HIERARCHICAL SKELETON EXTRACTION BASED ON A DEFORMABLE PARTICLE SYSTEM

Franck Angella[1], *Olivier Lavialle*[1,2] *and Pierre Baylou*[1]
[1] Equipe Signal & Image / ENSERB and PRC-GDR ISIS, CNRS
BP 99, 33402 Talence Cedex, France
[2] ENITA de Bordeaux
1, Cours du Général de Gaulle, BP201, 33175 Gradignan, France
Tel. +33 5 56 84 66 74 - Fax +33 5 56 84 84 06
e-mail: angella@goelette.tsi.u-bordeaux.fr

ABSTRACT

The use of a particle system as a skeleton extractor is introduced here. This system behaves as an active contour model embedding variable topology properties. As the particles propagate inside tree-shaped objects, we build their skeleton which is useful to determine the neighborhoods used for the computation of the regularization and interaction forces. This connected skeleton is directly returned and avoids a two-step approach based on a morphological operation followed by a tree analysis. In addition, the method gives information on the hierarchy of structures. Using this method, it is possible to generate a cartography of structures such as veins or channels.

1 INTRODUCTION

The problem of skeleton extraction has found many solutions essentially based on mathematical morphology algorithms [1]. Once the skeleton is obtained, identifying the oriented hierarchical structure of the skeleton thereof leads to use, for example, graph theory. Our paper proposes a different approach to find the skeleton of grayscale tree-like objects: it is based on active contour models [2]. Recently, different techniques have been studied to solve the problem of topology changes occurring while using active contours. Among them *particle systems* have been introduced to overcome the classical model limitation [3]. Their ability to propagate inside arteries and to pass through junctions leads to the extraction of skeletons, while simultaneously revealing the hierarchy of structures. In addition, particle systems methods can be linked with evolutionary algorithms given that each element is subject to general mechanical laws but moves independently [4][5]. By building a relation tree, we can determine the oriented structure of the objects while defining relationships between particles. Different methods are proposed in this paper to compute this tree using the trajectories of particles. Finally, some results are given on both synthesized and real 2-D images.

2 PARTICLE SYSTEM DESCRIPTION

Our algorithm simulates the evolution of a particle system which is subject to internal and external forces. Internal forces control and regulate the expansion of the system whereas external forces drive its evolution according to the image I we are working on.

2.1 Model Definition

We define a set of nodes (or particles):

$$\left\{ M_i | i = 1, ..., N_{part}^{(k)} \right\} \tag{1}$$

where $N_{part}^{(k)}$ stands for the number of particles at step k. Each particle M_i can move in the x-y image plane in accordance with the different exerted forces. These forces are divided into two groups: internal forces and external forces [3].

·*Internal forces:*

We define an interaction potential function gathering a long-range attraction term and a short-range repulsion term using a Lennard-Jones function [3]:

$$\phi(r) = \frac{B}{r^n} - \frac{A}{r^m} \text{ with } n > m \tag{2}$$

where A and B are positive coefficients used to set the equilibrium characteristics between two particles separated with distance r. Using ϕ we can compute a global interaction force $\mathbf{f}_{int}^{(i,k)}$:

$$\mathbf{f}_{int}^{(i,k)} = -k_r \cdot \sum_{M_j \in V(M_i,k)} \nabla\phi\left(d_{ij}\right) \tag{3}$$

where d_{ij} is the euclidian distance from node i to node j and ∇ denotes the gradient operator. k_r tunes the effect of the force and controls the global expansion speed of the particle system. $V(M_i, k)$ represents the neighborhood of the node M_i at iteration k. The determination of this neighborhood is described in section 3. A special case of formula (2) is the Coulombian electrical analogy ($A = 0$, $B = 1$ and $n = 1$) which leads to:

$$\mathbf{f}_{int}^{(i,k)} = -k_r \cdot \sum_{M_j \in V(M_i,k)} \frac{\mathbf{u}_{ij}}{d_{ij}^2} \tag{4}$$

where $\mathbf{u}_{ij}$ is the unit vector from node i to node j. The main drawback of the electrical analogy lies in its instability. Choosing non-zero values for parameters A and B is useful to attract distant particles and to push away nearby ones. So ϕ allows us to have an expansible and regulated system.

