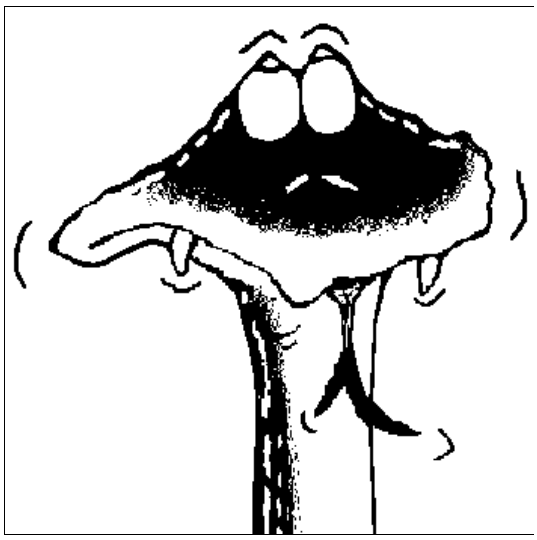
EVERYTHING YOU ALWAYS WANTED TO KNOW ABOUT SNAKES (BUT WERE AFRAID TO ASK)

Jim Ivins† & John Porrill

AIVRU Technical Memo #86, July 1993
(Revised June 1995; March 2000)

Artificial Intelligence Vision Research Unit
University Of Sheffield
England S10 2TP



"Reeling and Writhing, of course, to begin with," the Mock Turtle replied;
"and then the different branches of Arithmetic – Ambition, Distraction,
Uglification, and Derision."

Alice's Adventures In Wonderland, by Lewis Carroll

---

† Please send feedback and corrections regarding this document to Jim Ivins.

E-mail: ivinsj@cs.curtin.edu.au.

ABSTRACT

Active contour models – known colloquially as *snakes* – are energy-minimising curves that deform to fit image features. Snakes lock on to nearby minima in the *potential energy* generated by processing an image. (This energy is minimised by iterative *gradient descent* according to forces derived using *variational calculus* and *Euler-Lagrange Theory*.) In addition, *internal (smoothing) forces* produce *tension* and *stiffness* that constrain the behaviour of the models; *external* forces may be specified by a supervising process or a human user. As is characteristic of gradient descent, the energy minimisation process is unfortunately prone to oscillation unless precautions – typically the use of small time steps – are taken.

Active contour models provide a unified solution to several image processing problems such as the detection of light and dark lines, edges, and terminations; they can also be used in stereo matching, and for segmenting spatial and temporal image sequences. Snakes have often been used in medical research applications; for example, in reconstructing three-dimensional features from planar slices of volume data such as NMR or CT images. In addition, many motion tracking systems use snakes to model moving objects. The main limitations of the models are (i) that they usually only incorporate edge information (ignoring other image characteristics) possibly combined with some prior expectation of shape; and (ii) that they must be initialised close to the feature of interest if they are to avoid being trapped by other local minima.

KEY WORDS

- ❑ Calculus Of Variations
- ❑ Euler-Lagrange Theory
- ❑ Gradient Descent
- ❑ Snakes: Active Contour Models

1 INTRODUCTION

Low-level visual tasks such as edge detection and stereo matching are often treated as autonomous bottom-up processes. However, this sequential approach propagates mistakes to higher processes without providing opportunities for correction. A more attainable goal for low-level processing is to provide several interpretations of the image data, from which higher processes† or a human user may choose. *Active contour models* – first described by **Kass et al (1987; 1988)** – provide one possible method for generating these alternative interpretations.

Active contour models are often called *snakes* because they appear to slither across images (a phenomenon known as *hysteresis*); they are one example of the general technique of matching a *deformable model* to an image using *energy minimisation*. From any starting point, subject to certain constraints, a snake will deform into alignment with the nearest salient feature in a suitably processed image; such features correspond to local minima in the energy generated by processing the image. Snakes thus provide a low-level mechanism that seeks appropriate local minima rather than searching for a global solution. In addition, high-level mechanisms can interact with snakes – for example, to guide them towards features of interest. Unlike most other techniques for finding image features, snakes are always minimising their energy. Changes in high-level interpretation can therefore affect a snake during the minimisation process, and even in the absence of such changes the model will still exhibit hysteresis in response to a moving stimulus.

Snakes do not try to solve the entire problem of finding salient image features; they rely on other mechanisms to place them somewhere near a desired solution. For example, automatic initialisation procedures can use standard image processing techniques to locate features of interest that are then refined using snakes. Even in cases where automatic initialisation is not possible, however, active contour models can still be used for image interpretation. An expert user need only push a snake towards an image feature, and the energy minimisation process will fit the model to the data. This behaviour has been exploited in numerous interactive image processing systems – for example, see **Kass et al (1987; 1988); Hill et al (1992); Porrill and Ivins (1994)**.

A snake is typically driven by a *potential energy* generated by processing the underlying image data. For example, Gaussian smoothing followed by convolution with a

---

† **Scott (1987)** is critical of this philosophy because the unspecified high-level process rarely materialise except in human form!

gradient-squared operator generates a potential in which extrema correspond to edges in the original image. Over a series of iterations the force generated by this energy drives the snake into alignment with the nearest salient edge. However, the snake must also satisfy some *internal constraints* – for example, it must be smooth and continuous in outline. Sometimes the user imposes additional *external constraints* such as attraction or repulsion.

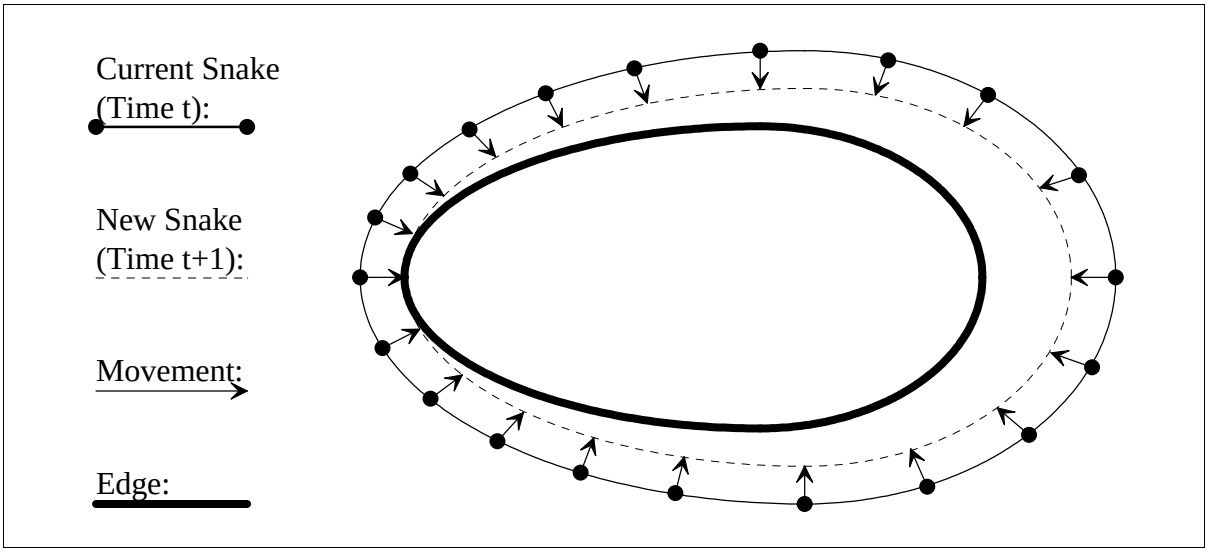


Figure 1: A Closed Active Contour Model. This diagram shows a snake with its ends joined so that it forms a closed loop. Over a series of time steps the snake moves into alignment with the nearest salient feature (in this case an edge).

Both internal and external energy constraints are discussed in Section 2; potential energy is discussed in Section 3. Section 4 uses these energy terms to derive explicit forces that can be used to drive active contour models to minimise their energy by iterative *gradient descent*. The original (semi-implicit) method proposed by **Kass et al (1987)**, which is related to the explicit use of forces, is then described in the next two sections. First, the *calculus of variations* is used to derive the *Euler-Lagrange equation* in Section 5; this equation is then used to find the minimum energy condition for an active contour model. Section 6 explains how to solve the minimum energy equation using a semi-implicit relaxation method based on a fast matrix inversion algorithm. (Both the explicit and semi-implicit methods use finite differences to compute derivatives as described in Appendix A; Appendix B contains six mathematical notes that provide simple background information.) Oscillation, the main drawback of relaxation methods, is discussed in Section 7. Finally, Section 8 considers the use of inter-snake energy terms in three of the most common applications – stereo matching, and segmentation of spatial and temporal image sequences.

2 SNAKE ENERGY FUNCTIONALS

A snake is a parametric contour that deforms over a series of iterations (time steps). Each element **x** along the contour therefore depends on two parameters:

$$\mathbf{x}(s,t) \quad \begin{cases} s = \text{ space (curve) parameter} \\ t = \text{ time (iteration) parameter} \end{cases}$$

The contour is influenced by internal and external constraints, and by image forces, as outlined below.

- ❑ **Internal forces.** Internal constraints give the model tension and stiffness.
- ❑ **External forces.** External constraints come from high-level sources such as human operators or automatic initialisation procedures.
- ❑ **Image forces.** Image energy is used to drive the model towards salient features such as light and dark regions, edges, and terminations.

Representing a snake parametrically as explained in Mathematical Note 1, $\mathbf{x}(s) = (x(s), y(s))$ where s is usually taken to vary between 0 and 1. The total energy of the model E_{snake} is given by the sum of the energy for the individual snake elements:†

$$E_{snake} = \int_0^1 E_{element}(\mathbf{x}(s)) \; ds \qquad (2.1)$$

The integral notation used in this section implies an open-ended snake; however, joining the first and last elements makes the snake into a closed loop as shown in Figure 1.

Equation 2.1 can be rewritten in terms of three basic energy *functionals*:††

$$E_{snake} = \int_0^1 E_{intern}(\mathbf{x}) \; ds + \int_0^1 E_{extern}(\mathbf{x}) \; ds + \int_0^1 E_{image}(\mathbf{x}) \; ds \qquad (2.2)$$

The curve parameter s is omitted where no ambiguity arises. The gradients of the three energy functionals in Equation 2.2 correspond to the three forces listed above. The internal and external energy functionals are considered in more detail below; image (potential) energy is dealt with in the next section.

---

† To implement an active contour model in computer software the continuous representation is approximated discretely by N snake elements; however, continuous notation is used wherever possible because of its greater mathematical elegance.

†† A functional is a function of one or more functions, giving a scalar result.

2.1 INTERNAL (INTRA-SNAKE) ENERGY

Using subscripts to indicate derivatives, the internal energy of a snake element is defined as:

$$E_{intern}(\mathbf{x}) = \underbrace{\alpha(s)\,|\mathbf{x}_s(s)|^2}_{\text{Tension}} + \underbrace{\beta(s)\,|\mathbf{x}_{ss}(s)|^2}_{\text{Stiffness}} \tag{2.3}$$

This energy contains a first-order term controlled by $\alpha(s)$, and a second-order term controlled by $\beta(s)$. The first-order term makes the snake contract like an elastic band by introducing *tension*; the second-order term makes it resist bending by producing *stiffness*. In other words, the parametric curve is predisposed to have constant (preferably zero) 'velocity' and 'acceleration' with respect to its parameter.

