

LABORATORY OF MATHEMATICS IN IMAGING

**Fast Sub-Voxel Reinitialization of the Distance Map
for Level Set Methods.**

Karl Krissian, Carl-Fredrik Westin

N 0001

November 2003

Technical Report

Laboratory of Mathematics in Imaging

Harvard Medical School

Brigham and Women's Hospital, department of Radiology

75 Francis Street

02115 Boston, MA, USA

Fast Sub-Voxel Reinitialization of the Distance Map for Level Set Methods.

Karl Krissian, Carl-Fredrik Westin ¹

February 2, 2004

¹Harvard Medical School Brigham and Women's Hospital Dep of Radiology Thorn 323
Boston, MA 02115 USA Fax : (+1) 617-264-6887 Emails: {karl,westin}@bwh.harvard.edu

Abstract

Redistancing the implicit surface is currently the most time-consuming stage in Level Set Methods, usually accomplished by applying the Fast Marching algorithm to a binarized image. We propose to apply a faster and linear approximation of the Euclidian Distance while maintaining the sub-voxel accuracy of the interface.

Keywords

Level Set Methods, Euclidean Distance Map, Segmentation.

Contents

1	Introduction	2
2	Sub-Voxel reinitialization of the Narrow Band	2
	2.1 Sub-Voxel reinitialization	3
	2.2 Comparison with a binary reinitialization	4
3	Narrow-Banded Fast Distance Transform	5
	3.1 Chamfer Distance Transform	7
	3.2 Algorithm	7
4	Interpretation	9
5	Experiments	11
	5.1 Accuracy experiments	11
	5.2 Speed experiments	12
	5.3 Segmentation of the white matter	14

1 Introduction

Although the Level Set Method [OS88, Set99b] has only relatively recently been introduced as an image processing tool, it has already proved useful in delineating contours and in segmenting objects. Because it relies on an implicit representation of the evolving contour as the zero-crossings of an image, it deals naturally with changes in topology. However, since this method represents contours as the zero-crossings of an image, the dimensionality of the problem is increased by one and processing time may increase considerably. To offset this problem, we typically create a narrow band of the image points lying within a given distance to the contour, and then process the Partial Derivative Equation of the contour evolution inside this narrow band. The use of the narrow band and numerical considerations require computing the distance to the evolving contour every few iterations.

Several techniques have been proposed for computing this distance [SF99, Set99a].

In [SSO94], the authors propose to solve the following equation in order to compute the signed distance to the interface:

$$\begin{aligned}\phi(x, 0) &= \phi_0 \\ \phi_t &= \text{sign}(\phi_0)(1 - \|\nabla\phi\|).\end{aligned}\tag{1}$$

In particular, in [SF99], they formulate a volume preserving constraint designed to prevent the straying of the zero level set from the initial position even after many iteration; they demonstrate that they only need to solve the equation up to time $t = L$ where L is the thickness of the narrow band about the interface in which a distance function is needed; and they use high order methods in time as well as space for solving the equation. However, this method which has good properties is computationally expensive, especially noticeable when dealing with high resolution three-dimensional images.

In [RS00], the authors identify the problem of preserving the exact position of the interface and propose to solve a new Partial Derivative Equation for this purpose. In [Set99a], a $n\log(n)$ algorithm is proposed to compute the distance by propagation until a given distance to the contour is reached. In this paper, we propose a novel fast algorithm, which both preserves the sub-voxel accuracy of the contour position and computes its signed, narrow-banded distance with a linear complexity.

Add: Ross Whitaker, Tsikilis, Strain, Osher...

2 Sub-Voxel reinitialization of the Narrow Band

The Level Set Method used for segmentation is intrinsically a sub-voxel estimation of the contours of the object, because it represents the object as the zero-crossings of a gray scale image. The surface is generally extracted at the end of the evolution using the Marching Cubes Algorithm [LC87], which hypothesises tri-linear interpolation between

the voxels. It is common to reinitialize the Distance Transform from a binary image every few iterations, however doing this compromises the sub-voxel accuracy.

2.1 Sub-Voxel reinitialization

We will limit our discussion to the isosurface of value zero. Hence, we designate a voxel “neighbor” to the isosurface if this voxel is either of intensity zero or if the isosurface crosses one of the six segments (or four in 2D images) with its direct neighbors.

To calculate this distance, for each point designated as neighbor to the surface, we could compute the real distance to the interpolated isosurface as the minimum of the distance to the close triangles. However, such a formula would require extracting the triangles and computing for each triangle its distance to several close voxels, a process far too time consuming for our application.

Alternatively, we could estimate the distance to the surface, while retaining as much as possible of the same interpolated surface. To achieve this, we could maintain the relative position of the points of the surface with their neighboring voxels.

