A Discrete Dynamic Contour Model

Steven Lobregt and Max A. Viergever

Abstract— **A discrete dynamic model for defining contours in 2-D images is developed. The structure of this model is a set of connected vertices. With a minimum of interaction, an initial contour model can be defined, which is then automatically modified by an energy minimizing process. The internal energy of the model depends on local contour curvature, while the external energy is derived from image features. Solutions are presented to avoid undesirable deformation effects, like shrinking and vertex clustering, which are common in existing active contour models. The deformation process stops when a local minimum of the energy function is reached. The final shape of the model is a reproducible approximation of the desired contour. Results of applying the method to computer-generated images, as well as clinical images, are presented.**

I. Introduction

MANY applications in medical imaging rely on the definition of object contours. Object contour definition can be accomplished in a number of ways:

1) *Completely Manual*: An operator will have to sit before a screen and spend a considerable amount of time to draw the required contours manually, using some kind of pointing device like a mouse or a graphics tablet. This requires expert knowledge of the operator about the clinical problem, as well as a certain level of skill in drawing contours with the available tools. Manual definition of contours is a difficult and time consuming process which presents a serious bottleneck in processing large (three-dimensional) data sets, e.g., for diagnostic purposes or therapy planning. Moreover, manual contour definition suffers from a very low degree of reproducibility [5].
2) *Fully Automatic*: The techniques which are presently available for automatic contouring are not sophisticated enough for many typical applications. The use of these techniques is therefore restricted to the detection of simple contours of specific objects, like endocardial contours of the left ventricle.
3) *Automated First Guess, Followed by Manual Editing*: The first guess is usually based on simple techniques like thresholding or region growing. In the manual editing phase, the operator modifies the generated contour, using not only image information, but also contextual knowledge about local anatomy and pathology. This approach is still very time consuming and leads to results that are not reproducible.
4) *Manual Rough Delineation, Followed by Automated Contour Definition*: The user interaction is now limited, since only a rough delineation is required to serve as an initial contour. The following automatic contour definition process will refine the contour, controlled by (possibly) application-specific parameters. The influence of the operator on the final result is, in this case, indirect. A certain amount of variation in initial contours (for instance, drawn by different operators) will still allow for the same final result to be produced, provided that the same parameter settings are used for the automatic contour definition process, leading to a high degree of reproducibility.

The present paper follows an approach as described under 4), because of the high reproducibility combined with limited user interaction. The proposed dynamic contour model may serve as a basis for an automated contour definition method.

Dynamic contour models have become en vogue with the Snake model of Terzopoulos and coworkers [2], [4], [9], and have, since then, been investigated and applied in various ways [6]–[8], [11], [12], [14]–[17].

Terzopoulos' Snake model builds a deformable contour consisting of connected spline segments and lets the contour approximate a desired form by minimizing an energy function containing internal and external energy. The *internal energy* is the bending energy of the spline, the *external energy* is calculated by integrating image features, like the presence of lines or edges, along the paths of the spline segments. User-defined constraints, like springs that pull on the spline segments, yield a third energy term: the *constraint energy*, which is used to locally restrict the deformation of the contour according to the user's wishes.

A more recently introduced contour model is the *Geometrically Deformed Model* (GDM) of Miller cf. [6]–[8], [11], which describes a contour as a set of vertices, connected by edges. The energy function defined to control the GDM is quite different from that of the Snake model: a *topology preserving energy* term, dependent on an estimation of local curvature and the distance between a vertex and its neighbors, an *image event energy* term derived from the thresholded pixel values, and a locally defined *deformation potential* driving the vertices outward or inward. The energy function is evaluated only for the vertex positions, not for the trajectory of the connecting edge segments. This makes the model discrete; the length of the connecting edges defines, in fact, its resolution.

Our approach follows the GDM only by adopting the basic structure of the model: vertices connected by edges. The

Manuscript received July 30, 1993; revised June 21, 1994. The Associate Editor responsible for coordinating the review of this paper and recommending its publication was J. Duncan.

S. Lobregt is with the Department of Advanced Development, Common Digital Systems, Philips Medical Systems, 5680 DA Best, The Netherlands; e-mail: slobregt@globe.ms.philips.nl.

M. A. Viergever is with Utrecht University, The Netherlands.

IEEE Log Number 9408767.

0278–0062/95$04.00 © 1995 IEEE

dynamic process controlling the contour refinement differs in nature from that of the GDM, however, and is described in terms of forces and force fields acting on the vertices rather than in terms of energies. This description of the deformation process is completely equivalent with a description in terms of (potential) energy contributions [1]. However, a description in terms of forces is more convenient for our discrete model because the forces relate directly to the acceleration and displacement of the vertices.

Starting from an initial shape, which is created with a minimum of user interaction, the dynamic contour model actively modifies its shape, thus approximating some desired contour. The driving force behind the shape deformation is calculated from internal forces, derived from the shape of the contour model itself, and an external force field, derived from some image feature energy distribution. The internal forces will try to minimize local contour curvature, while the external forces will try to make the model follow a valley or a ridge through the "landscape" formed by the image feature. An image feature may be simple, like the pixel or voxel gray value, or the magnitude of the gray value gradient, but can also be quite complex, like those derived by means of differential geometry [13], [18], [19]. By applying both internal and external forces with user definable weight factors, the user can determine to either follow the image feature landscape in a very global way, or very precisely, or anything in between.

The deformation process is performed in a number of discrete steps, after each of which the situation with regard to position, velocity, and acceleration is evaluated for each of the vertices. In this evaluation, internal and external forces on a vertex are calculated from the position of the vertex and its neighbors. These forces result in an acceleration, which changes the velocity of the vertex. This velocity determines the displacement of the vertex during the next deformation step. After a number of deformation steps, a stable end situation will be reached in which there is an equilibrium, which means that velocity *and* acceleration are zero for each vertex. Described in terms of energies, this situation represents a local minimum of the energy function.

During deformation, there are two undesirable effects which may occur: shrinking of closed models owing to internal forces, and clustering or gathering of vertices in corners of the model, owing to external forces. We found a solution to the first problem in a proper definition of local curvature at the vertices, combined with internal forces that are zero for parts of the contour where the curvature is constant. The solution to the second problem is found by allowing only locally radial vertex displacements. Both issues will be discussed in Section II.

The goal at which our work is aimed is to develop a tool to assist the operator with the definition of contours in medical images. Such a tool should reduce the required user interaction significantly, thereby saving a lot of time, and provide reproducible results. The method to be developed can be applied to any pixel- or voxel-type description of some object space. This type of data is routinely generated nowadays by various data acquisition modalities in the clinical environment like CT, MRI, SPECT, PET, and US. The use of contour finding methods like the one described here is, however, not restricted to medical applications. Other areas in which deformable models have been used are computer graphics and animation [3], [10]. Compared to model independent contour detection or extraction methods, the strength of a model-based method is that it treats a contour as one single connected object. Connectivity of the contour is therefore guaranteed, also in image areas where the image features used to find the desired contour are locally very weak or even absent.

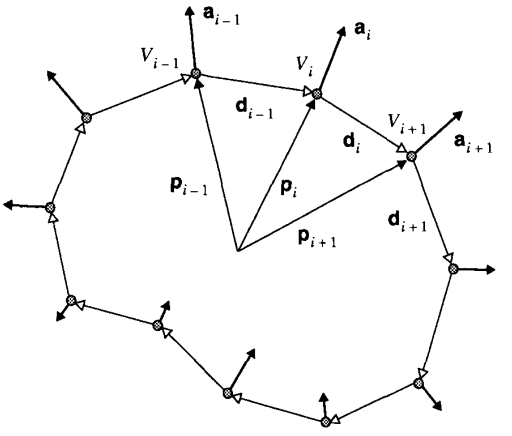

Fig. 1. The model consists of a set of vertices V_i which are connected by edges $\mathbf{d}_i$. Deformation is caused by acceleration forces $\mathbf{a}_i$ acting on the vertices.

In the present work, the 2-D version of the model is described. Extension of the model to 3-D images is straightforward, inasmuch as the structure of the model will still be edge-connected vertices, and evaluation will still be performed locally on vertex positions.

The organization of the paper is as follows. Section II describes the structure of the model and defines the forces which control its behavior; the internal forces will be described in Section II-A and the external force field in Section II-B. A description of the dynamics of the deformation process is presented in Section III-A, and a technique for local resampling of the contour in Section III-B. Section IV describes an application of the method to a variety of computer-generated and clinical images. Section V concludes the paper with a brief discussion.

