Registration of medical images for surgical action : use of optical sensors and matching algorithm

M. Richard*, M. Fleute*, L. Desbat*, S. Lavallée* & J. Demongeot*‡,
TIMC-IMAG* & IUF‡
Faculty of Medicine
University J. Fourier of Grenoble
38 700 La Tronche France

Summary

We propose in this paper to deal with the notion of medical image registration : both physicians and surgeons need in the patient room or in the operation theater images of the medical target to reach and treat ; these images can be very precise and come from a pre-operative acquisition device (NMR, CT-scanner,...) or from a per-operative set of sensors (active or passive infra-red sensors, laser acquisition, numerical 2D X rays, 3D echography,...). The main problem is then how to compare these multi-modal images having different resolutions and extracting different features (anatomic or functional) from the patient medical reality ? In a first Section, we present the hard side with essentially optical sensing methods ; after, in a second Section, we present an example of matching algorithm built in order to compare and super-impose pre- and per-operative images. This algorithm computes a confinement tree thanks to the watershed lines and we explain under which conditions the confinement tree can be an invariant parameter of the image for some topological transformations. We propose a simplification algorithm for this tree. Then, for two topologically closed images, for which the simplified trees are the same, we show how to match it, and how it is possible to consider those two matched trees as an initial condition for matching the remaining nodes of the initial trees. Then, a spline interpolation can be used in order to obtain the continuous transformation from the first image to the second one, using very few hypotheses. Finally, we describe several medical actions or surgical operations for which a physician or a surgeon needs such registration tools.

0. Introduction

A physician or a surgeon needs in order to perform medico-surgical actions in the patient room or in the operation theater a lot of information about the best opening zone and the best way to reach the final target, minimizing the invasive iatrogenic impact of the operation. The present danger is the overflow of information, leading to bad decisions because of a lack of synthesisbetween clinical data, multimodal pre-operative imaging and per-operative observation (visual or indirect through medical sensing and imaging).

We can decompose the work of the physician or of the surgeon into three main steps :

- to perceive, either directly by visual observation, or by looking on a screen at 3D-imaging structures reconstructed from pre-operative acquisitions (anatomic : CAT-scanner (Computer Aided Tomography), NMR (Nuclear Magnetic Resonance) or functional : PET-scanner (Positron Emission Tomography), MEG (Magneto-Encephalo-Graphy,...) or at 3D- and 2D-imaging coming from per-operative observations through numerical X-rays and echography. This perception step, if it is too complex because of its multimodal character, has to be aided by augmented reality techniques (i.e. by indications given by other ways than the visual one : auditive, epidermic, lingual [P. Bach y Rita et al. 1998 and in the present volume],...) and also by combining on a screen synoptic views of different images of the same object of medical interest (e.g. a 3D view presenting the final target with the best trajectory reaching it super-imposed on a per-operative 3D-echography)
- to decide in real time what is the best gesture to do : the physician or the surgeon can decide himself his own action, or follow up a direction given by a laser beam, or leave his hand to be guided through a succession of virtual cones of admissible local translation motions (such a virtual cone he can force to

In *Sixth International Conference on Education and Training in Optics and Photonics*,
J. Javier Sánchez-Mondragón, Editor, SPIE Vol. 3831 (2000) • 0277-786X/00/$15.00

escape or respect to remain into virtual boundaries given by a sound, a light, or an epidermic or lingual signal transmitted from a control computer, which compares the present trajectory to the pre-decided one)
- to perform the medical action (kidney puncture, pericardic puncture,...) or the surgical one (installing electrodes in the brain, putting screws in the back bone, inserting ligaments in the knee,...), respecting the above proposed real time decision, taken in agreement with an eventual pre-operative strategy.

All these steps need the use of both local optical sensors (able to help the registration between the 3D spatial referential corresponding to the pre-operative localization of the objects of medical or surgical interest to puncture or to treat, and the 3D spatial referential corresponding to the patient body on the operation table) and matching algorithms (able to calculate the – rigid or not – transformation relating these two referentials).
In the following, we will present rapidly, in Section 1., active and passive optical sensors used in patient room or in operation theater, and, in Section 2., an example of matching algorithm, based on the simple intuitive idea that a medical image, made of a collection of pixels provided with grey levels, is just a geographic landscape we can fulfill with water, thus detecting a succession of basins, we can connect together at singular points called saddle points (analogous to mountain passes). In order to match two images, we can then identify their saddle points trees and (if they are at least 3 for a rigid deformation) we can reconstruct the operator permitting to pass from a 3D spatial referential to another. Finally, in Section 4, we will present some medical and surgical applications realized by the medico-surgical community and we will conclude about future realistic developments in the new research field represented by the CAS (Computer Aided Surgery) in terms of robots, holograms,...

1. Optical sensors

Medical optronics is dealing with a large variety of optical sensors :

- active infra-red sensors (photo-diodes) located on the top of surgical tools or of medical positioning devices (like echographic sources, Fig. 5), the emission being captured by CCD cameras (like Optotrak^{R} system – see Fig. 4 – or Polaris^{R} system – see Fig. 2 [Demongeot et al. in the present volume])
- passive infra-red sensors (catadioptric reflectors illuminated by an infra-red source)
- laser acquisition (laser beam deplaced over the skin, whose profile on the patient body is acquired by a CCD camera)
- microinterferometers [Demongeot et al. in the present volume]).

These optical sensors are used in the patient room or in the operation theater, in order to make more precise the location of invariant parts of the patient body, like face skin shape. This information is completed by the position, under the skin, of specific deep organs, acquired during the medico-surgical action by numerical X-rays (eventually 3D, see Fig. 7) or by echography. The couple between the surface and the depth information constitutes the per-operative 3D-referential. The corresponding pre-operative information is coming both from the segmentation of the NMR or CAT –scanner skin (made by using contrasting and contouring algorithms) and from the 3D spline reconstruction of deep objects of medical interest from the NMR or CAT-scanner 2D slices.

The medical optronics constitues an entire chain of acquisition and processing of medical data, capable to create the medical knowledge a surgeon or a physician needs for diagnosis or therapy purposes. The present tendency to give a holistic education in medical imaging and instrumentation is called "Model driven Acquisition" learning. The acquisition process is not isolated, but takes part of a whole programme of observing and modelling the medical reality, serving for example as a template for generating an augmented reality a physician and a surgeon can exploit in order to improve their classical diagnostic or therapic procedures. This augmented reality has to be communicated to the surgeon via an auditory, visual or lingual stimulus, forcing them to remain in the limits of a predefined region of surgical action (in which he is allowed to cut organs, to perform biopsy punctures or to pinch vessels).

2. Watershed matching algorithm

2.1. Introduction

The watershed line is a concept firstly defined by geographers in order to characterize the main features of a landscape : a drop of rain that reaches the ground will flow down to a sea or an ocean. In the case of France, the watershed line splits the country in two parts, the atlantic zone and the mediterranean zone. Those zones are called 'catchment basins', and the oceans are the minima of them. They define a partition of the relief (which separate minima), and the frontiers of catchment basins define the watershed line (see section 2.2.).We can easily understand the interest of this concept in image processing : gray level images can be considered as reliefs, and the watershed line is a good way to separate light zones from dark ones, or, applied to the gradient of the image, a way to locate contours on the set of inflexion points of the surface, also called Dupin's lines.

