ELSEVIER

Medical Image Analysis 4 (2000) 389–402

MEDICAL IMAGE ANALYSIS

www.elsevier.com/locate/media

A 3D generalization of user-steered live-wire segmentation

Alexandre X. Falcão[a], Jayaram K. Udupa[b,*]

[a]*Institute of Computing, State University of Campinas, Campinas SP, Brazil 13083-970*
[b]*Medical Image Processing Group, Department of Radiology, University of Pennsylvania, Philadelphia, PA 19104-6021, USA*

Abstract

We have been developing user-steered image segmentation methods for situations which require considerable human assistance in object definition. In the past, we have presented two paradigms, referred to as *live-wire* and *live-lane*, for segmenting 2D/3D/4D object boundaries in a slice-by-slice fashion, and demonstrated that live-wire and live-lane are more repeatable, with a statistical significance level of $P < 0.03$, and are 1.5–2.5 times faster, with a statistical significance level of $P < 0.02$, than manual tracing. In this paper, we introduce a 3D generalization of the live-wire approach for segmenting 3D/4D object boundaries which further reduces the time spent by the user in segmentation. In a 2D live-wire, given a slice, for two specified points (pixel vertices) on the boundary of the object, the best boundary segment is the minimum-cost path between the two points, described as a set of oriented pixel edges. This segment is found via Dijkstra's algorithm as the user anchors the first point and moves the cursor to indicate the second point. A complete 2D boundary is identified as a set of consecutive boundary segments forming a ''closed'', ''connected'', ''oriented'' contour. The strategy of the 3D extension is that, first, users specify contours via live-wiring on a few slices that are orthogonal to the natural slices of the original scene. If these slices are selected strategically, then we have a sufficient number of points on the 3D boundary of the object to subsequently trace optimum boundary segments automatically in all natural slices of the 3D scene. A 3D object boundary may define multiple 2D boundaries per slice. The points on each 2D boundary form an ordered set such that when the best boundary segment is computed between each pair of consecutive points, a closed, connected, oriented boundary results. The ordered set of points on each 2D boundary is found from the way the users select the orthogonal slices. Based on several validation studies involving segmentation of the bones of the foot in MR images, we found that the 3D extension of live-wire is more repeatable, with a statistical significance level of $P < 0.0001$, and 2–6 times faster, with a statistical significance level of $P < 0.01$, than the 2D live-wire method, and 3–15 times faster than manual tracing.

Keywords: Image segmentation; Boundary detection; Shortest-path algorithms; Optimal graph searching; Active boundaries; 3D imaging

1. Introduction

Despite nearly 40 years of progress, image segmentation remains one of the major challenges in imaging science (Haralick and Shapiro, 1985). In any application, particularly medical, which is the focus of our research, delineation of objects in images is a prerequisite to their effective visualization, manipulation and analysis. Even image processing operations other than these depend on the availability of object information for their effectiveness. There are repeated evidences in the literature as to how object information improves image filtering (Gerig et al., 1992), interpolation (Raya and Udupa, 1990) and registration (Maintz et al., 1996). Thus, segmentation is one of the most fundamental image processing operations.

Segmentation consists of two tightly coupled tasks – *recognition* and *delineation*. Recognition is the process of identifying roughly the whereabouts of a particular object of interest and distinguishing it from other objects that are invariably present in the image. Delineation is the process of specifying the precise spatial extent (and composition, if needed, in case of heterogeneity) of the object. Considering the possible variation in the degree of automation in recognition and delineation, existing segmentation methods cover a wide spectrum from completely manual recognition and delineation, as in manual boundary tracing or painting of regions, to completely automatic recognition and delineation, as in thresholding (Prewitt and Mendelsohn, 1966) wherein an image intensity value is considered to be adequate to recognize and delineate the object.

*Corresponding author. Tel.: +1-215-662-6780; fax: +1-215-898-9145.

E-mail address: jay@mipg.upenn.edu (J.K. Udupa).

1361-8415/00/$ – see front matter
PII: S1361-8415(00)00023-2

A particular application together with a particular image acquisition protocol defines a segmentation situation. Although many theories and algorithms have been proposed for segmentation, to make a particular general technique work effectively in a given application typically requires considerable further research and development. This is usually true even when the segmentation situation changes slightly, as for example, when magnetic resonance (MR) images are obtained for the same body region using different protocols. From the application view point, therefore, developing general methodologies that can be quickly adapted to a given application and that offer acceptable precision, accuracy and efficiency of segmentation is undoubtedly a worthwhile goal.

With this similar overall goal, and for routine use in applications involving a large number of data sets, and with the requirement of no application-tailored modifications of the method, we have been developing user-steered segmentation strategies with two specific aims: (i) to provide as *complete* a *control* as possible to the user on the segmentation process *while* it is being executed, and (ii) to minimize the user involvement and the total user's time required for segmentation. The aspect of complete user control, while the process is underway, guarantees the success of segmentation as determined by the expert user. The requirement of minimizing the user's time makes the segmentation method viable in practical situations. Our strategy in these methods has been to actively exploit the synergy between the superior abilities of human operators (compared to computer algorithms) in object recognition and the superior abilities of computer algorithms (compared to human operators) in object delineation.

Toward this goal, a class of approaches has evolved aiming mainly at rather difficult segmentation situations. These are commonly known as active boundary approaches (Kass et al., 1987; Cohen, 1991; McInerney and Terzopoulos, 1996; Park et al., 1996; Amini et al., 1997). The mode of operation here consists of the specification of an initial boundary (2D or 3D) by the user. This offers recognition help to the method. This boundary is subsequently deformed under external and internal forces, with true boundary locations ideally exerting large attractive forces. Under assumed stiffness properties of the boundary, the boundary with the minimum energy is considered to be the one we are seeking.

In the past, we have presented two user-steered segmentation paradigms, referred to as *live-wire* and *live-lane* (Falcão et al., 1998). Although the live-wire method has its origin in some early joint work between Barrett and Udupa (Morse et al., 1991; Mortensen et al., 1992; Udupa et al., 1992), this method has been subsequently developed independently by the two groups (Mortensen and Barrett, 1995, 1998; Barrett and Mortensen, 1996, 1997; Falcão et al., 1996, 1998; Falcão and Udupa, 1997). There are many differences between the live-wire method developed by each group, as previously explained by Falcão et al. (1998). Besides these differences, we have extended the ideas underlying the live-wire method to create new methods, live-lane (Falcão et al., 1998) and the method presented in this paper.[1] Live-wire and live-lane methods segment 2D/3D/4D object boundaries in a slice-by-slice fashion. Based on over 2000 tracings in one application (defining the bone boundaries in the tarsal joint complex of the foot via MRI), we have found that these methods are more repeatable (precision), with a statistical significance level of $P < 0.03$, and 1.5–2.5 times faster (efficiency), with a statistical significance level of $P < 0.02$, than the manual method. The methods are in routine use in several applications (Hirsch et al., 1996; Rhoad et al., 1998; Udupa et al., 1998; Stindel et al., 1999) with over 15 000 tracings done so far.

The live-paradigms presented by Falcão et al. (1998) are fundamentally different from active boundary approaches in four ways. (1) They consider a slice as a weighted and directed graph with the vertices of the pixels representing the nodes of the graph and the oriented pixel edges representing the arcs. Each oriented pixel edge in the slice is a potential boundary element *b*, which is called a bel for short. Every bel $b = (p, q)$ has a location and an orientation. The location of *b* is that of the unique edge shared by the two four-adjacent pixels *p* and *q*. We assume, without loss of generality, that its orientation is such that *p* is always ''inside'' the boundary, *q* is ''outside'' the boundary, and the ''inside'' is always to the left of *b* (see Fig. 1). Clearly, any bel should be in one of four orientations as shown in Fig. 1. To each bel *b*, we assign a set of features whose values characterize the ''boundariness'' of *b*. These values are converted to a single joint cost value per bel *b*. These costs are usually significantly different for two oppositely oriented bels between the same pair of 4-adjacent pixels. This allows these methods to discriminate between different boundaries that come close together

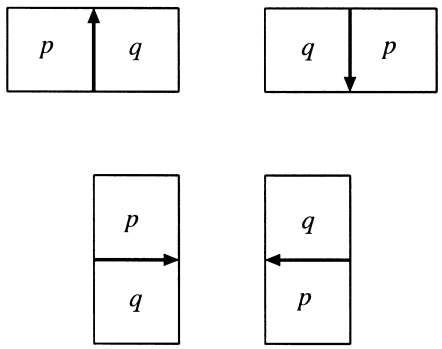

Fig. 1. A boundary element (bel) is an oriented pixel edge. The four possible types of bels in a slice are shown. The ''inside'' of the boundary is to the left of the bel and the ''outside'' is to the right.

