detail in a forthcoming paper.

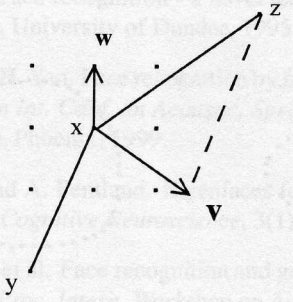



Figure 5: Converting forces into potential energy.

tions, it is considered active. After all snaxels are considered not active for 3 or 4 rounds, the current snake state is considered to be the final solution.

As mentioned before, choosing parameters is a difficult job for a balloon model. In our experiments, the best sets of parameters are chosen by trial and error. Some good guidelines for choosing parameters can be found in [6, 3]. For the self-adjusting mechanism, the threshold for influence measure difference is set to 0.5. The scaling factor for the influence force $\kappa$ is chosen to be of the same order as those for other forces, such as the balloon force.

The first experiment uses a synthetic line-drawing image (Fig. 7 (a)). Due to the high level of noise, using a constant balloon force produces unsatisfactory results. Two typical runs are shown in Figs. 7 (c) and (d). The result produced by our method is shown in Fig. 7 (b). Aided by the extra stiffness force from the self-adjusting mechanism, we can choose a larger balloon force to deal with noise edges without the fear of pushing the snake out of range. Figs. 7 (e) and (f) illustrate how the new model copes with big gaps on boundaries. Figs. 7 (g) and (h) show an example which is used to test the robustness of the model against bad initialization. The second experiment uses a cup image with fuzzy boundaries. Figs. 6 (a) and (b) show the image with an initialization and our solution respectively. Fig. 6 (c) is an intermediate result of the original balloon model. Note how the "peeling effect" takes place and results in a worse result (Fig. 6 (d)). The algorithm has been run on many synthetic and real images and the results are promising.

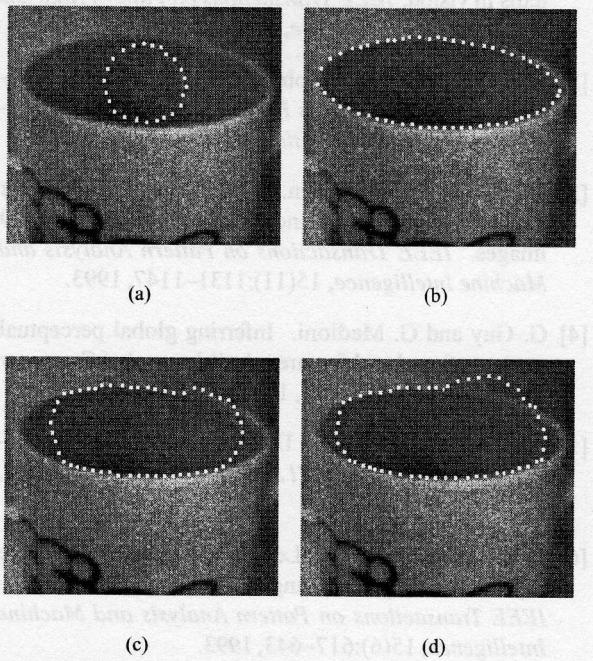



Figure 6: A cup image example: (a) An initialization; (b) Result by our method; (c) & (d) Results by the original balloons.

## 6 Conclusions

The self-adjusting mechanism presented in this paper reflects the way people extract boundaries: First locate obvious boundaries, then use the shape and position of these parts to help locate other parts. The new method copes with the "peeling effect" we observed in using the balloon model, and is designed to work with noisy images. The influence measures are computed iteratively, which is considered reliable because information propagated from distant snaxels are combined in the process. Experiments show that a few isolated noise edges have little effect on neighbors' measures. The general ideas embodied in this method are similar to those behind the snake growing model [2] and the ziplock snake [7]. However, our model is dynamic, and the "growing" or "zipping" happens at multiple places simultaneously, not limited to the two end points as in [2, 7]. Since there is no *a priori* shape information involved, our method is not able to recover noise-immersed corners. In that case, a second-step model-based method should be used to refine the results. The method is designed to work with the backtracking snake we presented in a previous paper [9]. Both methods are part of a larger project which aims to produce an aggressively searching snake model.

## Acknowledgements

This research is supported in part by the Canadian Natural Sciences and Engineering Research Council under Grant OGP9198.

## References

[1] A. A. Amini, T. E. Weymouth, and R. C. Jain. Using dynamic programming for solving variational prob-

lems in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(9):855–867, 1990.

[2] M.-O. Berger and R. Mohr. Towards autonomy in active contour models. In *Proceedings of Int'l Conference on Pattern Recognition*, pages 847–851, 1990.

