April 2000

Towards Continuous Image Representations

Frédéric Labrosse

Department of Mathematical Sciences
Computing Group

University of Bath
Bath
United Kingdom

F.Labrosse@maths.bath.ac.uk

Abstract

In the cinema industry, special effects performed during post-production generally use pixel based versions of the movie frames. Although this type of representation is easy to obtain, it has problems like the amount of data and processing difficulties. (How do you remove an object in an image when its outline is not well defined because of the blur naturally present in the image?)
We propose to use continuous, *i.e.* vectorial, representations. They are indeed easy to manipulate and it has been shown that they can be used to render very high resolution images, which is necessary for cinema, in affordable times.
In this report, we address a first step towards such representations: the extraction from images of smooth continuous contours at sub-pixel accuracy and some ways of representing their interior. The sub-pixel accuracy is necessary to obtain representations that are resolution independent.
Images are decomposed into structural regions that correspond to specified image characteristics. This is done using standard relaxation labelling. Information taken at different stage during the relaxation is used to extract *structural contours*. Sub-pixel accuracy is obtained by using snakes as well as the blur present in images (because of the acquisition process). We propose solutions, adapted to our context, to often mentioned problems of snakes, namely initialisation, parameter determination, and instability.
The interior of the structural regions must be represented to allow the rendering of images as close as possible to the original ones. We propose here two schemes, one using a single colour for each region, the second sampling the original image to allow smoothly varying colour in each region.

This work was supported by the European Union *Training and Mobility of Researchers* project and by the UK *Engineering and Physical Sciences Research Council*.

Published in April 2000.

Towards Continuous Image Representations

Frédéric Labrosse
(Department of Mathematical Sciences, University of Bath)

To get a copy of this document, see the following electronic address:

`http://www.maths.bath.ac.uk/~masfl`

or contact:

Department of Mathematical Sciences
Computing Group
University of Bath
Claverton Down
Bath
BA2 7AY
United Kingdom

1 Introduction

When film special effects are created at the post-production stage, they are often performed on digitised versions of the movie frames. By "digitised", we mean "turned into pixels." The required optical quality means that each frame must be digitised at high resolution, when the large number of frames involved combines to ensure that a great deal of processing is required. Moreover, digitisation introduces its own problems. For example, one basic problem is *How do you remove an object when its boundary spreads across several pixels in width because of the blur introduced by the acquisition system?* We claim that special effects should be performed using vectorial representations of the images. Indeed, such representations are easy to transform, merge, and can be rendered at any resolution (Froumentin & Willis 1999).

We will first decompose images into regions that correspond to key parts of the images. This decomposition is based on statistical properties: a region will be a pixel group having a given average and variance of colour. This will allow a structural decomposition of the image based on colours and will lead to *structural regions* and *contours*. For the purpose of image representation, we must characterise the interior of the structural regions (to allow image synthesis as close as possible to the original image). We thus propose two schemes, each being applicable to some region types or for different applications. The first uses a single colour and is suitable for single colour regions or for "simplified" rendering for non-photorealistic applications such as cartoon-like rendering or technical illustration. The second samples the original image and is suitable for smoothly varying regions. We do not pretend that these schemes are good for all regions, but they are certainly good ones for some. Other schemes to represent the interior of structural regions are currently under investigation particularly based on texture representations.

To extract contours that are independent of the image resolution and the region's position/orientation in the image, we must work at a sub-pixel accuracy. Sub-pixel information is present in images in the form of blur introduced either by the acquisition process or by the anti-aliasing of synthetic images. We will exploit that information to extract smooth boundaries to sub-pixel accuracy. In this way, we try to obtain as accurate a representation as the source data will allow.

In Section 2, we show how the image is segmented into structural regions while Section 3 will show how structural contours are obtained. Section 4 describe the continuous image representation and some results are given in Section 5.

2 Segmenting an image into structural regions

The first step is to segment the image to get structural regions, *i.e.* regions that are homogeneous given a criterion. This is done using relaxation labelling.

2.1 Relaxation labelling

The relaxation labelling (Rosenfeld, Hummel & Zucker 1976, Hummel & Zucker 1983) builds a mapping from a set of objects to a set of labels by propagating local rules. Briefly, and following Hummel & Zucker (1983), variables $p_i(\lambda)$ are associated with object i according

to:

$$p_i(\lambda) = \begin{cases} 1 & \text{if label } \lambda \text{ is associated with object } i \\ 0 & \text{if label } \lambda \text{ is } \mathbf{not} \text{ associated with object } i. \end{cases} \quad (1)$$

Intermediate values indicate intermediate degrees of association. Moreover, two more requirements are:

$$0 \leq p_i(\lambda) \leq 1 \quad \forall i, \lambda$$

and

$$\sum_{\lambda=1}^{m} p_i(\lambda) = 1 \quad \forall i,$$

where m is the number of possible labels. Although p_is are not necessarily probabilities[1], we call them probabilities.

Λ_i is the set of labels attached to object i and Λ_{ij} is the set of all pairs (λ, λ') such that label λ associated with object i is compatible with label λ' associated with object j. Compatibilities between neighbouring objects have to be expressed. For example, they can be:

$$r_{ij}(\lambda, \lambda') = \begin{cases} 1 & \text{if } (\lambda, \lambda') \in \Lambda_{ij} \\ 0 & \text{if } (\lambda, \lambda') \notin \Lambda_{ij}. \end{cases} \quad (2)$$

The important point here is that $r_{ij}(\lambda, \lambda')$ must be positive for compatible labels, negative for incompatible labels, and zero if the labels have no influence on each other.

The local support is a function of the number of neighbours of object i having labels compatible with label λ associated with object i:

$$s_i(\lambda) = \sum_{j=1}^{n} d_j \sum_{\lambda'=1}^{m} r_{ij}(\lambda, \lambda') p_j(\lambda'), \quad (3)$$

where n is the number of objects in the neighbourhood of the current object and d_j are weights associated with each object in the neighbourhood. These weights can be used to emphasise (or preserve) particular spatial relationships between neighbouring objects (see (Richards, Landgrebe & Swain 1981) for the case of pixel labelling).

The relaxation labelling updates the probabilities to obtain a consistent and unambiguous labelling, *i.e.* a labelling associating a unique label (non-ambiguity) with each object while verifying (consistency):

$$s_i(\lambda_i) \geq s_i(\lambda) \quad \forall \lambda, i.$$

We use the following rule to update the probabilities (Rosenfeld et al. 1976):

$$p_i^{t+1}(\lambda) = \frac{p_i^t(\lambda)(1 + s_i^t(\lambda))}{\sum_{\lambda'=1}^{m} p_i^t(\lambda')(1 + s_i^t(\lambda'))}. \quad (4)$$

This updating rule is an approximation of the more formal one proposed in (Hummel & Zucker 1983) but is widely used as it is simple and works well (see, *e.g.*, (Kittler & Illingworth 1985, Garbay 1986, Hansen & Higgins 1997)).

[1]Even if these quantities can have a probabilistic meaning at the beginning of the process (we initialise them as probabilities), there is no guarantee that this interpretation remains valid after several iterations (Kittler & Illingworth 1985).

2.2 Image segmentation

In the case of image segmentation, objects are pixels (denoted by their coordinates (x, y)) and labels correspond to sets of region attributes (a_i).

Note that since we do not know in advance — and do not want to emphasise — any particular image structures, all d_j are set to 1 in Equation (3). Also, the neighbourhood is a 3×3 square around the current pixel and includes the central pixel (Richards et al. 1981).

The relaxation labelling produces for each pixel a probability for it to belong to a region having each possible attribute set. This means that for each possible attribute set, we get, at the end of the process, an image whose pixels are white — or very bright — if they belong to a region having this attribute set, black – or very dark — if not, and with a grey transition from one value to the other on the region boundary. That transition is smaller than the blur in the original image because of the convergence properties of the relaxation labelling, which will remove small regions and sharpen the transitions from one region to another.

We currently use as region attributes the average colour μ and the colour variance σ (Garbay 1986, Hansen & Higgins 1997). The attribute set for the region i is thus $a_i = \{\mu_i, \sigma_i\}$. We use the CIE $L^*a^*b^*$ colour space because of its perceptual uniformity (Wyszecki & Stiles 1982, Henricsson 1998, Meyer & Greenberg 1987). In image segmentation it is common to disregard luminance because changes in luminance do not usually represent object boundaries but rather changes in lighting conditions (clouds, shadows, lights, *etc.*). In our case, we are interested in regions having homogeneous colour properties, not in regions corresponding to physical evidence. We thus consider all three coordinates of the CIE $L^*a^*b^*$ space.