Thanks to the watershed line, we can compute a tree, called the confinement tree, which is invariant for topological transformations (see section 2.3. and 2.4. for the computation of the confinement tree). Our goal is then to match two trees, in a robust way. For this, we start with an initial condition (computed thanks to the noise properties of our images), which is a simplification of our tree, for which the matching is easlily possible. Then, we start our algorithm in order to match more complex trees (see section 2.5.).

2.2. Computing the n dimension watershed line

The watershed line is computed on discrete image. The method for computing this on discrete images is called immersion simulation, because we simulate an immersion of our image, and locate the watershed line on the meeting points of several catchment basins. First discrete algorithms of watershed computing by immersion simulation were proposed in [F. Meyer 92] and [Bertrand et al. 97a], [Bertrand et al. 97b], [Beucher et al. 93] for a discrete operator related to the watershed.

In our case, the watershed line is computed on the inverse image, in order to have one and only one local maximum (of the original image) into each catchment basin. Then the resulting labelling (still not a partition) is used on the original image. We use the Vincent and Soille algorithm (see [Vincent et al. 1991]) on discrete images with a linear complexity (about 7,25 N, where N denotes the number of pixels in the image), which can be used in n dimensions. The result of computation gives a partition of image, with labels from 1 to n_l the number of catchement basins. The label 0 is used for pixels for which the associated catchment basin cannot be determined. The 0 labeled pixels constitute a thin unconnected line (which could be considered as the discrete watershed line). When the maxima are separated into catchment basins, we need to know at what threshold level two neighboring maxima could be separated into two different threshold compacts. This level is obviously the highest point on the computed watershed line separating the two maxima (in the continuous case). In the discrete case, if the maxima are labelled m_1 and m_2, it is the highest point for the pixels of labels 0, m_1 and m_2 which are neighbors of a pixel with a different label but still 0, m_1 or m_2. That's why we can replace the label of every watershed pixel (i.e. 0 label) by the label of one of its neighbor arbitrarily chosen. This leads to a partition of the original image into n_l compacts.

2.3. The confinement or component tree

2.3.1. Confinement tree definition

Let's consider a 2D image on which a thresholding of level h is done. This procedure leads to a binary image, where pixels of a level less than h have the value 0, and those of a level greater than h a value of 1. The aforementioned pixels of value 1 define a finite number of connected components. When you consider the threshold of level h+1, obviously each new connected component is included into one of the components of the level h, the threshold of level 0 being the support of the whole image.

Thus, it is possible to define a tree, called component or confiner tree. The root of this tree corresponds to the threshold of level 0, and the nodes of depth h+1 are recursively defined from the nodes of level h. Let's consider the threshold of level h, which comprises k connected components (no matter if they are 4 or 8-connected) associated to k nodes in the tree. Let C be one of those component and N be the node associated to, let p be the number of components of the threshold of level h+1 which are included into C. Then we create p nodes in the tree, all of them sons of N, and every one associated to one and only one of the p components (if p=0, then N is a leaf) ([Bertrand et al. 97b], [Demongeot et al. 86], [Mattes et al. 98]). We have then defined the component tree, called like that because it is defined through the notion of connected component. But it is already possible to define it through the notion of 'confiner', a confiner being a contour line of a continuous relief. We call those curves 'confiners' because they are confining to local maxima when the level

is increasing ([Demongeot et al. 86], [Mattes et al. 99], [Meyer 92]). We can remark that the confinement tree is a particular case of the Reeb graph on manifolds ([Kergosien 92], [Sakarovitch 84]).

From the confinement tree, we can define the bifurcation tree, which is the confinement tree from which every one-descendant node is contracted.

2.3.2. Matching through component tree

This tree is a very useful concept in image processing, particularly for matching closed images, either for comparing two slices of the same object, or following the movement of mobile objects.

Isomorph partitionning

Given a partition on an image, we can build the partitionning graph associated to the partition. Each vertex of the graph is associated to one and only one cluster on the partition, and each edge to a neighboring relation between clusters. (see Fig. 1).

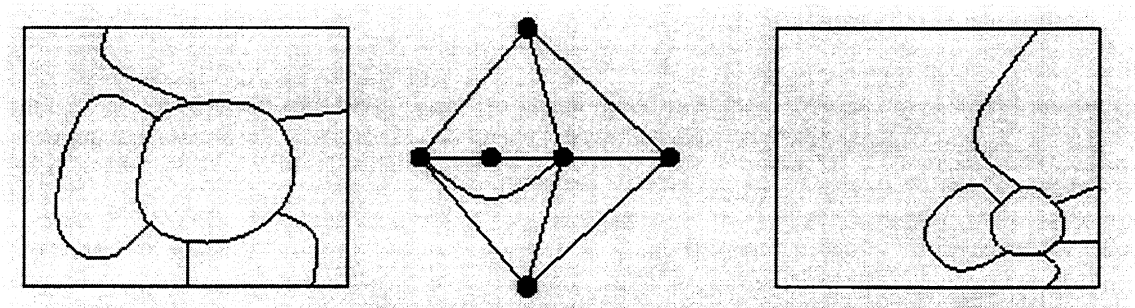


Fig. 1. *Two isopomorph partitions and their common partitionning graph*

To partitions are isomorph if their partitionning graph is the same. This relation between partitions gives a cluster to cluster isomorph transformation which preserves neighboring relation and which is an elastic matching procedure, i.e. whithout geometrical criteria.

hierarchical partitionning

A partition can be a tree (or hierarchy) when several neighboring clusters are grouped into one, defining a new and smaller partition. This new cluster is partitionned into the preceding smaller ones; hence it could be considered as the father in a tree of those clusters. Then, a hierarchical partition is made of a tree, and of a partition for every level in the tree (the root of the tree being the trivial partition which is the image itself). Given two hierarchical partitions, if the trees are the same and every level in the tree is isomorph to the same level in the other tree, we have a elastic matching procedure between the two images.

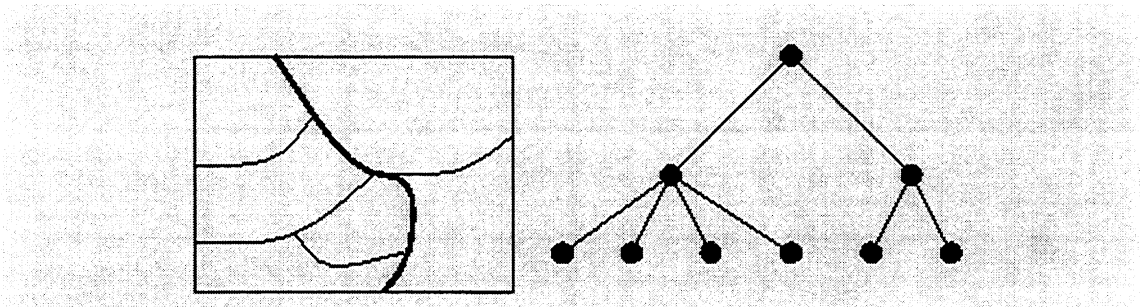


Fig. 2. *A hierarchical partition and its associated tree (each level of the tree is associated to a partition, and then to a partitionning graph)*

When the trees are not the same, or the different levels are not isomorph, we propose a procedure of tree and cluster matching by extraction of isomorph components, described in the following section. We will see that the component tree is the hierarchical partitionning tree of a very known partition, which is the watershed line on the inverse image (the watershed line on the inverse image separates local maxima). Hence, after computing the watershed line, we will see how to build the tree (called now the watershed tree), and how to match watershed trees, and then how to find cluster correspondance, which gives a point to point and cluster to cluster matching procedure.

