An affine-invariant active contour model (AI-snake) for model-based segmentation

Horace H.S. Ip*, Dinggang Shen

Image Computing Group, Department of Computer Science, City University of Hong Kong, Tat Chee Avenue, Kowloon, Hong Kong

Abstract

In this paper, we show that existing shaped-based active contour models are not affine-invariant and we addressed the problem by presenting an affine-invariant snake model (AI-snake) such that its energy function are defined in terms local and global affine-invariant features. The main characteristic of the AI-snake is that, during the process of object extraction, the pose of the model contour is dynamically adjusted such that it is in alignment with the current snake contour by solving the snake-prototype correspondence problem and determining the required affine transformation. In addition, we formulate the correspondence matching between the snake and the object prototype as an error minimization process between two feature vectors which capture both local and global deformation information. We show that the technique is robust against object deformations and complex scenes. © 1998 Elsevier Science B.V.

Keywords: Active contour; Affine invariant; Correspondence matching; Curvature; Deformable model; Model-based; Object tracking; Snake

1. Introduction

Deformable templates [1–5] have been developed to detect and to locate objects which have been distorted due, for example, to changes in viewing geometry. As a deformable model, the active contour model (snake) has been successfully applied to a variety of problems such as the detection of edges and subjective contours [5], motion tracking [7] and segmentation of magnetic resonance images [6]. The general snake model uses smoothness constraints to restrict its solutions to the class of controlled continuity splines. However, when prior knowledge of shape is available for a specific application, information concerning the shape of the desired objects can be incorporated into the formulation of the snake model. In the following, we will briefly review current status of shape-based snake models and note that none of them are really affine invariant.

The active shape model [4] was built by learning patterns of variability from a training set of correctly annotated images, and used as a snake where the points' movement was controlled to satisfy some shape constraints. Amano et al. [8] used a sample image and a sample contour to define the internal energy terms of snake, so that the detected contour would be the same as or close to the sample contour

model. Similarly, Radeva and coworkers [9,10] formulated the internal energy of the snake by the model contour, trying to preserve the model shape. The generalized active contour model (g-snake) [11] incorporated a shape matrix as a prior model to create bias towards a particular type of contour. Although the shape matrix was proven to be affine transformation invariant, it should be stressed that the g-snake was not an affine invariant model. Wang and Wang [12] used a set of imaginary springs to enforce the consistency in the position, orientation and/or curvature measurements of the elastic grid and the desired shape.

As deformable models do not by themselves provide a method of computing canonical descriptions for recognition, or of establishing correspondence between sets of data [13], when applying shape-based snakes, two fundamental problems must be solved. Since we cannot guarantee that the initial poses between the object and prototype model are similar, before the snake model can be used to extract object boundary from a complex image, correspondence between the initial snake and the prototype (shape model) must be established. More importantly, while the snake is searching for, or tracking the object of interest, its shape will be repeatedly deformed in order to compromise itself between the image data and the model. Consequently, correspondence between the snake contour and the prototype (shape model) must be frequently re-established, and at the same time the prototype (shape model) should be

* Corresponding author. Tel.: 00 85 227888641; fax: 00 85 227888614.

automatically affine-transformed to best fit the currently deformed snake contour.

Given the above problems concerning the shape-based snake, the contributions of this paper are two-fold: (a) we present the development of a model-based active contour model (AI-snake) which is affine-invariant; and (b) we provide efficient solution for establishing correspondence between data and the shape-model based on affine-invariant local and global feature vectors defined for each point of the snake contour. We demonstrate that the approach is robust for highly cluttered scenes and significant deformations between data and model.

In Section 2, we first show that features based on areas formed by successive points (or snaxels) along a contour are affine invariant and then present a model-based affine invariant snake (AI-snake), such that the energy functionals of the AI-snake are defined in terms of these features and show that the resulting snake model is also affine invariant. An efficient algorithm for correspondence establishment between a snake contour and a shape prototype is presented in Section 3. In Section 4 we apply the AI-snake for model-based object segmentation and tracking for highly complex scenes.

2. An affine-invariant model-based active contour model (AI-snake)

The active contour model integrates model-driven and data-driven analysis through the deployment of an energy function and a set of regularization parameters. The energy function consists of two basic components: the data-driven component and the model-driven component. The former usually depends on image data, while the latter uses prior knowledge of the shape model of the desired objects. In the following, we will first develop an affine invariant feature which will form the basis of our affine-invariant snake model and then present an affine-invariant energy function. We provide an efficient solution to the correspondence problem in Section 3 since it is necessary for model-based segmentation using the AI-snake.

2.1. Affine invariant feature—area

The affine invariant feature here is developed by analysing the property of curvature. As a local representation, curvature is able to carry information at varying resolutions, and it has been applied to shape matching [14], shape analysis [15] and object recognition [16–18]. Curvature is invariant under rotation and translation of the shape, and can easily be normalized with respect to scale changes. However, curvature is not invariant under affine transformation. Accordingly, the original snake model, where the internal energy is defined by using the traditional definitions of curvature, cannot be an affine invariant model as shown in the following section.

2.1.1. Curvature is not an affine invariant feature

In the Cartesian coordinate system, we may represent a curve (or shape) C by the parametric equations $r(t) = [x(t), y(t)]$, where t is the parameter of this representation. The curvature $k(t)$ of the curve C is defined as

$$k(t) = \frac{\dot{x}(t)\ddot{y}(t) - \ddot{x}(t)\dot{y}(t)}{(\dot{x}(t)^2 + \dot{y}(t)^2)^{3/2}}$$

where

$$\dot{x}(t) = \frac{dx(t)}{dt}, \quad \ddot{x}(t) = \frac{d^2x(t)}{dt^2}, \quad \dot{y}(t) = \frac{dy(t)}{dt} \text{ and } \ddot{y}(t) = \frac{d^2y(t)}{dt^2}$$

Letting s represents the arc length of C, we then have

$$(ds)^2 = (dx)^2 + (dy)^2$$

In this case, the expression of curvature $k(t)$ can be rewritten as follows:

$$K(t) = \frac{\begin{vmatrix} \dot{x}(t) & \ddot{x}(t) \\ \dot{y}(t) & \ddot{y}(t) \end{vmatrix} \cdot (dt)^3}{(ds)^3} = \frac{ang}{dist} \tag{1}$$

where

$$ang = \begin{vmatrix} \dot{x}(t) & \ddot{x}(t) \\ \dot{y}(t) & \ddot{y}(t) \end{vmatrix} \cdot (dt)^3$$

and $dist = (ds)^3$. Later we will show that, under an affine transformation, the value of ang is linearly transformed while at the same time the value of $dist$ is non-linearly changed. That is, ang, when appropriately normalized is affine invariant, while $dist$ could not be affine invariant.

