Chapter 5

Results

This chapter is devoted to explain and discuss the experiments associated with the contributions described in chapters three and four.

5.1 Supervised Clustering Competition Scheme results

Supervised Clustering Competition Scheme is a suitable tool for addressing several fields in classification. In particular, this section shows the experiments and results of the method when casted in a semi-supervised clustering framework and in particularization problems, as illustration of its potential.

5.1.1 Semi-supervised results: The writers test

Illustrating the benefits of our SCCS on standard databases, in some cases, is not directly possible as they are designed mainly to compare supervised pattern recognition algorithms. Still we use data from standard databases to simulate real applications of semi-supervision when it is supposed to be available small amount of correctly labelled and large amount of unlabelled data. To this purpose, we tested our method as a semi-supervised classification scheme in a test of writers characters. We used the MNIST digits database with images of 128×128 pixels. The feature extraction process was a 16×16 zoning procedure on the 64×64 central values of the image. The number of white pixels is counted and stored as a feature. Therefore we have a 16 dimensional feature space. The training set is composed by 100 different hand-written sets of 120 digits each one (The first 100 writers of the HSF7). Each set corresponds to a different writer and contains about 30 symbols per character. The test set contains 500 different hand-written sets of 120 digits each one (The first 500 writers of the HSF6). We center the experiment in a two-class classification process distinguishing number ONE from number TWO. From the set of features we have selected automatically two of the most discriminative ones using discriminant analysis.

Table 5.1 shows the results in the writers test. The first column shows the amount

Supervised Data	Unsupervised Data	Recognition Rate
0.02%	80%	89.69%
0.01%	0%	77.92%
0.01%	20%	75.62%
0.01%	50%	86.62%
0.01%	80%	86.57%

Table 5.1: Semi-supervised results for the writers test.

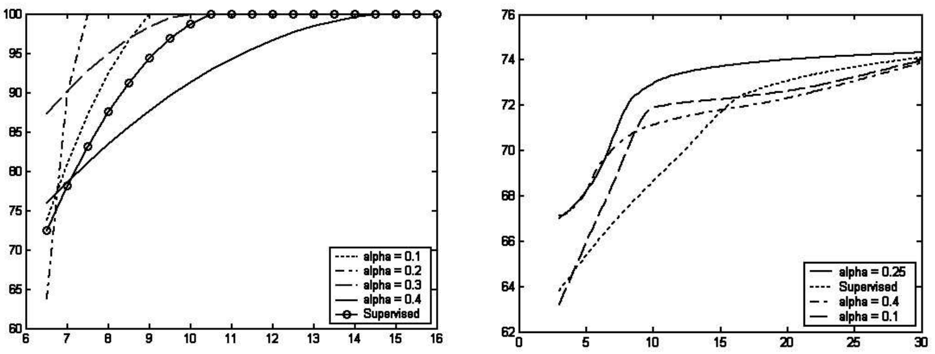
of supervised data used in the experiment. The second column displays the percentage of unsupervised data we have involved as unlabelled data. This table illustrates how the recognition rate varies as we add more unsupervised data to the process. Using a semi-supervised classification approach with 0.02% (about 40-50 samples) of supervised data and 80% of the training data as unlabelled data, we have obtained an error rate for the training set of 10.31%, that is 89.69% of the data has been correctly assigned to the classes. This semi-supervised clustering can be used to classify the writers test set with an overall accuracy of 87.65%. So both supervised and the semi-supervised process are quite comparable. With a 0.01% of supervised data (approx. 25 samples), we obtain a recognition rate of 77.92% that can be improved up to 86.57% using 80% of unsupervised data. As we add more data the recognition rate increases: 75.62% of recognition rate with 20% of the data and 86.62% with 50% of the data. Our experiment shows that at a given ration of unlabelled data, the process stops its improvement. In particular, adding more data to the process could be detrimental since more data implies a greater possibility of outliers and conflictive samples, that is if bad classified are added it would degrade the performance of the final classifier. Summarizing, using unlabelled data definitely improves the final outcome of supervised classification but the amount of unlabelled data cannot surpass given limits.

5.1.2 Semi-supervised results: UCI Database

Using the UCI database [140] we evaluated the algorithm using a 40%-60% split for test and training sets. The training set was supposed to be unlabelled with partial labelling. The supervised rate is the percentage of labelled data used from the training set.

Figure 5.1.a shows how the semi-supervised algorithms compares with the supervised one changing the α value. The graphics show the recognition rate as a function of amount of labelled data that has been considered. The figure displays the recognition rate and the percentage of labelled data used. We can see that with α around 0.4 the learning task with few data is worse than the supervised process. However, the slope of the recognition rate quickly increases as α decreases, achieving a maximum slope at $\alpha = 0.2$. For lower α the algorithm needs again a greater amount of data for good recognition rates.

Figure 5.1.b illustrates the same behavior in a more complex real data set, the Pima-Indians-Diabetes test. In this figure we observe that there is a range of labelled data percentage ($< 17\%$) in which our method outperforms classic supervision for



(a) (b)

Figure 5.1: Recognition rate for different alpha values. (a) IRIS database. (b) Pima-Diabetes database.

the different configurations of α ($0 < \alpha < 0.5$). In this sense, it is clear that for a small labelled data set this kind of semi-supervised methods indeed improves the performance of the classifier, though all these methods tend to converge to the same recognition rate as the number of samples in the labelled data set increases. We can also see that before convergence, with α around 0.25 our scheme is clearly superior to the recognition rates obtained using the supervised approach.

We follow these lines with several results using the UCI database. We chose the value $\alpha = 0.25$ as empirically it shows a good performance as a semi-supervised mixing value on different standard databases.

DB Name	Sup. Rate	Recog. Rate	DB Name	Sup. Rate	Recog. Rate
Balance-Scale	8%	86.12%	New-Thyroid	7%	60.10%
Balance-Scale	18%	94.49%	New-Thyroid	10%	91.63%
Glass	15%	81.00%	New-Thyroid	12%	93.72%
Glass	22.5%	84.12%	Wine 2D	4.55%	80.00%
Wine	16%	92.40%	Wine 2D	6.18%	88.36%
Wine	23%	98.11%	Wine 2D	9.15%	90.00%

Table 5.2: Semi-supervised results for UCI database $\alpha = 0.25$.

Table 5.2 shows the performance of our method in different UCI data sets. The second and the fifth columns show the percentage of data from the training set used as labelled data. Note the database name "wine 2D", this is the first two features subset of the "wine" data set. As we can observe, the resulting recognition rate is quite good for the low amount of data used in the data set. When analyzing the results of the process we must take into account that the process involves a supervised functional estimation, which needs at least a minimum amount of data to be significant. This must be taken into account when comparing the performance for really non-significant amount of labelled data.

The figures obtained using our approach are comparable to those present in the literature. However, this formulation has different advantages that makes it a perfect candidate technique for several problems. First of all, the imposition of a model allows to relax the necessity of having a perfect labelling or labelling of large amount of data, since the model is more permissive to labelling errors, even with low amount of data. This formulation allows us to impose a constraint as an approximation to a supervised classifier. This fact helps the clustering process to choose the label of the particular class of each sample. However, it also has an important drawback, since it relies on a model, it could need more support in terms of number of samples for the model to be reliable.

5.1.3 Particularization results: The writers test.

Experiment settings

In order to exemplify the behavior of the methodology we have chosen a full supervised classification scenario in which an OCR is used to distinguish between two hand written digits. In order to ensure structure in the data, we aim for the classification of data of a single writer at a time, in the same way as if the OCR was scanning and recognizing the digits in a single sheet of paper.

We have used the MNIST digits database with images of 128×128 pixels. The feature extraction process has been a 4×4 zoning procedure on the 64×64 central values of the image. The number of white pixels is counted and stored as a feature. Therefore we have a 16 dimensional feature space. The training set is composed by 100 different hand-written sets of 120 digits each one (The first 100 writers of the HSF7). Each set corresponds to a different writer. The test set is a 500 different hand-written sets of 120 digits each one (The first 500 writers of the HSF6). We center the experiment in a two-class classification process distinguishing number ONE from number TWO. From the set of features we have selected automatically two of the most discriminative ones using discriminant analysis.

Figure 5.2 shows the two class classification problem. The figure represents the feature space associated to the training set of the digits "two" and "one". As one can see, the training set is general and contains different instances of the digit "two" with different particularities. However, not always a general purpose classifier is needed. If our particular instance of the problem is a subset of the general set, we can focus on extracting some information of the test data, so we can constrain the training space. The asterisks are the test data for the digits "one" and "two" written by the same writer. The shadowed areas represents possible subsets of the training data which characterizes our test data. We can see that this subproblem has much more structure than the original one, and therefore we can use mixed approach for trying to solve it.

Figure 5.3 shows a representative set of hand-written digits. Figure 5.3.a shows the variability of the digit "two" written by different writers. Figure 5.3.b shows the variability of the digit "two" written by just one author.

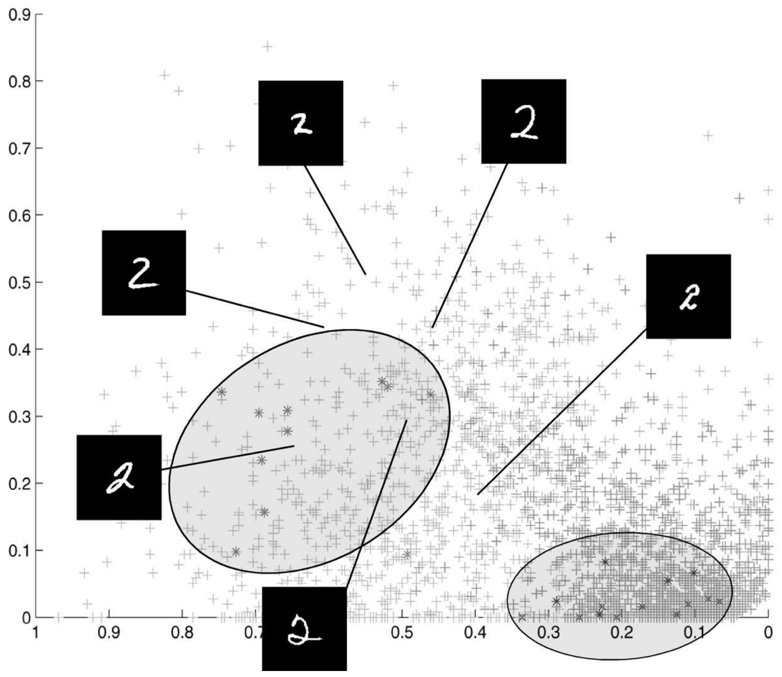


Figure 5.2: Feature space representation of the training set. The asterisks represents the test set. The shadow areas represents a possible partition of the training set that solves the two class problem.