Typical attribute sets must be specified before the relaxation process begins. This is done interactively by selecting parts in the image to be segmented that are characteristic of each region. The interactive scheme to specify these parts is acceptable in our application because we want to analyse image sequences and because the region attributes usually do not change dramatically between two consecutive images in a sequence. Moreover, this user interaction is very fast because only rough drawings are required to select the image parts.

Given these region attributes (μ and σ), we use the Mahalanobis distance which can be interpreted as being a normalised distance between a pixel value and an attribute set (Duda & Hart 1973):

$$d_{(x,y)}(a_i) = \frac{(I(x,y) - \mu_i)^2}{\sigma_i^2}.$$

$d_{(x,y)}(a_i)$ is the Mahalanobis distance between the pixel at position (x, y) and the region whose attributes are $a_i = \{\mu_i, \sigma_i\}$. This distance is used to compute the initial probability of each pixel to have each attribute set:

$$p^{(0)}_{(x,y)}(a_i) = \frac{d_{(x,y)}(a_i)^{-1}}{\sum_{j=1}^{m} d_{(x,y)}(a_j)^{-1}}, \tag{5}$$

where m is the number of regions. We use (2) as compatibility function.

Several criteria have been used to test when the relaxation should be stopped(Richards et al. 1981, Kittler & Illingworth 1985). They are usually based on the unambiguity of the labelling, *i.e.* one and only one label must be associated with each object with a probability of 1. Of course, this must be somehow slackened since we use real number probabilities that will never be 1 or 0. In our case, a pixel is said to belong to the region having the highest

probability(Hansen & Higgins 1997). The relaxation is stopped when the labelling does not change anymore. Figure 1 shows the percentage of pixels (with respect to the image size)

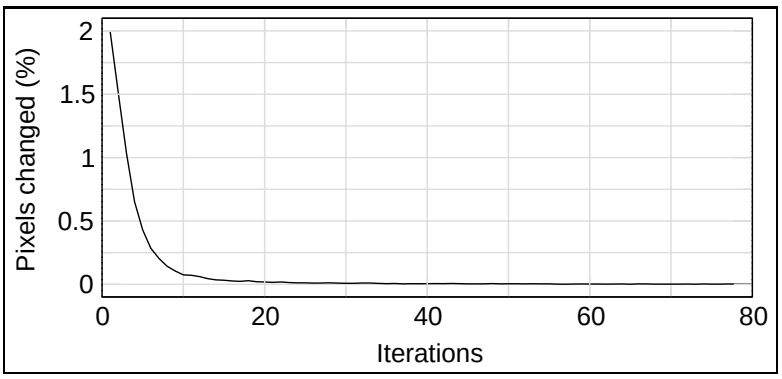


Figure 1: Changes in the labelling against the number of iterations (percentage with respect to the size of the image of pixels whose associated label has changed)

whose label has changed at each iteration. As can be seen, the stability is quickly reached and testing that value against a threshold can be used to stop the relaxation. Note that an early stop (depending on the image) will usually produce a great number of small regions. On the contrary, a late stop will remove all these small regions but will remove a great deal of detail in the images, in particular, it will round the corners. This latter point is not really a problem since the result of the labelling is only a starting point of the sub-pixel extraction. Moreover, image details could be preserved as in (Richards et al. 1981).

3 Structural contours

The structural regions are given with pixel accuracy. We need now to reach sub-pixel accuracy. This is done using snakes.

3.1 Snakes

Snakes (Kass, Witkin & Terzopoulos 1988), also known as active contours, allow the extraction of linear events in images. Following Kass et al. (1988), a snake, parametrically represented by $\mathbf{v}(s) = (x(s), y(s))$, has associated with it an energy that measures how far from an ideal model the snake is. The energy is made of two terms:

$$E = \int E_i(s)ds + \int E_e(\mathbf{v}(s))ds, \tag{6}$$

where $E_i(s)$ is the internal energy at the curvilinear abscissa s and $E_e(\mathbf{v}(s))$ is an external energy at the same abscissa.

The internal energy consists of a first-order term responsible for the snake continuity and a second-order term responsible for the snake rigidity:

$$E_i(s) = \frac{1}{2}\left(\alpha(s)\left|\frac{d\mathbf{v}(s)}{ds}\right|^2 + \beta(s)\left|\frac{d^2\mathbf{v}(s)}{ds^2}\right|^2\right), \tag{7}$$

where $\alpha(s)$ and $\beta(s)$ are parameters that control the relative importance of the two terms. Discontinuities and corners can be allowed at some positions by respectively setting $\alpha(s)$ and $\beta(s)$ to zero at the corresponding points.

The external energy is the value of a potential filed at the corresponding snake point. The field can take into account different kinds of sources. Kass et al. (1988) used forces produced by images and localised springs and repulsors. The energy coming from the image, using different operators, can be used to extract bright lines over dark background, edges, or line terminations.

3.2 Minimising the snake's energy

To make the snake fit its ideal model, the energy must be minimised. Several methods have been proposed to minimise the snake's energy. The first one, primarily used by Kass et al. (1988) but also by David & Zucker (1990), Cohen & Cohen (1993), and Berger & Mohr (1990), uses a variational approach. Basically, Equation (6) can be transformed into (Kass et al. 1988):

$$\mathbf{A}\mathbf{v}_t + \mathbf{f}(\mathbf{v}_{t-1}) = -\gamma(\mathbf{v}_t - \mathbf{v}_{t-1}) \tag{8}$$

obtained by the discretisation of the linear abscissa and approximating the time derivatives by finite differences. $\mathbf{v}_t$ is the snake's coordinates at a given curvilinear abscissa at time t. $\mathbf{A}$ is a penta-diagonal banded matrix depending only upon αs and βs. γ is the inverse of a step size (the average displacement of the snake's control points). $\mathbf{f}(\mathbf{v}_t)$ is the external force applied to a given snake control point at a given iteration, *i.e.* the variation of the external energy at a given control point at a given time:

$$\mathbf{f}(\mathbf{v}_t) = \left.\frac{\partial E_e}{\partial \mathbf{v}}\right|_{\mathbf{v}=\mathbf{v}_t}. \tag{9}$$

Equation (8) can be solved iteratively by matrix inversion:

$$\mathbf{v}_t = (\mathbf{A} + \gamma\mathbf{I})^{-1}(\gamma\mathbf{v}_{t-1} - \mathbf{f}(\mathbf{v}_{t-1})). \tag{10}$$

The matrix $(\mathbf{A} + \gamma\mathbf{I})$, which is of size $n \times n$ where n is the number of control points of the snake, can be inverted in linear time (Benson & Evans 1977).

Problems have often been reported with this optimisation approach. The main one is that the condition that leads to Equation (8) is only a necessary condition: the minimum is at a point where the derivative of the energy vanishes. This means that the optimisation is not guaranteed to find the absolute minimum. It is not even guaranteed that a minimum will be found. Indeed, the optimisation can stop at a stationary point. Moreover, if the snake is initially far from the minimum, it will not be attracted by it and thus will never find it (see (Amini, Weymouth & Jain 1990) for a detailed explanation). A second problem is that equilibrium is usually not reached because of instabilities introduced mainly by the non-smoothness of the external energy term[2].

To solve these problems, some solutions have been proposed. A bilinear interpolation of the forces implied by the image has been used by Cohen (1991). In (Cohen 1991, Cohen & Cohen 1993), an attraction potential is added to the energy of the snake; this potential

[2]Remember that this term comes from an image.

is created by the convolution of a Gaussian with a binary image produced by an edge-detection operator. A fully analytical term for the internal forces has been used by Rueckert & Burger (1995) by using C^2 polynomial spline patches instead of the discrete curve used by Kass et al. (1988). Another optimisation technique has also been proposed: dynamic programming (Amini et al. 1990, Geiger, Gupta, Costa & Vlontzos 1995). This technique has the advantage that the global minimum is guaranteed and that hard constraints (constraints that cannot be violated) can be added to the problem. But this method also has drawbacks. In particular, it is slower and requires more memory than the previous one. The major drawback in the case of our particular application is that the contour's control points are kept on the image grid, preventing a sub-pixel contour from being extracted.