We denote $I(x, y, z)$ the intensity of the level set image at the point $(x, y, z) \in \mathbb{N}^3$. A point of the surface M , when the surface is interpolated by trilinear interpolation, is always a linear combination of two voxels $V_1 = (x_1, y_1, z_1)$ and $V_2 = (x_2, y_2, z_2)$.

$$M = \alpha_1 V_1 + \alpha_2 V_2 \quad (2)$$

with $\alpha_1 = \frac{I_2}{I_2 - I_1} \in [0, 1]$ and $\alpha_2 = 1 - \alpha_1 \in [0, 1]$, where $I_1 = I(V_1) = I(x_1, y_1, z_1)$ and $I_2 = I(V_2)$.

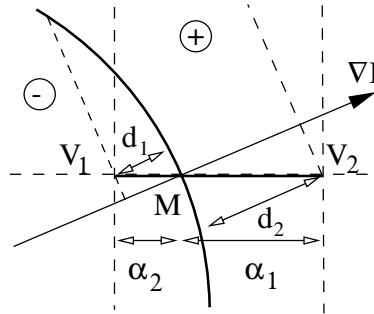


Figure 1: Distance from the neighbor points to the interpolated surface.

We denote \tilde{I} the new image with estimates of the distance to the isosurface for voxels neighbor to the isosurface. Thus, the values of \tilde{I} at V_1 and V_2 are estimated by projecting

the vector $\overrightarrow{V_1V_2}$ in the gradient direction ∇I :

$$\begin{aligned} \tilde{I}(V_1) &= d_1 = \beta I_1 \quad \text{and} \\ \tilde{I}(V_2) &= d_2 = \beta I_2 \quad \text{with} \\ \beta &= \frac{1}{I_2 - I_1} \overrightarrow{V_1V_2} \cdot \frac{\nabla I}{\|\nabla I\|}. \end{aligned} \tag{3}$$

Moreover, the vector $\overrightarrow{V_1V_2}$ is always oriented in one of the axes of the image coordinates, and the scalar product $\overrightarrow{V_1V_2} \cdot \frac{\nabla I}{\|\nabla I\|}$ is thereby reduced to taking one of the components of the unit vector in the gradient direction.

Remarks:

- Owing to the characteristically discrete grid, it is not possible to retain the same interpolated surface after the distance estimation in the neighborhood. In the simple case shown in fig.1, however, the position of the interpolated surface is preserved. Because the same voxel can share several edges that intersect with the isosurface, we need only capture the minimal distance value estimated for this voxel.
- Our estimate approximates the real distance to the surface; the accuracy of which hinges on how closely the gradient vector of the image approximates the real normal to the isosurface. The use of the same gradient vector for both points V_1 and V_2 allows keeping the same position of the interpolated point M . However, in regions where the isosurface exhibits high curvature, the gradient vector will quickly fluctuate; estimates, in turn, become less accurate. To more accurately estimate the gradient vector, one option is to use a linear combination of the gradient estimation in V_1 and V_2 with weights α_1 and α_2 .

2.2 Comparison with a binary reinitialization

Figure 2 shows the evolution of an initial circle of radius three pixels, evolving with a constant expansion force of 0.8, a curvature coefficient of 0.2, a time step of 0.1 and a distance reinitialization every 10 iterations. On the top line we show the zero-isosurface of the 3D image created by stacking the 2D images obtained during 200 iterations. To better understand why the binary reinitialization evolves to a square, on the bottom line we show the zero-isocontours of the distance reinitialization from a circle of radius 5.5. The binary distance reinitialization is achieved by assigning a value of ∓ 0.5 to the points neighbors to the zero-isocontour.

Figure 3 shows the surface evolution on a Maximum Intensity Projection of an image representing an aneurysm. On the left column, we have superimposed the isocontour of the estimated distance in dashed red to the isocontour of the initial image in black. The bottom left image shows that the two isocontours are almost identical, even when the curvature is high, and that small variations in the isocontour shape are preserved. The

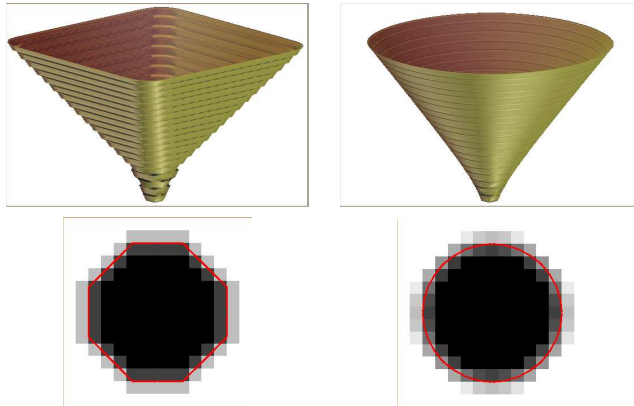


Figure 2: Evolution from a pixel with constant propagation force. Left, evolution using a binary reinitialization. Right, evolution using the sub-voxel reinitialization.

right column shows that the isocontours of threshold 1 and -1 are well situated around the zero-isocontour.