In the absence of other constraints, an active contour model simply collapses to a point like a strip of infinitely-elastic material; however, if the ends of the model are anchored then it forms a straight line along which the elements are evenly spaced. Adjusting the weights $\alpha(s)$ and $\beta(s)$ controls the relative importance of the tension and stiffness terms. For example, setting $\beta(s) = 0$ in one part of the model allows it to become second-order discontinuous and develop a corner. For simplicity, the tension and stiffness weightings are assumed to be uniform throughout the remainder of this document, so that $\alpha(s) = \alpha$ and $\beta(s) = \beta$.

2.2 EXTERNAL (EXTRA-SNAKE) ENERGY

Both automatic and manual supervision can be used to control *attraction* and *repulsion* forces that drive active contour models to or from specified features. For example, a spring-like attractive force can be generated between a snake element and a point **i** in an image using the following external energy term:

$$E_{extern}(\mathbf{x}) = k\,|\mathbf{i} - \mathbf{x}|^2 \tag{2.4}$$

This energy is minimal (zero) when $\mathbf{x} = \mathbf{i}$, and it takes the value of k when $\mathbf{i} - \mathbf{x} = \pm 1$ as shown in Figure 2. Mathematical Note 2 reviews the properties of extrema in functions.

An external energy term E_{extern} can also be used to make part of an image repel an active contour model:

$$E_{extern}(\mathbf{x}) = \frac{k}{|\mathbf{i} - \mathbf{x}|^2} \tag{2.5}$$

This energy is maximal (infinite) when $\mathbf{x} = \mathbf{i}$; it is unity when $\mathbf{i} - \mathbf{x} = \pm k$. Because of the singularity, the repulsion term must be clipped as the denominator approaches zero.

Negating the (positive) constant k in these equations converts attraction to pseudo-repulsion, and repulsion to pseudo-attraction; however, these pseudo energy terms are unusable because their minima are infinite. (During energy minimisation, the singularities completely dominate the behaviour of an active contour model, at the expense of all other energy terms.) The forces produced by these energy terms are easily found by differentiation.

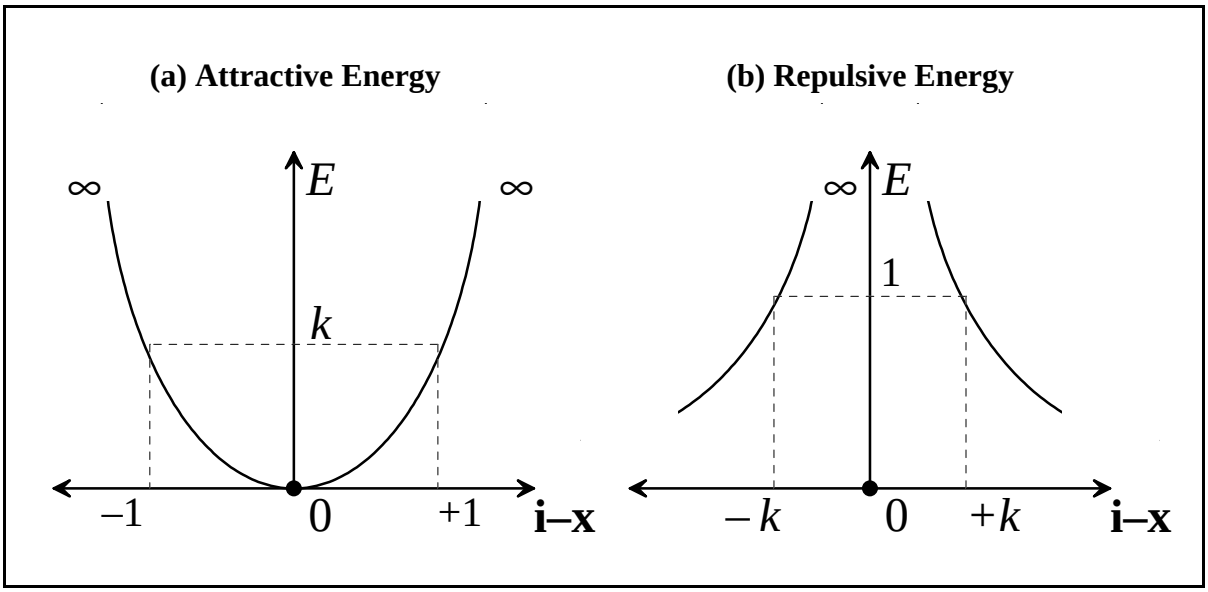


Figure 2: Attraction And Repulsion Energy. These graphs show the attractive and repulsive energy terms. Both functionals have maximal values that are infinite; the minima are zero.

3 IMAGE (POTENTIAL) ENERGY FUNCTIONALS

The potential energy P generated by processing an image $I(x, y)$ produces a force that can be used to drive snakes towards features of interest. Three different potential (image) energy functionals are described below; these attract snakes to lines, edges, and terminations. The total potential energy can be expressed as a weighted combination of these functionals:

$$P = E_{image} = w_{line}E_{line} + w_{edge}E_{edge} + w_{term}E_{term} \quad (3.1)$$

The nearest local minimum the potential energy can be found using gradient descent as described in Section 4:

$$\mathbf{x} \rightarrow \mathbf{x} + \delta\mathbf{x} \quad (3.2)$$

The image forces $\delta\mathbf{x}$ produced by each of the terms in Equation 3.1 are derived below, in advance of the main discussion of energy minimisation and forces in Sections 4–6.

If just a small portion of an active contour model finds a low-energy image feature then the internal constraints will pull neighbouring elements towards that feature. This effect can be enhanced by spatially smoothing the potential energy field. Typically, a snake is first allowed to reach equilibrium on a very smooth potential; the blurring is then gradually reduced – see **Witkin et al (1986)**. At very coarse scales the snake does a poor job of localising features, and fine detail is lost; however, it is attracted to local minima from far away. Reducing the amount of blurring allows the snake to form a more accurate model of the underlying image.

3.1 REGION FUNCTIONAL

The simplest potential energy is the unprocessed image intensity so that $P(\mathbf{x}) = I(\mathbf{x})$:

$$E_{line} = \int_0^1 I(\mathbf{x}(s)) \, ds \quad (3.3)$$

According to the sign of w_{line} in Equation 3.1, the snake will be attracted either to light or dark regions of the image.

Using ∇ to indicate image gradient, the corresponding image force $\delta\mathbf{x}$ is given by:

$$\delta\mathbf{x} \propto -\frac{\partial P}{\partial \mathbf{x}} = -\frac{\partial I}{\partial \mathbf{x}} = -\nabla I(\mathbf{x})$$

Local minima in the image intensity can therefore be found by taking small steps in $\mathbf{x}$:

$$\mathbf{x} \rightarrow \mathbf{x} - \tau \nabla I(\mathbf{x}) \quad (3.4)$$

The positive time step τ is chosen to suit the problem domain; however, it is almost invariably one or two orders of magnitude less than unity to prevent oscillation (see Section 7). For an

extension of this idea for segmenting textures and colours see the work on active region models by **Ivins and Porrill (1995)**.

3.2 EDGE FUNCTIONAL

By far the most common use for active contour models is as semi-global edge-detectors that minimise a potential energy in which minima correspond to strong edges – see Figure 3.

(a) Unprocessed Image **(b) Potential (Edge) Energy**

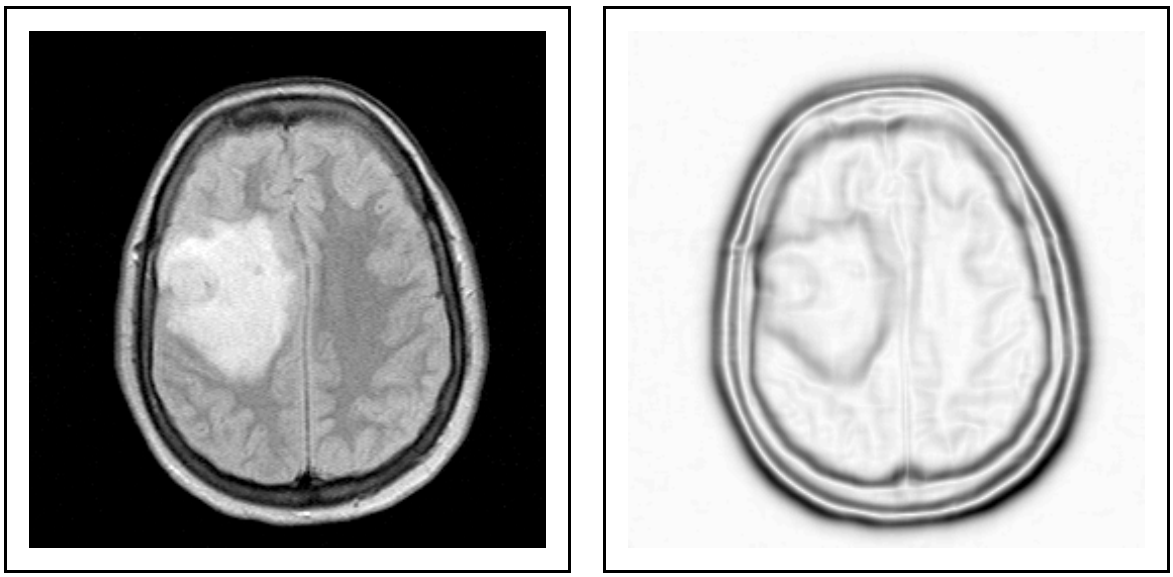


Figure 3: Potential (Edge) Energy. (**a**) An unprocessed 256-by-256 pixel NMR image. (**b**) The potential energy generated by smoothing the image, convolving it with a simple gradient operator, and negating the result (the image has been re-scaled for display). Strong edges produce correspondingly low (dark) local minima; however, fine detail is lost during the smoothing process, which is necessary to eliminate noise and spread out legitimate edges.

Edges can be found with a gradient-based potential energy functional such as:

$$E_{edge} \quad = \quad -\int_0^1 \left| \frac{\partial I}{\partial \mathbf{x}} \right|^2 \quad ds \tag{3.5}$$

For example, consider a snake element $\mathbf{x} = (x, y)$ with potential energy $P(\mathbf{x}) = -|\nabla I(\mathbf{x})|^2$; the image force acting on this element is given by:

$$\delta\mathbf{x} \quad \propto \quad -\frac{\partial P}{\partial \mathbf{x}} \quad = \quad \frac{\partial}{\partial \mathbf{x}}(|\nabla I|^2) \quad = \quad 2\,\nabla\nabla I(\mathbf{x})\,\nabla I(\mathbf{x})$$

The term $\nabla\nabla I(\mathbf{x})$ is the Hessian matrix of second-order image derivatives. Strong edges can therefore be found using:

$$\mathbf{x} \quad \rightarrow \quad \mathbf{x} \quad + \quad \tau\,\nabla\nabla I(\mathbf{x})\,\nabla I(\mathbf{x}) \tag{3.6}$$

3.3 TERMINATION FUNCTIONAL

The ends of line segments, and therefore corners, can be found using an energy term based on the curvature of lines in a slightly smoothed image $C(x, y) = G_\sigma(x, y) * I(x, y)$. If the gradient direction is given by $\theta = \tan^{-1}(C_y / C_x)$ then the unit vectors along, and perpendicular to, the image gradient are given by:

$$\text{Tangent:} \quad \mathbf{n} = \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix} \qquad \text{Normal:} \quad \mathbf{n}_\perp = \begin{pmatrix} -\sin\theta \\ \cos\theta \end{pmatrix}$$

The curvature of a contour in $C(x, y)$ can be written:

$$E_{term} = \int_0^1 \frac{\partial\theta}{\partial\mathbf{n}_\perp} \, ds = \int_0^1 \frac{\partial^2 C / \partial\mathbf{n}_\perp^2}{\partial C / \partial\mathbf{n}} \, ds \tag{3.7}$$

Expanding the derivatives:

$$E_{term} = \int_0^1 \frac{C_{yy}C_x^2 + C_{xx}C_y^2 - 2C_{xy}C_xC_y}{\left(C_x^2 + C_y^2\right)^{3/2}} \, ds \tag{3.8}$$

This energy formula provides a simple means for attracting snakes towards corners and terminations.