In the two previous optimisation techniques, the first- and second-order derivatives are approximated using finite differences:

$$\left|\frac{d\mathbf{v}_i}{ds}\right|^2 \approx |\mathbf{v}_i - \mathbf{v}_{i-1}|^2 \tag{11}$$

and

$$\left|\frac{d^2\mathbf{v}_i}{ds^2}\right|^2 \approx |\mathbf{v}_{i+1} - 2\mathbf{v}_i + \mathbf{v}_{i-1}|^2. \tag{12}$$

Williams & Shash (1992) pointed out that these approximations presuppose two assumptions: the control points are evenly spaced at unit intervals and that the parameter s is the arc length of the curve. If the control points are not at unit intervals but evenly spaced, Equation (11) should be divided by d^2 and Equation (12) by d^4, where d is the distance between the points. This omitted term can be taken into account in α and β. If the control points are no longer evenly spaced, then the approximation for the first-order term penalises long distances between control points and the second-order term gives too large estimates for the curvature. The result is that the snake has a tendency to shrink and eventually ends as a point (or a line if it is not closed) if the snake is not subject to external forces. To solve that problem, a new formulation for the first-order term as well as a better approximation for the second-order term are proposed by Williams & Shash (1992). The average distance between consecutive control points is computed and the first-order term tends to minimise the difference between that average distance and each individual distance. That way, the first-order term favours evenly spaced control points. They also propose a new algorithm to perform the optimisation. This algorithm is based on a discrete grid and is faster and less memory consuming than the dynamic programming one.

We previously said that the initial snake should be close to the attracting image feature to converge properly. The original snake algorithm was created in an interactive application (Kass et al. 1988) in which the initial contour was drawn over the image and possibly pushed or pulled interactively to solve convergence problems. These interactions are not always possible and several methods to initiate snakes have been proposed. In (David & Zucker 1990), a short open snake is started at each pixel where an edge was detected[3] in a small valley created by an elongated Gaussian centred at each pixel and oriented as the edge. Each snake then evolves following Equation (8) and is made longer to produce a global covering of the edges. A similar approach is proposed in (Berger & Mohr 1990, Berger 1990) where a short snake is started following strong local evidence for an edge and elongated

[3]These edges are the result of the computation of the tangent field of the image (Zucker 1985).

locally following the edge to extract the complete edge. In (Cohen 1991), another approach is used. A new term is added to Equation (6) that inflates the snake[4]: this is the balloon model. This way, the initial snake can be far from any image feature (but must be inside them), the new term inflating the snake until it reaches such a feature. A dual active contour approach is proposed in (Gunn & Nixon 1997). Two linked snakes are used to delimit the image feature: one starts from the inside of the feature and expands while the other starts from the outside and shrinks. In (Lai & Chin 1995), a new model for the deformable contour is proposed: the generalised snake. The g-snake is a snake whose model takes into account a prior model. In that case, global deformations are also possible and the initial contour position is found using the generalised Hough transform. In this model, the control points are also positioned on the image grid.

Another problem with the method proposed in (Kass et al. 1988) is that it requires a lot of parameters that are manually fine-tuned. Fua & Leclerc (1990) propose a way to automatically and dynamically determine the snake's curvature weight (β, Equation (7), they use the same value for all control points) and the step size (γ, Equation (8))[5]. The β value is chosen so that the internal and external energies evolve the same way when the snake is at its optimum position, which is approximated by the snake at its initial position when close to the solution. γ is dynamically changed to ensure a monotonic convergence towards the optimum, *i.e.* reduced when the energy increases instead of decreasing. In (Cohen 1991), α, β, and γ are automatically determined. α and β are determined experimentally as respectively h^2 and h^4 where h is the distance between control points. γ is of the order of the pixel size and the external force $\mathbf{f}$ (Equation (9)) is modified as $\mathbf{f} = \frac{\partial E_e}{\partial \mathbf{v}} / \left| \frac{\partial E_e}{\partial \mathbf{v}} \right|$ to give the same importance to all edge values in the edge image.

3.3 Sub-pixel extraction

We use the snakes to reach sub-pixel accuracy from the pixel accurate structural region boundaries. We thus use these boundaries as snakes in their initial configuration. Then we make the snake evolve to extract the sub-pixel boundaries.

Since we want sub-pixel snakes, we use the minimisation that was proposed by Kass et al. (1988). The internal energy of a snake only contains the second-order term as the first-order term tends to shrink the snake, which will make it go inside the true boundary. We have indeed noticed that the first-order term is not very useful if the snake's control points are already evenly spaced, which is the case here. The internal energy thus becomes:

$$E_i(s) = \frac{1}{2}\beta(s)|\mathbf{v}_{ss}(s)|^2. \tag{13}$$

The potential field of a snake is made of an edge image that is the gradient modulus of an image I' built as follows from the original image I. The image I'

- is white (or at least bright) in the region whose boundary has made the snake,

[4]The term "inflate" applies only to closed snakes but the force can in fact be applied to non-closed snakes: it suffices to define a partition of the world into an inside and an outside of the snake to define the force. This partition is defined by specifying a normal to the snake on each point.

[5]Fua & Leclerc (1990) do not automatically determine the snake's continuity weight because they don't use that term in the snake's energy.

- black (or at least dark) outside of that region, and
- has a smooth transition between the bright and dark regions that reflects the image blur.

It is made of the probabilities at some point during the relaxation (Section 2). We do not use the final probabilities because they are too sharp and the edges in these do not represent correctly the edges in the original image. On the other hand, the initial probabilities lead to edges that are not well defined (as with the original image). Moreover, it is important to have the same region topology for the initial snakes (taken from the region boundaries at the end of the relaxation) and the edges in the potentials. Indeed, if they are different, then the snakes will be initially far from their final position and holes can remain between regions (where small regions have disappeared). When the number of labelling changes gets small (with respect to the labelling changes after the first iteration), then changes become very small in terms of regions and we can use the probabilities at that stage to extract the edges used as snake potentials. Moreover, probabilities at that stage are still smooth. Typically, we use the probabilities when the labelling changes cross the threshold of one tenth of the first labelling change. Figure 2 shows an image of a goose and the image I' corresponding to the dark parts. The edge image is then interpolated using bicubic Bézier patches to get a smooth analytic surface which is the potential. The term $E_e(\mathbf{v}(s))$ in (6) becomes:

$$E_e(\mathbf{v}(s)) = E_e(x(s), y(s)) = -\text{Interp}(|\nabla I'(x(s), y(s))|^2). \tag{14}$$

In the original implementation of snakes (Kass et al. 1988), all parameters are manually set. We propose an automatic way of choosing them. The parameter β in (13) is set to allow angles at some control points in the snake. It is first initialised to 1 for each control point. Then, control points making angles are sought along the initial snake configuration[6]. At these control points, β is set to the negated, then clamped to 0, cosine of the angle. Finally, all βs are normalised so that the amplitudes of the snake rigidity and external energy are equal. The rigidity is assumed to lie between its initial value and 0 (all control points are roughly aligned at the end, except where a discontinuity has been allowed, in which case the contribution is small). The external energy is assumed to lie between its initial value and the value found by adding the minimum contribution of each control point in its 3×3 neighbourhood (with increments of 0.5 pixels). This latter value is close to the minimum value since the snake control points are initially at less than one pixel from their final position.

The parameter γ in (8) is automatically and dynamically determined. It is initially computed so that a given average displacement (typically a quarter of a pixel) of the control points is allowed. Basically, (8) gives n vectorial equations from which we can extract n values for γ given the required average step-size, where n is the number of control points. The initial γ value is taken as the average of these n values. Then γ is increased until the snake becomes still. For this, we compute the energy's derivative (averaged over time) and when that derivative is below a given threshold, we reduce the step-size. Using this method, the energy ends up at a slightly higher value (Figures 3 and 4), but we have seen no lower quality of the result. The result is insensitive to the amount of increase of the damping

[6] An angle is detected at a control point when the minimum cosine of the angle made from the control point and neighbouring ones is higher than all neighbouring cosines.

