HAL
archives-ouvertes.fr

Iso-surface Extraction in 4D with Applications related to Scale Space

Márta Fidrich

▶ To cite this version:

Márta Fidrich. Iso-surface Extraction in 4D with Applications related to Scale Space. RR-2833, INRIA. 1996. inria-00073858

HAL Id: inria-00073858
https://hal.inria.fr/inria-00073858

Submitted on 24 May 2006

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Iso-surface Extraction in 4D with Applications related to Scale Space

Márta Fidrich

N° 2833

March 1996

PROGRAMME 4

ISSN 0249-6399

Iso-surface Extraction in 4D with Applications related to Scale Space

Márta Fidrich *

Programme 4 — Robotique, image et vision
Projet Epidaure **

Rapport de recherche n ° 2833 — March 1996 — 28 pages

Abstract: We present a method for extracting iso-surfaces and their intersections in 4D. Our work is a significant extension of the 3D Marching Lines algorithm with new orientation and implementation considerations. As a practical tool, it can be applied to track efficiently space curves, defined by differential invariants, across increasing scale.

Key-words: Iso-surface extraction, Marching Lines algorithm, Topology, Scale space

(Résumé : tsvp)

*e-mail: marta.fidrich@sophia.inria.fr
**http://zenon.inria.fr:8003/Equipes/EPIDAURE-eng.html

Unité de recherche INRIA Sophia-Antipolis
2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex (France)
Téléphone : (33) 93 65 77 77 – Télécopie : (33) 93 65 77 65

Extraction d'iso-surfaces en 4D avec des applications à la théorie multi-échelle

Résumé : Dans ce rapport, nous présentons une méthode pour l'extraction des iso-surfaces et de leurs intersections en 4D. Notre travail est une extension importante de l'algorithme tri-dimensionnel "Marching Lines" avec des nouvelles considérations sur l'orientation et l'implémentation. En pratique, cette méthode peut être utilisée pour suivre efficacement dans l'espace multi-échelle des courbes spatiales définies par des invariants différentiels.

Mots-clé : Extraction d'iso-surfaces, Algorithme "Marching Lines", Topologie, Espace multi-échelle

1 Introduction

Due to the intrinsic nature of intensity in medical images, iso-surface extraction is a generally used method in 3D medical image processing; see [10] for a good survey of existing works. As 4D images are becoming increasingly common, the challenge of the extraction of 4D hyper-iso-surfaces and their intersections has emerged. Practical applications consist of processing temporal 3D images like beating heart sequences (3 space dimension and 1 time), or tracking topological changes of 3D characteristic lines in scale space (3 space dimension and 1 scale). The latter is necessitated for high-level computerized tasks, such as inter-patient registration or building reliable anatomical atlases. In fact, our motivation for designing a 4D algorithm comes from multiscale analysis of 3D images, which is a natural continuation of our previous work [5, 3]. As far as we know, there is no published paper about the extraction of 4D hyper-iso-surfaces and their intersection curves.

A 4D (or higher dimensional) space is challenging to imagine: though there is nothing special in treating the 4th coordinate mathematically, we have to revise our way of thinking about orientation: left-right, top-bottom do not exist in 4D contrary to the 3D world. To avoid any mysticism that one can feel while treating higher dimensional spaces, we describe our concepts following an intuitive approach based on dimensional analogy. However, this analogy could be misleading as the next simple example shows: the circumference of a 2D circle is $2\pi r$, the surface of a 3D sphere is $4\pi r^2$, while the hyper-surface of a 4D hyper-sphere is not $6\pi r^3$ nor $8\pi r^3$, but $2\pi^2 r^3$. Hence, it is particularly important to check our ideas, specially about orientation, if we want to design an algorithm which guarantees good topological properties of the result.

This work is inspired by the Marching Lines algorithm [18], which is briefly summarized in the first section. In the next section we describe its extension to the 4D case. Then we show how this method can be applied to process 3D images at multiple scales. Finally, we discuss some implementation details.

2 3D overview

There are several techniques to extract 3D iso-surfaces, where the classical example is the Marching Cubes algorithm [13]. To compute the 3D intersection

curves of two iso-surfaces, the Marching Lines algorithm [18] is an effective tool. Like the Marching Cubes algorithm, it uses a beveled-form representation [10], i.e. voxels are viewed as 3D grid points, 8 adjacent voxels form a cubic *8-cell*. By means of interface and orientation conventions it resolves singular situations in a consistent and plausible manner and ensures good topological properties. (The Marching Cubes algorithm in its original form can produce surfaces with holes.) The reconstructed discrete surface is oriented, complete i.e. without holes (except at the boundary of the 3D grid) and contains no self-intersections. Also, each intersection curve is oriented (linked list of 3D points where the order of points reflects the orientation) and closed (if the curve is contained within the 3D grid). Since the Marching Lines algorithm is a starting point of our 4D extension, we now summarize the conventions used and the main steps of the algorithm; for proof of correctness see [18].

2.1 Extraction of iso-surfaces

Definition 1 (iso-surface as interface) *The iso-surface is the interface between the regions of the image f: $f \geq I$ (the inside) and $f < I$ (the outside), where I is a constant.*

According to this convention, each voxel is labeled either to be positive ($f \geq I$) or negative ($f < I$); to follow the explication see Fig. 1, 2. If at least one of the voxel label of the 8-cell differs from the others, the 8-cell will contain some iso-surface. The points at which the iso-surface intersects the edges of the cube, called *vertices of iso-patches* (2 or 4 par face), are obtained by linear interpolation along the cell edges giving sub-voxel accuracy. To define adjacency-relations between vertex pairs on a *4-side*, the next conventions are used:

Definition 2 (left-hand orientation of a 4-side) *When following a closed planar curve its inside should always be on the left.*

Definition 3 (outside orientation of a 8-cell) *The surface of a cubic cell is oriented toward the outside of the cell.*

So each face of a cube is oriented according to the left-hand convention if it is seen from outside.