Experiment results

As a ground-truth unsupervised classifier we have used Expectation-Maximization with K-Means initialization, and a supervised labelling for the cluster centers. This process yields a recognition rate of 89.35%.

Table 5.3 shows the behavior of the process for different fixed values of the parameter α. The parameter γ is set to adapt automatically. The second column refers to the recognition rate of the overall process. It can be clearly seen that a mixture of supervised and unsupervised improves the performance of the supervised classifier. We must take into account that when $\alpha \geq 0.5$ the process behaves as a pure supervised classifier, thus, we only show the results for α between 0 and 0.5. The third column (Overall Gain) represents the percentage of gain or degradation of the process as the difference between the error of the process and the error achieved using the supervised classifier. Positive values are gain in performance. Negative values are degradation of the process performance. Hence, for $\alpha = 0.1$ the process performs worse than the supervised classifier. This is particularly obvious in the sense that low values of α means a nearly no contribution of the supervised process, and therefore, the classification is made with the unsupervised process alone. The fourth column shows the mean percentage of gain per sample. That is, the percentage of improvement per sample expected for the process to classify correctly at each experiment.

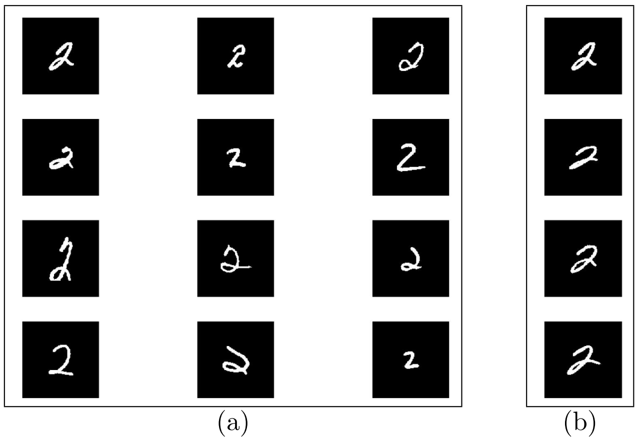


Figure 5.3: (a) Set of digit "two" written by different authors. (b) Set of digit "two" written by the same writer

Mixing value	Recognition Rate	Overall Gain	Mean Sample Gain
0	87.73%	0.08%	0.01%
0.05	87.16%	−0.49%	−0.06%
0.1	87.49%	−0.16%	−0.02%
0.15	87.89%	0.24%	0.03%
0.2	87.73%	0.08%	0.01%
0.25	88.14%	0.49%	0.06%
0.3	89.67%	2.02%	0.25%
0.35	90.65%	3.00%	0.37%
0.4	93.07%	5.42%	0.67%
0.45	92.59%	4.94%	0.61%
0.5	87.65%	0.00%	0.00%

Table 5.3: Comparative table for fixed α values.

Table 5.4 complements the former table with some other information of interest. The second column shows the percentage of experiments with gain/degradation respectively. The third column is the average gain and degradation of the cases when a gain or degradation is present. The last column is the maximum gain and degradation obtained in the set of experiments. This column shows that there are some outlier cases in the overall experiment that can be highly degraded or highly improved by the process.

As we have seen, we can obtain good discriminative power using the algorithm. However the difference in performance between the supervised classifier and the resulting mixed-classifier is over 40%, in the sense that they differ in more than 40% of the tests. One can aim to have a more "stable" classifier, so that the disagreement of both classifiers could be lower, and aim for a reduction of performance in degradation cases. To address both points we can meddle with the α-parameter. We can try to

Mixing value	Cases G/D	Mean G/D	Max G/D
0	26.25% / 22.44%	7.33% / 8.54%	28.00% / 28.00%
0.05	25.85% / 21.24%	7.38% / 9.25%	28.00% / 36.00%
0.1	25.65% / 21.64%	7.41% / 8.89%	28.00% / 36.00%
0.15	25.85% / 21.24%	7.38% / 8.83%	32.00% / 36.00%
0.2	25.85% / 21.24%	7.41% / 8.89%	32.00% / 36.00%
0.25	25.65% / 20.84%	7.44% / 8.88%	32.00% / 36.00%
0.3	26.45% / 21.04%	7.48% / 8.23%	32.00% / 28.00%
0.35	26.65% / 21.04%	7.43% / 7.66%	32.00% / 20.00%
0.4	27.86% / 18.84%	7.25% / 7.15%	28.00% / 20.00%
0.45	25.85% / 15.43%	5.77% / 5.71%	20.00% / 16.00%
0.5	0.00% / 0.00%	0.00% / 0.00%	0.00% / 0.00%

Table 5.4: Comparative table for fixed α values.

make a test dependent α for each experiment instead of maintaining a fixed α. In this way, the algorithm can choose to resemble a supervised or unsupervised classifier depending on the disposition of the test samples and the difficulty to classify them.

We propose two approaches. The first one consists of defining a parameter dependent on an estimated complexity of the discrimination of both classes. This can be done using for instance the *Bhattacharyya distance*. This distance is defined as follows:

$$B = -ln[\int_{-\infty}^{\infty} \sqrt{p_1(x)p_2(x)dx}]$$

However, in order to obtain a closed formula we can particularize this measure assuming that the *pdf* of both classes are gaussian. Under such hypothesis the *Bhattacharyya distance* can be written as:

$$B = \frac{1}{8}(m_1 - m_2)^T \left(\frac{\Sigma_1 + \Sigma_2}{2}\right)^{-1} (m_1 - m_2) + \frac{1}{2} ln \frac{|(\Sigma_1 + \Sigma_2)/2|}{|\Sigma_1|^{1/2}|\Sigma_2|^{1/2}}$$

This value is lower as both classes become closer. As we want to keep our algorithm with a conservative behavior, we precisely want α to be higher if both classes seem to be very entangled. The first thing to do is to find a first simple classification to define both classes *a priori*. We have chosen a simple Bayesian classifier with a density estimation based on a gaussian function. Over this classification we can find the distance B. However, we need to map this distance in a limited range between $\{0 \dots 0.5\}$. Therefore, we will use a reshaping function to do so. In particular, we have chosen

$$\alpha = K_1 * e^{-B}$$

where K_1 is a normalization constant value (we set this parameter to value 4 for this experiment). However, this parameter gives a theoretic approximation of how entangled are both distributions, based on a simple classification. But, we can further analyze the test set and infer more information to help us to set up the α parameter. We also propose a measure of the difficulty of the problem. To do so we can build a function related to the distance of a sample from the theoretical supervised classification borderline. This function can be expressed as:

$$U = \frac{1}{N}\sum_{i=1}^{N} |P(x_i|C_1) - P(x_i|C_2)|$$

where N is the number of test samples, and x_i is a test sample. In this case, again we can estimate the density using a mixture model and evaluate the function. The value U (*Uncertainty*) is higher as the number of samples in a "gray" area increases (that is, the number of samples near the border between both classes increases). Again, we need to adapt the α value to this measure. In this sense we will use this function:

$$\alpha = K_2 \frac{1}{U}$$

where K_2 is a normalization constant value ($K_2 = 10^{-4}$ for this example).

Method	R Rate	O Gain	M S Gain
B	90.56%	2.91%	0.36%
B + U	90.24%	2.59%	0.32%

Table 5.5: Comparative table for variable α values.

Method	Cases G/D	Mean G/D	Max G/D
B	16.43% / 10.82%	6.78% / 6.96%	32.00% / 20.00%
B + U	11.42% / 6.21%	5.54% / 5.03%	20.00% / 12.00%

Table 5.6: Comparative table for variable α values.

Using these methods we obtain the results in tables 5.5 and 5.6. As we can see, the disagreement percentage has been reduced dramatically from 40% to barely 26% using just the *Bhattacharyya measure* while maintaining a good discrimination rate. This effect is further improved with barely no changes in recognition rate using a combination of both methods Bhattacharyya and Uncertainty. In both cases the recognition rate improves nearly a 3%.

5.2 Performance of parametric generative snakes

In order to prove the advantages of the generative snakes we ran it on images from different domains: synthetic images, natural scene images and medical (ultrasound) images. Even considering different domains, we implemented and tested our statistic snakes in the same framework, with the same parameters. To concretize our approach to a single feature space we have decided to use texture feature space as a reference.

5.2.1 General implementation issues

We have used images with different sizes ranging from 100 to 400 pixels in width and height. The co-occurrence matrix measures have been shown to attain the best performance with neighborhoods of 10 to 16 pixels, $G = 256$ and distances of 2 and 3. Analysis of the Fisher Linear Discriminant transformation matrix shows that the first three eigenvectors already contain 99% of the energy. Hence, we reduced the original 24-dimensional space provided by the 6 measures at angles $\{0^o, 45^o, 90^o, 135^o\}$ down to a 3-dimensional space without losing discriminability.