(a) (b)

(c)

Figure 2: An image of a goose (a), the image I' for the dark regions (b), and a close up of I' (c)

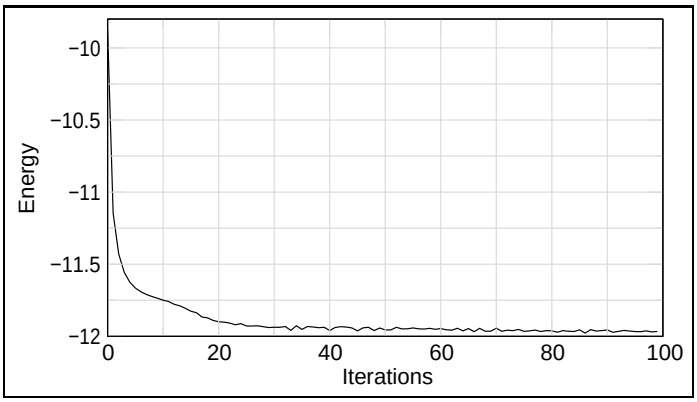


Figure 3: The energy of a snake while it converges: no snake damping

factor as well as to the value of the energy derivative threshold.

Though inspired by (Fua & Leclerc 1990), our approach is different because our context is different. For example, we cannot assume that the initial estimate of the snake is close to the final answer (even if this is true in terms of distance) since this is exactly that difference that makes the contour reaching sub-pixel accuracy. Also, in (Fua & Leclerc 1990), the damping

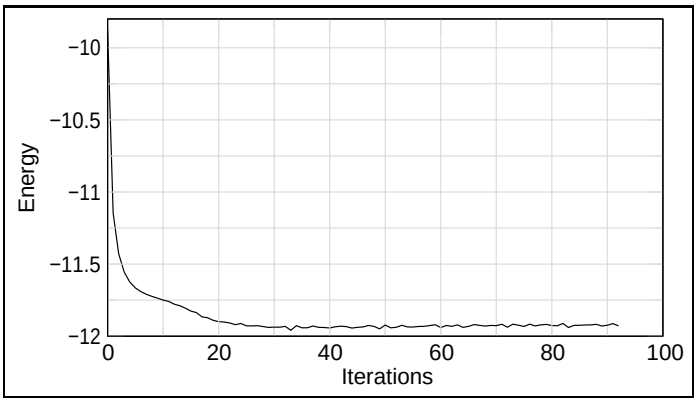


Figure 4: The energy of a snake while it converges: with snake damping

occurs when the energy increases instead of decreasing (with a backtracking procedure to ensure that the energy never becomes higher). We found that this prevents the snakes from reaching their global minimum but makes them converge towards a local minimum. Indeed, the energy, when becoming low, shows oscillations that help the snake to converge correctly (Figure 3). Figure 4 shows that, despite the damping we introduce, the energy still shows some oscillations, which help the snake converge towards the proper configuration. By contrast, we have seen noticeably lower sub-pixel quality with the method in (Fua & Leclerc 1990).

4 Image representation

We address in this section problems related to the re-rendering process: how to represent the image, given the regions previously extracted, in order to be able to synthesise a new image as close as possible to the original image.

The previous stage provides us with a segmentation of the image: the image is decomposed into homogeneous regions. The region boundaries are at a sub-pixel resolution but expressed in terms of control points (the control points used to control the corresponding snake). The first stage is thus to convert these boundaries into vectorial form (Section 4.1). The second stage is to represent the interior of each region to keep enough information for a proper synthesis (Section 4.2).

4.1 Region boundaries

Snakes interpolate their control points. We thus have to generate an analytical equation representing a curve passing through these points. We use NURBS curves because of their ability to represent complex shapes with a reasonable number of parameters. The control points can easily be converted into a NURBS curve interpolating the control points and having a controlled continuity (Piegl & Tiller 1997).

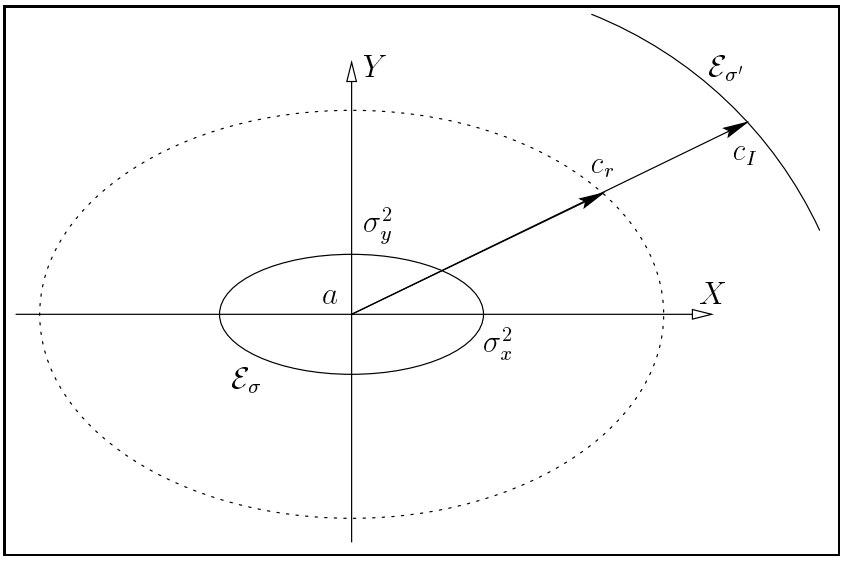


Figure 5: Projection of the image colours to make the region colours (see text)

4.2 Region interiors

Currently, two approaches are implemented. One is useful to render flat colours, the other to render smoothly varying colours.

In the case of flat colours, we simply associate with a region its average colour (taken from the corresponding region attribute). When rendered, such a region appears flat, which is perfect for non-realistic cartoon rendering.

In the case of smoothly varying colours, we use colours inside the regions taken from the original image. Each region is triangulated using a quality conforming Delaunay triangulation, where the area and angles of the triangles can be controlled (Shewchuk 1996). A colour is then associated with each vertex of the triangulation, the colour being a function $\mathcal{C}(c)$ of the colour of the nearest pixel in the original image. This function is defined as follows. In the CIE $L^*a^*b^*$ space, we can consider the ellipsoid whose centre is the region's average colour (a), whose axis are aligned with the space axis, and whose dimensions are specified by the region's colour variance (σ_{L^*}, σ_{a^*}, and σ_{b^*}). Figure 5 shows such an ellipsoid ($\mathcal{E}_\sigma$) in a 2D projection of the CIE $L^*a^*b^*$ space (the X and Y coordinates can be any of the L^*, a^*, or b^*). We define the inside-outside function as (Whaite & Ferrie 1991):

$$f(c,a,\sigma) = \left(\frac{c_{L^*} - a_{L^*}}{\sigma_{L^*}}\right)^2 + \left(\frac{c_{a^*} - a_{a^*}}{\sigma_{a^*}}\right)^2 + \left(\frac{c_{b^*} - a_{b^*}}{\sigma_{b^*}}\right)^2 .$$

We have $f(c,a,\sigma) = 1$ if c is on the ellipsoid of centre a and dimensions σ, $f(c,a,\sigma) > 1$ if c is outside the ellipsoid, and $f(c,a,\sigma) < 1$ if c is inside the ellipsoid. All image colours c_I that are inside or on the ellipsoid (*i.e.* $f(c_I,a,\sigma) \leq 1$) are taken without modification since they comply with the colour statistical properties of the region. Image colours outside of the ellipsoid (*i.e.* $f(c_I,a,\sigma) > 1$) are far from the possible region properties. This is possible partly because the relaxation has removed small regions inside large ones and partly because of the smooth transitions between colours in the original image. We thus have to modify these colours to make them more compliant with the region's properties. Projecting the colours

onto the ellipsoid $\mathcal{E}_\sigma$ results in a flat image (when re-rendered from the image representation). c_I defines an ellipsoid whose dimensions are σ_I: $f(c_I, a, \sigma') = 1$. We create the region colour c_r by projecting c_I onto an ellipsoid that is between the two. Its dimensions are σ_r are:

$$\sigma_r = \sigma + \xi(\sigma_I - \sigma),$$

where ξ is the contraction factor that controls the colour flatness, typically 0.85. To summarise:

$$\mathcal{C}(c) = \left| \begin{array}{ll} c_I & \text{if } f(c_I, a, \sigma) \leq 1, \\ c_r \mid f(c_r, a, \sigma + \xi(\sigma_I - \sigma)) = 1 & \text{if } f(c_I, a, \sigma) > 1. \end{array} \right.$$

By using this function, colours still reflect what was initially in the image but are closer to the region properties. This prevents colours very different from the region ones spreading into the region.

