Faculty of Engineering and Applied Science
Department of Electronics and Computer Science
University of Southampton
United Kingdom

A thesis submitted for fulfilment of the degree
of Master of Philosophy

Extending the Snake Model to Incorporate Velocity

by
Robert Roddis

2 April 2001

This thesis is dedicated to my dedicated parents.

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING AND APPLIED SCIENCE
DEPARTMENT OF ELECTRONICS AND COMPUTER SCIENCE

A thesis submitted in fulfilment of the degree of Master of Philosophy

Extending the Snake Model to Incorporate Velocity

by Robert J. Roddis

Active contour models, or snakes, have become a well established technique in the image processing community. This field has been a fertile area of research, producing many and varied extensions to the original Kass snake. Of primary importance to this thesis are the extensions which incorporate motion.

This work presents a novel approach to incorporate velocity into the formulation of an active contour model. Whereas most time based techniques are centred around a tracking framework, we present a new model offering a single description of the target under investigation and its velocity rather than a series of snapshots.

The active contour models we have developed shows the advantage of incorporating the information across a sequence of frames to extract the boundary of an object moving with constant velocity. The information averaging inherent in the model aids in extracting occluded objects and also performs suitable rejection of static background data.

Two models have been developed, firstly a *global velocity* model to test the efficacy of the approach and secondly, a *local velocity* model to address the short-commings of the local velocity model.

Keywords: Image Processing, Snake, Velocity, Active Contour

Acknowledgements

I would like to thank my supervisor Dr. John Carter for helping me through the course of this research, and for my employment as an undergraduate laboratory demonstrator. I am indebted to Dr. Mark Nixon for his witicisms and his little yellow book, which really got me going.
The author gratefully acknowledges the Engineering and Physical Sciences Research Councils for tenure of a studendtship.

Contents

List of Figures

Chapter 1

Introduction

1.1 Background

When seeking to extract a moving object's outline from a sequence of images, we can inspect each image independently of the others, in the sequence or, we can consider the sequence as a whole. By considering the motion of the object, we have more information at our disposal to aid the extraction process. To this end, we have developed two new contour models, firstly the *global velocity snake.* and secondly the *local velocity snake.*

The global velocity snake model uses a simple velocity parameterisation and a hard constraint, making it suitable to extract contours of a known velocity. The local velocity model has a softer velocity constraint and was developed to segment moving objects who's velocity is not well known.

The technique is based on the active contour model developed by Kass et al. [10]. We aim to extend the *Active Contour* by the inclusion of temporal information derived from video sequences, as demonstrated by the *Velocity Hough Transform* (VHT) [13].

The active contour, or *Snake*, is an energy minimisation process, using a combination of external and internal energy terms, or cost functions, to find a boundary corresponding to a local minima. The external energy is derived from a chosen image functional, such as edge strength data, to measure desired image features. The internal energy is defined by the physical properties of the snake, such as resistance to bending and stretching.

The extension to the snake model is similar to the manner in which the VHT extends the *Hough Transform for Circles* (HTC) [see 14, sec. 5.4.3]. The motion snake is a local, evolutionary model incorporating motion within the energy functional. Whereas, the VHT uses an exhaustive global search to find a solution while incorporating motion into the evidence gathering process.

1.2 Problem Statement

In the extraction of object outlines from image sequences, the classic approach is to use a tracking framework, such as a Kalman snake [18] or using the related CONDENSATION algorithm [8], where essentially the result from one or more preceding frames is used to provide the initialisation for the current one [10, 15]. This framework is generally robust once the contour has been initialised, but the contour does not provide a single description of the object and its motion, and generally will deform under occlusion.

The problem under investigation may be stated as follows: Given a sequence of images of an object moving with constant velocity, can we generate a single description of the object's contour and its motion, by combining information from the time domain with the traditional snake formulation? How robust is this formulation to noise and other factors, such as occlusion, compared to the traditional techniques?

1.3 Outline of Thesis

This Thesis is divided into five parts:

Part I:	A description of snakes and an introduction to the energy minimisation framework.
Part II:	Extensions to the snake model which have been developed to incorporate velocity.
Part III:	Development of a Global velocity model
Part IV:	Development of a Local velocity model
Part V:	Discussion of results and possible future work.

Chapter 2

Properties of Snakes

Snakes, or active contour models are now an established image processing technique employing well understood mathematical optimisation methods. Essentially, a snake is an elastic deformable model, in which an initial estimate of the contour moves and deforms subject to external and internal forces. External forces arise from the image data, or imposed constraints, while the physical properties of the contour define the internal forces.

It is worth noting that the snake was originally termed an *active* contour, as it is continually minimising its energy functional and exhibits dynamic behaviour [10]. Whereas curve trackers using prior dynamical models are referred to as *dynamic* contours [2].

2.1 Motivation for the Snake Model

In computer vision research it is common for a low-level task such as edge detection to be autonomous and rigidly sequential using only the information in the image itself. This approach propagates mistakes made in the initial stages without the possibility for correction. It therefore imposes stringent demands on the reliability of these low-level mechanisms. By incorporating prior knowledge into its formulation, the snake makes use of good, low-level, local operators and a global active contour to extract continuous boundaries.

What is normally required in edge detection is the detection and location of sharp changes in image intensity. This requires numerical differentiation of the image data, which amplifies noise. Edge detection is not robust in noisy situations. By restricting the class of admissible solutions, with the introduction of prior knowledge, the snake restores the edge detection's robustness to noise.

Like many problems in vision, the extraction of an object boundary from an image is an ill-posed problem. By reducing the class of admissible solutions with the application of constraints based on prior knowledge, we can restore well-posedness.

2.2 Snake Model

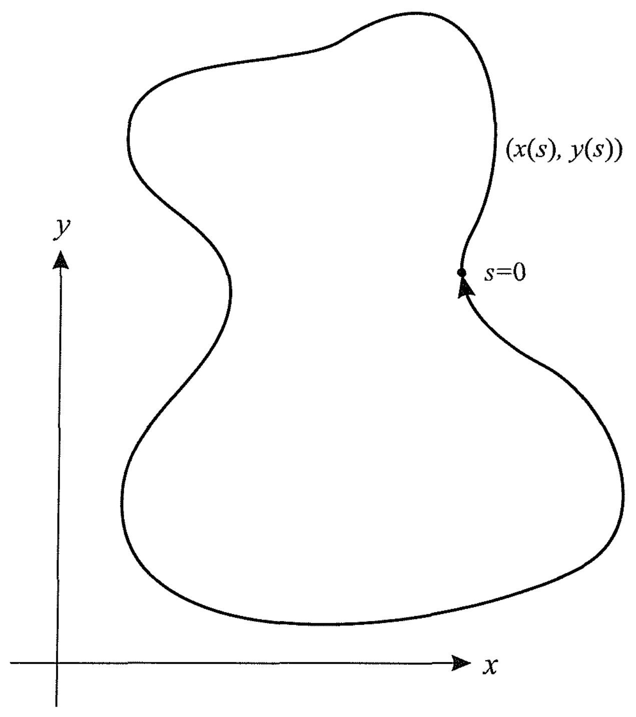


Figure 2.1: Parametric representation of a snake

The snake advocates the paradigm that the presence of an edge does not only depend on the image intensity gradient at a specific point, but also on the spatial distribution of the gradient. This is done by considering curvature, continuity and local edge strength under an energy minimisation framework, and it is worth noting that the snake may be formulated as either an open, or a closed contour.

The snake uses energy minimisation to refine an initial estimate of the contour. There are two energy or cost functions associated with it, the internal and the external energy. The internal energy is a measure of the desired properties of the contour's shape, such as smoothness and continuity. The external energy is derived from an image functional to measure desired image features, such as edges.

By representing the contour parametrically, as $\mathbf{r}(s) = (x(s), y(s))$, given $s \in [0, 1)$ is the normalised arc length of the snake, as shown in figure 2.1, the snake model defines the energy of the snake, E_{snake} as:

$$E_{snake} = \int_{s=0}^{1} E_{int}(\mathbf{r}(s)) + E_{ext}(\mathbf{r}(s))\, ds \tag{2.1}$$

The energy integral is a functional, since its independent variable is a function. E_{int} is the internal energy of the snake and provides the curvature and smoothness constraints of the contour; this is *a priori* information. E_{ext} is the energy term derived from one or more image functionals. This constrains the snake to salient image features and as such, provides *a posteriori* information. The internal energy is used to regulate the external energy.

A solution to the problem is defined to be a local minimum of equation 2.1, a solution of equation 2.2. The contour, $\mathbf{r}(s)$, is evolved to a solution from an initial estimate by minimising equation 2.1. However, this may be the result of either a maxima or a point of inflexion, which can be determined by the sign of the second derivative of E_{snake}. This is seldom necessary, as a minimum is the only stable solution.

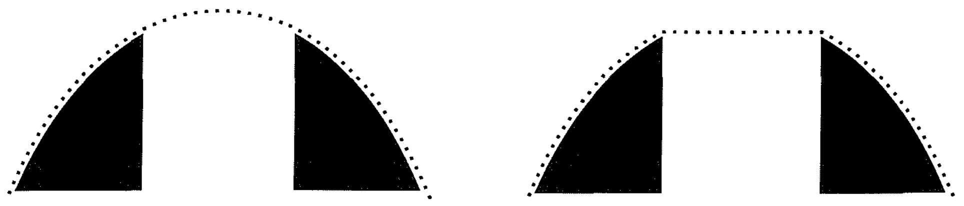
$$\frac{dE_{snake}}{d\mathbf{r}(s)} = 0 \tag{2.2}$$

2.2.1 Internal Energy

E_{int}, the internal spline energy is defined to be a weighted summation of first and second-order derivatives around the contour.

$$E_{int} = \alpha(s) \left| \frac{d\mathbf{r}(s)}{ds} \right|^2 + \beta(s) \left| \frac{d^2\mathbf{r}(s)}{ds^2} \right|^2 \tag{2.3}$$

The first–order continuity term, weighted by $\alpha(s)$, measures the energy due to stretching. The second–order curvature term, weighted by $\beta(s)$, measures the energy due to bending. By altering the values of $\alpha(s)$ and $\beta(s)$, the behaviour of the snake is modified as illustrated in figure 2.2. By setting $\beta(s) = 0$ at a point, the snake becomes second–order discontinuous, and is no longer required to be smooth, allowing a corner to develop. Setting $\alpha(s) = \beta(s) = 0$, the contour becomes discontinuous, and breaks at that point.



(a) Acting as a stiff bar   **(b)** Acting as a thin plate

Figure 2.2: Interpolation properties of E_{int} on an occluded arc

The interpolation properties of the elasticity and curvature functionals are shown in figure 2.2. Figure 2.2(a) shows the interpolation function arising from the curvature term, making the snake act as a stiff bar. In figure 2.2(b) the interpolation function is due to the elastic term, making the snake act as a thin plate.

It is worth noting that as $\alpha(s)$ and $\beta(s)$ are functions of the curve parameter, the natural behaviour of curve segments can be varied, as Lai and Chin [12] have done. This can only be exploited if the desired shape and orientation of the contour is known or can be estimated.

2.2.2 Image Energy

The image functional defines the regions of the image that have a low energy and therefore attract the contour. The scheme proposed in [10] uses three energy functionals which attract a snake to lines, E_{line}, edges, E_{edge} and line terminations, E_{term}. The total image energy can be expressed as a weighted sum of the three energy functionals

$$E_{image} = w_{line}E_{line} + w_{edge}E_{edge} + w_{term}E_{term} \tag{2.4}$$

By adjusting the weights, a wide range of snake behaviour can be created. The simplest useful line functional is the image intensity itself. If we set

$$E_{line} = I(x, y) \tag{2.5}$$

then depending on the sign of w_{line} the snake will be attracted to either light lines or dark lines. Finding edges in an image can be done with a very simple energy functional. If we set

$$E_{edge} = -|\nabla I(x, y)|^2 \tag{2.6}$$

the snake is attracted to contours with large image gradients.

An edge detector applied to an image will produce an edge map consisting mostly of thin edges. This means that the edge strength function tends to be zero in most places, apart from on a few lines. It has been suggested by Waite and Welsh [19] that the edge map should be blurred to broaden the width of the edges, to help the snake come under the influence of an edge. This modifies equation 2.6 to become

$$E_{edge} = G_\sigma(x, y) * \{-|\nabla I(x, y)|^2\} \tag{2.7}$$

where $G_\sigma(x, y)$ is a Gaussian smoothing function and $*$ represents convolution. Cohen and Cohen [5] further suggest, E_{edge} should be the result of a binary edge detector, such as the Canny edge extractor [3], convolved with a Gaussian smoothing kernel, $G_\sigma(x, y)$, to produce an *attraction potential.*

The curvature of level intensity contours, where $I(x, y)$ is constant, can be used in a slightly smoothed image to create an energy term that attracts the snake to terminations of line segments and corners.

Given $C(x, y) = G_\sigma(x, y) * I(x, y)$, is a slightly smoothed version of the image, the gradient angle, θ, is then

$$\theta = \tan^{-1}\left(\frac{C_y}{C_x}\right) \tag{2.8}$$

where the subscripts denote partial derivatives.

Let $\hat{\mathbf{n}} = (\cos\theta, \sin\theta)$ and $\hat{\mathbf{n}}_\perp = -(\sin\theta, \cos\theta)$ be unit vectors along and perpendicular to the gradient direction, then the curvature of the level contours can be written

$$E_{term} = \frac{\partial\theta}{\partial\hat{\mathbf{n}}_\perp} = \hat{\mathbf{n}}_\perp \bullet \nabla\theta \tag{2.9}$$

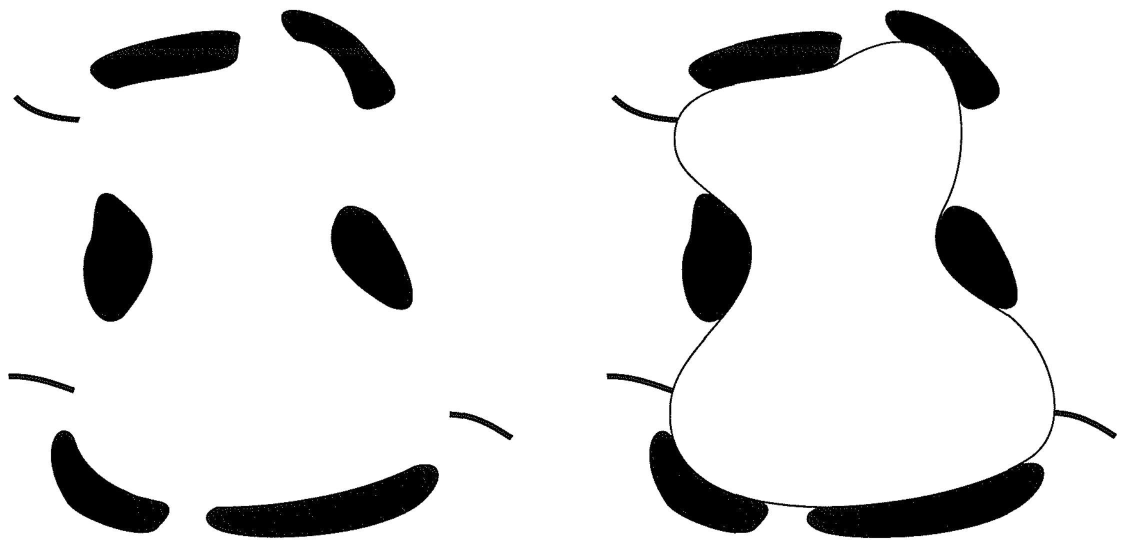
Expressing $\hat{\mathbf{n}}_\perp$ as a function of $C(x, y)$

$$\hat{\mathbf{n}}_\perp = \frac{1}{\sqrt{C_x^2 + C_y^2}}(-C_y, C_x) \tag{2.10}$$

Then it follows from equations 2.8, 2.9 and 2.10 that the termination function can be expressed as a function of first and second-order partial derivatives of $C(x, y)$, as

$$E_{term} = \frac{C_{yy}C_x^2 - 2C_{xy}C_xC_y + C_{xx}C_y^2}{(C_x^2 + C_y^2)^{3/2}} \tag{2.11}$$

The termination functional computes the energy of the point (x_c, y_c) as the curvature of the level contour which passes through (x_c, y_c) defined by $(C_x, C_y) = C(x_c, y_c)$. By combining E_{edge} and E_{term} to form E_{image} we create a snake that is attracted to edges and line terminations as shown in figure 2.3.



(a) Standard subjective contour

(b) Snake in equilibrium on subjective contour

Figure 2.3: Edge/termination snake on a subjective contour

2.3 Snake Implementations

The original snake implementation [10] evolved a set of points, where each point is defined on a discrete grid, using the numerical method of finite differences to iteratively minimise equation 2.1. This model has been adapted to use finite element techniques [19], [5].

Following is an outline of the snake models used in the development of the motion snake. These are:

- Kass Snake: Original snake model developed by Kass et al. [10].
- Greedy Snake: Fast, discrete snake algorithm developed by Williams and Shah [20].
- Xu Snake: Robust Snake with insensitive parameters developed by Xu et al. [22].

2.3.1 Kass Snake

The original snake model, was developed as a tool to assist in feature extraction, rather than as an autonomous technique, requiring much user interaction and careful parameter selection to get the best results.

In this formulation, the snake is defined as an ordered set of points, $\mathbf{r} = [\mathbf{r}_1, \mathbf{r}_2, \dots, \mathbf{r}_n]$ where each point $\mathbf{r}_i$, is defined on the finite grid: $\mathbf{r}_i \in \mathbf{E} = \{(x,y) : x, y = 1, 2, \dots, M\}$. We seek a set of snake points, $\hat{\mathbf{r}}(s) = (\hat{x}(s), \hat{y}(s))$, for which E_{snake} in equation 2.1 is minimised. Minimising equation 2.1 is trivial. The snake has a degenerate solution as the single point which is chosen to minimise E_{ext}. Clearly, the ideal situation is to seek a local minimum in the locality of the initial position of the snake.