Let $C' = [u(\tau), v(\tau)]$ represents the affine-transformed version of the curve (or shape) C. Their relationship can be expressed by

$$\begin{bmatrix} u(\tau) \\ v(\tau) \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x(t) \\ y(t) \\ 1 \end{bmatrix} \tag{2}$$

From Eq. (2), we can obtain

$$ang = \begin{vmatrix} \dot{u}(\tau) & \ddot{u}(\tau) \\ \dot{v}(\tau) & \ddot{v}(\tau) \end{vmatrix} (d\tau)^3$$

$$= \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} \begin{vmatrix} \dot{x}(t) & \ddot{x}(t) \\ \dot{y}(t) & \ddot{y}(t) \end{vmatrix} (dt)^3 = (a_{11}a_{22} - a_{12}a_{21}) \cdot ang \tag{3}$$

where

$$\dot{u}(\tau) = \frac{du(\tau)}{d\tau}, \quad \ddot{u}(\tau) = \frac{d^2u(\tau)}{d\tau^2}, \quad \dot{v}(\tau) = \frac{dv(\tau)}{d\tau} \text{ and } \ddot{v}(t) = \frac{d^2v(\tau)}{d\tau^2}$$

This expression tells us that, if a curve $[x(t), y(t)]$ is affine transformed to become another curve $[u(\tau), v(\tau)]$, then the

value of *ang* is simply linearly increased by a factor of $(a_{11}a_{22} - a_{12}a_{21})$. That is, *ang* when appropriately normalized is affine invariant. A by-product of Eq. (3) is the concept of 'affine length', defined as $\sqrt[3]{\dot{x}(t)\ddot{y}(t) - \dot{y}(t)\ddot{x}(t)}\,\mathrm{d}t$, which has been used as a locally affine-invariant parameter in Refs. [19–22].

Similarly, from Eq. (2), we can obtain

$$\begin{bmatrix} \mathrm{d}u \\ \mathrm{d}v \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} \mathrm{d}x \\ \mathrm{d}y \end{bmatrix}$$

and further produce

$$dist = \left((\mathrm{d}u)^2 + (\mathrm{d}v)^2\right)^{3/2}$$

$$= \left((a_{11}\mathrm{d}x + a_{12}\mathrm{d}y)^2 + (a_{21}\mathrm{d}x + a_{22}\mathrm{d}y)^2\right)^{3/2}$$

This expression implies that the value of *dist* is non-linearly changed under affine transformation, and therefore *dist* could not be affine invariant except when the scaling parameters on both dimensions are identical. This observation also leads us to conclude that the g-snake [11] is not affine-invariant.

The above analysis clearly indicates that the curvature of a curve and the Euclidean distance are both not affine invariant, while *ang* when appropriately normalized is an affine invariant quantity. This leads us to adopt *ang* as the basis for formulating new energy functionals of our AI-snake in Section 2.3 and Section 2.4, and for defining an affine-invariant feature vector for each feature point localized on the object boundary for solving the model-data correspondence problem in Section 3.

2.1.2. Area obtained from curvature is an affine invariant feature

In the following, we will show that the expression $(1/2)ang$ is in fact the area of a triangle defined by three vertices of the digital curve C. This expression will be used later as the basis of the energy function of our AI-snake.

The first and second derivatives of $x(t)$ and $y(t)$ with respect to t can be approximately written as

$$\dot{x}(t) = \frac{\mathrm{d}x(t)}{\mathrm{d}t} \doteq \frac{x(t) - x(t - \Delta t)}{\Delta t},$$

$$\ddot{x}(t) = \frac{\mathrm{d}^2 x(t)}{\mathrm{d}t^2} \doteq \frac{x(t) - 2\cdot x(t - \Delta t) + x(t - 2\cdot\Delta t)}{\Delta t\cdot\Delta t},$$

$$\dot{y}(t) = \frac{\mathrm{d}y(t)}{\mathrm{d}t} \doteq \frac{y(t) - y(t - \Delta t)}{\Delta t},$$

$$\ddot{y}(t) = \frac{\mathrm{d}^2 y(t)}{\mathrm{d}t^2} \doteq \frac{y(t) - 2\cdot y(t - \Delta t) + y(t - 2\cdot\Delta t)}{\Delta t\cdot\Delta t}$$

Using the above approximations, we can obtain

$$\frac{1}{2}ang \doteq \frac{1}{2}\begin{vmatrix} x(t - 2\cdot\Delta t) & x(t - \Delta t) & x(t) \\ x(t - 2\cdot\Delta t) & y(t - \Delta t) & y(t) \\ 1 & 1 & 1 \end{vmatrix}$$

This expression indicates that the value of $(1/2)ang$ represents the area of a triangle, formed by three vertices:

$$\left(x(t - 2\cdot\Delta t), y(t - 2\cdot\Delta t)\right),$$

$$\left(x(t - \Delta t), y(t - \Delta t)\right) \text{ and } \left(x(t), y(t)\right).$$

In fact, the area of any triangle can be made affine invariant. Let $area_{t_1,t_2,t_3}$ be the area of a triangle, whose three vertices are $\left(x(t_1), y(t_1)\right)$, $\left(x(t_2), y(t_2)\right)$ and $\left(x(t_3), y(t_3)\right)$. That is

$$area_{t_1,t_2,t_3} = \frac{1}{2}\begin{vmatrix} x(t_1) & x(t_2) & x(t_3) \\ y(t_1) & y(t_2) & y(t_3) \\ 1 & 1 & 1 \end{vmatrix}$$

If we use Eq. (2) to express the relationship of the affine transformation between two curves C and C', the areas of the same triangle before and after any affine transformation are related as follows:

$$area_{t_1,t_2,t_3}^{affine} = \frac{1}{2}\begin{vmatrix} u(\tau_1) & u(\tau_2) & u(\tau_3) \\ v(\tau_1) & v(\tau_2) & v(\tau_3) \\ 1 & 1 & 1 \end{vmatrix}$$

$$= \frac{1}{2}\begin{vmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ 0 & 0 & 1 \end{vmatrix}\begin{vmatrix} x(t_1) & x(t_2) & x(t_3) \\ y(t_1) & y(t_2) & y(t_3) \\ 1 & 1 & 1 \end{vmatrix}$$

$$= \left(a_{11}a_{22} - a_{12}a_{21}\right)\cdot area_{t_1,t_2,t_3}$$

The above expression indicates that the area of any triangle when appropriately normalized is affine invariant.

2.2. Formulation of AI-snake energy functionals

A snake C_{snake} is defined as an ordered set of points, commonly called snaxels, $\{V_i | i = 1, 2, ..., N\}$. Each V_i defines the corresponding position (u_i, v_i) in the image plane. The total energy of a snake is the weighted summation of several particular energy terms. The number of the particular terms is dependent on the application. For our purpose, the energy function of the snake is defined as follows:

$$E_{\text{snake}} = E_{\text{smooth}} + E_{\text{model}} + E_{\text{data}}$$

where

$$E_{\text{smooth}} = \sum_{i=1}^{N} e_{\text{smooth}}(V_i), \quad E_{\text{model}} = \sum_{i=1}^{N} e_{\text{model}}(V_i) \text{ and}$$

$$E_{\text{data}} = \sum_{i=1}^{N} e_{\text{data}}(V_i)$$

The energy of the snake, E_{snake}, consists of three energy terms: the internal energy term E_{smooth}, the model-based

energy term E_{model} and the external energy term E_{data}. The internal energy term E_{smooth} enforces smoothness along the snake, while the constraint energy term E_{model}, encodes the prior shape model of the desired objects, and hence constraining the snake to be of similar shape to that of the prior model. The external energy term E_{data}, usually defined in terms of image intensity gradients, forces the snake to move towards image features.

Since the energy minimization process of the snake model is usually carried out iteratively, an initialization for the snake is required. Basically, the initial snake should be near to the desired boundary, otherwise the snake may be trapped by local energy minima and give an undesired solution. The method of initializing the snake depends on the applications. For some applications, the original snake has to be interactively initiated by user, while for other special applications the initialization process can be completed automatically [11,23,24].

Once a snake model has been defined and its initial snake has been given, the process of extracting object from the complex image becomes a process of minimizing the snake energy. In the literature, many different approaches for minimizing the snake energy have been reported, e.g. finite difference method [5], finite element method [25], dynamic programming [26], greedy algorithm [27] and neural network [28]. These approaches all have a trade-off between the quality of the solution and the computational complexity.

