A Model Based Method for Characterization and Location of Curved Image Features

Thierry Blaszka, Rachid Deriche

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

A Model Based Method for Characterization and Location of Curved Image Features

Thierry Blaszka and Rachid Deriche

N° 2451

Décembre 1994

PROGRAMME 4

ISSN 0249-6399

A Model Based Method for Characterization and Location of Curved Image Features

Thierry Blaszka* and Rachid Deriche*

Programme 4 — Robotique, image et vision
Projet Robotvis

Rapport de recherche n°2451 — Décembre 1994 — 25 pages

Abstract: This report deals with the development of a parametric model based method to locate and characterize precisely important curved features such as ellipses and B-splines based curves. The method uses all the grey level information of the pixels contained within a window around the feature of interest and produces the complete parametric model that best approximates in a mean-square sense the observed grey level image intensities within the working area. Different solutions have been developed to reduce the computational time required by such approaches and a large number of experiments involving real images have been carried out in order to test and compare the reliability, the robustness and the efficiency of the different proposed approaches.

Key-words: Low-Level Image Processing, Physics of Image Formation, Curved Features.

(Résumé : tsvp)

*{blaszka}{der}@sophia.inria.fr

Unité de recherche INRIA Sophia-Antipolis
2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex (France)
Téléphone : (33) 93 65 77 77 – Télécopie : (33) 93 65 77 65

Une Méthode par Modèles pour la Caractérisation et la Localisation des Primitives Courbes des Images

Résumé : Ce rapport présente le développement d'une méthode fondée sur des modèles paramétriques pour localiser et caractériser précisément des primitives courbes importantes telles que des ellipses et des courbes B-Splines. La méthode décrite ici utilise toute l'information sur les niveaux de gris des pixels contenus dans une fenêtre autour du centre d'intérêt qu'est la courbe et donne le modèle paramétrique complet qui approxime le mieux au sens des moindres carrés les niveaux de gris observés. Différentes solutions ont été développées afin de réduire le temps de calcul requis par de telles approches et un grand nombre d'expériences sur des images réelles ont été effectuées de manière à tester et comparer la fiabilité, la robustesse et l'efficacité des différentes approches proposées.

Mots-clé : Vision bas niveau, formation des images, Primitives courbes.

Contents

1 Introduction

This paper presents an approach which is a natural extension and generalization of the work presented in [DB93]. It deals with the localization and characterization of curved image features.

Since the pioneering work of Terzopoulos [Ter86] and Kass [KWT87] using deformable contours, different approaches have been described by [FL90], [MSMM90], [YH92], [CB92a], [CAS92], [CB92b], [TS92], [BD93b], [BD93a].

Following these ideas and the parametric model based approach developed by Rohr ([Roh92]), we have defined grey level intensities models of features of interest such as straight edges, corners, and triple junctions in [DB93]. These deformable surfacic models include geometric parameters such as their position and orientation, as in the previous methods, but also radiometric parameters. As an extension of this work [DB93] and in order to deal with more general features, we propose to extract and characterize closed curved features directly from the image by searching the parameters of a B-spline based closed curve that best approximate the observed grey level image intensities. However, it is important to note that the model associated to each feature includes a large number of intrinsic parameters (grey level intensities, geometric parameters of the curve ...) but also an important parameter which is related to the blurring effect due to the acquisition system. This is not the case while using classical deformable approaches where only the gradient magnitude information and some smoothness constraint on the resulting curve are taken into account in the energy minimization process.

It is interesting to note that the parameterization used to define the models allows us to have a sub-pixellic precision, due to the fact that the model is defined by geometric parameters but also takes into account the grey level intensities and the blur introduced by the acquisition system. The estimation of the parameters of the searched model is done by the use and combination of different minimization approaches. However, due to the large number of parameters and the cost of the smoothing used to describe the blur introduced by the acquisition system [PDTH89], it is important to pay a particular attention to the computational time required and develop efficient approaches in term of CPU and accuracy. To this end, we have elaborated different solutions to this problem.

The paper is organized as follows : After this introduction, a first section is devoted to the modelization of the image features, then the next one will present the evaluation of the parameters of our models, the third section will be devoted to the experimental results and finally we will conclude with the perspectives and the applications of this work.

2 Characterization of Image Features

Our motivation is to have a complete characterization of curved features and to propose an approach that allows us to detect them with a sub-pixellic accuracy. The considered features are all of the same global type : n regions with constant intensity defined by lines or by curve boundary in the working area. This type of attributes can be defined by the use of the function of Heaviside U defined as :

$$U(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{if else} \end{cases} \tag{1}$$

This function allows us to define linear features such as step edges, but in the real images such attributes don't appear because of the blur introduced by the acquisition system. Then, we will consider the convolution operation with some smoothing kernel S to characterize this blur :

$$\tilde{I}(x,y) = \int_{\alpha=-\infty}^{+\infty} \int_{\beta=-\infty}^{+\infty} S(\alpha,\beta) I(x-\alpha, y-\beta) d\beta d\alpha = I \otimes S \tag{2}$$

where $\tilde{I}$ denotes the smoothed image I. The smoothing kernels that we consider are the Gaussian one :

$$g(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}} \qquad G(x,y) = g(x)g(y) \tag{3}$$

and the exponential one :

$$e(x) = \frac{\alpha}{4}\left(\alpha|x| + 1\right) e^{-\alpha|x|} \qquad E(x,y) = e(x)e(y) \tag{4}$$

These filters lead us to define the models which we will denote, in the rest of the paper, as Gaussian model and exponential model depending of the smoothing

kernel used. In the sequel, we introduce the models of the features and start by presenting the approach that has been developed for the case of linear features.