By the application of the calculus of variations [see 17, chap. 20], and numerical methods [16], the snake equation can be rendered to a system of non-linear equations which needs to be solved iteratively. At each iteration, co-ordinates are calculated as real functions, and a complete set of new contour points is provided. The new set of contour points become the starting set for the next iteration. This is a continuous formulation that solves for all contour points at once.

The application of finite difference and finite element analysis are both presented in depth by Waite and Welsh [19], while [4], [10] and [14, sec 5.5] provide implementations using finite differences and Cohen and Cohen [5] provide further treatment on the finite element approach.

The Kass snake does suffer from a high sensitivity to initial conditions and parameter selection, and as a result has been the target of research for example to, reduce parameter sensitivity and improve robustness to initialisation.

Various improvements have been proposed for the Kass snake. Cohen [4] suggests the addition of a ballooning force to inflate the contour, giving a more dynamic behaviour, while stopping the snake from contracting to a point under the influence of E_{int}. Cohen and Cohen [5] have also extended the snake model to 3-D with surface fitting Finite Element Methods.

2.3.2 Greedy Snake

The Greedy snake is a discrete, fast and conceptually simple energy minimisation algorithm. Computing the index of S snake points, $\mathbf{r}_S$, $\forall s \in (0, 1, \ldots, S-1)$ modulo S, the energy functional minimised for each snake point is:

$$E_{greedy}(s) = E_{int}(\mathbf{r}_s) + E_{image}(\mathbf{r}_s) \tag{2.12}$$

This is expressed as

$$E_{greedy} = \alpha(s)\left|\frac{d\mathbf{r}_s}{ds}\right|^2 + \beta(s)\left|\frac{d^2\mathbf{r}_s}{ds^2}\right|^2 + \gamma(s)E_{edge} \tag{2.13}$$

E_{image} is usually taken as the magnitude of the edge functional E_{edge}, normalised to lie in the range $[0, 1)$, where a strong edge has low energy.

The first-order difference term imposes a continuity constraint on the points and is approximated as the modulus of the difference between the average Euclidean distance between contour points and the Euclidean distance between the currently selected image point, $\mathbf{r}_s$ and the next contour point

$$\left|\frac{d\mathbf{r}_s}{ds}\right|^2 = \left|\frac{\sum_{i=0}^{S-1}||\mathbf{r}_i - \mathbf{r}_{i+1}||}{S} - ||\mathbf{r}_s - \mathbf{r}_{s+1}||\right|^2 \tag{2.14}$$

The second-order differential imposes a curvature constraint on the contour and is implemented as an estimate of the curvature between the next and previous contour points, $\mathbf{r}_{s-1} = (x_{s-1}, y_{s-1})$ and $\mathbf{r}_{s+1} = (x_{s+1}, y_{s+1})$, and the snake point currently under inspection, $\mathbf{r}_s = (x_s, y_s)$

$$\left|\frac{d^2\mathbf{r}_s}{ds^2}\right|^2 = |\mathbf{r}_{s-1} - 2\mathbf{r}_s + \mathbf{r}_{s+1}|^2 \tag{2.15}$$

The Greedy algorithm attempts to directly minimise equation 2.12 by iteratively solving for each snake point in turn, selecting as the new position, the neighbouring image point with the lowest snake energy E_{snake}. This is done until the snake points no longer change between successive iterations. As the Greedy algorithm uses only neighbouring image points in the energy functionals, convergence to a local minimum is not guaranteed. This can lead to the snake oscillating between two solutions. Cohen [4] proposed bi-linear interpolation of the image data and using a sub-pixel step size, this could be overcome at the cost of greater computation.

2.3.3 Xu Snake

The Xu snake was developed by Xu, Segawa, and Tsuji [22] to overcome the problem of parameter sensitivity through the modification of the internal energy term of the snake.

Xu et al. identified that parameterisation problems with the Kass snake arise from the normal force (perpendicular to the contour) generated by the contour's internal energy (cf. equation 2.3), is a function of not only the internal parameters, but also of the contour's shape. Where sections of high curvature, or with widely spaced points produce a higher normal force, even if the intensity of the edge gradient is the same for all points. This causes the snake to overrun some edge points and converge on others depending on the contour's shape if all edge points have the same intensity. Although in theory it is possible to calculate a proper pair of weights (α and β) for the internal energy terms to overcome this, it is in practice very difficult.

In order to overcome this Xu et al. developed an additional energy term, such that all contour points have the same normal force regardless of contour shape. The formulation exhibits the following new properties:

- No longer sensitive to internal parameters
- Can converge to high curvature points, like corners
- Can be easily made to inflate or deflate
- As the internal normal force is virtually zero, the contour will pass through small gaps and will not form subjective contours

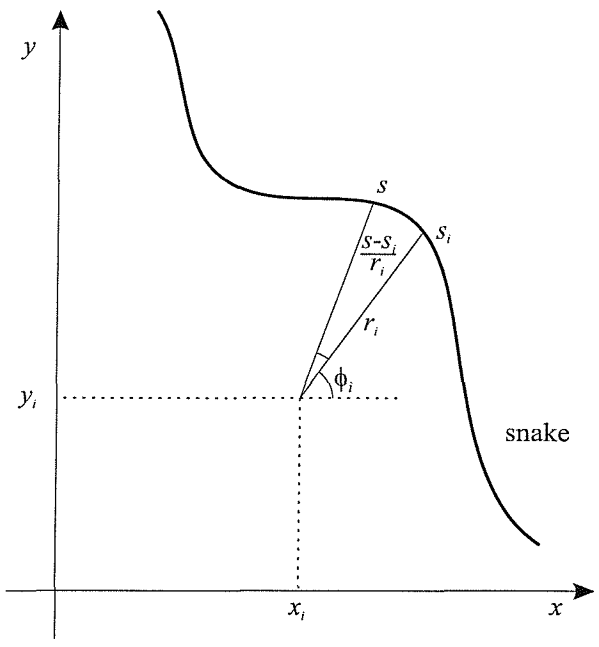


Figure 2.4: Approximating the contour

In order to nullify the internal normal force at each point, an approximation of the internal normal force is calculated, $w_{press}E_{press}$ and is applied opposing the internal normal force allowing the contour to remain stationary regardless of shape. In order to move the contour towards edges of interest, a controlling force, $w_{control}E_{control}$ is also added, giving a new snake equation:

$$E_{snake} = \int_{s=0}^{1} \{E_{int} + w_{press}E_{press} + w_{control}E_{control} - w_{ext}E_{ext}\mathbf{r}(s)\}\, ds \tag{2.16}$$

$w_{press}E_{press}$ is calculated by approximating the contour as arcs at each point. Let the radius of curvature at point $s_i = (x_i, y_i)$ be r_i, as shown in figure 2.4. The contour $(x(s), y(s))$ can then be re-written as:

$$\begin{aligned} x(s) &\approx r_i \cos\left(\frac{s - s_i}{r_i} + \phi_i\right) + x_i \\ y(s) &\approx r_i \sin\left(\frac{s - s_i}{r_i} + \phi_i\right) + y_i \end{aligned} \tag{2.17}$$

where

$$\phi_i = \tan^{-1}\frac{y(s_i) - y_i}{x(s_i) - x_i} = \tan^{-1} - \frac{x_s(s_i)}{y_s(s_i)} \tag{2.18}$$

Adding a new force perpendicular to the contour with weight w_{press} the Euler equations without image energy terms become

$$\begin{aligned} -\alpha x_{ss} + \beta x_{ssss} + w_{press}\frac{\partial E_{press}}{\partial x} &= 0 \\ -\alpha y_{ss} + \beta y_{ssss} + w_{press}\frac{\partial E_{press}}{\partial y} &= 0 \end{aligned} \tag{2.19}$$

Since

$$\begin{aligned} \frac{\partial E_{press}}{\partial x} &= \cos\phi_i \\ \frac{\partial E_{press}}{\partial y} &= \sin\phi_i \end{aligned} \tag{2.20}$$

substituting (2.18) and (2.20) into (2.19) we have for each point s_i

$$\begin{aligned} \left(\frac{\alpha}{r_i} + \frac{\beta}{r_i^3} + w_{press}(s_i)\right)\cos\phi &= 0 \\ \left(\frac{\alpha}{r_i} + \frac{\beta}{r_i^3} + w_{press}(s_i)\right)\sin\phi &= 0 \end{aligned} \tag{2.21}$$

Now we can see that the contour will be in equilibrium if we calculate $w_{press}(s_i)$ as

$$w_{press}(s_i) = -\left(\frac{\alpha}{r_i} + \frac{\beta}{r_i^3}\right) \tag{2.22}$$

As the snake equations are solved using the Euler equations, we need only calculate the apparent centre of the arcs, (x_i, y_i) from which we can calculate r_i and ϕ_i. Calculating the new forces is computationally expensive when compared to the Kass implementation, and is therefore considerably slower.

While Xu has shown that the normal force of the contour is a function of not only the internal parameters but also the contour's shape, by removing the normal force, he has also removed the regularisation properties of the snake, by removing the smoothness constraint.

2.3.4 Further Extensions

The problems outlined in the snake models above have been the subject of much research, and a number of possible solutions have been proposed.

Cohen and Cohen [5] have proposed normalising the snake forces to reduce parameter sensitivity for internal and external forces. They suggest a local normalisation of the energy functional so that the energy functional along the length of the contour lie within the range $[-1, 1]$

$$\mathbf{F}_s = \frac{-1}{k}\begin{pmatrix} \frac{\partial E_{ext}}{\partial x}\Big|_{\mathbf{r}_s} \\ \frac{\partial E_{ext}}{\partial y}\Big|_{\mathbf{r}_s} \end{pmatrix} \quad \text{where} \quad k = \max_{x,y}\left(\sqrt{\left.\frac{\partial E_{ext}}{\partial x}\right|_{\mathbf{r}_s}^2 + \left.\frac{\partial E_{ext}}{\partial y}\right|_{\mathbf{r}_s}^2}\right) \tag{2.23}$$

this formulation is suitable for use with smoothed binary edge data. The global normalisation of equation (2.24) ensures that the energy functional of the whole image lies within the range $[-1, 1]$, and is required if edge strength data is used instead of a binary edge-map. Otherwise, the information associated with edge strength functional is disregarded.

$$\mathbf{F}_s = \frac{-1}{k} \begin{pmatrix} \left.\frac{\partial E_{ext}}{\partial x}\right|_{\mathbf{r}_s} \\ \left.\frac{\partial E_{ext}}{\partial y}\right|_{\mathbf{r}_s} \end{pmatrix} \quad \text{where} \quad k = \max_{x,y} \left(\sqrt{\frac{\partial E_{ext}}{\partial x}^2 + \frac{\partial E_{ext}}{\partial y}^2} \right) \tag{2.24}$$

To make the snake behaviour more dynamic and remove the tendency of the snake to minimise to the degenerate solution of a single point, Cohen and Cohen [5] propose the addition of a ballooning force. This ballooning force causes the snake to expand rather than contract when not under the influence of external forces.

A superior solution to the problem is to change the contour's formulation, keeping the regularisation and making the internal forces invariant of contour scale and rotation. Gunn and Nixon [7] have developed a local shape model, where an estimate of any shape can be included as an equilibrium state. The natural contour shape is then modified. This technique is only applicable if the target object's shape is known or can be estimated.

Different contour descriptions have been used to include prior shape, such as B-splines [2], wavelets [11], and the local shape model mentioned previously.

In addition to the local shape model, Gunn and Nixon [7] have developed a technique, the *Dual Active Contour* to overcome weak local minima. By searching for a global minima in a constrained region, weak local minima are rejected, reducing the need for careful selection of the initial contour position. This has been implemented as an evolutionary technique, similar to the original formulation and also as a search based technique, using dynamic programming techniques demonstrated by Amini et al. [1].

The *Gradient Vector Flow* (GVF) snake developed by Xu and Prince [21] provides a new external force. This force, called gradient vector flow (GVF), is computed as a diffusion of the gradient vectors of a gray-level or binary edge map derived from the image. It differs fundamentally from traditional external forces in that it cannot be written as the negative gradient of a potential function, and the corresponding snake is formulated directly from a force balance condition rather than a variational formulation. This overcomes problems associated with initialisation and poor convergence to concave boundaries.

Chapter 3

Extending the Snake Model

Before looking at extending the snake model to include velocity, we will briefly see how a velocity term was incorporated in the parameterisation of the Hough Transform for Circles by Nash et al. [13] to produce the Velocity Hough Transform.

3.1 How the VHT Extends the HT

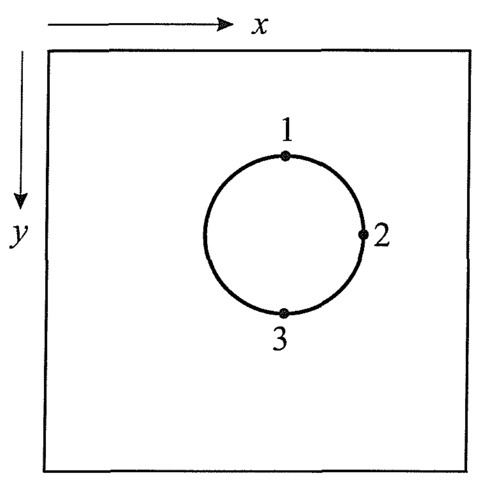
The Hough Transform is an exhaustive global search that requires parameterization of the shape we are looking for and extensive computational power and storage. We will concern ourselves with the simple case of looking firstly for circles in single images and then moving circles in image sequences.

3.1.1 Hough Transform for Circles

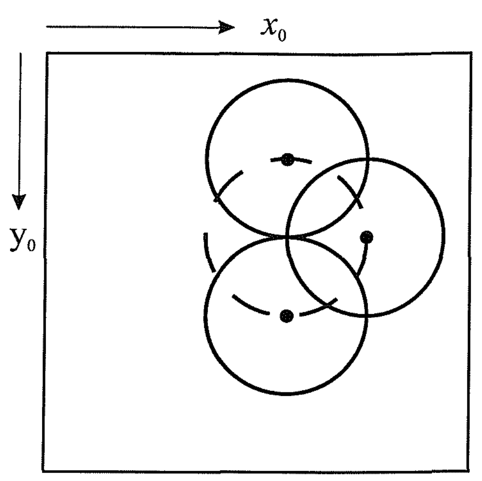
The Hough Transform for Circles is conceptually simple, and can be found in many books on image processing [e.g. 14, sec. 5.4.3]. A circle may be parameterized as a locus of points (x, y) centred on an origin (x_0, y_0), with a radius r, according to:

$$(x - x_0)^2 + (y - y_0)^2 = r^2 \tag{3.1}$$

The circle can also be visualised as a locus of points (x_0, y_0) centre (x, y) radius r in an accumulator space.



(a) Image containing a circle



(b) Accumulator space

Figure 3.1: The Hough Transform for circles

The accumulator space is represented as a three-dimensional array, with all elements initialised to zero. When a vote is cast in the accumulator, the weighted value of the image point under inspection is added to the accumulator element in question. For each point in the image of Figure 3.1(a) a cone of votes is cast in the accumulator space according to equation 3.1, as in Figure 3.2. Figure 3.1(b) shows a slice taken through the accumulator at the radius of the circle in Figure 3.1(a). We can see that the location in the accumulator with the largest number of votes corresponds to the intersection of all the cones of votes and this corresponds to the parameters of the circle in Figure 3.1(a).

3.1.2 VHT for Circles

The HT has been extended to find circles moving through a sequence of images with constant velocity. The VHT uses the polar parametric form of the equation of a circle to locate a moving circle

$$\mathbf{x} = x_0 + v_x t + r\cos\theta \tag{3.2}$$

$$\mathbf{y} = y_0 + v_y t + r\sin\theta \tag{3.3}$$

where $\mathbf{x}$ and $\mathbf{y}$ are the points on the circumference, x_0 and y_0 are the centre co-ordinates in the initial image, r is the radius and θ varies from 0 to 360°. v_x and v_y are the velocities

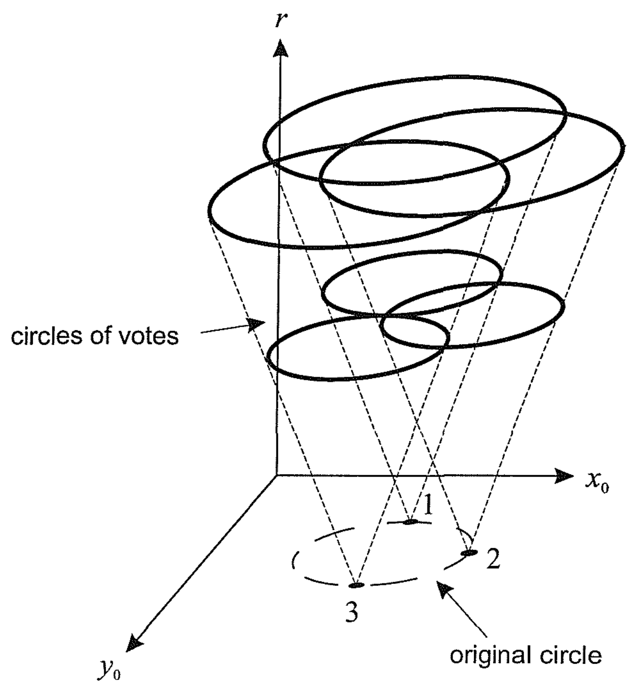


Figure 3.2: The 3D accumulator space

of the circle in the x and y direction respectively measured in pixels per second and t is the time elapsed since the first frame at $t = 0$.

Figure 3.4 shows how the circle is described in any frame as a function of its centre co-ordinates in the first frame, its velocity and the time elapsed since the first frame.

The voting process occurs in a five-dimensional accumulator space to estimate the parameters x_0, y_0, r, v_x and v_y. The process starts by scanning for edge pixels in the first frame. For each edge pixel found, Equations 3.2 and 3.3 are employed to determine the possible center co-ordinates of a circle for a predefined range of radii and x and y velocities, given the current time reference, t, of the frame. The accumulator cells corresponding to each value are incremented. Once the whole image has been processed, the process repeats for subsequent frames, voting in the same accumulator. The peaks in the accumulator after voting has been completed correspond to the best estimates of the parameters of moving circles in the image.

