AUTOMATING SNAKES FOR MULTIPLE OBJECTS DETECTION

Baidya Nath Saha, Nilanjan Ray and Hong Zhang
Department of Computing Science, University of Alberta, Edmonton, Canada
{baidya, nray1, hzhang} @ualberta.ca

ABSTRACT

We propose a novel snake initialization as well as validation technique to automate snake/active contour for multiple objects detection. We apply a probabilistic quad tree based approximate segmentation to find the regions of interest (ROI) in an image, evolve snakes within ROIs and finally classify the snakes into object and non-object classes using boosting. We propose a novel loss function for boosting that is more robust to outliers and we derive a modified Adaboost algorithm by minimizing the proposed loss function to achieve better classification results. Extensive experiments have been carried out on two datasets: one has importance in oil sand mining industry and the other one is significant in bio-medical engineering. Performances of both proposed snake initialization and validation have been compared with competitive methods. Results show that proposed algorithm is computationally less expensive and can delineate objects up to 30% more accurately as well as precisely.

Index Terms— snake, active contour, object segmentation, object detection, boosting.

1. INTRODUCTION

Snake/active contour [4] has made its recognition as an interactive image segmentation tool for the last two decades. However, it is yet to be seen as a completely automated segmentation tool. Snake algorithms consist of three sequential steps: snake initialization, snake evolution and snake validation. For multiple objects detection, seeds are chosen inside the objects at the initialization step, then snakes are evolved from those seed points, and finally the evolved snakes are passed through a validation procedure to examine whether the snakes delineate the desired objects [11]. Substantial endeavors have taken place on the initialization and evolution steps towards snake automation. Most of the existing initialization algorithms [2] exploit the local maxima or other characteristics of the external energy that help to generate seed points within the objects. However, clutters in the noisy and poorly illuminated images generate considerable amount of seed points and snakes evolved from those seeds do not converge to the object boundaries. This necessitates a good validation scheme after snake evolution. Unfortunately, the validation step has not received much attention till date.

Saha *et al*. [11] proposed a snake validation scheme using principal component analysis (PCA). Their method places seeds blindly on the entire image and evolve one snake from each seed. When all snakes converge, a pattern image (an annular band) is formed across each snake contour. Each pattern image is then projected into an already trained PC (principal component) space and PCA reconstruction error is computed. The snakes associated with lower reconstruction errors than a threshold are considered as objects. Pattern images bear information regarding bright-to-dark (or *vice-versa*) transition across the object contours and show good discrimination capability between object and non-object classes. This validation technique is effective when the gradient strength of object boundaries is considerably high. Besides, throwing a large number of seeds blindly over an entire image might not be feasible for some applications, since the snake evolution can be computationally expensive. Thus, a handful of crucial seed points are always helpful.

In this paper, we propose a probabilistic quad tree based snake initialization scheme (QT) which is computationally inexpensive. QT automatically seeks ROIs from an image where the probabilities of locating objects are very high. We throw seeds only within ROIs and evolve a modified Gradient Vector Flow (GVF) snake [10] from each seed. Then we validate each evolved snake to verify whether they belong to the object or the non-object class. During validation, each snake is passed through a strong classifier formed by Adaboost [3]. We classify snake contours into objects and non-objects based on a set of features and we apply Adaboost for selecting important features. Here, it is noted that one shortcoming of the exponential loss function associated with Adaboost algorithm is that the penalty increases exponentially on negative margins incurring high misclassification error rates due to outliers [3]. We propose a novel loss function that imposes smaller penalties on negative margins, and thus make Adaboost more robust to outliers. We exploit the advantages of multiple features including region, edge and shape over PCA-based intensity feature proposed earlier [11].

Our proposed initialization and validation can be successfully used as "plugins" with any existing snake evolution techniques. We have carried out experiments on two real datasets: (a) oil sand mining images [10]: analyzing these images help to improve the performance of oil sand extraction process and (b) leukocyte images [1]: processing these images help in the study of inflammation as well as in

the design of anti/pro– inflammatory drugs. Results illustrate that our proposed algorithm is faster, more reliable and robust than competitive methods.