2.1 Linear Features

This type of feature is easy to define because of the nature of their boundaries which are lines. Indeed the sign of a line equation allows to discriminate the two half-planes defined by this line, and then with the use of the Heaviside function we can have an analytical model of step edge. In order to take into account the blur introduced by the acquisition system, we use the convolution of a smoothing kernel such as Gaussian or exponential filter. In the case of the edge, the model is defined by 5 parameters which are : the orientation of the line and its distance to the origin, the grey-level in each half-plane defined by this line, and the factor of blur.

In the case of a corner we need two intersecting lines, and with the same method as for the edge, the model may be defined with the Heaviside function. To introduce the blur in our modelization this "ideal" model is convolved with one of the smoothing kernels, Gaussian or exponential. In the case of the Gaussian smoothing, the convolution operation cannot be simplified and a large number of numerical integration is required to perform this operation. This renders the method very time consuming and inefficient in term of CPU time. Our solution to solve this important problem has been to use the exponential filter which is formally integrable, and leads to a close form of our model. No numerical integration are required and the CPU time is largely reduced. The parameters defining the corner model are : the location of the corner, the orientation of its axis, its aperture, the grey-levels inside and outside the corner and the blur coefficient.

The model of the triple junction is simply the sum of two corners with the same location and with one common line. These constraints lead to a model with 9 parameters which are : the location of the junction, the orientation of the axis of the first corner, the apertures of the two corners, the grey levels within the 3 regions, and the blur coefficient. The gain in CPU time by the use of the exponential filter instead of the Gaussian one is also important in this case.

2.2 Curved Features

The previous models are very useful for indoor scenes because of their polygonal environment. For more general processes we would like to have primitives not limited to lines, and to this end we consider features delimited by curves. For a great generality we want to use curves with a great degree of freedom, but this implies a great complexity of the curve, of the model, and then a great CPU time to evaluate this model. To test our approach and to valid it, the first type of considered curves is the closed curves. The open curves could have been considered without more problem than the closed ones, this is just an arbitrary choice.

The definition of our models of closed curves starts with the use of ellipses because of their simplicity. This is the object of the next sub-section. The results obtained with the ellipse models (see section 4) lead us to pursue the definition of closed curve features in a more general way. The choice of the curve support for a more general closed curve model fell on the closed B-Spline and will be described in the sub-section following the Ellipse Models.

2.2.1 Ellipse Models

In our context, the simplest way to define ellipses, is to consider their analytical formulation :

$$N_l(x,y) = \frac{x^2}{a^2} + \frac{y^2}{b^2} - 1 = 0 \tag{5}$$

where a and b denotes the lengths of the ellipse axis. Like in the case of a line this equation leads us to use the Heaviside function to distinguish the inside from outside of the ellipse. A sharp ellipse can be defined by $U(N_l(x,y))$. In order to consider more general ellipses we must take into account the orientation θ of its axis, and the position (x_0, y_0) of its center as shown in the figure 1. Moreover, we add the grey-level intensities inside (A) and outside (B) the ellipse. Then the expression of the model becomes :

$$N'_l(x,y,x_0,y_0,\theta,a,b,A,B) = (A-B)U(N_l(x',y')) + B \tag{6}$$

with :

$$\begin{cases} x' = & (x-x_0)\cos(\theta) + (y-y_0)\sin(\theta) \\ y' = & -(x-x_0)\sin(\theta) + (y-y_0)\cos(\theta) \end{cases} \tag{7}$$

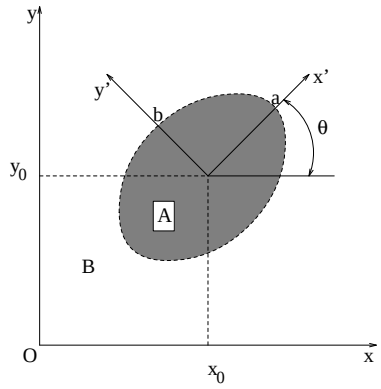

Figure 1: Ellipse model

Considering the blur leads to the following expression :

$$M_l(x, y, x_0, y_0, \theta, a, b, \sigma, A, B) = N'_l(x, y, x_0, y_0, \theta, a, b, A, B) \otimes S(\sigma) \quad (8)$$

where $S(\sigma)$ denotes the smoothing kernel used (Gaussian or exponential). This yields to a model defined by eight parameters. At this stage, one can note the difference between the approach of Lipson *et al* in [LYO+90] which define the ellipses geometrically and after having evaluated the ellipse parameters, computes the mean grey-level inside it, while our model includes intrinsically the informations on the grey-levels and adds the important information on the blur.

2.2.2 Closed B-Spline Models

In order to deal with a larger class of curved features, we have also defined another type of curve model based on B-Spline closed curves.

With the B-Spline curves and their intrinsics parameters, we can define a lot of features such as smooth curves, open or closed, with a variable degree of continuity(see [BBB87]). In fact we limit ourself to the smooth closed curves defined by their degree d, and their control vertices which are all of multiplicity one and then the B-spline curve is C^{d-2}. For example the points of such a B-Spline curve are defined as :

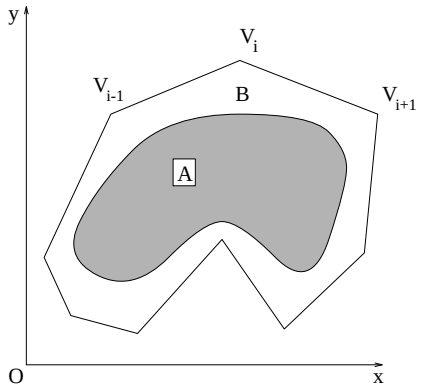

Figure 2: B-Spline model.

$$M(t) = \sum_{i=1}^{n} V_i B_i^d(t) + \sum_{i=1}^{d} V_i B_{n+i}^d(t) \tag{9}$$

where t is the parameter varying along the curve, n the number of control vertices V_i, d the degree of this B-Spline, and B_k^d the basis functions of degree d (see [BBB87]).