II. Forces and Force Field

Fig. 1 presents the basic structure of the model: a contour consisting of vertices which are connected by straight line segments or edges. The position of a vertex V_i is represented by a vector $\mathbf{p}_i$, and the edge between V_i and V_{i+1} by a vector $\mathbf{d}_i$. (We assume the coordinate system to be Cartesian.) Deformation is caused by a combination of forces which act on the vertices; the resulting acceleration in vertex V_i is denoted by a vector $\mathbf{a}_i$. Another property of a vertex, not shown in the figure but important for the dynamic behavior of the model, is its velocity, denoted by $\mathbf{v}_i$ for vertex V_i.

The length d_i of an edge segment represents the local resolution of the model: if it is large, the model will not be able

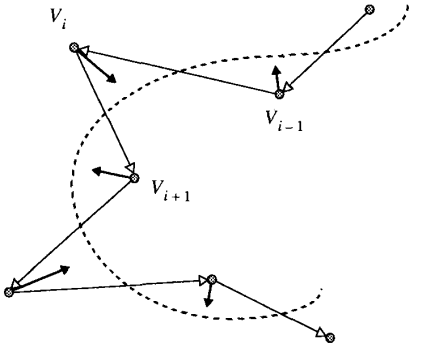

Fig. 2. The main objective of the internal forces is to minimize the local contour curvature.

to follow variations of small scale in the image feature energy distribution. The length d_i may change at every deformation step, thus causing local variation in the resolution of the model. To keep this variation limited, the edge lengths are evaluated at regular intervals; where necessary, vertices are removed or inserted, thus keeping the model resolution close to a user-specified scale. The details of this local resampling process will be discussed in Section III-B.

A. Internal Forces

The internal forces defined in our model are related to the local contour curvature, in conformity with existing active contour models. The main objective of introducing internal forces or energy functions is to minimize local curvature (see Fig. 2), thus forming a counterbalance to external forces that try to shape the model according to all the variations of the image feature landscape.

Before continuing, we must first define the concept of local curvature in our discrete model, which is not a trivial matter. Strictly speaking, local curvature is zero on the straight edge segments in between the vertices, while it is not defined at the exact position of a vertex (first-order discontinuity), which however happens to be exactly the position where we *need* to define it.

We found a satisfactory solution to this problem by defining local curvature at the position of a vertex to be the difference between the directions of the two edge segments that join at that location. The edge segment leaving from vertex V_i is represented by a vector $\mathbf{d}_i$; its direction is described by the unit vector $\hat{\mathbf{d}}_i$. According to the definition above, the local curvature $\mathbf{c}_i$ at V_i is described by (see Fig. 3)

$$\mathbf{c}_i = \hat{\mathbf{d}}_i - \hat{\mathbf{d}}_{i-1}. \tag{1}$$

Defined in this way, local curvature has length (strength), as well as direction, and provides a usable and unique measure for the angle between two joining edge segments. Moreover, the length of our curvature vector depends *only* on this angle, and is not influenced by the lengths of the two joining edge segments.

We will also define locally radial and tangential directions at the position of a vertex. For this, we again make use of

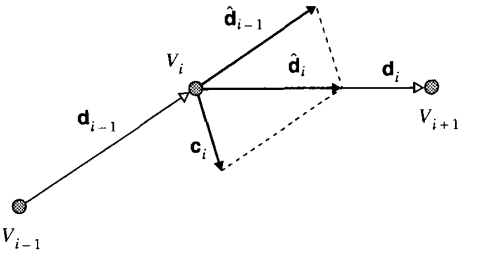

Fig. 3. Local curvature $\mathbf{c}_i$ at the position of a vertex V_i is defined as the difference between the directions of the segments that meet in the vertex. These directions are defined by the unit vectors $\hat{\mathbf{d}}_{i-1}$ and $\hat{\mathbf{d}}_i$.

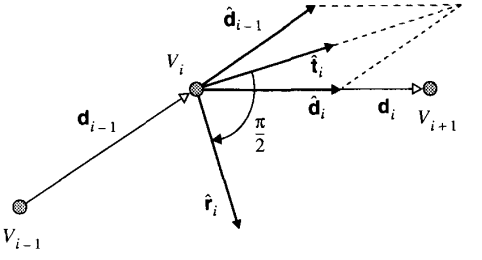

Fig. 4. The local tangential direction $\hat{\mathbf{t}}_i$ at a vertex V_i is defined as the normalized sum of $\hat{\mathbf{d}}_{i-1}$ and $\hat{\mathbf{d}}_i$. The local radial direction $\hat{\mathbf{r}}_i$ is defined as a $\pi/2$ rotation of $\hat{\mathbf{t}}_i$.

the unit vectors $\hat{\mathbf{d}}_i$ representing the directions of the edge segments $\mathbf{d}_i$. To define a locally tangential unit vector $\hat{\mathbf{t}}_i$, we use the normalized sum of the unit vectors of two joining edge segments (see Fig. 4)

$$\hat{\mathbf{t}}_i = \frac{\hat{\mathbf{d}}_i + \hat{\mathbf{d}}_{i-1}}{||\hat{\mathbf{d}}_i + \hat{\mathbf{d}}_{i-1}||}. \tag{2}$$

The unit vector $\hat{\mathbf{r}}_i$ in the local radial direction is derived from $\hat{\mathbf{t}}_i$ by a rotation over $\pi/2$ radians

$$\hat{\mathbf{r}}_i = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \hat{\mathbf{t}}_i. \tag{3}$$

The vectors $\hat{\mathbf{t}}_i$ and $\hat{\mathbf{r}}_i$ now represent a local coordinate system at the position of vertex V_i, which will prove useful for the calculation of internal as well as external forces.

Both closed and open models are allowed. If the number of vertices equals n and the model is closed, the first vertex V_0 and the last vertex V_{n-1} are connected, so V_0 has two neighbors: V_{n-1} and V_1. If, however, the model is open, V_0 and V_{n-1} are not connected and both have just one neighbor: V_0 only connects to V_1, and V_{n-1} only connects to V_{n-2}. This situation requires special measures for the calculation of the local tangential and radial directions $\hat{\mathbf{t}}_i$ and $\hat{\mathbf{r}}_i$ as well as the curvature vector $\mathbf{c}_i$. At the position of the open ends, we define the local tangential direction to be equal to the direction of the first or the last contour segment: $\hat{\mathbf{t}}_0 = \hat{\mathbf{d}}_0$ and $\hat{\mathbf{t}}_{n-1} = \hat{\mathbf{d}}_{n-2}$. The length of the curvature vector is set to zero for both end positions: $\mathbf{c}_0 = \mathbf{c}_{n-1} = 0$.

If we describe the local curvature vector $\mathbf{c}_i$ in terms of the local r, t-coordinate system, we see that $\mathbf{c}_i$ is pointing either

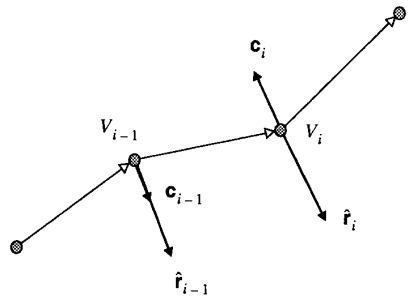

Fig. 5. Positive curvature $\mathbf{c}_{i-1}$ points in the same direction as $\hat{\mathbf{r}}_{i-1}$, while negative curvature $\mathbf{c}_i$ points in the opposite direction of $\hat{\mathbf{r}}_i$.

in exactly the direction of $\hat{\mathbf{r}}_i$, or in the opposite direction. In other words, $\mathbf{c}_i$ is a vector along the local r-axis and its length can be described by the dot product $(\mathbf{c}_i \cdot \hat{\mathbf{r}}_i)$. We will make use of the fact that, according to this definition, the length c_i of the curvature vector can be positive as well as negative

$$\mathbf{c}_i = (\mathbf{c_i} \cdot \hat{\mathbf{r}}_\mathbf{i})\hat{\mathbf{r}}_i. \tag{4}$$

See also Fig. 5 for an illustration of positive and negative local curvature.