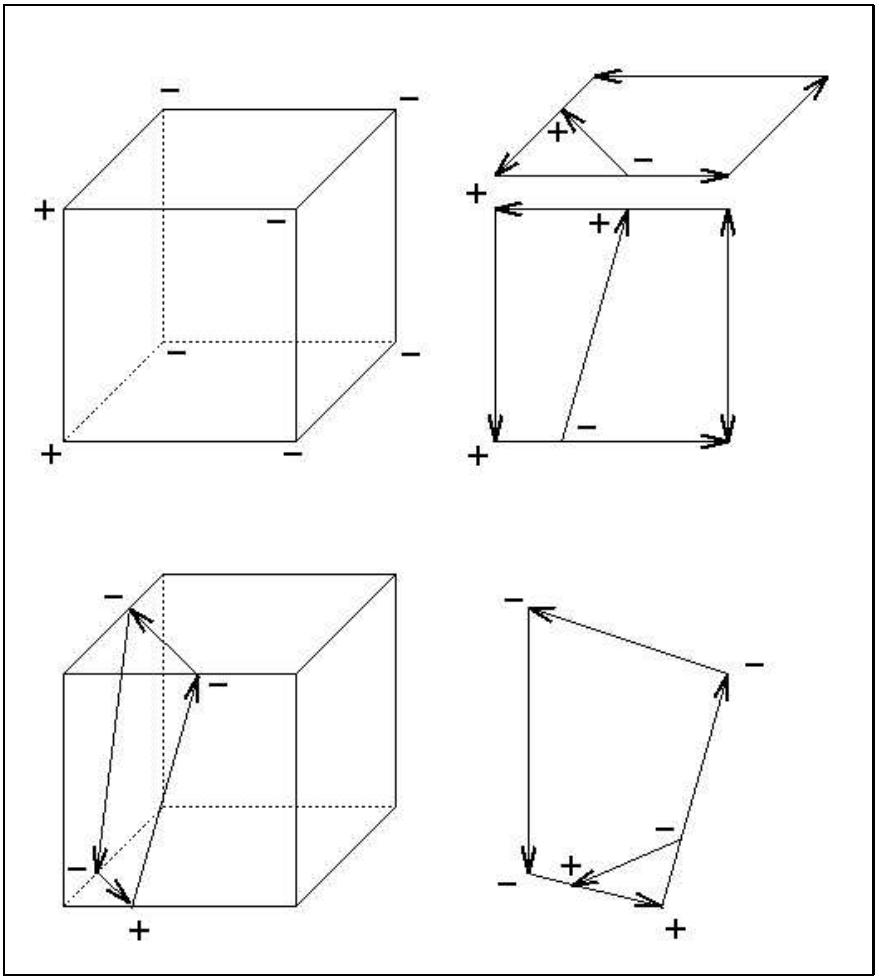

Figure 1: Explication of the 3D Marching Lines algorithm.
(1) Voxels are labeled according to $(f \geq I)$ or $(f < I)$ given by the first image. *(2)* Faces are oriented using the left hand convention; vertices of the iso-patch are computed with linear interpolation and labeled with the label of the endpoint of the left-hand directed edges. *(3,4)* The directed bi-iso-segment on the iso-patch is similarly obtained with interpolation, this time using the intensity values of the second image.

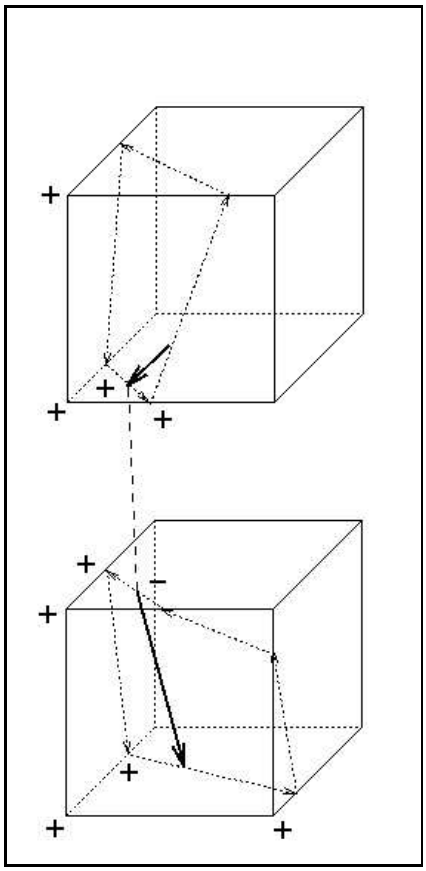

Figure 2: Explication of the 3D Marching Lines algorithm (continued).
The iso-patches have opposite orientation in adjacent cubic cells, the common endpoint of the two bi-iso-segments is oppositely labeled on the two iso-patches. So the bi-iso-segments can be easily followed using their direction.

On each 4-side the vertices of iso-patches are labeled with the label of the endpoint of the left-hand oriented edges. The vertices are formed to iso-segments from points labeled $-$ to points labeled $+$. The ambiguous case of alternately labeled voxels – giving 4 vertices – on a 4-side is resolved by the mean-value rule. (A non-deterministic treating of this case caused the holes in the original form of the Marching Cubes algorithm.)

Definition 4 (mean-value rule) *Assuming that 4 interpolated vertices are generated: If the mean-value of a 4-side is positive ($f \geq I$) two segments are formed to give one positive component, otherwise to give two components.*

As a result of the previous three conventions, the vertices of a cubic 8-cell (giving directed iso-segments on each 4-side) can be organized into iso-patch cycles. Moreover, each iso-patch is oriented: its vertices are connected so that the positive part of the corresponding 4-side should be oriented using the left hand convention. Also, iso-patches have opposite orientation in adjacent cubic cells.

2.2 Extraction of intersection curves

To get the intersection line segments, we consider a regular 3D grid whose voxels are labeled according to two images f and g. We first compute iso-patch cycles with the intensity values of f, then we proceed to get *bi-iso-segments* (similarly as we did on a 4-side), this time using the iso-value corresponding to the second iso-surface at the iso-patch vertices. To avoid the ambiguity at tangent intersection of surfaces, the order of iso-surfaces is preset:

Definition 5 (intersection order of iso-surface) *Let S ($f \geq I, f < I$) and T ($g \geq J, g < J$) be two iso-surfaces. $C(f, g)$ is an iso-contour on S defined as the interface between the regions of S: $g \geq J$ and $g < J$. $C(g, f)$ is similarly defined; the difference between the curves is how they bypass the tangency region.*

An important application is to extract feature lines e.g. parabolic lines, lines of extrema of the principal curvatures... Here the values of g are the corresponding differential expressions with iso-constant $J = 0$, while (f, I) defines a

usual iso-surface. Analogously, we can also compute the intersection of three iso-surfaces as well [17].

Once again, there is an ambiguity of forming bi-iso-segments in case of more than 2 bi-interpolated points. This has no matter for the 3D case; however in 4D it should be raised in a deterministic way, otherwise holes can be created. We propose to connect the consecutive points (this ensures that bi-iso-segments do not intersect) using the mean-value rule, which is to be applied on the bi-interpolated points.

3 4D extension

The good topological properties of the purely local Marching Lines algorithm are ensured with the help of interface *(1,4,5)* and orientation *(2,3)* conventions. If we want to extend this algorithm, we have to find a dimension-independent generalization of the orientation conventions, while the other conventions can be used as they are.

3.1 Orientation by induction

A discrete nD image is considered as a regular nD grid composed of *hypercubes* i.e. 2^n*-cells*, which is formed by 2^n adjacent voxels. Both the hypercube and its orientation is defined inductively:

Definition 6 (hypercube)

0D *point*

1D *unit line-segment*

nD *(n-1)D hypercube is translated along the nth mutually perpendicular direction with a unit vector tracing out an nD hypercube.*

Definition 7 (orientation of hypercube)

0D *without orientation*

1D *arbitrary vector direction*

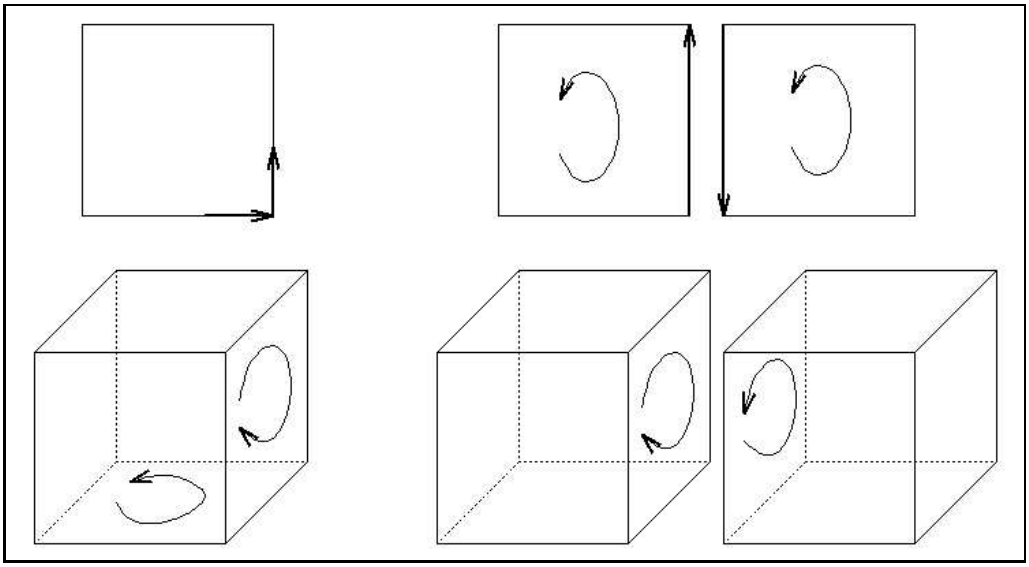

Figure 3: *top:* Orientation in 2D: There is one incoming and one outgoing edge at each voxel *(left).* $\Longleftrightarrow$ The same edge has opposite directions in adjacent squares *(right).*
bottom: Orientation in 3D: The orientation of faces having a common edge is opposite *(left).* $\Longleftrightarrow$ The same face has opposite orientations in adjacent cubes *(right).*

nD *The orientation of two (n-1)D hypercube, having a common (intersected) (n-2)D hypercube, should be opposite in the given nD hypercube.*
Which is equivalent to:
The intersection of two adjacent nD hypercube, i.e. a (n-1)D hypercube should have opposite orientation in the two nD hypercube.