For our application, we consider a curve with a fixed degree and then its parameters are only the control vertices ; this leads to a model with $2n + 3$ parameters which are the coordinates of the control vertices (V_i), the grey-level intensities inside and outside the curve (A and B), and the blur coefficient (σ)(see figure 2).

This representation of curve prohibits the use of the Heaviside function in its previous form, and consequently we can't derive a closed form for the smoothed model. To solve this problem, we use an algorithmical solution which consists to create a synthetic image of the curve and to fill this curve with one classic algorithm like in [Smi79]. At this stage, we have a model of B-Spline curve $N_b(V_1, \ldots, V_n, A, B)$ including the grey-level intensities inside and outside the curve, but without smoothing. The next step is to smooth this image, using a discrete convolution. As for the previous features, we have defined two models of B-Spline curve depending on the smoothing filter :

$$M_b^g(V_1, \ldots, V_n, \sigma, A, B) = N_b(V_1, \ldots, V_n, A, B) \otimes G(\sigma) \tag{10}$$

which use the Gaussian filter, and

$$M_b^e(V_1,\ldots,V_n,\sigma,A,B) = N_b(V_1,\ldots,V_n,A,B) \otimes E(\sigma) \tag{11}$$

which use the exponential one. Due to the CPU time needed by a direct convolution operation, we use the recursive implementation of the Gaussian smoothing described in [Der92] and the recursive implementation of the exponential filter given in [Der90]. These approaches lead us to reduce our computation time twice, at least, without any lost of precision.

3 Approximation of the data

Having defined our feature models, we are now interested in detecting them from the images. First, we consider that we have a window centered in an initial estimate of the localization of a feature and we know which type of feature lies in this area. Then, we can use an iterative method to characterize this feature, but we need a first vector of parameters to initiate our process.

Even if the method has been proven to be robust to the case where the initialization is far from the solution (see the experimental part), starting with a parameter vector far from the solution leads to a convergence time that may be too long and then inefficient in term of CPU. To tackle this problem, we have developed a fast method that starts from initial conditions that may be far and produce closer initial conditions by providing a rough solution to the minimization problem. This method is denoted the *variance descent approach.*

3.1 Variance descent approach

This method is based on the remark that every considered feature defines n iso-intensity regions in the working area. If we know exactly the parameters of this feature and if there is no blur then the sum of the standard deviations within each region will be null. Therefore our method consist to define an energy criterion Σ which is the sum of the standard deviations of each region defined by the feature in the working area. Let's define the energy term as :

$$\Sigma^2 = \sum_{k=1}^{n} \left(\frac{\sum_{(i,j)\in R_k} (I(i,j) - \bar{I}_k)^2}{Card(R_k)} \right) \tag{12}$$

where n is the number of regions R, $I(i,j)$ is the intensity of the pixel (i,j), and $\bar{I}_k$ denotes the mean intensity of the region R_k. After having defined this energy, we want to minimize it with respect to the parameters of the considered feature. Two cases are considered. The first one is related to the linear features (edges, corners, triple junctions), and the second one is related to the case of curved features.

3.1.1 Linear features

For this type of feature, our methodology is as follow : first we extract 2 or 3 points of edge (2 for corner and edge, 3 for triple junction) in the border of the working area using a 1-D edge detector [Der90]. If we are looking for an edge, this stage is sufficient because with two points we have a line, the grey-level intensities can be initialized with the means of the intensities in the two half-planes defined by this line, and we can initialize the blur coefficient to 1. But for a corner or a triple junction we need the location of the vertex. To this end, we proceed as follow : starting from the center of the window as initial position for the considered feature, we minimize Σ with the following algorithm : evaluate the energy for the center of the working area and its eight neighbors, we choose as new candidate for the location of the feature the point with the minimum energy term and we iterate until the energy stops to decrease, which means that a local minimum has been reached. This yields a location of our feature with pixellic precision. The angular parameters are defined by the angle defined by the edge points and the location of the vertex, the intensities grey-level are initialized with the means of each region, and the blur coefficient is set to 1. This set of parameters is then used as the initial conditions for the model based approach (see in the section 3.2) that will provide the final result.

3.1.2 Curved features

For this type of feature the approach is different from the previous one. We consider that the first curve is given inter-actively by the user or by a previous process, and then we propose just to refine the parameters of this feature.

Case of ellipses : To define an ellipse we require a vector of 5 parameters : $\vec{P} = (x_0, y_0, a, b, \theta)$ where (x_0, y_0) denotes the position of the ellipse, θ the

orientation of its principal axis, and a and b the lengths of its axis. Let's note $P[i]$ the ith composant of the vector $\vec{P}$. To approximate $\vec{P}$, we use the following method :

First, the energy Σ corresponding to the initial vector of parameters is evaluated. Each parameter is moved from its initial position, keeping the others unchanged, then the energy term corresponding to this new parameter vector is computed. The set of parameters corresponding to the minimal energy term computed is retained and the process iterates until the energy stops to decrease. The way the parameters are moved depends on which parameter is considered : the center of the ellipse is moved in the eight directions corresponding to its eight neighbors, the axis lengths and the orientation are increased and decreased separately. This yields to this algorithm :