The component tree is the tree of a hierarchical partition of the image. The following subsection describes the image continuous transformations which preserve isomorph hierarchical partitions.

2.3.3. Image transformations and tree transformations

2.3.3.1. Bicontinuous transformation

As a matter of facts (as shown in [Malpica et al. 97] and [Mattes et al. 99]), given two images *I* and *I'*, let's define a topological transformation as an application T: R^2 -> R^2 bijective and bicontinuous (e.g. T et T^{-1} are both continuous). Under certain assumptions (based on the conservation of certain physical characteristics), we have the same component tree for I and I'. It is due to the fact that, if f denotes the gray density function, we have: $f_{I'} = f_I \circ T^{-1}$.

Thus, if we consider that the point y of I at the time t moved according to the transformation T between t and t+1, we can write that $f_{t+1}(y) = f_t(T^{-1}(y))$, and then that $f_t(y) = f_0(T^{-t}(y))$, when T does not depend on t. This defines an autonomous process for which we can define its component tree, which is the component tree common to all images, if T is continuous ([Mattes et al. 99]).

Hence, it is possible to make a correspondence between the nodes of the tree of I and the nodes of the tree of I'. As each node is associated to a union of binary components, also called a cluster, it is possible to link the clusters of I and I' without any geometrical criterion. Thus, we can follow the movement of cells or the transformation of organs, or match slices of a 3D object if those slices are closed enough.

Let's mention that in actual practice, the noise on the images I and I' makes their confinement trees different. It is possible, then, to use the notion of distance between trees to realize the matching. The ideal case of a zero distance rarely occurs, even if images are filtered ([Mattes et al. 99], [Meyer 92]).

Let precise that the confinement tree of an image and its inverse have two independent trees. Hence, it is possible to have more matching points, when we apply our matching algorithm twice (on the image and its inverse).

2.3.3.2. Gray level transformation

We can observe that in the case of bicontinuous transformations, the gray level repartition function is the same for both images, i.e. there is a bijection between I_h and I'_h (which is $T_{|h}$), where I_h denotes the h-level points of I. In the general case, we need that the watershed partition on both images are the same. It does not mean that for every h, I_h and I'_h are isomorph. In fact, given g: $[0,h_{Imax}]$->$[0,h_{I'max}]$ increasing strictly, I_h and $I'_{g(h)}$ are isomorph and the watershed partition is the same. Hence, the image transformation I' = T(g(I)) where g is the strictly increasing gray level tranformation and T bicontinuous preserves the partitions and the trees.

In pratical cases, it means that the matching procedure can be applied on not-normalized images, or with an unknown gray level transformation. Moreover, the g bijection has not to be computed for using our algorithm.

2.3.3.3. Image resolution

The image resolution is quite important for the watershed line computing, but not for the matching algorithm. It is nor possible to explain here the way we compute the watershed line. The algorithm is linearly dependant of N (the number of pixels). If, due to the resolution, no local maximum and no saddle point is lost between I

and I', the watershed partition are isomorph. Hence, a different resolution between I and I' has no consequence for our matching procedure.

We will see in section 5 how to deal with the problem of over or under-segmentation, thanks to the partition tree.

2.4. Watershed line and confinement tree

We call "node" a multiple point on the watershed line (i.e. a point of the watershed which is neighbor of at least 3 catchment basins). We call "branch" the watershed part which relates 2 and only 2 nodes, these nodes included.

2.4.1. The watershed graph

Let consider the watershed line of the inverse image. Consider two neighbor maxima on this line. It is obvious that if you take a threshold of a level lower than those of the highest point on the watershed line part which separates the two maxima (i.e. the branch), the resulting connected component (made of the points higher than this level) contains the two maxima.

On the other side, if you take a threshold of a higher level than the highest point, the aforementioned maxima are disconnected. Hence, it is possible to reconstruct the confinement tree thanks to the watershed line.

We first need to define the "watershed graph", in which the vertices are the maxima on the image. When two maxima are separated by a watershed branch, we build an edge between the two vertices with a value (called potential) equal to the highest point level found on the watershed branch. We propose in the following subsection an algorithm which gives the confinement tree thanks to this graph.

2.4.2. The watershed tree

Given the watershed graph (which is supposed to be connected), we call "potential" of a simple path the smallest potential found on the belonging edges.

We can see that every leaf of the bifurcation tree (i.e. every local maximum on the image) is associated to a vertex on the watershed graph, and that every node of degree n ($n > 1$) of the bifurcation tree is associated to at least n-1 edges of the watershed graph, which are the lowest edges on the graph among the highest path between two maxima. For a 2-bifurcation, if there is 2 different highest paths in the graph, we associate the bifurcation to the two appropriate edge.

The aim of this algorithm is to build the tree thanks to the watershed graph. The root of the watershed tree is called r. The algorithm begins with a given watershed graph and a fictive root r of level -1. The vertices are marked when their treatment ends, in order not to be treated twice. The treatment of a given vertex v consists in looking for every of its adjacent edges, and in creating a new bifurcation in the tree if needed. The algorithm is done in that way : given the new vertex v, if its leaf l is not created, create it; else, if the next edge has two already created leaves (our current vertex v with leaf l, associated to the other extremity of the edge, with leaf l'), we have to verify that the edge is or not associated to a bifurcation in the tree; if not, we have to create a new bifurcation at the path r - l, and make the fusion between the paths r-l and r-l' (in fact, if a denotes the common ancestor of l and l', we have to make the fusion between a-l and a-l').

Let us denote, for any vertex v of the watershed tree :

h, the potential of the current edge
e, the current edge
f, the father of v in the tree
for a given node n in the tree, n-> father gives its father and n-> alt its altitude (level of the corresponding threshold on the image)
next-edge(v) gives the untreated edge adjacent to v with the highest potential, and NIL if there is no untreated edge left

v->leaf gives the leaf associated to v if it has already been created. If not, returns NIL.
extremity(v, e) gives the vertex of the watershed graph adjacent to e and different from v
create-leaf(f, v) creates the leaf descendant from its father f and associated to the vertex v in the graph
create-bifurcation(n, h) creates the bifurcation between nodes n and n->father, with altitude h (implies that n->father->alt ≤.h ≤ n->alt)
common-ancestor (l1, l2) gives the first common ancestor of l1 and l2
fusion-branch (l1,l2) finds the first common ancestor a of l1 and l2, and make the fusion of the two paths a-l1 and a-l2 in the tree, such that it remains only one path with two leaves (l1 and l2), with ancestors in l1 and l2 in the increasing order (i.e. every path in the tree from the root to a leaf has the value of each node in the increasing order).

We start with a fictive root of altitude -1, and a vertex v: call of create-tree(f, v). Our algorithm is then:

```
l = create-leaf(f, v)
while (next-edge(v))
        {
        e = next-edge(v)
        h = e->alt
        l = v->leaf
        while (l->father->alt > h) l = l->father
        f = l->father
        new-e = extremity(v, e)
        if ( new-e->leaf == NIL)
                {new-node = create-bifurcation(l, h)
                 create-tree (new-node, new-e)\}
        else
                {n = common-ancestor(l, new-e)
                 if (n->alt < h) fusion(l, new-e)}
        }
```

It is obviously a linear function of the number of vertices N_m of the watershed graph. It also depends on the number n_e of edges and on the number h of gray levels. Each edge is treated at most twice. Each treatment of one edge needs at most 2.h operations (fusion in the tree). Hence, the complexity is at most $N_m+2.n_e.h$. While n_e and N_m are linearly dependent on N, say for example in a 4-neighbouring pixels image, if N is the image size (number of pixels), we have $N_m < N/2$ and $n_e < N/2$. Then we have a complexity of O(N).