4 GRADIENT DESCENT USING FORCES

The previous two sections can be summarised by stating that, at its simplest, the energy E of an active contour model $\mathbf{x}(s)$ is defined as:†

$$E(\mathbf{x}(s)) = \underbrace{\int_0^1 P(\mathbf{x}(s))\ ds}_{\text{Potential}} + \underbrace{\frac{\alpha}{2}\int_0^1 \left|\frac{\partial \mathbf{x}(s)}{\partial s}\right|^2 ds}_{\text{Tension}} + \underbrace{\frac{\beta}{2}\int_0^1 \left|\frac{\partial^2 \mathbf{x}(s)}{\partial s^2}\right|^2 ds}_{\text{Stiffness}} \qquad (4.1)$$

This section considers the task of minimising these energy functionals. First, the general technique of minimisation by iterative gradient descent is introduced; an equation is then derived to describe the energy changes that occur when an active contour model is moved, and this equation is used to calculate forces for energy minimisation by gradient descent.

4.1 CONJUGATE GRADIENT DESCENT

In general, an energy function $E(\mathbf{x})$ can be minimised by altering each variable according some small quantity $\delta\mathbf{x}$ that is guaranteed to reduce the value of the function:

$$\mathbf{x} \leftarrow \mathbf{x} + \delta\mathbf{x} \qquad (4.2)$$

Local linear approximation gives an expression for the new energy:

$$E(\mathbf{x}+\delta\mathbf{x}) \approx E(\mathbf{x}) + \frac{\partial E}{\partial \mathbf{x}} \cdot \delta\mathbf{x} \qquad (4.3)$$

Clearly, $\delta\mathbf{x}$ must be chosen so that the energy decreases at each iteration. The gradient descent rule is based on the fact that steps down an energy hypersurface (see Figure 4) can be guaranteed by making small changes in the direction of the negated gradient:

$$\delta\mathbf{x} \propto -\frac{\partial E}{\partial \mathbf{x}} \qquad (4.4)$$

The new value of the energy function is given by:

$$E(\mathbf{x}+\delta\mathbf{x}) \approx E(\mathbf{x}) - \tau\left(\frac{\partial E}{\partial \mathbf{x}}\right)^2 \qquad (4.5)$$

The negative sign and square power (dot product) in this equation guarantee that E will decrease at each iteration until the minimum is reached; however, the (small) time step τ must be chosen carefully to avoid oscillation (see Section 7) and is almost invariable less than unity.

Conjugate gradient descent, as illustrated in Figure 4, finds the nearest local minimum in an energy hypersuface, with no consideration of global properties. Unfortunately, this

† For simplicity, external constraints are omitted from the remainder of this document.

simplicity can lead to problems when there are several minima close together because a snake can be attracted to a feature (energy minimum) other than that intended by the user.

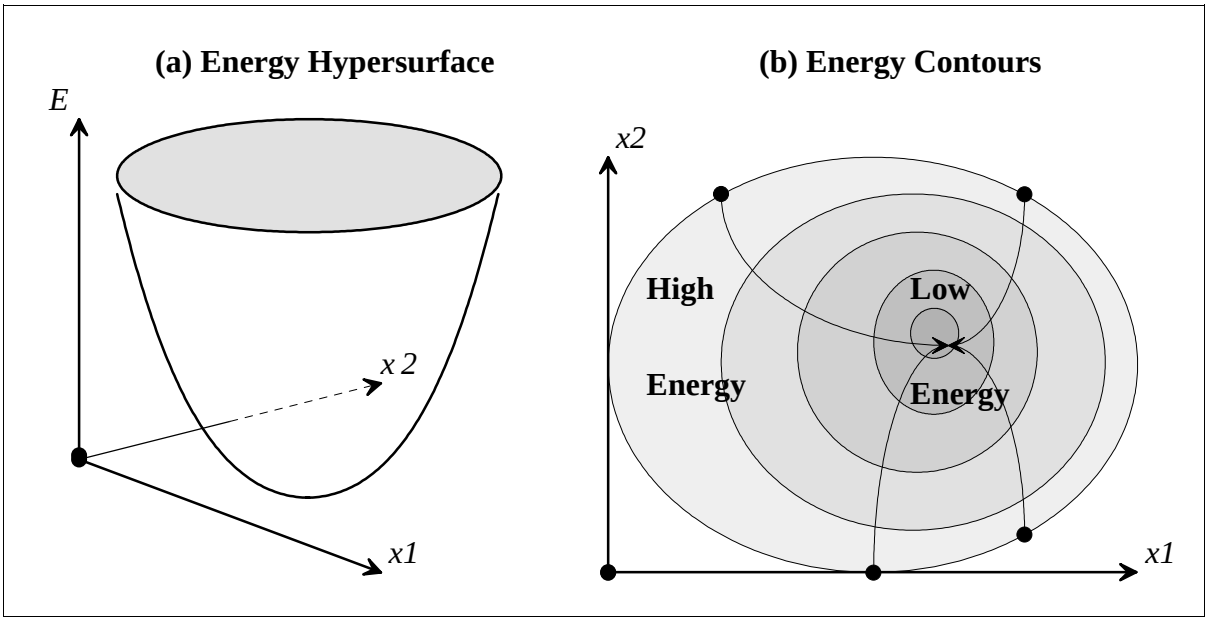


Figure 4: Conjugate Gradient Descent. This figure shows four alternative paths down a three-dimensional energy surface. At each iteration the gradient descent algorithm moves the energy value towards the nearest local minimum by making a small change in the direction given by the negated energy gradient (orthogonal to the local energy contours). The process is repeated until this gradient (force) is zero, at which point none of the variables can be altered without increasing the energy.

4.2 ENERGY GRADIENT FOR AN ACTIVE CONTOUR MODEL

Before gradient descent can be used to minimise the energy of an active contour model it is necessary to obtain an expression for the corresponding energy gradient which determines the changes that are made to the model (forces) at each iteration.

From Equation 4.1 the basic energy of a closed active contour model† is given by:

$$E(\mathbf{x}) = \oint P(\mathbf{x})\,ds + \frac{\alpha}{2}\oint |\mathbf{x}'|^2 ds + \frac{\beta}{2}\oint |\mathbf{x}''|^2 ds \qquad \begin{cases} \mathbf{x} \equiv \mathbf{x}(s) \\ \mathbf{x}' \equiv \partial\mathbf{x}/\partial s \\ \mathbf{x}'' \equiv \partial^2\mathbf{x}/\partial s^2 \end{cases} \qquad (4.6)$$

Note the use of dashes to indicate derivatives. The ends of this model are joined so that it forms a closed loop; in the discrete approximation to this equation, the first and last of the N snake elements are consecutive so that $\mathbf{x}(0) \equiv \mathbf{x}(N)$.

---

† The energy functionals in this version of the equation are integrated around a closed snake as shown by the integral signs; this removes the need to specify limits.

If the snake changes slightly then the total energy of the new configuration is:

$$E(\mathbf{x}+\delta\mathbf{x}) \quad = \quad \oint P(\mathbf{x}+\delta\mathbf{x})\, ds \quad + \quad \frac{\alpha}{2}\oint |\mathbf{x}' + \delta\mathbf{x}'|^2 ds \quad + \quad \frac{\beta}{2}\oint |\mathbf{x}'' + \delta\mathbf{x}''|^2 ds \qquad (4.7)$$

This equation can be simplified using the following approximations:

$$P(\mathbf{x}+\delta\mathbf{x}) \quad = \quad P(\mathbf{x}) \quad + \quad \delta P(\mathbf{x}) \quad \approx \quad P(\mathbf{x}) \quad + \quad \frac{\partial P}{\partial \mathbf{x}} \cdot \delta\mathbf{x}$$

$$|\mathbf{x}+\delta\mathbf{x}|^2 \quad = \quad \mathbf{x}\cdot\mathbf{x} \quad + \quad 2\mathbf{x}\cdot\delta\mathbf{x} \quad + \quad \underbrace{\delta\mathbf{x}\cdot\delta\mathbf{x}}_{\text{Negligible}} \quad \approx \quad \mathbf{x}^2 \quad + \quad 2\mathbf{x}\cdot\delta\mathbf{x}$$

Equation 4.7 therefore simplifies to:

$$E+\delta E \quad \approx \quad \oint P(\mathbf{x}) + \frac{\partial P}{\partial \mathbf{x}} \cdot \delta\mathbf{x} \quad ds$$

$$+ \quad \frac{\alpha}{2}\oint \mathbf{x}'^2 + 2\mathbf{x}' \cdot \delta\mathbf{x}' \quad ds \quad + \quad \frac{\beta}{2}\oint \mathbf{x}''^2 + 2\mathbf{x}'' \cdot \delta\mathbf{x}'' \quad ds \qquad (4.8)$$

Subtracting 4.6 from 4.8 gives an approximation for the energy change that arises from a small adjustment to the configuration of the snake:

$$\delta E \quad = \quad \oint \frac{\partial P}{\partial \mathbf{x}} \cdot \delta\mathbf{x} \quad ds \quad + \quad \alpha \oint \mathbf{x}' \cdot \delta\mathbf{x}' \quad ds \quad + \quad \beta \oint \mathbf{x}'' \cdot \delta\mathbf{x}'' \quad ds \qquad (4.9)$$

This approximation is simplified using integration by parts (see Mathematical Note 5) to eliminate $\delta\mathbf{x}'$ and $\delta\mathbf{x}''$:

$$\delta E \quad = \quad \oint \frac{\partial P}{\partial \mathbf{x}} \cdot \delta\mathbf{x} \quad ds \quad - \quad \alpha \oint \mathbf{x}'' \cdot \delta\mathbf{x} \quad ds \quad + \quad \beta \oint \mathbf{x}'''' \cdot \delta\mathbf{x} \quad ds \qquad (4.10)$$

Equation 4.10 can be factorised to give a simple expression that includes the energy gradient:

$$\delta E \quad = \quad \oint \left(\frac{\partial P}{\partial \mathbf{x}} \quad - \quad \alpha\, \mathbf{x}'' \quad + \quad \beta\, \mathbf{x}'''' \right) \cdot \delta\mathbf{x} \quad ds \qquad (4.11)$$

Negating this expression gives the local direction of steepest descent down the energy hyper-surface; however, it does not indicate how far to move and must be treated with caution since it is only a local description of the surface.