1. evaluate the energy Σ for the set of initial parameters $\vec{P}$.
2. $\Sigma' = \Sigma$ and $\vec{P}' = \vec{P}$
3. for i from 1 to 8 do
   - $\vec{p} = \vec{P}$, $p[1] = P[1] + \delta\ d_x[i]$, $p[2] = P[2] + \delta\ d_y[i]$
   - evaluate the energy Σ_i corresponding to $\vec{p}$
   - if $\Sigma_i < \Sigma'$ then $\Sigma' = \Sigma_i$, $\vec{P}' = \vec{p}$
   - endfor
4. for i from 3 to 5 do
   - $\vec{p} = \vec{P}$, $p[i] = P[i] + \delta$
   - evaluate the energy Σ_i corresponding to $\vec{p}$
   - if $\Sigma_i < \Sigma'$ then $\Sigma' = \Sigma_i$, $\vec{P}' = \vec{p}$
   - $\vec{p} = \vec{P}$, $p[i] = P[i] - \delta$
   - evaluate the energy Σ_i corresponding to $\vec{p}$
   - if $\Sigma_i < \Sigma'$ then $\Sigma' = \Sigma_i$, $\vec{P}' = \vec{p}$
   - endfor
5. if $\Sigma' < \Sigma$ then $\vec{P} = \vec{P}'$, $\Sigma = \Sigma'$ and goto 3 endif

where δ is a step in pixel for the parameters (x_0, y_0, a, b) and in radian for the parameter θ. The vectors $\vec{d}[i]$ represents the eight directions corresponding to the eight neighbors of a point :
$\{\vec{d}[i]\} = \{(-1,-1),(0,-1),(1,-1),(-1,0),(1,0),(-1,1),(0,1),(1,1)\}$
The couple $(\Sigma', \vec{P'})$ corresponds to the minimum energy and the corresponding parameter vector, respectively, during the current iteration, and the couple $(\Sigma_i, \vec{p})$ is the current candidate of energy and parameter vector. The couple $(\Sigma, \vec{P})$ represents the energy and the parameter vector of the previous iteration. The initialization vector is composed of the founded geometric parameters $\vec{P}$, the grey-level intensities are set to the means of each region and the blur coefficient is set to 1.

Case of B-Spline curves : the method used in this case is derived directly from the ellipse algorithm : the initial energy term is calculated ; each control point is moved in the eight directions of its eight neighbors, while keeping the other control points invariants ; the energy term corresponding to this new set of control points is calculated ; if the energy term is reduced the corresponding set of control points is retained and the process iterates until the energy term stop to decrease. In this case all the parameters of the curve are metric parameters then we have just one step δ to consider. The vector of parameters defining a B-Spline curve is $\vec{P} = (x_1, y_1, \ldots, x_n, y_n)$, and our algorithm becomes :

1. evaluate the energy Σ for the initial parameters $\vec{P}$
2. $\Sigma' = \Sigma$, $\vec{P'} = \vec{P}$
3. for i from 1 to n do, for j from 1 to 8 do
   - $\vec{p} = \vec{P}$, $p[i*2] = P[i*2] + \delta\ d_x[j]$, $p[i*2+1] = P[i*2+1] + \delta\ d_y[j]$
   - evaluate the energy Σ_i corresponding to the vector of control points $\vec{p}$
   - if $\Sigma_i < \Sigma'$ then $\Sigma' = \Sigma_i$, $\vec{P'} = \vec{p}$
   - endfor, endfor
4. if $\Sigma' < \Sigma$ then $\Sigma = \Sigma'$, $\vec{P} = \vec{P'}$ goto 3

where the couple $(\Sigma', \vec{P}')$ corresponds to the minimum energy and the corresponding set of control points, respectively, during the current iteration, the couple $(\Sigma_i, \vec{p})$ is the current candidate of energy and set of control points, and the couple $(\Sigma, \vec{P})$ represents the energy and the parameter vector of the previous iteration. The vectors $\vec{d}[i]$ are those defined in the previous paragraph on ellipses.

As expected due to the number of control points of the curve, this direct method, denoted direct *vda* approach in the rest of the paper, is computationally very expensive. Then, we have been interested in defining a more efficient method in term of CPU, and developed the following steepest gradient method, denoted gradient *vda* method in the rest of the paper. This approach corresponds to evaluate the initial energy, to compute the gradient of the energy function, to find the best step in order to minimize the energy function in the gradient direction and to iterate until the process stop to decrease. This yields to the following algorithm :

1. evaluate the energy Σ for the initial parameters $\vec{P}$
2. evaluate the gradient $\vec{g}$ of Σ with respect to $\vec{P}$
3. find α so that $\Sigma(\vec{P} + \alpha\vec{g})$ is minimal with respect to α
4. if $\Sigma(\vec{P} + \alpha\vec{g}) < \Sigma$ then $\Sigma = \Sigma(\vec{P} + \alpha\vec{g})$, $\vec{P} = \vec{P} + \alpha\vec{g}$, goto 2 endif

The evaluation of the gradient is made by its approximation by finite differences, that is :

$$\vec{g} = (g_1, \ldots, g_q) \quad \text{where} \quad g_j = \frac{\Sigma(\ldots, p_j + h, \ldots) - \Sigma(p_1, \ldots, p_j, \ldots, p_q)}{h} \tag{13}$$

The minimization of the step 3 is made by the routine E04ABF of the **nag** library which is based principally on quadratic interpolation and a dichotomic research to minimize a function of a single variable. This method has been proven to be faster than the previous one. The initialization parameter vector of our model-based approach is composed of the set of control points $\vec{P}$ founded with one of the previous *vda* approaches, and, like in the ellipse case, the grey-level intensities are set to the means of each region and the blur coefficient is set to 1.

3.2 Model-based approach

The final step is now to start from the close initial conditions provided by the previous approaches and use the following method to get a more accurate solution. The solution provided by the variance descent approach, as explained above is at a pixel precision. To improve this result, one apply a numerical method to evaluate the parameters with sub-pixellic precision while taking into account the blur introduced by the acquisition system. One can note that the *vda* approach is very fast because of the lack of smoothing, which is the most computing part in the evaluation of our models.
To estimate the searched parameters and the blur, we have defined an error function which takes into account only the pixel intensities :

$$F(\vec{P}) = \frac{\sum_{i=1}^{m}(M_a(i,\vec{P}) - I(i))^2}{m} \tag{14}$$

where M_a denotes the model of the considered feature, $I(i)$ the intensity of the pixel i, m the number of pixels of the working area, and $\vec{P}$ the vector of parameters of the considered feature model. Our objective is to minimize this function which is a sum of squares of non-linear functions. To this end, we use the Levenberg-Marquardt algorithm described in [BD94] and developed for the linear features characterization. One can note that this minimization routine requires only the image data, and any other preprocessing on the image such as smoothing or derivating.