Texture modelling concerns with finding a model that can be expressed using a limited set of parameters and preserves strong discrimination power. The mixture model is composed of a sum of fundamental distributions, following the next equation

$$p_i(x|\Theta) = \sum_{k=1}^{C} p_k(x|\theta_k)P_k \tag{5.1}$$

where C is the number of mixture components, P_k is the *a priori* probability of the component k, and θ_k represents the unknown mixture parameters. In our case, we have chosen **gaussian mixture models** $\theta_k = \{P_k, \mu_k, \sigma_k\}$ for each set of texture data we want to model. The mixture model is estimated using EM algorithm.

Figure 5.4 shows the modelling process. Fig.5.4.(a) shows the original image and target texture. We process this image computing the co-occurrence matrix measures. Fig. 5.4.(b) shows the response of one of these measures at a given orientation (Entropy at 45^o). These measures are reduced in dimensionality via FLDA. Fig. 5.4.(c) shows the reduction of the 24-dimensional data onto a 2-dimensional space. The target texture projected features of the pixels from the white square in fig.5.4.(a) are surrounded by a square in fig.5.4.(c) (The scale and relative distance between the sets of points are the ones shown in the figure). In the end, the mixture model is built to approximate the target distribution of the data (fig. 5.4.(d)).

The mixture model used is a compound of singular gaussian distributions. The number of gaussians used in our models ranges from 1 to 4. However, a small number of gaussians, up to two, in the mixture model yields a more regular and smoother likelihood map. The λ parameter of the ELM formulation is equal to 1 in all the examples shown in the work except for figure 5.7 using $\lambda = 0.5$.

An application of a regularization process on the ELM contours has been shown. Although GGVF is not able to absolutely overcome the weaknesses of the concave region problem, it has some desirable properties such as the extension of the attraction range to our features and the smoothness of the generated field.

Parameter μ in (4.10) is introduced to control the smoothing term. Decreasing this parameter we keep close to the original gradient and also increase stability of the numerical implementation. Also parameter K, forces equation 4.10 to coincide with ELM gradient near edges. In all tests, the ELM contours have been processed using a gradient vector flow scheme with $\mu = 0.2$ and $K = 0.2$ and snakes have been initialized at the border of the image.

5.2.2 Application to synthetic images

The method has been applied to a wide variety of texture images from the Brodatz database. It must be noted that co-occurrence matrices perform extremely well with these textured images. The images were of size 128x128 and 256x256 pixels. Different compositions have been done to evaluate the reliability of the boundaries of our

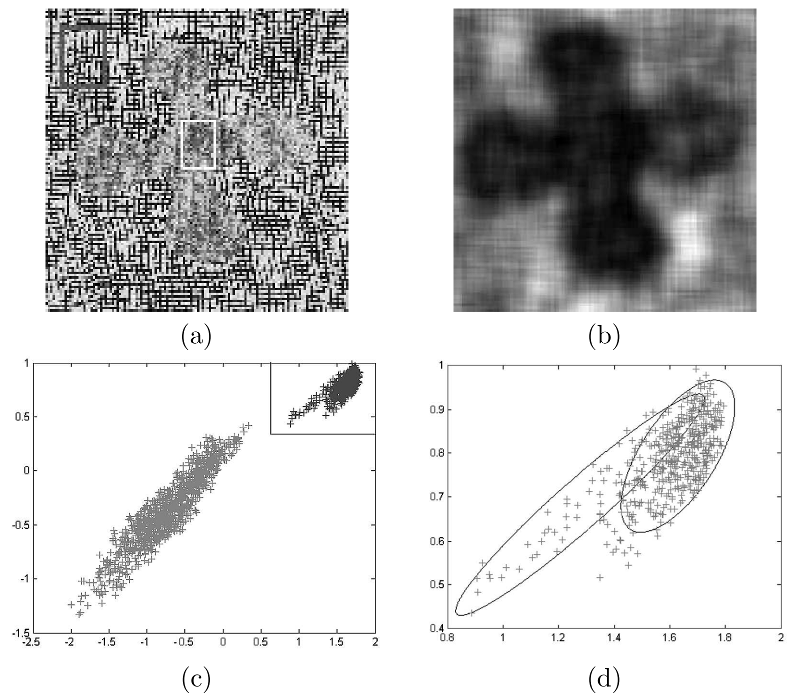


Figure 5.4: (a) Original image and selected texture. (b) One of the 24 responses of the co-occurrence matrix measures (this one corresponds to Entropy at 45^o) (c) Dimensionality reduction using FLDA. (d) Modelling of the target texture using 2 gaussian.

method. In this kind of images both classification methods as well as our statistical method performed excellently.

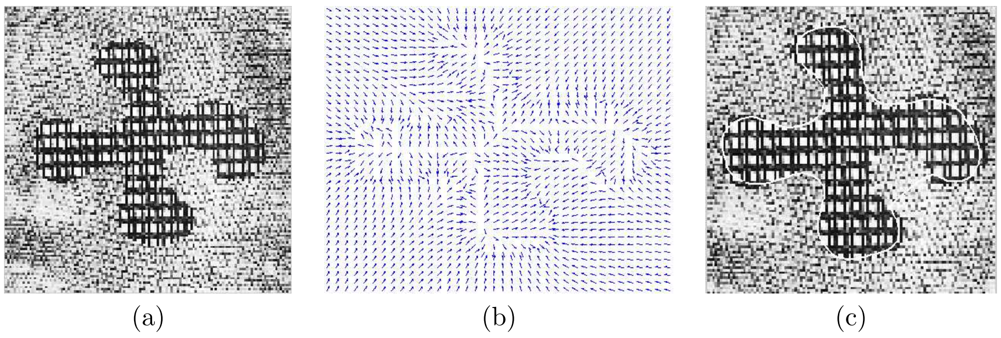


Figure 5.5: (a) Original synthetic image, (b) Vector flow field, (c) Resulting segmentation. [Parameters $\alpha = \beta = 0.7$, $K = 0.2$, $\mu = 0.2$]

Figure 5.5 shows the results using our approach on synthetic images. We have chosen this image because it is indicative of both the advantages and disadvantages of the method. Figure 5.5.(a) shows the original image and 5.5.(b) the regularized vector field. Notice that in fig. 5.5.(c) the segmentation has located the boundaries as we desired. However, the original GGVF has a drawback[80]: the incapability to adapt to almost closed concave region (as can be appreciated in figure 5.5.(c)). This shows that if the image does not have any closed concave regions there will not be any problem for the snake to adapt. The parameters for these images are $\alpha = \beta = 0.7$, $K = 0.2$, $\mu = 0.2$.

5.2.3 Application to natural scenes

Natural scenes have been chosen from an animal database. The method has been validated on 30 images of natural scenes. These images show combinations of different textures with a target region of interest. In natural scenes concave forms are frequent, an important issue when working with snakes.

The first test on natural scene images (fig.5.6) is the segmentation of a cheetah. In the image we distinguish clearly three different textured regions. We train our system with a sample of the cheetah texture and the background, and deform a snake from the frame leading to the shown result. The snake was able to find the whole cheetah body except a part of the head that is missing the texture pattern.

Figure 5.7 shows the whole process for the segmentation of a salamander on a natural scene image. Fig 5.7.(a) shows the original image. Fig. 5.7.(b) shows the likelihood map of the salamander. Fig. 5.7.(c) and (d) show the contours of the likelihood map and of the enhanced likelihood map for the salamander texture respectively. Fig 5.7.(e) shows the resulting vector field. And figure 5.7.(f) illustrates

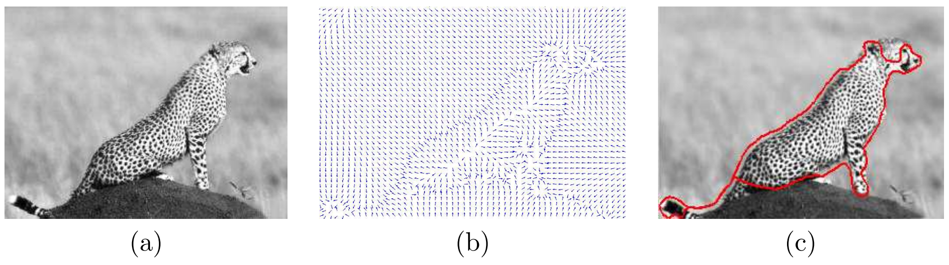


Figure 5.6: (a) Original cheetah image, (b) regularized vector field of SML, (c) Resulting segmentation.

the final result. The misclassification in the tail, observed in the figure is caused by the lack of convergence to the tail of the salamander due to the characteristics of the GGVF.

Figure 5.8 shows the results of our method applied to two different natural scenes. The first scene is a scene with a zebra; again, three different textures can be seen. In the result one can observe that a misclassification has been caused by the pattern of the background texture, which locally resembles the zebra pattern. The second scene is another salamander. In this scene minimal remarks must be made, the desired texture is the dotted body of the salamander, and the segmentation is quite well performed.

5.2.4 Application to medical imaging

We have applied our method to medical imaging, in particular to Intravascular Ultrasound Images[77]. The goal in this kind of images is different, we want to segment the lumen (blood) from the plaque (tissue).

Although lumen appears as a darker region, grey level alone is not enough to obtain a consistent segmentation, texture analysis has to be applied. The initial snake is aligned with the top line of the image. Note that all artifacts in the upper part of the image are successfully bypassed by the snake given that they are not characterized by the texture pattern of the tissue.

Figure 5.9 shows a segmentation of an IVUS. Fig. 5.9.(a) shows the original IVUS image as is captured by the IVUS machine. In fig. 5.9.(b) the IVUS image is transformed to cartesian coordinates for better performance of the texture descriptors. Figures 5.9.(c) and (d) illustrate our method, fig. 5.9.(c) shows the enhanced likelihood map; and fig. 5.9.(d) the final segmentation. In this image the contour we are looking for is not closed, it is just a boundary, so we will use an open parametric snake. Note that geodesic snakes are not able to deal with open contours, this fact illustrates our decision to show the performance of parametric snakes.

The results have been validated using 450 images from different patients. We have used images from 15 pullbacks of IVUS images, 30 images for each pullback.