The complexity of the watershed computation being about 7,25.N, the resulting complexity is a O(N) function (depending on the kind of image).

2.5. Matching algorithm

2.5.1. Tree transformations

The watershed tree is a tree in which every node has one parameter, the altitude of the corresponding bifurcation(s), and every leaf is associated to a cluster (which is a catchment basin on the image).

For a node n in the tree, and its k direct descendants $l_0,\ldots,l_k$, each leaf l_i being associated to the basins B_i, we can associate to n the cluster $C_n = U_{i=0}^{k} B_i$. Hence, for every node, its direct descendants define a partition of its cluster. We also can associate to every node the weight m of its associated cluster.

Hence, on the initial tree we can use the node parameter (weight) and edge parameter (which is a potential, the difference between the altitudes of the two associated nodes).

For the initial condition, we will use weight and potential, in order to find significant (high potential) edges and significant (high weight) nodes. The matching is made on those simplified trees. Then, it becomes possible to have a very precise matching, using the connectivity property of the unmatched clusters with already matched ones. It leads to a partition matching of the two images, in which we can find triple points

(points neighbor to 3 or more clusters) for numerical application. It is possible to refine this algorithm on clusters with not still matched descendants.

2.5.2. Initial condition

It is possible to contract an edge in the tree without changing the properties of the tree.

The first step is then to contract every unsignificant edge, with a parameter s, which denotes the minimal significant potential on an image (i.e. contract every edge of potential p, if $p < s$). This parameter may be not a constant. For example, on echographic images, with h denoting the gray level, we have $s / h = 1,91$, hence the contraction depends on both the edge potential, and on the value of the altitude of its lowest extremity. On MRI images, as the variance parameter is not only dependent on the gray level, we must use the highest possible value (beween 10 and 15 on brain images, depending on the slice).

The second operation needed is the *node scission*. Given an admissible or matchable node n of weight m, with no still admissible descendant, then we take the descendant with the highest weight. We start with the root (whose weight is the weight of the image) and we build both trees by scissionning their nodes untill these trees become different. The number of scissions done before observing a difference is called the "resemblance coefficient" (R) of our two images. For MRI images, this parameter is between 15 and 30 for two following slices.

We have computed this coefficient by this way : starting with $s_0 = 110$ (where 110 is the number of gray levels), we have computed the h-contracted trees (which had only one node). Then, we compute the R_0 parameter of our two trees. We made the same with $s_1 = s_0 / 2$, and compute R_1. If $R_n > R_{n-1}$, we choose $s_{n+1} = s_n / 2$, else $s_{n+1} = (s_n + s_{n-1})/2$. We stop when the R coefficient becomes stable, and find for our slices $s = 14$ and $R = 18$.

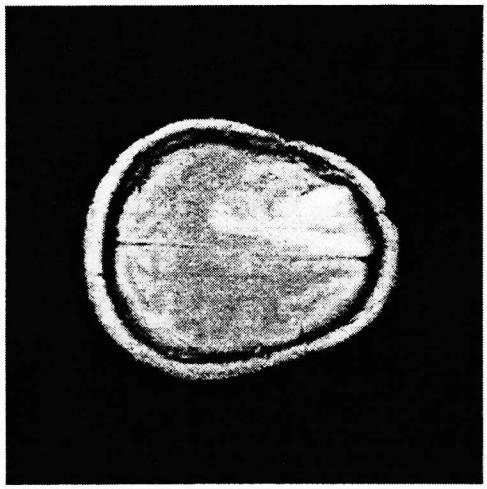
2.5.3. Matching the trees

This step is very simple, as the two operations described before allow us to find exactly two identical trees.

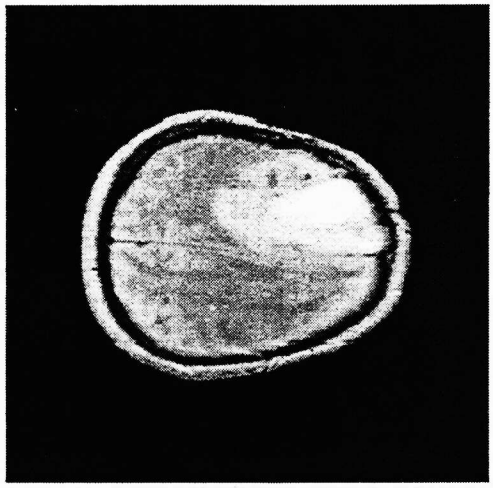
The principle of the matching procedure is the following : if nodes n_1 and n_2 have been matched and have admissible descendants, we match the heaviest weight descendant of n_1 with the heaviest weight one of n_2, and so on (let us remark that if n_1 and n_2 are matched, they have the same number of descendants)
\footnote{if there is several descendants with the same weight, we can introduce a geometrical criterion based on the neighbouring of associated clusters, or use a perimeter criterion}. Hence, our matching procedure is not directly dependent on the weight variations of the objects, but still holds under the assumption that the order of weights of objects is unchanged (which is a valid hypothesis in medical imaging).

You can see on the figure the two initial trees, where the descendants of a node are drawn following their weight, the heaviest weight one being at the left. The trees are exactly identical (the order of the weight of brothers being preserved), and the associated levels at any bifurcation are the same in both trees, with a great precision. If a node n in the first tree has the level h, its associated node n' has a level in [h-2, h+2].

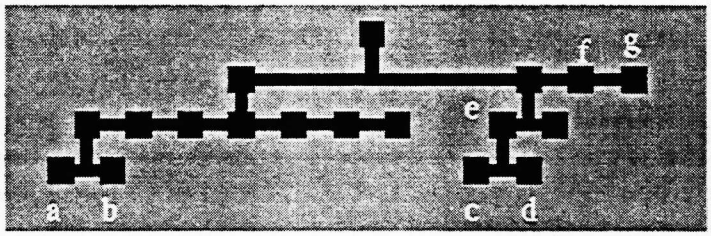
We can see the initial tree on Fig. 3. The most significant clusters have been shown. The other ones are included into the brain, and are difficult to see without a couloured computer interface. The clusters are represented in a 256 gray level original image. With an image of 256/14 gray levels, there will be no remaining nodes (i.e. the tree which is shown on the figure will be the whole watershed tree). The goal of the next section is to give a procedure which allows us to match remaining nodes with the original gray level resolution. Note that contours of clusters do not represent the real contours of objects (a threshold or any other method apllied on clusters can be used for the contouring).