4 Experimental results

To test our models and the robustness of our method we have made a lot of experimentations on noisy synthetic data and on real images.

In the case of linear features we will present only results on real images. In the figure 3 we have the application of the *vda* on a corner and a triple junction where the path from the center of the zone to the final solution of this method is marked with black arrows. We can note that the result is close to the desired feature and then is very good to initiate our model-based approach. Another remark is about the CPU time which is very interesting because of order of few seconds. A second test has been done for the robustness of our model-based

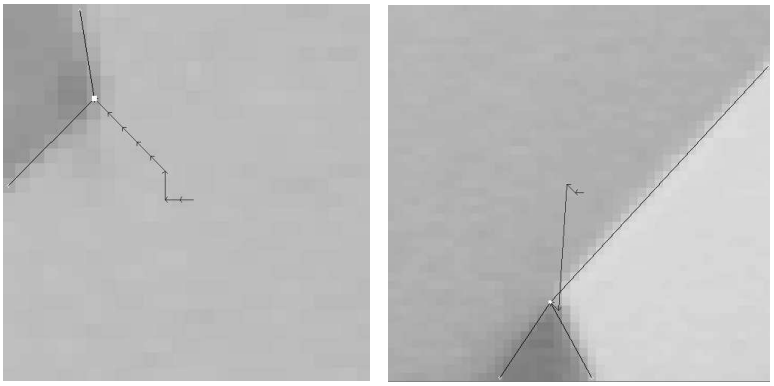

Figure 3: Convergence of *vda* for a corner and a triple junction

approach and is illustrated in the figure 4 where the initialization (in black) has been given inter-actively far from the solution (in white). The last image of the figure 4 presents the use of the *vda* (in black) as initialization, and we can see the result of our model-based approach (in white). The use of the *vda* in this case leads to the same final result but the CPU time is reduced twice, at least.

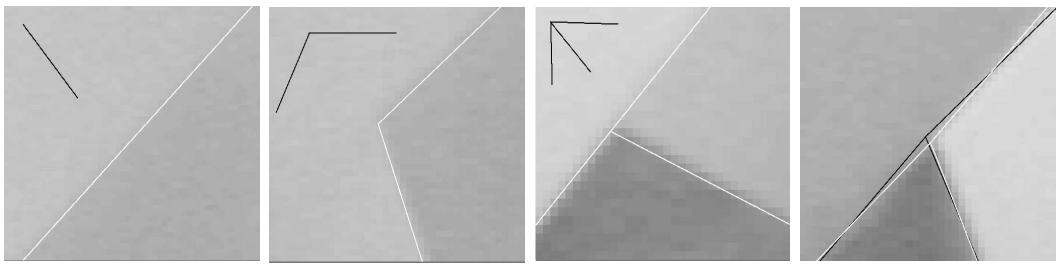

Figure 4: Convergence of the model-based approach on linear features.

In the case of curved features we have made some experiments on real images of ellipses as shown in the figures 5 and 6 ; the first figure presents the application of model of ellipse and the figure 6 presents the application of the model of B-Spline curves. On the first image of figure 5 one can see the manual initialization in black, and the convergence of the *vda* approach in white. In the second image of this figure lies the result of the model-based approach (in

white) initialized by the *vda* in black. The same scheme has been taken for the images in the figure 6 but with a model of B-spline curve. We can note that the final results are the same in the two types of model, but the time required by the approach with B-Spline is the half of this required by the approach with ellipses.

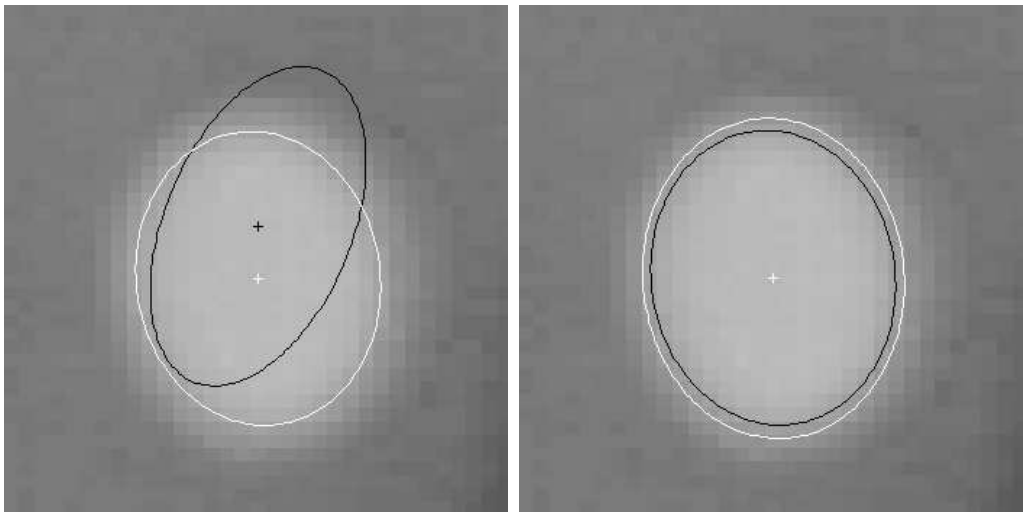

Figure 5: Application of ellipse models on real ellipse images (see text).