2. QUAD TREE BASED SNAKE INITIALIZATION

Quad tree (QT) [6] based segmentation algorithm receives an image as an input, and then divides it into four adjacent, non-overlapping quadrants if it meets pre-specified criteria, subsequently each quadrant is divided similarly and the process proceeds iteratively until it fails the pre-defined criteria. Consequently, the algorithm locates objects by smaller rectangular boxes. In our application here, at each iteration, the QT algorithm computes a posterior probability and splits the current region into four quadrants if the value of the posterior probability is less than a predetermined threshold. We locate objects by finding homogeneous regions based on local brightness and texture properties. We compute the posterior probability of a region (O) being object/non-object: $P(O/T, B) \alpha P(T/O) P(B/O) P(O)$,

where $P(O)$ is the prior probability. $P(T/O)$ and $P(B/O)$ are the likelihood of the region regarding texture and brightness respectively. Proposed probabilistic QT algorithm converges faster and delineates objects more accurately than deterministic QT algorithm if a suitable, application specific prior can be chosen. We compute texture energy (T) by the response of Gabor filters [6] and brightness (B) by the maximum singular value decomposition (SVD) [7] of the region. Maximum SVD encodes average brightness and Gabor filter response represents discriminative texture information for the objects. The computational complexity of QT is $O(r)$, where size of the image is $2^r \times 2^r$.

3. SNAKE VALIDATION USING BOOSTING

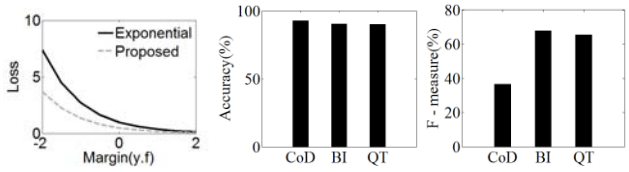
We compute different features for each converged snake contour, such as, contour shape features (form factor, convexity, extent, modification ratio [9] etc.), regional features (intra and inter class variance, entropy etc.), and edge based features (GICOV [1], gradient strength etc.) for snake validation. We use Adaboost for selecting important features. At the training phase, boosting picks only important features for snake validation from a host of features computed on training snake contours and finds the weights associated with those features. We place seeds blindly over the training images and evolve one snake from each seed and manually classify the snakes as objects that converge at object contours found on the ground truth made by the experts; otherwise consider the snakes as non-objects and thus form a training set. The Adaboost algorithm forms a strong classifier by combining the outputs of a set of weak learners linearly in an iterative manner [3]. We use decision stump (threshold) [3] as a weak classifier. A decision stump is a single level decision tree. Decision stump, $G_j(x)$ for feature f_j is defined as, $G_j(x) = 1$ if $f_j(x) > \theta_j$, otherwise, $G_j(x) = 0$, where θ_j is some feature value of f_j chosen as the threshold. Finding the best decision stump at each stage is similar to learning a node in a decision tree. We search over all possible features $f = [f_1, f_2, f_3, \ldots, f_n]$ and for each feature, we search over all possible thresholds θ induced by sorting the observed values of f and picking a value for θ that gives lowest misclassification error during training. At the test phase, proposed QT algorithm, discussed in section 2, locates ROIs over the test images where the probability of localizing objects is greater than a predetermined threshold. We place seeds only within ROIs and grow one snake from each seed. When all snakes are fully converged, we compute the values of the important features for each snake and multiply them with the weights associated with the features chosen by boosting during training phase and subsequently add them to form a strong classifier,

$f(x) = sign\left(\sum_{m=1}^{M} \alpha_m G_m(x)\right)$, where, α_m is the weight associated with the weak classifier $G_m(x)$. If the sign of $G(x)$ for a snake contour is positive then it is classified into object class, otherwise it is classified into non-object class. For classification, Adaboost minimizes an exponential loss function: $L(y, f(x)) = \exp(-yf(x))$, where y is the response and f is the prediction. The drawback of this exponential loss function is that it incurs substantial misclassification error rate as the penalty increases exponentially for large increasing negative margin due to outliers [3]. To address this problem, we propose a novel loss function:

$L(y, f(x)) = \exp(-yf(x) + \lambda|y - G(x)|)$, where $\lambda < 0$ and $G(x)$ is the prediction of the weak classifier chosen at the current stage. We have incorporated one extra term in the existing exponential loss function. At any boosting iteration, the proposed loss function is the same as the existing loss function if the misclassification error rate at current stage is 0 (proposed term vanishes when $y = G(x)$). The only difference between the proposed and the exponential loss function is that the penalty associated with the proposed loss function is less than that of the exponential one, if there is misclassification at the current stage (shown in Fig. 1(a) where loss is plotted against a function of the classification margin $y.f$). This modification leads to a low misclassification error rate and it becomes more robust to outliers. One additional advantage of this proposed loss function is that the user can adjust the amount of penalty for negative margins after observing the classifier performance over a validation data set. Accordingly, we determine the value of λ through cross validation (λ is a function of k shown in the appendix and the value of k is determined experimentally). We derive a modified Adaboost algorithm by minimizing the proposed loss function (The derivation is shown in Appendix).