It has been shown that considering the data from all the frames at once when constructing the accumulator, the VHT is more successful in locating a moving circle than a series of HTs on each frame individually. This holds true, both in the presence of strong

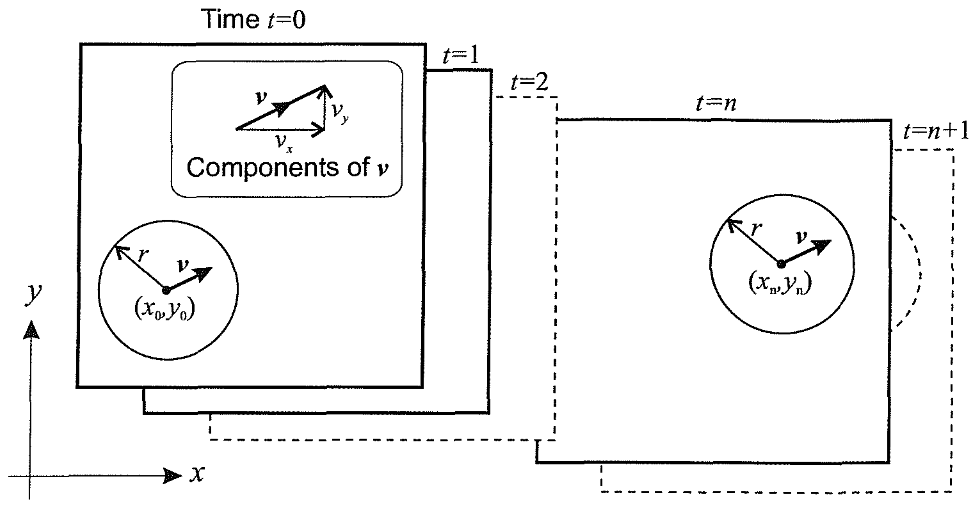


Figure 3.3: Circle with constant velocity in an image sequence

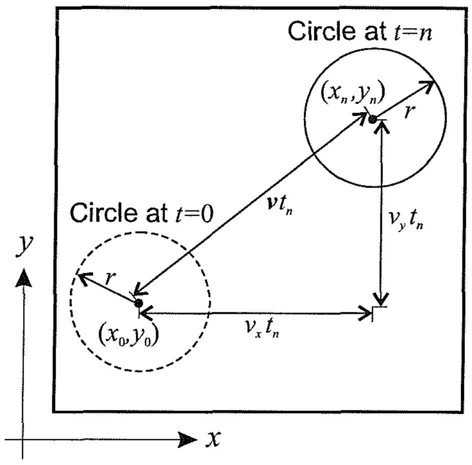


Figure 3.4: Defining the velocity parameters for the VHT

additive white Gaussian noise and when occluding the same region of each frame in the sequence.

3.2 Adding Velocity to the Snake Model

Based on the success of the VHT we propose to extend the snake model to bring to bear the information arising from the correlation between image frames over the entire sequence.

Two approaches have been taken, a *global velocity approach, where each point on the contour has the same velocity and a* local velocity approach, where each point on the contour has its own velocity.

3.2.1 Global Velocity

In this approach, each point on the contour, $\mathbf{r}(s,t)$ is defined as having a velocity, $\mathbf{v}$. Expressing $\mathbf{r}(s)$ and $\mathbf{v}$ in Cartesian coordinates as $(\mathbf{x}(s), \mathbf{y}(s))$ and (v_x, v_y) respectively, the contour at time, t, is described by:

$$(\mathbf{x}(s,t), \mathbf{y}(s,t)) = (\mathbf{x}(s) + v_x t,\ \mathbf{y}(s) + v_y t) \tag{3.4}$$

From Equation 3.4, it is evident that any change in the contour velocity, $\mathbf{v}$, will affect the position of all points in a frame equally, resulting in no change to the internal energy of the contour. This method can be used to easily extend the contour models presented in Section 2.3, by modifying the external energy term and incorporating a local search method to find the correct velocity. In this way, this method provides a greedy velocity search and is affected by problems of discretisation and instability, much like the greedy snake.

In spite of its shortcomings, this method is adequate for evaluating the efficacy of incorporating the velocity from all frames at once on feature extraction performance.

3.2.2 Local Velocity

With this approach, each contour point has a velocity term which is a function of contour position. Using a local velocity model, the solution to both the contour shape and the contour velocity is driven by the image information from all frames at once.

In this approach, a contour, $\mathbf{r}(s,t)$ is defined to have a velocity, $\mathbf{v}(s)$,which is a function of the contour arc length, s. The contour is described at time t by:

$$\mathbf{r}(s,t) = \mathbf{r}(\mathrm{s}) + \mathbf{v}(s)t \tag{3.5}$$

where $\mathbf{r}(s)$ is the contour shape at time $t = 0$. From Equation 3.5, it is clear that the contour shape in each frame will differ, and that varying $\mathbf{v}(s)$, will cause the internal energy of the contour to change.

This formulation is more complicated than the global velocity approach, but this affords a continuous velocity solution, where the contour velocity is updated at once, in a continuous manner, much like the manner in which the Kass snake updates spatial contour information. This formulation goes on to address the problems of the Global velocity model mentioned above.

3.3 Global Velocity Snake Models

Here we describe the active contour models developed to study the effects of adding a discrete velocity term to the standard active contour models presented in Section 2.3.

Each new active contour model is presented in terms of how the snake energy function, E_{snake}, of the original active contour model has been modified to incorporate velocity. Before looking at the snake models themselves, it is instructive to understand how the global velocity search is implemented. This extension of the Greedy Snake forms the basis of all the discrete time techniques discussed below.

3.3.1 Global Velocity Search

Global velocity snake models use a search algorithm, to estimate the best velocity term for the entire contour in each iteration.

A greedy velocity search has been implemented, where the contour evolves a new set of points and new velocity at each iteration, which is demonstrated by the pseudo-code of Figure 3.5.

The iterative loop is initialised with an array of nine contours, `contour[0..8]` all set to the initial contour, `initial_contour`. At the start of each iteration, the best contour from the previous iteration, `contour[best]`, is copied to nine temporary contours, `contour[0..8]`, each `contour[i]` is given a velocity displacement of $(v_x + d_{ux}, v_y + d_{uy})$ according to the contour index, `i`, (Figure 3.5(b)), providing a 3-by-3 local neighbourhood search of the velocity. Each `contour[i]` is iterated once with the contour evolution

```
contour[0] = initial_contour;
best = 0;
do {
  for (i=0;i<9;i++){
    contour[i]=contour[best];
  }
  for(i=0;i<9;i++){
    evolve(contour[i],ux[i],uy[i]);
  }
  best = best_contour(contour);
  update_ux(best);
  update_uy(best);
} while (!terminated());
```

(a) Pseudo-code Iterative loop

Contour index, `i` x velocity, **`ux[i]`** y velocity, **`vx[i]`**

`i`=0 u_x-$d_{u\,x}$ u_y+$d_{u\,y}$	`i`=1 u_x u_y+$d_{u\,y}$	`i`=2 u_x+$d_{u\,x}$ u_y+$d_{u\,y}$
`i`=3 u_x-$d_{u\,x}$ u_y	`i`=4 u_x u_y	`i`=5 u_y u_x+$d_{u\,x}$
`i`=6 u_x-$d_{u\,x}$ u_y-$d_{u\,y}$	`i`=7 u_x u_y-$d_{u\,y}$	`i`=8 u_x+$d_{u\,x}$ u_y-$d_{u\,y}$

(b) 3×3 Velocity search grid

Figure 3.5: Local neighbourhood velocity search

function, `evolve(...)`. The nine new contours are all evaluated with the Greedy Snake evaluation routines, `best_contour(...)` to decide which `contour[i]` has the lowest energy, and therefore is chosen as initial contour for the next iteration. The new values of v_x and v_y are updated with `update_ux(best)` and `update_uy(best)`

This requires an extension of the external energy term of each snake model to incorporate the image information from all frames when updating the contour position. When calculating the image energy of the ith contour point, $E_{image}(x_i, y_i)$, we take a weighted sum of the image energy, E_{image_j}, for (x_i, y_i) in each frame,

$$E_{image}(x_i, y_i) = \frac{1}{n} \sum_{j=0}^{n-1} E_{image_j}(x_i + v_x t_j,\ y_i + v_y t_j) \tag{3.6}$$

where t_j is the time elapsed since the first frame at $t = t_0$. This value of $E_{image}(x_i, y_i)$ is used in subsequent calculations when determining the contribution from all frames to E_{image}.

3.3.2 Velocity Greedy Snake

The Velocity Greedy Snake, or vGreedy snake uses the Greedy Snake of Section 2.3.2 as the basis of the active contour. This provides a rapid and conceptually simple implementation although it is not robust to initialisation parameters. As a consequence of the local evolution of the contour it is prone to passing over the features of interest and oscillating around local minima.

The vGreedy implementation is based on the pseudo code of Figure 3.5(a). E_{image} is calculated from all image frames at once, such that for a contour $\mathbf{r}(s)$, at each point s_i, $E_{image}\mathbf{r}(s_i)$ is calculated as in equation 3.6 which gives the following expression for the contour energy

$$E_{vGreedy} = \int_{s=0}^{1} \alpha(s) \left| \frac{d\mathbf{r}_s}{ds} \right|^2 + \beta(s) \left| \frac{d^2\mathbf{r}_s}{ds^2} \right|^2 + \frac{\gamma(s)}{n} \sum_{j=0}^{n-1} E_{image_j}(\mathbf{r}(s))\, ds \tag{3.7}$$

At each iteration, the candidate contours are evolved in the same local manner as the Greedy Snake, using finite difference approximations of the derivatives in Equation 3.7, and the candidate with the lowest total energy is elected to become the initial contour of the next iteration.

3.3.2.1 Qualitative performance

The snake parameters require careful selection in order to get the snake to converge on the object boundary, and not pull through. The vGreedy snake, due to its discrete nature shows problems of convergence. It tends to either oscillate on the object boundary, or more commonly, to locate part of the object boundary and to then proceed to pull itself through.

3.3.3 Velocity Kass Snake

To overcome the local nature of the vGreedy snake, the vKass snake was developed. As the name implies, this snake uses the Kass snake of Section 2.3.1 as the basis of the active contour. This was found to be as sensitive as the greedy snake to initial parameters, but provides a more stable solution with less tendency to oscillate, owing to the continuous contour formulation and also the sub-pixel resolution afforded by the bilinear interpolation of the energy landscape used in this implementation.

The snake equation, E_{vKass} is derived from Equation 2.1 where E_{ext} is modified with equation (3.6), to consider the information from all frames at once

$$E_{vKass} = \int_{s=0}^{1} \frac{1}{2}(E_{int}(\mathbf{r}(s)) + \frac{\gamma}{n}\sum_{j=0}^{n-1} E_{image_j}(\mathbf{r}(s))\, ds \tag{3.8}$$

3.3.3.1 Qualitative performance

This snake requires careful parameter selection to ensure it locates features of interest and does not pass over them, or fail to be attracted by a local energy potential. Quite often, once the contour has located the boundary of an object, it will continue to contract, albeit more slowly and *push* itself through the object boundary, eventually converging to a point.

3.3.4 Velocity Xu Snake

This snake is based on the Xu snake of Section 2.3.3. The Xu active contour model was chosen as it exhibits robustness to initial parameter selection, and additionally, the deflation force can be easily controlled to prevent the contour passing right over features of interest.

Again the velocity component of the snake model uses Equation 3.6. The snake equation for the vXu snake, derived from Equation 2.16, is modified to become:

$$E_{vXu} = \int_{s=0}^{1} E_{int} + w_{press}E_{press} + w_{control}E_{control} - \frac{w_{ext}}{n}\sum_{j=0}^{n-1} E_{image_j}(\mathbf{r}(s))\, ds \tag{3.9}$$

3.3.4.1 Qualitative performance

This snake shows robustness to initial parameter settings, requiring little or no experimentation with the snake parameters to get the snake to converge on the object boundary. Once the snake latches onto an object boundary, it rarely pulls itself through.

3.4 Testing the global velocity snake model

A series of experiments have been conducted to ascertain the effectiveness of the snakes to extract moving boundaries in various situations.

3.5 Experiments

As the vXu snake showed the least sensitivity to intial conditions in the qualitative analysis, it was chosen as the test candidate for the quantitative analysis. Simply, because it would be the model best suited to running in an autonomous framework without user input. The vXu snake was tested on synthetic images, against structured backgrounds from the Brodatz texture database, in varying levels of additive white Gaussian noise and also in cases of simulated occlusion. A set of performance metrics have also been developed to evaluate snake performance.

3.5.1 Performance Metrics

In order to test the performance of the snake, a meaningful test metric is required. A

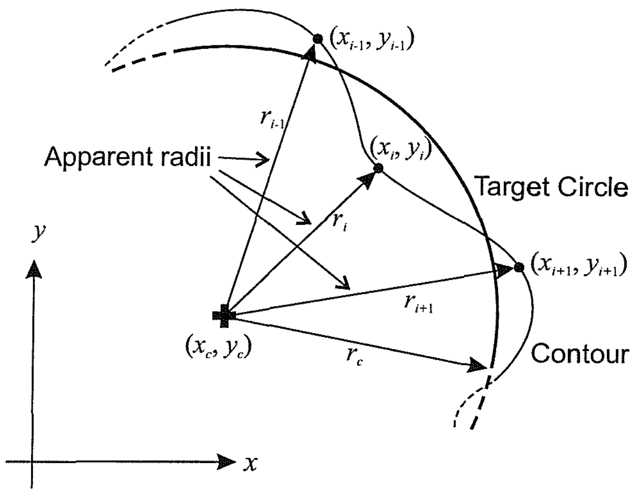


Figure 3.6: Calculating the Performance Metric

simple performance metric has been devised to make automated testing of both the Xu and vXu snake possible.

Considering a single contour seeking a target circle centred on (c_x, c_y) with radius r_c, we write the locus of the circle circumference, C, as in equation 3.10

$$C = (x_c + r_c \cos\theta, y_c + r_c \sin\theta) \tag{3.10}$$

. If the contour converges on the circle, the centre of the contour will sit at (c_x, c_y), and we can calculate the apparent radius for the contour at a point, $\mathbf{r}_i = (x_i, y_i)$, as in equation 3.11, and shown in figure 3.6.

$$r_i = \sqrt{(x_i - x_c)^2 + (y_i - x_c)^2} \tag{3.11}$$

We can calculate the distance from the contour along the normal to the circle as $Error_i = |r_i - r_c|$. To calculate the RMS error for the entire contour, we use equation 3.12

$$Error_{RMS} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\{r_c - \sqrt{(x_i - x_c)^2 + (y_i - y_c)^2}\}^2} \tag{3.12}$$

When evaluating a velocity aware snake, moving with velocity (v_x, v_y), and the target has velocity (u_x, u_y), equation 3.12 is modified to calculate the RMS error of all points on all contours in all frames:

$$\begin{aligned} Error_{RMS} &= \sqrt{\frac{1}{f+1}\sum_{f=0}^{frames-1} Error^2_{frame_n}} \\ Error_{frame_n} &= \frac{1}{n}\sum_{i=1}^{n}\{\sqrt{(x_i - x_c + f(v_x - u_x))^2 + (y_i - y_c + f(v_y - u_y))^2} - r_c\} \end{aligned} \tag{3.13}$$

In addition to calculating the RMS distance error for contour points, we can also calculate the percentage of contour points within some number, d, pixels of the target contour using equation 3.14.

$$\frac{100}{N}\sum_{n=1}^{N} E_n \quad \text{where} \quad E_n = \begin{cases} 0 & \text{if } |r_c - r_n| > d, \\ 1 & \text{otherwise} \end{cases} \tag{3.14}$$

This gives a measure of how well the final contour fits the target.

3.5.2 Additive White Gaussian Noise

The vXu snake is tested for its ability to reject noise due to the information averaging across the image sequences, these results are compared to the Xu snake.

The synthetic image sequences are of white discs moving against a black background. Each image then has White Gaussian noise with same variance added, and is then sobel edge detected, as can be seen in Figure 3.7

The contour is then iterated 100 times, with the contour position and calculated snake energy for each iteration being stored in a file, to allow the user to select the termination point using a results analysis tool. This allows the data generation to run autonomously, but also to allow the user to select the termination point. This is done as the greedy nature of the velocity search can cause jumps in the evolution energy and complicate the selection of a good autonomous termination criterion. In order to estimate the contour energy, the Greedy snake equations (2.14 and 2.15) and the E_{image} functional equation (3.6) are used. At each noise level, 50 runs are conducted, each with a different random number seed, producing a different noise pattern, but retaining statistical similarity.

The vXu snake is tested with a concentric initialisation, with a velocity close to the target velocity. The Xu snake is tested with the same initialisation as the vXu snake in the middle frame of the sequence. Section 3.6.1 shows the results of these trials.