These colour projections are made in the CIE $L^*a^*b^*$ space (like all our colour manipulations). Indeed, they are not feasible in the RGB colour space since the projection (as well as the Euclidian distance) does not correspond to any psychophysical reality, thus creating intermediate colours that are visually wrong.

Note that the size of the triangles in the mesh must be of the order of the finest detail to be represented, which means that for highly textured regions, the number of primitives can be large.

The flat mesh produced can then be rendered using a standard mesh renderer such as OpenGL™. We currently use IRCS (Image Rendering and Compositing Software), a software package developed at the University of Bath, UK (Froumentin & Willis 1999). It allows the rendering of very large images (32k × 32k pixels) on (relatively) low memory computers. It can use NURBS as shape primitives and can render smoothly varying colours on primitives. Moreover, the file format used is particularly suitable for rendering only parts of the complete data and for modifying parameters of parts.

5 Results

We first present some results that assess the precision of the sub-pixel extraction (Section 5.1). We then show some steps of the construction of the representation for real images (Section 5.2) and, finally, we show some re-synthesised images (Section 5.3).

5.1 Precision of the sub-pixel extraction

Figures showing the results (Figures 6, 8, and 11) show the shape (in grey), the original NURBS curve used to generate the shape (dashed line, when visible), the snake at its initial position (jagged continuous line), and the snake at its final position (smooth continuous line). Error curves shown at Figures 7, 9, and 12 show the percentage (with respect to the area of the shape) of the reconstructed shape that is outside, inside, and both outside and inside of the true shape. These quantities are measured as follows. We generate on the same image at very high resolution the true shape and the reconstructed shape with two different colours and with appropriate mixing rules such that pixels belonging to one shape *or* the other are of the corresponding colour. Then we count the number of pixels having one or the other colour and the number of pixels in the true shape to obtain the percentages shown.

To get shapes closer to the sub-pixel curves, we anti-alias the image (as was the original image). Since we count pixels having *any* amount of one or the other colour as being inside or outside, the percentages we give are over-estimated. In fact, Figure 10 shows the total error depending on the image resolution at which it has been measured in a typical case (the CARDIO CIE $L^*a^*b^*$ image). All given values are measured at a resolution of 4000 pixels (in fact 4000×4000 pixels, the original images having 200×200 pixels).

5.1.1 Image CARDIO

The first image (Figure 6) shows a convex shape having a sharp angle. Figure 7 shows that

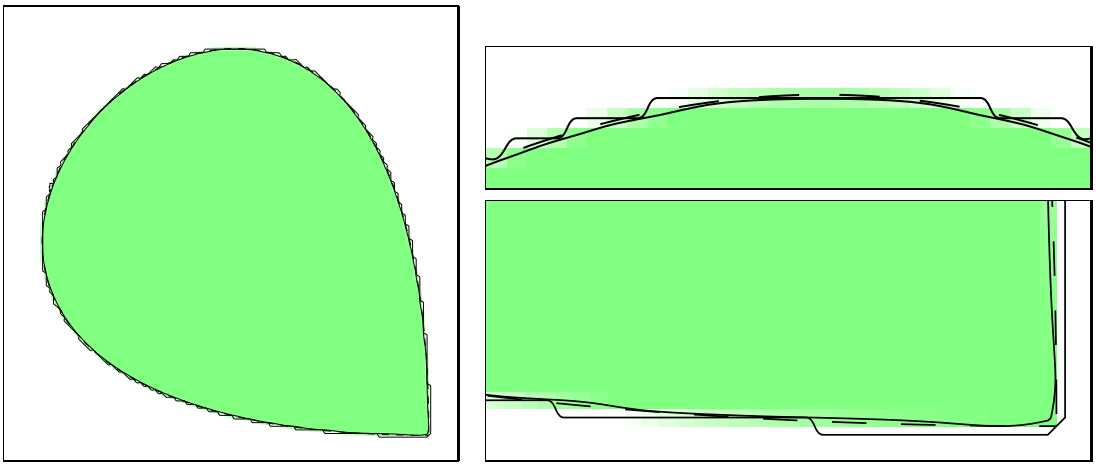


Figure 6: CARDIO

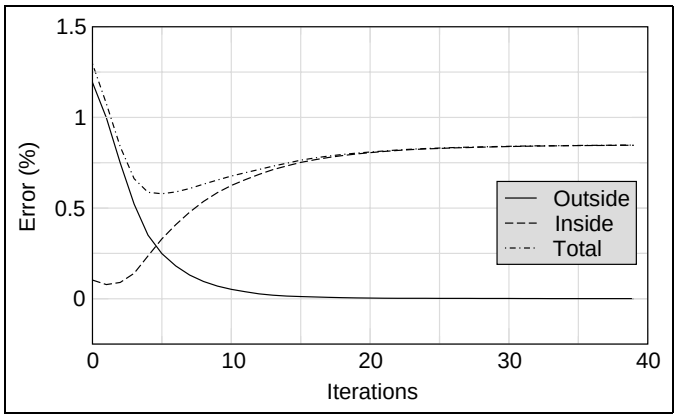


Figure 7: CARDIO: error between the reconstruction and the original

the resulting curve is completely inside the true curve. Figure 6 shows that this is partly due to the sharp angle but mainly due to the reconstructed curve being systematically inside the true curve. This systematic error is due to the anti-aliasing of the original image that has been done in the RGB space while all sub-pixel treatments are done in the CIE $L^*a^*b^*$ space. The experiment in Section 5.1.2 shows that this explanation is correct.

5.1.2 Image CARDIO CIE $L^*a^*b^*$

This experiment is the same as the one in Section 5.1.1 with the following difference: the anti-aliasing of the original image has been done in the CIE $L^*a^*b^*$ space instead of the RGB space. Figures 8 and 9 clearly show that the systematic error towards the inside of the shape has disappeared, but that the sharp angle is still reconstructed inside the original contour. This is due to natural tension in the snake. The angle is however correctly detected (an angle is present in the reconstructed contour).

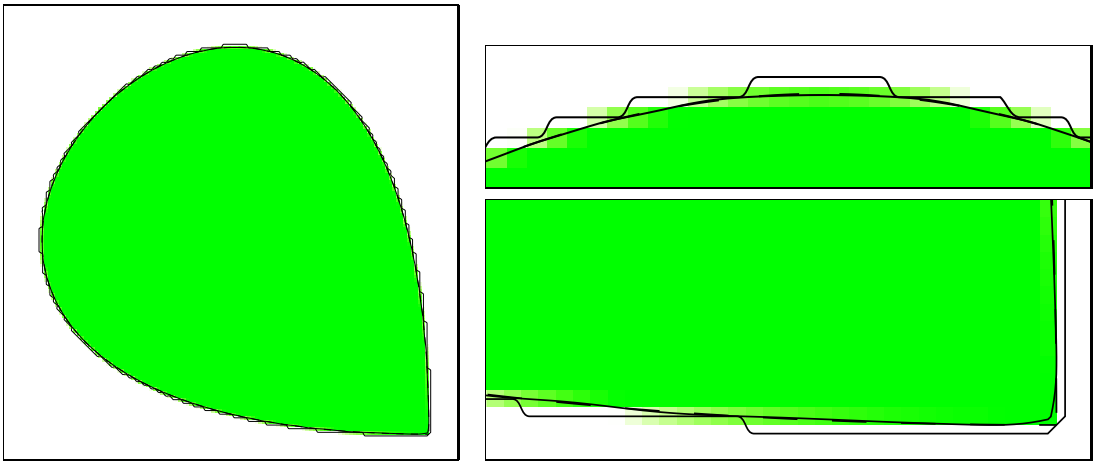


Figure 8: CARDIO CIE $L^*a^*b^*$

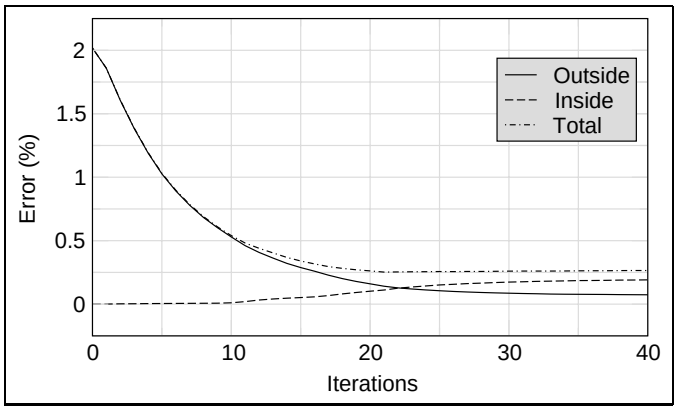


Figure 9: CARDIO CIE $L^*a^*b^*$: error between the reconstruction and the original

5.1.3 Image Y

In this experiment, we use a shape having no sharp angles but high curvatures and concavities. Results show that the concavities are no problem. We can see that the high curvature parts are still inside the original shape, but less than in the case of a sharp angle (a third of a pixel instead of a full pixel at the same resolution).