Remarks:

- In this section, we have presented results on 2D images for clarity. The extension to 3D is straightforward.
- In case of anisotropic voxels, we can apply the same distance estimates by taking into account the voxel size in each dimension.
- The use of a binary reinitialization can also lead stagnation in the position of the level set if it fails to move rapidly enough between two re-initializations.

3 Narrow-Banded Fast Distance Transform

Once the distance to the interpolated level set is estimated, we apply a very fast approximation of the Distance Transform (DT) to this level set that is restricted to the size of the narrow band.

We use a new, modified algorithm to compute the Chamfer Distance with appropriate coefficients [Bor96]. These coefficients ensure that the error is bounded by 7.3 % of the exact distance. Other Exact Distance Transforms could be used in this application [Dan80, Cui99]. However, a comparison of the speed of several Distance Transforms [Cui99] shows that the Chamfer DT is two or three times faster than the others. Moreover, an approximation of the Distance Transform is sufficient for our level set application as long as the distance itself is not an explicit part of the Hamilton Jacobi Equation. This Distance Transform has two purposes: 1) to regularize the equation and

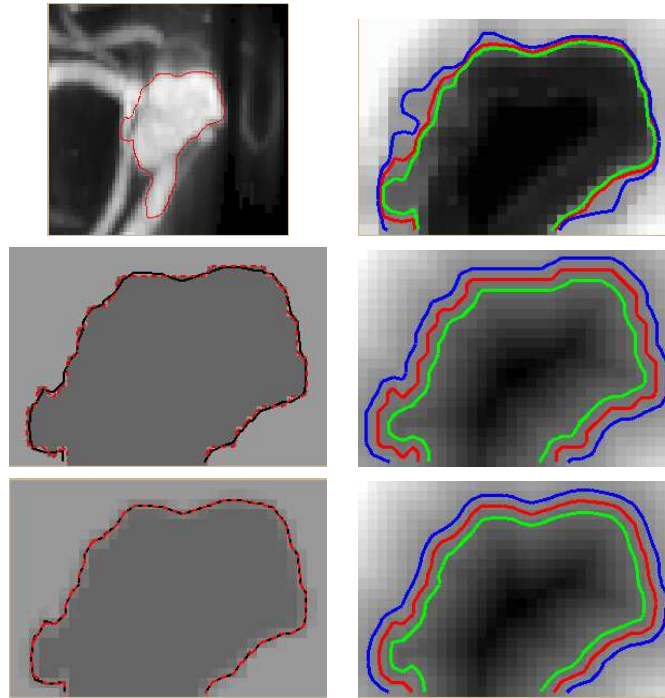


Figure 3: From left to right and top to bottom, the initial image with the current position of the level set, the level set image, the binary image with values -0.5 (inside) and $+0.5$ (outside), the Distance Transform obtained from this binary image, the estimated distance close to the level set, the Distance Transform obtained from this estimated image. The solid black line represents the isocontour of the level set, the red, blue, and green lines represent respectively the isocontours of intensities $0,1$ and -1 of the displayed images.

avoid numerical instabilities; 2) to estimate the distance to the contour for selecting the points belonging to the narrow band.

3.1 Chamfer Distance Transform

Let us consider a binary image representing a shape F . A distance transformation (DT) converts this binary image to a distance image, where each voxel has a value measuring the distance to the closest voxel in F . Yamashita and Ibaraki [YI86] define the distance between two points x and y as the length of the shortest path connecting x and y in an appropriate graph. They prove that any distance is definable in the above manner, by choosing an appropriate neighborhood relation and an appropriate definition of path length.

In 3D cubic space, each voxel has three types of neighbors: 6 area neighbors, 12 edge neighbors and 8 point neighbors.

The $3 \times 3 \times 3$ Weighted (or Chamfer) Distance Transform is defined by three values $\langle a, b, c \rangle$ which represent the local distance for each of the three types of neighbors (area, edge and point neighbors). The local Euclidean distance between these three types of neighbors would give a $\langle 1, \sqrt{2}, \sqrt{3} \rangle$ DT which “leads to very rough approximation of the global Euclidean Distance Transform” [Bor96].

Not all combinations of local distances a , b and c result in useful distance transforms. In 2D, a regular distance transform is a metric [RP68]. In 3D, the conjecture is that regular DTs are metrics too. The regularity is defined as follows:

Consider two pixels that can be connected by a straight line, i.e., by using only one type and direction of local step. If that line defines the distance between the pixels, i.e., is a minimal path, and if there are no other minimal paths, then the resulting distance transform is *regular*.