4.3 FORCES

Assuming it is not already at a minimum, the energy of a snake will decreases at each iteration if $\delta\mathbf{x}$ is a negated fraction (the time step δt, which must be positive) of the energy gradient given by Equation 4.11:

$$\delta\mathbf{x} \quad = \quad -\delta t \left(\frac{\partial P}{\partial \mathbf{x}} \quad - \quad \alpha\, \mathbf{x}'' \quad + \quad \beta\, \mathbf{x}'''' \right) \qquad (4.12)$$

Substituting this expression back into Equation 4.11 gives:

$$\delta E = -\delta t \oint \left(\frac{\partial P}{\partial \mathbf{x}} - \alpha \mathbf{x}'' + \beta \mathbf{x}'''' \right)^2 ds \tag{4.13}$$

The iterative rule for conjugate gradient descent is therefore:

$$\mathbf{x} \leftarrow \mathbf{x} + \delta \mathbf{x} \qquad \frac{\delta \mathbf{x}}{\delta t} = -\frac{\partial E}{\partial \mathbf{x}} = \alpha \mathbf{x}'' - \beta \mathbf{x}'''' - \frac{\delta P}{\delta \mathbf{x}} \tag{4.14}$$

At the limit of infinitesimal steps:

$$\frac{\partial \mathbf{x}}{\partial t} = \underbrace{\alpha \frac{\partial^2 \mathbf{x}}{\partial s^2}}_{\text{Tension Force}} - \underbrace{\beta \frac{\partial^4 \mathbf{x}}{\partial s^4}}_{\text{Stiffness Force}} - \underbrace{\frac{\partial P}{\partial \mathbf{x}}}_{\text{Image Force}} \tag{4.15}$$

The energy of an active contour model can therefore be minimised by calculating this resultant force for, and applying it to, each snake element in turn.

Note that in mechanical systems, force is the product of mass and acceleration:

$$\mathbf{f} = m \frac{\partial^2 \mathbf{x}}{\partial t^2}$$

However, in Equation 4.15 force and velocity are equivalent:

$$\mathbf{f} = \frac{\partial \mathbf{x}}{\partial t}$$

An active contour model driven using this equation therefore behaves as though travelling in a viscous medium such that inertia is negligible and movement with constant velocity requires a constant force to be applied.

(a) Initial Snake **(b) Final Snake**

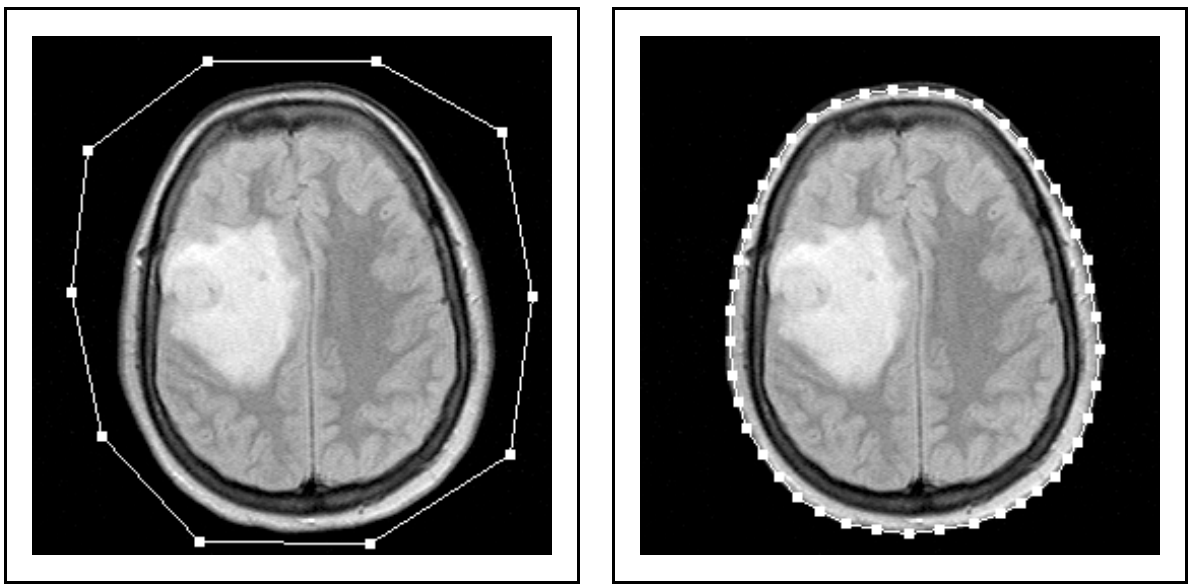


Figure 5: An Active Contour Model. This figure shows two views of an MR image (the potential energy generated by smoothing the image and convolving it with a simple gradient

operator is shown in Figure 3). (**a**) An initial snake configuration marked by the user. (**b**) The final snake configuration after energy minimisation by gradient descent; the snake is modelling the skin over the skull. (Note that the snake has been re-parameterised during energy minimisation; this process is not discussed further in this document.)

Gradient descent using explicit forces is not the only way to minimise the energy of an active contour model. For example, dynamic programming was proposed by **Amini et al (1988)** as a method for finding minima that are guaranteed to be global within some predetermined search range; however, this method suffers from increased computational complexity and will not be discussed further in this document. The semi-implicit method originally used by **Kass et al (1987)** is a faster alternative that relies on an efficient matrix inversion algorithm to solve a set of simultaneous equations by relaxation; the solutions to these equations describe the minimum energy state of an active contour model. The semi-implicit method is described in the next two sections.

5 CALCULUS OF VARIATIONS

This section uses variational calculus to derive the Euler-Lagrange equation, which describes extrema in functionals; this equation is then used to obtain an equation that describes the minimum energy condition of an active contour model.

5.1 THE EULER-LAGRANGE EQUATION

Consider the problem of minimising (or maximising) a functional E such as:

$$E(y) \quad = \quad \int_a^b F(\, x,\, y(x),\, y'(x)\,) \quad dx \tag{5.1}$$

(The independent variable x can be omitted where there is no ambiguity). Making a small change δy to the value of the function y generates a corresponding change in E:

$$E(y+\delta y) \quad = \quad \int_a^b F(\, x,\, y+\delta y,\, y'+\delta y'\,) \quad dx \tag{5.2}$$

Using the Taylor expansion (see Mathematical Notes 3 and 4) and ignoring terms above first order:

$$E+\delta E \quad \approx \quad \int_a^b F \quad + \quad \frac{\partial F}{\partial y}\,\delta y \quad + \quad \frac{\partial F}{\partial y'}\,\delta y' \quad dx \tag{5.3}$$

Subtracting 5.1 from 5.3 gives:

$$\delta E \quad \approx \quad \int_a^b \frac{\partial F}{\partial y}\,\delta y \quad + \quad \frac{\partial F}{\partial y'}\,\delta y' \quad dx \tag{5.4}$$

Eliminating $\delta y'$ using integration by parts as described in Mathematical Note 5:

$$\delta E \quad \approx \quad \int_a^b \frac{\partial F}{\partial y}\,\delta y \quad - \quad \frac{d}{dx}\left(\frac{\partial F}{\partial y'}\right)\delta y \quad dx \tag{5.5}$$

At extrema in E a small change in y produces almost no change in the value of the functional:

$$\int_a^b\left(\frac{\partial F}{\partial y} \quad - \quad \frac{d}{dx}\left(\frac{\partial F}{\partial y'}\right)\right)\delta y \quad dx \quad \approx \quad 0 \tag{5.6}$$

As δy is known to be non-zero, Equation 5.6 gives rise to the *Euler-Lagrange Equation* which is satisfied at extrema in F:

$$\frac{\partial F}{\partial y} \quad - \quad \frac{d}{dx}\left(\frac{\partial F}{\partial y'}\right) \quad = \quad 0 \tag{5.7}$$

Of course, the extremum described by the Euler-Lagrange equation could be a maximum or a point of inflection rather than a minimum. If necessary, the second-order partial derivative (which will be positive at minima, negative at maxima, and zero at points of inflection) can sometimes be calculated to resolve the ambiguity.

To summarise, for a small change δy to a functional $F(x, y, y')$:

$$\delta E = \int_a^b \frac{\delta E}{\delta y(x)}\,\delta y \quad dx \qquad \frac{\delta E}{\delta y(x)} = \frac{\partial F}{\partial y} - \frac{d}{dx}\left(\frac{\partial F}{\partial y'}\right)$$

The *first variation* $\delta / \delta y(x)$ plays the equivalent role for functionals that the first derivative d / dx plays for functions, so that $\delta E / \delta y(x) = 0$ at extrema; it can therefore be used to find the minimum energy condition for a snake.

5.2 MINIMA IN SNAKE ENERGY FUNCTIONALS

From Equation 4.1 the basic energy of an active contour model is given by:

$$E(\mathbf{x}(s)) = \int_0^1 F(s, \mathbf{x}(s), \mathbf{x}'(s), \mathbf{x}''(s))\; ds \quad \begin{cases} \mathbf{x} \equiv \mathbf{x}(s) \\ \mathbf{x}' \equiv \partial\mathbf{x}/\partial s \\ \mathbf{x}'' \equiv \partial^2\mathbf{x}/\partial s^2 \end{cases} \tag{5.8}$$

Consider the effect of a small change in the vector $\mathbf{x}$:

$$E(\mathbf{x}+\delta\mathbf{x}) = \int_0^1 F(\mathbf{x}+\delta\mathbf{x},\quad \mathbf{x}'+\delta\mathbf{x}',\quad \mathbf{x}''+\delta\mathbf{x}'')\quad ds \tag{5.9}$$

Using the Taylor expansion:

$$E+\delta E \approx \int_0^1 \left(F + \frac{\partial F}{\partial \mathbf{x}}\cdot\delta\mathbf{x} + \frac{\partial F}{\partial \mathbf{x}'}\cdot\delta\mathbf{x}' + \frac{\partial F}{\partial \mathbf{x}''}\cdot\delta\mathbf{x}''\right)\quad ds \tag{5.10}$$

Subtracting 5.8 from 5.10:

$$\delta E \approx \int_0^1 \frac{\partial F}{\partial \mathbf{x}}\cdot\delta\mathbf{x} + \frac{\partial F}{\partial \mathbf{x}'}\cdot\delta\mathbf{x}' + \frac{\partial F}{\partial \mathbf{x}''}\cdot\delta\mathbf{x}''\quad ds \tag{5.11}$$

Terms in $\delta\mathbf{x}'$ and $\delta\mathbf{x}''$ are eliminated using integration by parts:

$$\delta E \approx \int_0^1 \frac{\partial F}{\partial \mathbf{x}}\cdot\delta\mathbf{x} - \frac{d}{ds}\left(\frac{\partial F}{\partial \mathbf{x}'}\right)\cdot\delta\mathbf{x} + \frac{d^2}{ds^2}\left(\frac{\partial F}{\partial \mathbf{x}''}\right)\cdot\delta\mathbf{x}\quad ds \tag{5.12}$$

Factorising:

$$\delta E \approx \int_0^1 \left(\frac{\partial F}{\partial \mathbf{x}} - \frac{d}{ds}\left(\frac{\partial F}{\partial \mathbf{x}'}\right) + \frac{d^2}{ds^2}\left(\frac{\partial F}{\partial \mathbf{x}''}\right)\right)\cdot\delta\mathbf{x}\quad ds \tag{5.13}$$

This yields the Euler-Lagrange equation for extrema in E:

$$\frac{\partial F}{\partial \mathbf{x}} - \frac{d}{ds}\left(\frac{\partial F}{\partial \mathbf{x}'}\right) + \frac{d^2}{ds^2}\left(\frac{\partial F}{\partial \mathbf{x}''}\right) = 0 \tag{5.14}$$

Again, this equation describes all types of extrema, not just minima. Fortunately, when minimising the energy of a snake the ambiguity is easily resolved by changing the sign of each term in the equations of motion.