[1] A preliminary version of this work was previously presented in a conference paper (Falcão and Udupa, 1997).

which otherwise have similar properties. Exploiting the orientedness of the boundaries for improving segmentation by explicitly building this property into the model has not been done to date in the published boundary detection literature that we are aware of. (2) They consider a 2D boundary to be a connected, oriented, closed contour made up of bels. (3) Boundaries are found not by minimizing total energy but by minimizing cost piecewise. The user initially picks a point (a pixel vertex) on the boundary and all possible (globally) minimum cost segments from this point to all other points in the image are computed via Dijkstra's algorithm (Cormen et al., 1991). At this time a ''live-wire'' is displayed in real time from the initial point to any subsequent position of the cursor. If the cursor is close to the desired boundary, the live-wire snaps on to the boundary even in the presence of distractions from nearby boundaries. The cursor is now deposited and a new live-wire segment is found next. Fig. 2 illustrates the behavior of live-wire in three different situations: delineating a boundary in the presence of other boundaries with similar properties (Fig. 2(a)), jumping gaps (Fig. 2(b)), and passing through noisy regions (Fig. 2(c)). Typically, 2–5 points selected in this fashion are enough to segment the whole boundary. Within a single implementation, the live-paradigms offer a continuum between totally manual tracing and near-automatic tracing. (4) The efficacy of user-controlled boundary finding methods is very much application dependent. The precision, accuracy and efficiency of live-methods have been thoroughly evaluated for the applications in which they are currently used. Such statistical evaluations, especially efficiency (speed improvement) over manual tracing, which tacitly characterize the behavior of the algorithm for practical use, do not seem to have been carried out for any published active contour or other boundary detection methods.

In this paper, we present an extension of the live-wire method to segment 3D boundaries so as to further reduce user involvement and at the same time not to compromise precision and accuracy. In this method, the user initially divides the set of natural slices into slabs of constant object topology. In each slab, the 3D object may be represented by one or more structures (3D connected objects) such that each structure defines one and only one 2D boundary in any given slice of the 3D data set. The user then selects a structure and specifies contours via live-wiring the 2D boundary of the structure on a few slices that are orthogonal to the natural slices of the 3D scene. This process is repeated for other structures of the 3D object. If the orthogonal slices are selected strategically, then we get a sufficient number of points on the 3D boundary of each structure to trace optimum boundary segments automatically in all natural slices of the 3D data set. Given a natural slice and a 2D boundary of interest in this slice, the set of points found as above on this boundary has an order of processing such that when the best boundary segment between each pair of consecutive points in the ordered set is computed, a closed, connected, oriented contour results. The order of the points on each 2D boundary is found from the way the users choose the orthogonal slices. This boundary detection process is repeated for each 2D boundary of each structure in each natural slice of the data set and so the delineation of the 3D object boundary subsequently becomes a highly automatic process. At this point, we would like to comment on some observations. First, slab definition may seem difficult and nonintuitive for complex-shaped objects. In fact, the efficiency of the

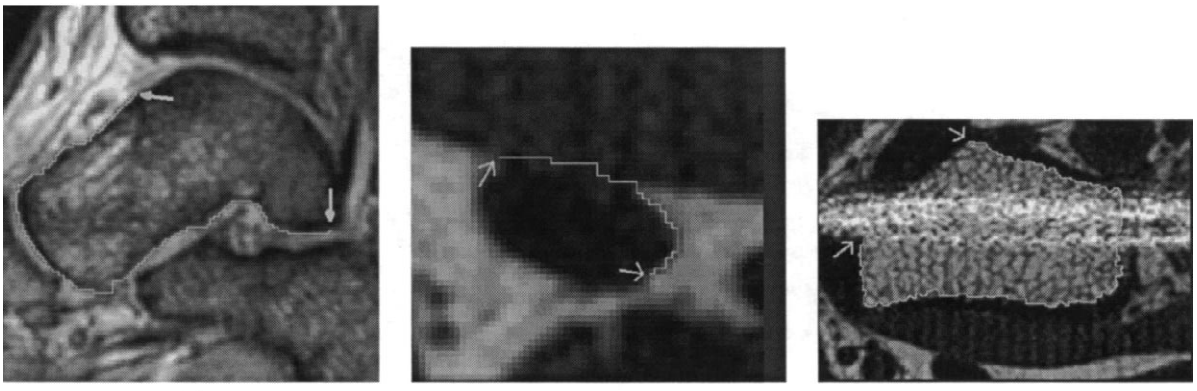

Fig. 2. Illustration of live-wire tracing in three different situations, all via MRI. (a) Delineating a boundary in the presence of other boundaries with similar properties. The boundary of interest is that of the bone talus in the foot (see Fig. 6 for context). Cortical bones appear as dark bands. Talus is close to three other bones: the navicular at the lower left, the tibia at the top, and the calcaneus at the lower right. The boundaries traced here are assumed to have an anti-clockwise orientation. Due to the orientedness property, the boundaries of the navicular, the tibia and the calcaneus are actually in an opposite direction in the parts where they come close to the talus boundary and therefore repell the talus boundary. (b) Jumping gaps. An MR image of a wrist, the object of interest being a vessel. (c) Passing through noisy regions. An MR image of a wrist with the internal boundary of bone being the boundary of interest. Note how the optimum boundary is not deterred by the horizontal noisy streaks present in the image. In (a)–(c), one anti-clockwise oriented boundary segment found as a globally optimal path between the two points indicated by the arrows is shown.

method is inversely related to the number of slabs. However, most objects that we have encountered for which more automatic methods fail are not so complex as to make it impossible to specify slabs of constant topology. Second, to maximize efficiency, it would be ideal if an interactive method did not require user's corrections at the end of the segmentation process. On the other hand, to provide complete user control, the method should allow the user to make corrections during and after the segmentation process. The main idea in 3D live-wire is that, for a given application, where the same object has to be segmented in a large number of 3D data sets, if the user selects orthogonal slices strategically, the 3D boundary of the object can be subsequently detected automatically in all 3D data sets, with no need for further user's corrections. Third, unfortunately there is no rule to select orthogonal slices. For a given application involving many 3D data sets, the user has to spend some minutes planning these slices. In our experience, this is not only possible in many applications but also saves a considerable amount of the user's time for segmentation.

Live wire has been used in 3D to define interactive volume cutting in the 3D space (Wong, 1998). However, the method presented in the present paper is the first attempt to extend live-wire for 3D object segmentation. The use of graph searching techniques in boundary finding has also been investigated for many years now (Martelli, 1971; Cappelletti and Rosenfeld, 1989). Perhaps due to the lack of computer power, these early techniques tried to simplify the problem by using heuristics. This resulted in a graph that is deficient in many respects which compromised their effectiveness in segmentation. They also put severe restrictions on the form and shape of 3D objects – circular and smooth (Cappelletti and Rosenfeld, 1989). In this sense, the work presented here and the work reported in (Falcão et al., 1998) not only bring about a very general use of graph searching techniques in finding boundaries, but also demonstrate that it is possible to solve the problem optimally (although piecewise), without heuristics or other restrictions that would have made these methods application-dependent.

In the next section, we describe the 3D extension of live-wire. In Section 3, we assess comparatively the utility of 2D live-wire and 3D live-wire methods based on speed and repeatability of segmentation. We state our conclusions and outline some further directions in Section 4.

2. 3D live-wire

Let a 3D cuboid region of finite size in 3D Euclidean space be subdivided into small cuboids by three mutually orthogonal families of parallel and equal unit-spacing planes. Assume that to each small cuboid is assigned an intensity value lying in an interval $[L,H]$. We call the small cuboids *voxels*, the set of all voxels in the cuboid region together with their intensity values a *3D scene*, and the set of voxels the *3D scene domain*. The intersection between any plane in 3D Euclidean space and the 3D scene defines a 2D scene. Thus, we think of a 3D scene in a Cartesian coordinate system (x,y,z) as a set $\boldsymbol{C}^{(m)} = \{\boldsymbol{C}_1, \boldsymbol{C}_2, \ldots, \boldsymbol{C}_m\}$ of m consecutive and parallel 2D scenes $\boldsymbol{C}_i$ in the xy-plane, called *natural slices*, for each fixed z coordinate. We define a *3D object* O of interest as a single connected set of voxels in the 3D scene. Given a natural slice $\boldsymbol{C}_i$, $1 \leqslant i \leqslant m$, of $\boldsymbol{C}^{(m)}$, the boundary of the 3D object in $\boldsymbol{C}_i$ may be represented by a set of 2D boundaries (each being a closed, connected and oriented contour). To segment a 3D object using the 2D live-wire method, the user has to trace each 2D boundary in each natural slice $\boldsymbol{C}_i$ of $\boldsymbol{C}^{(m)}$ as we described in (Falcão et al., 1998) and in Section 1.

Given a 3D scene $\boldsymbol{C}^{(m)}$ and a 3D object of interest in $\boldsymbol{C}^{(m)}$, any cross section perpendicular to the natural slices of $\boldsymbol{C}^{(m)}$ defines a 2D scene $\boldsymbol{C}_o$, called an *orthogonal slice*. The boundary of a 3D object in $\boldsymbol{C}_o$ may be represented by multiple 2D boundaries. Costs are assigned to the oriented pixel edges in any given natural slice $\boldsymbol{C}_i$, $i = 1, 2, \ldots, m$, as well as in any given orthogonal slice $\boldsymbol{C}_o$ as described in (Falcão et al., 1998). The main idea in 3D live-wire is that if the user specifies 2D boundaries of the 3D object in a few orthogonal slices using the 2D live-wire method, and if these slices are selected strategically, then we have a sufficient number of points on the boundary of the 3D object, on each natural slice, to trace optimum boundary segments automatically in all natural slices $\boldsymbol{C}_i$, $i = 1, 2, \ldots, m$, of $\boldsymbol{C}^{(m)}$. This idea requires the solution of two problems:

- P_1 *Object topology.* In each natural slice, the 3D object may be represented by multiple connected 2D boundaries. To trace these 2D boundaries using the points which fall in a given natural slice, we should be able to identify the set of points that belong to each connected 2D boundary in this natural slice.
- P_2 *Ordering points.* Given a set of points on a 2D boundary in a given natural slice, we should be able to obtain the order of the points in this set such that a closed, connected and oriented contour can be obtained as a sequence of optimum boundary segments computed between each pair of consecutive points in this ordered set.