Now that we have defined local curvature as a one-dimensional variable in the local radial direction, we can proceed toward defining the internal forces which should act on the model's vertices and put restrictions on the deformation process. In order to get a clear understanding of the contribution that internal forces will make to the deformation of the model, we consider a situation in which external forces are completely absent. If we compare Figs. 2, 3, and 5, we are tempted to believe that the desired internal forces (Fig. 2) and the curvature vectors (Fig. 3 and 5) are very closely related. This is indeed true: both are vectors with the same orientation. It would, however, be unwise to define the internal forces as being proportional to the local curvature vectors. The reason for this is illustrated in the left part of Fig. 6(a). Any simple closed shape would, in the absence of external forces, be deformed into the shape having the minimum overall local curvature, i.e., a circle (or rather, because of the discretization, a symmetric polygon). Then, however, the deformation process would not stop, but continue to move the vertices in the direction of the center of the model, as will be clear from Fig. 6(a). During this last phase, the model is shrinking while local curvature will not change, which means that the internal forces are not doing what they were intended to do: reduce local curvature. The model will implode into a single point.

Miller [6]–[8], [11] related the GDM's internal constraint forces directly to his local curvature measure: the angle between joining edges (in 2-D). He therefore faced the above shrinking problem, which he managed to bring to a halt by introducing a second constraint on the position of a vertex with respect to its neighbors. This second constraint acts like an elastic force that keeps the distance between neighboring vertices between certain limits. Vertices which are moving closer together will experience an increasingly repelling force that, at some point, will stop the shrinking

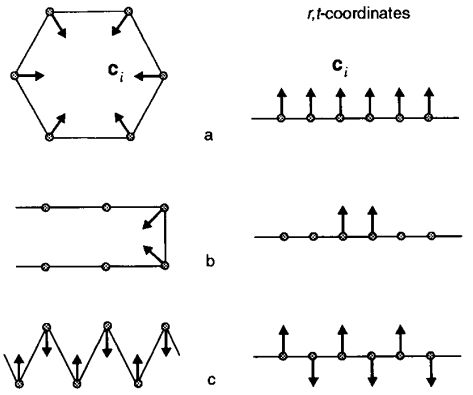

Fig. 6. Curvature vectors $\mathbf{c}_i$ for some typical situations; on the left shown in Cartesian representation, on the right in local r, t-coordinates.

process. The point where the shrinking process comes to a stop depends on the weights that are assigned to the two internal constraint forces. In other words, it is required to optimize the balance between the internal elastic force and the internal curvature minimizing force. This may not always be possible in the presence of external forces, as we will show in the next section. In order to avoid the introduction of the second—rather artificial—constraint and the problems entailed by it, we have sought an alternative definition of the internal curvature reducing force. Our requirement was that *local curvature should be reduced without affecting parts of the contour with constant curvature.*

Fig. 6 shows, besides the already introduced symmetric polygon of (a), some typical shapes which may occur in a discrete contour model, and which we will use to illustrate our solution to the shrinking problem. Fig. 6(b) shows a turn in the direction of a contour over π radians and Fig. 6(c) a part of a contour with alternating curvature direction.

The problem of the shape of Fig. 6(a) has been discussed above; in the absence of external forces, the polygon will implode if the internal forces are taken to be proportional to the local curvature.

The shape of Fig. 6(b) presents a similar problem. It forms an extended part of a contour. If we would apply a force proportional to local curvature to the vertices, the closed end of the contour would become shorter, as shown in the figure. However, as with the shrinking of the shape of Fig. 6(a), this would not actually reduce local curvature, but only displace the curved region. The shape of Fig. 6(c) does not pose a problem; we include this shape because it represents a typical situation in which local curvature reduction may be required. Any solution to the shrinking problem should still perform well on the shape of Fig. 6(c).

On the right side of Fig. 6, the same three shapes are shown, but now in local r, t-coordinates. Examination of the curvature vectors in *this* coordinate system, combined with our intention to derive from these vectors the internal forces that would meet our above-stated criteria, led us to a simple solution. First, internal forces $\mathbf{f}_{\text{in},i}$ which act on the vertices V_i should have the same (radial) direction as the curvature vectors. This means

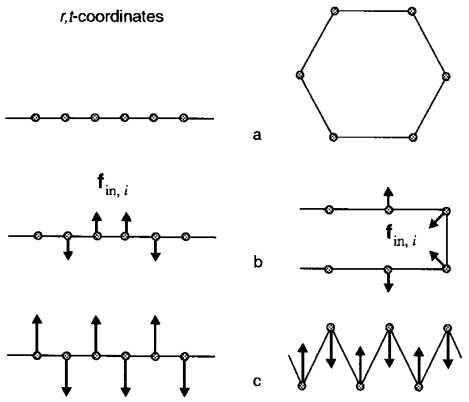

Fig. 7. Internal forces $\mathbf{f}_{\text{in},i}$ as derived from the curvature vectors $\mathbf{c}_i$ of Fig. 6.

that internal forces can be derived from the curvature vectors by modifying only their lengths. Second, in order to reduce local curvature without affecting areas of constant curvature, the lengths of the internal force vectors should be zero for parts of the contour with constant curvature. Both conditions can be met if we consider the sequence $\mathbf{c}_i \cdot \hat{\mathbf{r}}_i$ along the contour as a discrete scalar function, dependent on the position i, and use the convolution of this function with a discrete filter k_i as the representation of the sequence of internal force vector lengths $\mathrm{f}_{\text{in},i}$

$$\mathrm{f}_{\text{in},i} = (\mathbf{c}_i \cdot \hat{\mathbf{r}}_i) \otimes k_i. \tag{5}$$

The first condition can be met by using $\hat{\mathbf{r}}_i$ as the direction of $\mathbf{f}_{\text{in},i}$

$$\mathbf{f}_{\text{in},i} = \mathrm{f}_{\text{in},i}\hat{\mathbf{r}}_i. \tag{6}$$

The second condition is met by choosing appropriate filter coefficients for k_i. If we want the result of convolving a sequence of constant values with k_i to be zero, then we must define k_i such that it is a symmetric discrete filter with a zero frequency component equal to zero. A wide class of filters complies with this condition. We have chosen and applied the simplest form, which is a uniform filter consisting of three nonzero coefficients with the values

$$k_i = \left\{\cdots, 0, 0, -\tfrac{1}{2}, 1, -\tfrac{1}{2}, 0, 0, \cdots\right\} \tag{7}$$

where the value 1 applies to position i and the values $-\frac{1}{2}$ to positions $i-1$ and $i+1$.

We intend to investigate other possible filters in the future because this convolution may provide an interesting way to optimize the behavior of the model for specific applications. In particular, it may be worthwhile to make k_i adaptive.

Fig. 7 shows the internal force vectors for the contour shapes of Fig. 6, derived from the curvature vectors according to (5)–(7). The left side of the figure shows the internal forces in local r, t-coordinates; the right side shows the shapes and internal forces in Cartesian coordinates. The constant curvature of the shape of Fig. 7(a) has produced internal forces that are zero everywhere, so that the shrinking problem has been solved. The π radians turn of the shape of Fig. 7(b) is deformed in a way that appears more natural. The closed end of the contour becomes not only shorter, but also *wider*, which is what we would expect from curvature reduction. The alternating curvature of the shape of Fig. 7(c) will still be reduced effectively, because the effect of convolution with k_i on a purely alternating signal comes down to multiplication with a constant.

In consequence, the internal forces $\mathbf{f}_{\text{in},i}$, as defined by (5)–(7), produce the desired deformation effects to our contour model, in the absence of any external forces.

The next section will cover the external force field and present a solution for the vertex clustering problem related with it.

B. External Forces

To provide a driving force for the deformation of the model, an external potential energy distribution is assumed, which represents the energy or strength of some kind of image feature or combination of image features. The selection of appropriate image features for specific applications is an important research subject, but not within the scope of this paper. In future work, we aim at using the results of the investigations into scale space and differential invariants [13], [18], [19] to define application-specific image features. In this paper, however, we will demonstrate the behavior of the model using simple, but effective, image features like the pixel or voxel gray value itself and the length of the gray value gradient.

If, for instance, we want the model to follow a maximum gradient path through the image, we could use the gradient length as an image feature and define an energy distribution that is high for large values of this feature. A maximum gradient path is then represented by a ridge in this energy distribution. The implementation of the deformation process is such that it will attempt to pull the vertices into local minima of the energy distribution, which is a natural behavior. For the model as a whole, this means that it will end up following a path of low energy or a *valley* through the feature energy landscape. If we want the model to follow a *ridge* instead of a valley, we can do so by inverting the energy distribution. We will call the resulting distribution of potential energy E_{im}. The force field that will pull an object in the direction of lower energy, can be described by the following simple relation:

$$\mathbf{f}_{im} = -\nabla E_{im}. \tag{8}$$

If we apply this force to the vertices of our model, and consider a situation in which there are no internal forces, then the result will be that, in the end, the contour model connects local energy minima, following a valley, through the external energy distribution.