Our modified Adaboost finds the feature weight, $\alpha_m = \log\left(k(1 - err_m)/err_m\right), k \geq 1$ where, for the existing Adaboost algorithm the value of k is always 1. This leads to

the weights associated with misclassified observations at any stage k times as much as the existing Adaboost (derivation is shown in the appendix). The value of k for our modified Adaboost is determined by cross-validation and is discussed in the next section.



(a) (b) (c)

Fig. 1: (a) Loss functions for two class classification. (b) Accuracy and (c) F-measure for three different snake initialization methods.

4. RESULTS AND DISCUSSIONS

We have carried out experiments on two data sets: oil sand images and leukocyte microscopy images.

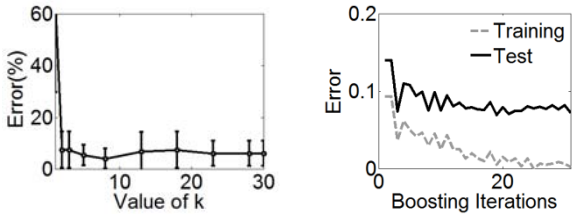
First, we construct training set using 20 images and test set using 100 images sampled randomly from a video of oil sand particles over a conveyor belt. For QT-based snake initialization, we find the distribution for prior and likelihood as well as the threshold value of the posterior probability (P_{th}) experimentally from the training set. A region will have high oil sand particles density if $P(O/T,B) \geq P_{th}$.

Datasets	*# of objects*	*# of Seeds generated by*		
		Proposed QT	*CoD*	*BI*
Oil Sand	349	686	3786	3000
Leukocyte	193	799	2402	4375

Table 1: Comparison among three snake initialization techniques.

Regions of Interest (ROI) generated by QT and seeds generated by Center of Divergence (CoD) [2] method are shown in Fig. 3. Table 1 illustrates the number of seeds generated by the proposed QT, CoD and blind initialization (BI) [11]. CoD refers to the local maxima of the external Gradient Vector Flow (GVF) field. The point from which the GVF vectors to all of its neighboring pixels radiate is considered as CoD. CoD is supposed to be located within the object and the snake evolved from CoD converges to the actual boundary of the object in noise-free settings. Fig. 1(b) and 1(c) respectively show accuracy and F-measure for CoD, BI and QT techniques with the proposed modified Adaboost-based validation. F-measure combines both recall and precision into a single entity [8]. Results show that though all techniques possess similar accuracy, both BI and QT achieve 30% more F-measure value than that of CoD but QT generates significantly fewer seeds (Table 1).

Next, we determine the value of k (discussed regarding feature weight in section 3) using five-fold cross validation [3] technique[3]. We compute misclassification errors for different values of k. They are shown in Fig. 2(a). Standard error bars indicate the standard errors of the individual misclassification error rates for each of the five parts. It is observed that both the average misclassification error rate and standard error is minimum for $k = 8$ for oil sand images. For existing Adaboost algorithm, the value of k is always 1. Modified Adaboost always outperforms the existing Adaboost algorithm because modified one can select the best value of k for which the misclassification error is minimum. The misclassification error rate for boosting with decision stumps [3], as a function of the number of iterations for $k = 8$ is shown in Fig. 2(b).



(*a*) *Misclassification Error* (*b*) $k = 8$

Fig. 2: (a) fivefold cross validation curve with standard error bars; the curve has minima at $k = 8$. (b) Misclassification error rate over the number of iterations for oil sand images.