We can see with the help of Fig. 3 what these definitions mean in case of a 2D square and a 3D cube. (If the reader is interested in more detail the geometry of polytopes, he/she is referred to an excellent introduction [2].) Once the direction of any one line-segment is fixed, then the orientation of hypercubes in an nD grid is determined. To get a coherent generalization of the orientation

conventions used in the Marching Lines algorithm, we complete this description with the following convention (see also Fig. 4):

Definition 8 (fixing the orientation) *We direct the first line segment so as to regain the left-hand orientation for squares (and consequently the outside orientation for cubes).*

We remark that the property of opposite orientation of adjacent faces is used implicitly in the proof of correctness of the Marching Lines algorithm.

Not only can these definitions be defined inductively, but also the structure of the nD generalized algorithm can be recursively given. In the sequel, we restrict ourselves to the 4D case, though the statements are valid for any dimension n.

3.2 Reducing the complexity

It is convenient to number the voxels of a hypercube in such a way that adjacent voxels have labels whose binary codes differ in 1 bit. Fig. 4 shows an example for the 3D and 4D case. The hyper-faces (4-side for 3D, 8-cell for 4D case) can be obtained from the equation of the corresponding hyper-plane. The following notations are employed:

3D cube

	0	1
z	0	5
y	1	4
x	2	3

4D hypercube

	0	1
t	0	7
z	1	6
y	2	5
x	3	4

which is to be read as e.g.: face numbered by 0 (5) of a 3D cube is given by equation $z = 0$ (1). Naturally, hyper-faces of an nD hypercube are equivalent to (n-1)D hypercubes; we call the one coded by 0 as the *base hyper-face*. The order of vertices in a hyper-face specified by the equation of the corresponding hyper-plane does not necessarily indicates the correct orientation. That is why the hyper-faces coded by odd numbers should be reflected as Fig. 5 shows,

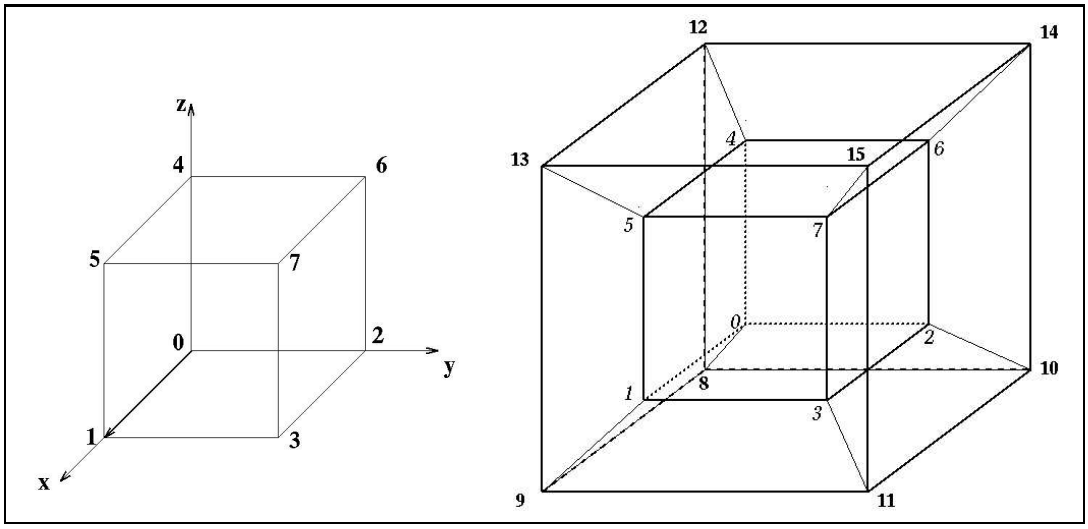

Figure 4: Voxels of a 3D cube *(left)* and a 4D hypercube *(right)* are numbered according to 1-distance binary codes which are relative to a standard origin in the cell. Also, the direction of the $\vec{01}$ edge is fixed to define the orientation coherently with the left-hand / outside conventions.

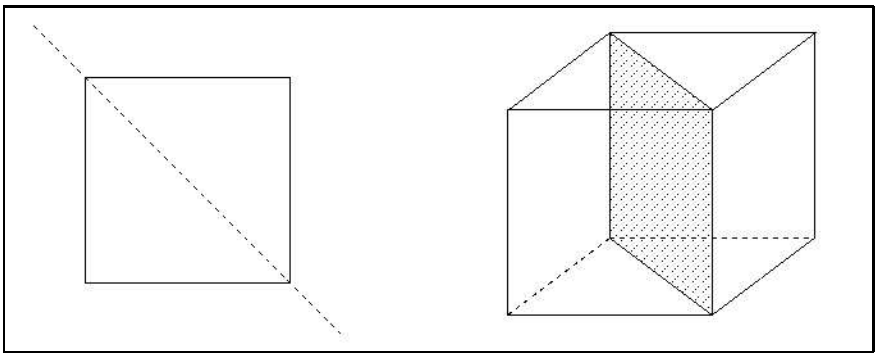

Figure 5: Reflection axes of a 2D square *(left)* and a 3D cube *(right)*.

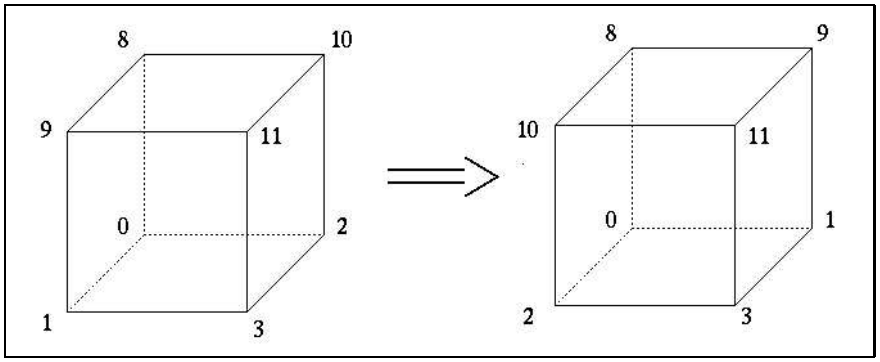

Figure 6: Vertices of the face-cube numbered by 1 given by the equation $z = 0$, before *(left)* and after reflexion *(right)*.

which establishes an *orientation-preserving bijection* between the base and the other hyper-faces. For example, the face-cube numbered by 1 contains the vertices *{0, 1, 2, 3, 8, 9, 10, 11}* given by the equation $z = 0$ (see Fig. 6), but this specified order of vertices indicates a wrong orientation. After reflection we get *{0, 2, 1, 3, 8, 10, 9, 11}* which is orientation-equivalent to the base cube if we consider *{0, 2, 1, 3}* as the base face. Moreover, the bijection re-establishes the original orientation of this face-cube, which is opposite to the one of the base cube in the sense that the common face is inversely oriented ($0 \rightarrow 1 \rightarrow 3 \rightarrow 2$ in the base cube and $0 \rightarrow 2 \rightarrow 3 \rightarrow 1$ in the face cube 1). As a consequence, we can trace back any hyper-face of an nD hypercube to a base (n-1)D hypercube in an orientation-preserving way.