2.3. Internal energy functional—an affine-invariant smoothness constraint

In the original snake model, proposed in Ref. [5], the internal energy of the snake is composed of two energy terms: the continuity term and the curvature term. The continuity term is defined on the first derivative of the snake for controlling tension of the snake, while the curvature term is defined on the second derivatives of the snake for controlling the smoothness along the snake. Since the original energy formulation has the undesirable property that the snake has a tendency to shrink [5] or expand [27], several methods [29,30] have been proposed to overcome this problem. Although some of the energy formulations are scale and rotational invariant, none of the formulations reported

to date are affine invariant. This is beacuse the original definition of the curvature is not affine invariant as shown in Section 2.1.

In addition, as indicated in Section 2.1, the area of a triangle, when appropriately normalized, is affine invariant and here we define the curvature of a point on the curve in terms of the area formed by the point itself and two other points lying on the same curve.

Let the area of the ith triangle $area_i$ be defined by three consecutive points, V_{i-1}, V_i, and V_{i+1} along a curve as shown in Fig. 1(a). The necessary and sufficient condition for the area of the ith triangle to be zero, i.e. $area_i = 0$, is when the three points are co-linear. There are three possible situations for $area_i = 0$, which are shown in Fig. 1(b) and (c). For the two situations shown in Fig. 1(c), the point V_i is not located in between V_{i-1} and V_{i+1}.

However, the effects of the other energy terms and the searching direction for each snaxel, as shown in Fig. 1(a), which is made to move towards the domain between V_{i-1} and V_{i+1}, prevent the two possibilities in Fig. 1(c) from occurring.

Based on this idea, the internal energy term for enforcing smoothness of the snake is defined as

$$E_{smooth} = \sum_{i=1}^{N} e_{smooth}(V_i)$$

$$e_{smooth}(V_i) = \alpha \frac{|area_i|}{AREA_{now}} \cdot \frac{AREA_{mdl}}{\frac{1}{2}d^2}$$

where

$$area_i = \frac{1}{2}\begin{vmatrix} u_{i-1} & u_i & u_{i+1} \\ v_{i-1} & v_i & v_{i+1} \\ 1 & 1 & 1 \end{vmatrix}$$

$AREA_{now}$ is the total area of the snake in the present iteration. It can be calculated by using

$$AREA_{now} = \frac{1}{2}\sum_{i=1}^{N}\left\| \begin{matrix} u_i & u_{i+1} & u_{cent} \\ v_i & v_{i+1} & v_{cent} \\ 1 & 1 & 1 \end{matrix} \right\|$$

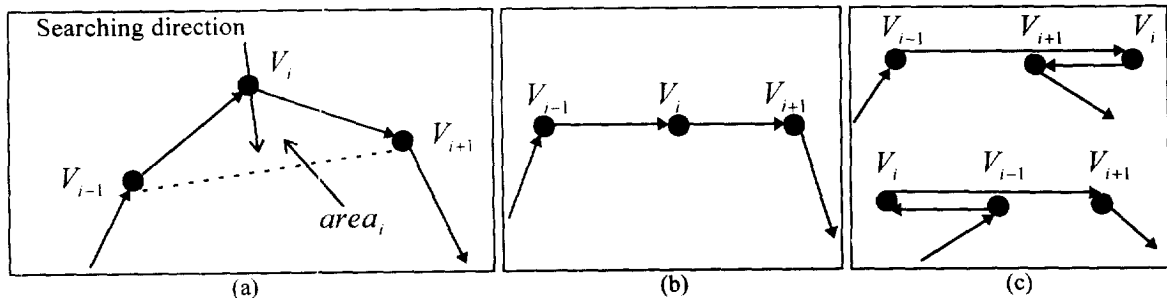where the point (u_{cent}, v_{cent}) is the centroid of the current
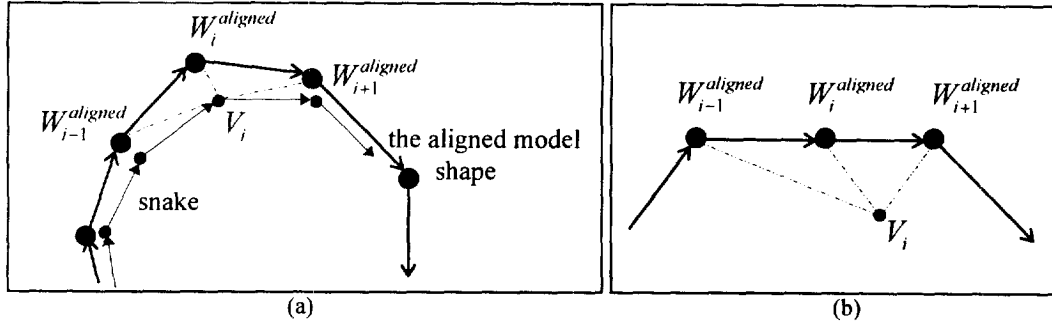

Fig. 1. Notations used for formulating E_{smooth}.

Fig. 2. The relationship between the snake and the aligned prototype.

snake, that is,

$$\begin{bmatrix} u_{cent} \\ v_{cent} \end{bmatrix} = \frac{1}{N} \sum_{i=1}^{N} \begin{bmatrix} u_i \\ v_i \end{bmatrix}$$

$AREA_{mdl}$ is the total area of the prototype (shape model), and d is a constant representing the average inter-snaxel distance measure of the prototype. It follows from the conclusion drawn in Section 2.1 that E_{smooth} is invariant to affine transformation.

2.4. An affine invariant energy term for specifying the prior model

The prototype $C_{prototype}$ is defined as an ordered set of points, $\{W_i | i = 1, 2, ..., N\}$, and $W_i = (x_i, y_i)$. The snake C_{snake} can be seen as the deformed and affine transformed version of the prototype $C_{prototype}$. Based on the method to be described in Section 3, we can determine the correspondence between the prototype $C_{prototype}$ and the snake C_{snake}. To simplify the description of the analysis, we assume for the moment that the (labelling) order number of the points on the snake C_{snake} and that on the prototype $C_{prototype}$ are the same. That is, the point on the snake C_{snake} which corresponds to the ith point $W_i = (x_i, y_i)$ on the prototype $C_{prototype}$ has been defined as the ith snaxel point, $V_i = (u_i, v_i)$. Let A be a 3×3 affine transform matrix that relates the prototype and the snake, the evolution of the snake can be guided by the aligned prototype, $C_{aligned}$. The set of points on the aligned prototype, $C_{aligned}$, $\{W_i^{aligned} = (x_i^{aligned}, y_i^{aligned}) | I \neq i = 1, 2, ..., N\}$, can be obtained by

$$\begin{bmatrix} x_i^{aligned} \\ y_i^{aligned} \\ 1 \end{bmatrix} = A \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

such that the point $W_i^{aligned} = (x_i^{aligned}, y_i^{aligned})$ corresponds to the point $V_i = (u_i, v_i)$. That is, the point correspondence is kept fixed. To make the shape of the snake similar to the prototype, we expect that the relative distance between each point on the snake and its corresponding point on the aligned prototype $C_{aligned}$ is small. Typically, the model-based energy functional is often defined by using the Euclidean distances, which will make the model-based energy term