To solve the first problem, we need some concepts of topology. Since we are not developing any mathematical concepts in this paper, we will give only an intuitive description of the ideas with an example. The concepts we need are those of *internal boundaries*, *external boundaries* and of a *containment tree*. These are applicable to n-dimensional digital spaces for any $n \geqslant 2$. See (Udupa and Ajjanagadde, 1990; Udupa, 1994) for details. However, we will exemplify for the case $n = 2$, as illustrated in Fig. 3.

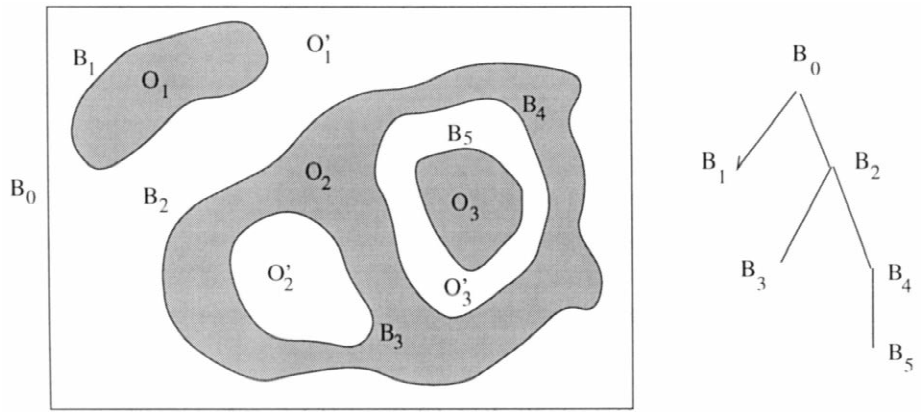

Fig. 3. A binary scene representation (left) of three objects O_1, O_2 and O_3 and its containment tree (right). $B_0, \ldots, B_5$ are the boundaries.

Suppose there are three objects O_1, O_2, O_3 in a scene domain, as in Fig. 3, wherein a binary scene representation is given. The boundary B_0 of the scene domain is considered to be an internal boundary and forms the root of the containment tree. It contains immediately inside two other boundaries B_1 and B_2, both of which are external boundaries. This is indicated by two arcs connecting B_1 and B_2 to B_0. B_1 does not contain any boundaries and hence is a leaf node. B_2 contains B_3 and B_4 (both internal boundaries) and B_4 contains B_5 (external boundary). Note that the levels in the tree alternate between internal and external. A node corresponding to an external boundary together with its children constitutes an object. For example, B_2 with B_3 and B_4 constitutes O_2. Analogously, a node corresponding to an internal boundary together with its children constitutes a background component. The purpose of all this description is to point out that a boundary can be identified with a unique component – each external boundary with an object component and each internal boundary with a background component – following a bottom-up order in the containment tree. In our example, these are as follows: B_5 with O_3, B_4 with O'_3, B_3 with O'_2, B_2 with O_2, B_1 with O_1 and B_0 with O'_1. Although we explained these concepts for 2D for ease of illustration, these are true for any higher-dimensional objects of any shape.

In practice, we are interested in forming the boundary of one connected object whose boundary may not be connected, such as the three separate components B_2, B_3, B_4 of the boundary of O_2. To solve problem P_1, we divide $\boldsymbol{C}^{(m)}$ into contiguous *slabs of constant object topology* taking the user's help, and within each slab, identify all boundaries of the 3D object O. A *slab I of constant object topology* (or *slab* for short) is a set of consecutive natural slices of $\boldsymbol{C}^{(m)}$, such that, for any two consecutive natural slices $\boldsymbol{C}_i$ and $\boldsymbol{C}_{i+1}$ in i all of the following conditions are satisfied: (1) the number of 2D connected components of O is the same; (2) the number of 2D connected components of $\overline{O}$ is the same; (3) each 2D component of O in $\boldsymbol{C}_i$ is adjacent to exactly one 2D component of O in $\boldsymbol{C}_{i+1}$; (4) each 2D component of $\overline{O}$ in $\boldsymbol{C}_i$ is adjacent to exactly one 2D component of $\overline{O}$ in $\boldsymbol{C}_{i+1}$. Fig. 4 shows a 3D

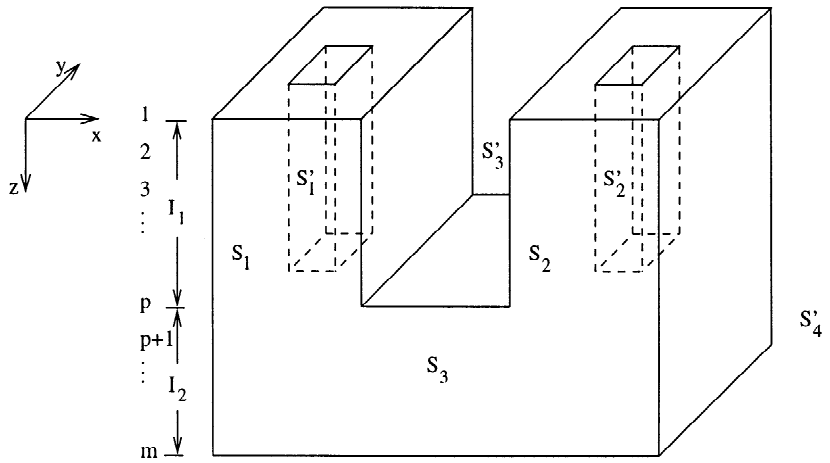

Fig. 4. A 3D object defined by three structures S_1, S_2 and S_3, and two slabs $I_1 = \{\boldsymbol{C}_1, \boldsymbol{C}_2, \ldots, \boldsymbol{C}_p\}$, $p < m$, and $I_2 = \{\boldsymbol{C}_{p+1}, \boldsymbol{C}_{p+2}, \ldots, \boldsymbol{C}_m\}$ of constant topology. S_1 and S_2 belong to slab I_1 and S_3 belongs to slab I_2. I_1 has three co-structures S'_1 and S'_2 (the tunnels) and S'_3 (the outside component) and I_2 has one co-structure S'_4.

example. Here $\boldsymbol{C}^{(m)}$ consists of two slabs, $I_1 = \{\boldsymbol{C}_1, \boldsymbol{C}_2, \ldots, \boldsymbol{C}_p\}$, $p < m$, and $I_2 = \{\boldsymbol{C}_{p+1}, \boldsymbol{C}_{p+2}, \ldots, \boldsymbol{C}_m\}$. Of course, neither the 3D scene nor the binary scene that contains the object is shown here, but hopefully the idea is clear from the schematic. For further usage, we will refer to a 3D connected component of O in a slab I as a *structure* in I and a 3D connected component of $\overline{O}$ in I as a *co-structure* in I. In Fig. 4, I_1 contains two structures S_1 and S_2 and three co-structures S_1', S_2' and S_3' (the background component outside O). I_2 contains one structure S_3 and one co-structure S_4'. Our approach will be to take the user's help in identifying slabs in $\boldsymbol{C}^{(m)}$ and, for each slab, in selecting a structure/co-structure for uniquely indicating a boundary to be created.

The second problem P_2, stated previously is solved from the way the users select orthogonal slices for a given 3D structure/co-structure as we describe in Sections 2.1 and 2.2.

The complete segmentation process consists of five main steps: (i) slab definition, (ii) structure/co-structure selection, (iii) selection of orthogonal slices, (iv) ordering of points and (v) boundary detection.

In Step (i), given a 3D scene $\boldsymbol{C}^{(m)}$ and a 3D object of interest in $\boldsymbol{C}^{(m)}$, the user initially recognizes all structures of the 3D object by observing all natural slices of $\boldsymbol{C}^{(m)}$ in a montage display and then specifies the first natural slice of each slab. The slabs are contiguous so the first natural slice of a current slab determines the last natural slice of the previous slab, and $\boldsymbol{C}_m$ is the last natural slice of the last slab by default.

In Step (ii), to select a structure/co-structure S in a given slab I, the user chooses any natural slice of I and specifies a point c in the interior of S in this natural slice. The point c should be chosen such that the intersection between an axis Z_c, which is parallel to the z axis and which goes through c, and any other natural slice in I is a point in the interior of S.

In Step (iii), given a structure/co-structure S in I, a few orthogonal slices are selected such that the intersection among them is Z_c. The user traces the 2D boundary of S in each orthogonal slice as shown in Fig. 5. The restrictions involving Z_c in (ii) and (iii) are paramount to guarantee the success of step (iv). After tracing contours of S in Step (iii), a set of points is generated on each 2D boundary of S in each natural slice of I.

In Step (iv), the points in each set are ordered according to the orientation of the boundary of S such that each 2D boundary of S in each natural slice of I can be automatically detected in a slice-by-slice fashion in step (v).

In step (v), the user may choose to watch the boundary detection process for S or to have this process running in background while steps (ii), (iii) and (iv) are repeated for other structures of the same or of another slab. If an error is detected in step (v), the user may (a) interrupt the boundary detection process at some natural slice to correct the error and restart the process from this slice on, or (b) let the process go on and make the corrections at the end. However, if the user spends some minutes planning Step (iii), it is possible to select orthogonal slices that avoid further corrections in boundary detection.