If the functional F is to represent the potential energy, tension and stiffness of a snake then:

$$F = P(\mathbf{x}) + \frac{\alpha}{2}\mathbf{x}'^2 + \frac{\beta}{2}\mathbf{x}''^2 \tag{5.15}$$

Assuming α and β are constants, the partial derivatives for Equation 5.14 are as follows:

$$\frac{\partial F}{\partial \mathbf{x}} = \frac{\partial P}{\partial \mathbf{x}} \qquad \frac{\partial F}{\partial \mathbf{x}'} = \alpha\,\mathbf{x}' \qquad \frac{\partial F}{\partial \mathbf{x}''} = \beta\,\mathbf{x}''$$

Combining 5.14 and 5.15 gives the minimal energy condition for a snake:

$$\underbrace{\frac{\partial P}{\partial \mathbf{x}} - \alpha\,\mathbf{x}'' + \beta\,\mathbf{x}''''}_{\text{Energy Gradient}} = 0 \tag{5.16}$$

Unfortunately, these equations are difficult to solve analytically because **x** must be known before $\partial P/\partial \mathbf{x}$ can be found. However, the equations can be solved using semi-implicit relaxation methods as described in Section 6.

6 SEMI-IMPLICIT MINIMISATION

Section 5 showed that, at equilibrium, each element in a snake satisfies a vector equation which states that it does not move during time steps:

$$\frac{\partial \mathbf{x}}{\partial t} = \alpha \frac{\partial^2 \mathbf{x}}{\partial s^2} - \beta \frac{\partial^4 \mathbf{x}}{\partial s^4} - \frac{\partial P}{\partial \mathbf{x}} = 0 \qquad (6.1)$$

The energy of the model can therefore be minimised by solving all of these equations simultaneously using semi-implicit relaxation methods.

The vector terms in Equation 6.1 are separable into x and y components. Writing u_j where $j = 0, 1, N–1$ as a discrete approximation for $x(s)$ or $y(s)$, and using superscript t to denote iteration, Equation 6.1 becomes:

$$\frac{\partial u_j^t}{\partial t} = \alpha \frac{\partial^2 u_j^t}{\partial s^2} - \beta \frac{\partial^4 u_j^t}{\partial s^4} - \frac{\partial P}{\partial u_j^t} \qquad (6.2)$$

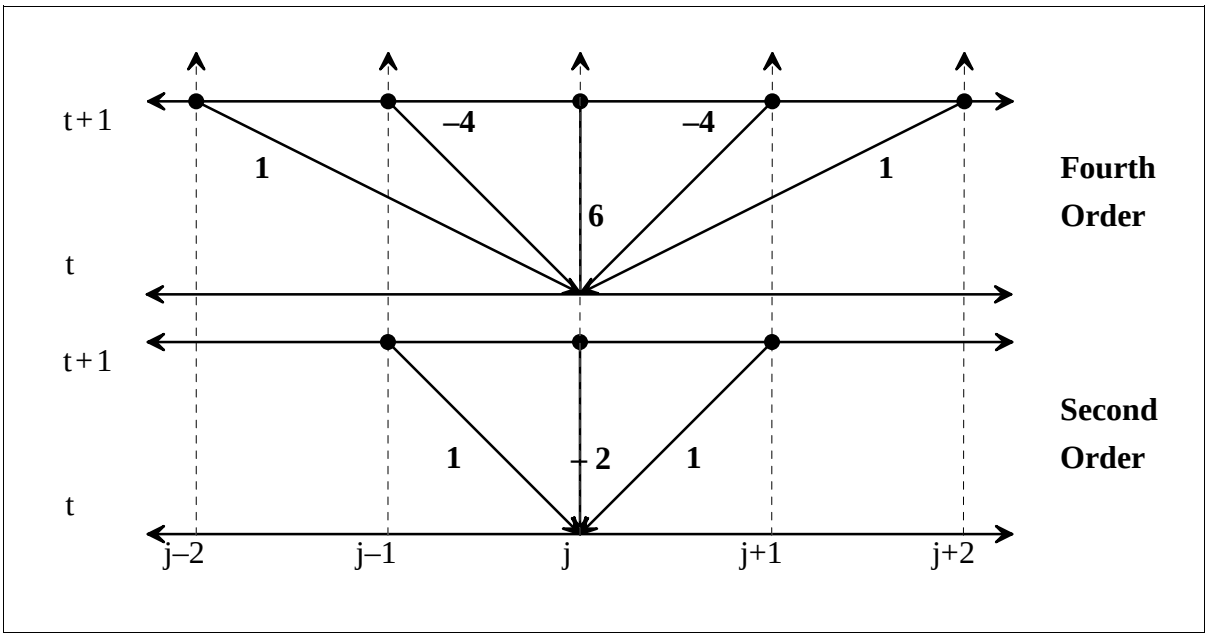


Figure 6: Approximating Derivatives With Finite Differences. The second-order derivative (tension force) is approximated over three elements; the fourth-order derivative (stiffness force) is approximated over five elements. In the semi-implicit method these derivatives are regarded as estimates for the next time step.

The derivatives in Equation 6.2 are estimated using *finite differences* as shown in Figure 6 and Appendix A:

$$\frac{\partial u}{\partial t} \rightarrow \frac{u_j^{t+1} - u_j^t}{\delta t} \qquad \frac{\partial^2 u}{\partial s^2} \rightarrow \frac{u_{j+1}^{t+1} + u_{j-1}^{t+1} - 2u_j^{t+1}}{\delta s^2}$$

$$\frac{\partial^4 u}{\partial s^4} \rightarrow \frac{u_{j+2}^{t+1} - 4u_{j+1}^{t+1} + 6u_j^{t+1} - 4u_{j-1}^{t+1} + u_{j-2}^{t+1}}{\delta s^4}$$

Note that the second and fourth derivatives are estimated as though at the *next* time step (t+1)†. Combining these approximations gives the finite difference equation:

$$\frac{u_j^{t+1} - u_j^t}{\delta t} = \frac{\alpha}{\delta s^2}\left(u_{j+1}^{t+1} + u_{j-1}^{t+1} - 2u_j^{t+1}\right) - \frac{\beta}{\delta s^4}\left(u_{j+2}^{t+1} - 4u_{j+1}^{t+1} + 6u_j^{t+1} - 4u_{j-1}^{t+1} + u_{j-2}^{t+1}\right) - \frac{\partial P}{\partial u_j^t} \tag{6.3}$$

Moving terms that cannot be estimated at time t over to the LHS gives:

$$bu_{j+2}^{t+1} - (a+4b)u_{j+1}^{t+1} + (1+2a+6b)u_j^{t+1} - (a+4b)u_{j-1}^{t+1} + bu_{j-2}^{t+1} = u_j^t + \delta t \frac{\partial P}{\partial u_j^t} \qquad \text{Note: } \begin{cases} a \equiv \alpha\,\delta t/\delta s^2 \\ b \equiv \beta\,\delta t/\delta s^4 \end{cases} \tag{6.4}$$

The RHS of Equation 6.4 can be evaluated using the potential energy at time t:

$$pu_{j+2}^{t+1} + qu_{j+1}^{t+1} + ru_j^{t+1} + qu_{j-1}^{t+1} + pu_{j-2}^{t+1} = \tilde{u}_j^{t+1} \qquad \begin{cases} p \equiv b \\ q \equiv -a-4b \\ r \equiv 1+2a+6b \end{cases} \tag{6.5}$$

$$\tilde{u}_j^{t+1} = u_j^t + \delta t \frac{\partial P}{\partial u_j^t}$$

This equation leads to a set of $2N$ simultaneous linear equations (for the x and y co-ordinates of each element in the snake) that can be written in standard matrix form.

$$\underbrace{\begin{bmatrix} r & q & p & & & p & q \\ q & r & q & p & & & p \\ p & q & r & q & p & & \\ & \ddots & \ddots & \ddots & \ddots & \ddots & \\ & & p & q & r & q & p \\ p & & & p & q & r & q \\ q & p & & & p & q & r \end{bmatrix}}_{\mathbf{M}} \underbrace{\begin{pmatrix} u_0^{t+1} \\ u_1^{t+1} \\ u_2^{t+1} \\ \vdots \\ u_{N-3}^{t+1} \\ u_{N-2}^{t+1} \\ u_{N-1}^{t+1} \end{pmatrix}}_{\mathbf{u}^{t+1}} = \underbrace{\begin{pmatrix} \tilde{u}_0^{t+1} \\ \tilde{u}_1^{t+1} \\ \tilde{u}_2^{t+1} \\ \vdots \\ \tilde{u}_{N-3}^{t+1} \\ \tilde{u}_{N-2}^{t+1} \\ \tilde{u}_{N-1}^{t+1} \end{pmatrix}}_{\tilde{\mathbf{u}}^{t+1}} \tag{6.6}$$

The constant values making up the matrix **M** are as follows:

$$p \equiv \beta \frac{\delta t}{\delta s^4} \qquad q \equiv -\alpha \frac{\delta t}{\delta s^2} - 4\beta \frac{\delta t}{\delta s^4} \qquad r \equiv 1 + 2\alpha \frac{\delta t}{\delta s^2} + 6\beta \frac{\delta t}{\delta s^4}$$

---

† This mathematical 'trick' produces a set of equations that describe the behaviour of the model over time. The idea is to move the snake according to the image forces at the current time step, and then to smooth the resulting model immediately at the start of the next iteration. At equilibrium these processes cancel out.

Each row of the matrix can be thought of as a convolution mask for evaluating the derivatives; the vectors represent the positions of the snake elements, both before and after adjustment to conform with the internal forces. Multiplying both sides of Equation 6.6 by the inverse of **M** gives the final solution (see Mathematical Note 6):

$$\mathbf{u}^{t+1} = \mathbf{M}^{-1}\left(\mathbf{u}^{t} + \delta t \frac{\partial P}{\partial \mathbf{u}^{t}}\right) \tag{6.7}$$

Note that **M** is a *cyclic symmetric pentadiagonal banded matrix* which can be inverted using the algorithm described by **Benson and Evans (1973; 1977)** making the solution of Equation 6.7 an O(*N*) process rather than O(N^3). If the tension and stiffness parameters and the number of elements are constant then the inverse matrix need only be calculated once.

```
#define N

const alpha=1.0, beta=0.5; // tension, stiffness
const ds=1.0, ds2=ds*ds, dt=0.05; // space, time

double x[N], y[N]; // snake

// code to create snake here

do {
     // external step
     for(int j=0; j<N; j++) {
          x[j] += dt * fx(x[j], y[j]); // image force
          y[j] += dt * fy(x[j], y[j]);
     }

     // internal step
     a=alpha*dt/ds2; b=beta*dt/ds2 // NB: constants?
     p=b; q=-a-4b; r=1+2a+6b;

     pentadiagonal_solve(p, q, r, x, n);
     pentadiagonal_solve(p, q, r, y, n);
}
while(!equilibrium);
```

Algorithm 1: Semi-Implicit Energy Minimisation. This C-style pseudo-code outlines the semi-implicit algorithm for minimising the energy of an active contour model. The co-ordinates of the `N` snake elements are specified by the arrays `x[s]` and `y[s]`. The external step moves each element according to the image forces computed using the (undefined) functions `fx()` and `fy()`. The internal step then smoothes the model by solving a set of equations in the form of a pentadiagonal banded matrix (see Equation 6.6). The process is repeated until equilibrium is detected in some way. (The effects of the internal and external steps cancel out at equilibrium.)