3.5.3 Structured Static Background

This investigates the vXu snake's ability to reject regularly structured backgrounds when locating a moving object. The backgrounds are provided from the Brodatz texture database, where each frame in the sequence has the same background, with a white

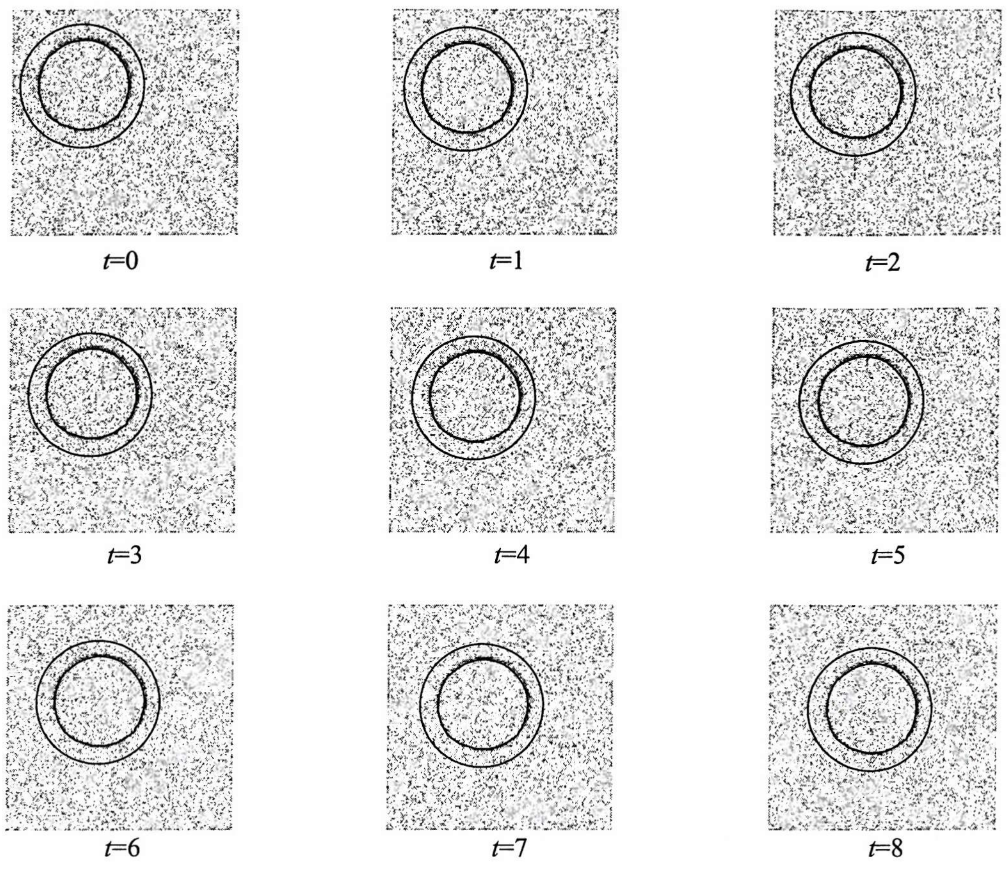


Figure 3.7: Initial and final positions of the contour. Initial Contour is the outer contour in each frame and the final result is the inner contour.

disc moving through the sequence, as shown in figure 3.8. The outer contour is the initialisation and the inner contour is the termination.

The Xu snake is also tested against the Brodatz database for comparison, but only in the centre frame of the vXu snake sequence. Section 3.6.2 shows the results of these trials.

3.5.4 Simulated Occlusion

This tests the ability of the vXu snake to interpolate missing data from other frames in the sequence.

Two types of occlusion are tested, bar occlusion and aperture occlusion. In bar occlusion, an occluding stripe is placed down the centre of each image to obscure a portion of the

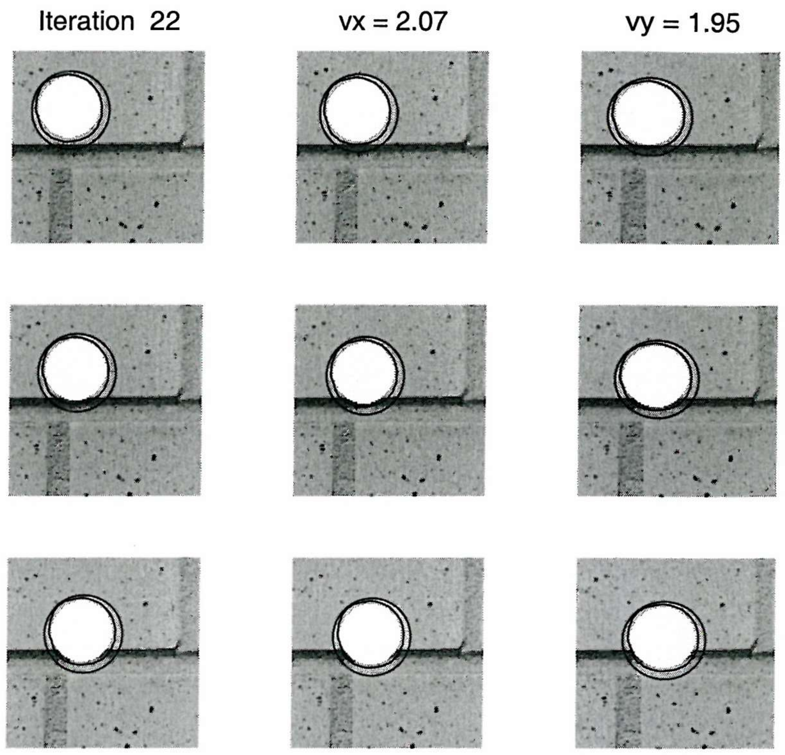


Figure 3.8: vXu snake being tested against Brodatz texture no. 25

target disc in each frame. In aperture occlusion, a window is placed on the centre of each image to show only a small section of each image in each frame, as demonstrated in figure 3.9.

3.5.4.1 Aperture

The vXu snake is tested with apertures of varying width and with discs moving at increasing velocities. If each part of the target occurs in at least one frame, the data averaging properties of the snake should extract the correct object contour. If the disc moves past the aperture with a large velocity, or if the aperture is narrow, only a small amount of data re-enforcement between frames can be achieved and the performance of the contour will suffer as a result.

The contour is initialised as shown in figure 3.9, where the outer line is the initialisation and the inner line is the final contour. The results are shown in section 3.6.3.1

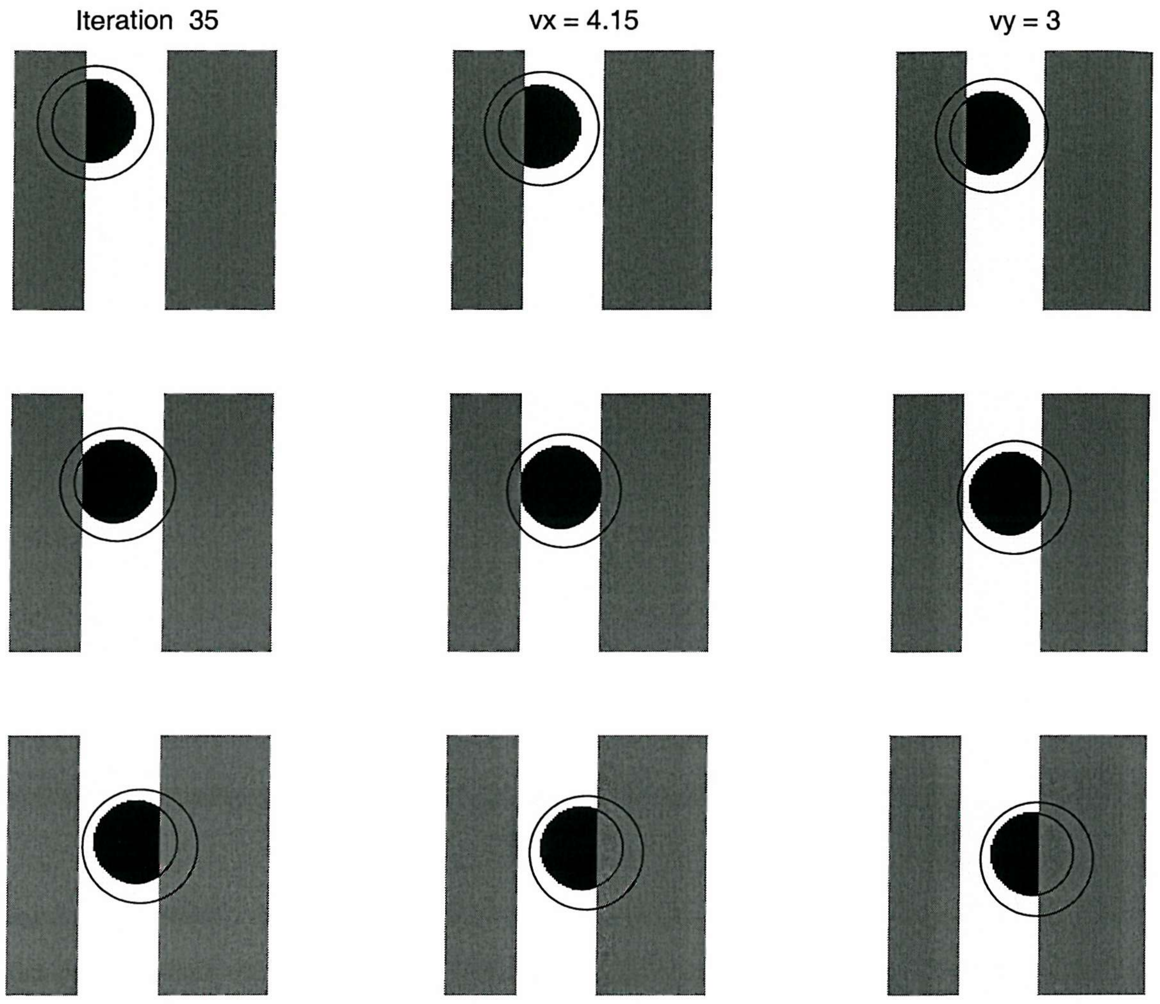


Figure 3.9: vXu snake being tested in aperture occlusion

3.5.4.2 Bar

As with the aperture occlusion, the vXu snake is tested with bars of varying width and with discs moving at increasing velocities. If each part of the target occurs in at least one frame, the data averaging properties of the snake should extract the correct object contour.

If the disc moves with a suitably large velocity, the data averaging properties of the contour will be able to reconstruct the contour of the disc, even if it is completely occluded in one frame. The results are shown in section 3.6.3.2.

3.6 Results

3.6.1 Additive White Gaussian Noise

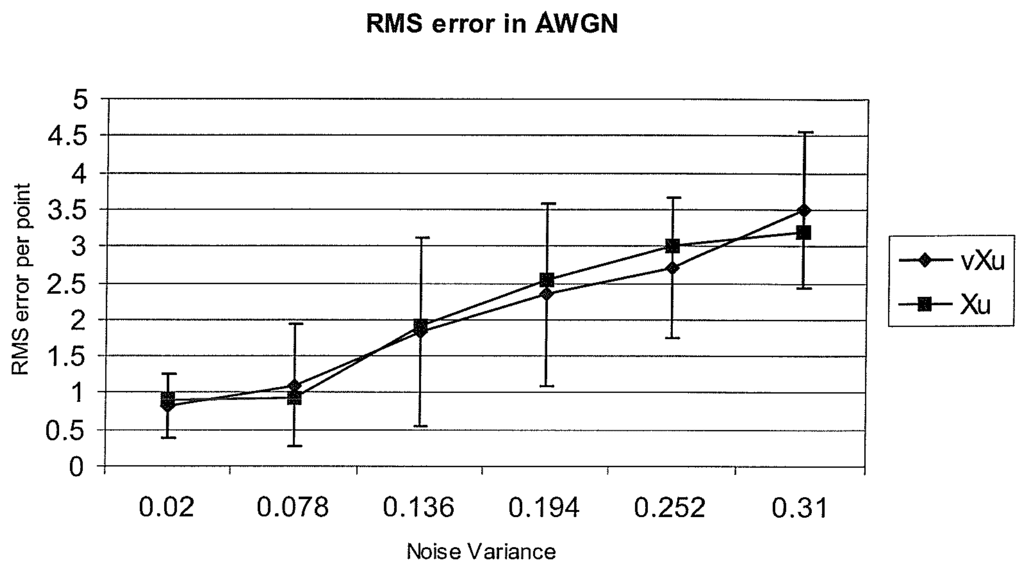


Figure 3.10: Comparison of noise performance of Xu and vXu snakes.

Figure 3.10 shows a comparison of noise performance between the velocity aware vXu snake and the original Xu. As can be seen, as the noise increases, so too does the RMS error, which is to be expected. The performance difference between the snakes is negligible, with the vXu snake giving marginally improved performance over the Xu snake. This difference is possibly due to the nature of the noise corrupting the boundary of the contour in all images and the averaging effect from the velocity component does not reduce the noise.

3.6.2 Structured Static Background

The vXu snake shows an improved error response compared to the Xu snake on most textures tested. This suggests that the averaging approach of the vXu snake does indeed improve the contour detection performance of the Xu snake.

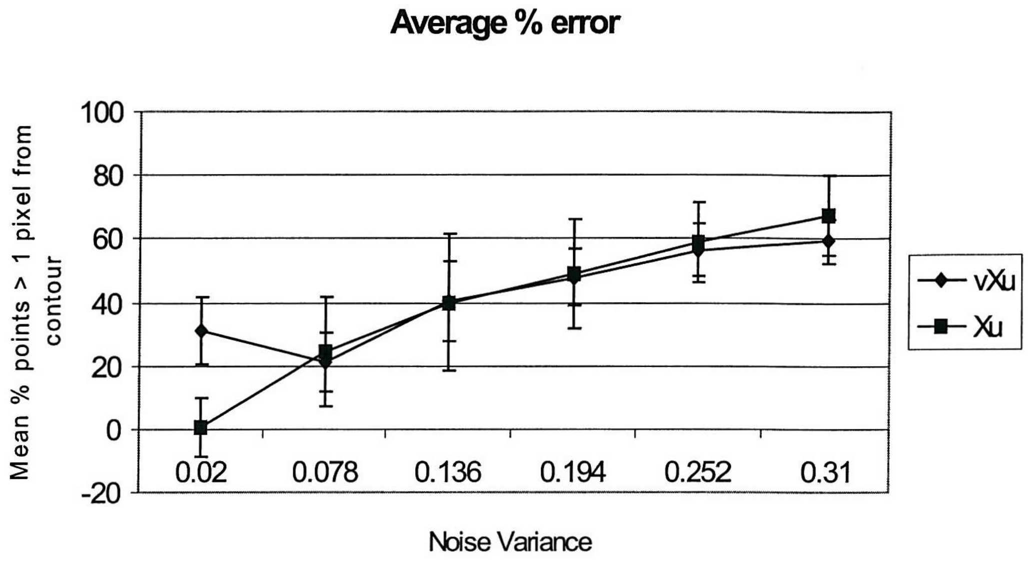


Figure 3.11: Comparison of noise performance of Xu and vXu snakes.

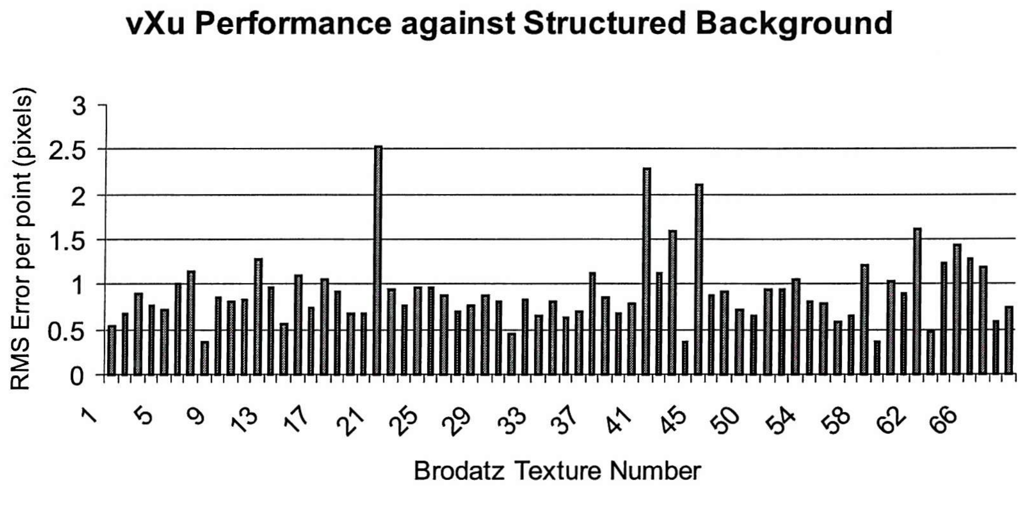


Figure 3.12: Performance of vXu snake against a variety of static backgrounds

3.6.3 Simulated Occlusion

The radius of the circle under investigation in these experiments is 20 pixels. The important velocity component in these results is the velocity in the x direction, vx, as the visible portion of the disc changes only if vx is non-zero.

3.6.3.1 Aperture

As can be seen in figure 3.14 there is a strong relationship between the velocity of the contour and the aperture width at which the snake fails to find the contour.

The interaction of the contour with the artifacts from the edges of the occluding bars prevent the contour finding the target completely in low velocity cases, even though the entire circle is visible in one frame.

Even if the aperture is smaller than the diameter of the disc the contour can still locate it, provided vx is sufficient that no part of the disc is obscured in every frame. In figure 3.15 the data suggests that the contour can have a low overall error, yet not find the target very well. This is due to the contour finding roughly the correct target shape, but not the correct velocity, with the frames towards the beginning of the sequence giving good localisation and in the frames towards the end of the sequence having an eccentric localisation around the target, giving a larger error than in the first frames.

3.6.3.2 Bar

In the case of Bar occlusion, the vXu snake again shows an ability to cope with missing data by the temporal averaging effect. Again there is a distinct relationship between the occlusion width, vx and the ability to fill in missing data. It can be seen in figure 3.16 that the contour can locate the target even though the width of occlusion completely obscures the target in frame 4 (the different colour of the occlusion in frame 4 is the result of normalising the image data to lie in the range $[0, 1]$).