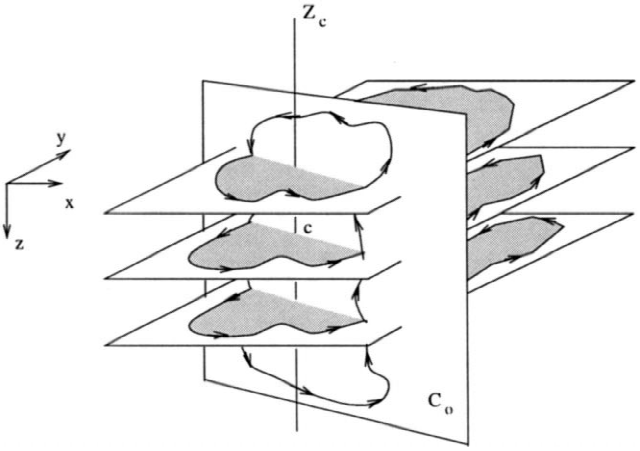

Fig. 5. Live wire tracing on an orthogonal slice $\boldsymbol{C}_o$, selected for a structure/co-structure S in a slab I, generates points on the 3D boundary of S. These points are utilized for automatically live-wiring the 2D boundary of S in each natural slice of I.

To complete the description of the 3D live-wire method just outlined, we present examples and more details of steps (iii), (iv) and (v) in the following sections.

2.1. Selection of orthogonal slices

Given a structure/co-structure S of a slab I and an axis Z_c, which intersects S in all natural slices of I, the user selects a few orthogonal slices that contain Z_c and traces the 2D boundary of S in each orthogonal slice using the 2D live-wire method (Fig. 5). Given a natural slice $\boldsymbol{C}$ of I and a contour representing the 2D boundary of S in a given orthogonal slice, we select the two farthest points of this contour that fall on the 2D boundary of S in $\boldsymbol{C}$. Therefore, each orthogonal contour contributes on each 2D boundary of S in each natural slice of I exactly two points. This is enough to simplify and to solve the ordering of points as we describe in Section 2.2. In this section, we give some examples to illustrate the selection of orthogonal slices.

In Fig. 6, we present a selection of orthogonal slices in an MRI 3D scene of a patient's foot. Fig. 6(a) shows a natural slice of this scene. The region corresponding to a particular bone, namely the talus is shown in Fig. 6(b). The talus is a 3D object which can be defined by one 3D boundary and hence by one structure S and one slab I in $\boldsymbol{C}^{(m)}$. Fig. 6(b) also shows a point c selected in the interior of S and two orthogonal slices, $\boldsymbol{C}_{o_1}$ and $\boldsymbol{C}_{o_2}$, containing Z_c which are represented by the two lines crossing c. $\boldsymbol{C}_{o_1}$ and $\boldsymbol{C}_{o_2}$ are shown in Fig. 6(c) and (d), respectively. In each of these figures, we show the 2D boundary of the talus delineated using 2D live-wire. (Each row in $\boldsymbol{C}_{o_1}$ and $\boldsymbol{C}_{o_2}$

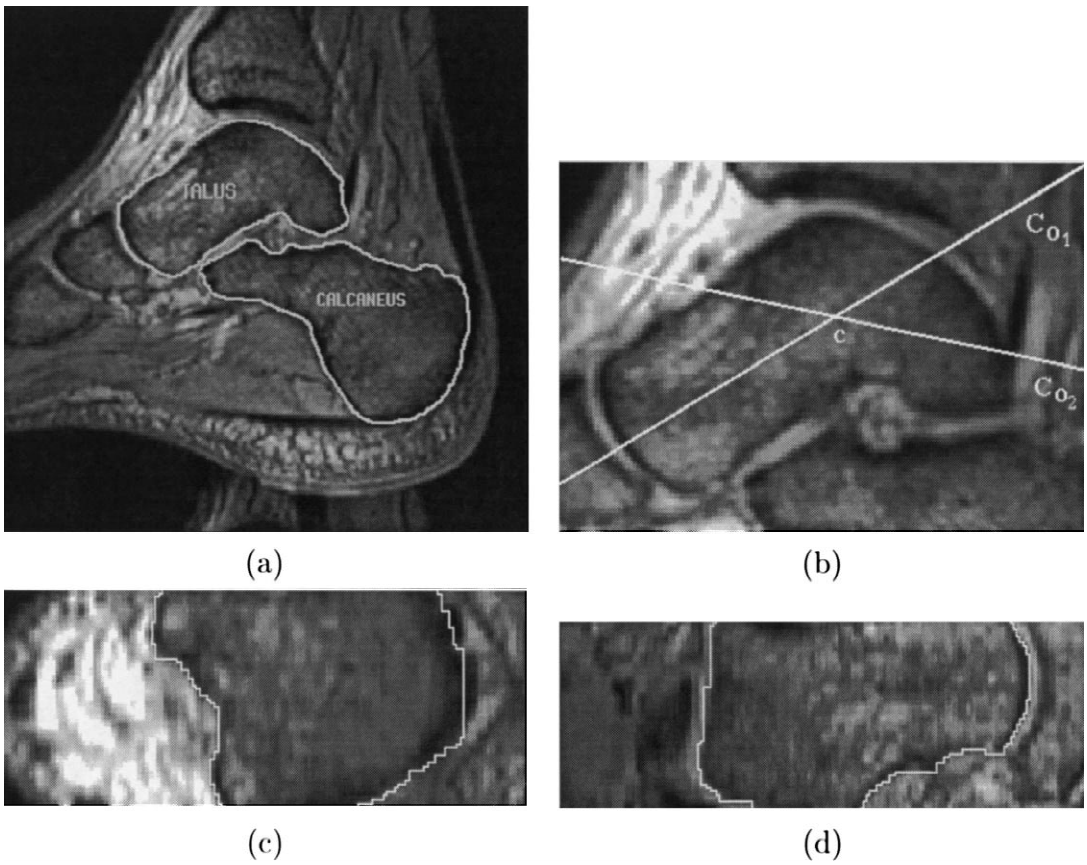

Fig. 6. Selection of orthogonal slices for the talus in a 3D scene of a patient's foot obtained by MRI. This bone can be defined by one structure and one slab of constant object topology. (a) A natural slice of the 3D scene. (b) A region of interest of the slice in (a) showing talus and a point c selected in the interior of the talus and two lines corresponding to the two orthogonal slices, $\boldsymbol{C}_{o_1}$ and $\boldsymbol{C}_{o_2}$, containing Z_c. (c, d) Live-wire tracings on the 2D boundary of the talus in $\boldsymbol{C}_{o_1}$ and $\boldsymbol{C}_{o_2}$, respectively.

represents a natural slice of $\boldsymbol{C}^{(m)}$ and the two horizontal lines near the top and bottom in Fig. 6(c) and (d) represent the first and the last natural slices of I, respectively. These horizontal lines guide the user to localize the structure of interest in the orthogonal slice. Note that only the left most point and the right most point of the orthogonal contour in each row of $\boldsymbol{C}_{o_1}$ and $\boldsymbol{C}_{o_2}$ are mapped to the 2D boundary of S in the corresponding natural slice of I. Thus the user does not need to care about the tracing in between these points, especially, the tracing along the upper and lower horizontal edges of the orthogonal slice.) After tracing these orthogonal boundaries, we have four points on the 2D boundary of S in each natural slice of I. It is clear that some domain-specific knowledge is needed to identify the structures' boundaries in orthogonal slices.

In Fig. 7, we give another example where the 3D object of interest is the calcaneus of a patient's foot (see Fig. 6(a)) in a 3D scene obtained via MRI. The calcaneus requires two slabs, I_1 and I_2, and three structures, S_1, S_2 and S_3. Structures S_1 and S_2 belong to I_1 and structure S_3 belongs to I_2. To generate points on the 3D boundary of S_1, the user selects a point c_1 in the interior of S_1 in a natural slice of I_1 and chooses an orthogonal slice $\boldsymbol{C}_{o_1}$ containing Z_{c_1}. To generate points on the 3D boundary of S_2, the user selects a point c_2 in the interior of S_2 in a natural slice of I_1 and chooses an orthogonal slice $\boldsymbol{C}_{o_2}$ containing Z_{c_2}. To facilitate our explanation, we use the orthogonal slices $\boldsymbol{C}_{o_1}$ and $\boldsymbol{C}_{o_2}$ with the same orientation both containing Z_{c_1} and Z_{c_2}. This situation and the desired 2D boundaries of S_1 and S_2 are shown in Fig. 7(a). Figs. 7(b) and (c) show the 2D live-wire tracings for S_1 and S_2 in $\boldsymbol{C}_{o_1}$ and $\boldsymbol{C}_{o_2}$, respectively, and the two horizontal lines indicating slab I_1. Note that points of the orthogonal contours, which are outside the region limited by the two horizontal lines in $\boldsymbol{C}_{o_1}$ and $\boldsymbol{C}_{o_2}$, fall in natural slices outside I_1 and, therefore, they are discarded. The live-wire tracings in $\boldsymbol{C}_{o_1}$ and $\boldsymbol{C}_{o_2}$ generate two points on the 2D boundary of S_1 and two points on the 2D boundary of S_2 in each natural slice of I_1. To generate points on the 3D boundary of S_3, the user selects a point c_3 in the interior of S_3 in a natural slice of I_2. Again, we choose the orthogonal slice $\boldsymbol{C}_{o_3}$ with the same orientation as that of $\boldsymbol{C}_{o_1}$ and $\boldsymbol{C}_{o_2}$ containing Z_{c_1}, Z_{c_2} and Z_{c_3}. This situation together with the desired 2D boundary of S_3 is indicated in Fig. 7(d). The orthogonal slice $\boldsymbol{C}_{o_3}$, the live-