non-linearly changed under the affine transformation. Since the area of a triangle, when appropriately normalized, can be made affine invariant, we define the model-based energy functional using area-based features. From Fig. 2(a), we know that the process of making V_i close to $W_i^{aligned}$ is equivalent to that of making the total areas of the triangle $\Delta_{W_{i-1}^{aligned} V_i W_i^{aligned}}$ and the triangle $\Delta_{W_i^{aligned} V_i W_{i+1}^{aligned}}$ close to zero. The energy functional E_{model} can therefore be defined as follows:

$$E_{model} = \sum_{i=1}^{N} e_{model}(V_i),$$

such that

$$e_{model}(V_i) = \beta \frac{\left| S_{W_{i-1}^{aligned} V_i W_i^{aligned}} \right| + \left| S_{W_i^{aligned} V_i W_{i+1}^{aligned}} \right|}{AREA_{now}} \frac{AREA_{mdl}}{d^2}$$

where $\left| S_{W_{i-1}^{aligned} V_i W_i^{aligned}} \right|$ and $\left| S_{W_i^{aligned} V_i W_{i+1}^{aligned}} \right|$ are the areas of the triangle $\Delta_{W_{i-1}^{aligned} V_i W_i^{aligned}}$ and the triangle $\Delta_{W_i^{aligned} V_i W_{i+1}^{aligned}}$, respectively. The definitions of $AREA_{now}$, $AREA_{mdl}$ and d have been given in Section 2.3. If the summation of $\left| S_{W_{i-1}^{aligned} V_i W_i^{aligned}} \right|$ and $\left| S_{W_i^{aligned} V_i W_{i+1}^{aligned}} \right|$ is zero, then the ith point V_i of the snake is pushed onto the point $W_i^{aligned}$. In particular, if the three points $W_{i-1}^{aligned}$, $W_i^{aligned}$ and $W_{i+1}^{aligned}$ are co-linear, then the sum of $\left| S_{W_{i-1}^{aligned} V_i W_i^{aligned}} \right|$ and $\left| S_{W_i^{aligned} V_i W_{i+1}^{aligned}} \right|$ will be zero only if the point V_i is also located on the same line [Fig. 2(b)]. Even for such cases, our definition of the model-based energy functional remains valid, since our goal is to align the points of the snake onto the prototype.

2.5. The external energy

An energy term based on the magnitude of the image gradient [5] alone does not provide enough information to guide the snake to move towards the desired object. This is particularly true when the image contains much confusing edges. To overcome this limitation, the directional information of the gradient has also been considered [23,29–32]. For each snaxel, the external energy is defined as the dot product of the normal vector of the snake with the direction of the gradient, weighted by the magnitude of the corresponding gradient. Similarly, our external energy based on

the image data constraint is defined as

$$E_{data} = \sum_{i=1}^{N} e_{data}(V_i)$$

such that

$$e_{data}(v_i) = \gamma\left(1 - |\nabla I(v_i)| \cdot |\mathbf{n}(V_i) \cdot \mathbf{h}(V_i)|\right)$$

where $|\nabla I(V_i)|$ is the magnitude of the gradient on snaxel V_i, and $\mathbf{h}(V_i)$ is the direction of the gradient. $\mathbf{n}(V_i)$ is the normal vector of the snake in a given snaxel V_i, directed towards the snake interior.

$$\mathbf{n}(V_i) = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

$$\times \left(\frac{V_i - V_{i-1}}{\|V_i - V_{i-1}\|} + \frac{V_{i+1} - V_i}{\|V_{i+1} - V_i\|} \right) \Bigg/ \Bigg\| \frac{V_i - V_{i-1}}{\|V_i - V_{i-1}\|}$$

$$+ \frac{V_{i+1} - V_i}{\|V_{i+1} - V_i\|} \Bigg\|$$

In summary, the final energy form of our AI-snake model can be defined as

$$E_{snake} = \alpha \sum_{i=1}^{N} \frac{|area_i|}{AREA_{now}} \cdot \frac{AREA_{mdl}}{\frac{1}{2}d^2}$$

$$+ \beta \sum_{i=1}^{N} \frac{|S_{W_{i-1}^{aligned} V_i W_i^{aligned}}| + |S_{W_i^{aligned} V_i W_{i+1}^{aligned}}|}{AREA_{now}}$$

$$\cdot \frac{AREA_{mdl}}{d^2} + \gamma \sum_{i=1}^{N}\left(1 - |\nabla I(V_i)| \cdot |\mathbf{n}(V_i) \cdot \mathbf{h}(V_i)|\right)$$

where α, β and γ are the set of regularization parameters.

3. Solving the correspondence problem between the snake and the prototype

The most common approach of determining the correspondence between sets of data is to find distinctive local features. Local features, such as geometric invariants [33], corners [34] and curvature [35], are less sensitive to occlusion. However, local features are sensitive to noise, unless a multi-scale framework is suggested [17,18]. Usually, the amount of local information is insufficient for robust matching. Furthermore, from Section 2.1 it is clear that curvature is not suitable as a local feature for detecting correspondence between two shapes under affine transformation. As a new method, modal matching [13] described objects in terms of generalized symmetries, which were defined by each object's eigenmodes. Since modes provide a global-to-local ordering of deformation, this approach allowed for selecting the appropriate types of deformations that were used in object alignment and comparison. However, the technique is computationally expensive since it requires the computation of eigenmodes and their interpolation.

Here, we present a much more efficient process of establishing correspondence between two shapes and develop the equation for determining the affine transformation relating the two shapes.

3.1. Feature vectors for point correspondence

3.1.1. Formulation

Based on Section 2.1, we here design a feature vector for every feature point of the given contour. Let $\{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$ be the set of feature points representing the given contour C. These feature points can be extracted directly from the sample points, which are evenly distributed along the given shape. To make feature points robust to noise, we take the same number of points from both sides of a sample point and average them to obtain a feature point. For the sake of discussion, N is assumed to be a odd number. For the ith feature point (x_i, y_i), its corresponding feature vector is defined as:

$$F_i = [f_1 f_{2i} \ldots f_{Mi}]^T, \text{ and } f_{ji} = \frac{1}{2}\begin{vmatrix} x_{(i-j)} & x_i & x_{(i+j)} \\ y_{(i-j)} & y_i & y_{(i+j)} \\ 1 & 1 & 1 \end{vmatrix}$$

where

$$M = \frac{N-1}{2}$$

Notice that $x_i = x_{(i+2N)\%N}$ and $y_i = y_{(i+2N)\%N}$, especially for $i, j < 0$ or $i, j > N$. This transformation is suitable for other variables such as these ones. f_{ji} is the area of a triangle defined by $(x_{(i-j)}, y_{(i-j)})$, (x_i, y_i) and $(x_{(i+j)}, y_{(i+j)})$. It can be seen that f_{ji} expresses the local feature of the contour as the value of j is small, while f_{ji} represents the global feature of the contour as the value of j becomes large. However, when the value of j is close to M, f_{ji} begins to express local features again. In summary, the elements in the feature vector F_i can be broken into three groups: local features, global features and local features. That is,

$$F_i = \Bigg[\underset{1i}{f} \ldots f_{ki}(\text{local features}) \quad \underset{(k+1)i}{f} \ldots f_{li}(\text{global features})$$

$$\times \underset{(l+1)i}{f} \ldots f_{Mi}(\text{local features}) \Bigg]^T$$

Since the affine transformation will make the feature vectors change from F_i to $(a_{11}a_{22} - a_{12}a_{21})F_i$, $i = 1, 2, \ldots, N$, these feature vectors have to be normalized before they are used to establish the correspondence between two shapes. We suggest using the following normalization:

$$\hat{F}_i = \frac{F_i}{\sum_{i=1}^{N}\sum_{j=1}^{M}|f_{ji}|}$$

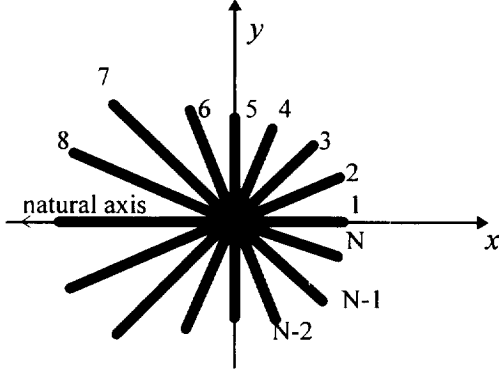In this way, point correspondence between two shapes can