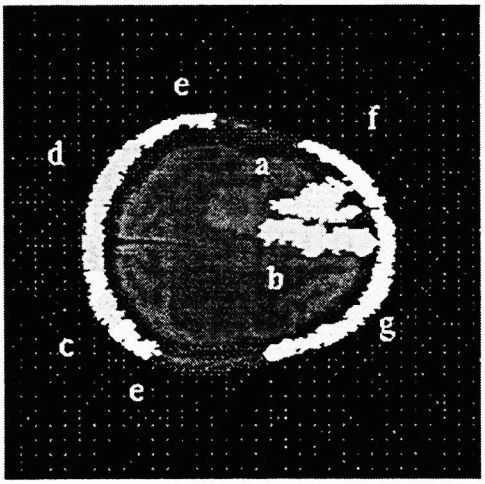
(1) brain image : slice n. 6



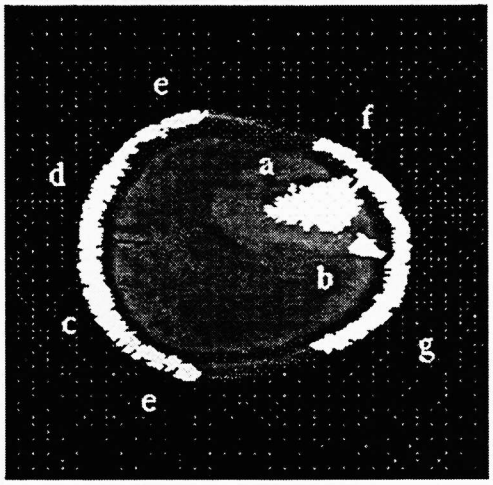
(2) brain image : slice n. 7



(3) common initial tree (18 nodes, with a potential of 14)



(4) clusters of sl. 6
(note that e contains c and d)



(5) clusters of sl. 7
(note that e contains c and d)

Fig. 3. *Initial tree for brain tumor MRI images*

2.5.4. Matching remaining nodes

Given an unmatched node n, we can use its matched brothers and ancestor in order to match it. This is done in a numerical goal, hence the matching of remaining nodes is not supposed to respect the tree structure.

If n has no common border with its father (i.e. n is only neighbored by its brothers), then we find its matched brothers which are also neighbors b_1, ..., b_i which are associated to c_1, ... , c_i in the other tree, and associate n to the cluster c of the other tree, which is the heaviest among the brothers which are neighbors of all c_1, ... , c_i. Of course, it is not always possible, but allows us to improve the number of matched brothers. If n has no admissible cluster in the other tree, it is possible to connect n with one of its unmatched brother if the two clusters are neighbor, and find the correspondent node in the other tree in the preceding way. But this method (matching groups of nodes) has a great complexity, because for a set of brothers, you need to compute all the possible connections , do the same for the other tree at the same level, and find admissible correspondences. This has not be implemented. If the unmatched node n has a common border with its father, the procedure is the same, but we have to take into account the neighboring between n and its father (remark that the cluster associated to n is included in the cluster of its father). The same procedure is used again for matching descendants of new matched nodes.

2.5.5. Finding matchable points

This step is very easy to do: as soon as we have two partitions (on both images) for wich we have a bijection from cluster to cluster, it is easy to find triple points (in our case, pixels which are included in a cluster, and neighbor to at least two clusters) and match it, one triple-point pixel of the first image being associated to one in the other image.

Given those points, it is possible to compute the continuous transformation with a great precision. The initial condition on both sides (the image and the inverse of it) gives 40 corresponding points. The simple cluster association gives some informations, for example a set of k pixel being associated to a set of k' in the other image, but it does not allow us to have a pixel to pixel correspondence.

We can improve this computing by two methods : first, between 2 triple points, there is a branch. Then we can match several points of those branches by extrapolation (say, for exemple, that the middle point on that branch is still the middle of the corresponding branch on the second image).

Secondly, we can compute a contour on a cluster (for example by thresholding at the corresponding node level) and match it to the contour of the cluster in the second tree (let remark that for this, we need an other hypothesis on the transformation in order to match contours, pixels to pixels).

3. Computer assisted surgery

The role of optronics in computer assisted surgery (CAS) is multiple :

- to localize the patient surface structures in the operation theater (OT) (by using passive or active infra-red sensors)
- to match the 3D referential corresponding to the deep anatomic structures (acquired by an X detector in the OT) with the 3D referential of the patient surface structures in the OT (Fig. 1 & 2)
- to match the deep and surface 3D referentials of the patient in the OT with the pre-operative (CT-scanner, anatomic or functional NMR, 3D US, PET or SPECT, MEG,..., image devices) referentials
- to guide (e.g. with a laser beam) the surgeon tools.