However, if we would actually apply the above-described force field, we would soon run into a problem which is common to all similar methods: the force $\mathbf{f}_{\text{im},V_i}$ which acts on the vertex V_i, will, in general, not only have a component perpendicular to the local direction of the contour model (locally radial component), but also a component *along* the path

of the model (locally tangential component). This tangential component may be substantial, or even dominant. If there were no restrictions on the curvature of the model (e.g., in the absence of internal forces), then, in the final situation, the model would actually pass *through* the local minima of the external energy distribution, in which case the force field would locally be directed *along* the path of the contour. The result of this locally tangential component is that vertices will move along the contour and form clusters in local minima of the external energy distribution, which is obviously a very undesirable effect.

Let us now get back to the option of introducing an elastic force as a second internal force in order to keep the distance between neighboring vertices within limits, assuming that such a force could be an effective answer to the clustering problem. In this situation, the strength of the elastic force would have to be tuned *locally* to the strength of the external force $\mathbf{f}_{\mathrm{im},V_i}$ at the location of vertex V_i. This external force varies, not only over the area of the image matrix, but also from image to image, and it depends on many parameters related to both acquisition and processing of the data, like, for instance, the feature extraction algorithm that was used. Careful tuning of the internal elastic force is important: If the elastic force is too weak, it may lead to shrinking of the model owing to internal curvature forces, or it may lead to clustering of the vertices owing to external forces. If the elastic force is too strong, it may obstruct the deformation process of the model, because, during deformation, the vertices *must* have the freedom to move and change the distance to their neighbors.

We have found experimentally that the needed compromise is unaccomplishable. The internal elastic force cannot be tuned locally to the external forces $\mathbf{f}_{\mathrm{im},V_i}$ and, at the same time, be tuned correctly to the internal forces $\mathbf{f}_{\mathrm{in},i}$.

Our solution to the clustering problem is an obvious one and also in line with our solution to the shrinking problem, which was discussed in the previous section: Vertex displacement *along* the path of the contour model does not make any contribution to the deformation of the model, which is what this model is all about. Therefore, we will not use the force $\mathbf{f}_{\mathrm{im},V_i}$ itself, but decompose this force into a locally radial and a locally tangential component, and use *only* the locally radial component to drive the vertices of the contour model.

If we denote the locally radial component of $\mathbf{f}_{\mathrm{im},V_i}$ by $\mathbf{f}_{\mathrm{im},\mathbf{r}_i}$, then its length is given by the dot product of $\mathbf{f}_{\mathrm{im},V_i}$ and $\hat{\mathbf{r}}_i$

$$\mathbf{f}_{\mathrm{im},\mathbf{r}_i} = (\mathbf{f}_{\mathrm{im},V_i} \cdot \hat{\mathbf{r}}_i)\hat{\mathbf{r}}_i. \tag{9}$$

See also Fig. 8. The locally radial forces $\mathbf{f}_{\mathrm{im},\mathbf{r}_i}$ and $\mathbf{f}_{\mathrm{in},i}$ provide a resulting driving force on the vertices of the contour model which is purely devoted to deformation of the model, without moving vertices along the path of the contour.

It is mandatory for a user of dynamic contour modeling tools to have control over the contouring process. Firstly, only few real-world images lend themselves to fully automatic contouring of satisfactory quality; some operator guidance will be desirable for most object definition purposes. Secondly, the state of minimum energy reached by the model will, in general, be but one of a large set of possible local minima,

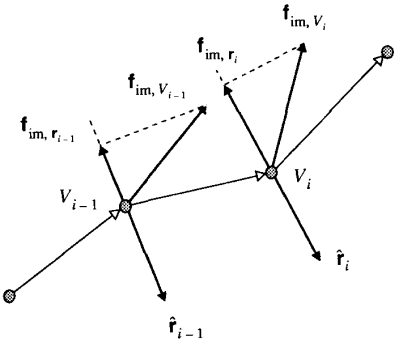

Fig. 8. Only the locally radial component $\mathbf{f}_{\mathrm{im},\mathbf{r}_i}$ of the external force $\mathbf{f}_{\mathrm{im},V_i}$ is used for deformation of the model.

and the operator must have the means to push the model out of a local minimum into another. In the Snake model, this was realized by introducing user defined "volcanos," "pits," and "anchored springs" [4]. One of the future extensions to our contour model will be to offer the user a possibility to interactively create and modify a user-defined external energy distribution E_{user}, which will be added to the calculated image feature energy distribution E_{im} to form the combined external energy distribution E_{ex}. The user-defined distribution enables the operator to create additional ridges and valleys through the energy landscape to locally force the model to follow a particular path. Once these two external energy distributions are combined, the situation is completely equivalent to the one above, and calculation of the resulting external force $\mathbf{f}_{\mathrm{ex},\mathbf{r}_i}$, acting on a vertex V_i of the model becomes

$$E_{\mathrm{ex}} = E_{\mathrm{im}} + E_{\mathrm{user}} \tag{10}$$

$$\mathbf{f}_{\mathrm{ex}} = -\nabla E_{\mathrm{ex}} \tag{11}$$

$$\mathbf{f}_{\mathrm{ex},\mathbf{r}_i} = (\mathbf{f}_{\mathrm{ex},V_i} \cdot \hat{\mathbf{r}}_i)\hat{\mathbf{r}}_i. \tag{12}$$

After thus having defined the internal and external model forces, we are well equipped to describe the dynamics of the deformation process. This will be the subject of the next section.

III. Deformation and Resampling

The distance between a vertex and its neighbors determines the resolution of the contour model: details of the external image feature energy distribution which are small enough may pass through the spaces between the vertices without having a significant influence on the final shape of the model. Because of the deformation of the model, the vertex to vertex distance will change constantly. This may result in local variation as well as in global changes in the resolution of the model. Both are undesirable, but slight local variation in the resolution is unavoidable because a deformation process can only be possible if the vertices have the freedom to move with respect to their neighbors. What we can do, however, is keep this variation between certain limits by periodically resampling the contour model along its path. In local r, t-coordinates, this translates into a resampling along the t-axis.

The actual deformation process will be discussed in the next subsection, followed by a description of the resampling process.

A. Deformation

The total force $\mathbf{f}_i$ acting on a vertex is a weighted combination of external and internal forces

$$\mathbf{f}_i = w_{\text{ex}}\mathbf{f}_{\text{ex},\mathbf{r}_i} + w_{\text{in}}\mathbf{f}_{\text{in},i}. \tag{13}$$

The weighting factors w_{ex} and w_{in} may have default values for each application but allow modification by the operator. Upon emphasizing external forces, one will make the model follow the extracted image features more precisely, while putting emphasis on internal forces will smooth out the path of the contour. As a result of the forces that act on a vertex V_i, this vertex will start to move and change its position $\mathbf{p}_i$. This position vector, together with the vertex velocity and acceleration vectors $\mathbf{v}_i$ and $\mathbf{a}_i$, describe the dynamic state of a vertex. A vertex will not stop moving until both $\mathbf{v}_i = \mathbf{0}$ and $\mathbf{a}_i = \mathbf{0}$. The deformation process of the model as a whole is not completed before the condition $\mathbf{v}_i = \mathbf{a}_i = \mathbf{0}$ is met by *all* its vertices.

It may, in principle, be a very long time before the contour model comes to a rest, or the model may even remain oscillating between two states which both represent a local energy minimum. We therefore added a third component to the force that is applied to a vertex V_i, viz., a damping force $\mathbf{f}_{\text{damp}}$, proportional to the vertex velocity $\mathbf{v}_i$

$$\mathbf{f}_i = w_{\text{ex}}\mathbf{f}_{\text{ex},\mathbf{r}_i} + w_{\text{in}}\mathbf{f}_{\text{in},i} + \mathbf{f}_{\text{damp},i} \tag{14}$$

$$\mathbf{f}_{\text{damp},i} = w_{\text{damp}}\mathbf{v}_i. \tag{15}$$

The weighting factor w_{damp} is negative and determines the amount of damping. Even a small damping factor w_{damp} suffices to ensure stability of the deformation process.