Taking all these into account, *hyper-iso-patch* cycles (i.e. 3D manifolds in 4D) in a 16-cell can be computed as follows:

- for all the 8 face-cubes
  - substitute the voxels of the base cube with the voxels of the given face-cube in the proposed orientation-equivalent way,
  - compute the iso-patch cycles in the base 8-cell as described in the 3D Marching Lines algorithm,

- organize the obtained cycles (1-chains) into hyper-cycles (2-chains).

Cycles or 1-chains are oriented, not necessarily planar polygons, while hyper-cycles or 2-chains are oriented, not necessarily 3D polyhedra whose faces are exactly the above mentioned cycles. For a complete definition the interested reader is referred to [2, 7]. We emphasize that not only the cycles, but also the reconstructed hyper-cycles are oriented, and this orientation is coherent with the orientation of the corresponding hypercubes. That is, adjacent cycles composing the hyper-faces of a hyper-cycle have opposite orientation, and likewise: hyper-cycles have opposite orientation in adjacent 16-cells.

The intersection of two hyper-iso-surfaces can be similarly reduced to the computation of *hyper-bi-iso-lines* (i.e. 2D manifolds in 4D):

- for all the 8 face-cubes
  - substitute the voxels of the base cube with the voxels of the given face-cube in the proposed orientation-equivalent way,
  - compute the iso-patch cycles then the bi-iso-segments in the base 8-cell as described in the 3D Marching Lines algorithm,
- organize the obtained segments into hyper-segments (1-chains).

A simple example for hyper-bi-iso-segment calculation is demonstrated in Fig. 7.

The proof of correctness of this algorithm is based on the inductive definition of hypercube orientation and the proposed orientation-preserving reduction.

0 By induction, it follows:

Each edge has opposite orientation in adjacent 4-sides, so:

4-side	8-cells
8-cell	16-cells

Which is equivalent to:

Adjacent 16-cells have opposite orientation, since:

8-cells
4-sides

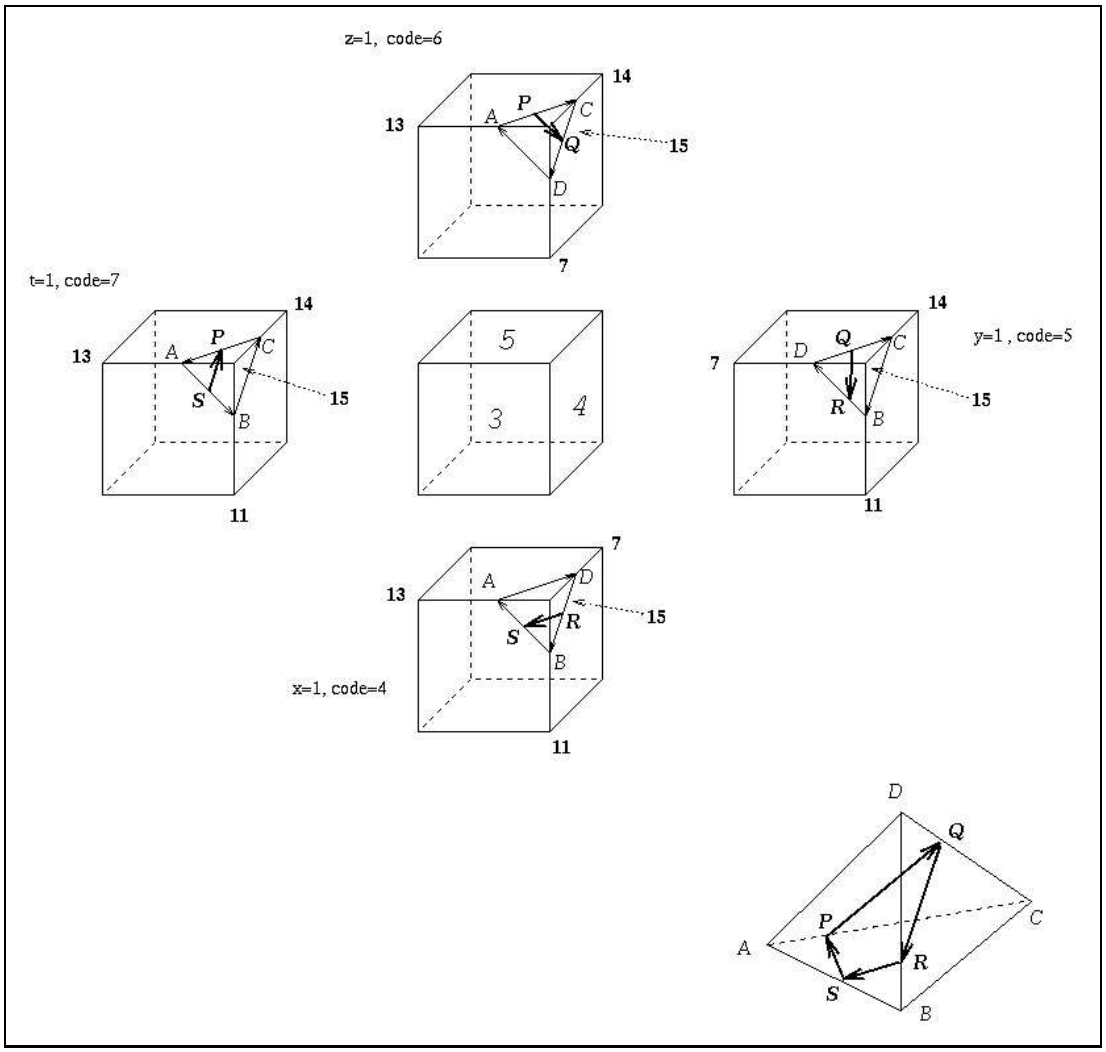

Figure 7: Computing a hyper-bi-iso-segment in a simple case when only the 15th voxel is labeled to be positive.
(left:) The inner cube is the base cube; to the left, the cube numbered 7 is placed. The other three cubes are adjacent to the marked (i.e. front, right, top) faces of the base cube. The corresponding iso-patches and bi-iso-segments are drawn in each cube.
(bottom right:) The iso-patch cycles and also the bi-iso-segments are properly organized due to the correct orientation. The reconstructed hyper-cycle is oriented; its adjacent hyper-faces (here: triangles) have opposite orientation (e.g.: $A \to D \to B \to A$ and $B \to D \to C \to B$) due to the opposite direction of their common edge (e.g.: $D \to B$ in cube 4 and $B \to D$ in cube 5).

1 Let us consider an 8-cell (this part of the proof is quite similar to the one of the Marching Lines algorithm). The procedure which computes the interpolated points is deterministic. So if we find a point P on a 4-side, we must find it (except at the boundary) on an adjacent 4-side. The shared edge has opposite direction in the two 4-sides, so point P has opposite labels. Since the iso-segments are directed from points labeled - to points labeled +, there is one incoming and one outgoing iso-segment for each point (except at the boundary); observe the analogy with Fig. 2, top left. Thus the iso-segments are connected and directed, and they form oriented iso-patches in an 8-cell. Note, that the orientation of iso-patches is consistent to the orientation of a 2D square.