In the case of B-Spline curves we have tested the two approaches of the *vda* on synthetic and real images. However, we will present only results on real images. In the figure 7 we can see the result of the application of the direct *vda* method, and in the figure 8 we can see the results of the application of the gradient *vda* method. The CPU time required for the first approach is about 200 seconds for the cloud image (left image of fig 7) and about 650 seconds for the dog image (right image of fig 7). But the gradient *vda* method takes roughly just 30 seconds. Due to the fact that the *vda* is designed to determine an initial estimate of our parameters, the gradient *vda* method is more adapted. However, it is worthwhile to note that the initial conditions provided by these two approaches both leads to the same result when applying the model based approach and consequently, it is better to use the one that provide the initial solution in a faster way in term of CPU time, namely the gradient *vda* method.

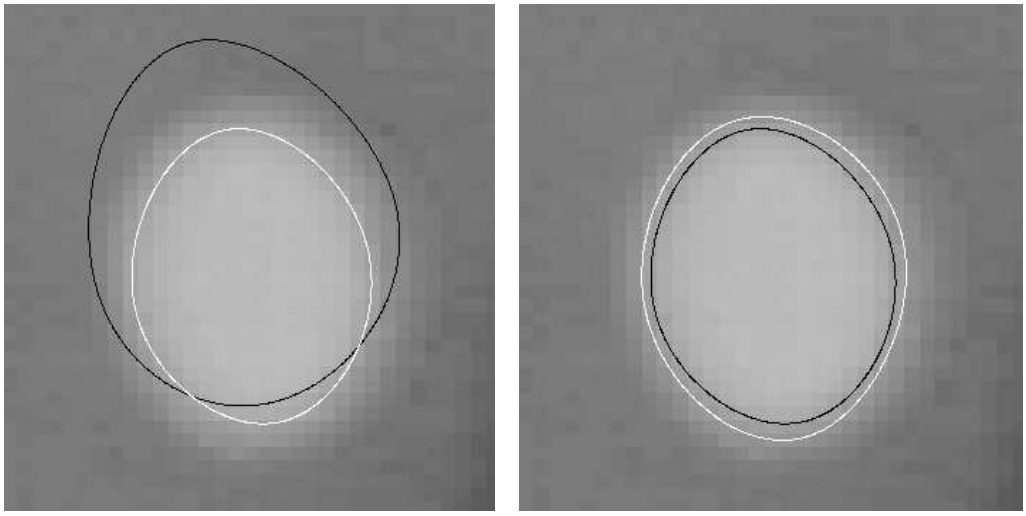

Figure 6: Results on real ellipse images with B-Spline models. (see text)

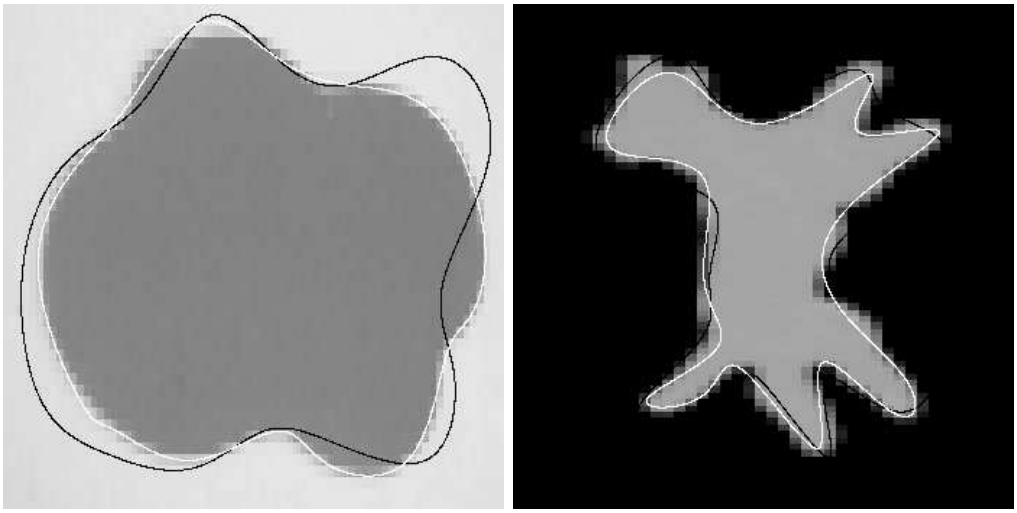

Figure 7: Application of the direct *vda* method in the case of B-Spline models on real images of B-Spline. (Initialization in black, result in white.)

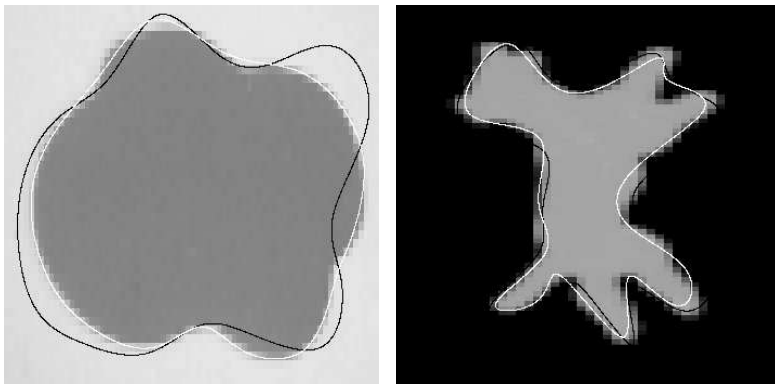

Figure 8: Application of the gradient *vda* method in the case of B-Spline models on real images of B-Spline. (Initialization in black, result in white.)