While the other weighting factors w_{ex} and w_{in} are dimensionless, w_{damp} is not. We chose to mimic the damping which is experienced by a moving particle in a fluid or a gas. In this case, the damping force is proportional to the velocity of the particle and points in opposite direction. This damping force is expressed in the formula of Stokes: $\mathbf{f} = 6\pi r\eta\mathbf{v}$. The factor w_{damp} can be written as $w_{\text{damp}} = c\eta$ for a specific particle in which c is a constant and η the viscosity coefficient of the fluid or the gas. The constant c depends on the radius of the particle and has the dimension [m]; the coefficient η has the dimension [kg/(m · s)]. The dimension of w_{damp} therefore is [kg/s]. It follows that $\mathbf{f}_{\text{damp}}$ has the dimension of a force [(kg · m)/s^2].

The actual deformation process is implemented as a so-called numerical time integration process in which the complete state of the contour model is calculated at a sequence of discrete positions in time. A similar method was described by Terzopoulos [9]. If we use the argument t to describe the state of the model at a certain point t in time, and $t + \Delta t$ for the situation at a time Δt later, then the deformation process during the incremental time Δt can be described by the following difference scheme:

$$\mathbf{p}_i(t + \Delta t) = \mathbf{p}_i(t) + \mathbf{v}_i(t)\Delta t \tag{16}$$

$$\mathbf{v}_i(t + \Delta t) = \mathbf{v}_i(t) + \mathbf{a}_i(t)\Delta t \tag{17}$$

$$\mathbf{a}_i(t + \Delta t) = \frac{1}{m_i}\mathbf{f}_i(t + \Delta t). \tag{18}$$

The value of $\mathbf{f}_i(t + \Delta t)$ in (18) is calculated from (14). The value m_i in (18) would, in a physical context, represent the "mass" of vertex V_i. We do not, however, make use of the possibility to assign different mass values to the vertices of the model in our present implementation. By assuming the same mass for all vertices, the factor $1/m_i$ reduces to a constant scaling factor.

An option, available for open contours, is a growing process which adds vertices to one or both of the open ends of the contour model. We integrated this growing process with the deformation of the model, which resulted in an efficient tracking method. Vertices which are added are positioned by extrapolating in the direction of the last (or first) edge segment, which is, at the same time, continuously being repositioned by the deformation process. We also added an automatic closing option, such that both ends of an open contour are automatically connected when the distance between them becomes smaller than a certain threshold. In Section IV, we will show examples of open contours that "grow" around an edge in the image, close themselves, and from there on behave like any other closed contour.

B. Resampling

In the prototype implementation that we developed, the user can interactively control a parameter l_{des}, the desired length for the model's edge segments. In this way, the resolution of the model can be chosen. From l_{des}, two other values l_{min} and l_{max} are derived, representing the minimum and maximum distance which is allowed between neighboring vertices.

The resampling step is implemented as a two-pass process: The first pass checks along the entire contour if any segment length has become shorter than the minimum length l_{min}. If this is the case, this edge segment is removed from the model by replacing the two vertices on both ends of this segment by one single vertex at a position exactly in between the replaced vertices. This is illustrated in Fig. 9(a). The second pass checks again along the entire contour, but now for segments with a length larger than the maximum length l_{max}. Such an edge segment is divided into two shorter ones of equal length, as illustrated in Fig. 9(b).

The values of l_{min} and l_{max} are derived from the user-defined parameter l_{des}. The relation between l_{min} and l_{max} is constrained by not admitting an oscillatory behavior, in which vertices are repeatedly removed in one resampling action and inserted again in the next. This leads to the requirement that $l_{\text{max}} > 2l_{\text{min}}$. In our prototype implementation, we used the following relationship for l_{min}, l_{max}, and l_{des}, with satisfactory result:

$$l_{\text{min}} = \tfrac{1}{2}\, l_{\text{des}} \tag{19}$$

$$l_{\text{max}} = \tfrac{3}{2}\, l_{\text{des}}. \tag{20}$$

When a vertex is placed in the contour model, it is also assigned a velocity $\mathbf{v}_i$ and acceleration $\mathbf{a}_i$ which are obtained

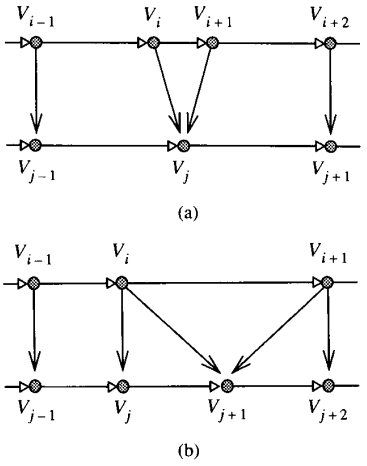

Fig. 9. Resampling of the contour, (a) an edge element shorter than $l_{\min}$ is removed, and (b) an edge element longer than $l_{\max}$ is split.

by averaging the velocities and accelerations of the two vertices which are replaced (Fig. 9(a)) or between which it is inserted (Fig. 9(b)), which is necessary to maintain continuity in the dynamic situation of the model.

IV. Results

We developed a prototype implementation of the described method and applied our contour model to computer-generated test images as well as to clinical images. In this section, we will present some of the results.

The first example in Fig. 10 shows the results of applying the contour model to a noisy synthetic image (256 × 256 pixels; eight bits). The shape of a cross, formed by two rectangles which intersect at a right angle, is visible through the noise in the image in Fig. 10(a). The average gray value in the background is 130, while in the "legs" of the cross it is 160, and in the center of the cross 190. The standard deviation of the noise is 32. Fig. 10(b) gives the histogram of gray values for the pixels inside the elliptical region in Fig. 10(a). It will be clear from this histogram that simple thresholding can not be used to separate the cross from the background.

Finding some approximation of the contour of both rectangles at the same time (the complete cross) may still be possible by using conventional approaches like blurring and thresholding combined with morphological operations. It will, however, require insight in the situation and the applied techniques as well as significant guidance from the operator. Finding the contour of just one of the rectangles is more difficult because of the change in gray value where both rectangles cross each other. We will show that our contour model can be used to extract the contour of one of the rectangles in different user friendly ways and with a minimum of interaction.

Fig. 10(c) shows the external energy distribution which will be used to guide the contour model. In this case, it represents the reciprocal magnitude of the gray value gradient after applying a Gaussian blurring with $\sigma = 5$ (pixel units). The contour model, which will attempt to follow the valleys in the external energy distribution, will thus be pulled toward maximum gradient values.

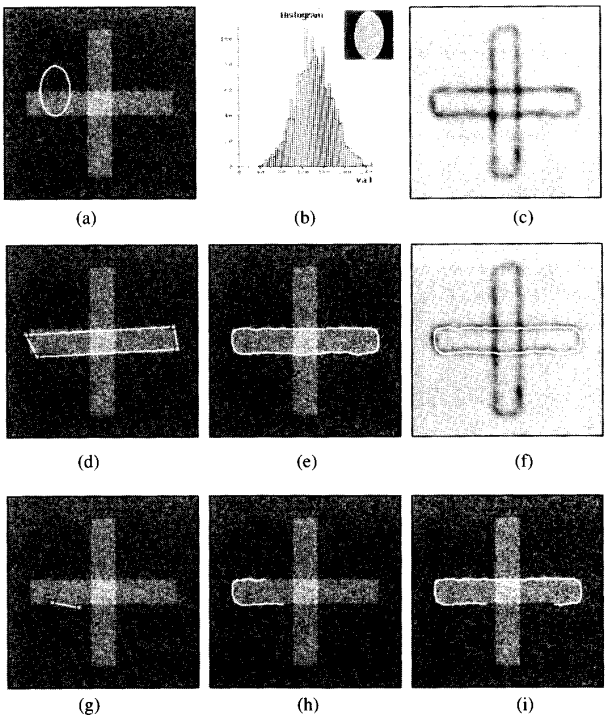

Fig. 10. (a) Computer-generated image of cross, with relatively high noise level and elliptical ROI for analysis of pixel values. (b) Histogram of gray values of the pixels inside the elliptical ROI. (c) External energy distribution for defining contours along edges. (d) Initial contour as drawn by the operator. (e) Resulting contour after deformation and resampling of the initial contour in (d), shown on image. (f) Resulting contour shown on external energy distribution. (g) Initial open contour as drawn by the operator. (h) Open contour model tracking along max gradient path. (i) Open contour model tracking along max gradient path.

In Fig. 10(d), we see an initial contour as it was drawn in the image by an operator. It consists of only four points which were deliberately positioned very roughly around the rectangular shape. Fig. 10(e) gives the final result of the deformation and resampling process of the contour model. The specified resolution of the model was, in this case, $l_{\text{des}} = 4$ (pixel units); the weighting factors w_{ex}, w_{in}, and w_{damp} were all equal to 0.5. In Fig. 10(f), it is shown that the contour model has settled itself along a path of local minima in the external energy distribution as was intended.