In [Bor96], the author compares the approximation error among several Chamfer DT's in three-dimensional images. With the restriction, for the Distance Transform, to be regular, a set of possible Distance Transforms are inferred, and the maximal difference with the Euclidean distance is computed. The best coefficients found are $\langle 0.92644, 1.34062, 1.65849 \rangle$, giving a relative maximal error of 7.356%.

3.2 Algorithm

Computing of the Distance Transform only requires two passes through the image, during which each pass computes the distances from the object to each voxel for half of the directions of the $3 \times 3 \times 3$ neighborhood. If the images are indexed by the coordinates $(x, y, z) \in [0, D_x - 1] \times [0, D_y - 1] \times [0, D_z - 1]$, then the algorithm, as described in [Bor96], is written as:

```
ALGORITHM 1 (STANDARD CHAMFER ALGORITHM).
// First pass
```

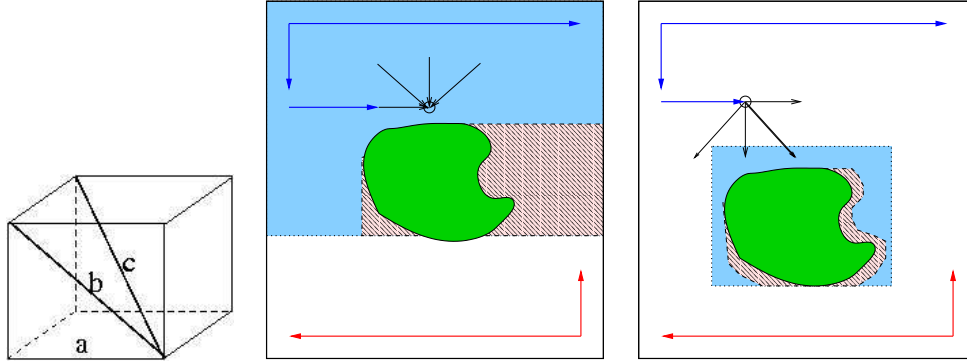


Figure 4: Neighbor types.

```

For z=1 to  $D_z-1$  do
For y=1 to  $D_y-1$  do
For x=1 to  $D_x-1$  do  $v_{x,y,z} = \min_{(i,j,k) \in \mathcal{N}} \{v_{i,j,k} + d_n\}$ .
// Second pass
For z= $D_z-2$  to 0 do
For y= $D_y-2$  to 0 do
For x= $D_x-2$  to 0 do  $v_{x,y,z} = \min_{(i,j,k) \in \mathcal{N}} \{v_{i,j,k} + d_n\}$ .

```

where v is the image containing the initial contour, \mathcal{N} is the set of already visited neighbors, and d_n is the appropriate local distance a , b or c . This algorithm is linear on the number of voxels of the image, and requires thirteen additions on each voxel for each pass. A small modification can give us the same result with only three additions per voxel, and the possibility to compute the distance only until a given distance to the contour. This modification consists in propagating the distance to the non-visited neighbors, instead of updating the current neighbor from the already visited neighbors. The resulting image is strictly the same as the standard Chamfer Algorithm because when the algorithm reaches a given voxel, all its visited neighbors have updated its value. We also adapted the algorithm in order to compute the positive and negative distance for the exterior and interior of the object, retaining only two passes through the image. Our new algorithm is thus written as:

ALGORITHM 2 (SIGNED NARROW BANDED CHAMFER ALGORITHM).

```

// First pass
For z=0 to  $D_z-2$  do
For y=0 to  $D_y-2$  do
For x=0 to  $D_x-2$  do
  If  $abs(v_{x,y,z}) \geq d_{max}$  then continue
  If  $v_{x,y,z} > -a$  do

```

```

 $d_a = v_{x,y,z} + a$ 
 $d_b = v_{x,y,z} + b$ 
 $d_c = v_{x,y,z} + c$ 
For (i,j,k) in  $\mathcal{N}_a$  do  $v_{i,j,k} = \min\{v_{i,j,k}, d_a\}$ 
For (i,j,k) in  $\mathcal{N}_b$  do  $v_{i,j,k} = \min\{v_{i,j,k}, d_b\}$ 
For (i,j,k) in  $\mathcal{N}_c$  do  $v_{i,j,k} = \min\{v_{i,j,k}, d_c\}$ 
EndIf
If  $v_{x,y,z} < a$  do
 $d_a = v_{x,y,z} - a$ 
 $d_b = v_{x,y,z} - b$ 
 $d_c = v_{x,y,z} - c$ 
For (i,j,k) in  $\mathcal{N}_a$  do  $v_{i,j,k} = \max\{v_{i,j,k}, d_a\}$ 
For (i,j,k) in  $\mathcal{N}_b$  do  $v_{i,j,k} = \max\{v_{i,j,k}, d_b\}$ 
For (i,j,k) in  $\mathcal{N}_c$  do  $v_{i,j,k} = \max\{v_{i,j,k}, d_c\}$ 
EndIf
EndFor.
// Second pass
For (z,y,x)=( $D_z-1, D_y-1, D_x-1$ ) to (1,1,1) do
  idem as first pass
EndFor.