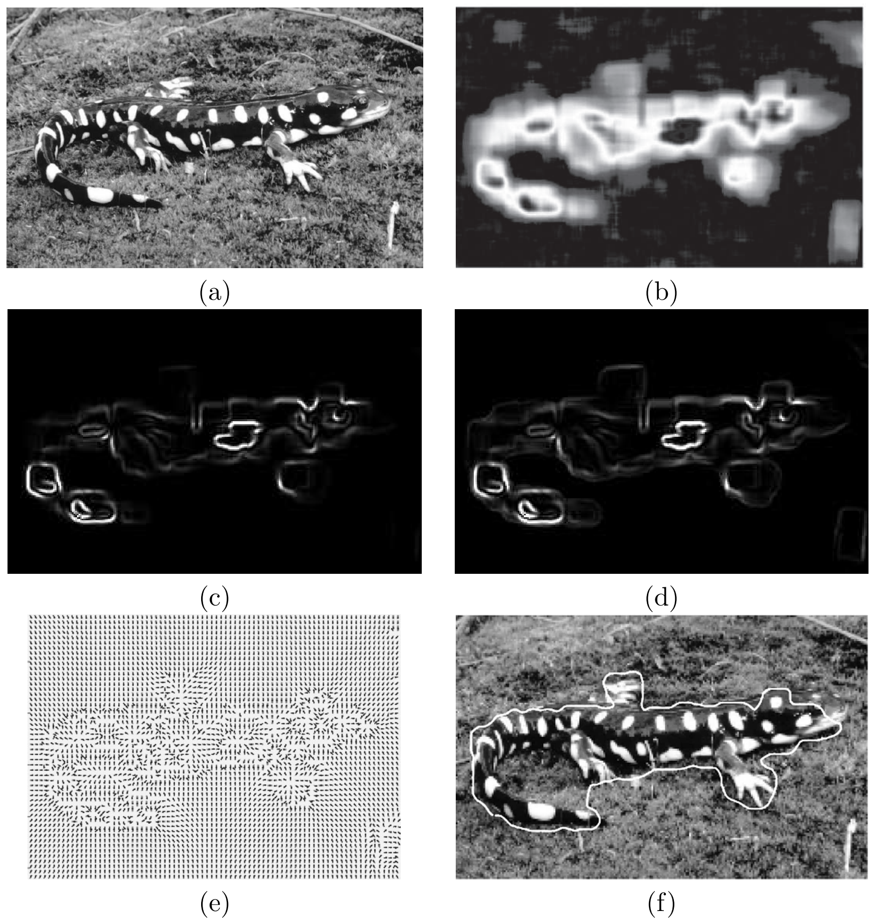


Figure 5.7: Segmentation of a natural scene: (a) Original image, (b) Likelihood Map of the salamander image, (c) Contours of the Likelihood Map (d) Contours of the enhanced likelihood map (e) Resulting vector field for the ELM, (f) Segmentation of the salamander.

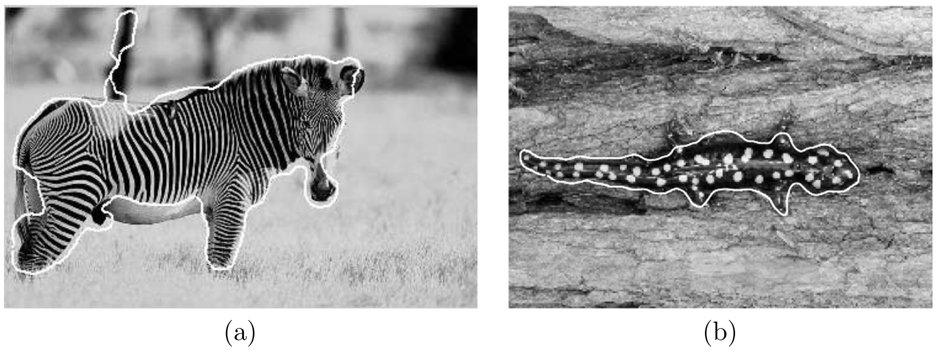


Figure 5.8: Segmentation of 2 natural scenes. (a) Segmentation of a zebra. (b) Segmentation of a salamander

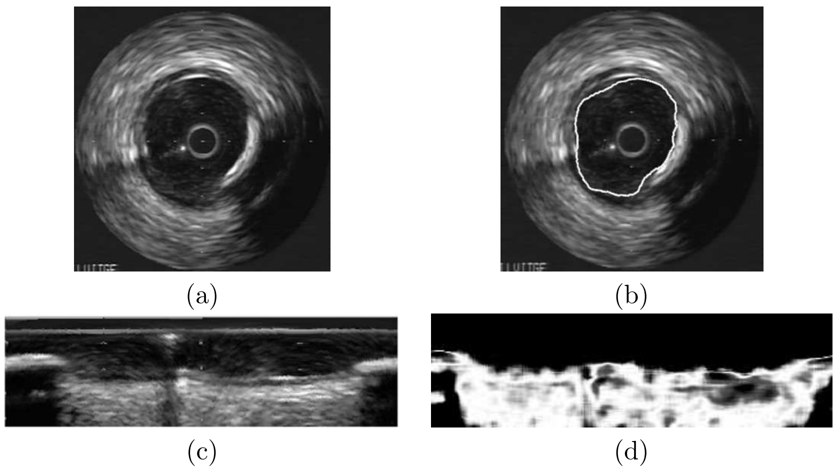


Figure 5.9: Segmentation of medical images.(a) Original IVUS image, (b) Location of the lumen-plaque border using the proposed method, (c) Original IVUS image in cartesian coordinates (d) enhanced likelihood map.

5.2.5 Discussion

The first issue to discuss is the utility of likelihood maps as an alternative to other classification schemes.

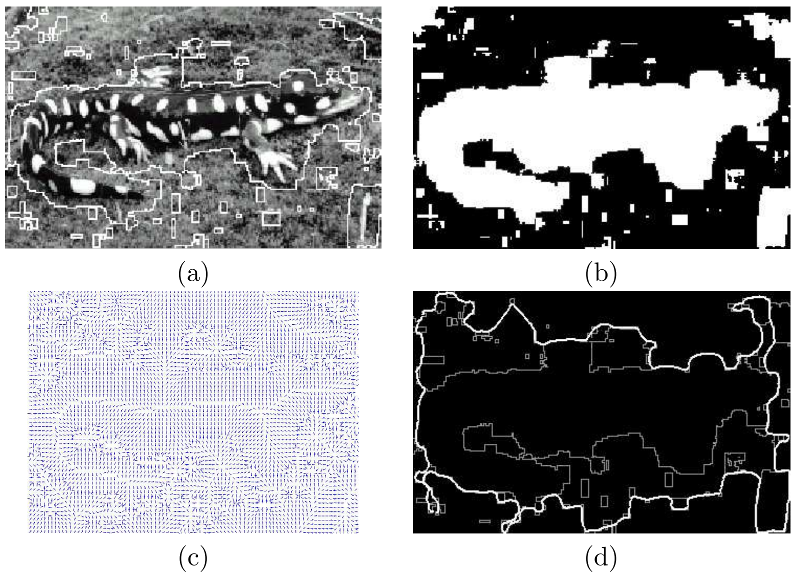


Figure 5.10: (a) Segmented salamander image using Mahalanobis distance, (b) Mask of the classification, (c) Vector field generated, (d) Resulting segmentation.

Usually, classification processes heavily rely on some kind of threshold or decision boundary, a trade-off of the probabilities involved in the classification process (i.e. Bayesian framework), distances from measures to the trained classes (i.e. Mahalanobis distance, Support Vector Machines, etc). All these classification techniques lead to unconnected regions and regions with holes. In this scenario, a high level tool such as deformable models is helpful for addressing these problems. On the other hand, when fusing classification schemes with deformable models, the decision boundary issue is of extreme importance, because wrongly classified regions will have great impact on the deformable model result. For this reason, we use a relaxation of this problem using the whole likelihood map information as an external force into the deformation process of the active contour model. Figure 5.10 shows a salamander image and a classification using Mahalanobis distance. As one can observe, the classification errors can be critical for the segmentation results (these images have been modelled using the same co-occurrence matrix parameters as the other result over the same image [Parameters: $\alpha = \beta = 0.3$, $K = 0.2$, $\mu = 0.2$]). Most of these erroneous regions have low probability of being generated by our desired pattern. However, this probability can be higher than the one belonging to the background class.

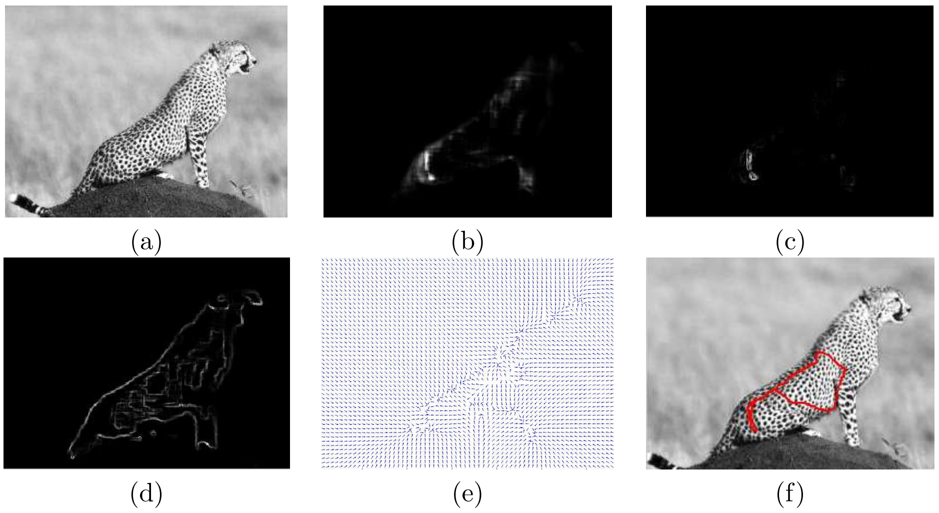


Figure 5.11: (a) Original cheetah image. (b) Likelihood map for the cheetah texture (c) Contours of the likelihood map (d) Contours of the enhanced likelihood map. (e) Vector field generated by the contours of the likelihood map (f) Classification results.