A second way of applying our model is illustrated in Fig. 10(g)–(i). An *open* contour is now used for initial contour as shown in Fig. 10(g). User interaction is now limited to positioning only two points roughly near the desired path of the contour. This open contour model can automatically start growing (clockwise direction), while, at the same time, it will be resampled and deformed, finding its path without any further user intervention as described in Section III-A. In Fig. 10(h), we see that it has rounded the first two corners. In Fig. 10(i), it has passed the crossing and rounded the next corners. The growing process will come to a stop when the two ends of the contour get closer than the distance $l_{\max}$, as

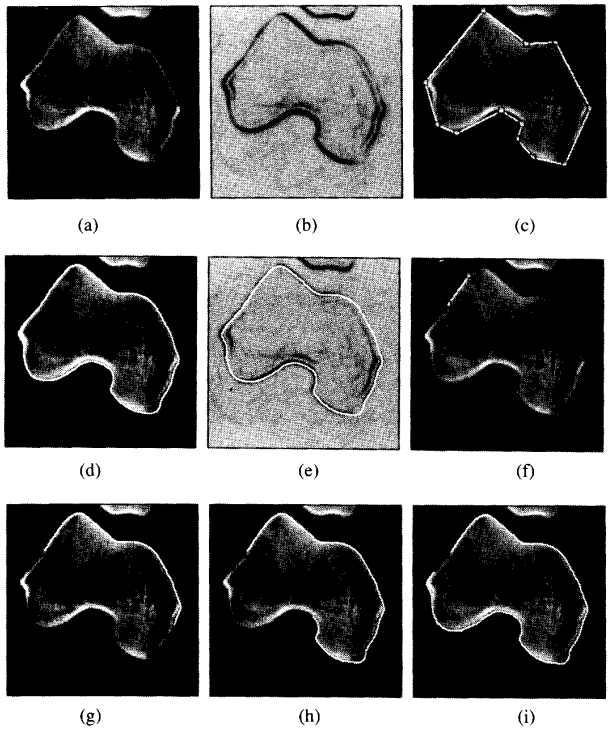

Fig. 11. (a) CT image through the knee showing cross-section through lower part of femur. (b) External energy distribution for defining contours along edges. (c) Initial contour for defining the bone edge, as drawn by the operator. (d) Resulting contour after deformation and resampling of the initial contour in (c), shown on image. (e) Resulting contour shown on external energy distribution. (f) Initial open contour as drawn by the operator. (g) Open contour model tracking along max gradient path. (h) Open contour model tracking along max gradient path. (i) Open contour model tracking along max gradient path.

described in the previous section. The model will, from then on, behave as a closed contour and come to a rest just like in the process shown in the frames of Fig. 10(d)–(f).

The next example in Fig. 11 concerns a clinical image. Fig. 11(a) is a CT scan (512 × 512 pixels; 12 b) representing a slice through the knee, showing a cross-section through the lower part of the femur, of which we want to extract the contour. The segmentation problem in this case is that the gray values locally drop to very low values inside the body of the bone, making it impossible to use thresholding without creating a large number of undesired contours. We will demonstrate the performance of our contour model using similar approaches, as in the previous example.

The external energy distribution which we will use is shown in Fig. 11(b): the reciprocal magnitude of the gray value gradient after blurring with $\sigma = 3$. The variation along the edge of the bone shows differences in "edge strength" which are more severe than in the previous example: the edge almost disappears in the lower right part of the image. Also, there are many other edges visible, some quite close to the one for which we are looking. While this situation requires some care in drawing the initial contour, just a few points suffice as shown in Fig. 11(c).

The result after deformation and resampling is presented in Fig. 11(d) on the pixel gray values, and in Fig. 11(e) on the external energy distribution. It can be seen from Fig. 11(e) that the contour correctly follows the intended edge, even in the areas where edges are weak and in the areas where other edges come close to the path. The resolution was $l_{\text{des}} = 4$, and the force balance was $w_{\text{ex}} = w_{\text{in}} = w_{\text{damp}} = 0.5$, just as in the previous example.

Fig. 11(f)–(i) illustrates the application of an open contour model to this image. Starting from the initial contour in Fig. 11(f), it grows again in a clockwise direction, tracking a path of local minima through the external energy distribution. Fig. 11(g)–(i) shows different stages in this process.

Fig. 12 shows an example of a typical intravascular ultrasound image. This image represents a digitized (256 × 256 pixels; eight-bit) cross-section through a blood vessel, scanned by a full 2π radians rotation of the transducer which is mounted on the tip of a catheter.

A number of more or less elliptical shapes are visible in the image, representing the layered structure of the vessel wall. Some of these elliptical shapes show up as bright ridges, while others show up as dark valleys. Both types are interesting from a clinical point of view. The area inside the vessel is partly blocked. The lumen (remaining open area) is visible as a dark region. Describing the edge of this open area with our model is also a meaningful application.

There are a number of reasons why it would be very difficult, if at all possible, to extract the desired contours from this image using conventional (local) edge extraction methods. As is usual in ultrasound images, the signal-to-noise ratio is very low and large parts of the image are obscured by shadows. An additional difficulty is caused by the presence of the catheter (in the lower part of the lumen, touching the vessel wall) and the overlaid artificial tickmarks for distance measurement in the horizontal and vertical directions. We will show that our contour model succeeds well in defining the contours which describe the bright elliptical shape in the vessel wall, as well as the edge of the open area inside the vessel.

First, we will define a contour describing the edge of the lumen. The obvious choice so as to position the desired path of this contour between dark and bright image regions is, as in the previous examples, to make it follow a path of maximum gray value gradient. As the contour model will follow valleys through the external energy distribution, the reciprocal length of the blurred (Gaussian, $\sigma = 3$) gray value gradient is the appropriate choice for this distribution. The resulting external energy is shown in Fig. 12(b). Parts of the edge of the open area now show up as dark valleys, which do not, however, form a closed shape. The initial model created by the operator, using only five vertex positions, is shown in Fig. 12(c) and the result after deformation is presented in Fig. 12(d), together with the original gray values; and in Fig. 12(e) with the external energy distribution.

Fig. 12(f)–(h) demonstrates that, even in this difficult case, it is possible to further reduce the required user interaction by applying an open contour to automatically track the desired edge. For defining this edge, we used the following parameter settings: $l_{\text{des}} = 8$ and again $w_{\text{ex}} = w_{\text{in}} = w_{\text{damp}} = 0.5$.

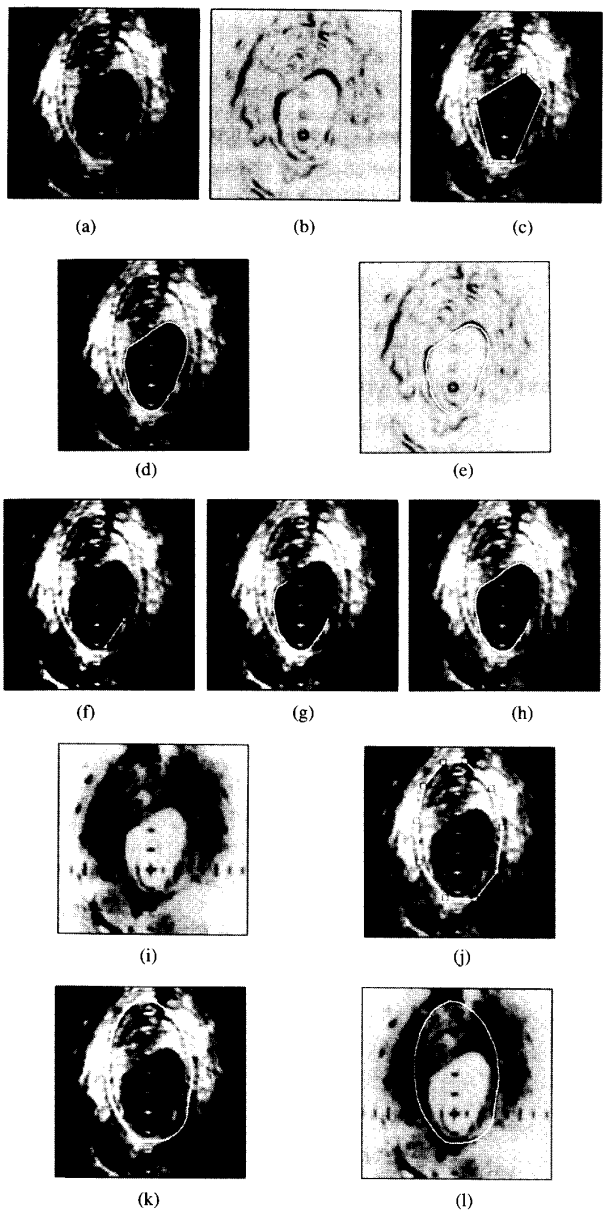

Fig. 12. (a) Intravascular ultrasound image showing cross-section through vessel. (b) External energy distribution for defining contours along edges. (c) Initial contour for defining the edge of the lumen, as drawn by the operator. (d) Resulting contour after deformation and resampling of the initial contour in (c), shown on image. (e) Resulting contour shown on external energy distribution. (f) Initial open contour as drawn by the operator. (g) Open contour model tracking along max gradient path. (h) Open contour model tracking along max gradient path. (i) External energy distribution for defining contours along ridges. (j) Initial contour for defining the shape of the relatively bright elliptical structure in the vessel wall, as drawn by the operator. (k) Resulting contour after deformation and resampling of the initial contour in (j), shown on image. (l) Resulting contour shown on external energy distribution.