```

where $\mathcal{N}_a, \mathcal{N}_b, \mathcal{N}_c$ denote the set of non-visited neighbors respectively for area, edge and point neighbors. This algorithm allows two speed improvements: The first obviates the need to compute voxels with intensity greater than the size of the narrow band, given by d_{max} . The second factorizes the additions for each kind of neighbor. The input image for this algorithm is the estimate of distance to the level set as described in section 2, and a distance value of d_{max} or $-d_{max}$ for exterior and interior voxels that are not neighbors to the zero-isosurface.

4 Interpretation

In this section, we interpret the meaning of computing the “exact” Distance Transform from a signed distance estimation for the voxels neighbor to the zero-isocontour (see section 2). We will only interpret the external distance (positive values).

When computing the “exact” Distance Transform from a binary image, we are actually computing, for each pixel or voxel, the shortest distance to the set of voxel centers included in the object. If we subtract 0.5 to the resulting distance and concentrate only on positive values, then we obtain the distance to estimated object boundaries defined by combining arc of circles of radius 0.5 (see fig. 5 top left).

When we first estimate the distance to the object boundaries for the voxels that are its neighbors, we are computing the distance to a more accurate sub-voxel estimation

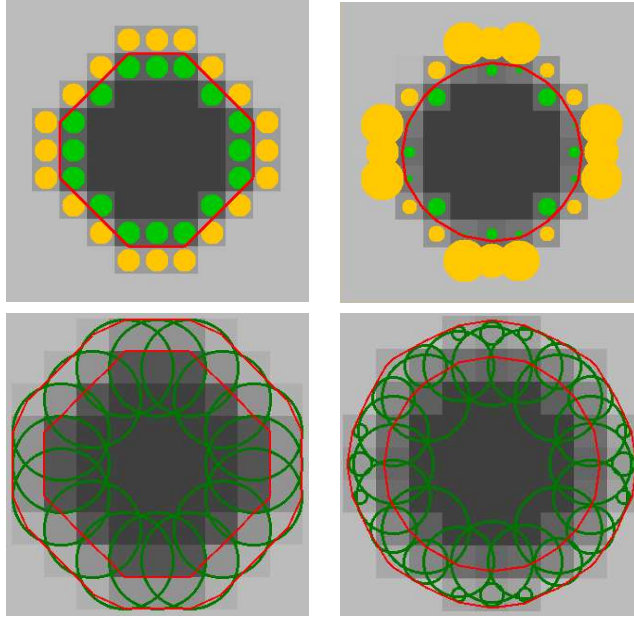


Figure 5: Example of the Distance Transform computed from a binary and from an interpolated distance to a circle of radius 3.3.

of the object boundaries. If we denote the initial distance estimate $\alpha_{\mathbf{n}} \in [-1, 1]$ for any point \mathbf{n} in the neighborhood \mathcal{S} of the contour and I the initial image, we can formally write our “exact” interpolated Distance Transform as:

$$\forall \mathbf{x} / I(\mathbf{x}) > 0, \quad D(\mathbf{x}) = \min_{\mathbf{n} \in \mathcal{S}} \{ \|\mathbf{x} - \mathbf{n}\| + \alpha_{\mathbf{n}} \} \quad (4)$$

If all the $\alpha_{\mathbf{n}}$ values were negative, we could draw small circles (or spheres), centered on \mathbf{n} and with radius $-\alpha_{\mathbf{n}}$, and interpret our Distance Transform as a distance to the boundary of an object created from these circles. Thus, if we consider the negative coefficients $\alpha'_{\mathbf{n}} = \alpha_{\mathbf{n}} - 1$, we can interpret the distance to the contour dilated by 1 as the distance to the set of circles centered on \mathbf{n} with radii $-\alpha'_{\mathbf{n}}$.

$$\forall \mathbf{x} / I(\mathbf{x}) > 0, \quad D(\mathbf{x}) - 1 = \min_{\mathbf{n} \in \mathcal{S}} \{ \|\mathbf{x} - \mathbf{n}\| + \alpha'_{\mathbf{n}} \} \quad (5)$$

The bottom line of figure 5 shows this set of circles for the binary (left) and the interpolated (right) distance estimate. In the case of the interpolated distance, the external contour of the set of circles offers a good estimate of the contour dilated by 1.