Fig. 3. 'Sun-liked object'.

be conveniently determined. Assume that the curve (shape) C' represents the curve (shape) C under an affine transformation, and that two feature sets containing the affine-invariant feature vectors for the curve C and the curve C' are $\{\hat{F}_i, \ i = 1, 2, ..., N\}$ and $\{\hat{F}'_i, \ i = 1, 2, ..., N\}$, respectively. In this case, the task of determining point correspondence between two curves becomes one of searching for an optimal match of the feature vectors between two feature sets. If $(i + \omega)$th feature point of the curve C' corresponds to the ith feature point of the curve C, then we expect the feature vector $\hat{F}'_{(i+\omega)}$ be identical to the feature vector $\hat{F}_i$. That is

$$E_{i,\omega} = \left(\hat{F}'_{(i+\omega)} - \hat{F}_i\right)^T \cdot \left(\hat{F}'_{(i+\omega)} - \hat{F}_i\right) = 0$$

where ω is a positive integer. Unfortunately, since there always exists noise and deformation on the given shape, it is usually not possible for $E_{i,\omega}$ to be zero. Practically, we seek to minimize the total matching error E_ω. That is

$$E = \min_{\omega \in \{0,1,2,...,N-1\}} E_\omega,$$

such that

$$E_\omega = \sum_{i=1}^N E_{i,\omega} = \sum_{i=1}^N \left(\hat{F}'_{(i+\omega)} - \hat{F}_i\right)^T \cdot \left(\hat{F}'_{(i+\omega)} - \hat{F}_i\right)$$

In general, since the number of feature points for the given curve is not too great, it is computationally feasible to calculate exhaustively all the E, $\omega = 0, 1, 2, ..., N - 1$, to obtain the optimal solution.

3.1.2. An accelerated algorithm

However, as stated in Section 1, when applying deformable model to object segmentation in an iterative process, it is necessary to repeatedly establish correspondence between the current snake and the prototype. Hence it is extremely useful and also necessary to formulate an even more computationally efficient method to determine correspondence.

For the feature set $\{\hat{F}_i, i = 1, 2, ..., N\}$, we design a corresponding 'sun-liked object' (Fig. 3). This 'sun-liked object' is composed of N radiating lines. The length and direction of ith radiating line are, respectively, defined as

$$length_i = \sqrt{\hat{F}_i^T \cdot \hat{F}_i} \text{ and } \theta_i = \frac{2\pi}{N}(i - 1) \text{ for } i = 1, 2, ..., N$$

Then the natural axis of the 'sun-liked object' can be defined as a vector, whose starting point is the origin O and ending point is the object's weighting centre. The weighting centre (X_{centre}, Y_{centre}) is calculated by

$$X_{centre} = \sum_{i=1}^N length_i \cdot \cos(\theta_i), \quad Y_{centre} = \sum_{i=1}^N length_i \cdot \sin(\theta_i)$$

Once the natural axis of the 'sun-liked object' is detected, the feature vector $\hat{F}_a$ will be regarded as the starting feature vector. Here a is a positive integer, defined as

$$a = \left[\frac{\phi}{2\pi} \times N\right] + 1$$

where ϕ is the counterclockwise angle between the x-axis and the natural axis of the 'sun-liked object'.

Similarly, the starting feature vector of curve C' can be detected. Let $\hat{F}_a$ be the starting feature vector for the curve C and $\hat{F}'_b$ be the starting feature vector for the curve C'. Then, we consider that the ath feature point on the curve C corresponds to the bth feature point on the curve C', and in the absence of noise $E_{(b-a)}$ is the minimal value among the set of $\{E_0, E_1, ..., E_{N-1}\}$.

In practice, the directional angle of the natural axis may be slightly affected by noise or there may not be an unique natural axis for some objects. This way, we can determine the optimal match by local searching from an ordered sequence of $E_\omega, \omega = 0, 1, 2, ..., N - 1$, i.e. the set of $E_{(b-a+j)}, j = 0, \pm 1, \pm 2, ..., \pm \frac{N-1}{2}$. Once the current
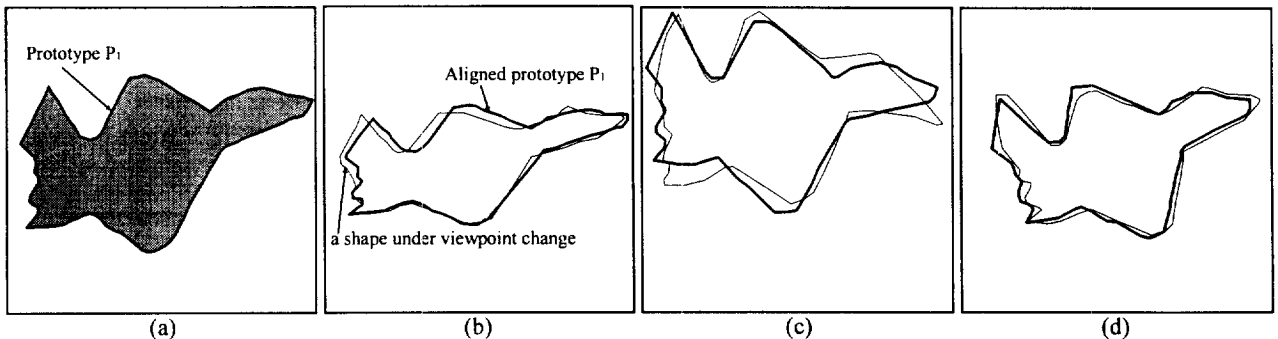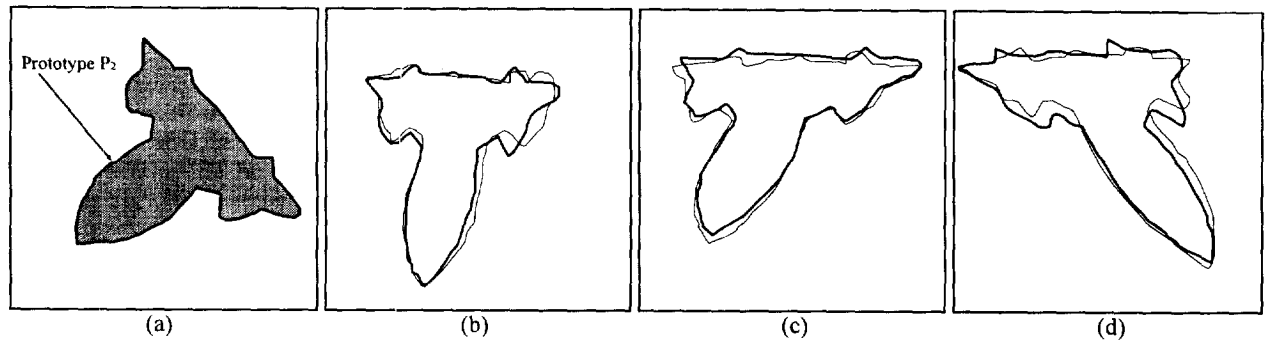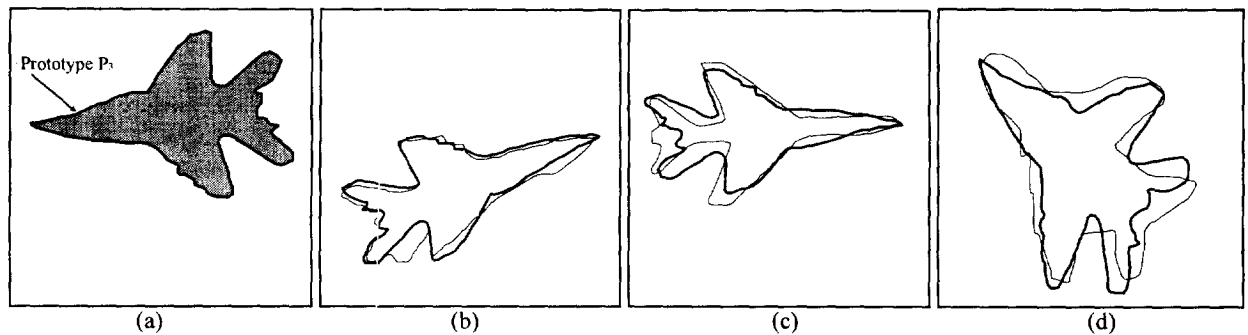