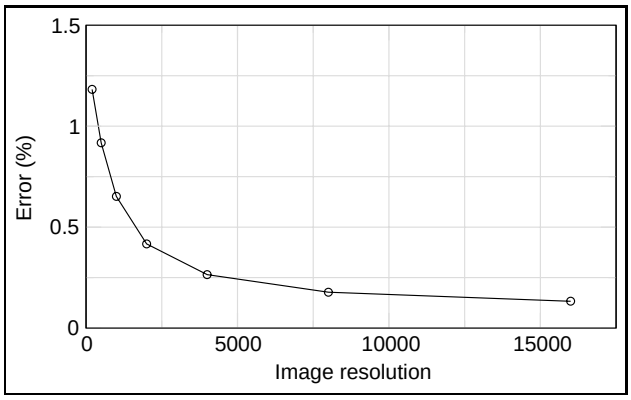


Figure 10: CARDIO CIE $L^*a^*b^*$: error variation depending on the image resolution

The higher total error for the Y image compared to the one for the CARDIO CIE $L^*a^*b^*$ image, despite the visually better result, comes from the fact that the curve to surface ratio is greater in the Y image than in the CARDIO CIE $L^*a^*b^*$ image.

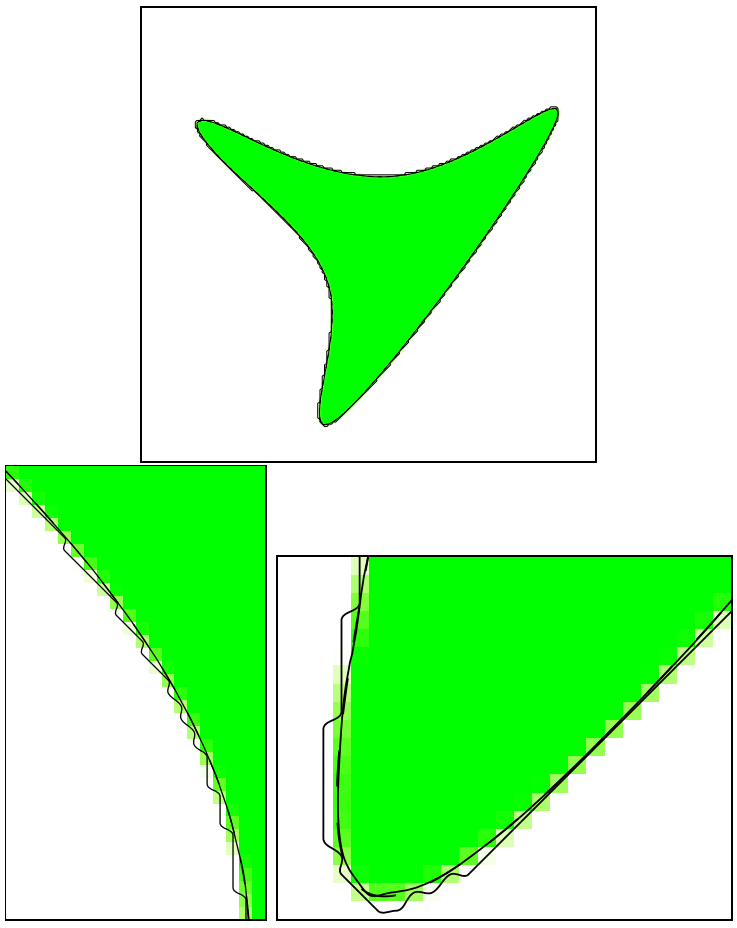


Figure 11: Y

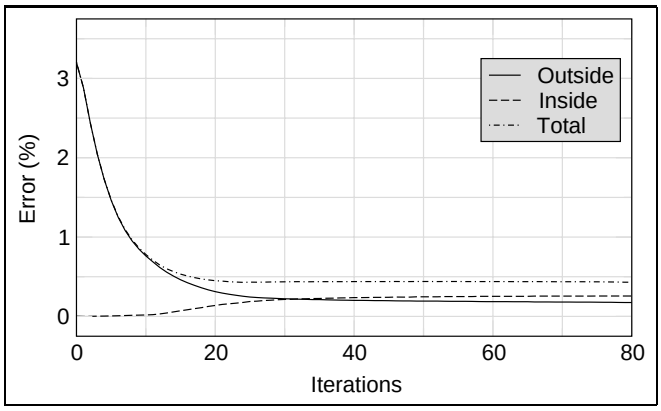


Figure 12: Y: error between the reconstruction and the original

5.1.4 Sub-pixel accuracy evaluation

We can see that the reconstructed contours are very close to the original ones. The total error is less than 0.5% of the shape's area. Moreover, errors are symmetric and the maximum error is about half a pixel at the resolution of the original image, this error happening only on sharp angles.

5.2 Steps in the representation construction

The first step is to specify (draw) image parts that are used to compute the region attributes (Section 2.2). Figure 13 shows the ones used for the GOOSE image. This drawing, which is

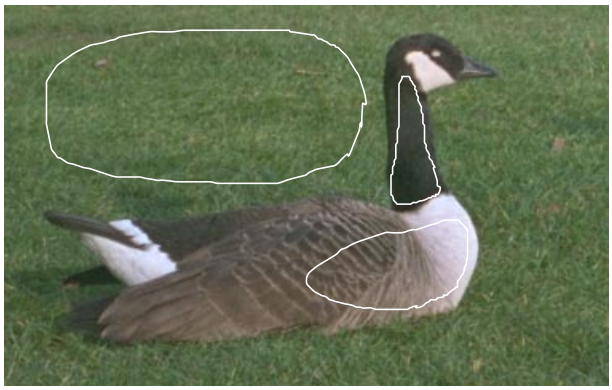
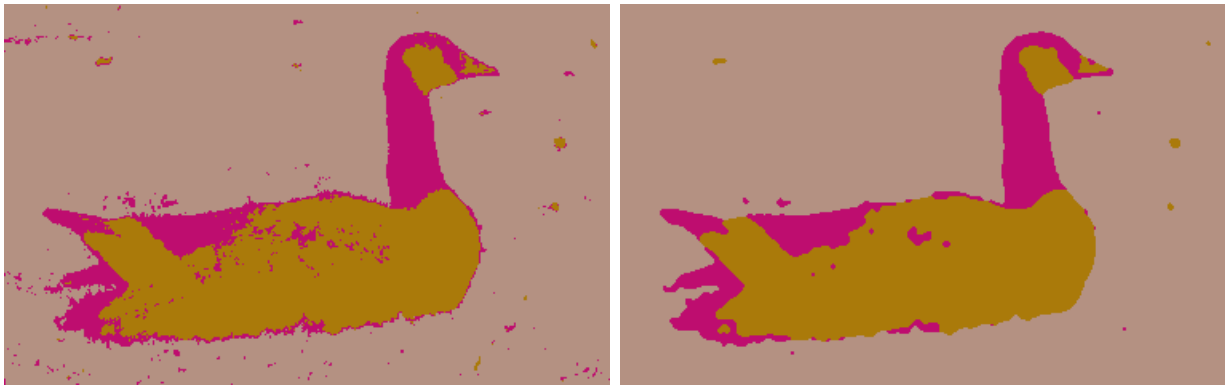


Figure 13: The image parts used to compute region attributes

the only necessary user intervention, is fast since the parts specification does not have to be precise.

Probabilities are then initialised (section 2.2) and optimised (Section 2.1) to produce the segmentation. Figure 14 shows the initial and final segmentations.