To find bi-iso-segments on an oriented iso-patch, we first compute their bi-interpolated endpoints. These endpoints are labeled with the labels of the endpoints of the corresponding edges of the iso-patch. The number of bi-interpolated endpoints is even, and by construction they are alternatively labeled to be - and +. A bi-iso-segment is formed from a point labeled - to the next point labeled + (this ensures that bi-iso-segments do not intersect), in the direction determined by the mean-value rule (this will ensure that no holes are created in 4D; see the 3rd point).

2 Adjacent 8-cells have opposite orientation, i.e. their shared 4-side has opposite orientation in the two 8-cells. So the iso-segment computed on the shared 4-side is inversely directed in the two 8-cells, as a consequence, iso-patches formed in the adjacent 8-cells are also oppositely oriented. Hence iso-patches constitute an oriented surface i.e. 2D manifolds in 3D.

Following bi-iso-segments on adjacent 8-cells is easy. The adjacent iso-patches have opposite orientation, and the computation of bi-interpolated endpoints is deterministic. So we must find a bi-interpolated endpoint on exactly two adjacent iso-patches with opposite labels. Analogously to iso-segments on adjacent 4-sides (1st point), there is one incoming and one outgoing bi-iso-segment for each bi-interpolated point. Hence bi-iso-segments constitute directed and closed lines (if they do not reach the boundary) which do not intersect, i.e. we get 1D manifolds in 3D.

3 Now let us consider a 16-cell. Similarly to the 2nd point, it can be easily seen that iso-patches computed independently in adjacent hyper-faces i.e. 8-cells are oppositely oriented. So they can be organized into oriented hyper-iso-patches, whose orientation is consistent to the orientation of a 3D cube. The orientation-preserving bijection between the base and the other hyper-faces supports a simplified implementation: by tracing back any hyper-face to the base-face the same procedure can be used to compute iso-patches, while insuring that the orientation of inscribed iso-patches is coherent to the orientation of the corresponding hyper-face. The correctness of the bijection can be explicitely verified by enumerating the 8 different cases.

About hyper-bi-iso-segments on an oriented hyper-iso-patch: Similarly to the second part of the 2nd point, we have that bi-iso-segments computed independently on iso-patches of adjacent hyper-faces constitute closed, non-intersecting cycles i.e. hyper-bi-iso-segments. Since these bi-iso-segments are directed in such a way that there is one incoming and one outgoing bi-iso-segment at each shared edge of adjacent iso-patches. Note, that the orientation of hyper-bi-iso-segments is consistent to the orientation of a 2D square.

4 Finally, it follows from the fact that adjacent 16-cells have opposite orientation, that hyper-iso-patches formed in adjacent 16-cells are also inversely oriented; its proof is analogous to points 2 & 3. Hence hyper-iso-patches constitute an oriented hyper-surface i.e 3D manifolds in 4D.

Likewise, hyper-bi-iso-segments obtained in adjacent 16-cells are oppositely oriented, since their circumscribed hyper-iso-patches have also opposite orientation. Thus hyper-bi-iso-segments constitute oriented hyper-lines i.e. 2D manifolds in 4D.

At last we note that the intersection of three (four) hyper-iso-surfaces can be similarly computed, giving lines, i.e. 1D manifolds in 4D (points, i.e. 0D manifolds in 4D).

4 Some scale-space applications

Beginning with Witkin's pioneering paper [19], multiscale analysis and applications have become wide-spread. The main idea of scale-space theory [11] is to treat an image at different levels of scale, applying logarithmically increased smoothing. At a coarse scale, only the most characteristic structures of the original image are conserved, however, they are delocalized and may have undergone important topological changes (splitting, merging, disappearing). To get precisely located and significant structures, they should be tracked from coarse to fine scale. In practice, sampling, delocalization and topological changes make this task difficult. A well-known scale-space property is that "no spurious detail is generated" when the scale is increased (i.e. fine $\rightarrow$ coarse). Now we examine three points related to iso-surface extraction and scale space; we will refer to the previous sentence as "simplification property".

4.1 About topology

We cannot guarantee that the topology of a reconstructed discrete iso-surface is exactly the same as the one of the original continuous object: All the information represented at a scale smaller than the voxel size is lost after discretization; moreover, the connectivity varies with the extraction scale, as the 2D dumpbell example shows in Fig. 8. Lots of techniques have been proposed (mean-value, gradient heuristics, ... see [10] for a survey) to correctly determine the topology, but non of them take into account that the border changes with scale and derivatives must be taken in some neighborhood whose size influences the result. A suitable attempt to treat this problem is to extract and consider iso-surfaces at increasing levels of scale i.e. to analyze a 4D hyper-iso-surface. If two parts of a real object are connected by a thin, short bridge, and this bridge is lost after image discretization, the connectivity may be re-established at a slightly higher scale; however to find this scale is still an open problem.

4.2 Following features across scale

Theoretical results about the behavior of *differential singularities* – features which can be defined as zero-crossings of differential expressions – have been

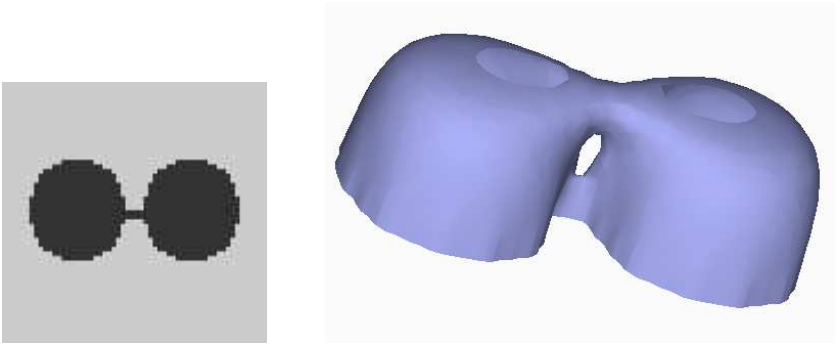

Figure 8: Original image *(left)* and its multiscale hyper-iso-contour i.e. a 3D iso-surface *(right)*. The coordinates of points are marked in the (x, y) plane, while resolution is measured logarithmically via the z axis. The delocalization effect can be immediately investigated. We can also observe that the connectivity of the iso-contour is not preserved: in the course of blurring, it splits into two, then merges and then splits again.

obtained only in the continuous case [12, 1, 6, 14, 8]. In practice, these are difficult to employ, discretization effects the detection, so we need to exemplify the orbits of some features in scale space to get a statistical overview. In previous articles [5, 4] we have described a multiscale representation method based on iso-surface detection, that we have applied to follow 2D *corner points* (absolute maxima of the iso-phote curvature) at increasing scales and analyzed their evolution. Now we have a method to investigate the scale-space behavior also of 3D characteristic lines like parabolic or *crest lines* (absolute maxima of the largest principal curvature). Parabolic lines are used e.g. to compute aspect graphs of smooth surfaces [15], while crest lines are already established as significant features in medical image processing [18].

Multiscale extraction, based on zero-crossings i.e. (hyper-)iso-surface detection, is preferable to local extrema search on each level of resolution followed by pairing between adjacent levels. Since simply matching features extracted at different levels can give false pairing due to possible strong delocalizations. However, in the case of (hyper-)iso-surface detection, the parent-child connec-

tion of singularities is established directly from the voxel structure, which ensures that the topological changes are automatically followed. Fig. 9, 10 show the change of parabolic lines, where the connectivity information is obtained during the 4D extraction. Incorporating simplification properties into the extraction avoids unnecessary computations making the algorithm faster. The interested reader can find a detailed description of this specialization in [4].