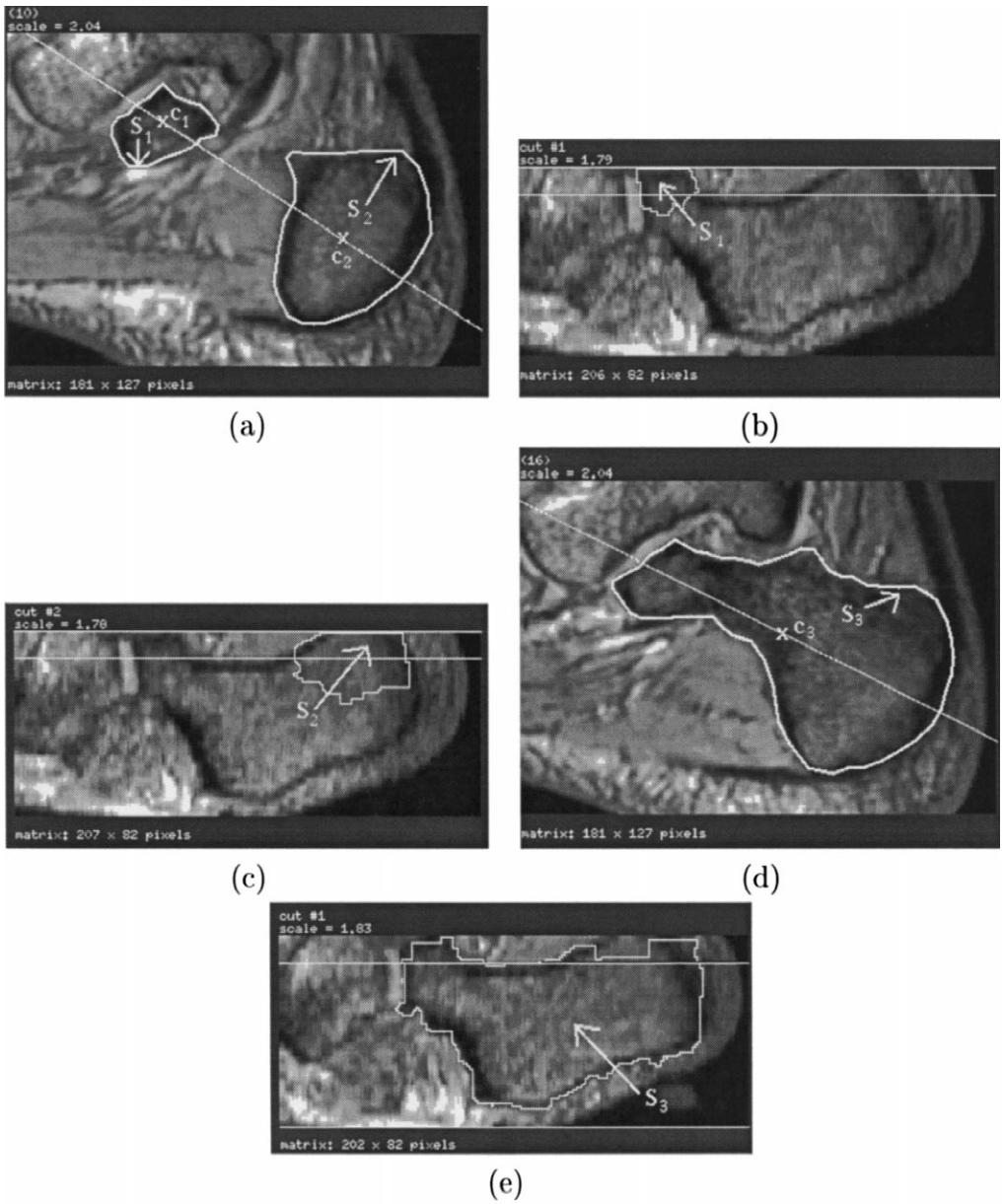

Fig. 7. Selection of orthogonal slices for the calcaneus in a 3D scene of a patient's foot obtained by MRI. This bone can be defined by two structures, S_1 and S_2, in the first slab I_1, and another structure, S_3, in the second slab I_2. (a) A natural slice in I_1 indicating the desired 2D boundaries of S_1 and S_2, a point c_1 in S_1, a point c_2 in S_2 and a line indicating two orthogonal slices, $\boldsymbol{C}_{o_1}$ containing Z_{c_1} and $\boldsymbol{C}_{o_2}$ containing Z_{c_2}, with the same orientation. (b, c) The live-wire tracings on the 2D boundary of S_1 in $\boldsymbol{C}_{o_1}$ and on the 2D boundary of S_2 in $\boldsymbol{C}_{o_2}$, respectively. (d) A natural slice in I_2 indicating the desired 2D boundary of S_3, a point c_3 in S_3 and a line indicating an orthogonal slice $\boldsymbol{C}_{o_3}$ containing Z_{c_3} with the same orientation as that of $\boldsymbol{C}_{o_1}$ and $\boldsymbol{C}_{o_2}$. (e) The live-wire tracing on the 2D boundary of S_3 in $\boldsymbol{C}_{o_3}$.

wire tracing of the 2D boundary of S_3 in $\boldsymbol{C}_{o_3}$, and the two horizontal lines indicating slab I_2 are shown in Fig. 7(e). This orthogonal contour contributes two points to each 2D boundary of S_3 in each natural slice of I_2.

2.2. Ordering the points

After selecting k orthogonal slices for a given 3D structure S of a given slab I in $\boldsymbol{C}^{(m)}$, there is a set V of $2k$ points on each 2D boundary of S in each natural slice of I. By interaction on orthogonal slices, the user has essentially indicated $2k$ key points on each 2D boundary of S in each natural slice $\boldsymbol{C}$. Our aim then ought to be to find globally the best (minimum cost) contour in $\boldsymbol{C}$ that contains these points. The method we have devised to solve this problem is based on the 2D live-wire solution which we have already used extensively and whose behavior we understand very well. Thus, the problem of finding the 2D boundary of S in a natural slice of I as the minimum cost contour, which is closed, connected, oriented and that

contains all points of V, is simplified to a problem of ordering the points of V such that the boundary can be defined from $2k$ optimum boundary segments computed via Dijkstra's algorithm between successive pairs of points of the ordered set. Of course, the solution is not globally optimal, but piecewise optimal as in the 2D method. The ordering process is as follows.

For any natural slice $\boldsymbol{C}$ in I, let $V=\{a_i, b_i | 1 \leq i \leq k\}$ be the set of points on the 2D boundary of S in $\boldsymbol{C}$ resulting from k orthogonal slices. We define a $x'y'$ Cartesian coordinate system in $\boldsymbol{C}$ with c as the origin as shown in Fig. 8. Let Θ be the set of angles, measured from the positive x' axis, of the lines ca_i and cb_i, for $1 \leq i \leq k$. We sort the angles in Θ in the increasing order and create an ordered sequence $\langle v_1, v_2, \ldots, v_{2k}\rangle$ of the points of V for this increasing order of angles. Since we usually define cost assignment to pixel edges such that the desired 2D boundaries in the natural slices are oriented anti-clockwise [see Falcão et al. (1998) for details], the order of the points along the oriented boundary is $v_1, v_2, \ldots, v_{2k}$. If the assumptions for the generation of the $2k$ points are satisfied, it is readily seen that the above method always produces the correct ordering for any input scene and for any object topology.

2.3. Boundary detection

Given the sequence $\langle v_1, v_2, \ldots, v_{2k}\rangle$ of points on the 2D boundary of a given structure/co-structure S in a natural slice of a given slab I in $\boldsymbol{C}^{(m)}$, the complete 2D boundary is now automatically detected as a sequence of $2k-1$ optimum boundary segments from v_i to v_{i+1}, $i=1, 2, \ldots,$ $2k-1$, together with the segment from v_{2k-1} to v_1. This delineation process is then repeated for each 2D boundary in each natural slice in a slice-by-slice fashion under user supervision. During boundary delineation, each optimum boundary segment is computed and displayed in real time. As the starting and the ending points of each live-wire segment are known before optimum path computation, we modified the algorithm described in (Falcão et al., 1998) to find the optimum path from v_i to v_{i+1}, instead of from v_i to all points in the slice.

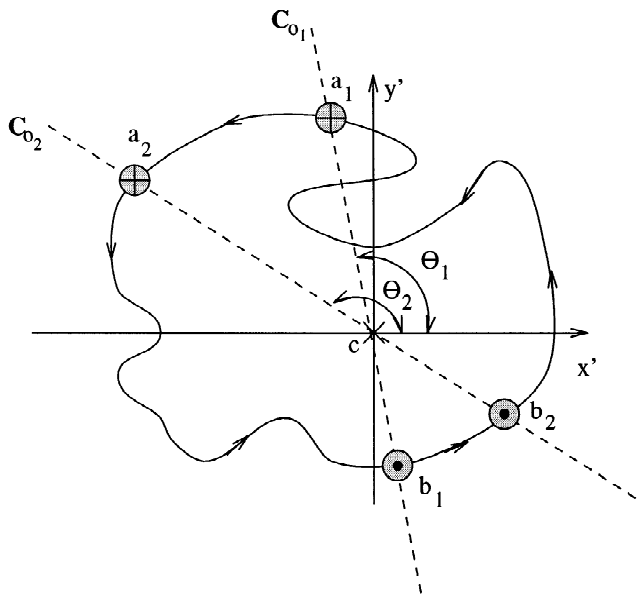

Fig. 8. A natural slice $\boldsymbol{C}$ of a slab I with two orthogonal slices indicated by dashed lines for a structure/co-structure S. Each orthogonal slice generates two points a_i and b_i on the 2D boundary of S in $\boldsymbol{C}$.