5 Experiments

5.1 Accuracy experiments

We performed experiments in two and three dimensions, by evolving an initial image (disk in 2D and sphere in 3D), under constant or mean curvature forces. In the case of constant speed evolution, the initial image was the signed distance to a disk/sphere of radius 30 pixels/voxels. The perfect evolution is then a slope $R(t) = 30 - t$, where t is the evolution time and R is the radius of the object. In the curvature evolution case, the initial image was the signed distance to a disk/sphere of radius 20 pixels/voxels. The perfect evolution is then given by $R(t) = \sqrt{20^2 - 2t}$. The error was computed on the set of points where the linearly interpolated level set crosses the grid in each of the main axis directions. The Root Mean Square error of the distance of these points to the center of the object shows that no significant additional error is introduced by the Chamfer distance approximation.

Results are depicted in figure figure 6.

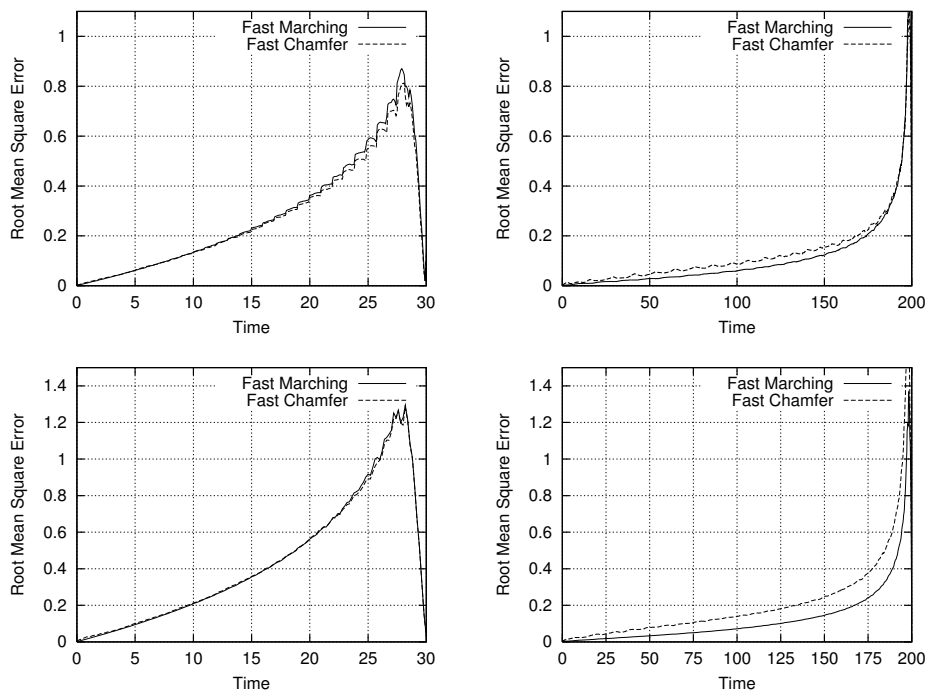


Figure 6: Root Mean Square Error on evolving a disk (top row) and a sphere (bottom row) under constant (left column) or curvature (right column) forces.

5.2 Speed experiments

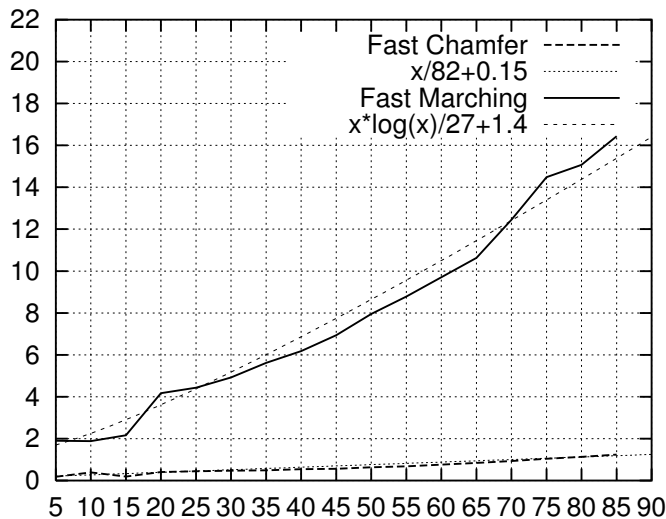


Figure 7: Computation of the Euclidean distance until distance 5, for spheres of different radii. Tested on images of size $200 \times 200 \times 200$. On the abscissa, the radius of the sphere. On the ordinates, the time in seconds for each method.