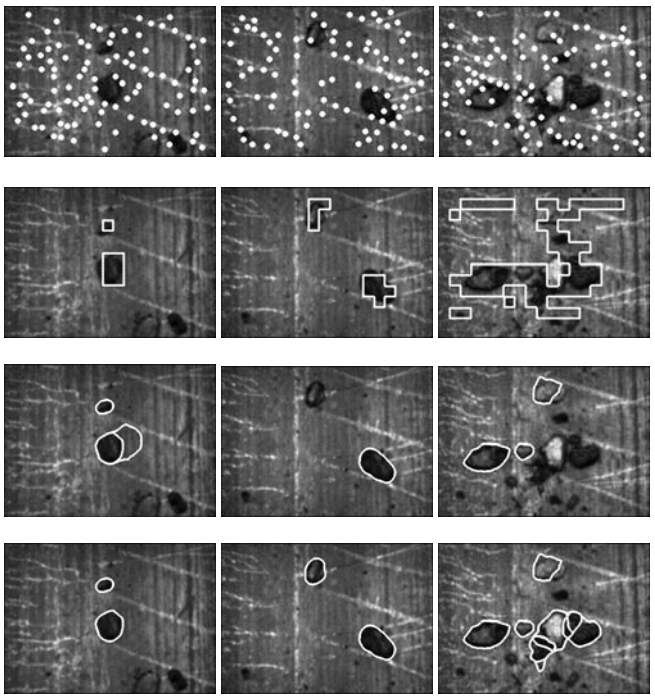


Fig. 3: Top row: results of COD; Second row (from above): results of proposed QT based initialization algorithm; Third row (from above): Results of PCA based validation technique; Bottom row: Results of proposed technique on oil sand images.

Fig. 3 shows the results of both Adaboost and PCA on oil sand images and their comparisons are shown in Fig. 5(a).

Finally, we have carried out experiment on a training set of 5 and a test set of 25 leukocyte images. Detections obtained by Adaboost and PCA techniques are shown in Fig. 4 and their comparisons are shown in Fig. 5(b).

One can interpret that Adaboost based validation is better than PCA based technique since it can detect more oil sand particles and leukocytes accurately and precisely.

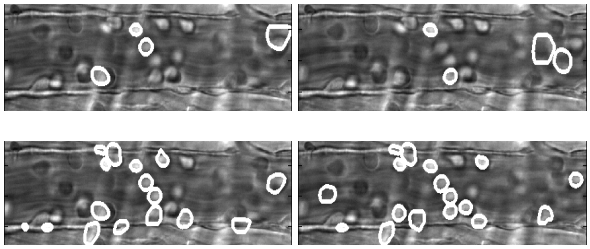
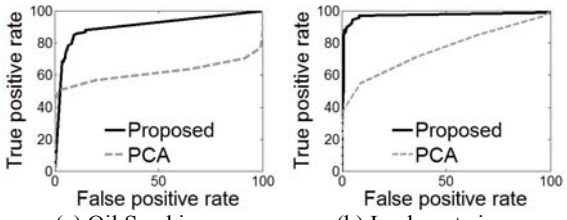


Fig. 4: Top row: Results of PCA based validation technique; Bottom row: Results of proposed technique on leukocyte images.



(a) Oil Sand images (b) Leukocyte images

Fig. 5: Receiver Operating Characteristic (ROC) curves.

5. CONCLUSION AND FUTURE WORK

Towards complete automation of snake algorithm, we have proposed a novel initialization and validation algorithm that could be utilized as a successful "plug-in" for snake/active contour tools. Existing research mainly focuses on the snake initialization and evolution steps and ignores the validation step. Here, we emphasize that we cannot omit the validation step in spite of applying the smart initialization technique of snake algorithm used for multiple objects detection. We have proposed probabilistic quad tree based approximate segmentation for snake initialization. We show that our proposed initialization outperforms existing initialization methods. We are looking ahead to improving this method by early sensing for active contour to avoid full convergence and combining initialization, evolution and validation step in a logical way to further speed up the whole algorithm.

Acknowledgements: This work is supported by NSERC, iCORE, Syncrude Canada and University of Alberta.