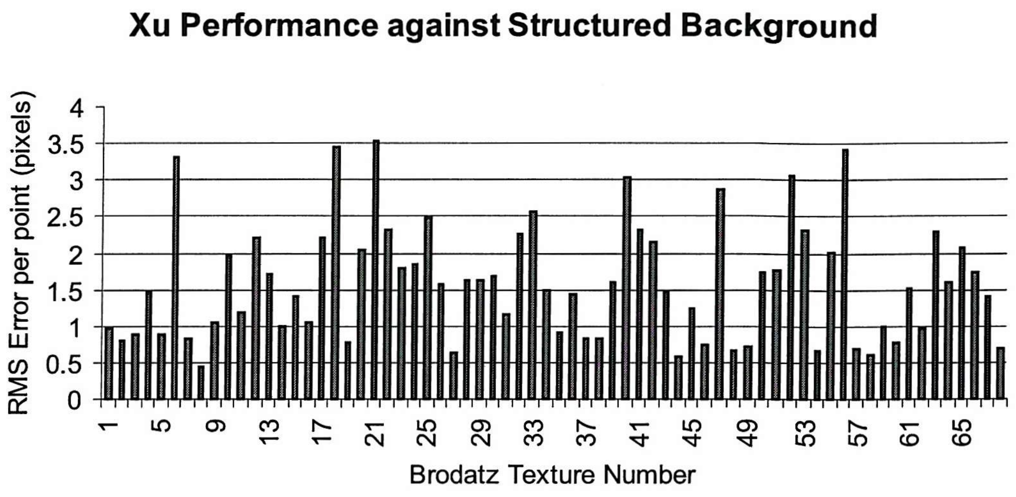


Figure 3.13: Performance of Xu snake against a variety of static backgrounds

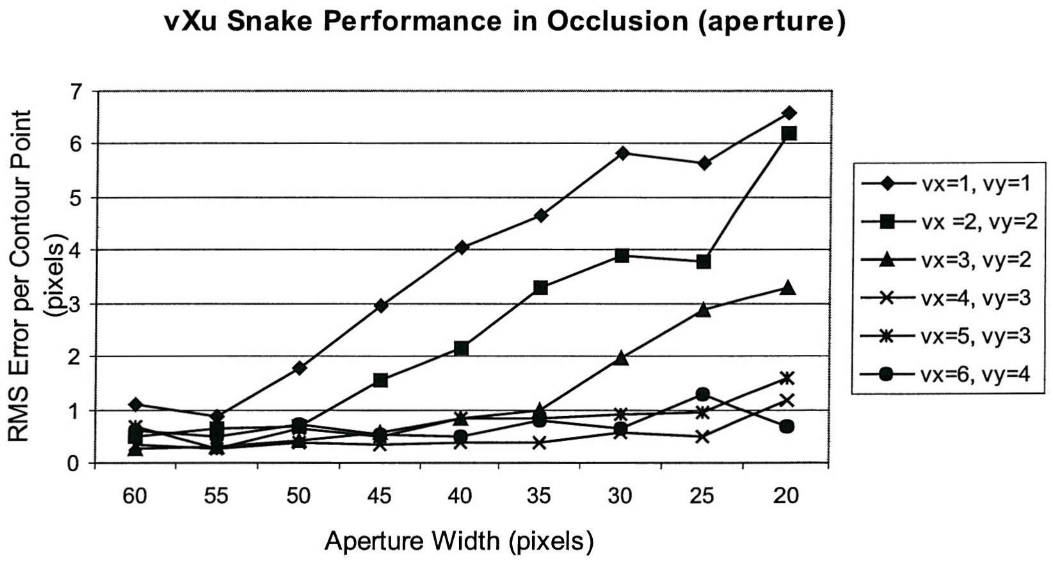


Figure 3.14: vXu snake performance with a simulated aperture of varying width and target velocities

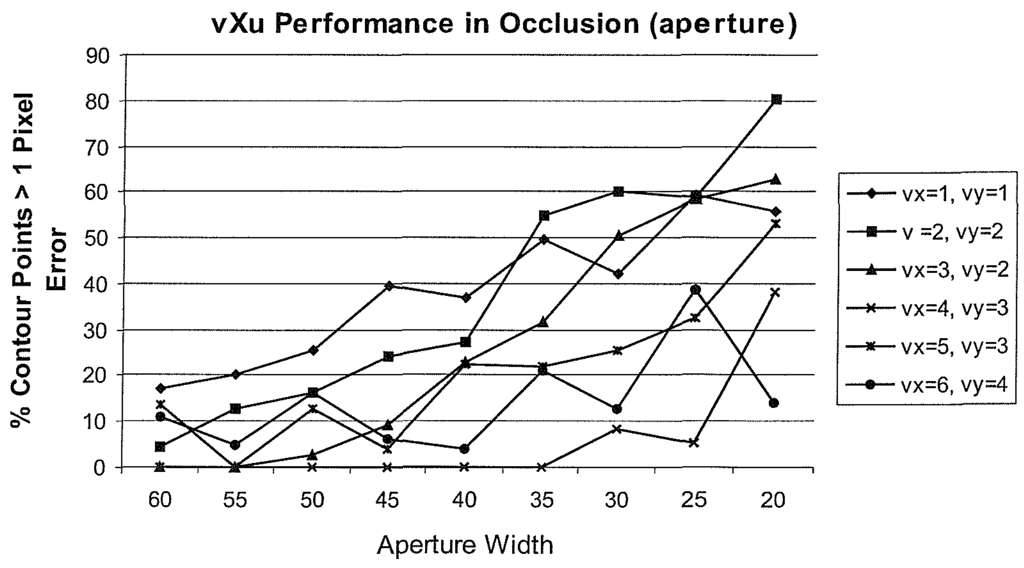


Figure 3.15: Goodness of fit for vXu snake in simulated aperture occlusion

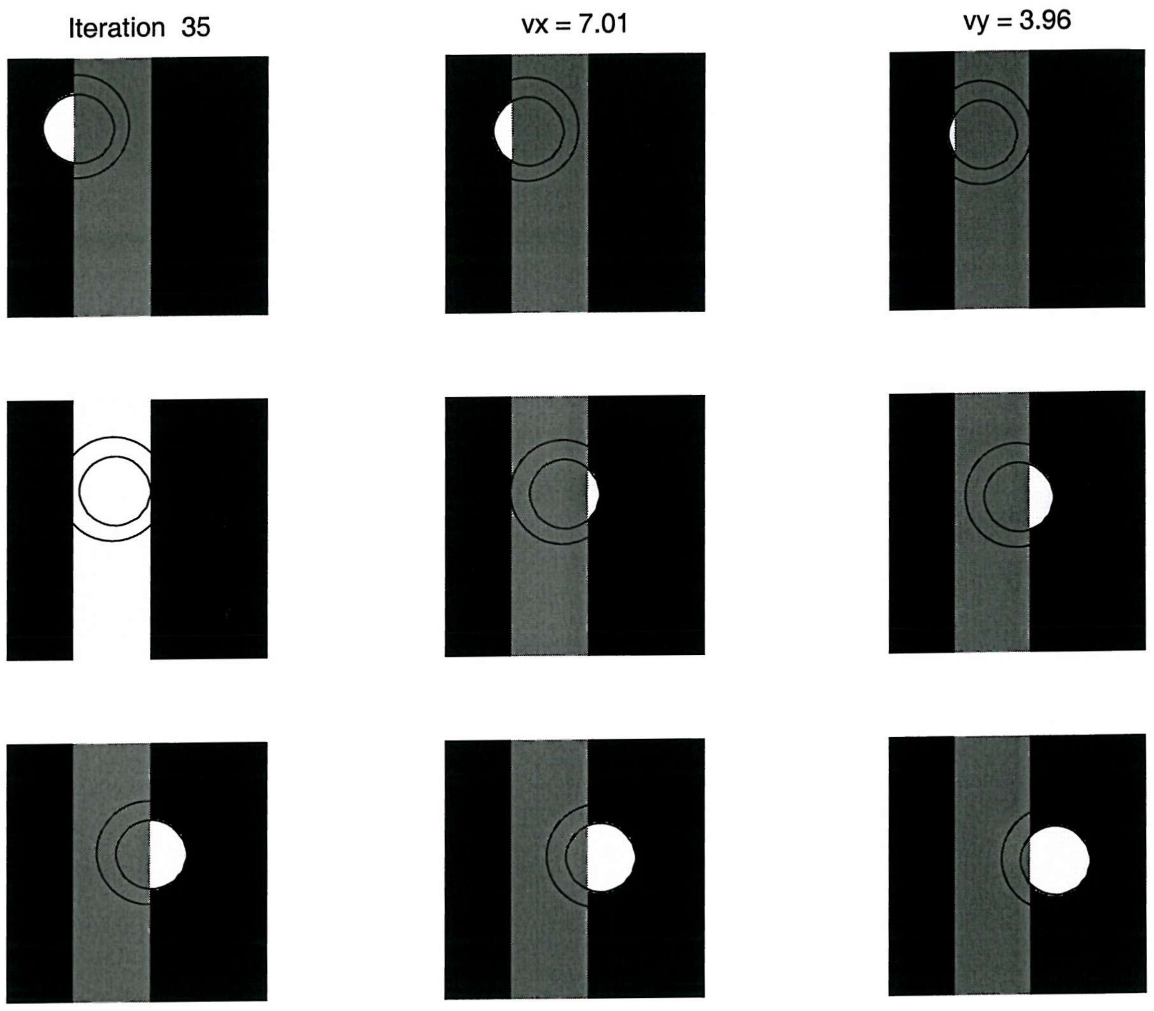


Figure 3.16: Good location even with target obscured in frame 4

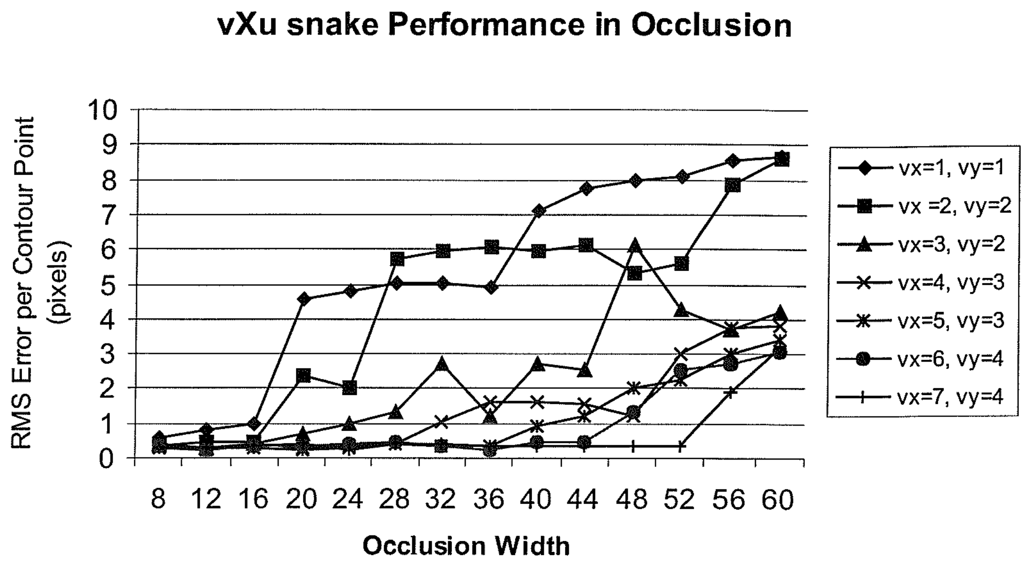


Figure 3.17: vXu snake performance with a simulated bar of varying width and target velocities

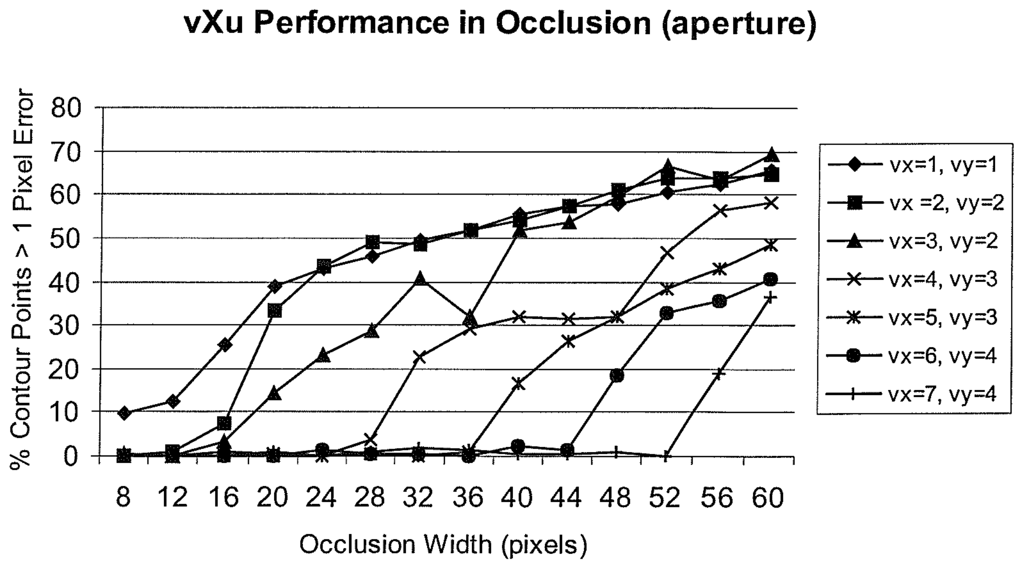


Figure 3.18: Goodness of fit for vXu snake in simulated bar occlusion

Chapter 4

Local Velocity Model

From the previous chapter we have seen that the global velocity snake model provides a good solution to finding moving objects, especially in the case of occlusion. However, the major flaw of this model is the hard constraints needed on the evolution of the velocity to ensure convergence to a moving shape. These hard constraints can be removed with the addition of a local velocity model.

When extending the snake model to incorporate a continuous time component the contour information needs to be integrated with respect to time and also with respect to the contour shape.

When the contour velocity of the global velocity snake changes often a small region of the contour can sufficiently reduce its contribution to the external energy to produce a reduction in the total contour energy. With this change in velocity the entire contour will move at the behest of a small section. This often results in moving another small section of the contour away from a feature of interest, and in turn, moving the contour away from the desired solution. If this moves the contour inside the object boundary, the internal forces of the snake contracts the contour, preventing it from returning to the correct solution.

To overcome this problem, the velocity is allowed to vary smoothly around the contour, with the contour arc length, s, now parameterising both the contour shape $\mathbf{r}(s)$ and velocity $\mathbf{v}(s)$, the local velocity snake contour is described thus,

$$\begin{aligned} \mathbf{w}(s) &= (\mathbf{r}(s),\ \mathbf{v}(s)), \\ &= (\mathbf{x}(s),\ \mathbf{y}(s),\ \mathbf{v}_x(s),\ \mathbf{v}_y(s)), \end{aligned} \tag{4.1}$$

and the position of the contour points at time t is described in Cartesian coordinates by,

$$\mathbf{p}(w(s),\ t) = (\mathbf{x}(s) + t\mathbf{v}_x(s),\ \mathbf{y}(s) + t\mathbf{v}_y(s)). \tag{4.2}$$

This permits local variations in velocity, enabling regions of the contour to explore the velocity space without affecting the entire contour. For Equation 4.1 to be meaningful, there must be continuity in the velocity around the contour, and because the target object does not deform with time, the velocity around the contour must also be constrained towards a single value.

4.1 Local Velocity Snake Implementation

By the calculus of variation, we shall consider an admissible solution, $\hat{\mathbf{w}}(s)$, perturbed by a small amount, $\epsilon\eta(s)$, which achieves minimum energy as:

$$\frac{\mathrm{d}E_{lvSnake}(\hat{\mathbf{w}}(s) + \epsilon\eta(s))}{\mathrm{d}\epsilon} = 0, \tag{4.3}$$

where the perturbation is spatial as the parameterisation will affect both the position and velocity of a snake point:

$$\eta(s) = (\eta_x(s),\ \eta_{vx}(s),\ \eta_y(s), \eta_{vy}(s)). \tag{4.4}$$

and,

$$\hat{\mathbf{w}}(s) = (\hat{\mathbf{x}}(s),\ \hat{\mathbf{v}}_x(s),\ \hat{\mathbf{y}}(s),\ \hat{\mathbf{v}}_y(s)) \tag{4.5}$$

This gives the perturbed snake in Cartesian coordinates at time, t, as:

$$\mathbf{p}(\hat{\mathbf{w}}(s)+\epsilon\eta(s),\ t)=\begin{pmatrix} \hat{\mathbf{x}}(s)+\epsilon\eta_x(s)+t(\hat{\mathbf{v}}_x(s)+\epsilon\eta_{vx}(s)) \\ \hat{\mathbf{y}}(s)+\epsilon\eta_y(s)+t(\hat{\mathbf{v}}_y(s)+\epsilon\eta_{vy}(s)) \end{pmatrix}^T, \tag{4.6}$$

$\hat{\mathbf{x}}(s)$ and $\hat{\mathbf{y}}(s)$ are the spatial x and y coordinates of the snake points, and $\hat{\mathbf{v}}_x(s)$ and $\hat{\mathbf{v}}_y(s)$ are the x and y velocities of the points at the solution:

We now define a new local velocity snake equation, $E_{lvSnake}$, derived from the original Snake Equation 2.1, as:

$$E_{lvSnake}(\mathbf{w}(s),t)=\iint_{t,s=0}^{1} E_{int}(\mathbf{w}(s))+E_{ext}(\mathbf{w}(s),t)\,\mathrm{d}s\,\mathrm{d}t. \tag{4.7}$$

Where the internal snake energy is not parameterised by time, as the snake shape is not intended to vary with respect to time, whereas the external energy does. In other words, changing $\mathbf{v}(s)$ will not alter $\mathbf{r}(s)$ directly through E_{int}, but if E_{ext} is varies by changing $\mathbf{v}(s)$, the regularisation imposed by E_{int} will act on $\mathbf{r}(s)$. This provides a mechanism for averaging image information across the frames in the image sequence.