To test the speed of our new Narrow-banded Distance Transform, we compare it with the Fast Marching Algorithm [Set99a], which has been proposed for reinitializing the distance map in Level Set Methods [Set99b]. The Fast Marching uses a min heap structure that allows computing the distance by propagation from the initial contour, though it requires more processing. Moreover, it results in a complexity of $n \log(n)$ where n is the number of voxels of the narrow band. In comparison, our method has a complexity of n , because it passes through each point of the narrow band only twice. We performed experiments on synthetic images of size $200 \times 200 \times 200$, computing the Euclidean distance in a narrow band of size 5 voxels for spheres of increasing radii from 5 to 90 voxels with an increment of 5. Results of the computation time of the distance reinitialization, including the sub-voxel estimation for the points neighbor to the contour, are depicted in figure 7. The experiments have been carried on a Pentium III with 1.1 GHz running on the Linux operating system. Our method outperforms the usual Fast Marching distance reinitialization, and the processing is up to 20 times faster for a sphere of radius 90 voxels, resulting in a subsequent acceleration of the overall Level Set Method when segmenting objects in three-dimensional medical data.

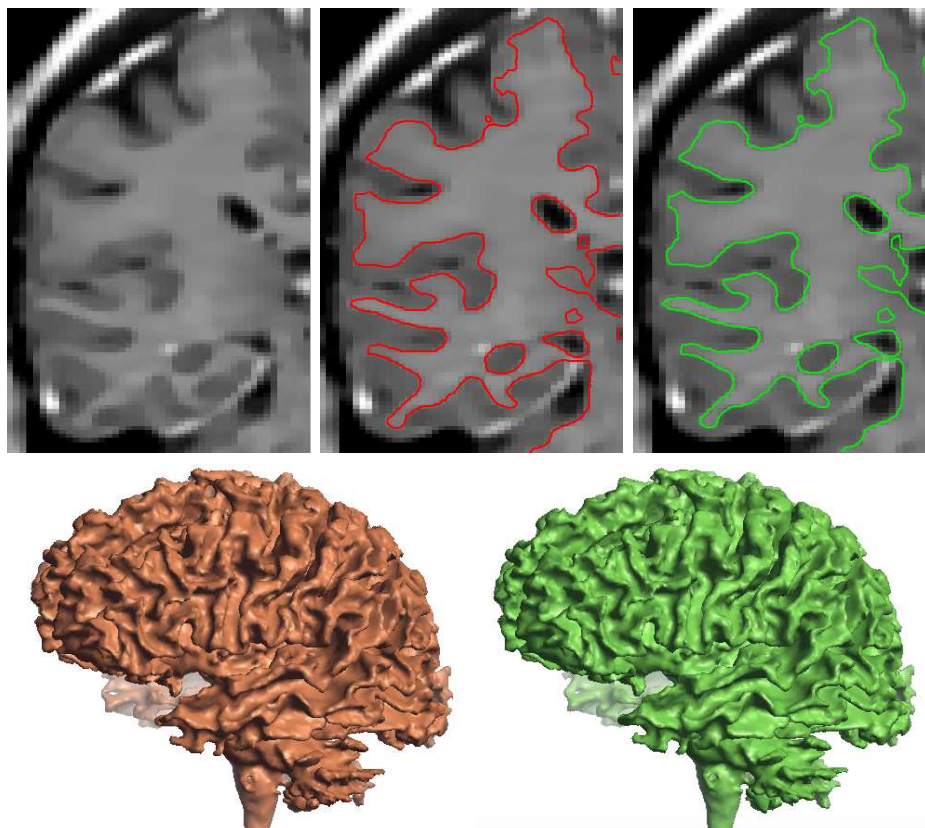
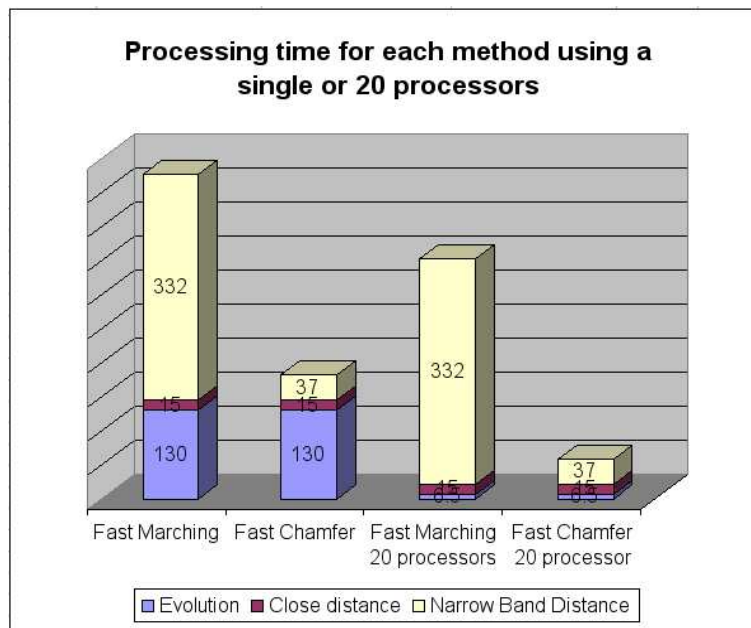


Figure 8: Result on segmenting the white matter in a SPGR MR Volume.