[3] L. D. Cohen and I. Cohen. Finite-element methods for active contour models and balloons for 2-D and 3-D images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11):1131–1147, 1993.

[4] G. Guy and G. Medioni. Inferring global perceptual contours from local features. *Int'l Journal of Computer Vision*, 20(1/2):113–133, 1996.

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

[6] 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):617–643, 1993.

[7] W. M. Neuenschwander, P. Fua, L. Iverson, G. Szekely, and O. Kubler. Ziplock snakes. *Int'l Journal of Computer Vision*, 25(3):191–201, 1997.

[8] A. Shaashua and S. Ullman. Structural saliency: The detection of globally salient structures using a locally connected network. In *Proceedings of Int'l Conference on Computer Vision*, pages 321–327, 1988.

[9] J. Wang, X. Li, and A. Bradley. Boundary searching snakes for segmenting noisy images. In *Proceedings of Vision Interface '98*, pages 107–114, 1998.

[10] D. J. Williams and M. Shah. A fast algorithm for active contours. In *Proceedings of Int'l Conference on Computer Vision*, pages 592–595, 1990.

[11] G. Xu, E. Segawa, and S. Tsuji. Robust active contours with insensitive parameters. *Pattern Recognition*, 27(7):879–884, 1994.

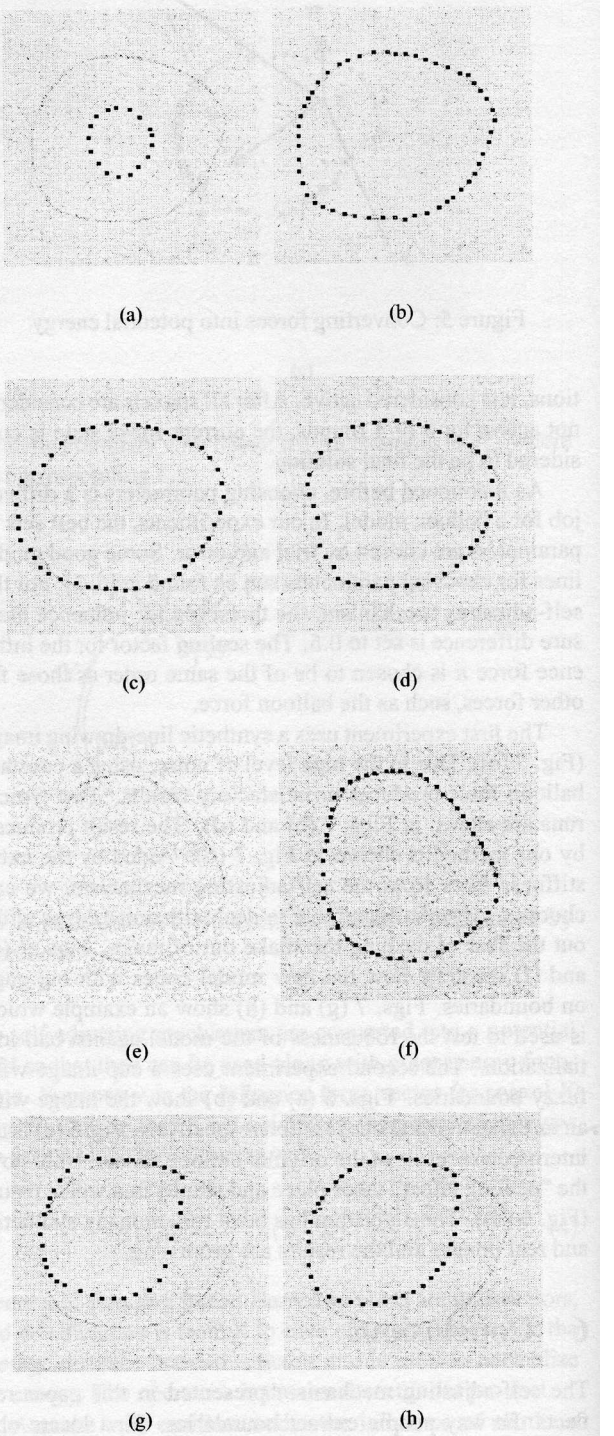



Figure 7: (a) A synthetic image; (b) Result by out method; (c)-(d) Results by the original balloons; (e)-(f) The self-adjusting mechanism can cope with contours with gaps; (g)-(h) The new method can handle bad initializations.

[9] H. Spies. Face recognition - a novel technique. Master's thesis, University of Dundee, 1995.

[10] T. Tan and H. Yan. Face recognition by fractal transformations. In *Int. Conf. on Acoustic, Speech and Signal Processing*, Phoenix, 1999.

[11] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

[12] L. Wiskott et al. Face recognition and gender determination. In *Proc. Intern. Workshop on Automatic Face- and Gesture-Recognition*, Zürich, 1995.