Tracking crest lines across scale is difficult compared to other differential characteristic. Crest lines are not defined in case of zero gradient or at umbilic points, that is why the Marching Lines algorithm gives not necessarily closed lines. Practically it means that we have to stop the marching at one scale but we have to continue at an other scale, while processing the same position (since zero gradient or umbilicity are singularities not only in space but also in scale); as a consequence, the extracted bi-iso-segments are degenerated to a point. Work is still in progress to find a proper solution to this problem. First results about multiscale following are presented in Fig. 11, 12.

4.3 A high-level task

Over the past decade, many digital anatomical atlases (see for example [9]) have been developed, whose main application fields include: reference database, normalized registration, shape and deformation analysis, etc. An atlas has to take into account the *resemblances* of the underlying organs without considering slight individual *differences* to constitute anatomically meaningful features. To build for example an "average" skull [16], based on automatically extracted differential invariants, the following scheme is used: feature extraction and registration, identification and average of the *common* feature subset and feature deformation analysis. Special attention has to be taken to find common features with both correct *topology* and high *precision*. Though the results are encouraging, the reliability of the method can be augmented using features extracted at multiple scales and applying the registration algorithm hierarchically from coarse to fine scale. In fact, a balance between similar - different, common - individual, topology - precision should be solved, which is just the same as the compromise between detection - localization in scale-space theory. We think, that the extraction of 3D characteristic curves at increasing

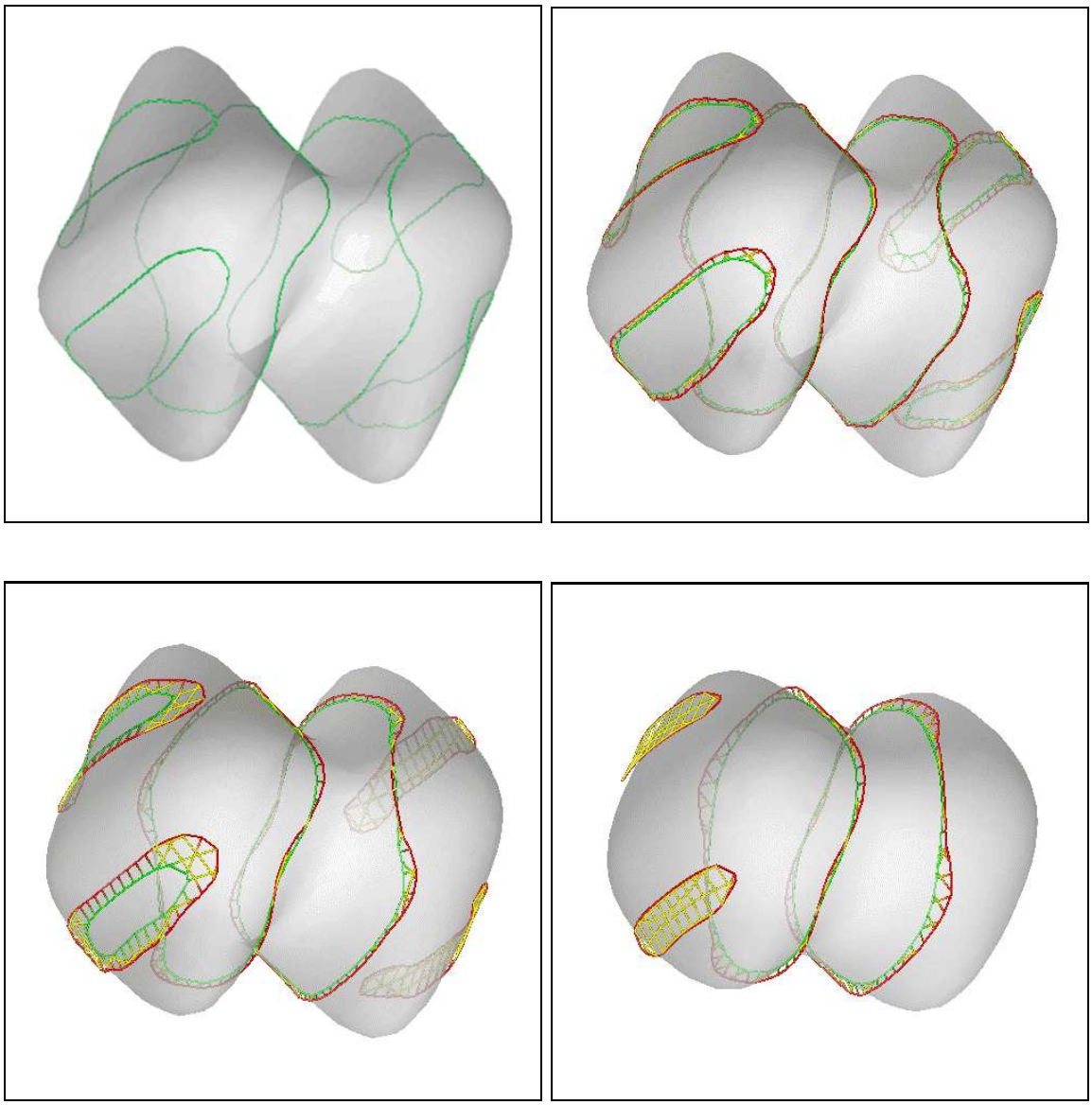

Figure 9: Change of parabolic lines with connectivity information.
Extraction scales are $\sigma_i = \varepsilon\tau^i$ where $i = 0, \ldots 5$, $\varepsilon = 1.0$ and $\tau = \sqrt{2}$. *(1)* Iso-surface at scale $\sigma = 1.0$, with parabolic lines in green obtained at that scale. *(2 - 4)* Iso-surfaces at scales $\sigma = \sqrt{2},\ 2,\ 2\sqrt{2}$, with parabolic lines: in green obtained at that scale, in red obtained at the previous scale and connecting segments in yellow.