Fig. 4. The prototype airplane P_1, aligned with three different views of itself. The curve in (a) represents the prototype of airplane P_1. In (b)-(d), the thick curves are the aligned prototypes, while the thin curves are the shapes under different views.

Fig. 5. The prototype airplane P_2, aligned with three different views of itself.

Fig. 6. The prototype airplane P_3, aligned with three very different views of itself.

matching error $E_{(b-a+j_0)}$ is less than the pre-defined permitted error, the searching process can be terminated and the point correspondence can then be established by regarding the ath feature point on the curve C as the corresponding point of the $(b+j_0)$th feature point on the curve C'.

For the objects tested in our experiments, the correspondence between two curves is usually established by calculating only a small number of E_ω. For examples, the number of feature points used in Figs 4–7 is 51, that is, $N = 51$. Less than five matching errors (E_ω) are calculated, respectively, for Figs 4–6, while less than 11 matching errors (E_ω) are calculated for Fig. 7.

3.2. Determining the affine transform

Once the correspondence between two curves has been established, it is straightforward to determine the affine transform which relates these two curves. Without loss of generality, assumed that, the $(i+j_0)$th feature point (u_{i+j_0}, v_{i+j_0}) of

the curve C' corresponds to the ith feature point (x_i, y_i) of the curve C. Our objective is to obtain the affine transform matrix A making the error equation ε_A minimal.

$$\varepsilon_A = \|Q' - A \cdot Q\|^2 \text{ where } A = \begin{bmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ 0 & 0 & 1 \end{bmatrix},$$

$$Q = \begin{bmatrix} x_1 & x_2 & \cdots & \cdots & x_N \\ y_1 & y_2 & \cdots & \cdots & y_N \\ 1 & 1 & \cdots & \cdots & 1 \end{bmatrix} \text{ and }$$

$$Q' = \begin{bmatrix} u_{1+j_0} & u_{2+j_0} & \cdots & \cdots & u_{N+j_0} \\ v_{1+j_0} & v_{2+j_0} & \cdots & \cdots & v_{N+j_0} \\ 1 & 1 & \cdots & \cdots & 1 \end{bmatrix}$$
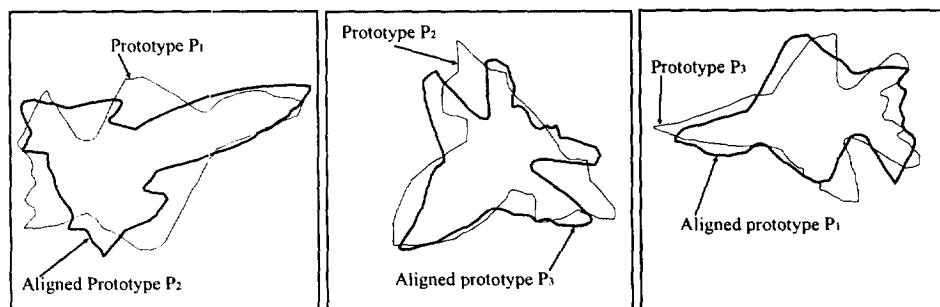

Fig. 7. Three different types of airplane images aligned with each other.

The solution is $A = (Q' \cdot Q^T)(Q \cdot Q^T)^{-1}$. Since $Q \cdot Q_T$ is a 3×3 matrix, it is easy and fast to obtain its inverse matrix. Hence, the calculation of the affine transform can be completed quickly.

4. Experimental results

We present two independent experiments to demonstrate the efficiency of the proposed methods. One set of experiments is focused on the detection of the correspondence and the affine transformation between two shapes. We do this by first matching an identical object under different views, then showing the robustness of the approach by matching shapes from different object prototypes. The other set of the experiments demonstrates the performance of our AI-snake. We do this be applying the AI-snake to segment and track objects from complex scenes.

4.1. Experiments on view-prototype alignment based on correspondence matching

In our experiments, three types of airplane images are used. They are airplanes P_1, P_2 and P_3, shown in Figs 4–6. The number of the feature points is 51, that is $N = 51$. Figs 4–6 demonstrate the ability of our method in determining affine transform that relates with two shapes from the same class. Notice that shapes in Fig. 4(a), Fig. 5(a) and Fig. 6(a) are three prototype airplanes. Additionally, Fig. 7 gives a demonstration of the robustness of our technique

in determining affine relationship of two very different shapes.

Thr shape in Fig. 4(a) is the prototype airplane P_1. In Fig. 4(b)–(d), thin curves represent the different views of the prototype P_1, while thick curves are the aligned prototype airplanes. The meanings of curves in Fig. 5 are similar to those in Fig. 4. The views shown in Fig. 6 are drastically different as they are obtained from very different viewing angles. However, the matching results are also satisfactory. Three different prototypes in Fig. 4(a), Fig. 5(a) and Fig. 6(a) are aligned to each other. The results are shown in Fig. 7. It is easy to see that the proposed technique is robust even for such situations.