The semi-implicit energy minimisation process is summarised in Algorithm 1. Each iteration takes implicit Euler steps with respect to the internal energy, and explicit Euler steps with respect to the external and image energy. The minimisation process is therefore stable in the presence of very high tension and stiffness. Furthermore, with ordinary relaxation methods the propagation of forces along a snake is slow; however, the semi-implicit procedure allows forces to travel arbitrary distances in a single O(N) iteration.

7 SMOOTHING AND OSCILLATION

Unfortunately, there are problems with the iterative minimisation techniques described in Sections 4–6. The disruptive effects of unwanted local minima have already been discussed in Section 4. In addition, to ensure that an active contour model will remain stable throughout the minimisation process, the time step at each iteration must be very small relative to the amount of smoothing dictated by the energy gradient. The smoothing process, and the inherent problem of oscillation, are discussed below.

7.1 SMOOTHING

A function *y* can be smoothed over a series of iterations according to the *diffusion equation*:

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial s^2} \tag{7.1}$$

The process described by this equation eliminates peak values in *u* and makes the amplitude of high frequencies decay very quickly towards zero; as a result, the function *u* is smoothed to a line. The process is equivalent to smoothing an active region model with a tension force.

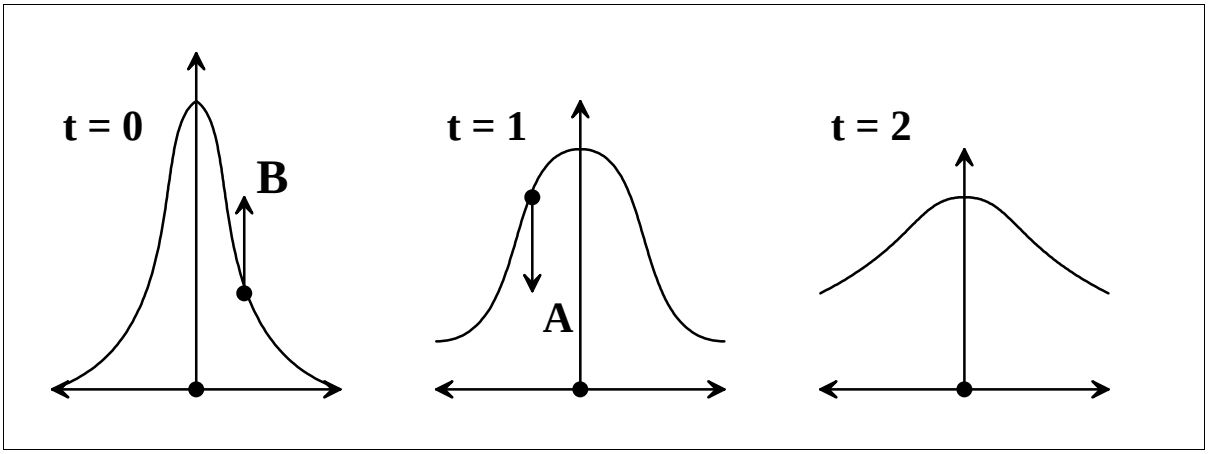


Figure 7: Smoothing. Forces at points A and B are acting in opposite directions. At A the acceleration along the curve is negative so the diffusion process smoothes the function towards the baseline; the acceleration at B is positive so the value of the function increases.

7.2 OSCILLATION: COURANT-HILBERT THEORY

Another form of the diffusion equation is a Gaussian known as the *heat kernel*:

$$g(s,t) = \frac{1}{\sqrt{4\pi t}}\, e^{-s^2/4t} = \frac{1}{\sigma\sqrt{2\pi}}\, e^{-s^2/2\sigma^2} \quad \text{where} \quad \sigma = \sqrt{2t} \tag{7.2}$$

Solutions to this equation take the form $u(s, t) = u_0(s)*g(s, t)$; at time t the solution is therefore the initial state blurred at scale σ.

For example, consider a function such as:

$$u(s) \quad = \quad e^{i\omega s}$$

The derivatives in the diffusion equation can be approximated using finite differences:

$$\frac{\partial u}{\partial t} \quad = \quad \frac{\partial^2 u}{\partial s^2} \qquad \Rightarrow \qquad \frac{u_s^{t+1} - u_s^t}{\delta t} \quad = \quad \frac{u_{s+1}^t + u_{s-1}^t - 2u_s^t}{\delta s^2}$$

Moving terms that can be evaluated at time t over to the RHS gives the rule for iterative smoothing:

$$u_s^{t+1} \quad = \quad u_s^t \quad + \quad \frac{\delta t}{\delta s^2}\left(u_{s+1}^t + u_{s-1}^t - 2u_s^t\right) \quad = \quad e^{i\omega s} \quad + \quad \frac{\delta t}{\delta s^2}\left(e^{i\omega(s+1)} + e^{i\omega(s-1)} - 2e^{i\omega s}\right)$$

This equation can be factorised:

$$u_s^{t+1} \quad = \quad \left(1 \quad + \quad \frac{\delta t}{\delta s^2}\left(e^{i\omega} + e^{-i\omega} - 2\right)\right) e^{i\omega s}$$

Using the identities $e^{i\omega} = \cos\omega + i \sin\omega$ and $e^{-i\omega} = \cos\omega - i \sin\omega$:

$$u_s^{t+1} \quad = \quad k\, e^{i\omega s} \qquad \text{where} \qquad k \quad = \quad 1 \quad + \quad \frac{\delta t}{\delta s^2} 2(\cos\omega - 1)$$

The maximum value of the constant term ($k = 1$) occurs when $\cos\omega = +1$; the minimum ($k = 1 - 4\delta t / \delta s^2$) occurs when $\cos\omega = -1$ so that:

$$1 \quad \geq \quad k \quad \geq \quad 1 - \frac{4\delta t}{\delta s^2}$$

However, $|k| > 1$ will produce growing oscillation rather than smoothing; therefore:

$$1 - \frac{4\delta t}{\delta s^2} \quad \geq \quad -1 \qquad \Rightarrow \qquad \delta t \quad < \quad \frac{\delta s^2}{2}$$

Recall that $\sigma = \sqrt{2t}$. Therefore:

$$\delta\sigma \quad = \quad \sqrt{2\delta t} \qquad \Rightarrow \qquad \delta\sigma \quad < \quad \sqrt{2\,\frac{\delta s^2}{2}} \quad = \quad \delta s$$

This inequality implies that the function $u(s)$ cannot be smoothed by more than δs in a single time step, or oscillation will occur.

8 APPLICATIONS: INTER-SNAKE ENERGY

Certain applications require the simultaneous use of multiple snakes that must be carefully co-ordinated. For example, paired snakes can be used to locate features in stereo images. Likewise, a linked stack of snakes is a powerful tool for analysing multiple cross-sections through three-dimensional images such as NMR data volumes. Similar principles can be used to track moving objects through real-time image sequences.

8.1 STEREO SNAKES

In stereo images, if two features correspond then the disparity between them should vary slowly along their contours, unless they recede in depth very quickly. This constraint is often used in stereo-matching algorithms – for example, see **Pollard et al (1985)**; it can also be expressed as an energy functional linking a pair of stereo snakes:

$$E_{stereo} \quad = \quad \frac{1}{2}\int_0^1 |\mathbf{x}_s^L(s) - \mathbf{x}_s^R(s)|^2 \quad ds \tag{8.1}$$

The subscripts indicate differentiation; $\mathbf{x}^{\mathrm{L}}(s)$ and $\mathbf{x}^{\mathrm{R}}(s)$ are snakes in the left and right images respectively. The energy after moving one of these snakes (in this case the left) is given by:

$$E(\mathbf{x}^L + \delta\mathbf{x}^L,\ \mathbf{x}^R) \quad = \quad \int_0^1 |\mathbf{x}_s^L + \delta\mathbf{x}_s^L \ - \ \mathbf{x}_s^R|^2 \quad ds \tag{8.2}$$

The corresponding energy change is given by:

$$\delta E_{stereo}^L \quad \approx \quad \int_0^1 (\mathbf{x}_s^R - \mathbf{x}_s^L)\cdot\delta\mathbf{x}_s^L \quad ds \quad = \quad \int_0^1 (\mathbf{x}_{ss}^R - \mathbf{x}_{ss}^L)\cdot\delta\mathbf{x}^L \quad ds \tag{8.3}$$

The corresponding force laws (for both snakes) are:

$$\delta\mathbf{x}^L \quad = \quad \mathbf{x}_{ss}^R - \mathbf{x}_{ss}^L \qquad\qquad \delta\mathbf{x}^R \quad = \quad \mathbf{x}_{ss}^L - \mathbf{x}_{ss}^R \tag{8.4}$$

During the process of localising a contour in one image, information about the corresponding contour in the other image is also considered. The stereo match therefore affects the detection and localisation of the features on which it is based. For further information on stereo snakes see **Kass et al (1987; 1988); Menet et al (1990); Bascle and Deriche (1993).**

8.2 INTER-SLICE (VOLUME) SNAKES

Medical research is one of the most popular application domains for active contour models. Typically, snakes are used to reconstruct three-dimensional objects, such as the ventricles of the heart, from two-dimensional slices through data volumes – see **Cohen (1991)**. Similarly,

microscopic structures such as neurons can be reconstructed from spatially consecutive electron micrographs – see **Carlbom et al (1991)**.

A sequence of M slices through a volume of interest can be represented as mappings into a three-dimensional Euclidean space; for each image i a contour $\mathbf{x}_i(s)$ is defined as:

$$\mathbf{x}_i(s) \quad = \quad (\, x_i(s),\, y_i(s),\, z_i \,) \quad \in \quad \boldsymbol{R}^3 \qquad \text{where} \qquad i \in [1,\, M] \tag{8.5}$$

The total energy of all the contours is given by:

$$E_{total} \quad = \quad \sum_{i=1}^{M} E^{i}_{image} \quad + \quad \sum_{i=1}^{M} E^{i}_{intern} \quad + \quad \sum_{i=1}^{M} E^{i}_{extern} \quad + \quad \sum_{i=1}^{M-1} \sum_{j=i+1}^{M} E^{ij}_{couple} \tag{8.6}$$

The energy E_{couple} that arises from coupling adjacent contours is given by:

$$E^{ij}_{couple} \quad = \quad \zeta \int_{0}^{1} \left(\left| \mathbf{x}_i(s) - \mathbf{x}_j(s) \right|^2 \quad - \quad (z_i - z_j)^2 \right) \, ds \qquad \text{where} \qquad j = i + 1 \tag{8.7}$$

The coupling energy is zero for contours that are not in neighbouring images. The non-negative parameter ζ determines the strength of the coupling force with respect to the other forces acting on the contours. Similar techniques can be used with time varying images; for example, **Richens et al (1992)** analysed images of the left ventricle during the cardiac cycle.