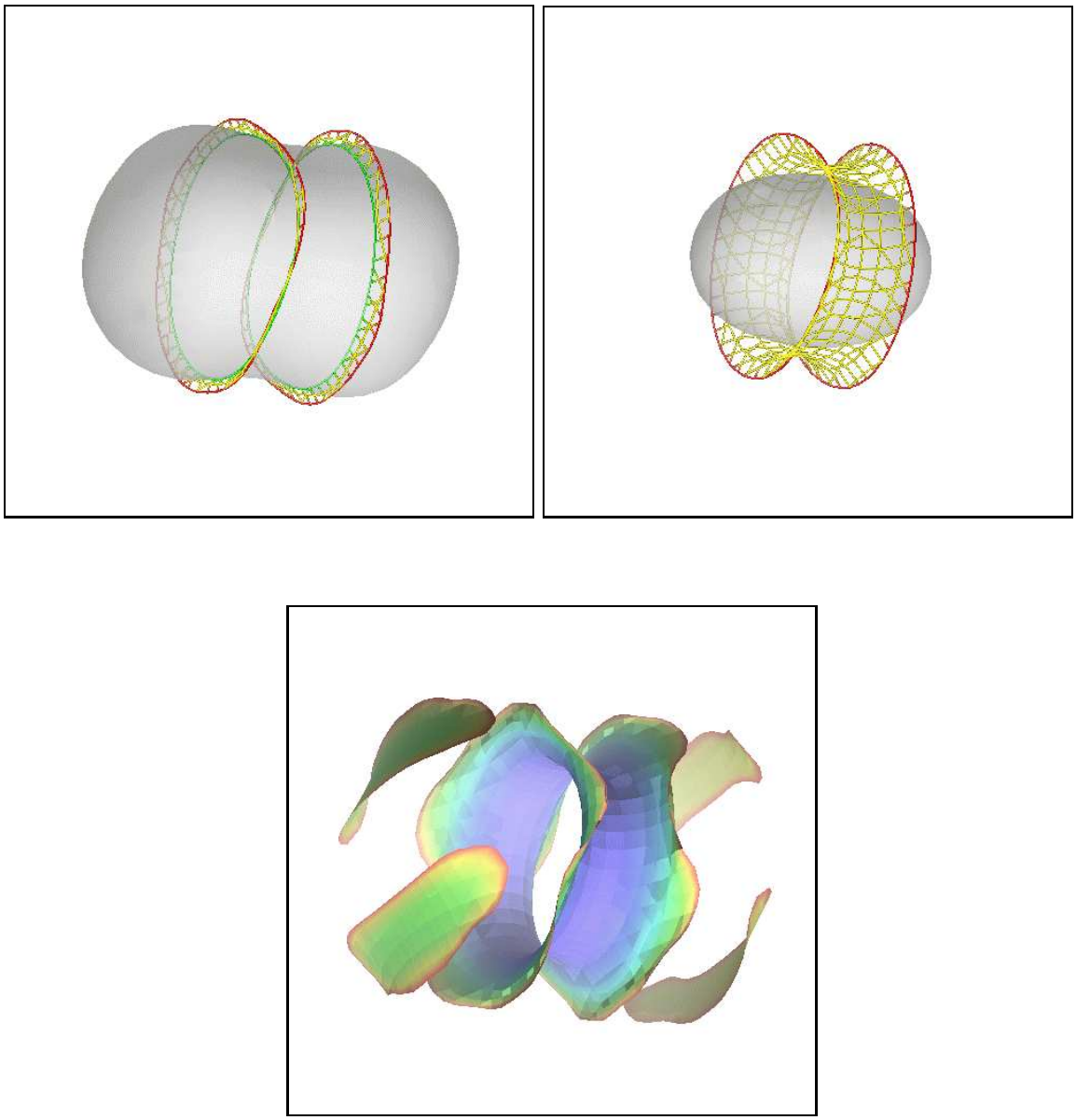

Figure 10: Change of parabolic lines with connectivity information (continued).
(1 - 2) Iso-surfaces at scales $\sigma = 4,\ 4\sqrt{2}$, with parabolic lines: in green obtained at that scale, in red obtained at the previous scale and connecting segments in yellow. *(3)* Change of lines via scales which is represented by colors from red to purple.

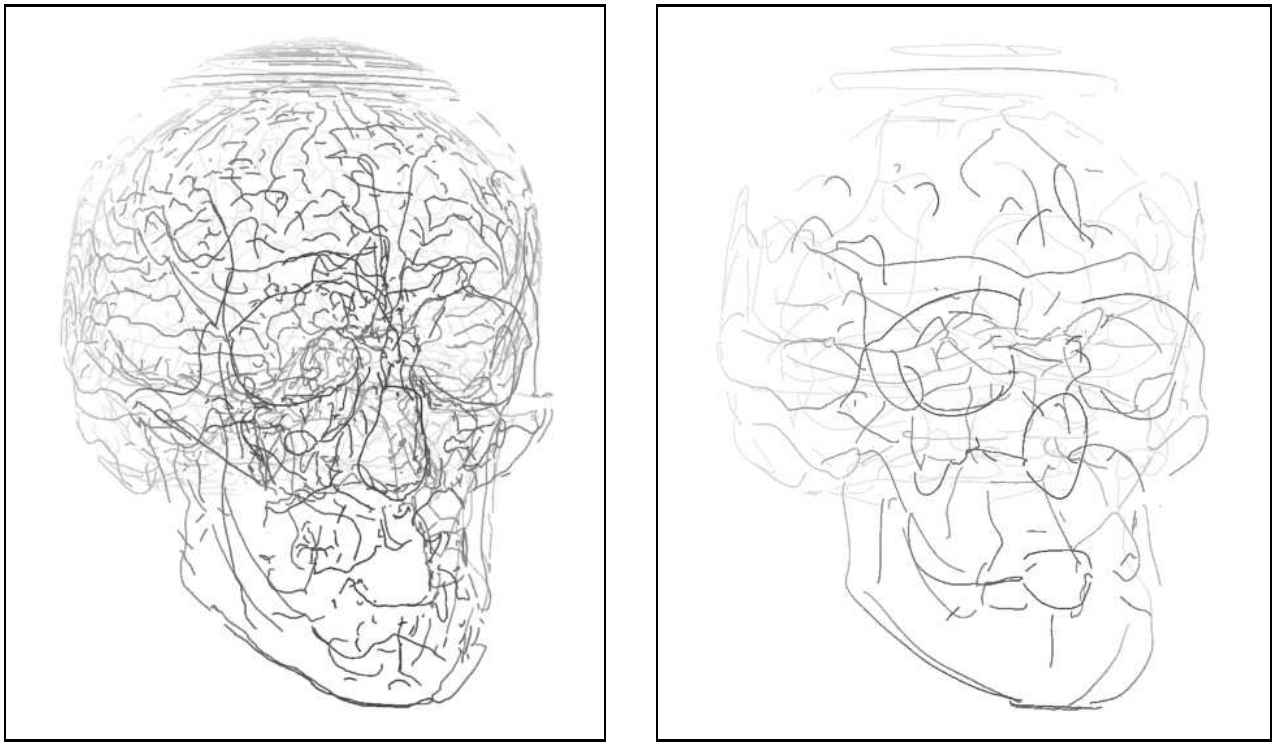

Figure 11: Crest lines of a segmented, binary image of the skull.
(left:) Extraction scale is $\sigma = 2$. We can observe that there are lot of artifactual curves due to discretization.
(right:) Extraction scale is $\sigma = 6$. Only the most significant curves are preserved, however they are not at all precisely located (e.g. the nose). Moreover, important topological changes have taken place (e.g. the orbit of the eye).

scales with the previously summarized method can substantially help to create useful anatomical atlases.

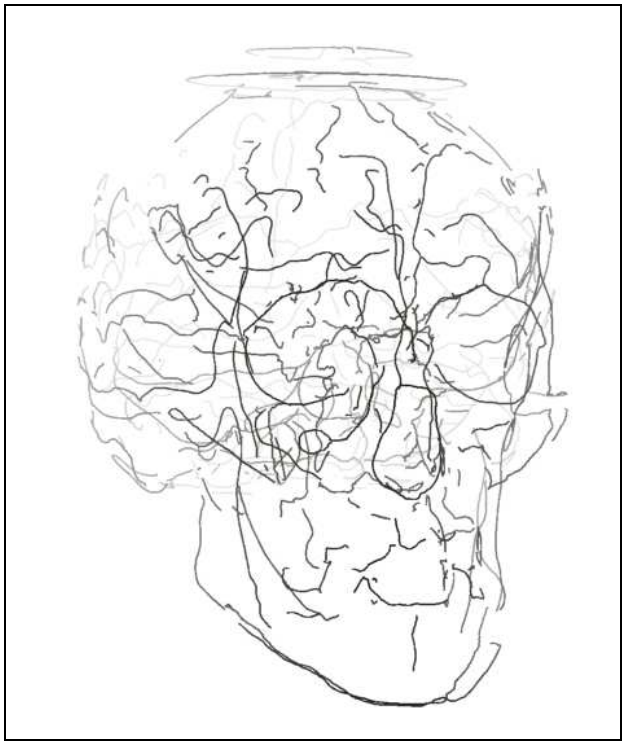

Figure 12: Crest lines of a segmented, binary image of the skull (continued). Only the lines extracted at $\sigma = 2$ and having a descendant at $\sigma = 6$. These lines are more characteristic than the complete set at $\sigma = 2$.