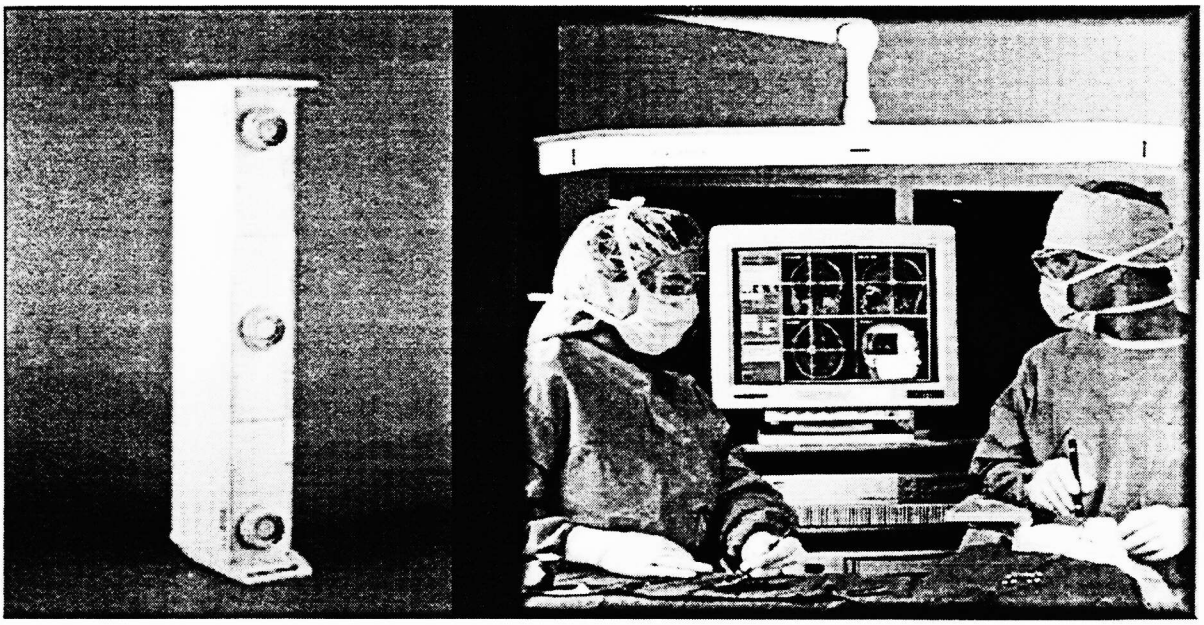


Fig. 4. *Use of active optical sensors in operation theater*

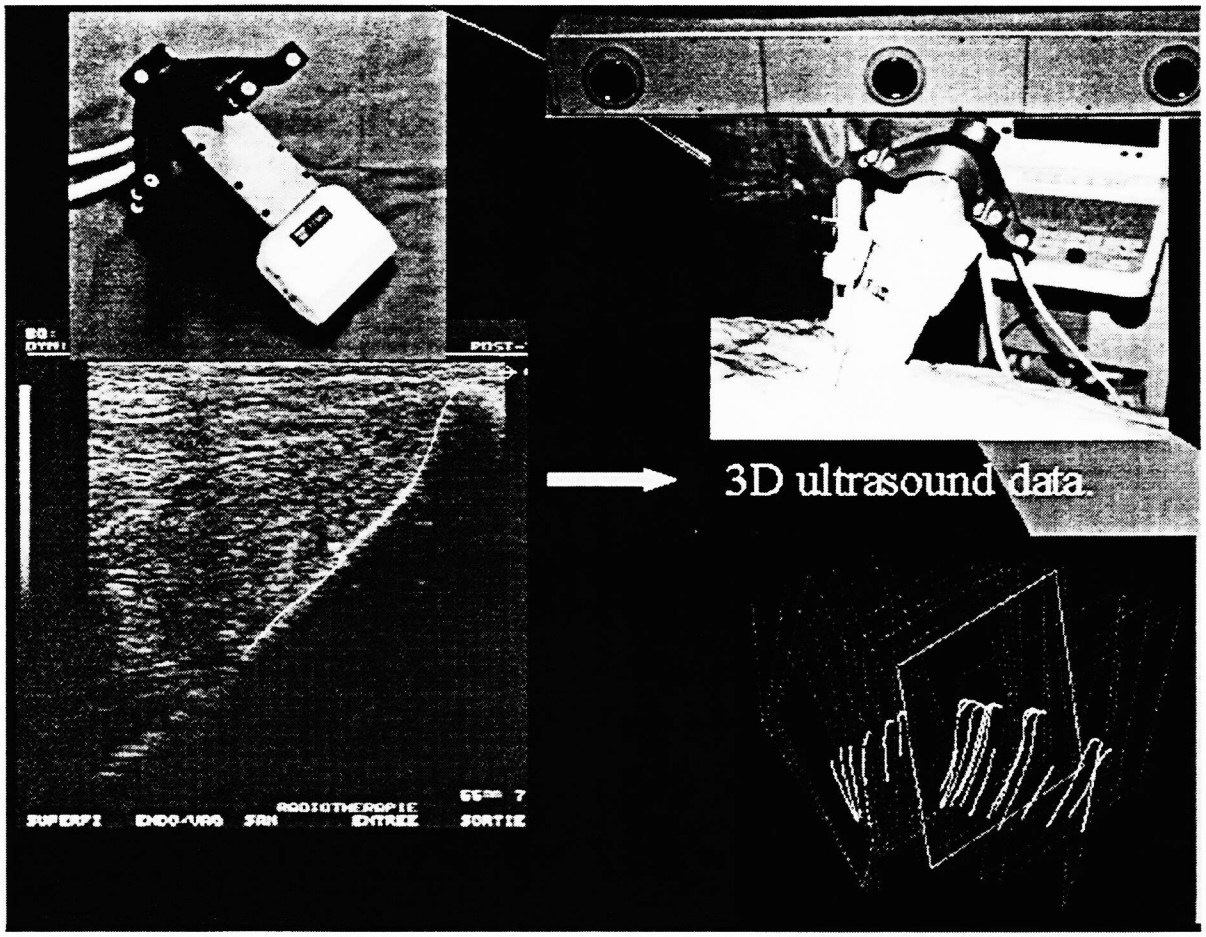


Fig. 5. *Ultra-sonuds acquisition located by active infra-red sensors*

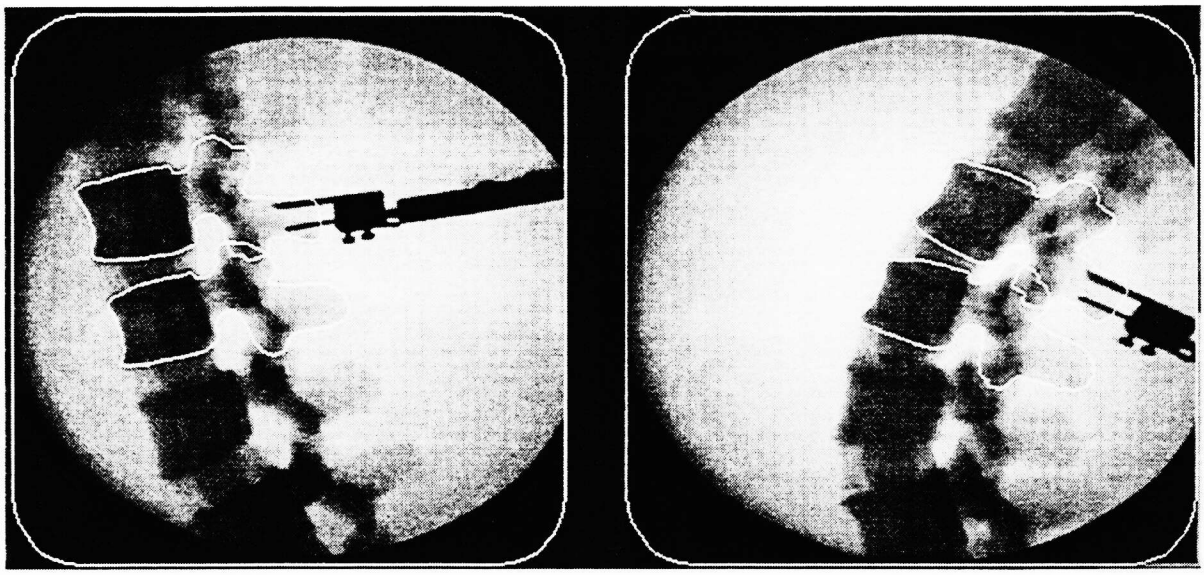


Fig. 6. *Comparison between vertebra contours coming from 2 different modalities (per-operative 2D X rays and pre-operative NMR – in white)*