8.3 INTER-FRAME SNAKES: MOTION TRACKING

Once an active contour model is locked on to a moving image feature, the model will simply follow the corresponding energy minimum – see **Kass et al (1987; 1988)**. Inter-frame constraints make the tracking more robust, as does the incorporation of mass into the snake design: new positions can then be predicted according to previous positions, velocity, and acceleration – see **Terzopoulos (1987)**.

9 SUMMARY AND CONCLUSIONS

An active contour model $\mathbf{u}(s) = (x(s), y(s))$ is characterised by an energy functional E that includes tension and stiffness terms, and a potential energy generated by processing an image:

$$E(\mathbf{u}(s)) = \oint \underbrace{P(\mathbf{u})}_{\text{Potential}} + \underbrace{\alpha(s)\,|\mathbf{u}'|^2}_{\text{Tension}} + \underbrace{\beta(s)\,|\mathbf{u}''|^2}_{\text{Stiffness}}\; ds \tag{9.1}$$

Note that $\mathbf{u}(0) \equiv \mathbf{u}(N)$ in the discrete approximation to this equation, so the snake forms a closed loop. Assuming that $\alpha(s) = \alpha$ and $\beta(s) = \beta$ are constants, extrema in E satisfy two independent Euler equations:

$$\beta\,\mathbf{u}'''' - \alpha\,\mathbf{u}'' = -\frac{\partial P}{\partial \mathbf{u}} \tag{9.2}$$

The gradient of the potential energy corresponds to an image force with components f_x and f_y. The derivatives in Equation 9.2 can be approximated with finite differences:

$$\begin{aligned} \beta\,(x_{s-2} - 4x_{s-1} + 6x_s - 4x_{s+1} + x_{s+2}) - \alpha\,(x_{s+2} + x_{s-2} - 2x_s) &= f_x(x,y) \\ \beta\,(y_{s-2} - 4y_{s-1} + 6y_s - 4y_{s+1} + y_{s+2}) - \alpha\,(y_{s+2} + y_{s-2} - 2y_s) &= f_y(x,y) \end{aligned} \tag{9.3}$$

The energy of the model can be minimised explicitly by moving each element in turn according to a small fraction of the force given by this equation. Alternatively, a semi-implicit method can be used to solve these equations for all elements of the snake simultaneously.

To use the semi-implicit method two sets of finite difference equations are formed to describe the x and y co-ordinates of the entire snake; these equations can be written in terms of a *cyclic symmetric pentadiagonal banded matrix* $\mathbf{M}$ based on the constants α and β:

$$\mathbf{M}\cdot\mathbf{x} = \mathbf{f}_x(\mathbf{x},\mathbf{y}) \qquad \mathbf{M}\cdot\mathbf{y} = \mathbf{f}_y(\mathbf{x},\mathbf{y}) \tag{9.4}$$

Note that $\mathbf{x}$ and $\mathbf{y}$ are vectors containing the x and y co-ordinates of all the snake elements; $\mathbf{f}_x$ and $\mathbf{f}_y$ are the corresponding vectors of image forces. After matrix inversion, these equations can be solved iteratively by introducing a small time step τ. The internal constraints are imposed at time t+1 after adjusting the snake according to the image forces at time t:

$$\begin{aligned} \mathbf{x}^{t+1} &= (\mathbf{M} + \tau\,\mathbf{I})^{-1} \cdot (\,\mathbf{x}^t + \tau\,\mathbf{f}_x(\mathbf{x}^t, \mathbf{y}^t)\,) \\ \mathbf{y}^{t+1} &= (\mathbf{M} + \tau\,\mathbf{I})^{-1} \cdot (\,\mathbf{y}^t + \tau\,\mathbf{f}_y(\mathbf{x}^t, \mathbf{y}^t)\,) \end{aligned} \tag{9.5}$$

Fortunately, the matrix $\mathbf{M} + \tau\,\mathbf{I}$ need only be inverted once because it consists entirely of constant terms. Equation 9.5 therefore provides a fast iterative method for minimising the energy of an active contour model. The method is implicit with respect to the internal energy, so it can deal with considerable tension and stiffness using large time steps. However, if the image forces become large then a small time step is necessary to prevent oscillation. There is

thus a trade-off between stability and speed: snakes converge quickly on energy minima when large time steps are used, but are then prone to oscillate; improved stability comes from using small time steps, but at the cost of slow convergence.

In addition to the oscillation problem, the basic active contour model suffers from at least two other deficiencies:

- ❑ Active contour models cannot exploit information such as the texture and colour of objects; they are limited to detecting features on the strength of the edges in an image.
- ❑ Active contour models are difficult to use when the edges of a feature are weak, noisy, and diffuse; they tend to get distracted and trapped by nearby edges that do not belong to the desired feature.

These weaknesses suggest that some extension to the basic snake model is needed to link it more strongly with the image data; this extension might exploit image characteristics such as texture and colour, as demonstrated by the active region models developed by **Ivins and Porrill (1995)**.

REFERENCES

Amini, A A; Tehrani, S; Weymouth, T E: 1988. Using dynamic programming for minimizing the energy of active contours in the presence of hard constraints. *Second ICCV: pp 95–99.*

Bascle, B; Deriche, R: 1993. Stereo matching, reconstruction and refinement of 3-D curves using deformable contours. *Fourth ICCV: pp 421–430.*

Benson, A; Evans, D J: 1973. An algorithm for the solution of periodic quindiagonal systems of linear equations. *Computer Journal: vol 16, # 3, pp 278-279.*

Benson, A; Evans, D J: 1977. A normalised algorithm for the solution of positive definite symmetric quindiagonal systems of linear equations. *ACM Transactions – Mathematical Software: vol 3, pp 96-103.*

Carlbom, I; Terzopoulos, D; Harris, K M: 1991. Reconstructing and visualising models of neuronal dendrites. *Patrikalakis, N M (Editor) – Scientific Visualisation Of Physical Phenomena: pp 623–638.*

Cohen, L D: 1991. Note: on active contour models and balloons. *CVGIP (Image Understanding): vol 53, # 2, pp 211–218.*

Hill, A; Taylor, C J; Cootes, T: 1992. A generic system for image interpretation using flexible templates. *Third BMVC: pp 276–285.*

Ivins, J; Porrill, J: 1995. Active region models for segmenting textures and colours. *Image and Vision Computing: vol 13, #5, pp 431–438.*

Kass, M; Witkin, A; Terzopoulos, D: 1987.† Snakes: active contour models. *First ICCV: pp 259–268.*

Menet, S; Saint-Marc, P; Medioni, G: 1990. B-Snakes: implementation and application to stereo. *Seventh Israeli Conference On AI and Computer Vision: pp 223–236.*

Pollard, S B; Mayhew, J E W; Frisby, J P: 1985. PMF: a stereo correspondence algorithm using a disparity gradient limit. *Perception: vol 14, pp 449–470.*

Porrill, J; Ivins, J: 1994. A semiautomatic tool for 3-D medical image analysis using active contour models. *Medical Informatics: vol 19, # 1, pp 81–90.*

Richens, D; Rougon, N; Bloch, I; Mousseaux, E: 1992. Segmentation by deformable contours of MRI sequences of the left ventricle for quantitative analysis. *International Conference On Image Processing And Applications: pp 393–396.*

Scott, G L: 1987. The alternative snake – and other animals. *Third AVC: pp 341–347.*

Terzopoulos, D: 1987. On matching deformable models to images: direct and iterative solutions. *Proceedings – Optical Society Of America, Technical Digest Series: vol 12, pp 160–167.*

Witkin, A; Terzopoulos, D; Kass, M: 1986. Signal matching through scale space. *Proceedings – American Association For Artificial Intelligence: pp 714–719.*

---

† See also *International Journal Of Computer Vision: vol 1 (1988), # 4, pp 321–331.*

APPENDIX A: NUMERICAL APPROXIMATIONS

This appendix gives discrete approximations for continuous derivatives.

A.1 APPROXIMATIONS FOR DERIVATIVES

Consider a function $I(x, y)$. Formulae for the second-order partial derivatives can be derived from the following approximations:

$$I_x \approx \frac{I(x+\delta) - I(x-\delta)}{2\delta} \qquad I_y \approx \frac{I(y+\delta) - I(y-\delta)}{2\delta}$$

$$I_{xy} \approx \frac{I_x(x, y+\delta) - I_x(x, y-\delta)}{2\delta} \tag{A.1}$$

An expression for the second-order x derivative can be obtained using:

$$I_{xx} \approx \frac{I_x(x+\delta) - I_x(x-\delta)}{2\delta}$$

Expanding this formula using the above approximations:

$$I_{xx} \approx \frac{1}{2\delta}\left(\frac{I(x+\delta+\delta) - I(x+\delta-\delta)}{2\delta} - \frac{I(x-\delta+\delta) - I(x-\delta-\delta)}{2\delta}\right)$$

This simplifies to:

$$I_{xx} \approx \frac{I(x+2\delta) + I(x-2\delta) - 2I(x)}{4\delta^2}$$

Finally, the original step size δ is halved:

$$I_{xx} \approx \frac{I(x+\delta) + I(x-\delta) - 2I(x)}{\delta^2} \tag{A.2}$$

Note that if $\delta = 1$ then:

$$I_{xx} \approx I(x+1) + I(x-1) - 2I(x)$$

An expression for the second-order y derivative can be obtained using:

$$I_{yy} \approx \frac{I_y(y+\delta) - I_y(y-\delta)}{2\delta}$$

This formula simplifies as above to give:

$$I_{yy} \approx \frac{I(y+\delta) + I(y-\delta) - 2I(y)}{\delta^2} \tag{A.3}$$

An expression for the derivative with respect to x, of the y derivative can be obtained using:

$$I_{xy} \approx \frac{I_x(x, y+\delta) - I_x(x, y-\delta)}{2\delta}$$

Expanding the *x* derivatives:

$$I_{xy} \approx \frac{1}{2\delta}\left(\frac{I(x+\delta,\ y+\delta)-I(x-\delta,\ y+\delta)}{2\delta} - \frac{I(x+\delta,\ y-\delta)-I(x-\delta,\ y-\delta)}{2\delta}\right)$$

This simplifies to:

$$I_{xy} \approx \frac{I(x+\delta,y+\delta) + I(x-\delta,y-\delta) - I(x+\delta,y-\delta) - I(x-\delta,y+\delta)}{4\delta^2} \quad \text{(A.4)}$$

A.2 PASCAL'S TRIANGLE

Convolution masks for discrete derivatives of a function of one variable can be obtained from the binomial series given by Pascal's Triangle:

	0 :							1										+		
	1 :						1		1								+		−	
Order:	**2 :**	Value:				1		2		1		Sign:				+		−		+
	3 :				1		3		3		1				+		−		+		−
	4 :			1		4		6		4		1		+		−		+		−		+

For example, the fourth derivative of a function $E(s)$ is given by:

$$\frac{dE}{ds} \approx E(s-2) - 4E(s-1) + 6E(s) - 4E(s+1) + E(s+2)$$

A.3 IMAGE PROCESSING MASKS

Pascal's Triangle and Equations A.1–A.3 are ideal for finding the derivatives of parametric curves such as snakes. However, a slightly different format is required when calculating image gradients. Simple image processing masks for first- and second-order derivatives follow:

$$\frac{\partial}{\partial x} \approx \begin{bmatrix} +1 & 0 & -1 \end{bmatrix} \qquad \frac{\partial}{\partial y} \approx \begin{bmatrix} +1 \\ 0 \\ -1 \end{bmatrix}$$

Note that each mask operates with unit step size $\delta = 1$:

$$\frac{\partial^2}{\partial x^2} \approx \begin{bmatrix} +1 & -2 & +1 \end{bmatrix} \qquad \frac{\partial^2}{\partial x \partial y} \approx \frac{1}{4}\begin{bmatrix} -1 & 0 & +1 \\ 0 & 0 & 0 \\ +1 & 0 & -1 \end{bmatrix} \qquad \frac{\partial^2}{\partial y^2} \approx \begin{bmatrix} +1 \\ -2 \\ +1 \end{bmatrix}$$

APPENDIX B: MATHEMATICAL NOTES

This appendix briefly reviews six mathematical topics: parametric form; the characteristics of extrema in functions; the Taylor series; local approximation using partial derivatives; integration by parts; and the use of matrix inversion to solve sets of linear equations.