Substituting for the perturbed snake points:

$$E_{lvSnake}(\hat{\mathbf{w}}(s)+\epsilon\eta(s),\ t)=\iint_{t,s=0}^{1} E_{int}(\hat{\mathbf{w}}(s)+\epsilon\eta(s))+E_{ext}(\hat{\mathbf{w}}(s)+\epsilon\eta(s),t)\,\mathrm{d}s\,\mathrm{d}t. \tag{4.8}$$

Incorporating the Kass internal snake energy defined in Equation 2.3,

$$E_{lvSnake}(\hat{\mathbf{w}}(s)+\epsilon\eta(s),t)=$$
$$\iint_{t,s=0}^{1}\left\{\begin{matrix} \alpha(s)\left|\dfrac{\mathrm{d}}{\mathrm{d}s}(\hat{\mathbf{w}}(s)+\epsilon\eta(s))\right|^2+\beta(s)\left|\dfrac{\mathrm{d}^2}{\mathrm{d}s^2}(\hat{\mathbf{w}}(s)+\epsilon\eta(s))\right|^2 \\ +E_{ext}(\hat{\mathbf{w}}(s)+\epsilon\eta(s),t)\end{matrix}\right\}\mathrm{d}s\,\mathrm{d}t. \tag{4.9}$$

Substituting for Equation 4.6,

$$E_{lvSnake}(\hat{\mathbf{w}}(s,t)+\epsilon\boldsymbol{\eta}(s,t)) =$$
$$\iint_{t,s=0}^{1}\left\{\begin{array}{l}
\alpha(s)\left\{\begin{array}{l}
\left(\frac{\mathrm{d}\hat{\mathbf{x}}(s)}{\mathrm{d}s}\right)^2+2\epsilon\frac{\mathrm{d}\hat{\mathbf{x}}(s)}{\mathrm{d}s}\frac{\mathrm{d}\boldsymbol{\eta}_x(s)}{\mathrm{d}s}+\epsilon^2\left(\frac{\mathrm{d}\boldsymbol{\eta}_x(s)}{\mathrm{d}s}\right)^2+\\
\left(\frac{\mathrm{d}\hat{\mathbf{v}}_x(s)}{\mathrm{d}s}\right)^2+2\epsilon\frac{\mathrm{d}\hat{\mathbf{v}}_x(s)}{\mathrm{d}s}\frac{\mathrm{d}\boldsymbol{\eta}_{vx}(s)}{\mathrm{d}s}+\epsilon^2\left(\frac{\mathrm{d}\boldsymbol{\eta}_{vx}(s)}{\mathrm{d}s}\right)^2+\\
\left(\frac{\mathrm{d}\hat{\mathbf{y}}(s)}{\mathrm{d}s}\right)^2+2\epsilon\frac{\mathrm{d}\hat{\mathbf{y}}(s)}{\mathrm{d}s}\frac{\mathrm{d}\boldsymbol{\eta}_y(s)}{\mathrm{d}s}+\epsilon^2\left(\frac{\mathrm{d}\boldsymbol{\eta}_y(s)}{\mathrm{d}s}\right)^2+\\
\left(\frac{\mathrm{d}\hat{\mathbf{v}}_y(s)}{\mathrm{d}s}\right)^2+2\epsilon\frac{\mathrm{d}\hat{\mathbf{v}}_y(s)}{\mathrm{d}s}\frac{\mathrm{d}\boldsymbol{\eta}_{vy}(s)}{\mathrm{d}s}+\epsilon^2\left(\frac{\mathrm{d}\boldsymbol{\eta}_{vy}(s)}{\mathrm{d}s}\right)^2
\end{array}\right\}+\\
\beta(s)\left\{\begin{array}{l}
\left(\frac{\mathrm{d}^2\hat{\mathbf{x}}(s)}{\mathrm{d}s^2}\right)^2+2\epsilon\frac{\mathrm{d}^2\hat{\mathbf{x}}(s)}{\mathrm{d}s^2}\frac{\mathrm{d}^2\boldsymbol{\eta}_x(s)}{\mathrm{d}s^2}+\epsilon^2\left(\frac{\mathrm{d}^2\boldsymbol{\eta}_x(s)}{\mathrm{d}s^2}\right)^2+\\
\left(\frac{\mathrm{d}^2\hat{\mathbf{v}}_x(s)}{\mathrm{d}s^2}\right)^2+2\epsilon\frac{\mathrm{d}^2\hat{\mathbf{v}}_x(s)}{\mathrm{d}s^2}\frac{\mathrm{d}^2\boldsymbol{\eta}_{vx}(s)}{\mathrm{d}s^2}+\epsilon^2\left(\frac{\mathrm{d}^2\boldsymbol{\eta}_{vx}(s)}{\mathrm{d}s^2}\right)^2+\\
\left(\frac{\mathrm{d}^2\hat{\mathbf{y}}(s)}{\mathrm{d}s^2}\right)^2+2\epsilon\frac{\mathrm{d}^2\hat{\mathbf{x}}(s)}{\mathrm{d}s^2}\frac{\mathrm{d}^2\boldsymbol{\eta}_y(s)}{\mathrm{d}s^2}+\epsilon^2\left(\frac{\mathrm{d}^2\boldsymbol{\eta}_y(s)}{\mathrm{d}s^2}\right)^2+\\
\left(\frac{\mathrm{d}^2\hat{\mathbf{v}}_y(s)}{\mathrm{d}s^2}\right)^2+2\epsilon\frac{\mathrm{d}^2\hat{\mathbf{v}}_y(s)}{\mathrm{d}s^2}\frac{\mathrm{d}^2\boldsymbol{\eta}_{vy}(s)}{\mathrm{d}s^2}+\epsilon^2\left(\frac{\mathrm{d}^2\boldsymbol{\eta}_{vy}(s)}{\mathrm{d}s^2}\right)^2
\end{array}\right\}\\
+E_{ext}(\hat{\mathbf{w}}(s)+\epsilon\boldsymbol{\eta}(s),t)
\end{array}\right\}\mathrm{d}s\,\mathrm{d}t \tag{4.10}$$

By expanding E_{ext} at the perturbed solution by Taylor series, we obtain

$$\begin{aligned}
&E_{ext}(\hat{\mathbf{w}}(s)+\epsilon\boldsymbol{\eta}(s),t)\\
&= E_{ext}(\hat{\mathbf{r}}(s)+\boldsymbol{\eta}_r(s)+t(\hat{\mathbf{v}}(s)+\epsilon\boldsymbol{\eta}_v(s))\\
&= E_{ext}(\hat{\mathbf{r}}(s),\hat{\mathbf{v}}(s))+\\
&\quad \epsilon\boldsymbol{\eta}_r(s)\frac{\partial E_ext}{\partial r}\bigg|_{\hat{\mathbf{x}},\hat{\mathbf{y}},\hat{\mathbf{v}}_x,\hat{\mathbf{v}}_y}+\epsilon\boldsymbol{\eta}_v(s)\frac{\partial E_ext}{\partial v}\bigg|_{\hat{\mathbf{x}},\hat{\mathbf{y}},\hat{\mathbf{v}}_x,\hat{\mathbf{v}}_y}+O(\epsilon^2)
\end{aligned}\tag{4.11}$$

The external image functional must be twice differentiable, which is true for edge information as well as intensity information. As ϵ is small, we disregard higher order terms, and reformulate Equation 4.10,

$$E_{lvSnake}(\hat{\mathbf{w}}(s,t)+\epsilon\boldsymbol{\eta}(s,t)) = E_{lvSnake}(\hat{\mathbf{w}}(s,t)) + \tag{4.12}$$
$$2\epsilon\iint_{t,s=0}^{1} \alpha(s)\frac{\mathrm{d}\hat{\mathbf{x}}(s)}{\mathrm{d}s}\frac{\mathrm{d}\boldsymbol{\eta}_x(s)}{\mathrm{d}s} + \beta(s)\frac{\mathrm{d}^2\hat{\mathbf{x}}(s)}{\mathrm{d}s^2}\frac{\mathrm{d}^2\boldsymbol{\eta}_x(s)}{\mathrm{d}s^2} + \frac{\boldsymbol{\eta}_x(s)}{2}\frac{\partial E_{ext}}{\partial x}\bigg|_{\hat{\mathbf{x}},\hat{\mathbf{y}},\hat{\mathbf{v}}_x,\hat{\mathbf{v}}_y} \mathrm{d}s\,\mathrm{d}t +$$
$$2\epsilon\iint_{t,s=0}^{1} \alpha(s)\frac{\mathrm{d}\hat{\mathbf{y}}(s)}{\mathrm{d}s}\frac{\mathrm{d}\boldsymbol{\eta}_y(s)}{\mathrm{d}s} + \beta(s)\frac{\mathrm{d}^2\hat{\mathbf{y}}(s)}{\mathrm{d}s^2}\frac{\mathrm{d}^2\boldsymbol{\eta}_y(s)}{\mathrm{d}s^2} + \frac{\boldsymbol{\eta}_y(s)}{2}\frac{\partial E_{ext}}{\partial y}\bigg|_{\hat{\mathbf{x}},\hat{\mathbf{y}},\hat{\mathbf{v}}_x,\hat{\mathbf{v}}_y} \mathrm{d}s\,\mathrm{d}t +$$
$$2\epsilon\iint_{t,s=0}^{1} \alpha(s)\frac{\mathrm{d}\hat{\mathbf{v}}_x(s)}{\mathrm{d}s}\frac{\mathrm{d}\boldsymbol{\eta}_{vx}(s)}{\mathrm{d}s} + \beta(s)\frac{\mathrm{d}^2\hat{\mathbf{v}}_x(s)}{\mathrm{d}s^2}\frac{\mathrm{d}^2\boldsymbol{\eta}_{vx}(s)}{\mathrm{d}s^2} + \frac{\boldsymbol{\eta}_{vx}(s)}{2}\frac{\partial E_{ext}}{\partial v_x}\bigg|_{\hat{\mathbf{x}},\hat{\mathbf{y}},\hat{\mathbf{v}}_x,\hat{\mathbf{v}}_y} \mathrm{d}s\,\mathrm{d}t +$$
$$2\epsilon\iint_{t,s=0}^{1} \alpha(s)\frac{\mathrm{d}\hat{\mathbf{v}}_y(s)}{\mathrm{d}s}\frac{\mathrm{d}\boldsymbol{\eta}_{vy}(s)}{\mathrm{d}s} + \beta(s)\frac{\mathrm{d}^2\hat{\mathbf{v}}_y(s)}{\mathrm{d}s^2}\frac{\mathrm{d}^2\boldsymbol{\eta}_{vy}(s)}{\mathrm{d}s^2} + \frac{\boldsymbol{\eta}_{vy}(s)}{2}\frac{\partial E_{ext}}{\partial v_x}\bigg|_{\hat{\mathbf{x}},\hat{\mathbf{y}},\hat{\mathbf{v}}_x,\hat{\mathbf{v}}_y} \mathrm{d}s\,\mathrm{d}t$$

Since we are seeking a minimum, the integrals of Equation 4.12 must be zero, giving four independent equations,

$$\iint_{t,s=0}^{1} \alpha(s)\frac{\mathrm{d}\hat{\mathbf{x}}(s)}{\mathrm{d}s}\frac{\mathrm{d}\boldsymbol{\eta}_x(s)}{\mathrm{d}s} + \beta(s)\frac{\mathrm{d}^2\hat{\mathbf{x}}(s)}{\mathrm{d}s^2}\frac{\mathrm{d}^2\boldsymbol{\eta}_x(s)}{\mathrm{d}s^2} + \frac{\boldsymbol{\eta}_x(s)}{2}\frac{\partial E_{ext}}{\partial x}\bigg|_{\hat{\mathbf{x}},\hat{\mathbf{y}},\hat{\mathbf{v}}_x,\hat{\mathbf{v}}_y} \mathrm{d}s\,\mathrm{d}t = 0 \tag{4.13}$$
$$\iint_{t,s=0}^{1} \alpha(s)\frac{\mathrm{d}\hat{\mathbf{y}}(s)}{\mathrm{d}s}\frac{\mathrm{d}\boldsymbol{\eta}_y(s)}{\mathrm{d}s} + \beta(s)\frac{\mathrm{d}^2\hat{\mathbf{y}}(s)}{\mathrm{d}s^2}\frac{\mathrm{d}^2\boldsymbol{\eta}_y(s)}{\mathrm{d}s^2} + \frac{\boldsymbol{\eta}_y(s)}{2}\frac{\partial E_{ext}}{\partial y}\bigg|_{\hat{\mathbf{x}},\hat{\mathbf{y}},\hat{\mathbf{v}}_x,\hat{\mathbf{v}}_y} \mathrm{d}s\,\mathrm{d}t = 0 \tag{4.14}$$
$$\iint_{t,s=0}^{1} \alpha(s)\frac{\mathrm{d}\hat{\mathbf{v}}_x(s)}{\mathrm{d}s}\frac{\mathrm{d}\boldsymbol{\eta}_{vx}(s)}{\mathrm{d}s} + \beta(s)\frac{\mathrm{d}^2\hat{\mathbf{v}}_x(s)}{\mathrm{d}s^2}\frac{\mathrm{d}^2\boldsymbol{\eta}_{vx}(s)}{\mathrm{d}s^2} + \frac{\boldsymbol{\eta}_{vx}(s)}{2}\frac{\partial E_{ext}}{\partial v_x}\bigg|_{\hat{\mathbf{x}},\hat{\mathbf{y}},\hat{\mathbf{v}}_x,\hat{\mathbf{v}}_y} \mathrm{d}s\,\mathrm{d}t = 0 \tag{4.15}$$
$$\iint_{t,s=0}^{1} \alpha(s)\frac{\mathrm{d}\hat{\mathbf{v}}_y(s)}{\mathrm{d}s}\frac{\mathrm{d}\boldsymbol{\eta}_{vy}(s)}{\mathrm{d}s} + \beta(s)\frac{\mathrm{d}^2\hat{\mathbf{v}}_y(s)}{\mathrm{d}s^2}\frac{\mathrm{d}^2\boldsymbol{\eta}_{vy}(s)}{\mathrm{d}s^2} + \frac{\boldsymbol{\eta}_{vy}(s)}{2}\frac{\partial E_{ext}}{\partial v_y}\bigg|_{\hat{\mathbf{x}},\hat{\mathbf{y}},\hat{\mathbf{v}}_x,\hat{\mathbf{v}}_y} \mathrm{d}s\,\mathrm{d}t = 0 \tag{4.16}$$

through integration by parts of Equation 4.13, we obtain:

$$\int_{t=0}^{1} \left\{ \begin{array}{c} \left[\alpha(s)\frac{\mathrm{d}\hat{\mathbf{x}}(s)}{\mathrm{d}s}\boldsymbol{\eta}(s)\right]_{s=0}^{1} - \int_{s=0}^{1} \frac{\mathrm{d}}{\mathrm{d}s}\left\{\alpha(s)\frac{\mathrm{d}\hat{\mathbf{x}}(s)}{\mathrm{d}s}\right\}\boldsymbol{\eta}_x(s)\mathrm{d}s + \\ \left[\beta(s)\frac{\mathrm{d}^2\hat{\mathbf{x}}(s)}{\mathrm{d}s^2}\frac{\mathrm{d}\boldsymbol{\eta}_x(s)}{\mathrm{d}s}\right]_{s=0}^{1} - \left[\frac{\mathrm{d}}{\mathrm{d}s}\left\{\beta(s)\frac{\mathrm{d}^2\hat{\mathbf{x}}(s)}{\mathrm{d}s^2}\right\}\boldsymbol{\eta}_x(s)\right]_{s=0}^{1} + \\ \int_{s=0}^{1} \frac{\mathrm{d}^2}{\mathrm{d}s^2}\left\{\beta(s)\frac{\mathrm{d}^2\hat{\mathbf{x}}(s)}{\mathrm{d}s^2}\right\}\boldsymbol{\eta}_x(s)\mathrm{d}s + \\ \frac{1}{2}\int_{s=0}^{1} \frac{\partial E_{ext}}{\partial x}\bigg|_{\hat{\mathbf{x}},\hat{\mathbf{y}},\hat{\mathbf{v}}_x,\hat{\mathbf{v}}_y} \boldsymbol{\eta}_x(s)\mathrm{d}s \end{array} \right\} \mathrm{d}t = 0 \tag{4.17}$$

As this is a closed contour, $\boldsymbol{\eta}_x(1) - \boldsymbol{\eta}_x(0) = 0$ and $\boldsymbol{\eta}_y(1) - \boldsymbol{\eta}_y(0) = 0$, making the first, third and fourth terms zero, reducing Equation 4.13 to,

$$\iint_{t,s=0}^{1} \left\{ -\frac{\mathrm{d}}{\mathrm{d}s}\left\{\alpha(s)\frac{\mathrm{d}\hat{\mathbf{x}}(s)}{\mathrm{d}s}\right\} + \frac{\mathrm{d}^2}{\mathrm{d}s^2}\left\{\beta(s)\frac{\mathrm{d}^2\hat{\mathbf{x}}(s)}{\mathrm{d}s^2}\right\} + \frac{1}{2}\left.\frac{\partial E_{ext}}{\partial x}\right|_{\hat{\mathbf{x}},\hat{\mathbf{y}},\hat{\mathbf{v}}_x,\hat{\mathbf{v}}_y} \right\} \eta_x(s)\mathrm{d}s\,\mathrm{d}t = 0 \tag{4.18}$$

Since this is valid for all $\eta_x(s)$, and t is independent of s,

$$\int_{t=0}^{1} \left\{ -\frac{\mathrm{d}}{\mathrm{d}s}\left\{\alpha(s)\frac{\mathrm{d}\hat{\mathbf{x}}(s)}{\mathrm{d}s}\right\} + \frac{\mathrm{d}^2}{\mathrm{d}s^2}\left\{\beta(s)\frac{\mathrm{d}^2\hat{\mathbf{x}}(s)}{\mathrm{d}s^2}\right\} + \frac{1}{2}\left.\frac{\partial E_{ext}}{\partial x}\right|_{\hat{\mathbf{x}},\hat{\mathbf{y}},\hat{\mathbf{v}}_x,\hat{\mathbf{v}}_y} \right\} \mathrm{d}t = 0 \tag{4.19}$$

Integrating and dividing through by t,

$$-\frac{\mathrm{d}}{\mathrm{d}s}\left\{\alpha(s)\frac{\mathrm{d}\hat{\mathbf{x}}(s)}{\mathrm{d}s}\right\} + \frac{\mathrm{d}^2}{\mathrm{d}s^2}\left\{\beta(s)\frac{\mathrm{d}^2\hat{\mathbf{x}}(s)}{\mathrm{d}s^2}\right\} + \frac{1}{2t}\int_{t=0}^{1} \left\{ \left.\frac{\partial E_{ext}}{\partial x}\right|_{\hat{\mathbf{x}},\hat{\mathbf{y}},\hat{\mathbf{v}}_x,\hat{\mathbf{v}}_y} \right\} \mathrm{d}t = 0 \tag{4.20}$$

by applying a similar treatment to Equations 4.14, 4.15 and 4.16, and combining them, we obtain the Euler PDE of Equation 4.7:

$$-\alpha(\mathbf{r}_{ss} + \mathbf{v}_{ss}) + \beta(\mathbf{r}_{ssss} + \mathbf{v}_{ssss}) + \frac{1}{2t}\int_{t=0}^{1} \left\{ \mathbf{F}_{rv}|_{\hat{\mathbf{r}},\hat{\mathbf{v}}} \right\} \mathrm{d}t = 0, \tag{4.21}$$

where $\mathbf{F}$ is the image functional and subscripts denote partial derivatives. This PDE yields a system of equations which can be solved iteratively by finite element analysis.