Once the final segmentation is obtained, the user can select the regions he wants in the final representation. This can be done manually by clicking in the desired regions or



(a) (b)

Figure 14: Segmentation of the GOOSE image: (a) initial and (b) final

automatically by providing a threshold on the area of the regions to consider. Then the snakes are created and optimised (Section 3.3 and Figure 15).

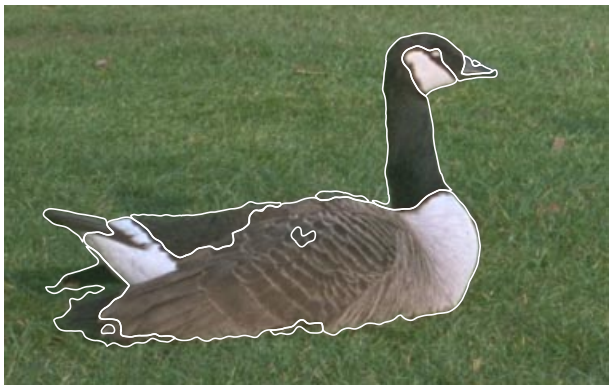


Figure 15: The snakes after convergence

Finally, the representation is built. For the goose, we used the smooth scheme to represent the regions (Section 4.2). Figure 16 shows the triangulation of the region corresponding to the neck of the goose (Figure 2).

Figure 17 shows all the stages as well as the type of information used and created to extract the sub-pixel boundaries of an image and to create its vectorial representation.

5.3 Re-synthesis

We show several original images as well as the re-synthesised version of them. As can be seen, both versions are very similar. There is a noticeable difference, though: the re-synthesised version lacks a great deal of the high frequencies present in the textured regions of the original images, even with the GOOSE image when triangles used to represent the image were no more than 2 pixels in area at the resolution of the original image (Figure 18 (c))[7]. Moreover, the

[7]Figure 16 shows the large triangles for the region of the neck.

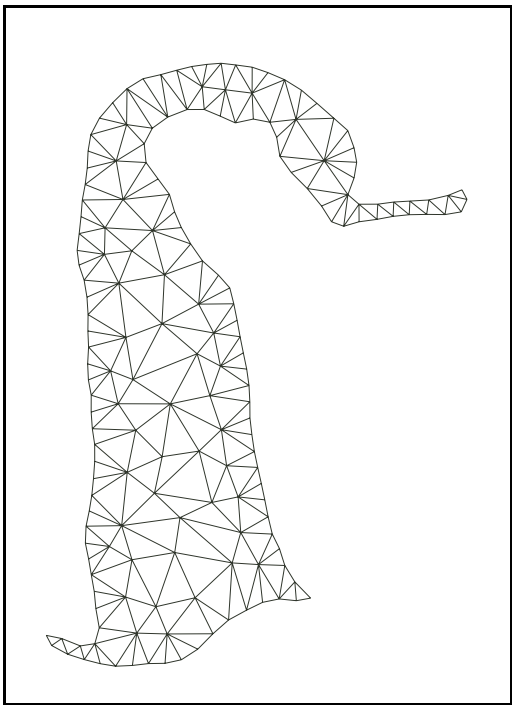


Figure 16: The triangulation of the goose's neck

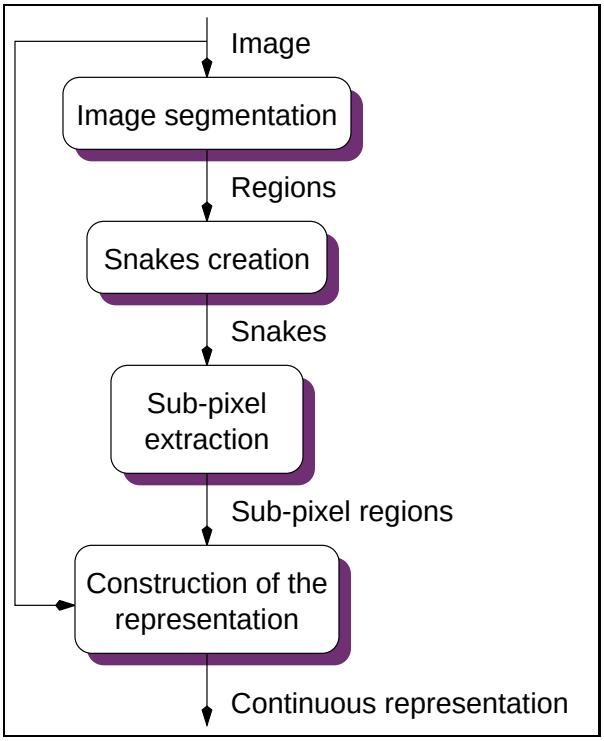
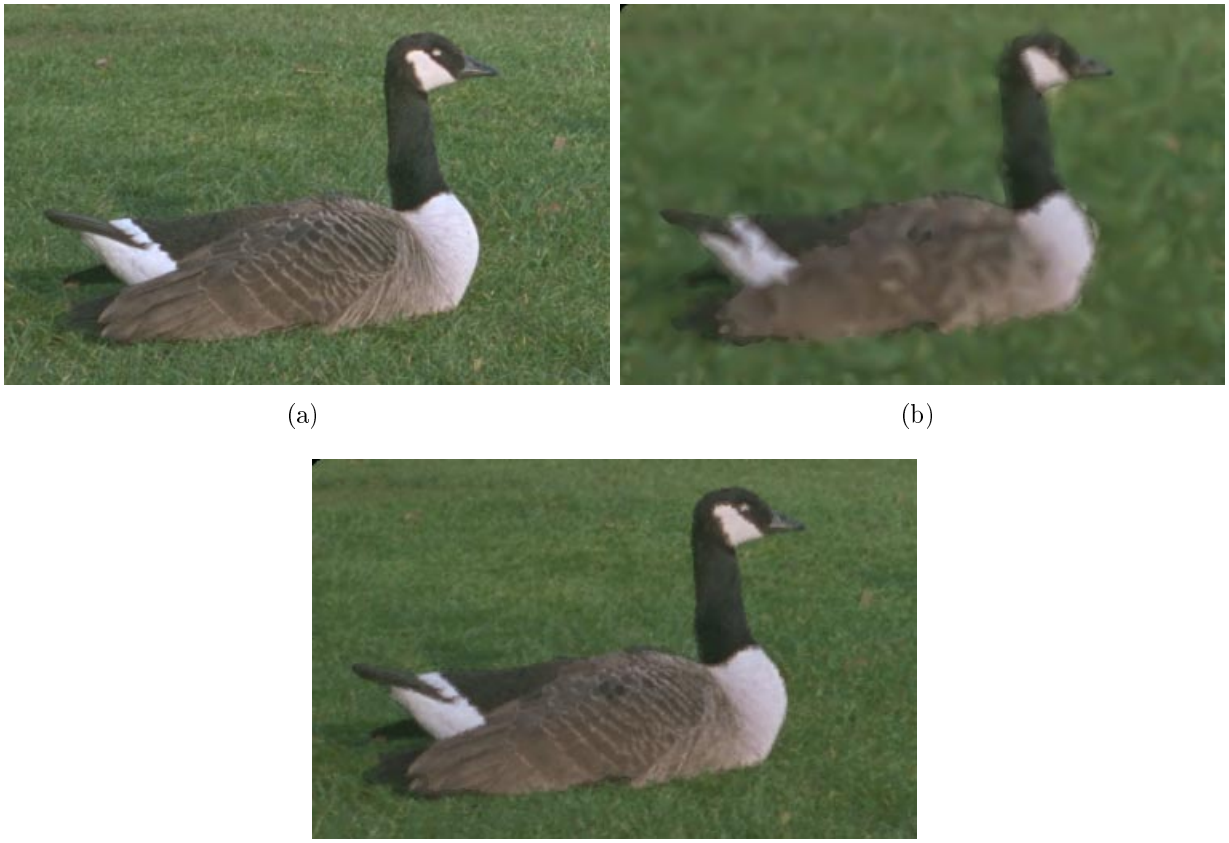


Figure 17: The whole process of the image representation

triangular structure of the data is evident in the result. This clearly shows the limits of the current scheme to represent textured regions. Note however that smooth regions are very



(a) (b)

(c)

Figure 18: The GOOSE image: (a) original, (b) re-synthesised from large triangles (area less than 50 pixels) and (c) from small triangles (area less than 2 pixels)

well handled. Note also that high frequencies are still present at the boundaries of regions, which is an improvement compared to traditional methods that change the resolution of images.