A second important issue is the reason to use the ELM contours. The likelihood map has some irregularities that lead to either misplaced contours or the appearance of non-desired contours. These effects make its direct use as an external force unreliable, leading us to define an enhanced approach using a regularization framework over the contours of the ELM. Figure 5.11 shows an example of the usage of the likelihood map and how the contours are improved for better performance. Figure 5.11.(c) shows the contours of the likelihood map, an irregular likelihood map leads to weak and sparse contours. The vector field guiding the deformation (shown in fig.5.11.(e)), points to a misclassification, as it is illustrated in fig.5.11.(f).

The problem is solved by extracting contours on the ELM and using them to guide the statistic snake (fig. 5.6)

5.3 Stop and Go performance

We have split experimental assessment of Stop and Go performance into two blocks. First, a comparison to other geometric snakes is carried out; second, we show robustness of our formulation to the critical parameters, Δt and V_0. The methodology of comparison we have used is as follows:

5.3.1 Experimental Setting

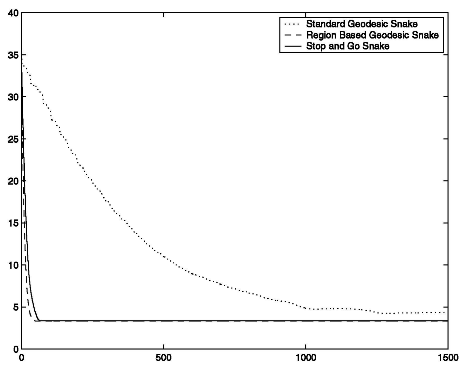
We have compared our approach with a standard geodesic evolution and a standard region-based approach. As our approach is based on a characteristic function estimation (likelihood maps), the experiments have been carried out using the same region information. In the standard geodesic evolution we have used the contours of a thresholded characteristic function estimation. In the standard region-based approach we have used the thresholded estimated characteristic function. All methods have been implemented using an explicit Euler scheme. Experiments focus on determining snake efficiency in terms of accuracy and speed of convergence. These quantities will be measured as follows:

- Snake accuracy is given by the maximum and mean distances to the target curve. Notice that because we use the Euclidean distance, it is its maximum value the measure that best detects failure to enter into concave regions.
- Speed of convergence corresponds to the number of iterations necessary to stabilize the snake. We recall that in an explicit Euler scheme, the number of iterations is proportional to the time step, Δt, used.

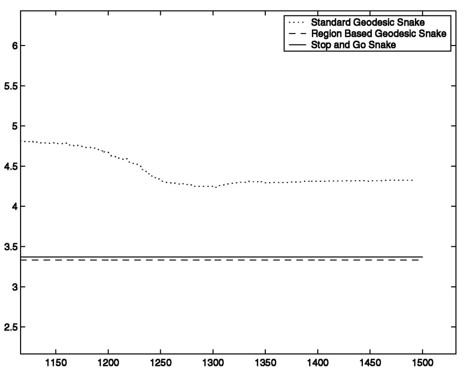
5.3.2 Comparison Results

We have tested the different methods on the tetra-foil (fig.5.13) and a highly non-convex shape (fig.5.14). The parameters used for geodesic snakes are $\{V_0 = 0.6, \Delta t = 0.2\}$ for the tetra-foil and $\{V_0 = 1.3, \Delta t = 0.2\}$ for the non-convex shape. In the first case, the balloon force value is the maximum admitted by the algorithm in order to converge to the concave area and not collapsing to a point; in the second it is the minimum value that allows the snake to surpass the saddle point that impedes the snake to converge in the concave region. In the case of the bracelet (figure 5.14), we also run geodesic snakes with a high balloon force in order to illustrate its importance in the convergence process. Region based snakes have been implemented using the highest time increment possible, $\Delta t = 100$, so that speed of convergence is optimized. Stop and Go snakes use their standard configuration $\{V_0 \in [0.3, 1], \Delta t = 1\}$, which are the ones achieving the best compromise between speed of convergence and accuracy. Plots of the mean and maximum distances during the evolution reflect snake efficiency and its evolution smoothness. Smooth motion in the last steps is crucial to stabilize the snake at the equilibrium curve as oscillations in its energy trouble any standard stop criterion.

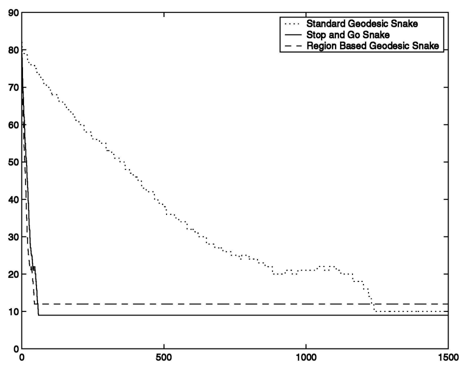
In the case of the tetra-foil (fig.5.13) all methods achieved their goal, correctly segmenting the shape (see images in the second row of fig.5.13). Graphics for convergence to the tetra-foil are shown in fig. 5.12, mean distances are in the first row and maximum in the second one. Solid line graphics in fig. 5.12 correspond to Stop and Go, dotted ones to geodesic snakes and dashed ones to region-based snakes. The ordinate axes are the distances given in pixels and the abscise axes represents the number of iterations. Concerning speed of convergence, Stop and Go standard configuration compares to region-based methods for both mean (fig. 5.12.(a)) and maximum distances (fig. 5.12.(c)). Meanwhile, in spite of using their optimal speed configuration,



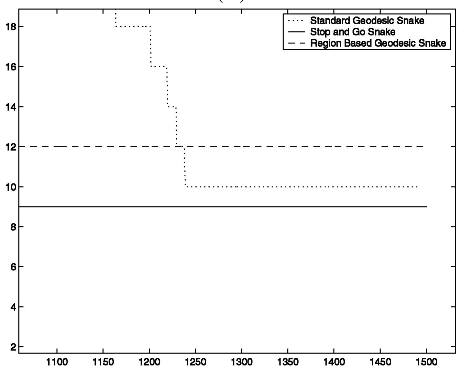
(a)



(b)



(c)



(d)

Figure 5.12: Efficiency assessment for the tetra-foil. Mean distance plot (a) and (b) detail. Maximum distance (c) and detail of the last steps of the evolution (d).

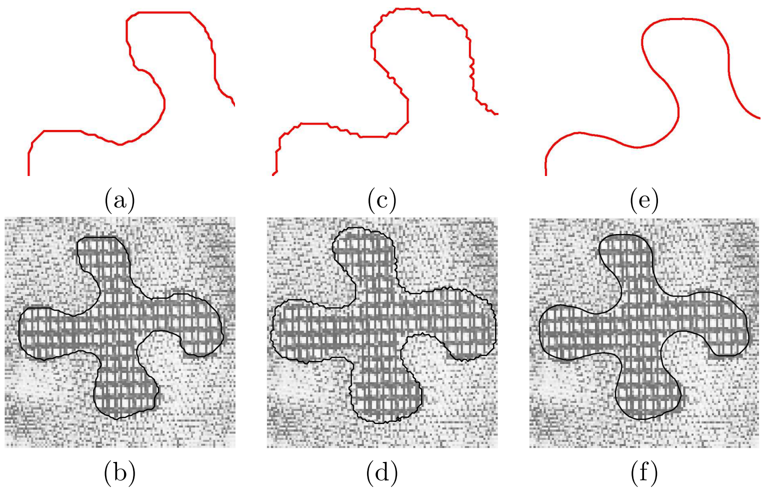


Figure 5.13: Segmentation of the tetra-foil. Details of (a) geodesic, (c) region-based and (e) *Stop and Go* snakes. Segmented images for (b) geodesic, (d) region-based and (f) *Stop and Go* snakes.

geodesic snakes are, by no means, the worst performers. It is worth noting that speed of convergence of Stop and Go can be increased (as our experiments in the next section show) without significantly reducing its accuracy. Maximum distances are similar for all techniques with a difference of less than 4 pixels (see detail of fig. 5.12.(d)). The hill in the geodesic snake maximum distance (fig. 5.12.(c)) corresponds to its entrance into the tetra-foil concave segments due to the local minimums that appear. This phenomenon is produced by ridges in the Euclidean distance map and is common to all methods. However, because Stop and Go and region-based snakes convergence scheme lacks of the curvature term, they easily overpass concavities, so that this convergence phase is hardly detected in their maximum distance graphics. Average distances clearly reflect out performance of Stop and Go and region-based snakes compared to the geodesic model. Differences are better appreciated in the detail of 5.12.(b).

The former accuracy measurements conform to the final segmenting curves drawn in fig.5.13 . Notice the importance of the constant speed in the geodesic scheme. The zoom of the first row of fig.5.13 captures the visual differences among the techniques. As expected, the regularity endowed by curvature makes the standard geodesic segmentation (fig.5.13 (a)) smoother than the oscillating contour given by a standard region-based approach (fig.5.13 (b)). Stop and Go differential curve outperforms the linear piecewise geodesic model.