In the case of the model based approach for B-Spline curves we have tested our models, first on synthetic images as shown in figures 9 and 10 but also on real images as seen before in the case of ellipses and in the case of more general curves which will be presented in the following. In the left image of the figure 9 we can see the application of the gradient *vda* method (in white) with a manual initialization (in black). The right image of fig 9 shows the application of the exponential model of B-Spline (in white) initialized by the result provided by the *vda* approach (in black). The images of the figure 10 illustrate the same kind of results while working on a slightly more complicated curve and using the Gaussian model of B-Spline in the step of minimization (in white on the right image) instead of the exponential one. One can see that the method developed is rather robust to the fact that the initial condition may be far (as seen in the left image of figure 10) and that the final result shown in the right of the fig 10 (in white) corresponds in a satisfactory way to the solution. This illustrates the fact that the model of B-Spline has a good convergence on images of smooth curves and the use of the *vda* allows to reduce drastically the CPU time of the convergence.

The application of our B-Spline models on real images is presented in the figures 11 and 12. The images on the left of the figures present in black the results of the application of the direct *vda* approach and the right side of the figures present in black the application of the *vda* gradient approach. In white

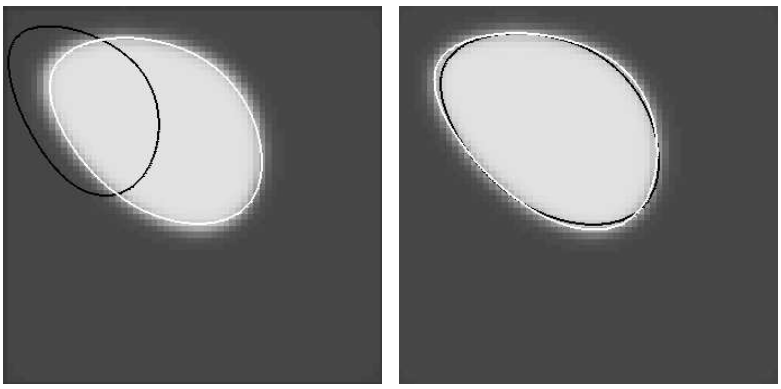

Figure 9: Convergence of B-Spline models on synthetic data. Left : in black the manual initialization and in white the result of the gradient *vda* method. Right : in black the previous result and in white the result of the exponential model of B-Spline.

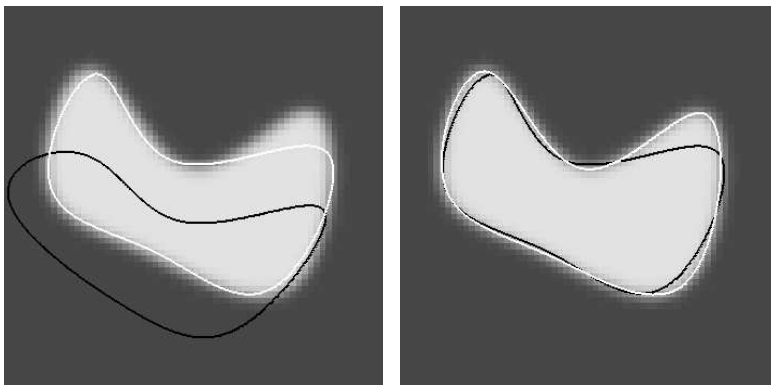

Figure 10: Convergence of B-Spline models on synthetic data. Left : in black the manual initialization and in white the result of the gradient *vda* method. Right : in black the previous result and in white the result of the Gaussian model of B-Spline.

on the figure 11 we can see the result of the application of our model-based approach with the Gaussian parametric model initialized with the black curve. In the figure 12 the white curves correspond to the result of the application of our model-based approach with the exponential parametric model initialized with the black curves.

In term of CPU time for the *vda* approaches, we have noted that the direct *vda* approach is very long, up to ten times longer than the gradient *vda* approach. In term of CPU time required by the model based approach that takes the *vda* output as initial condition to refine it and provide the final result, one have noted the following results :

- On the image representing the dog, the model based approach has converged in 250 sec with the output of the direct *vda* approach as initial condition (left fig 11) and in 300 sec with the output of the gradient *vda* approach as initial condition (right fig 12).

- On the image representing the cloud, the model based approach has converged in 140 sec with the output of the direct *vda* approach as initial condition (left fig 12) and in 150 sec with the output of the gradient *vda* approach as initial condition (right fig 11).

Therefore, one can see that the initialization by the direct *vda* allows to be slightly faster but globally the sum of the CPU time of the two steps is smaller in the case of the initialization by the *vda* gradient approach. Due to the fact that the accuracy of the model based approach is the same in the two cases of initialization, one can consider the *vda* gradient approach as the good way to produce a close initialization.

5 Conclusion

An efficient model-based method has been developed to locate and characterize precisely curved image features. Two different models have been developed to describe efficiently these features and a minimization process has been proposed to find the parameters that best approximate locally the observed grey level image intensities. There are two mains directions in which the approach presented in this paper can be extended: the first is to generalize the models in

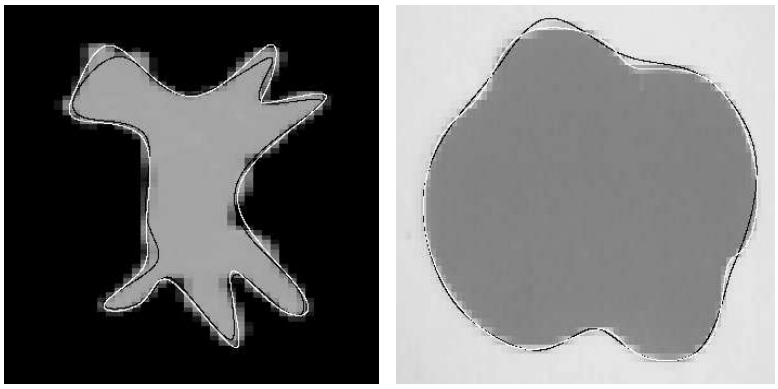

Figure 11: Application of the *vda* approaches in black (Left : direct *vda* method. Right : gradient *vda* method.) and of the Gaussian model of B-Spline in white in the two images (see text).