Figure 20 shows an object extracted from an image and synthesised on its own (without the background). This object is reused with the GOOSE image in Figure 21.

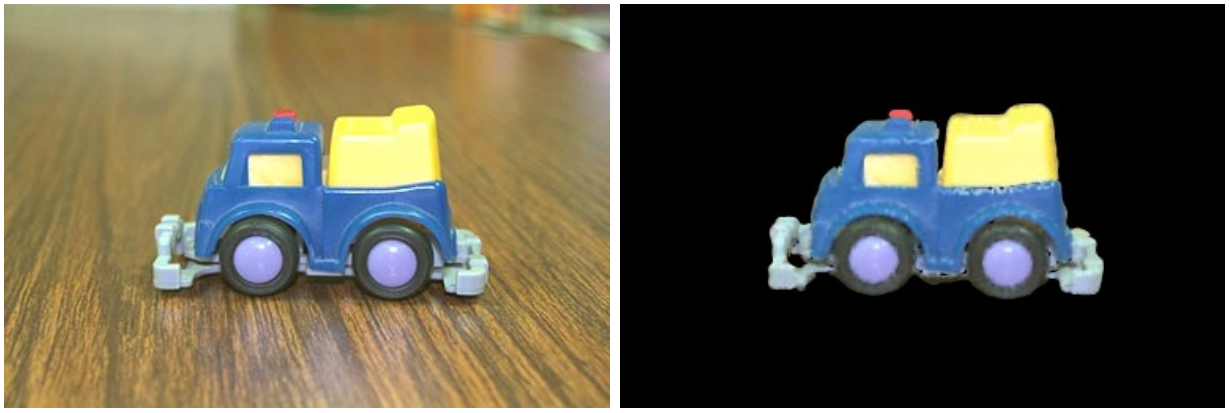
These images (as well as others) can be viewed in better condition (without the step of the printing) at the following address: `http://www.maths.bath.ac.uk/~masfl`.

6 Conclusion

We presented in this report a first step towards continuous image representations. To build such a representation, we first perform an image segmentation, using relaxation labelling, to decompose the image into homogeneous regions. The homogeneity is based on the average colour and colour variance in the regions. The segmentation is then used to initialise the snakes while the probabilities of the relaxation are used to make the snakes evolve towards the sub-pixel boundaries. The user has to draw free-form shapes on the image to define regions from which the statistical properties will be computed.

(a) (b)

Figure 19: The POND image: (a) original and (b) re-synthesised



(a) (b)

Figure 20: The BLUE TRUCK image: (a) original and (b) re-synthesised

Finally, the image representation is built using the sub-pixel boundaries. The repre-

Figure 21: Mixing image representations

sentation is made of NURBS curves to represent the boundaries and a region description is attached to each of these boundaries: a simple colour or a flat mesh with a colour at each vertex. This representation can be re-rendered to produce either an image close to the original or a controlled modification of it.

Results have shown that we are able to extract sub-pixel contours with a very good accuracy. Results of image re-synthesis have shown that we have a promising way of representing images but that other schemes must be developed to represent more accurately different kinds of regions, *e.g.* models based on texture representation. Moreover, it has been shown that these continuous representations are useful for applications like image warping (Froumentin, Labrosse & Willis 2000).

References

Amini, A. A., Weymouth, T. E. & Jain, R. C. (1990), 'Using dynamic programming for solving variational problems in vision', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **12**(9), 855–867.

Benson, A. & Evans, D. (1977), 'Algorithm 512: A normalized algorithm for the solution pf positive definite symmetric quindiagonal systems of linear equations', *ACM Transactions on Mathematical Software* **3**(1), 96–103.

Berger, M.-O. (1990), Snake growing, *in* 'Proceedings of the European Conference on Computer Vision', Antibes, France, pp. 570–572.

Berger, M.-O. & Mohr, R. (1990), Towards autonomy in active contour models, *in* 'Proceedings of the International Conference on Pattern Recognition', Vol. 1, Atlantic City, NJ, pp. 847–851.

Cohen, L. D. (1991), 'On active contour models and ballons', *CVGIP: Image Understanding* **53**(2), 211–218.

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

David, C. & Zucker, S. W. (1990), 'Potentials, valleys, and dynamic global coverings', *International Journal of Computer Vision* **5**(3), 219–238.

Duda, R. O. & Hart, P. E. (1973), *Pattern Classification and Scene Analysis*, John Wiley & Sons.

Froumentin, M., Labrosse, F. & Willis, P. (2000), 'Vector-based representation for image warping', *Computer Graphics Forum* **19**(3). To appear.

Froumentin, M. & Willis, P. (1999), 'An efficient $2\frac{1}{2}$D rendering and compositing system', *Computer Graphics Forum* **18**(3), C385–C394 and C428.

Fua, P. & Leclerc, Y. (1990), 'Model driven edge detection', *Machine Vision and Applications* **3**, 45–56.

Garbay, C. (1986), 'Image structure representation and processing: A discussion of some segmentation methods in cytology', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **PAMI-8**(2), 140–146.

Geiger, D., Gupta, A., Costa, L. A. & Vlontzos, J. (1995), 'Dynamic programming for detecting, tracking, and matching deformable contours', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **17**(3), 294–302.

Gunn, S. R. & Nixon, M. S. (1997), 'A robust snake implementation; a dual active contour', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **19**(1), 63–68.

Hansen, M. W. & Higgins, W. E. (1997), 'Relaxation methods for supervised image segmentation', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **19**(9), 949–962.

Henricsson, O. (1998), 'The role of color attributes and similarity grouping in 3-D building reconstruction', *Computer Vision and Image Understanding* **72**(2), 163–184.

Hummel, R. A. & Zucker, S. W. (1983), 'On the foundations of relaxation labelling processes', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **PAMI-5**(3), 267–287.

Kass, M., Witkin, A. & Terzopoulos, D. (1988), 'Snakes: Active contour models', *International Journal of Computer Vision* **1**(4), 321–331.

Kittler, J. & Illingworth, J. (1985), 'Relaxation labelling algorithms — a review', *Image and Vision Computing* **3**(4), 206–216.

Lai, K. F. & Chin, R. T. (1995), 'Deformable contours: Modeling and extraction', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **17**(11), 1084–1090.

Meyer, G. W. & Greenberg, D. P. (1987), Perceptual color spaces for computer graphics, *in* H. J. Durret, ed., 'Color and the Computer', Academic Press, Inc., San Diego, CA, USA, pp. 83–100.

Piegl, L. & Tiller, W. (1997), *The NURBS Book*, 2nd edn, Springer-Verlag, Berlin, Germany.

Richards, J., Landgrebe, D. & Swain, P. (1981), 'On the accuracy of pixel relaxation labeling', *IEEE Transactions on Systems, Man, and Cybernetics* **SMC-11**(4), 303–309.

Rosenfeld, A., Hummel, R. A. & Zucker, S. W. (1976), 'Scene labelling by relaxation operations', *IEEE Transactions on Systems, Man, and Cybernetics* **SMC-6**(6), 420–433.

Rueckert, D. & Burger, P. (1995), Contour fitting using an adaptive spline model, *in* 'Proceedings of the British Machine Vision Conference', Vol. 1, Birmingham, UK, pp. 207–216.

Shewchuk, J. R. (1996), Triangle: Engineering a 2D quality mesh generator and delaunay triangulator, *in* M. C. Lin & D. Manocha, eds, 'Applied Computational Geometry: Towards Geometric Engineering', Vol. 1148 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 203–222. From the First ACM Workshop on Applied Computational Geometry.

Whaite, P. & Ferrie, F. P. (1991), 'From uncertainty to visual exploration', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **13**(10), 1038–1049.

Williams, D. J. & Shash, M. (1992), 'A fast algorithm for active contours and curvature estimation', *CVGIP: Image Understanding* **55**(1), 14–26.

Wyszecki, G. & Stiles, W. S. (1982), *Color Science: Concepts and Methods, Quantitative Data and Formulae*, 2nd edn, John Wiley & Sons.

Zucker, S. W. (1985), 'Early orientation selection: tangent fields and the dimensionality of their support', *Computer Vision, Graphics, and Image Processing* **32**, 74–103.