In the case of non-convex shapes (figure 5.14), different configurations of standard geodesic snakes are shown, as well as their counterparts using standard region-based and *Stop and Go* snakes. The plots of maximum (dotted line) and mean (continuous line) distance to target boundary are also depicted. The ordinate axes are the distances given in pixels and the abscise axes represents the number of iterations. Figure

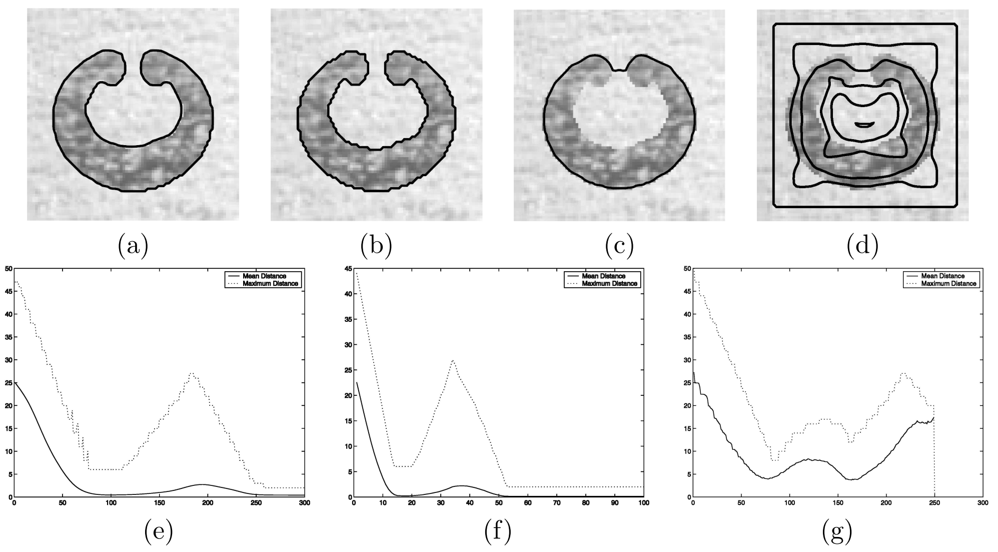
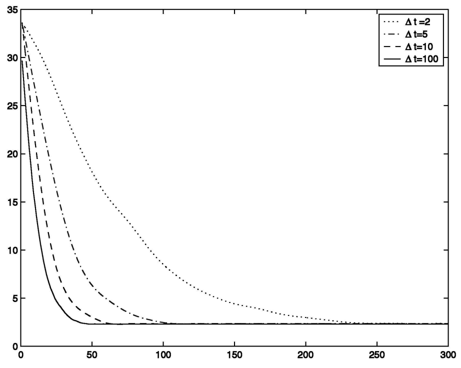


Figure 5.14: Segmentation of the bracelet. (a) Stop and Go, (b) region-based snakes, (c) non convergent geodesic snake and (d) collapsing of a geodesic snake.

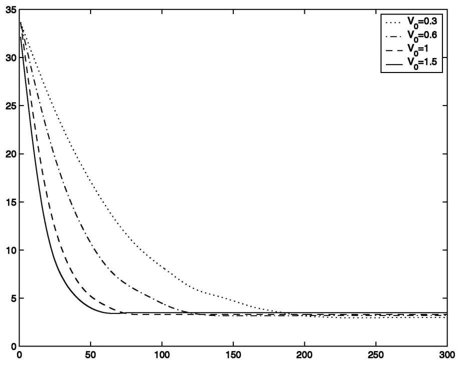
5.14.(a) shows the *Stop and Go* convergence using $\{V_0 = 0.25,\ \alpha = 0.35, \Delta t = 1\}$, it shows a smooth behavior and good convergence rate (which can be seen in figure 5.14.(e)). Figure 5.14.(b) depicts the final convergence of the region-based snake. The convergence is faster (figure 5.14.(f)), however the final shape is jagged. Figure 5.14.(c) shows a standard geodesic snake with a balloon force added $\{V_0 = 0.8,\ \alpha = 0.3,\ \Delta t = 0.2\}$. Although there is a balloon force it is still unable to converge in the concave area. Figure 5.14.(d) shows the evolution of a standard geodesic snake once we have raised the balloon force in order to penetrate the concave area. As can be seen it collapses to a point. The mean and maximum distance of this process are depicted in figure 5.14.(g) . The little hill in figure 5.14.e and figure 5.14.f are caused by the proximity of the saddle point, corresponding to the transition of the snake from a convex shape to a concave shape. At the saddle point there is a local minimum which has to be surpassed. In figure 5.14.g this minimum can not be surpassed, as can be seen in figure 5.14.c, or is surpassed causing the whole snake to collapse into a single point, as displayed in figure 5.14.d, an therefore preventing convergence to target boundary.

5.3.3 Stop and Go parameters study

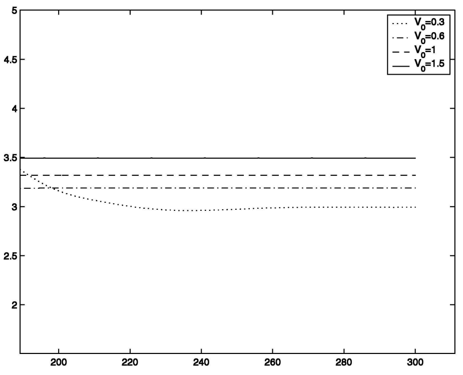
In order to asses *Stop and Go* parameters influence on the final segmentation, we have chosen the tetra-foil image (figure 5.13). Figure 5.15 plots the quality measurements of the behavior of the Stop and Go active model when the parameters are changed. Figure 5.15.(a) shows the robustness of the method to the variation of the time step, keeping fixed V_0. We can see that we can increase the time step at will without altering the performance if we maintain the following constraint $\Delta t \cdot \alpha \leq 0.4$ due to



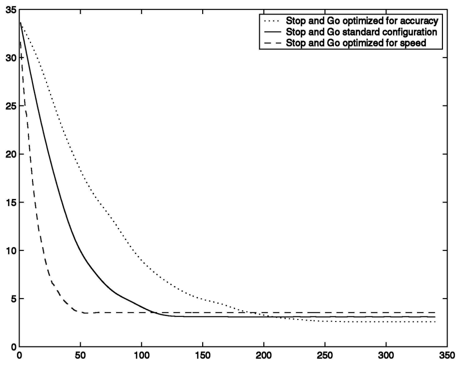
(a)



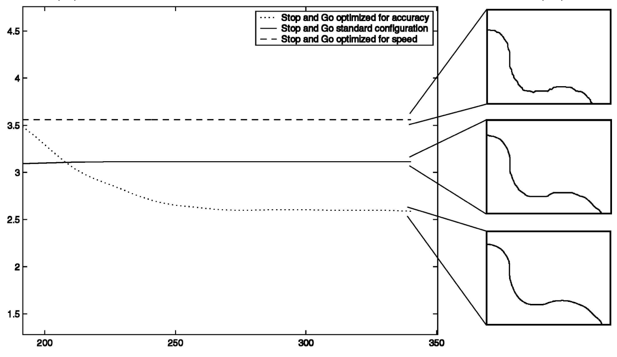
(b)



(c)



(d)



(e)

Figure 5.15: *Stop and Go* parameters study. Distances for (a) time increment variation, (b) V_0 variation and (d) optimal parameters. Details for (c) V_0 plot and (e) optimal parameters plot.

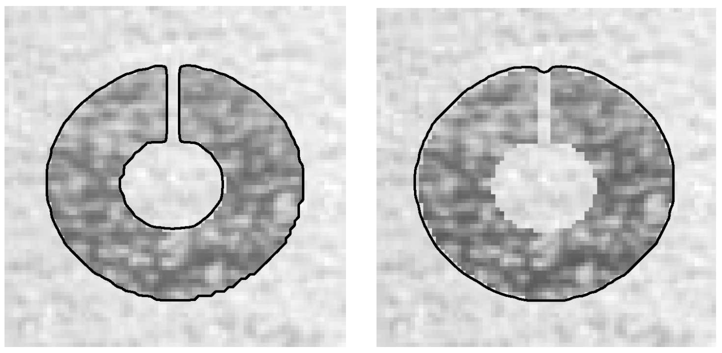


Figure 5.16: Segmentation of ambiguous image using *Stop and Go* snakes. (a) Concave contour and (b) convex closure.

the curvature effect constraints at the later steps of the process.

Figure 5.15.(b) shows the effect of the dynamic GO term characterized by V_0. Typical values of V_0 range from 0.1 to 1.7 for $\Delta t = 1$. Over 1.7 the GO term clearly exceeds the STOP term making the snake to collapse to a point. Figure 5.15.(c) shows a detail of figure 5.15.(b). In the figure we observe the effect in the accuracy when varying V_0. Low V_0 permits the STOP term to operate sooner and provides better accuracy in the results. The accuracy improvement is due to the fact that likelihood maps usually under-estimate the object of interest. Figure 5.15.(d) and figure 5.15.(e) show typical configurations of the *Stop and Go* scheme. The figures show how we can achieve different behaviors of the scheme ranging from very fast convergence to slower, smoother and more accurate results. The parameters used to illustrate the behavior in figures 5.15.(d) and 5.15.(e) are the following. Under the name "fastest configuration" we refer to the *Stop and Go* snake with parameters that allow it to converge as soon as possible, without considering accuracy to the contours. This kind of schemes can be useful for object detection. The fastest configuration showed uses the following parameters $\{V_0 = 1.3,\ \Delta t = 1.3 \text{ and } \alpha = 0.23\}$. The higher the values of V_0 and Δt, the fastest the convergence. The name "smoother configuration" designates a *Stop and Go* snake fully concerned in accuracy and smoothness. This kind of schemes is useful in segmentation if we are concerned with finding the exact boundaries of the object of interest. The smoother configuration uses the following parameters: $\{V_0 = 0.2,\ \Delta t = 0.5 \text{ and } \alpha = 0.6\}$. If V_0 is comparable to $\Delta t \cdot \alpha$ and Δt is low, the competition between both terms allows smoother behaviors. The trade-off between the former configurations has the following parameters: $\{V_0 = 0.4,\ \Delta t = 1 \text{ and } \alpha = 0.35\}$. In this case, we have raised V_0 and Δt, increasing the resulting convergence speed.