6. APPENDIX

Derivation of Proposed Discrete Adaboost Algorithm

Proposed loss function is :

$$L(y, f(x)) = \exp(-yf(x) + \lambda \mid y - G(x) \mid), \lambda < 0.$$

Let $f_m(x) = f_{m-1}(x) + \beta_m G_m(x)$ be the strong classifier composed of first m classifiers. We can pose m-th iteration of adaboost as the following optimization,

$$(\beta_m, G_m) =$$

$$\arg\min_{\beta, G} \sum_{i=1}^{N} \exp[-y_i (f_{m-1}(x_i) + \beta G(x_i)) + \lambda \mid y_i - G(x_i) \mid]$$

$$\Rightarrow (\beta_m, G_m) = \arg\min_{\beta, G} \sum_{i=1}^{N} w_i^m \exp[-y_i \beta G(x_i) + \lambda \mid y_i - G(x_i) \mid]$$

where, $w_i^m = \exp(-y_i f_{m-1}(x_i))$ is free of both β and $G(x)$.

$$\Rightarrow (\beta_m, G_m) = \arg\min_{\beta, G} [\exp(-\beta) \sum_{i: y_i = G(x_i)} w_i^m + \exp(\beta + 2\lambda) \sum_{i: y_i \neq G(x_i)} w_i^m]$$

$$= \arg\min_{\beta, G} [(\exp(\beta + 2\lambda) - \exp(-\beta)) \sum_{i: y_i \neq G(x_i)} w_i^m + \exp(-\beta) \sum_{i=1}^{N} w_i^m].$$

The solution for β_m and G_m can be obtained in two steps. First, for any value of $\beta > 0$, the solution for G_m is:

$$G_m = \arg\min_{G} \sum_{i=1}^{N} w_i^m I(y_i \neq G(x_i)) / \sum_{i=1}^{N} w_i^m.$$

Let $err_m = \sum_{i=1}^{N} w_i^m I(y_i \neq G(x_i)) / \sum_{i=1}^{N} w_i^m$, then

$$\beta_m = \frac{\partial}{\partial \beta} (\sum_{i=1}^{N} w_i^m ((\exp(\beta + 2\lambda) - \exp(-\beta)) err_m + \exp(-\beta))) = 0$$

$$\Rightarrow \beta_m = \frac{1}{2} (\log \frac{1 - err_m}{err_m}) - \lambda = \frac{1}{2} (\log k \frac{1 - err_m}{err_m}), \quad \text{where,}$$

$\lambda = -0.5 \log(k), \; k \geq 1.$ Now, $w_i^{m+1} = w_i^m \exp(-\beta_m y_i G_m(x_i))$.

Using the fact that $-y_i G_m(x_i) = 2I(y_i \neq G(x_i)) - 1,$ we get,

$w_i^{m+1} = w_i^m \exp(\alpha_m I(y_i \neq G(x_i))) \exp(-\beta_m)$ Where,

$\alpha_m = 2\beta_m = \log(k / ((1 - err_m) / err_m))$.

So, $w_i^{m+1} = w_i^m \exp(\alpha_m I(y_i \neq G(x_i)))$. The factor $\exp(-\beta_m)$ multiplies all weights by the same value, so it has no effect.

7. REFERENCES

[1] G. Dong, N. Ray and S. T. Acton, "Intravital leukocyte detection using the gradient inverse coefficient of variation", *IEEE Trans. on Med.l Im.*, Vol. 24, pp. 910-924, July 2005.

[2] X. Ge and J. Tian, "An automatic active contour model for multiple objects", *ICPR*, vol. 2, pp. 881-884, 2002.

[3] T. Hastie, R. Tibshirani and J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Springer, Second Edition, 2009.

[4] M. Kass, A. Witkins, and Terzopoulos, "Snakes: active contour models", *IJCV*, vol.1, No.4, pp.321-331, 1987.

[5] J. S. Maritz. *Distribution-Free Statistical Methods*, Chapman & Hall, Second Edition, 1995.

[6] M. Mirmehdi, X. Xie and J. Suri. Handbook of Texture Analysis, Imperial college Press, 2008.

[7] D. Omerčević, R. Perko, A. T. Targhi, Jan-Olof Eklundh and A. Leonardis, "Vegetation segmentation for boosting performance of MSER feature detector", *Computer Vision Winter Workshop*, Feb 4-6, 2008.

[8] C. J. van Rijsbergen. Information Retireval. Butterworths, London, 1979.

[9] J. C. Russ. The Image Processing Handbook. Third Edition, CRC & IEEE press.

[10] B. N. Saha, N. Ray and H. Zhang, "Computing oil sand particle size distribution by snake-PCA algorithm", *ICASSP*, pp. 977-980, April, 2008.

[11] B. N. Saha, N. Ray, and H. Zhang, "Snake validation: A PCA-based outlier detection method", *IEEE Signal Processing Letters*, vol. 16, issue 6, pp. 549-552, 2009.