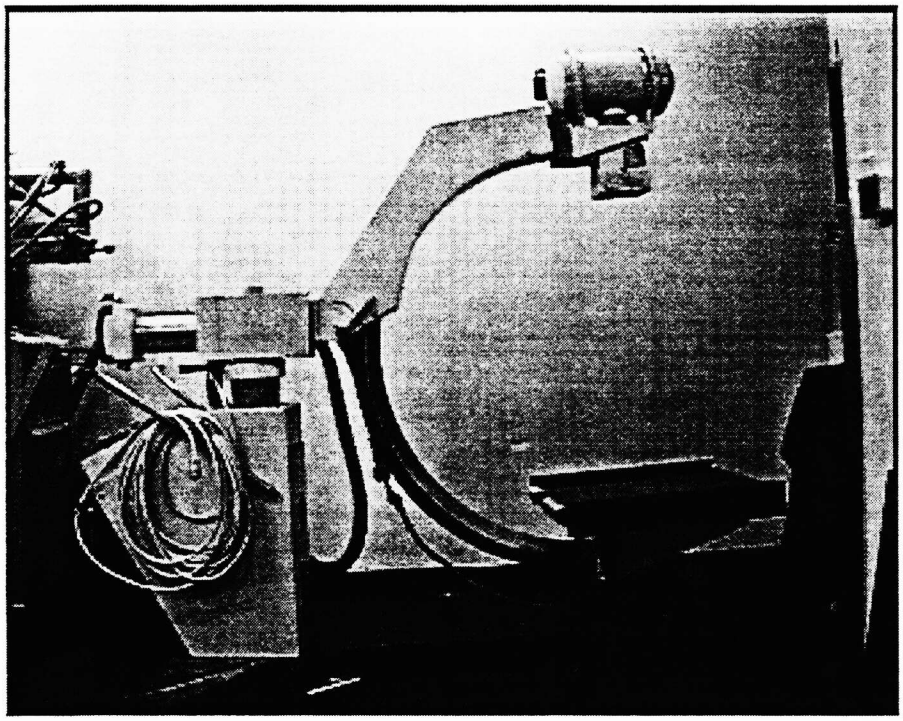


Fig. 7. *Per-operative 3D X rays acquisition device*

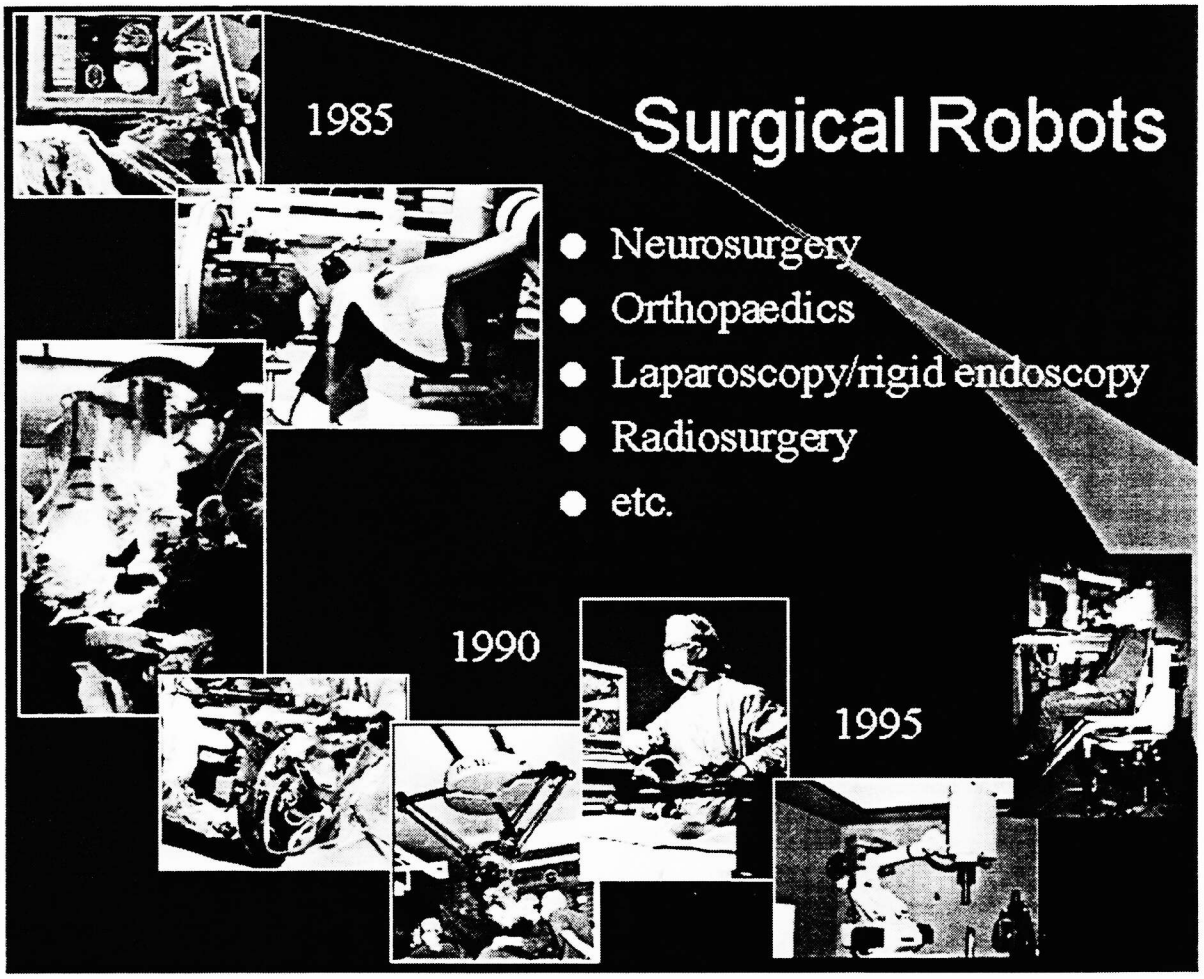


Fig. 8. ***Progressive improvements of the Computer Assisted Surgical Robotics***

These techniques can be applied to many domains of CAS : knee surgery (implanting internal cross ligaments), neuro-surgery (putting electrodes in deep structures involved in Parkinson's disease), back surgery (putting screws in vertebras, Fig. 6) and ant & maxillo-facial surgery (putting material in the mandible).
In a near future the own surgeon hand provided by a specific surgical tool (bistoury, biopsy puncture needle, Kocher pinch,...) will be guided at each time in a cone of admissible trajectories : if the surgeon wants to force the CAS system, he can, but a auditory, visual or lingual signal can recall him that he is escaping out the pre-chosen trajectory set of cones going from the skin opening to the final surgical target. To achieve such a hand real-time control, the successive 3D locations of the tool have to be precisely acquired by using an passive or active optical or electro-magnetic marker located on the top and on the bottom of the tool.

In the future, we can hope than an optimal combination of pre-operative information (given for example by a 3D holographic reconstruction of the objects of medico-surgical interest, see [von Bally et al. in the present volume]) with a completely automatized surgical procedure (by using a robot or a 6 degrees of freedom coded arm, Fig. 8), or with a real time aid for the surgeon hand control (as described above), could allow :

1) the choice of an optimal trajectory from the less invasive opening zone to the final target, by avoiding noble zones (of the cortex in neuro-surgery of the brain, or of the liver in the gastro-entero-surgery) or important vascular regions ; this choice could be made some days before the operation, by asking for the distant expertise by a specialist, playing with the 3D objects reconstructed via a hologram or via a ray tracing realistic representation on a screen
2) the real time control of the execution of the gesture realizing this optimal trajectory, by registering the two 3D spatial referentials, the pre-operative one, in which the optimal trajectory has been chosen and the per-

operative one, in which the actual surgical operation or medical puncture is performed. The control information can be given either visually on a screen, or can come from auditory, epidermic or lingual stimuli in an augmented reality framework.

3. Conclusion

We have proposed a new algorithm allowing us to build the component tree in a linear time according to the size of the image. The component tree is an image characteristic invariant in many deformation processes. In this problematics, in order to obtain the component tree, there is a direct way by slicing the gray level density function (see [Malpica et al. 1997]) and another way (presented in this paper) using the immersion process and watershed tree computation, which gives us the possibility to check the solution of a method by using the other. Indeed, the watershed tree defines for every node a meaningful component, contrarily to the component tree in which low thresholds have no real meaning, because the corresponding contour lines less respect at this level the topological structure of the image : they do not follow low values of the gradient, hence the watershed approach is less sensitive to the noise on the image gradient, letting invariant the extrema of the image gray level. The matching is then much easier to do, and allows us to use the noise properties of our images (by calculating global parameters on the images). In a last observation, the properties of the component tree telling that the cluster of any node contains the clusters of its sons and that the clusters of the sons have an empty intersection are still true, moreover the union of the sons' clusters is the father's one on the watershed tree. Those topological properties are fundamental for making an elastic matching of the two images (and it is not possible if the confinement tree is computed by thresholding or slicing), using local properties of our trees (through combinatorial optimisation approach). All the properties of watershed trees are here not explored, but this approach, using both topology and locally gray level fonction, seems to be efficient for continuous elastic matching of medical images.