In our previous work (Falcão et al., 1998), we introduced the idea of constraining the search for optimal paths in the next slice to an annular region (shell) of width W centered around the projection onto the next slice of the contour(s) traced in the current slice. We mentioned that with this approach live-wire segments appear almost instantaneously and many otherwise distracting boundaries are avoided. In the present paper we found that, as we know the end points of each segment during the automatic boundary delineation process, our algorithm can find optimal paths much faster and so optimum boundary segments appear almost instantaneously even without the annular region. However, this region is still useful to avoid otherwise distracting boundaries.

In summary, from an operator's point of view, the 3D live-wire process proceeds as follows, as implemented in our software system (Udupa et al., 1994). The operator first trains the algorithm by manually tracing a few typical boundary segments in a desired orientation. At the end of tracing, optimum cost assignment is computed (in a few seconds) and stored once for all in a file. Training is not needed for each 3D scene to be segmented as long as the boundary characteristics do not change drastically from scene to scene. This is usually true for scenes pertaining to the same application and acquired as per a given protocol. To begin segmentation of a given scene, a display of the natural slices of the scene is created as a montage. In this display, the operator then indicates the slabs by pointing at slices. In each slab, the operator then selects one slice, a structure/co-structure in this slice, a point in the structure/co-structure and several lines passing through the point to indicate orthogonal slice orientations. The orthogonal slices are then computed and displayed, on each of which the operator traces the boundary of the object using the 2D live-wire method. The selection of structure/co-structure and orthogonal slices and live-wire tracing is repeated for all slabs. At the end of this process, automatic delineation on natural slices starts with on-the-fly display of the boundaries. The operator may interrupt this process any time, make corrections, and resume the process. In our experience, some time spent initially in carefully studying how to select slabs, structures/co-structures, and orthogonal slices can minimize and even eliminate the need for interruption and correction. In applications dealing with a large number of scenes which are generated as per a fixed protocol, such a planning is possible and very effective.

3. Evaluation

In assessing the goodness of a segmentation method three factors – precision, accuracy and efficiency – need to be considered (Falcão et al., 1998). Precision refers to the repeatability of the method and can be measured by evaluating the variations in the result of segmentation because of subjective operator input. Accuracy refers to the degree of agreement with truth. In practice, especially in medical applications, true segmentation is almost impossible to establish, as such various surrogates of truth are used. Usually, delineation by a knowledgeable operator is used as one such alternative. Efficiency refers to the practical viability of the method. This consists of two parts, the computational time and the operator time. Computational time independent of operator involvement does not really matter so long as it is not prohibitive. In our situation, all such computations are done in less than 2 s for scenes with domain $256\times256\times64$ on a Silicon Graphics Indy workstation with 32MB RAM. Operator time required per study, or alternatively the effective number of slices that can be segmented per unit time, can be measured to express efficiency of the method. In (Falcão et al., 1998), based on 2000 tracings in a particular application and statistical analysis of the results, we have shown that the segmentations of the 2D live-wire method in general agree with those of manual tracing (accuracy) and that live-wire method is more repeatable (precision) and 1.5–2.5 times faster (efficiency) than the manual method.

The application considered in the previous evaluation (Falcão et al., 1998) was the kinematic analysis of the tarsal joints of the foot (Hirsch et al., 1996; Udupa et al., 1998; Stindel et al., 1999). In this application, we study the in vivo internal 3D kinematics of the tarsal joints via MR imaging for understanding and quantifying their normal and abnormal motion. We have used the live-wire approaches to segment four bones – the talus, calcaneus, navicular and the cuboid. The segmentation of these bones is difficult because of the lack of MR signal from bone, the presence of adjoining bone boundaries, and the tendons and ligaments that attach to the bones. Particularly, we have used the talus and the calcaneus in this evaluation study. In this section, we will evaluate the precision and efficiency of the 3D live-wire method compared to the 2D method for segmenting the same bones – the talus and calcaneus. For accuracy, we will compare the 3D results with that of the 2D method using essentially the 2D results as a surrogate of truth. To avoid playing any competitive role in performance between the two methods, we use the same cost assignments for both methods. Both methods were implemented in an internal version of 3DVIEWNIX (Udupa et al., 1994) and executed on a SGI Indy workstation.

Given a 3D object in $\boldsymbol{C}^{(m)}$, it is possible to find a suficient number of orthogonal slices needed to automatically delineate the 3D boundary of this object using 3D live-wire. The time spent by the user in 3D live-wire is thus independent of the number m of natural slices in $\boldsymbol{C}^{(m)}$. In the 2D method, this time increases with m. Therefore, for a 3D scene with a small number of natural slices, 2D live-wire may be more attractive than 3D. To evaluate this aspect statistically, we did a rough sample size calculation (Rosner, 1995) and found that at least seven 3D scenes were needed. We took a scene $\boldsymbol{C}^{(32)}$ consisting of 32 MR slices of the foot of a patient and created additional six scenes by linear interpolation without changing the level of discretization within the slice but changing only the slice separation. This gave us a set $\mathscr{C}=\{\boldsymbol{C}^{(21)}, \boldsymbol{C}^{(27)}, \boldsymbol{C}^{(32)}, \boldsymbol{C}^{(38)}, \boldsymbol{C}^{(43)}, \boldsymbol{C}^{(50)}, \boldsymbol{C}^{(63)}\}$ of seven 3D scenes. Each of two human operators, O_1 and O_2, segmented the 3D boundary of the talus using each of two methods, 2D live-wire ($2D$) and 3D live-wire ($3D$), in each 3D scene of C. In all experiments using 3D live-wire, the talus is segmented, without requiring corrections during or after the boundary detection process, with only five orthogonal slices selected by the user. The time (in minutes) spent by each human operator $o\in\{O_1, O_2\}$ using each method $x\in\{2D, 3D\}$ to segment the talus in each 3D scene of C is presented in Table 1. Note that the efficiency measure here includes the total time spent by an operator in all steps except training.

Subsequently, each of two human operators, O_1 and O_2, segmented each of the two bones, talus (T_a) and calcaneus (C_a), in $\boldsymbol{C}^{(32)}$ using each of the two methods, $2D$ and $3D$, in seven separate segmentation experiments, denoted E_1, $E_2, \ldots, E_7$. The reason for considering the talus and the calcaneus is that they have quite different slab topology and represent different degrees of complexity for the segmentation task using 3D live-wire. The talus consists of one 3D structure with well defined segments which needs five orthogonal slices to be correctly segmented with 3D live-wire. The calcaneus is represented by three 3D structures defined in two slabs and consists of mostly poorly defined segments. The calcaneus requires 20 orthogonal slices to be segmented with 3D live-wire without subsequent corrections. The segmentation of the talus generates one closed contour per natural slice in $\boldsymbol{C}^{(32)}$ while the segmentation of the calcaneus generates one or

Table 1
User's time (min) for both operators and methods for segmenting T_a in 3D scenes with different number of slices

Number of slices	User's time (min)			
	$2D$		$3D$	
	O_1	O_2	O_1	O_2
21	3.8	4.5	2.0	2.3
27	4.2	5.3	2.2	2.2
32	5.8	6.4	2.3	2.6
38	6.3	7.9	2.1	2.5
43	7.7	8.7	2.2	2.7
50	8.9	10.4	2.1	2.3
63	10.2	12.0	2.4	2.4

two closed contours per natural slice in $\boldsymbol{C}^{(32)}$. For any scene $\boldsymbol{C}^{(m)}$, we use the notation $\boldsymbol{C}^{(m)} = (C^{(m)}, g)$, where $C^{(m)}$ is the domain of $\boldsymbol{C}^{(m)}$ and, for any v in $C^{(m)}$, $g(v)$ is the scene intensity of v. Thus, each segmentation experiment $e \in \{E_1, E_2, \ldots, E_7\}$ in $\boldsymbol{C}^{(32)}$, involving a human operator $o \in \{O_1, O_2\}$, a method $x \in \{2D, 3D\}$ and a bone $b \in \{T_a, C_a\}$, resulted in a 3D binary scene $\boldsymbol{C}_e^{(32)} = (C_e^{(32)}, g_e)$, where $C_e^{(32)} = C^{(32)}$ and, for any voxel $v \in C^{(32)}$, $g_e(v) = 1$ if v was inside some contour of b in some natural slice of $\boldsymbol{C}^{(32)}$ and $g_e(v) = 0$ if v was outside all contours of b considering all natural slices of $\boldsymbol{C}^{(32)}$. Note that each 3D binary scene resulting from an experiment contains information about only one bone. Therefore, this evaluation study consists of 56 experiments ($7 \times 2 \times 2 \times 2$) in total.