Next, we will define a contour through the bright elliptical shape in the vessel wall. As this shape forms a kind of ridge through the image, and the model will try to follow valleys, we simply can use the inverse of the blurred (Gaussian, $\sigma = 3$) pixel gray value as the external energy distribution; the result

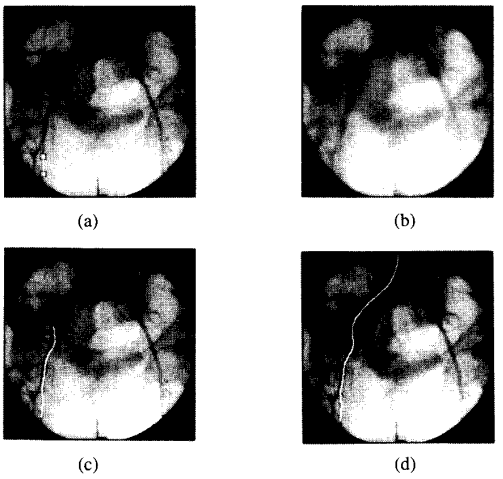

Fig. 13. (a) Vascular X-ray image showing main arteries in abdomen and legs and the initial open contour as drawn by the operator. (b) External energy distribution for defining contours along valleys. (c) Open contour model tracking along valley. (d) Final shape of open contour model.

is displayed in Fig. 12(i). An initial contour shape is defined by the operator by pointing out a small number of positions, as shown in Fig. 12(j). After the deformation process has stopped, the resulting shape of the model is presented in Fig. 12(k) together with the original gray values, and in Fig. 12(l) with the external energy distribution. The contour model follows the gray value ridge in a natural way. In those areas where the external energy is not strong enough or even absent, the shape of the model is mainly determined by internal forces, bridging the gaps in an equally natural way. The parameter settings were the same as for the definition of the edge of the lumen.

The next example illustrates the definition of an open contour following a valley in the image. It concerns an X-ray image (512 × 512 pixels; 10 b) for vascular application in which the main arteries in abdomen and legs are visualized by injection of a contrast medium so that these vessels show up in the image as dark valleys. The image is displayed in Fig. 13(a), together with the initial contour as defined by the operator. In this case, our goal is to follow the main arteries using an open contour model. As the vessels are already visible as dark valleys, a suitable external energy distribution is obtained by only blurring the image with $\sigma = 5$, as shown in Fig. 13(b). Fig. 13(c) and (d) shows different stages in the resulting open contour model. The parameter settings were in this case: $l_{\text{des}} = 10$ and $w_{\text{ex}} = w_{\text{in}} = w_{\text{damp}} = 0.5$.

The last clinical example we want to show concerns an MR image of the head (256 × 256 pixels; eight bits), shown in Fig. 14(a). This image represents a slice through the head at eye level. The dark area just left of the center is a tumor, which is partly located in the brain tissue (gray) and partly in the bony structures of the skull base (black area to the left of the tumor). This is a particularly difficult case for segmentation because different contours may be interesting from a clinical point of view. We will show that, also in this situation, a dynamic contour model like ours may be useful, by demonstrating that different meaningful results can be obtained

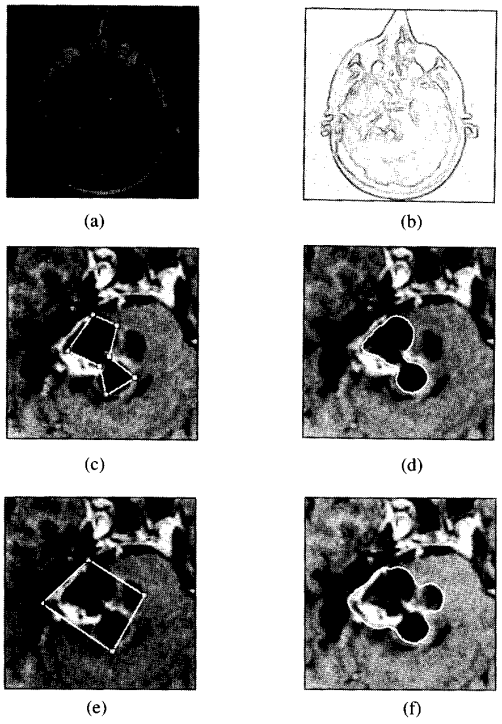

Fig. 14. (a) MR image through the head showing tumor left of center. (b) External energy distribution for defining contours along edges. (c) First initial contour as drawn by the operator. (d) Resulting contour after deformation and resampling of the initial contour in (c), shown on image. (e) Different initial contour as drawn by the operator. (f) Different resulting contour after deformation and resampling of the initial contour in (e), shown on image.

in a user friendly and predictable way by simply defining different initial contours.

The external energy distribution is calculated from the gradient magnitude as explained before, and shown in Fig. 14(b). In this case, we used $\sigma = 1$ for blurring because the resolution of the image is already relatively low and we want to avoid blurring the edges too much. There are obviously different edges which can be used to position the contour model in the area of the tumor. By being reasonably careful in positioning the initial contour, we can control what the final shape of the contour will be.

If we want to delineate the part of the tumor which shows up in the image as two more or less circular connected dark areas, we can use an initial contour like the one in Fig. 14(c) which produces the result as shown in Fig. 14(d).

If, however, we are interested in a larger area, and want to include the bright area on the left side of the tumor and the dark-gray area on the right, the simple initial contour in Fig. 14(e) will lead to the desired result as shown in Fig. 14(f).

For both results, we used the parameter settings: $l_{\text{des}} = 4$ and $w_{\text{ex}} = w_{\text{in}} = w_{\text{damp}} = 0.5$.

The previous example may raise the question of how accurately one has to draw the initial contour to get the desired and clinically meaningful result. The answer to this question greatly depends on the actual situation. In the example of Fig. 14, there are different possible paths close together. As the contour model does not have any knowledge of the pathology, it will not have a preference for any particular path. The expert knowledge of the operator is required to accurately position the initial contour. Visual feedback can be used to decide whether the result is satisfying; if this is not the case, the resulting contour can be adjusted, after which the deformation process is restarted. In the examples shown, no adjustment has been applied. The initial contours yielded the results without interference from the operator, using the specified parameter settings. In relation with the above, it is useful to briefly discuss the speed of the deformation process. There are two important factors which determine the time it takes for the contour model to come to a rest. One obvious factor is the number of vertices in the model, determined by the resolution, which can be set by the operator. Another factor is the number of steps it takes to reach the final shape starting from the initial outline, which depends on the difference between initial and final contour. The operator has a large influence here, as he can decide to put more or less effort in drawing initial contours. Accordingly, general statements about the speed are not very meaningful. We can say, however, that in all the cases demonstrated here, the deformation process took a fraction of a second on a SUN-IPX workstation. The number of vertices was, in all cases, less than 100.

We next discuss the sensitivity of the final result for the shape of the initial contour. In other words, how reproducible is the result when starting from different initial contours?

The cross image of Fig. 10 is used again to demonstrate that the shape of the initial contour generally has little influence on the result. Fig. 15 outlines our experiments. Frames (a), (d), and (g) of Fig. 15 show three quite different initial contours. The results after deformation are shown in Fig. 15(c), (f), and (i), intermediate shapes in Fig. 15(b), (e), and (h). It can be seen that the results do not depend strongly on the initial contour location, which is actually one of the reasons for using a deformable contour model opposed to drawing contours manually. The discrete nature of the model leads to a certain variation in the positions of the individual vertices, when the deformation process is started from different initial positions. However, the variations in the path of the contour are small as compared with the resolution of the contour. All parameter settings were the same as specified before for the results of Fig. 10.