4.2. Experiments on the AI-snake model

Here, we evaluate the performance of the AI-snake model. Experiments have been carried out on two types of grey-level images, a still image and a video sequence. Fig. 8 shows a still image with a plane located among complex boundaries. Fig. 9 shows the frame images containing a moving tennis racket among highly complex and cluttered boundaries. For the still image, the initial snake is initialized interactively, while for the frame images, only one initial snake for the first frame image is input. For both cases (i.e. airplane segmentation and racket tracking), the shape model guides the AI-snake to extract the desired object, while at the same time the intermediate snake contours influence the shape model by causing it to deform via affine transformations and aligning itself with the snake contours. In other
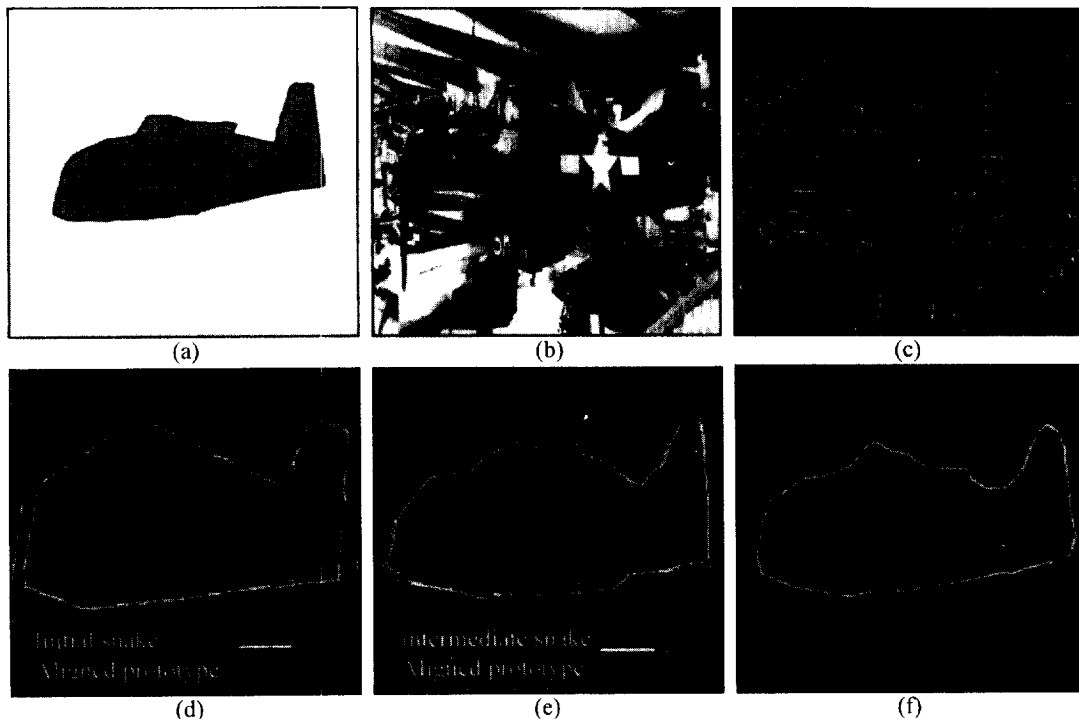

Fig. 8. Extraction of an airplane against highly cluttered background: (a) prototype airplane; (b) original image; (c) edge magnitude; (d) initial snake and the aligned prototype; (e) intermediate step with the snake contour and the presently aligned prototype; (f) final boundary.
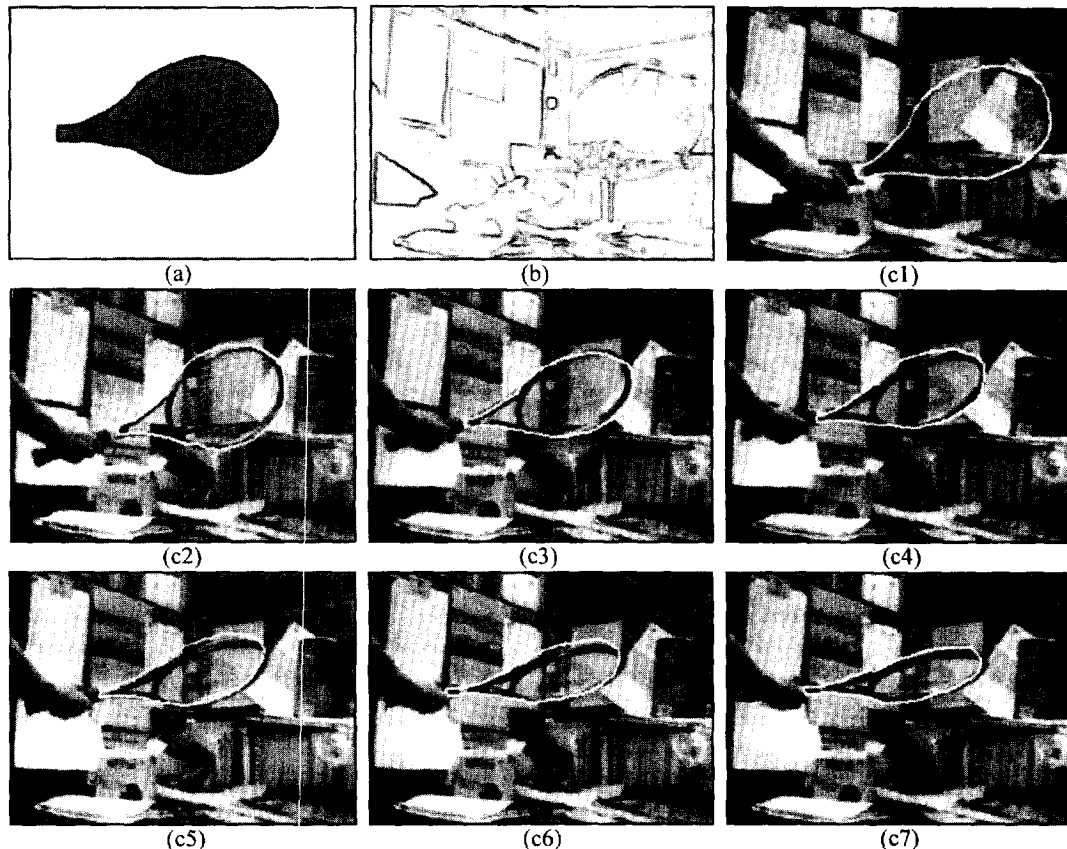
Fig. 9. Tracing the moving and deforming tennis racket among a complex scene sequence: (a) prototype tennis racket; (b) typical edge map of the frame images; (c1)–(c7) tracking results for the deformed tennis rackets.

words, there exists interactions between the shape model and the snake contour throughout the object extraction process.

Fig. 8 demonstrates the feasibility and the robustness of the model-based AI-snake in extracting the desired objects from complex scenes. The shape model of the airplane, taken from one standard image, is given in Fig. 8(a). We expect to use this shape model to guide the process of the object extraction from the grey-level image shown in Fig. 8(b). The edge image of Fig. 8(b) is given in Fig. 8(c). The initial snake, shown in Fig. 8(d), is input by a user. In order to see the snake contour clearly, we have linearly compressed the total grey-level of the original image from 255 to 128.

Once the snake is initialized, the prototype airplane aligned itself with the current snake. The aligned prototype is shown as a thin curve in Fig. 8(d). Notice that the aligned prototype fits the snake very well. Subsequently, the snake begins to move on the image, guided by the aligned prototype via the usual snake energy minimization process. We use the greedy algorithm for the energy minimization. Fig. 8(e) shows an intermediate step during the evolution of the snake, where the thick curve represents the snake while the thin curve represents the presently aligned prototype. After 25 iterations, the snake becomes stable

and the segmentation is completed. The result of the object extraction is shown in Fig. 8(f). Although the input image contains significantly cluttered background and confusing edges, the outline of the airplane has been extracted correctly.

Fig. 9 shows a demonstration of tracking a deforming tennis racket from the complex scenes. We evenly samples 14 frame images from a video sequence. A typical edge map of these frame images is given in Fig. 9(b). One can easily appreciate from these images that the images are rich in textures and the problem of tracking the desired tennis racket is a difficult one. The shape model (prototype) is depicted in Fig. 9(a). We initialize the snake for the first frame only. For the subsequent frame images the tracking result of the previous frame is used as the initial snake of the current frame. Once the initial snake is known for a frame image, the process of object extraction is the same as the process described for Fig. 8. We show here only half of the frame images (seven frame images) with the tracking results in Fig. 9(c1)–(c7). These seven frame images are the odd frame images of the 14 frame images processed. Although there exists significant deformations between the two tennis rackets seen in neighbouring frame images, the proposed method succeeds in correctly tracking the desired object through the complex scene sequence.