NOTE 1: PARAMETRIC FORM

If two variables x and y are each expressed in terms of a third variable, such as time t, then $x = f(t)$ and $y = g(t)$ is the *parametric form* of the equation relating x and y. The variable t is known as the *parameter*.

The first and second derivatives of a parametric equation are found as follows:

$$\frac{dy}{dx} = \frac{dy}{dt}\frac{dt}{dx} \qquad \frac{d^2y}{dx^2} = \frac{d}{dt}\left(\frac{dy}{dx}\right)\frac{dt}{dx}$$

For example, the parametric equations for an ellipse aligned with the co-ordinate axes are:

$$y = b\ \sin\theta \qquad x = a\ \cos\theta$$

Note that a and b are constants. To find the first derivative as a function of the parameter:

$$\frac{dy}{d\theta} = b\cos\theta \qquad \frac{dx}{d\theta} = -a\sin\theta$$

Therefore:

$$\frac{dy}{dx} = \frac{dy}{d\theta}\frac{d\theta}{dx} = b\cos\theta\,\frac{1}{-a\sin\theta} = -\frac{b}{a}\cot\theta$$

Similarly, to find the second derivative as a function of the parameter:

$$\frac{d}{d\theta}\left(\frac{dy}{dx}\right) = \frac{b}{a}\csc^2\theta$$

Therefore:

$$\frac{d^2y}{dx^2} = \frac{d}{d\theta}\left(\frac{dy}{dx}\right)\frac{d\theta}{dx} = \frac{b}{a}\ \csc^2\theta\,\frac{-1}{a\sin\theta} = -\frac{b}{a^2\sin^3\theta}$$

NOTE 2: EXTREMA

If x_0 is a *global minimum* of $E(x)$ then throughout the interval over which the function exists:

$$E(x_0) < E(x) \qquad \forall x: \quad x \neq x_0$$

If x_0 is a *local minimum* then there exists an interval such that:

$$E(x_0) < E(x) \qquad \forall x \in [x_0 - \delta_1,\ x_0 + \delta_2]: \quad x \neq x_o$$

Candidates for minima of a smooth functions are given by stationary and end points as shown in Figure B.1.

- ❑ **Stationary points.** The condition $dE/dx = 0$ is *necessary* for a local minimum; however, it is not *sufficient*. A sufficient check for twice-differentiable functions is to examine the second-order differential coefficient, which should be positive.
- ❑ **End points.** Extrema (usually local) occur at the limits of the interval over which a function is defined.

Real minimisation problems typically involve convoluted multi-dimensional hypersurfaces with numerous local minima.

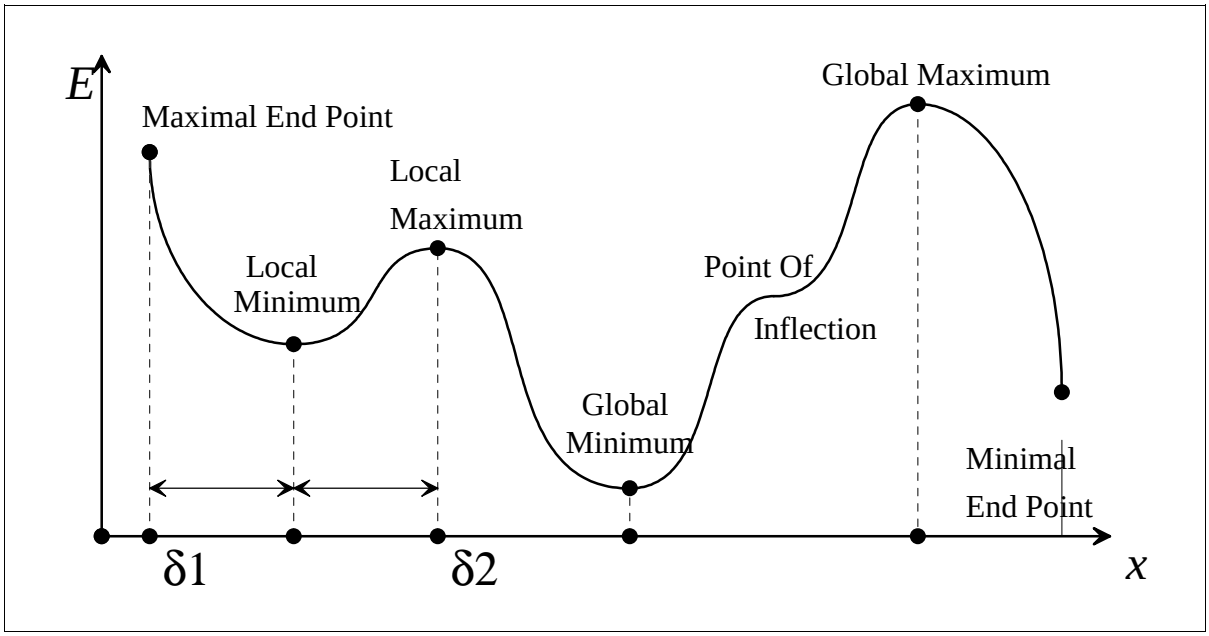


Figure B.1: Extrema In A 1-D Energy Surface. This diagram shows examples of local and global maxima and minima, end points, and a point of inflection, for a scalar function $E(x)$.

NOTE 3: TAYLOR'S THEOREM

Consider a d-times differentiable function:

$$E: \quad x \to y \quad = \quad E(x): \quad \boldsymbol{R} \to \boldsymbol{R}$$

The result of a small change h to this function can be approximated using the Taylor series:

$$E(x+h) \quad = \quad E(x) \quad + \quad h\,E'(x) \quad + \quad \frac{h^2}{2!}\,E''(x) \quad + \quad \cdots \quad + \quad \frac{h^d}{d!}\,E^d(x) \quad + \quad R$$

The remainder R is smaller than any of the terms before it in the series.

This theorem can be used to approximate changes in energy functionals; for example, consider the case when $E(x)$ is at least once differentiable:

$$E(x+h) \approx \underbrace{E(x)}_{\text{Constant}} + \underbrace{h\,E'(x)}_{\text{Linear}}$$

This equation describes a straight line; it can be used to approximate E for parameter values near x. The more derivatives that are included in the series, the more accurate the approximation becomes. (The equation will describe a quadratic, cubic, or higher-order polynomial as more terms are added).

NOTE 4: LOCAL LINEAR APPROXIMATION

If z is a function of x and y so that $z = f(x, y)$, and if x and y change by small amounts δx and δy, then the change in z will also be relatively small:

$$\delta z = A\,\delta x + B\,\delta y + \text{higher terms}$$

Note that higher powers of δx and δy can be ignored if the changes are small; A and B are unknown functions of x and y. If y is kept constant ($\delta y = 0$) then

$$\delta z = A\,\delta x + \text{higher terms} \quad \Rightarrow \quad \frac{\delta z}{\delta x} \approx A$$

Therefore, at the limit of infinitesimal steps:

$$\delta x \to 0 \quad \Rightarrow \quad A = \frac{\partial z}{\partial x}$$

A similar expression can be derived for the function B. Therefore:

$$\delta z \approx \frac{\partial z}{\partial x}\,\delta x + \frac{\partial z}{\partial y}\,\delta y$$

This result is quite general and can be extended to functions of many independent variables. For example, if $z = f(w, x, y)$ then for small changes:

$$\delta z \approx \frac{\partial z}{\partial w}\,\delta w + \frac{\partial z}{\partial x}\,\delta x + \frac{\partial z}{\partial y}\,\delta y$$

NOTE 5: INTEGRATION BY PARTS

The integration of some products requires the use of the following identity:

$$\int_a^b u\,\frac{dv}{ds}\,ds \equiv \Big[uv\Big]_a^b - \int_a^b v\,\frac{du}{ds}\,ds$$

The identity can often be used to simplify terms that contain powers, derivatives, logarithms, exponentials, or trigonometric functions. Care must be taken when deciding which factor to

differentiate and which to integrate. Note that if either $u(a) = u(b) = 0$ or $v(a) = v(b) = 0$, or if the function is periodic such that $u(a) = u(b)$ and $v(a) = v(b)$, then the identity simplifies to:

$$\int_a^b u'v \quad \equiv \quad -\int_a^b uv'$$

NOTE 6: SOLVING LINEAR EQUATIONS BY MATRIX INVERSION

Consider a set of linear equations such as:

$$\begin{array}{ccccccc} a_{11}x_1 & + & a_{12}x_2 & \cdots & a_{1N}x_N & = & b_1 \\ a_{21}x_1 & + & a_{22}x_2 & \cdots & a_{2N}x_N & = & b_2 \\ \vdots & & \vdots & & \vdots & & \vdots \\ a_{N1}x_1 & + & a_{N2}x_2 & \cdots & a_{NN}x_N & = & b_N \end{array}$$

These equations can be rewritten as a matrix multiplication:

$$\underbrace{\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N1} & a_{N2} & \cdots & a_{NN} \end{bmatrix}}_{\mathbf{M}} \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}}_{\mathbf{x}} = \underbrace{\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_N \end{bmatrix}}_{\mathbf{b}}$$

Multiplying both sides of this equation by the inverse of **M** gives:

$$\mathbf{M}^{-1}\mathbf{M}\,\mathbf{x} \quad = \quad \mathbf{M}^{-1}\mathbf{b} \qquad \text{Note:} \quad \mathbf{M}^{-1}\mathbf{M} \quad = \quad \mathbf{I}$$

Therefore, pre-multiplying **b** by the inverse of the matrix of coefficients **M** gives the solutions:

$$\mathbf{x} \quad = \quad \mathbf{M}^{-1}\,\mathbf{b}$$