In some cases the shape we are looking for is ambiguous in the sense that multiple interpretations can be offered for the same shape. For example, figure 5.16 shows a wheel-shaped image, however depending on our application we rather have a closed round contour or an explicit open contour. This kind of control on the evolving snake can be achieved using the parameter V_0. Figure 5.16.(a) shows the *Stop and Go* snake

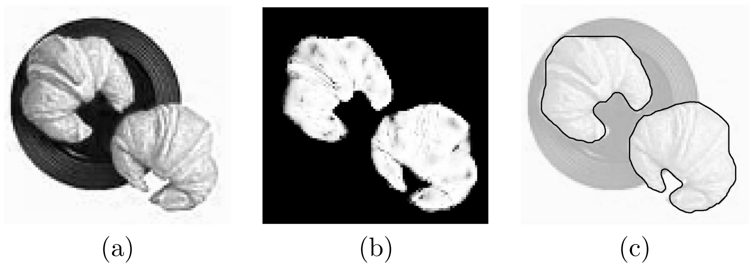


Figure 5.17: (a) Original croissant image. (b) Likelihood map associated to the croissant features. (c) Segmentation using *Stop and Go* snakes.

leaking through the hole to adapt explicitly to the shape. Figure 5.16.(b) illustrates the effect of reducing V_0 if our desire is to close the hole in the shape.

5.3.4 Application to Real Scenes Segmentation

This section provides visual validation of the *Stop and Go* scheme applied to real scene segmentation. The approach has been used in different application fields. We show examples of the results in texture segmentation and color segmentation fields.

Figure 5.17 shows the result of using the scheme in a color segmentation framework. Figure 5.17.(a) shows the original image to be segmented. In the image the croissants are depicted in yellow and orange tones. So we are looking for yellow and orange areas in the image. Figure 5.17.(b) shows the likelihood map obtained where the brighter areas correspond to higher likelihood values. This likelihood map is casted in the *Stop and Go* scheme resulting in the segmentations showed in figure 5.17.(c) and figure 5.17.(d). Figure 5.17.(c) illustrates the segmentation of the croissant image using a standard *Stop and Go* configuration $\{V_0 = 1, \Delta t = 1, \alpha = 0.3\}$. Figure 5.17.(d) depicts the smoother behavior achieved by altering the parameters.

Figure 5.18 shows how the proposed scheme can be applied with modifications to texture segmentation. Figure 5.18.(a) shows the likelihood map of the zebra texture. Figure 5.18.(b) shows the final segmentation using *Stop and Go* active models. Note the fact that the shadow of the zebra has lower probabilities than the rest. By using the *Stop and Go* active model we can surpass these low likelihood areas that none of the other snake models can. On a side note, notice also that the shadow under the rear legs of the zebra is really intense, and the likelihood value at that area is quite high, therefore, unless we change the likelihood map the rear legs will not be segmented. Figure 5.18.(c) shows the likelihood map of a salamander texture applied over image 5.18.(d). We can see that there are a good amount of false positive regions that does not correspond to the salamander itself but to the background. Using the *Stop and Go* formulation we are able to bypass those areas. Notice the control of the snake on low-likelihood valued areas that is one of the advantages of the method when used on real scenarios.

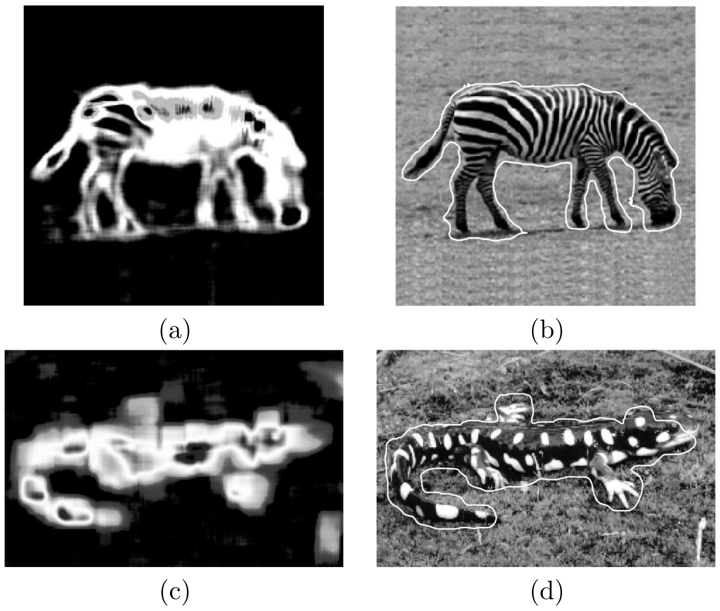


Figure 5.18: Texture-based segmentations. Zebra segmentation: (a) Likelihood map and (b) *Stop and Go* segmentation. Salamander segmentation: (c) Likelihood map and (d) *Stop and Go* segmentation.

5.4 Performance of Generative Stop and Go

With the former enhancements, the likelihood map is suitable to be used in real scenes. In this section we validate our approach providing evidences on how different frameworks affect the technique.

5.4.1 Framework descriptions

In order to illustrate and validate the approach proposed in the paper we have chosen a supervised texture problem. We assume *a priori* information on the desired regions is available. The first step is the learning process. A model is created for each of the regions of interest. Once a feature descriptor is chosen and our data modelled, a likelihood density function is estimated and the likelihood map created. Two basic frameworks have been used: the first uses co-occurrence matrix measures, linear discriminant analysis and gaussian mixture models; the second uses gabor bank of filters and a pseudo-naive bayes compounding of the likelihood of the responses for each pixel of the image. Next sections describe both approaches.

Co-occurrence framework This framework uses co-occurrence matrix measures as a feature extraction. Once the features have been computed a linear discriminant analysis is performed and the dimensionality is reduced. After the dimensionality reduction the likelihood density function is estimated by means of a gaussian mixture modelling process.

Gabor bank of filters framework The starting features for this framework are the Gabor bank of filters responses. Once they have been obtained, each of the responses is modelled by a gaussian mixture model. The final likelihood value will be the combination of the likelihood values weight by a validity measure.

Once the filter responses have been obtained we compute a gaussian mixture model for each of the features. The final likelihood measure is a compound of those marginal probabilities. First a quality measure can be attained. This measure is related to how good each feature models target textures. A natural way for obtaining these measures is by letting them inversely depend on the false positive and false negative of the training data. Therefore, by counting the misclassified pixels in the training data for each of the features we have a reliability measure that can be used as a weighting term. The final likelihood map is the sum of the weighted marginal probabilities for each of the responses,

$$LM(x, y) = \sum w_{i,j} \cdot p_i(I(x, y)|C_j)$$

where $p_i(F(I(x, y))|C_j)$ is the marginal probability for each of the responses of the pixels of the image to a filter (represented by i), under the hypothesis that they belong to class C_j. Each of the marginal probabilities are calculated using a gaussian mixture model.

5.4.2 Example results

Using the frameworks described in the previous section, we have validated the approach by segmenting natural scenes. We have used 50 images.

Figure 5.19 shows a segmentation result using the co-occurrence framework on synthetic data. Figure 5.19.a shows the original likelihood map for the tetra-foil texture. Figure 5.19.b shows the enhanced likelihood map after the two-class enhancement. We can observe that the likelihood after the enhancement is more compact and better defined . Figure 5.19.c shows the result after the topological enhanced segmentation. Figure 5.19.d depicts the final segmentation. The parameters for the segmentation are $\lambda = 1$, $\alpha = 0.3$, $\gamma = 0.3$ using a step of 1. It must be noted that parameters λ and α will not be further modified in the rest of the co-occurrence matrix framework results. Although it has not been exhaustively tested, the parameter λ seems to depend mostly on the feature extractor. On the other hand, α can be kept constant to the value 0.3 due to the fact that the likelihood map is normalized between 0 and 1 and therefore the maximum curvature integration step is upper bounded by 0.4.

Figure 5.20 shows a segmentation result using a zebra image. In this figure the co-occurrence matrix framework displays some undesired regions with low likelihood. Figure 5.20.a shows the original likelihood map. Figure 5.20.b displays the two-class enhanced likelihood map. Figure 5.20.c shows the topological enhanced likelihood map. Figure 5.20.d illustrates the final result. Two things must be noted in this example: first, the shadow of the back legs is captured due to its local resemblance to a horizontal stripe; second, the rest of the shadow is not segmented because it has lower likelihood values areas allowing the active model to leak through those holes.

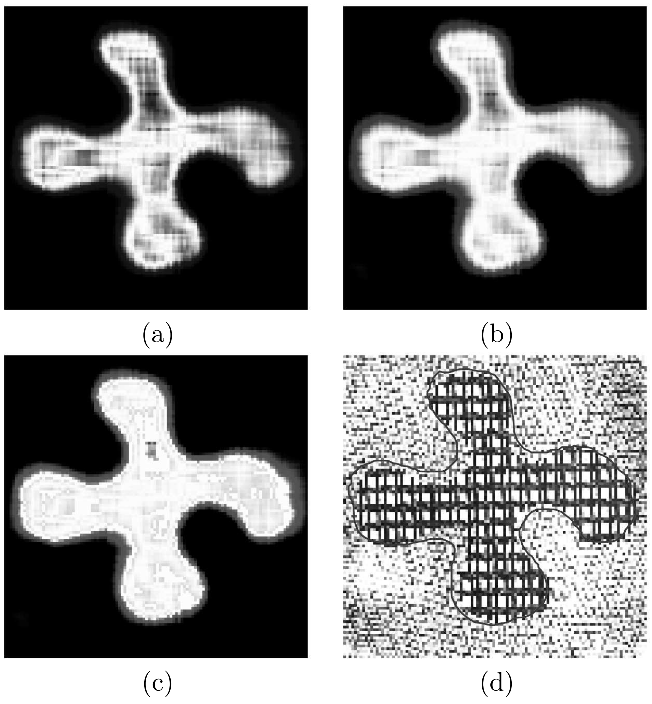


Figure 5.19: Example of a segmentation of the clover. (a) Original likelihood map. (b) Two-class enhancement likelihood map. (c) Topological enhanced likelihood map. (d) Final segmentation.