The new algorithm is now used to register images coming from optical sensors and from positioning image devices located in the patient room and in the operation theater, with pre-operative images coming from NMR or CAT-scanner examinations. The only way to improve the real time assisted surgery is to pass through all the steps described above : to define the per-operative sensing, to match it with the pre-operative information and finally to perform a medico-surgical action based on the correspondence between the two (pre-operative and per-operative) 3D spatial referentials, in order to be more accurate and rapid and hence to improve the quality of care and diminish the iatrogenic effects of the surgical therapy.

5. References

Bach y Rita, P., Kaczmarek, K.A., Tyler, M.E. and Garcia-Lara, J.. (1998). *Form perception with a 49-point electrotactile stimulus array on the tongue : a technical note*, {J. Rehabilitation R&D} {35}, 427-430.

Benabid, A.L., Cinquin, P., Lavallée, S., Le Bas, J.F., Demongeot, J. and de Rougemont, J. (1987). *A computer driven robot for stereotactic surgery connected to cat-scan magnetic resonance imaging. Technological design and preliminary results*, {Applied Neurophysiology} {50}, 153-154.

Benabid, A.L., Lavallée, S., Hoffmann, D., Cinquin, P., Demongeot, J. and Danel, V.F. (1992). *Potential use of robots in endoscopic neurosurgery*, {Acta Neurochir.} {54}, 93-97.

Bertrand, G., Everat, J.C. and Couprie, M. (1997). *Topological grayscale watershed transformation. Vision Geometry VI*, {SPIE} {3168}, 136-146.

Bertrand, G., Everat, J.C. and Couprie, M. (1997).*Image segmentation through operators based upon topology,*. {Journal of Electronic Imaging} {6}, 395-405.

Beucher, S. and Meyer, F. (1993). *The morphological approach to segmentation: the watershed transformation.* In : E.R. Dougherty, ed., {*Mathematical Morphology in Image Processing*} : 433-481. Marcel Dekker, New York.

Bettega, G. (1992). {*A la recherche de la 4ème dimension*}, Grenoble, PhD Thesis Université J. Fourier.

Blankenship, T. (1987). *Real-time enhancement of medical ultrasound images*, In : L.W. Kessler, ed., {*Acoustical Imaging*} : 187-195. Plenum, New York.

Brunie, L., Lavallée, S., Troccaz, J., Cinquin, P. and Bolla, M. (1993). *Pre- and intra-radiotherapy multimodal image registration*, {J. Radiotherapy and Oncology} {29}, 244-252.

Brunie, L., Leitner, F., Berthommier, F., Cinquin, P. and Demongeot, J. (1995). *Interpretation of multimodal medical images using connectionist and variational methods*, {Technology and Health Care} {3}, 91-100.

Cinquin, P., Demongeot, J., Troccaz, J., Lavallée, S., Champleboux, G., Brunie, L., Leitner, F., Sautot, P., Mazier, B., Perez, A., Djaid, M., Fortin, T., Chenin, M. and Chapel, A. (1992). *Igor : image guided operating robot. Methodology, medical applications, results*, {ITBM} {13}, 373-393.
Demongeot, J., Cosnard, M. and Jacob, C. (1986). *Attractors and confiners: deterministic and stochastic approaches.* In : S. Diner, D. Fargue and G. Lochak, eds., {*Dynamical Systems - A Renewal of Mechanism, Centennial of George David Birkhoff*} : 20-37. World Scientific, Singapore.
Demongeot, J. (1995). *Chirurgie assistée par ordinateur : bilan et perspectives*, {Chirurgie} {120}, 300-307.
Fay, D.A. and Waxman, A.M. (1991). *Real-time early vision neurocomputing*, {IEEE International Joint Conference on Neural Networks} {1}, 621-626.
Haibo, L., Roivainen, P. and Forchheimer, R. (1993). *3-d motion estimation in model-based facial image coding,* {IEEE Trans. Pattern Anal. & Machine Intell.} {15}, 545-556.
Hanusse, P. and Guillautaud, P. (1992). *Sémantique des images par analyse dendronique.* {8ème Conf. Reconnaissance des Formes et Intelligence Artificielle RFIA} {2}, 577-588. AFCET, Paris.
Healey, B. G. and Walt, D. R. (1997). *Fast temporal response fiber-optic chemical sensors based on the photodeposition of micrometer-scale polymer arrays,* {Analytical Chemistry} {69}, 2213-2216.
Heidman, R. G. and Veldhuis, G.J. (1996). *Fabrication and packaging of integrated chemo-optical sensors,* {Sensors and Actuators B} {35-36}, 234-240.
Kergosien, Y.L. (1992). *Topology and visualisation : from generic singularities to combinatorial shape modelling*. In : , T. Kunii, ed., {*Modern Geometric Methods for Computing*} : 31-54. Springer, Berlin.
Malpica, N., Desolorzano, C.O., Vaquero, J.J., Santos, A., Vallcorba, I., Garciasagredo, J.M. and Delpozo, F. (1997). *Applying watershed algorithms to the segmentation of clustered nuclei,* {Cytometry} {28}, 289-297.
Mattes, J., Richard, M. and Demongeot, J. (1999) *Tree representation for image matching and object recognition*. In : G. Bertrand, M. Couprie and L. Perroton, eds, {*Proc. 8th Conf. on DGCI'99*} : 298-309. Springer, Berlin.
Mattes, J. and Demongeot, J. (1999) Dynamic confinement, classification and imaging. *Proc. 22nd Annual Conference, GfKl., Dresden March 98*. In : W. Gaul and H. Locarek-Junge, eds, {*Studies in classification, data analysis, and knowledge organization*} : 205-214. Springer, Berlin.
Mattes, J. and Demongeot, J. (1999) *Tree representation and implicit tree matching for coarse to fine image matching algorithm*. In : A. Colchester and A. Sedgemore-Schulthess , eds, {*Proc. MICCAI'99*} : 200-209. Springer, Berlin.
Meyer, F. (1992). *Un algorithme optimal de ligne de partage des eaux.* {*8ème Conf. Reconnaissance des Formes et Intelligence Artificielle RFIA*} {2} : 847-859. AFCET, Paris.
Mirkin, B. (1996). {*Mathematical Classification and Clustering*}. Kluwer Academic Publishers, Dordrecht.
Sakarovitch, M. (1984). {*Optimisation combinatoire - Méthodes mathématiques et algorithmiques*}. Hermann, Paris.
Shasha, D., Tsong-Li Wang, J., Kaizhong Zhang, F.Y. and Shih, F.Y. (1994). *Exact and approximate algorithms for unordered tree matching*, {IEEE Trans. on Systems, Man and Cybernetics} {24}, 668-678.
Shinagawa, Y., Kergosien, Y. and Kunii, T. (1991). *Surface coding based on morse theory,* {IEEE CG & A Magazine} {11}, 66-78.
Toet, A. (1992). *Multiscale contrast enhancement with applications to image fusion*, {Optical Engineering} {31}, 1026-1031.
Vachier, C. (1995). {*Extraction de caractéristiques, segmentation d'images et Morphologie* Mathématique}. PhD Thesis, Ecole des Mines, Paris.
Vincent, L. and Soille, P. (1991). *Watershed in Digital Spaces : an Efficient Algorithm Based on Immersion Simulations,* {IEEE Transactions on Pattern analysis and Machine Intelligence} {13}, 583-598.