Consider any fixed $o \in \{O_1, O_2\}$, $e \in \{E_1, E_2, \ldots, E_7\}$, $x \in \{2D, 3D\}$, and $b \in \{T_a, C_a\}$. We define the speed of segmentation SP_{oexb} (in slices/min) of a human operator o in an experiment e using a method x to segment a bone b in a 3D scene $\boldsymbol{C}^{(m)}$ as m/t_e, where t_e is the user's total interaction time required in this segmentation experiment. We define the speed of segmentation SP_{oxb} of a human operator o using a method x to segment a bone b in a 3D scene $\boldsymbol{C}^{(m)}$ to be the average of all speeds SP_{oexb} over all segmentation experiments e involving o, x and b. Finally, we define the speed of segmentation SP_{xb} of a method x to segment a bone b in a 3D scene $\boldsymbol{C}^{(m)}$ to be the average of all speeds SP_{oxb} over all human operators o involving x and b.

Table 2 lists the values of SP_{oexb} for all possible values of o, e, x and b. Table 3 lists the difference in speed between methods 2D and 3D for each human operator o and object b. The second (bottom) entry in this table indicates the statistical significance (the P value) of the difference in speed obtained by conducting a Student's t-test (Rosner, 1995). Table 4 shows the values of SP_{oxb} for all o, x and b values, and Table 5 shows the values of SP_{xb} for all x and b values.

Table 2
Segmentation speeds (slices/min) SP_{oexb} for all possible values of o, e, x and b

		T_a		C_a	
		O_1	O_2	O_1	O_2
2D	E_1	5.4	4.6	3.7	3.9
	E_2	5.7	4.9	4.1	4.1
	E_3	5.5	5.0	3.7	4.1
	E_4	5.4	5.1	4.3	4.2
	E_5	5.4	5.2	4.4	4.3
	E_6	5.7	5.3	4.4	4.3
	E_7	5.6	5.3	4.5	4.3
3D	E_1	13.7	11.4	6.0	4.6
	E_2	16.0	11.7	5.5	4.7
	E_3	14.8	12.8	4.7	5.3
	E_4	13.8	13.7	5.6	5.6
	E_5	14.2	14.2	5.4	5.5
	E_6	16.3	13.9	5.2	5.7
	E_7	14.4	15.0	5.4	5.8

Table 3
Difference in speeds $|\mathrm{SP}_{oxb} - \mathrm{SP}_{ox'b}|$ (top) and its level of statistical significance (bottom) for all operators o and objects b and $x' = 3D$ and $x = 2D$

	T_a	C_a
O_1	9.3	1.2
	0.000	0.008
O_2	8.2	1.1
	0.001	0.010

Table 4
Segmentation speeds (slices/min) SP_{oxb} for all possible values of o, x, and b

	T_a		C_a	
	O_1	O_2	O_1	O_2
2D	5.5	5.0	4.2	4.2
3D	14.7	13.2	5.4	5.3

Let $\boldsymbol{C}_e^{(32)}$ be a 3D binary scene resulting from a segmentation experiment e using the 3D scene $\boldsymbol{C}^{(32)}$. We use the notation $|\boldsymbol{C}_e^{(32)}|$ to denote the number of voxels of $\boldsymbol{C}_e^{(32)}$ with value 1. The exclusive OR operation EOR between any two 3D binary scenes $\boldsymbol{C}_{e_1}^{(32)} = (C_{e_1}^{(32)}, g_{e_1})$ and $\boldsymbol{C}_{e_2}^{(32)} = (C_{e_2}^{(32)}, g_{e_2})$ such that $C_{e_1}^{(32)} = C_{e_2}^{(32)}$, written as $\boldsymbol{C}_{e_1}^{(32)} \mathrm{EOR} \boldsymbol{C}_{e_2}^{(32)}$, results in a 3D binary scene $\boldsymbol{C}_e^{(32)} = (C_e^{(32)}, g_e)$ where $C_e^{(32)} = C_{e_1}^{(32)} = C_{e_2}^{(32)}$ and, for any voxel $v \in C_e^{(32)}$, $g_e(v) = 1$ if $g_{e_1}(v) \neq g_{e_2}(v)$ and $g_e(v) = 0$ if $g_{e_1}(v) = g_{e_2}(v)$. Let e_1 and e_2 be any segmentation experiments involving the same fixed object $b \in \{\mathrm{T}_a, \mathrm{C}_a\}$ and the 3D scene $\boldsymbol{C}^{(32)}$, and let $\boldsymbol{C}_{e_1}^{(32)}$ and $\boldsymbol{C}_{e_2}^{(32)}$ be the 3D binary scenes resulting from these experiments. We define the repeatability of segmentation, $\mathrm{RP}_{e_1e_2}$, for object b in scene $\boldsymbol{C}^{(32)}$ in experiments e_1 and e_2 as

$$\mathrm{RP}_{e_1e_2} = 1 - \frac{|\boldsymbol{C}_{e_1}^{(32)} \mathrm{EOR} \boldsymbol{C}_{e_2}^{(32)}|}{|\boldsymbol{C}_{e_1}^{(32)}| + |\boldsymbol{C}_{e_2}^{(32)}|}. \quad (1)$$

For any fixed $o_1 \in \{O_1, O_2\}$, $o_2 \in \{O_1, O_2\}$, $e_1 \in \{E_1, E_2, \ldots, E_7\}$, $e_2 \in \{E_1, E_2, \ldots, E_7\}$, $x \in \{2D, 3D\}$ and $b \in \{\mathrm{T}_a, \mathrm{C}_a\}$, the repeatability of segmentation $\mathrm{RP}_{o_1o_2e_1e_2xb}$ by human operators o_1 and o_2 in experiments e_1 and e_2 using method x for object b is the average of all $\mathrm{RP}_{e_1e_2}$ for all segmentation experiments e_1 and e_2 which are such that $e_1 \neq e_2$, e_1 and e_2 are conducted on the same 3D input scene $\boldsymbol{C}^{(32)}$, e_1 involves human operator o_1, e_2 involves

Table 5
Segmentation speeds (slices/min) SP_{xb} for all possible values of x and b

	T_a	C_a
2D	5.3	4.2
3D	14.0	5.4

Table 6
Repeatability $RP_{oo'xb}$ for all possible values of o, o', x and b

		T_a		C_a	
		O_1	O_2	O_1	O_2
2D	O_1	0.9922	0.9890	0.9845	0.9713
	O_2	0.9890	0.9909	0.9713	0.9737
3D	O_1	0.9950	0.9925	0.9814	0.9767
	O_2	0.9925	0.9948	0.9767	0.9875

human operator o_2, and both involve method x and object b. $RP_{o_1o_2e_1e_2xb}$ measures inter-operator repeatability when $o_1 \neq o_2$, and it measures intra-operator repeatability when $o_1 = o_2$. We define the repeatability of segmentation $RP_{o_1o_2xb}$ by human operators o_1 and o_2 using method x for object b as the average of the repeatabilities $RP_{o_1o_2e_1e_2xb}$ over all combinations of experiments e_1 and e_2. Finally, the repeatability of segmentation RP_{xb} using method x for object b is the average of all $RP_{o_1o_2xb}$ over all combinations of human operators o_1 and o_2.

Table 6 lists the segmentation repeatabilities $RP_{o_1o_2xb}$ for all possible values of o_1, o_2, x and b. Table 7 shows the difference in segmentation repeatability between 2*D* and 3*D* for each human operator and object. The second (bottom) entry in this table indicates the statistical significance of the difference in repeatability obtained by conducting a Student's *t*-test (Rosner, 1995). Finally, we show in Table 8 the segmentation repeatabilities RP_{xb} for all values of x and b. Note that the repeatability measure defined in Eq. (1) can be used also to compare how well the 3D segmentations agree with the 2D results. This can be used to establish the accuracy of 3D segmentation using the 2D results as representing truth. For scene $\boldsymbol{C}^{(32)}$, the mean degree of overlap between the segmentation results from the 2*D* and 3*D* methods were found to be 0.97 for T_a and 0.96 for C_a computed using Eq. (1).

The following observations may be made from the results presented in Tables 1–8. Users need about 2 min of interaction time to segment the talus using 3D live-wire and this time is independent of the number of slices in the 3D scene unlike for the 2D live-wire method. We note here that training is required only once for an application and is not needed on a per study basis. This is typically under 5 min [see (Falcão et al., 1998) for details], and is not included in the speed measurements. By varying the number of natural slices from 21 to 63, we found that 3D live-wire is 2–6 times faster than 2D live-wire in segmenting the talus. For a 3D scene with 32 natural slices, the 3D live-wire is 2.6 times faster than 2D live-wire for segmenting the talus and 1.3 times faster than 2D live-wire for segmenting the calcaneus. This means that there may be objects that may require more user control and involvement than others, in which case the advantage of 3D over 2D may be marginal. Overall, 3D live-wire is usually 2–6 times faster (with a statistical significance level of $p < 0.01$) than 2D live-wire. We already know that 2D live-wire is 1.5–2.5 times faster than manual tracing (Falcão et al., 1998) for the segmentation task considered here. Consequently, we conclude that 3D live-wire is 3–15 times faster than manual tracing for this segmentation task. We also conclude that 3D live-wire is more repeatable, with a statistical significance level of $p < 0.0001$, than 2D live-wire, and hence, than manual tracing. This is understandable because as the degree of automation increases, the precision of the method increases. In summary, for the particular application tested, we conclude that there is strong evidence that suggests the order 3D>2D>manual tracing for precision and efficiency of segmentation. All results have a high degree of agreeability to the tracings by knowledgeable operators.