A general method of deriving the Euler PDE of an equation in two variables is given in Appendix A.

4.2 Solving the Euler PDE

We can solve Equation 4.21 by the method of finite differences. Regarding only the x coordinates initially, and using the approximations,

$$\frac{\mathrm{d}x}{\mathrm{d}s} \simeq \frac{1}{h}(\mathbf{x}_{s+1} - \mathbf{x}_s), \tag{4.22}$$

$$\frac{\mathrm{d}^2x}{\mathrm{d}s^2} \simeq \frac{1}{h^2}(\mathbf{x}_{s+1} - 2\mathbf{x}_s + \mathbf{x}_{s-1}), \tag{4.23}$$

for the first and second order differences respectively and using the following approximation for the integral term,

$$\frac{1}{2t}\int_{t=0}^{1}\left\{\left.\frac{\partial E_{ext}}{\partial x}\right|_{\hat{\mathbf{x}},\hat{\mathbf{y}},\hat{\mathbf{v}}_x,\hat{\mathbf{v}}_y}\right\}\mathrm{d}t \simeq \frac{1}{2N}\sum_{n=0}^{N-1}\left.\frac{\partial E_{ext}}{\partial \mathbf{x}}\right|_{(\mathbf{x}_s+n\mathbf{v}_{xs},\ \mathbf{y}_s+n\mathbf{v}_{ys})} \tag{4.24}$$

for a sequence of N equally time-spaced frames. Equation 4.20 for a contour discretised into S equi-spaced points with an arc length h gives:

$$-\frac{1}{h}\left\{\alpha_{s+1}\frac{(\mathbf{x}_{s+1}-\mathbf{x}_s)}{h}-\alpha_s\frac{\mathbf{x}_s-\mathbf{x}_{s-1}}{h}\right\}+$$
$$\frac{1}{h^2}\left\{\beta_{s+1}\frac{(\mathbf{x}_{s+2}-2\mathbf{x}_{s+1}+\mathbf{x}_s)}{h^2}-2\beta_s\frac{(\mathbf{x}_{s+1}-2\mathbf{x}+\mathbf{x}_{s-1})}{h^2}+\beta_{s-1}\frac{(\mathbf{x}_s-2\mathbf{x}_{s-1}+\mathbf{x}_{s-2})}{h^2}\right\}+$$
$$\frac{1}{2N}\sum_{n=0}^{N-1}\left.\frac{\partial E_{ext}}{\partial \mathbf{x}}\right|_{(\mathbf{x}_s+n\mathbf{v}_{xs},\ \mathbf{y}_s+n\mathbf{v}_{ys})}=0 \tag{4.25}$$

by collecting coefficients of different points, equation 4.2 can be re-written as

$$f_s = \mathbf{a}_s\mathbf{x}_{s-2}+\mathbf{b}_s\mathbf{x}_{s-1}+\mathbf{c}_s\mathbf{x}_s+\mathbf{d}_s\mathbf{x}_{s+1}+\mathbf{e}_s\mathbf{x}_{s+2} \tag{4.26}$$

where,

$$\begin{aligned}
f_s &= -\frac{1}{2N}\sum_{n=0}^{N-1}\left.\frac{\partial E_{ext}}{\partial x}\right|_{(\mathbf{x}_s+n\mathbf{v}_{xs},\ \mathbf{y}_s+n\mathbf{v}_{xs})}\\
\mathbf{a}_s &= \frac{\beta_{s-1}}{h^4}\\
\mathbf{b}_s &= -\frac{2(\beta_s+\beta_{s-1})}{h^4}-\frac{\alpha_s}{h^2}\\
\mathbf{c}_s &= \frac{\beta_{s+1}+4\beta_s+\beta_{s-1}}{h^4}+\frac{\alpha_{s+1}+\alpha_s}{h^2}\\
\mathbf{d}_s &= -\frac{2(\beta_{s+1}+\beta_s)}{h^4}-\frac{\alpha_{s+1}}{h^2}\\
\mathbf{e}_s &= \frac{\beta_{s+1}}{h^4}
\end{aligned} \tag{4.27}$$

Expressed as a matrix equation:

$$\mathbf{Ax}=\frac{1}{2N}\sum_{n=0}^{N-1}\mathbf{F}_x(\mathbf{r}_x+n\mathbf{v}_x,\ \mathbf{r}_y+n\mathbf{v}_y) \tag{4.28}$$

Where $\mathbf{F}_x$ is the first order difference of the image energy along the x axis and where $\mathbf{A}$ is the penta-diagonal banded matrix:

$$\mathbf{A} = \begin{bmatrix} c_1 & d_1 & e_1 & 0 & \cdots & a_1 & b_1 \\ b_2 & c_2 & d_2 & e_2 & 0 & \cdots & a_1 \\ a_3 & b_3 & c_3 & d_3 & e_3 & 0 & \\ \vdots & \vdots & \vdots & \vdots & \vdots & & \\ & & & & & & \\ e_{s-1} & 0 & \cdots & a_{s-1} & b_{s-1} & c_{s-1} & d_{s-1} \\ d_s & e_s & 0 & \cdots & a_s & b_s & c_s \end{bmatrix} \tag{4.29}$$

Essentially, the snake matrix, $\mathbf{A}$, performs a low pass filtering on the contour and propagates contour shape information around the contour at each iteration.

Applying a similar treatment to the terms in $\mathbf{y}$, $\mathbf{v}_x$ and $\mathbf{v}_y$ of Equation 4.21 yields:

$$\mathbf{Ay} = \frac{1}{2N}\sum_{n=0}^{N-1}\mathbf{F}_y(\mathbf{r}_x + n\mathbf{v}_x,\ \mathbf{r}_y + n\mathbf{v}_y) \tag{4.30}$$

$$\mathbf{Av}_x = \frac{1}{2N}\sum_{n=0}^{N-1}\mathbf{F}_{v_x}(\mathbf{r}_x + n\mathbf{v}_x,\ \mathbf{r}_y + n\mathbf{v}_y) \tag{4.31}$$

$$\mathbf{Av}_y = \frac{1}{2N}\sum_{n=0}^{N-1}\mathbf{F}_{v_x}(\mathbf{r}_x + n\mathbf{v}_x,\ \mathbf{r}_y + n\mathbf{v}_y) \tag{4.32}$$

where $\mathbf{F}_y$ is the first order difference of the image functional along the y axis. $\mathbf{F}_{v_x}$ is an approximation of the first order difference in $\mathbf{v}_x$ and $\mathbf{F}_{v_y}$ is an approximation of the first order difference in $\mathbf{v}_y$ given by,

$$\mathbf{F}_{v_x}|_{(\mathbf{x}+n\mathbf{v}_x,\mathbf{y}+n\mathbf{v}_y)} \simeq \mathbf{F}|_{(\mathbf{x}+n(\mathbf{v}_x+\delta_{\mathbf{v}_x}),\mathbf{y}+n(\mathbf{v}_y+\delta_{\mathbf{v}_y}))} - \mathbf{F}|_{(\mathbf{x}+n\mathbf{v}_x,\mathbf{y}+n\mathbf{v}_y)} \tag{4.33}$$

where

$$\delta_{\mathbf{v}_x} = \frac{1}{N}\mathbf{v}_x, \quad \text{and} \tag{4.34}$$

$$\delta_{\mathbf{v}_y} = \frac{1}{N}\mathbf{v}_y. \tag{4.35}$$

4.2.1 Evolution Equation

The contour is solved for using the semi-explicit equations,

$$\begin{aligned}
\mathbf{x}^{<i+1>} &= (\mathbf{A}+\gamma\mathbf{I})^{-1}(\gamma\mathbf{x}^{<i>} + \frac{1}{2N}\sum_{n=0}^{N-1}\mathbf{F}_x(\mathbf{x}^{<i>} + n\mathbf{v}_x^{<i>}, \mathbf{y}^{<i>} + n\mathbf{v}_y^{<i>})), \quad (4.36)\\
\mathbf{y}^{<i+1>} &= (\mathbf{A}+\gamma\mathbf{I})^{-1}(\gamma\mathbf{y}^{<i>} + \frac{1}{2N}\sum_{n=0}^{N-1}\mathbf{F}_y(\mathbf{x}^{<i>} + n\mathbf{v}_x^{<i>}, \mathbf{y}^{<i>} + n\mathbf{v}_y^{<i>})), \quad (4.37)\\
\mathbf{x}^{<i+1>} &= (\mathbf{B}+\gamma\mathbf{I})^{-1}(\gamma\mathbf{x}^{<i>} + \frac{1}{2N}\sum_{n=0}^{N-1}\mathbf{F}_{v_x}(\mathbf{x}^{<i>} + n\mathbf{v}_x^{<i>}, \mathbf{r}_y^{<i>} + n\mathbf{y}^{<i>})), (4.38)\\
\mathbf{v}_y^{<i+1>} &= (\mathbf{B}+\gamma\mathbf{I})^{-1}(\gamma\mathbf{v}_y^{<i>} + \frac{1}{2N}\sum_{n=0}^{N-1}\mathbf{F}_{v_y}(\mathbf{x}^{<i>} + n\mathbf{v}_x^{<i>}, \mathbf{r}_y^{<i>} + n\mathbf{y}^{<i>})). (4.39)
\end{aligned}$$

where i is an evolution index, $\mathbf{r}^{<i>}$ is the array of current x coordinates of the contour and $\mathbf{r}^{<i+1>}$ is the array of x coordinates for the next iteration. γ is the evolution constraint parameter, which sets the evolution step size. Large values of γ make the contour evolve in large steps, but with the possibility of stepping over features of interest, while a small γ can take many more iterations to reach a stable solution.

A and **B** are both penta-diagonal banded matrices, which allow $\alpha(s)$ and $\beta(s)$ to be selected independently for the spatial and velocity evolution.

This model essentially forms a pair of Kass snakes, one which controls spatial evolution of the contour, and uses the evolution matrix **A**, while the other controls the evolution of the velocity contour using the evolution matrix **B**. In order to apply a strong constraint on the velocity evolution, to ensure the velocity around the contour tends to a singular value, it is necessary to set $\alpha(s)$ and $\beta(s)$ to be larger for **B** than **A**.

4.2.2 A note on computational efficiency

If the contour is to be re-parameterised at each step to provide a constant point spacing, the snake matrices A and B need to be inverted at each iteration, and this provides a serious computational load on the evolution of the local velocity snake. An explicit formulation has been proposed by Cohen and Cohen [5], but this does not perform satisfactorally in this formulation.

4.3 Local velocity model testing

The snake is tested on structured backgrounds (brodatz database) and also is shown to work on a real image sequence.

In these tests, we show how the snake evolves to find a moving object, which moves at a velocity different to the snakes initial velocity, unlike the Global velocity model.

The structured background tests are of a bean-shaped synthetic object on a series of brodatz textures, with the aim of extracting the target boundry without the close initialiasations which characterise the global velocity testing.

4.3.1 Similarity measure

For these experiments a non-circular shape has been used, leaving the radial metric of Section 3.5.1 unsuitable. This has led to the deployment of complex Fourier Descriptors, first developed by Granlund [6].

The chosen metric is to measure the mean square deviation of contours in the frequency domain. Fourier descriptors allow arbitrary shape description and have the advantageous property of being computed directly from the discrete contour description, without having to re-sample the contour. They offer a compact shape description and the first 10 terms are usually sufficient to form a good approximation of a shape [9].

We measure the simmilarity between an evolved contour, $\mathbf{r}$, and a reference contour, $\mathbf{q}$. The chosen measure is the Euclidean feature distance of the contours' fourier descriptors, μ_n and v_n,

$$d(\mathbf{r},\mathbf{q})\sqrt{\sum_{n=-M}^{M} |\mu_n - v_n|^2} \tag{4.40}$$

We describe the contour as a complex vector,

$$z(s) = x(s) + jy(s) \tag{4.41}$$

Where s is the contour arc length and $0 \leq s \leq L$, where L is the contour length. The Fourier descriptor transform pair for a continuous contour are defined as:

$$\begin{aligned} a_n &= \frac{1}{L}\int_{s=0}^{L} z(s)e^{-jn(\frac{2\pi}{L})s}\mathrm{d}s, \\ z(s) &= \sum_{n=-\infty}^{\infty} a_n e^{jn\frac{2\pi}{L}s} \end{aligned} \tag{4.42}$$

and the Fourier descriptors for a discrete contour, $\mathbf{r}_k$, is given by:

$$\begin{aligned} a_n &= \begin{cases} \frac{1}{2L}\sum_{k=1}^{N} |\mathbf{r}_{k+1} - \mathbf{r}_k| \left(\mathbf{r}_{k+1} + \mathbf{r}_k\right) & n = 0 \\ \frac{1}{4\pi n^2}\sum_{k=1}^{N}(b_{k-1} - b_k)e^{-jn\frac{2\pi}{L}l_s} & n \neq 0 \end{cases} \\ l_s &= \sum_{i=1}^{k} |\mathbf{r}_i - \mathbf{r}_{i-1}|, \quad l_0 = 0 \\ b_k &= \frac{\mathbf{r}_k + 1 - \mathbf{r}_k}{|\mathbf{r}_{k+1} - \mathbf{r}_k|} \end{aligned} \tag{4.43}$$

As we are only interested in shapes of a fixed size, there is no need to apply a normalisation to the descriptors to make them invariant of contour size.

We do however, need to be careful to select the same starting point for each contour. This is done by starting at the point with zero argument with respect to the first moment of area of the contour.

4.3.2 Brodatz results

The local velocity snake was tested against the Brodatz texture database, with the aim of testing the snakes ability to converge on the target velocity against a structured background. The result is presented in Figure 4.1. The images are 100 by 100 pixels, and cropped from the bottom left of each texture. The image is pre-processed by Gaussian smoothing followed by Sobel edge detection, before being normalised in the range $[0, 1)$. Figure 4.1 shows the lowest Mean Square Error for the first 10 Fourier descriptors for finding the bean shape against the Brodatz database. The initialisation is as shown in Figure 4.2, and the vast majority of initialisations performed well in locating the contour, despite the large velocity offset. Figure 4.2 shows a good location result of the contour on Brodatz texture number 22. The circular contour in each frame is the initialisation and the inner contour is the final position after 400 iterations. In spite of the large error in the initial velocity, the contour finds the target shape with a good location.

4.3.3 Real Image Sequences

Figure 4.3 shows the contour on a real image sequence. The image sequence is a taken from the video database held at KOGS/IAKS Universitaet Karlsruhe Germany and is of cars moving through an intersection.

The images are from frames 34 through to 42 in the sequence and have been Gaussian smoothed with a three-by-three kernel, and normalized over the range $[0, 1]$. The image intensity functional was chosen for E_{ext}, as there is sufficient contrast between the vehicle and the road to segment the image easily. The filtering has smoothed out the raster scan-lines from the video footage.

The contour has located the car in the image, despite small distortions in the vehicles outline arising from small changes in perspective of the vehicle as it moves.

4.3.4 Pathalogical Data

The sequence has a number of lines running parallel to the flow of traffic, namely pavements, street markings and noticeably a shadow border on the pavement on the left of the junction. Figure 4.4 shows the contour preferring to attach itself to the shadow than the moving vehicle.

Any line lying parallel to the direction of motion will act as a locus of points moving with a velocity equal to that of the motion. And the contour will deform to match it. As is clearly visible, the final contour (the light one) has attahced itself to the right hand side of the car, but the left hand side was ignored in preference to the shadows

By modifying the initial position, the contour has found the car in Figure 4.3.