Figure 5.21 shows a segmentation result using a salamander image.Figure 5.21.a shows the original likelihood map for the salamander texture. Figure 5.21.b shows the two-class enhanced likelihood map. Figure 5.21.c shows the topological enhanced likelihood map. This figure is particularly interesting because it shows the collapsing feature mentioned in former sections (section 2 and 4). The low likelihood areas that are not bypassed by the "Go" term gradually collapse to a point and disappear thanks to the curvature term, allowing a neat segmentation of the salamander, as depicted in figure 5.21.d.

5.4.3 Discussion

In this section we briefly discuss some of the results and compare our method with a classical region segmentation approach.

Figure 5.22 shows the comparison between our method and a classical statistic region method proposed by Zhu and further evolved by Paragios. Figure 5.22.a shows the original salamander image. Figure 5.22.b depicts the original likelihood map for the salamander texture. As can be seen, the "safety" effect is clear in this image. Remember that the "safety" effect concerns the regions of high likelihood being an internal sub-region of the desired object. We can see that there is are lots of low

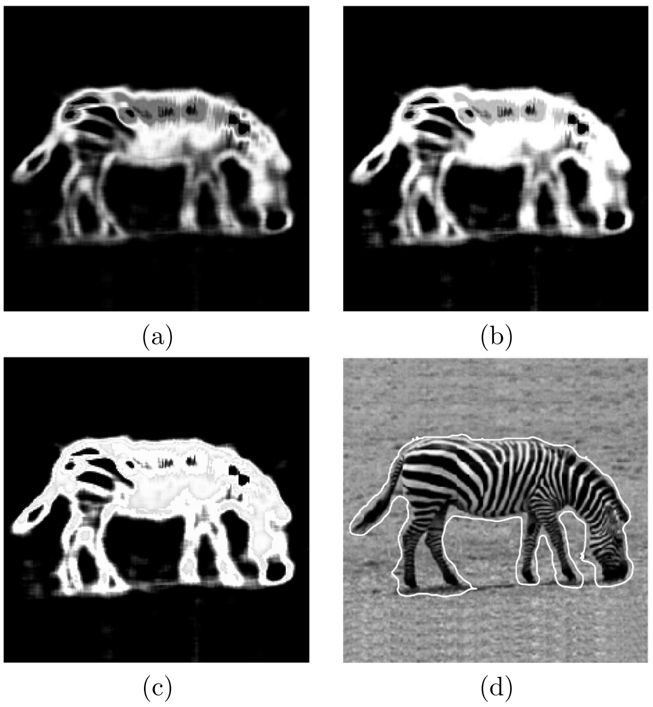


Figure 5.20: Example of segmentation of a zebra image. (a) Original likelihood map. (b) Two-class enhanced likelihood map. (c) Topological enhanced likelihood map. (d) Final results.

likelihood regions. We can apply a classic bayesian classification scheme to obtain the regions in which the likelihood of being a salamander is higher than the likelihood of not being a salamander (provided we can estimate that likelihood). Figure 5.22.c shows the bayesian classification result. As can be seen, little regions with low likelihood of being a salamander appear. Using a classic region competition scheme ($\alpha \log \frac{P_{Background}}{P_{Target}}$) will result on the image of figure 5.22.e. Note that, as we expected, the resulting segmentation is really close to the bayesian classification result. Our scheme is illustrated in figures 5.22.d and 5.22.f. Figure 5.22.d shows the enhanced likelihood map. In the figure we can se that most of low likelihood areas has been removed and the "safety" area defines much better the object of interest. Figure 5.22.f shows the final result after using our method. It must be noted that though one could argue in the classical scheme that some kind of area filtering could have been performed (not really easy because the classification is implicit on the curve evolution), the non solved question for the region competition schemes is what happens if a low likelihood large area is detected. In that situation, area filtering does not solve the problem. However, our approach overcomes that problem by definition of the evolution scheme. In fact, this effect is one of the motivations for our work.

Figure 5.23 has been segmented using the second framework. The parameters for the segmentation of the tiger are adjusted for the feature extraction process and

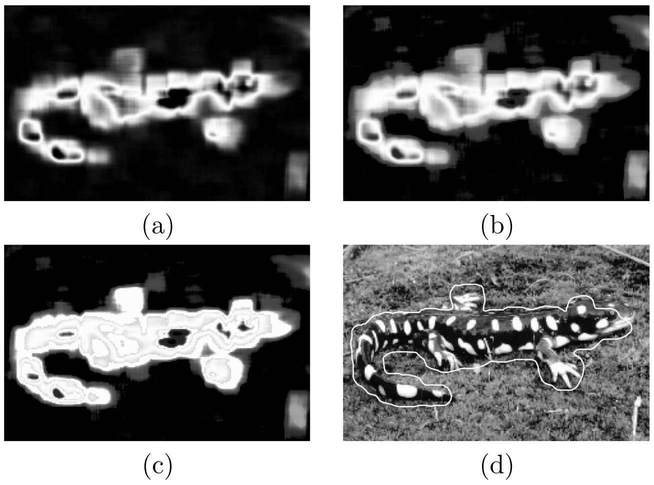


Figure 5.21: Salamander segmentation. (a) Original likelihood map for the salamander texture. (b) Two-class enhanced likelihood map (c) Topological enhanced likelihood map. (d) Final segmentation.

kept constant for the rest of the process. The 2-class parameter $\lambda = 0.7$ and the deformable model curvature parameter is set to $\alpha = 0.3$. This figure illustrates the area control effect of the snake. Figure 5.23.a shows the topological enhanced likelihood map. As can be seen in the figure, different likelihood estimation processes lead to different characteristic likelihood maps. Figures 5.23.b, 5.23.c and 5.23.d shows the influence of parameter γ to the method. Figure 5.23.b shows the segmentation using $\gamma = 0.2$. Figure 5.23.c shows the segmentation using $\gamma = 0.45$. Figure 5.23.d shows the segmentation using $\gamma = 0.7$. By varying γ we can "control" the influence of the higher likelihood regions. This effect is due to the competition between the three terms, if the "Go" and curvature term surpass the "Stop" term the region will collapse and disappear.

In practice, the only parameter that has to be adjusted depending on the effect we want is the gamma parameter (γ). This parameter regulates the trade-off of the terms. If γ is increased, higher convergence speed is achieved, removal of little regions is emphasized, but lesser smoothness in the contours is obtained. As a side note on the smoothness issue, the smoothness achieved by means of our method is heavily constrained by the STOP term. Therefore, there is a limit in how smooth the curve is. This means we can not impose the desired smoothness if it is outside the smoothness range of the method.

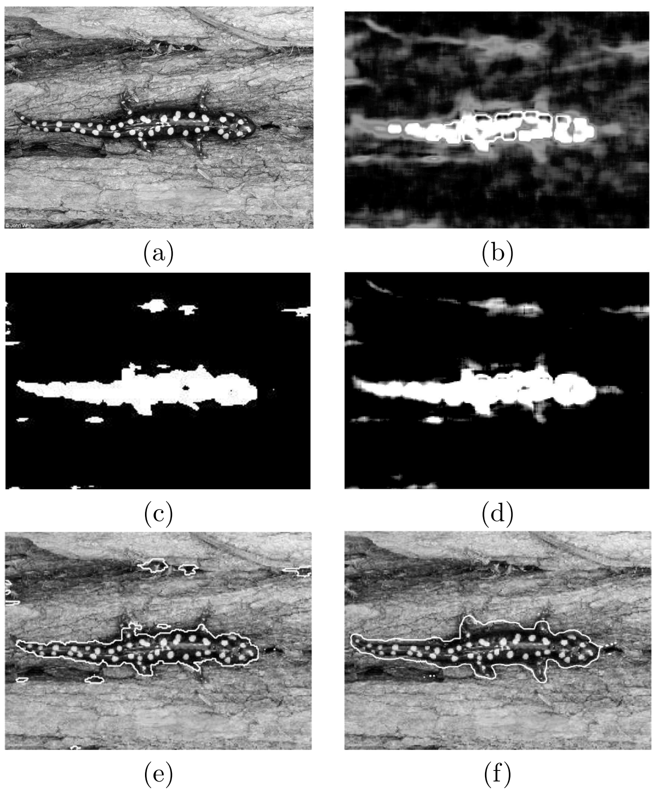


Figure 5.22: (a) Original salamander image. (b) Original likelihood map for the salamander texture. (c) Bayesian classification result. (d) Enhanced likelihood map. (e) Region based segmentation. (f) Segmentation using our method.

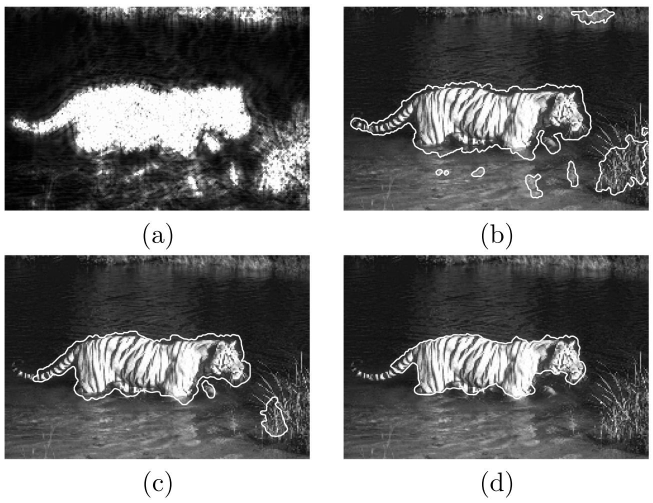


Figure 5.23: Tiger segmentation using naive bayes framework. (a) Topological enhanced likelihood map. (b) Segmentation using $\gamma = 0.2$. (c) Segmentation using $\gamma = 0.45$. (d) Segmentation using $\gamma = 0.7$.