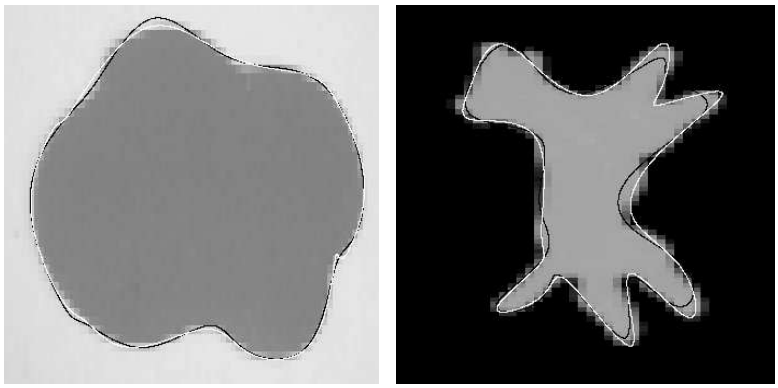

Figure 12: Application of the *vda* approaches in black (Left : direct *vda* method. Right : gradient *vda* method.) and of the Exponential model of B-Spline in white in the two images (see text).

order to take into account non planar intensity regions. The second is related to the application of the estimation of the blurring parameter to the problem of recovering depth from focus.

We believe that the ideas presented in this paper represent an important step in the application of a powerful formulation of the low level vision problems. While, a first step in this formulation has shown that some computational time requirements may decrease the interest of such model based approach, we have shown in this paper that some simplifications may be done that make possible to pursue this formulation in depth and although much work remains to be done for non-planar regions, the results obtained so far indicate that this approach to extract and characterize image features is very promising.

References

[BBB87] R. Bartels, J. Beatty, and B. Barsky. *An Introduction to Splines for use in Computer Graphics and Geometric Modeling.* Morgan Kaufmann, 1987.

[BD93a] B. Bascle and R. Deriche. Energy-based methods for 2d curve tracking, reconstruction and refinement of curves of 3d curves and applications. *Proceedings of Geometric Methods in Computer Vision, San Diego, July*, 1993.

[BD93b] B. Bascle and R. Deriche. Stereo matching, reconstruction and refinement of 3 d curves using deformable contours. *Proceedings of 4th ICCV, Berlin, 11-14 May*, 1993.

[BD94] Thierry Blaszka and Rachid Deriche. Recovering and characterizing image features using an efficient model based approach. Research Report 2422, INRIA, December 1994.

[CAS92] I. Cohen, N. Ayache, and P. Sulger. Tracking points on deformable objects using curvature information. In *Proc. 2nd ECCV, Santa Margherita Ligure, Italy*, May 1992.

[CB92a] R. Cipolla and A. Blake. Surface orientation and time to contact from image divergence and deformation. In *Proc. 2nd ECCV, Santa Margherita Ligure, Italy*, pages 187–202, May 1992.

[CB92b] R. Curwen and A. Blake. chapter dynamic contours: Real-time active splines. In Blake and Yuile, editors, *Active Vision*, pages 39–57. MIT Press, 1992.

[DB93] R. Deriche and T. Blaszka. Recovering and Characterizing Image Features Using An Efficient Model Based Approach. In *Computer Vision And Pattern Recognition*, pages 530–535, New-York, June 14-17 1993.

[Der90] R. Deriche. Fast Algorithms For Low-Level Vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(1):78–88, January 1990.

[Der92] R. Deriche. Recursively Implementing the Gaussian and Its Derivatives. In *Proc. Second International Conference On Image Processing*, pages 263–267, Singapore, September 7-11 1992.

[FL90] P. Fua and Y.G. Leclerc. Model Driven Edge Detection. *Machine Vision and Application*, 3:45–56, 1990.

[KWT87] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. In *First International Conference on Computer Vision*, pages 259–268, June 1987.

[LYO+90] P. Lipson, A.L. Yuille, D. O'Keeffe, J. Cavanaugh, J. Taaffe, and D. Rosenthal. Deformable Templates for Feature Extraction from Medical Images. In O.D. Faugeras, editor, *First European Conference on Computer Vision*, pages 413–417, Antibes France, April 1990.

[MSMM90] A. Menet, P. Saint-Marc, and G. Medioni. Active contour models: Overview, implementation, and applications. In *IEEE Conf. Syst. Man. Cyb. L.A.*, Nov 90.

[PDTH89] A.P. Pentland, T. Darrell, M. Turk, and W. Huang. A Simple Real-Time Range Camera. In *Proc. International Conference on Computer Vision and Pattern Recognition*, pages 256–261, 1989.

[Roh92] K. Rohr. Modelling and Identification of Characteristic Intensity Variations. *Image and Vision Computing*, 10(2):66–76, March 1992.

[Smi79] A.R. Smith. Tint Fill. In *SIGGRAPH-ACM*, pages 276–283, August 1979.

[Ter86] D. Terzopoulos. On matching deformable models to images. Technical Report 60, Schlumberger Palo Alto Research, Nov 86.

[TS92] D. Terzopoulos and R. Szeliski. Tracking with kalman snakes. In Blake and Yuile, editors, *Active Vision*, pages 3–20. MIT Press, 1992.

[YH92] A. Yuille and P. Hallinan. Deformable templates. In Blake and Yuile, editors, *Active Vision*, pages 21–38. MIT Press, 1992.

Unité de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unité de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unité de recherche INRIA Rhône-Alpes, 46 avenue Félix Viallet, 38031 GRENOBLE Cedex 1
Unité de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unité de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
ISSN 0249-6399