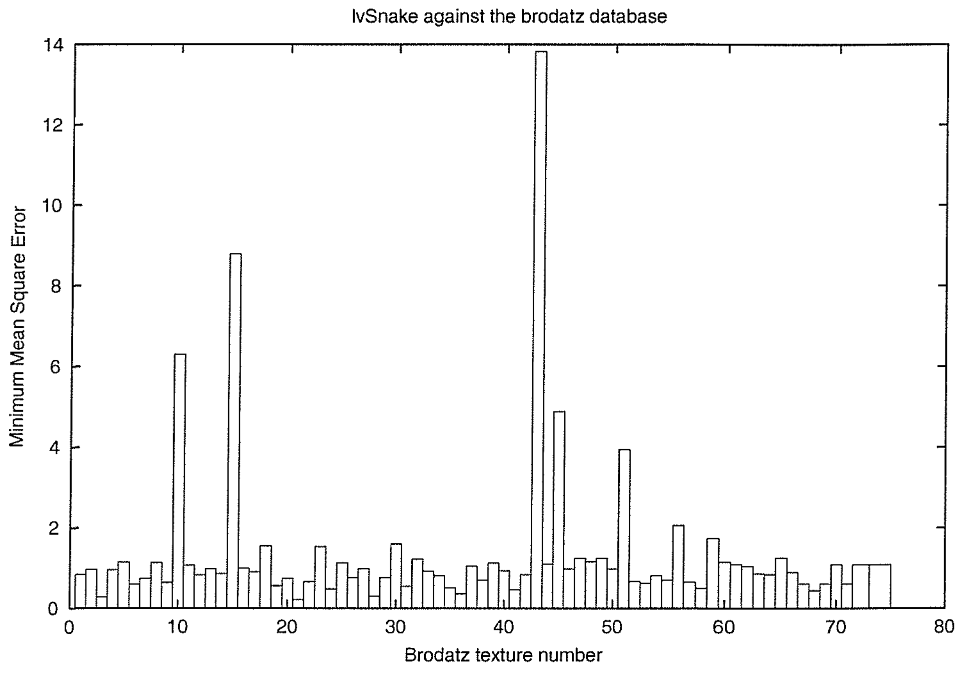


Figure 4.1: MSE results for the Brodatz database for the lvSnake

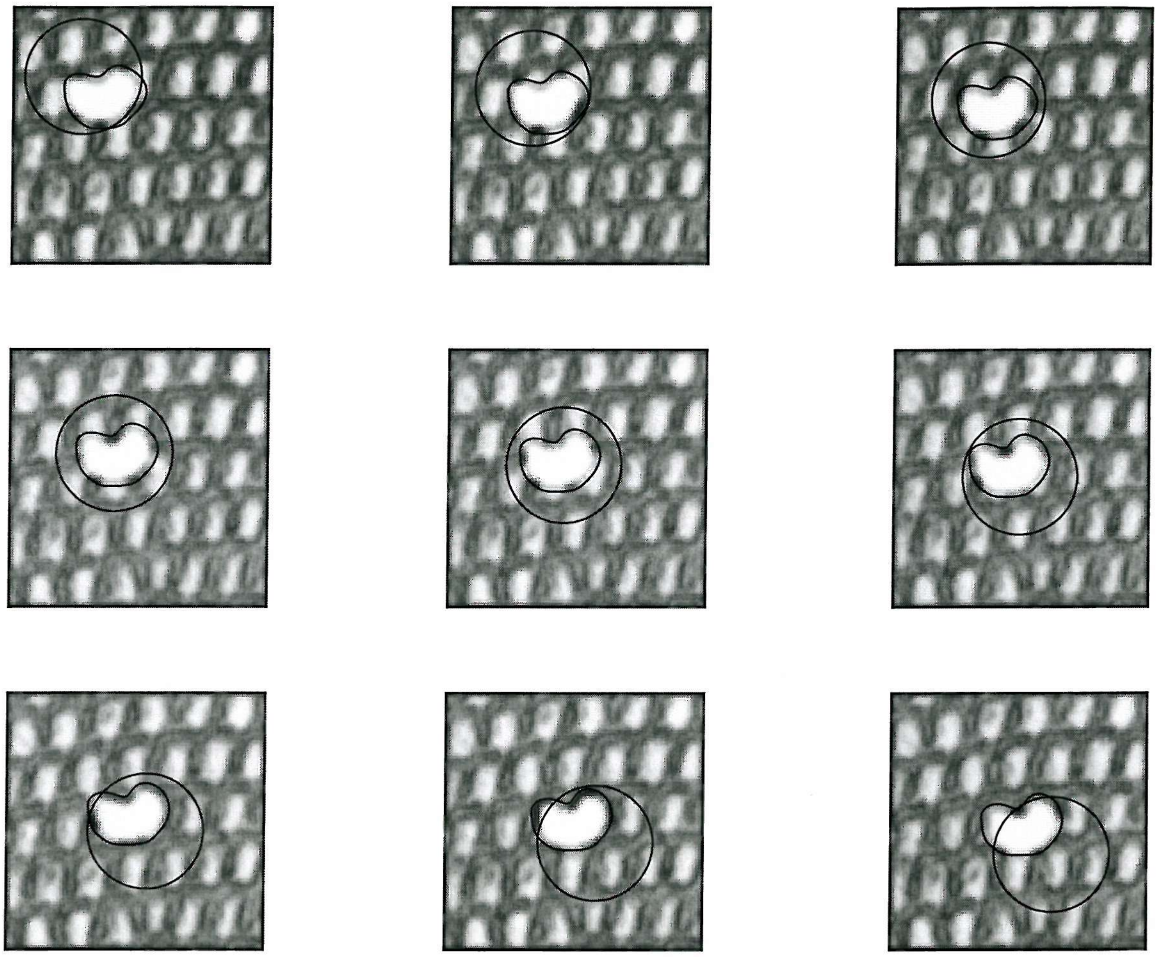


Figure 4.2: Finding the bean boundary on Brodatz texture 22, despite large error in initial velocity

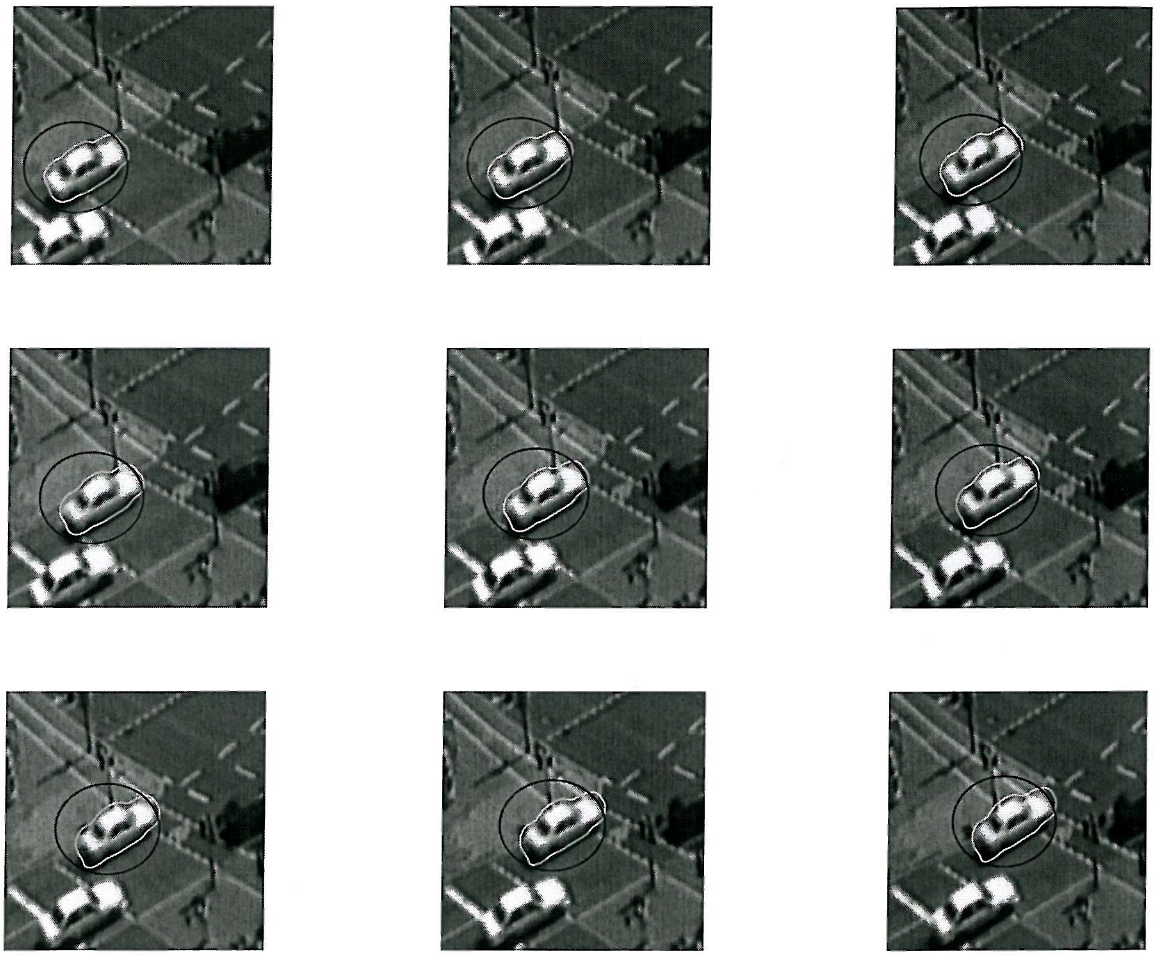


Figure 4.3: Location of the white car from the KOGS/IAKS sequence

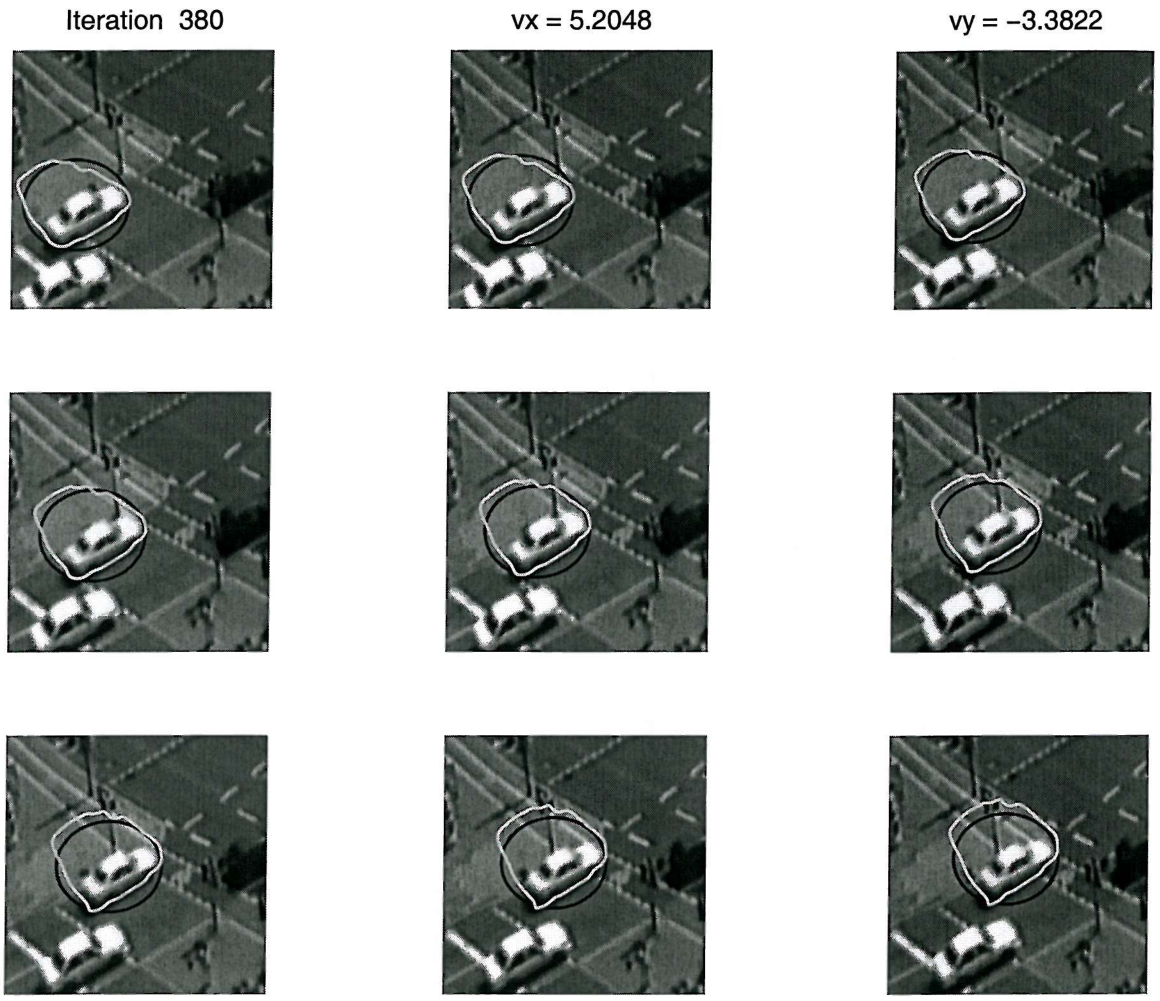


Figure 4.4: Patahlogical data capturing the contour

Chapter 5

Conclusions and Further Work

5.1 Conclusions

We have developed three global velocity aware snakes, the vGreedy, the vKass and the vXu, and a local velocity snake model, the lvSnake. The vXu snake has been chosen to test the efficacy of a velocity aware snake despite the poor ability to modify its velocity to locate object of unknown velocity.

The approach of the local velocity snake is to extend the inforamtion averaging process of the global velocity contour and to allow the snake to search for a target velocity, and this it has performed satisfactorally.

In the case of the global velocity snake, we have created a simple performance metric to measure the ability of a snake to converge on a target circle, in an automated test system. The results of testing have shown that there is indeed a positive reinforcement of data between frames in the case of a velocity aware snake and that this can overcome occlusion and structured backgrounds.

Although the occlusion results are not as impressive as those for the VHT, the constraint on the VHT is that it can only locate circles, whereas this technique can be used to locate any smooth continuous shape, as shown by the real world sequence, and with the addition of a local shape model, it is no longer constrained to a velocities close to the initial conditions.

This research has shown that the vXu snake does provide a solution to locating objects moving with constant velocity. The occlusion results are particularly encouraging as they suggest that this approach will work where the optical flow of a moving object breaks down. An application of this technique would lie in the extraction of moving objects from time-lapse video such as surveillance camera footage.

Appendix A

Deriving the Euler PDE

We wish to find a stationary solution of

$$I = \iint_{\bar{S}} \mathbf{F}(x, y, w, w_x, w_y, w_{xx}, w_{yy}, w_{xy})\, dx\, dy, \tag{A.1}$$

where subscripts denote partial derivatives.

Admissible functions of $w(x, y, z)$ must posses continuous fourth order derivatives and the functional $\mathbf{F}$ is assumed to posses continuous third order derivatives with respect to all arguments of $\mathbf{F}$ for all x and y in $\bar{S}$

Let $u(x, y)$ make I stationary and $\epsilon\eta(x, y)$ be a small perturbation from the solution at $u(x, y)$,

$$w(x, y) = u(x, y) + \epsilon\eta(x, y), \tag{A.2}$$

where,

$$\eta = 0 \quad \text{and} \quad \eta_x = \eta_y = \eta_{xx} = \eta_{yy} = \eta_{xy} = 0 \tag{A.3}$$

on the contour of $\bar{S}$, as

$$I(\epsilon)\big|_{\epsilon=0} = 0 \quad \text{and} \quad \left.\frac{dI}{d\epsilon}\right|_{\epsilon=0} = 0. \tag{A.4}$$

Equations A.1 and A.2 combine to give,

$$\iint_V \{\eta \mathbf{F}_u + \eta_x \mathbf{F}_{ux} + \eta_y \mathbf{F}_{uy} + \eta_{xx} \mathbf{F}_{uxx} + \eta_{yy} \mathbf{F}_{uyy} + \eta_{xy} \mathbf{F}_{ux} y\} \, dx \, dy = 0. \qquad \text{(A.5)}$$

Application of the divergence theorem or integration by parts to Equation A.5 yields the Euler P.D.E,

$$\mathbf{F}_u - \frac{\partial}{\partial x}\mathbf{F}_{ux} - \frac{\partial}{\partial y}\mathbf{F}_{uy} + \frac{\partial^2}{\partial x^2}\mathbf{F}_{uxx} + \frac{\partial^2}{\partial y^2}\mathbf{F}_{uyy} + \frac{\partial^2}{\partial xy}\mathbf{F}_{uxy} = 0. \qquad \text{(A.6)}$$

Bibliography

[1] A.A. Amini, T.E. Weymouth, and R.C. Jain. Using dynamic programming for solving variational problems in vision. *PAMI*, 12(9):855–867, September 1990.

[2] A. Blake and M. Isard. *Active Contours.* Springer-Verlag, 1998. ISBN 3-540-76217-5.

[3] Canny. Optimal edge detection in noise. *IEEE Transactions on Pattern Analysis and Machine Inteligence*, 1989.

[4] L. D. Cohen. Note:On Active Contour Models and Balloons. *CVGIP:Image Understanding*, 53(2):211–218, 1991.

[5] L. D. Cohen and I. Cohen. 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, 1993.

[6] G. H. Granlund. Fourier preprocessing for hand print character recognition. *IEEE Transactions on Computations*, (21):195–201, 1972.

[7] S.R. Gunn and M.S. Nixon. A robust snake implementation: a dual active contour. *IEEE Trans. PAMI*, 1997.

[8] M. Isard and A. Blake. CONDENSATION – conditional density propagation for visual tracking. *IJCV*, 29(1):5–28, 1998.

[9] A. K. Jain. *Fundamentals of Digital Image Processing.* Prentice Hall International, 1989. ISBN 0-13-332578-4.

[10] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active Contour Models. *International Journal of Computer Vision*, 1:312–331, 1988.

[11] C. Knoll, M. Alcañiz, V. Grau, C. Monserrat, and M. C. Juan. Outlining of the prostate using snakes with shape restrictions based on the wavelet transform (doctoral thesis: Dissertation). *Pattern Recognition*, (32):1767–1781, 1999.

[12] K. F. Lai and R. T. Chin. Deformable contours - modeling and extraction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(11):1084–1090, 1995.

[13] J. M. Nash, J. N. Carter, and M. S. Nixon. Dynamic Feature Extraction via the Velocity Hough Transform. *Pattern Recognition Letters*, 18:1035–1047, 1997.

[14] M. S. Nixon. *Feature Extraction in Image Processing and Computer Vision with Mathcad Implementation.* Southampton University Press, 1998. Alpha Version.

[15] N. Peterfreund. Velocity Snake. In *Proceedings of the IEEE Nonrigid and Articulated Motion Workshop*, pages 70–79, 1997.

[16] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C.* Cambridge University Press, 2nd edition, 1992. ISBN 0-521-43108-5.

[17] K.F. Riley, M.P. Hobson, and S.J. Bence. *Mathematical Methods for Physics and Engineering.* Cambridge University Press, 1997. ISBN 0-521-55529-9.

[18] D. Terzopoulos and R. Szeliski. Tracking with Kalman snakes. In A. Blake and A.L. Yuille, editors, *Active Vision*, chapter 1, pages 3–20. MIT Press, Cambridge, MA, 1992.

[19] J. B. Waite and W. J. Welsh. Head Boundary Location using Snakes. *Br. Telecom Technology Journal*, 8(3):127–136, 1990.

[20] D. J. Williams and M. Shah. A Fast Algorithm for Active Contours and Curvature Estimation. *CVGIP:Image Understanding*, 55(1):14–26, 1992.

[21] C. Xu and J. L. Prince. Snakes, shapes, and gradient vector flow. *IEEE Transactions on Image Processing*, 7(3):359–369, March 1998.

[22] G. Xu, E. Segawa, and S. Tsuji. Robust Active Contours with Insensitive Parameters. *Pattern Recognition*, 27(7):879–884, 1994.