5.3 Segmentation of the white matter

We segmented the white matter on a 3D SPOiled Gradient Recalled (SPGR) Magnetic Resonance Image. We used a Level Set Method, initialized with five spheres of radius eight voxels each, in order to obtain a three-dimensional surface representing the white matter. We experimented with both Fast Marching and the proposed Fast Chamfer Distance for re-initializing the distance to the level set. In both cases, the Distance Transform was initialized with the distance to the interpolated contour as described in section 2.

Figure 8 shows the results obtained with both techniques. The top line displays a slice of the initial image on the left, and the isocontours of the segmented white matter using Fast Marching in the middle, and using Fast Chamfer Distance on the right. The bottom line displays the isosurface of the zero level set after 180 iterations, using Fast Marching on the left, and using Fast Chamfer Distance on the right.



processing	F. Marching	F. Chamfer	Ratio
Evolution	2 min 10 s	2 min 10 s	1
Close Dist.	15 s	15 s	1
Narrow Band Dist.	5 min 32 s	37 s	9
Total/1 process.	7 min 57 s	3 min 02 s	2.6
Total/20 process.	5 min 39 s	44 s	7.7

Table 1: Processing time for each method using a single or twenty processors.

While these results are almost equivalent, table 1 demonstrates the advantage of using the Fast Chamfer Distance Transform compared to the Fast Marching. The processing time for computing the distance in the narrow band is increased by a factor of nine, while the global processing time is increased by a factor of 2.6 using one processor. A parallel multi-threaded implementation can be used for processing the Partial Derivative Equation in the narrow band and for computing the close distance. It simply consists in splitting the narrow band into n parts, and processing each part independently. Unfortunately, none of the Distance Transforms can be parallelized because the value of a voxel depends on the values of its previously computed neighbors. When using 20 processors, the global processing time is then increased by a factor of 7.7.

Conclusion

We have presented a new, fast and accurate reinitialization of the distance map in a narrow band, as applied to Level Set Methods. We have also shown the importance of preserving the sub-voxel position of the level set during this reinitialization; we have further proposed a fast, linear algorithm based on the Chamfer Distance for computing the distance to the contour. Experiments confirm that our new algorithm outperforms the standard Fast Marching method while keeping enough accuracy in the Distance Transform. We have also interpreted applying the Distance Transform from an initial estimate of the distance to the interpolated contour, as compared to the usual binary input. We are currently investigating extensions of our algorithm to obtain exact distance with the same complexity, to take into account the voxel anisotropy in the Distance Transform and to adapt the algorithm for parallel processing.

Acknowledgments

Special thanks to Nancy Drinan for editing the manuscript.

Bibliography

- [Bor96] G. Borgefors. On Digital Distance Transforms in Three Dimensions. *Computer Vision and Image Understanding*, 64(3):368–376, November 1996.
- [Cui99] O. Cuisenaire. Fast Euclidean distance transformations by propagation using multiple neighbourhoods. *Computer Vision and Image Understanding*, 76(2):163–172, November 1999.
- [Dan80] P.-E. Danielsson. Euclidian distance mapping. *Computer Graphics Image Processing*, 14:227–248, 1980.
- [LC87] W.E. Lorensen and H.E. Cline. Marching cubes: A high resolution 3-D surface construction algorithm. *Computer Graphics*, 21(4):163–169, 1987.
- [OS88] S. Osher and J.A. Sethian. Fronts propagating with curvature dependent speed: algorithms based on the hamilton-jacobi formalism. *J. Computational Physics*, 79:12–49, 1988.
- [RP68] A. Rosenfeld and J Pfaltz. Distance functions in digital pictures. *Pattern Recognition*, 1:33–61, 1968.
- [RS00] G. Russo and P. Smereka. A Remark on Computing Distance Functions. *Journal of Comp. Phys.*, 163:51–65, 2000.
- [Set99a] J.A. Sethian. Fast Marching Methods. *SIAM Review*, 41(2):199–235, 1999.
- [Set99b] J.A. Sethian. *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision and Materials Science*. Cambridge University Press, 1999. New Edition.
- [SF99] M. Sussman and E. Fatemi. An Efficient, Interface-Preserving Level Set Re-distancing Algorithm and Its Application to Interfacial Incompressible Fluid Flow. *SIAM J. Sci. Comp.*, 20(4):1165–1191, 1999.
- [SSO94] M. Sussman, P. Smereka, and s.J. Osher. A level set approach for computing solutions to incompressible two-phase flow. *J. Comp. Phys.*, 94:146–159, 1994.

-
- [YI86] M. Yamashita and T. Ibaraki. Distance defined by neighborhood sequences. *Pattern Recognition*, 19:237–246, 1986.