5. Conclusion

In this paper, we have formulated an affine invariant snake model (AI-snake) for model-based segmentation and presented an efficient method of establishing the correspondence between the model and the data.

In particular, we show that existing shaped-based active contour models are not affine-invariant and we addressed the problem by developing local and global affine-invariant features which form the basis of our AI-snake energy functionals. In the process of object extraction using the AI-snake, the correspondence between the prototype and the snake contour is first established, the prototype is then aligned with the current snake contour based on the estimated affine transformation. Subsequently, the snake movement is guided by the aligned prototype through the usual snake energy minimization process.

The key issue of correspondence matching between the snake and the object prototype in model-based segmentation has been formulated as an error minimization process between two feature vectors which capture both local and global deformation information. Since the local features describe the local deformations of the object and the global features describe the global deformations of the object, the proposed method is very robust to object deformation and disturbing noise.

Experiments which demonstrated the proposed method of correspondence matching, object extraction and tracking using our AI-snake have shown the efficiency and the effectiveness of the techniques.

References

[1] A.K. Jain, Y. Zhong, S. Lakshmnan, Object matching using deformable templates, IEEE Transactions on Pattern Analysis and Machine Intelligence 18 (3) (1996) 267–278.

[2] L.H. Staib, J.S. Duncan, Boundary finding with parametrically deformable models, IEEE Transactions on Pattern Analysis and Machine Intelligence 14 (11) (1992) 1061–1075.

[3] G. Storvik, A Bayesian approach to dynamic contours through stochastic sampling and simulated annealing, IEEE Transactions on Pattern Analysis and Machine Intelligence 16 (10) (1994) 976–986.

[4] T.F. Cootes, C.J. Taylor, D.H. Cooper, J. Graham, Active shape models—their training and application, Computer Vision and Image Understanding 61 (1) (1995) 38–59.

[5] M. Kass, A. Witkin, D. Terzopoulos, Snakes: active contour models, International Journal of Computer Vision 1 (1988) 321–331.

[6] L.D. Cohen, 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) (1993) 1131–1147.

[7] F. Leymarie, M.D. Levine, Tracing deformable objects in the plane using an active contour model, IEEE Transactions on Pattern Analysis and Machine Intelligence 15 (6) (1993) 617–634.

[8] A. Amano, Y. Sakaguchi, M. Minoh, K. Ikeda, Snake using a sample contour model, ACCV'93 Asian Conference on Computer Vision, Osaka, Japan, 1993, pp. 538–541.

[9] P. Radeva, E. Marti, An improved model of snakes for model-based segmentation, CAIP'95 Proceedings of International conference on Computer Analysis and Image Processing, Prague, Czech Republic, 1995.

[10] P. Radeva, J. Serrat, E. Marti, A snake for model-based segmentation, ICCV'95 Proceedings of International Conference on Computer Vision, MIT, USA, 1995.

[11] Kok F. Lai, Roland T. Chin, Deformable contour: modeling and extraction, IEEE Transactions on Pattern Analysis and Machine Intelligence 17 (11) (1995) 1084–1090.

[12] Y.F. Wang, J.F. Wang, Surface reconstruction using deformable models with interior and boundary constraints, IEEE Transactions on Pattern Analysis and Machine Intelligence 14 (5) (1992) 572–579.

[13] A. Sclaroff, A.P. Pentland, Modal matching for correspondence and recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 17 (6) (1995) 545–561.

[14] E.E. Milios, Shape matching using curvature process, Computer Vision, Graphics and Image Processing 47 (1989) 203–226.

[15] M. Baroni, G. Barletta, Digital curvature estimation for left ventricular shape analysis, Image and Vision Computing 10 (7) (1992) 485–494.

[16] M.H. Han, D. Jang, The use of maximum curvature points for the recognition of partially occluded objects, Pattern Recognition 23 (1/2) (1990) 21–33.

[17] F. Mokhtarian, Silhouette-based isolated object recognition through curvature-scale space, IEEE Transactions on Pattern Analysis and Machine Intelligence 17 (5) (1995) 539–544.

[18] F. Mokhtarian, Silhouette-based object recognition with occlusion through curvature scale Space, Proceedings of the European Conference on Computer Vision, Cambridge, UK, Vol.1, 1996, pp. 566–578.

[19] D. Cyganski, R.F. Vaz, A linear signal decomposition approach to affine invariant contour identification, Pattern Recognition 28 (12) (1995) 1845–1853.

[20] R.F. Vaz, D. Cyganski, Generation of affine invariant local contour feature data, Pattern Recognition Letters 11 (1990) 479–483.

[21] K. Arbter et al., Application of affine-invariant Fourier descriptors to recognition of 3-D objects, IEEE Transactions on Pattern Analysis and Machine Intelligence 12 (7) (1990) 640–647.

[22] Q.M. Tieng, W.W. Boles, An application of wavelet-based affine-invariant representation, Pattern Recognition Letters 16 (1990) 1287–1296.

[23] Y.L. Fok, J. Chan, R.T. Chin, Automated analysis of nerve-cell images using active contour models, IEEE Transactions of Medical Imaging 15 (3) (1996) 353–368.

[24] M.D. Jolly, S. Lakshmanan, A.K. Jain, Vehicle segmentation and classification using deformable templates, IEEE Transactions on Pattern Analysis and Machine Intelligence 18 (3) (1996) 292–308.

[25] L.D. Cohen, Note on active contour models and balloons, CVGIP: Image Understanding 53 (2) (1991) 211–218.

[26] A.A. Amini, T.E. Weymouth, R.C. Jain, Using dynamic programming for solving variational problems in vision, IEEE Transactions on Pattern Analysis and Machine Intelligence 12 (9) (1990) 855–867.

[27] D.J. Williams, M. Shah, A fast algorithm for active contours and curvature estimation, Computer Vision Graphics, Image Processing 55 (1992) 14–26.

[28] C.T. Tsai, T.N. Sun, P.C. Chung, Minimizing the energy of active contour model using a Hopfield network, IEE Processings Part E: Computers and Digital Techniques 140 (6) (1993) 297–303.

[29] K.F. Lai, Roland T. Chan, On regularization, formulation and initialization of the active contour models (snakes), ACCV'93 Asian Conference on Computer Vision, Osaka, Japan, 23–25 November, pp. 542–545.

[30] G. Xu, E. Segawa, T. Saburo, Robust active contours with insenstitive parameters, Pattern Recognition 27 (7) (1994) 879–884.

[31] P. Radev, J. Serrat, E. Marti, A snake for model-based segmentation, Proceedings of International Conference on Computer Vision (ICCV'95), MIT, USA, 1995.

[32] M. Worring et al., Parameterized feasible boundaries in gradient vector fields, CVGIP: Image Understanding 63 (1) (1996) 135–144.

[33] E. Rivlin, I. Weiss, Local invariants for recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 17 (3) (1995) 226–238.

[34] A. Rattarangsi, R.T. Chin, Scale-based detection of corners of planar curves, IEEE Transactions on Pattern Analysis and Machine Intelligence 14 (4) (1992) 430–449.

[35] D.M. Tsai, M.F. Chen, Curve fitting approach for tangent angle and curvature measurements, Pattern Recognition 27 (5) (1994) 699–711.