Table 7
Difference in repeatability $|RP_{oxb} - RP_{ox'b}|$ (top) and its level of statistical significance (bottom) for all operators o and objects b and $x' = 3D$ and $x = 2D$

	T_a	C_a
	2D, 3D	2D, 3D
O_1	0.0028	0.0031
	0.0000	0.0001
O_2	0.0039	0.0137
	0.0000	0.0001

Table 8
Repeatability RP_{xb} for all possible values of x and b

	T_a	C_a
2D	0.9903	0.9752
3D	0.9937	0.9806

4. Concluding remarks

Any difficult segmentation task requires a certain minimum degree of user assistance to produce, at an acceptable level of precision, results that are agreeable to the tracings by knowledgeable human operators. Our aim in segmentation research for practical applications ought to be to develop solutions that require a degree of user assistance as close to this theoretical minimum as possible. Ideally, the aim ought to be to develop such solutions in an application-independent manner without compromising precision and accuracy. Toward these goals, we have presented a new method which is an extension of our previous user-steered image segmentation paradigms, namely live-wire and live-lane (Falcão et al., 1998). In this 3D extension, after the user specifies contours via live-wiring on a few strategically selected orthogonal slices, several points are generated on the 3D boundary of the object. These are sufficient to trace optimum boundary segments automatically in all natural slices of the 3D scene. Strategic selection of orthogonal contours and

thereby a quick ''sprinkling'' of points on the 3D boundary is the essence of our effort toward providing the minimum required assistance. We use this approach when the required number of orthogonal slices is much less than the number of natural slices in the 3D scene. In situations where a larger number of orthogonal slices are necessary to segment the 3D object, the advantage of the 3D method over the 2D live-wire approach may not be significant (or actually the 2D method may be better). When the objects are very complex in shape then the 3D method may take a lot more time for user interaction than for blob-like objects. For topologically very complex objects, such as the gray matter or white matter object in the brain, the 3D method (or any boundary-based approach) is not appropriate and there are better region-based methods, such as fuzzy connectedness (Udupa and Samarasekera, 1996), for segmenting them.

We have observed that as the users become more familiar with the procedures in the 3D live-wire, they are able to find out faster the suficient number of orthogonal slices needed for the subsequent automatic delineation of the object and, therefore, they can finish segmentation quicker. For the application tested in this paper, we have observed a 3–15-fold speedup of segmentation compared to slice-by-slice manual tracing and a 2–6-fold speedup compared to 2D live-wire. The precision of the 3D method is also better than that of the 2D and manual methods.

One possible initial disadvantage of the method is the somewhat unfamiliar object cross-section presented on arbitrarily oriented orthogonal slices. In applications involving the processing of a large number of similar data sets, some time spent initially in understanding this issue and in planning the selection of slabs and orthogonal slices, can significantly improve the efficiency of the method.

There is a possible preprocessing strategy to improve the efficiency of the 3D live-wire method for segmenting 3D objects with multiple structures per slab. The 3D scene, which contains this object, can be reformatted into a different orientation (i.e., sagittal, coronal or oblique) in order to reduce the number of structures and slabs for representing the 3D object. The advantage of this strategy is that the user needs to select a smaller number of orthogonal slices in the new scene for automatic delineation compared to the number of orthogonal slices in the original scene.

Acknowledgements

The work of A.X.F. was partially supported by CNPq (Proc. 300698/98-4) and FAPESP (Proc. 98/06314-5 and 98/12308-8). The work of J.K.U. was supported by an NIH grant NS37172 and an ARPA grant DAMD179717271.

References

Amini, A., Bookstein, F.L., Wilson, D.C. (Guest Editors), 1997. Special Issue on Biomedical Image Analysis. Comput. Vis. Image Understanding 2, 66.

Barrett, W.A., Mortensen, E.N., 1996. Fast, accurate, and reproducible live-wire boundary extraction. In: Proceedings of Visualization in Biomedical Computing. Hamburg, Germany. pp. 183–192.

Barrett, W.A., Mortensen, E.N., 1997. Interactive live-wire boundary extraction. Medical Image Analysis 1 (4), 331–341.

Cappelletti, J.D., Rosenfeld, A., 1989. Three-dimensional boundary following. Comput. Vis. Graphics Image Processing 48 (1), 80–92.

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

Cormen, T., Leiserson, C., Rivest, R., 1991. Introduction to Algorithms. MIT Press, New York, NY.

Falcão, A.X., Udupa, J.K., 1997. Segmentation of 3D objects using live-wire. In: Proceedings of SPIE on Medical Imaging. Newport Beach, CA. Vol. 3034, pp. 228–239.

Falcão, A.X., Udupa, J.K., Samarasekera, S., Hirsch, B.E., 1996. User-steered image boundary segmentation. In: Proceedings of SPIE on Medical Imaging. Newport Beach, CA. Vol. 2710, pp. 278–288.

Falcão, A.X., Udupa, J.K., Samarasekera, S., Sharma, S., Hirsch, B.E., Lotufo, R.A., 1998. User-steered image segmentation paradigms: live-wire and live-lane. Graphical Models and Image Processing 60 (4), 233–260.

Gerig, G., Kübler, O., Kikinis, R., Jolesz, F., 1992. Nonlinear anisotropic filtering of MRI data. IEEE Trans. Medical Imaging 11, 221–232.

Haralick, R.M., Shapiro, L.G., 1985. Image segmentation techniques. Comput. Vis. Graphics Image Processing 29 (1), 100–132.

Hirsch, B.E., Udupa, J.K., Samarasekera, S., 1996. A new method of studying joint kinematics from 3D reconstructions of MRI data. J. Am. Podiatric Med. Ass. 86 (1), 4–15.

Kass, M., Witkin, A., Terzopoulos, D., 1987. Snakes: active contour models. Int. J. Computer Vision 1 (4), 321–331.

Maintz, J., van den Elsen, P., Viergever, M., 1996. Comparison of edge-based and ridge-based registration of CT and MR brain images. Medical Image Analysis 1, 151–161.

Martelli, A., 1971. An application of heuristic search methods to edge and contour detection. Comm. ACM 19, 73–83.

McInerney, T., Terzopoulos, D., 1996. Deformable models in medical image analysis: a survey. Medical Image Analysis 1 (2), 91–108.

Morse, B.S., Barrett, W.A., Udupa, J.K., Burton, R.P., 1991. Trainable optimal boundary finding using two-dimensional dynamic programming. Medical Image Processing Group, Department of Radiology, University of Pennsylvania, Technical Report MIPG180.

Mortensen, E.N., Barrett, W.A., 1998. Interactive segmentation with intelligent scissors. Graphical Models and Image Processing 60 (5), 349–384.

Mortensen, E.N., Morse, B.S., Barrett, W.A., Udupa, J.K., 1992. Adaptive boundary detection using live-wire two dimensional dynamic programming. IEEE Proc. Comput. Cardiology, 635–638.

Mortensen, E.N., Barrett, W.A., 1995. Intelligent scissors for image composition. In: Proceedings of Computer Graphics (SIGGRAPH'95). Los Angeles, CA. pp. 191–198.

Park, J., Metaxas, D., Axel, L., 1996. Analysis of left ventricular wall motion based on volumetric deformable models and MRI-SPAMM. Medical Image Analysis 1 (1), 53–71.

Prewitt, J.M.S., Mendelsohn, M.L., 1966. The analysis of cell images. Ann. N.Y. Acad. Sci. 128, 1035–1053.

Raya, S., Udupa, J., 1990. Shape-based interpolation of multidimensional objects. IEEE Trans. Medical Imaging 9, 32–42.

Rhoad, R.C., Klimkiewicz, J.J., Williams, G.R., Kesmodel, S.B., Udupa, J.K., Kneeland, B., Iannotti, J.P., 1998. A new in vivo technique for 3D shoulder kinematics analysis. Skeletal Radiology 27, 92–97.

Rosner, B., 1995. Fundamentals of Biostatistics. Duxbury Press, New York.

Stindel, E., Udupa, J.K., Hirsch, B.E., Odhner, D., Couture, C., 1999. 3D MR image analysis of the morphology of the rear foot: application to classification of bones. Comput. Medical Imaging and Graphics 23, 75–83.

Udupa, J.K., 1994. Multidimensional digital boundaries. CVGIP: Graphical Models and Image Processing 56 (4), 311–323.

Udupa, J.K., Ajjanagadde, V.G., 1990. Boundary and object labelling in three-dimensional images. Comput. Vis. Graphics Image Processing 51, 355–369.

Udupa, J.K., Samarasekera, S., 1996. Fuzzy connectedness and object definition: theory, algorithms and applications in image segmentation. Graphical Models and Image Processing, May, 246–261.

Udupa, J.K., Samarasekera, S., Barrett, W.A., 1992. Boundary detection via dynamic programming. Proc. of SPIE 1808, 33–39.

Udupa, J.K., Odhner, D., Samarasekera, S., Goncalves, R.J., Iyer, K., Venugopal, K., Furuie, S., 1994. 3DVIEWNIX: An open, transportable, multidimensional, multimodality, multiparametric imaging software system. SPIE Proc. 2164, 58–73.

Udupa, J.K., Hirsch, B.E., Samarasekera, S., Hillstrom, H., Bauer, G., Kneeland, B., 1998. Analysis of in vivo 3D internal kinematics of the joints of the foot. IEEE Trans. Biomedical Engineering 45, 1387–1396.

Wong, K.C. et al., 1998. Interactive volumme cutting. In: Proceedings of Graphics Interface. Vancouver, BC. pp. 99–105.