As can be seen from the examples, the operator-defined initial model may have a very low resolution without jeopardizing the contour finding process. The initial resolution is automatically and quickly increased by the resampling mechanism, which is integrated in the deformation process, until it is on the level that was specified via the parameter l_{des}. This will be realized after only a few deformation steps, and from there on the resampling process will keep the resolution at the required level. Consequently, the amount of user interaction in defining an initial model can be very small. In a number of well-defined applications, it seems feasible to generate the initial contour without user interaction, thereby making the entire method automatic and reproducible.

At any point the deformation process can be stopped, which offers the user the opportunity to change any parameter, after

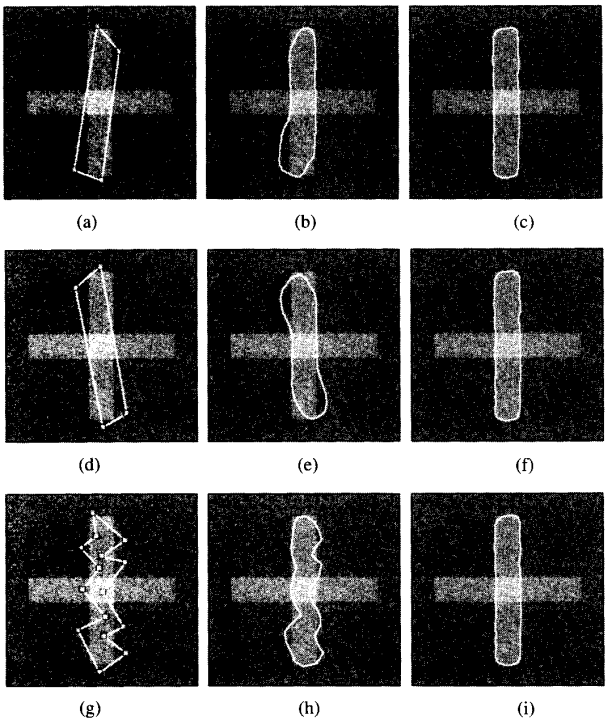

Fig. 15. (a) Initial contour on synthetic cross image. (b) Intermediate result of deformation and resampling of the initial contour in (a). (c) Final result of deformation and resampling of the contour in (a). (d) Different initial contour on same image. (e) Intermediate result of deformation and resampling of the initial contour in (d). (f) Final result of deformation and resampling of the contour in (d). (g) Much different initial contour on same image. (h) Intermediate result of deformation and resampling of the initial contour in (g). (i) Final result of deformation and resampling of the contour in (g).

which the deformation can be resumed. This option was not used in the examples discussed.

Optimization of the weight factors for a particular application can be done interactively, by judging the result visually. Our experience shows that, after having determined the desired parameter settings, these settings can be applied to similar situations (same modality, same acquisition parameters, same goal) without modification, which offers the possibility to build a collection of typical parameter settings from which a user could make a choice in frequently occurring situations.

Setting the parameters will, in most cases, be restricted to setting σ, the width of the blurring function, which determines the scale of the image, and setting l_{des}, the desired distance between vertices which determines the resolution of the contour model. The force balance, determined by w_{ex}, w_{in}, and w_{damp}, was the same for all the discussed examples. Although the images which we used were quite different, this was possible because of a constant scaling which we applied to the calculated external energy distribution such that minimum and maximum values were equal in all cases.

V. Concluding Remarks

We have developed a discrete contour model that combines conceptual and computational simplicity with variable resolution and adjustable behavior. It is based on a simple structure and its deformation is controlled by basic physical rules. It incorporates elegant and efficient solutions to the shrinking and clustering problems from which active model approaches suffer. The method is easy to use because of the small number of parameters that control the deformation process. The examples show that our discrete contour model has potential in the area of contour definition for a large variety of clinical applications.

As compared with conventional contour extraction methods, the strong argument in favor of the method presented here is that it handles and processes a contour as one topologically consistent object, yet the deformation process which controls the shape of the model is based on local operations on the vertices of the model.

As compared with manual contour definition methods, the advantage of the present approach is the minimum of user interaction which is required, and the reproducibility of the result.

A sensitive point of our method is the dependency of the final result on the image features that are used to drive the contour deformation. The method relies on the assumption that some process exists for the extraction of adequate image features in a particular applicational context. Extraction of useful image features is therefore an important subject for further investigation. Its importance for the success of our contour model should not be overestimated though, because simple image features like gray value and gray value gradient were found to be quite appropriate in a variety of situations.

Future research will go in the direction of selection of appropriate image features for specific applications, adaptivity of the model to local image context, and extension of the method to 3-D images.

References

[1] M. Alonso and E. J. Finn, *Fundamental University Physics*. Reading, MA: Addison-Wesley, 1968.
[2] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer, "Elastically deformable models," *Comput. Graphics*, vol. 21, no. 4, pp. 205–214, 1987.
[3] A. Witkin, K. Fleischer, and A. Barr, "Energy constraints on parameterized models," *Comput. Graphics*, vol. 21, no. 4, pp. 225–232, 1987.
[4] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *Int. J. Comput. Vision*, pp. 321–331, 1988.
[5] J. Eilbert, C. Gallistel, and D. McEachron, "The variation in user drawn outlines on digital images: Effects on quantitative autoradiography," *Comput. Med. Imag. Graphics*, vol. 14, no. 5, pp. 331–339, 1990.
[6] J. V. Miller, D. E. Breen, and M. J. Wozny, "Extracting geometric models through constraint minimization," Rensselaer Polytechnic Institute, Tech. Rep. No. 90024, 1990.
[7] ——, "Extracting closed geometric models of two-dimensional cavities with deformable meshes," Rensselaer Polytechnic Institute, Tech. Rep. No. 90025, 1990.
[8] J. V. Miller, "On GDM's: Geometrically deformed models for the extraction of closed shapes from volume data," Rensselaer Polytechnic Institute, Tech. Rep. No. 90051, 1990.
[9] D. Terzopoulos and M. Vasilescu, "Adaptive surface reconstruction," *SPIE*, vol. 1383, pp. 257–264, 1990.
[10] A. Witkin and W. Welch, "Fast animation and control of nonrigid structures," *Comput. Graphics*, vol. 24, no. 2, pp. 243–252, 1990.
[11] J. V. Miller, D. E. Breen, W. E. Lorensen, R. M. O'Bara, and M. J. Wozny, "Geometrically deformed models: A method to extract closed geometric models from volume data," *Comput. Graphics*, vol. 25, no. 4, pp. 217–226, 1991.

[12] Y. F. Wang, J. Lee, and J. Wang, "Unification scheme for 3-D surface reconstruction using physically based models," *Int. J. Imag. Syst. Technol.*, vol. 3, pp. 279–299, 1991.

[13] L. M. J. Florack, B. M ter Haar Romeny, J. J. Koenderink, and M. A. Viergever, "On scale and differential structure of images," *Image Vision Computing*, vol. 10, no. 4, pp. 567–588, 1992.

[14] M. E. Hyche, N. F. Ezquerra, and R. Mullick, "Spatiotemporal detection of arterial structure using active contours," *SPIE, Visualization in Biomedical Computing*, vol. 180, pp. 52–62, 1992.

[15] L. H. Staib and J. S. Duncan, "Deformable Fourier models for surface finding in 3-D images," *SPIE, Visualization in Biomedical Computing*, vol. 1808, pp. 90–104, 1992.

[16] Y. F. Wang and J. Wang, "Surface reconstruction using deformable models with interior boundary constraints," *IEEE Pattern Anal. Mach. Intell.*, vol. 14, no. 5, pp. 572–579, 1992.

[17] D. J. Williams and M. Shah, "A fast algorithm for active contours and curvature estimation," *CVGIP: Image Understanding*, vol. 55, no. 1, pp. 14–26, 1992.

[18] L. M. J. Florack, B. M ter Haar Romeny, J. J. Koenderink, and M. A. Viergever, "Cartesian differential invariants," *J. Math. Imag. Vision*, vol. 3, no. 4, pp. 327–348, 1993.

[19] B. M. ter Haar Romeny and L. M. J. Florack, "A multiscale geometric model of human vision," in *The Perception of Visual Information*, W. R. Hendee and P. N. T. Wells, Eds. New York: Springer-Verlag, 1993, pp. 73–114.