5 Implementation details

One can argue that processing 4D images cannot effectively be done in practice, because of the high computation time and the large amount of memory needed. However, the program can be structured in such a way that only two adjacent levels of a 4D image (i.e. two 3D images) are processed at the same time, profiting from the fact that the 4th coordinate is essentially different from the other three spatial coordinates. For example, in multiscale applications or when following time evolution, we can suppose that the adjacent levels are

fairly related. Moreover, thanks to the simplification property in scale space, the levels at the two finest scale contain much more details than the rest. (We are not sure that similar statements exist in case of time evolution.) So intensive computation is only needed at two neighboring levels, corresponding to the highest scales, to start the algorithm, then the obtained (hyper-)cycles are to be propagated for the next levels, that is, we have to process the rest voxels only if they are on a reached 16-cell. However, this way new emerging events are not necessarily detected, contrary to an intensive sequential scan. A possible variant, which is more useful in practice, is the coarse-to-fine propagation: in this case we extract features at the finest scale only if they have a descendant at the coarsest scale. We control the propagation with two temporary queues (one for the actual level, one for the next), as it is summarized below.

To start (e.g. at the two finest / coarsest scales), we make an extensive computation. For each 16-cell:

- If a hyper-cycle containing a cycle in the face-cube numbered 7 is found, we store into `nextQueue` the standard origin of the adjacent 16-cell. (This ensures the propagation to the next level; `actQueue` remains empty.)

To proceed level-by-level, we first take `actQueue` := `nextQueue`, `nextQueue` := `empty`. While there is an elem in `actQueue`:

- We take it out and calculate hyper-cycles in the corresponding 16-cell.
- If a cycle in face-cubes numbered 1..6 is found, the standard origin of the adjacent 16-cell (if it is not stored yet) is stored into `actQueue`. (This is necessitated by topological changes between two consecutive levels.)
- If a cycle in the face-cube numbered 7 is found, the standard origin of the adjacent 16-cell is stored into `nextQueue`.

As we have seen, the 4D algorithm can be traced back to 3D calculations via orientation-preserving bijections. The transformation of hyper-faces can be obtained by a look-up table procedure, as opposed to solving equations. This greatly speeds up the 4D $\rightarrow$ 3D reduction; similarly, other topological information like edge and face connection can also be stored in look-up tables.

Finally, a remark: the results are rather difficult to visualize; the best way found so far is animation and extensive use of colors.

6 Conclusion

We have presented a method to extract 4D hyper-iso-surfaces and their intersection curves, which is a natural extension of the 3D Marching Lines algorithm with new orientation and implementation considerations. Though our main discussion and also the implementation have been done in 4D, all the statements are valid for any dimension. We have also shown some possible applications related to scale space. Currently, an important task is to follow 3D characteristic curves, such as crest lines, across scale. In the future we intend to investigate the multiscale behavior of these lines so as to get robust detection and good base for high-level medical image processing tasks. Also, we plan to apply this method to analyze gated SPECT images of the beating heart.

7 Acknowledgment

We wish to thank Alexis Gourdon and Jean-Philippe Thirion for stimulating discussions about the Marching Lines algorithm, also to Hervé Delingette and Stéphane Cotin who have drawn our attention to some results of topology. We thank Bruce Latimer, Director at the Cleveland Museum of Natural History, Court Cutting, David Dean and André Guéziec for the CT-scan data of the skull. This work was partially supported by a fellowship from "Ministère de l'Enseignement Supérieur et de la Recherche".

References

[1] Haruo Asada and Michael Brady. The curvature Primal Sketch. *IEEE PAMI*, 8, 1986.

[2] H.S.M. Coxeter. *Regular Polytopes*. Dover Publications, 1973.

[3] Márta Fidrich, Jacques Feldmar, and Jean-Philippe Thirion. Topological Changes in Scale Space as a Function of Image Transformations. In Theo Moons, editor, *Computer Vision and Applied Geometry*, LNCS, Nordfjordeid, Norway, August 1995.

[4] Márta Fidrich and Jean-Philippe Thirion. Multiscale Extraction of Features from Medical Images. In V. Hlaváĉ and R. Šára, editors, *International Conference on Computer Analysis of Images and Patterns*, volume 970 of *LNCS*, pages 637–642, Prague, September 1995.

[5] Márta Fidrich and Jean-Philippe Thirion. Multiscale Representation and Analysis of Features from Medical Images. In N. Ayache, editor, *International Conference on Computer Vision, Virtual Reality and Robotics in Medicine*, volume 905 of *LNCS*, pages 358–364, Nice, April 1995.

[6] John M. Gauch and Stephen M. Pizer. Multiresolution Analysis of Ridges and Valleys in Grey-Scale Images. *IEEE PAMI*, 15, 1993.

[7] Marvin J. Greenberg and Harper John R. *Algebraic Topology: A First Course*. The Benjamin/Cummings Publishing Company, 1981.

[8] Lewis D. Griffin and Alan C. F. Colchester. Superficial and deep structure in linear diffusion scale space: isophotes, critical points and separatrices. *Image and Vision Computing*, 13(7):543–557, Sept 1995.

[9] K. H. Höhne, A. Pommert, M. Riemer, Th. Schiemann, R. Schubert, and U. Tiede. Framework for the generation of 3D anatomical atlases. In Richard A. Robb, editor, *Visualization in Biomedical Computing*, volume 1808 of *SPIE*, pages 510–519, Chapel Hill, North Carolina (USA), October 1992. SPIE.

[10] Alan D. Kalvin. A Survey of Algorithms for Constructing Surfaces from 3D Volume Data. Technical Report RC 17600, IBM Research Division, January 1992.

[11] Jan J. Koenderink. The structure of Images. *Biological Cybernetics*, 50:363–370, 1984.

[12] Tony Lindeberg. Scale-space behaviour of local extrema and blobs. *Journal of Mathematical Imaging and Vision*, 1:65–99, March 1992.

[13] William E. Lorensen and Harvey E. Cline. Marching Cubes: A High Resolution 3D Surface Reconstruction Algorithm. *Computer Graphics*, 21(4), July 1987.

[14] Farzin Mokhtarian and Alain K. Mackworth. Scale-Based Description and Recognition of Planar Curves and Two-Dimensional Shapes. *IEEE PAMI*, 8, Jan 1986.

[15] A. Noble, D. Wilson, and J Ponce. Computing Aspect Graphs of Smooth Shapes from Volumetric Data. In *SIAM Workshop on Mathematical methods in Biomedical Image Analysis*, San Francisco, California USA, June 1996.

[16] G. Subsol, J.Ph. Thirion, and N. Ayache. A General Scheme for Automatically Building 3D Morphometric Anatomical Atlases: application to a Skull Atlas. In *Medical Robotics and Computer Assisted Surgery*, pages 226–233, Baltimore, Maryland (USA), November 1995.

[17] Jean-Philippe Thirion. New Feature Points based on Geometric Invariants for 3D Image Registration. Technical Report 1901, INRIA, April 1993. (Accepted for publication in IJCV).

[18] Jean-Philippe Thirion and Alexis Gourdon. Computing the Differential Characteristics of Isointensity Surfaces. *CVGIP*, pages 190–202, March 1995. (also a Tech. Report n^o 1881).

[19] Andrew P. Witkin. Scale Space Filtering. In *Proc. Int. Conf. Artificial Intelligence*, volume 511, 1983. Karlsruhe.

Unité de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unité de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unité de recherche INRIA Rhône-Alpes, 46 avenue Félix Viallet, 38031 GRENOBLE Cedex 1
Unité de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unité de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
ISSN 0249-6399