A second force $\mathbf{f}_{ph}^{(i,k)}$ is used to guide the particle according to the trajectory of others particles. We build a velocity map $\mathbf{I}_{ph}$ using the velocity of each particle at each step of the algorithm. This map allows us to know the *average* speed vector for a particle M_i given its position $\mathbf{r}_i(k)$ on the image I, *i.e.* the mean speed calculated using:

$$\mathbf{I}_{ph} = \overline{\mathbf{v}_j(m)}\Big|_{\mathbf{r}_j(m)=\mathbf{r}_i(k)} \tag{5}$$

where $\mathbf{v}_j(m)$ stands for the velocity of the particle M_j at iteration m with $m < k$. We can then define:

$$\mathbf{f}_{ph}^{(i,k)} = k_{ph} \cdot \begin{bmatrix} \mathbf{I}_{ph}^X \\ \mathbf{I}_{ph}^Y \end{bmatrix}_{\mathbf{r}_i(k)} \tag{6}$$

where $\mathbf{I}_{ph}^X$ and $\mathbf{I}_{ph}^Y$ are $\mathbf{I}_{ph}$ components in the x-y plane. k_{ph} controls the influence of all particles trajectories until iteration k on M_i. It acts like ant pheromones, driving particles preferably on the way already used by prior particles [4].

The last internal force is a regularization force $\mathbf{f}_{reg}^{(i,k)}$ weighted by k_{reg}. This force is used to correct the position of M_i with its neighbors in $V(M_i, k)$. When M_i has two neighbors, which means that three nodes are parts of the same branch, we add a force defined as follows:

$$\mathbf{f}_{reg}^{(i,k)} = -k_{reg} \cdot \left(\frac{\mathbf{r}_i^1(k) + \mathbf{r}_i^2(k)}{2} - \mathbf{r}_i(k) \right) \tag{7}$$

$\mathbf{r}_i(k)$ is the position of node M_i at iteration k and $\mathbf{r}_i^1(k)$ and $\mathbf{r}_i^2(k)$ are the positions of M_i neighbors. We can notice that this force is similar to the one created by the second order regularization term (rigidity) used in the *snakes* internal energy [2]. Thus, this tree includes regularizing properties, such as those of active contours.

·*External forces:*

We want our system to locally retrieve information from the image I and to be finally located inside objects. This is the reason why we use external forces $\mathbf{f}_g^{(i,k)}$ directly depending on the gradient of I, defined as follows:

$$\mathbf{f}_g^{(i,k)} = -k_g \cdot \nabla \left(\|\nabla G * I\|_{M_i} \right) \tag{8}$$

G is a filter used to extend the influence area of objects boundaries in I. k_g is used to control this influence.

A friction force is also used on M_i:

$$\mathbf{f}_f^{(i,k)} = -k_f \cdot \mathbf{v}_i\,(k-1) \tag{9}$$

k_f is the friction coefficient and $\mathbf{v}_i(k-1)$ is the velocity of M_i at step $k-1$.

So the global force applied on M_i at iteration k is:

$$\mathbf{f}^{(i,k)} = \left(\mathbf{f}_{int}^{(i,k)} + \mathbf{f}_{ph}^{(i,k)} + \mathbf{f}_{reg}^{(i,k)} \right) + \left(\mathbf{f}_g^{(i,k)} + \mathbf{f}_f^{(i,k)} \right) \tag{10}$$

2.2 Evolutionary Model

Here we do not solve the evolution problem with an energetic approach like in [2] or [6]. If we consider that each particle is a mechanical system, its evolution follows the classical Newtonian mechanical relation given below.

$$m_i \cdot \frac{d\mathbf{v}_i(t)}{dt} = \mathbf{f}_i(t) \tag{11}$$

where m_i is the mass of the particle i (in our case $m_i = 1, \forall i$). $\mathbf{f}_i(t)$ is the total resultant force on M_i. We use the Euler method to compute equation (11). At each step k, each particle moves according to:

$$\begin{cases} \mathbf{v}_i(k) = \mathbf{v}_i(k-1) + \mu \cdot \mathbf{f}^{(i,k)} \\ \mathbf{r}_i(k) = \mathbf{r}_i(k-1) + \mu \cdot \mathbf{v}_i(k) \end{cases} \tag{12}$$

$\mathbf{v}_i(k)$ and $\mathbf{r}_i(k)$ are the velocity and position vectors for M_i. μ is the time interval used as parameter for the Euler method. The more sophisticated Runge-Kutta numerical integration techniques could also be used.

All particles are created at the same location $\mathbf{r}_{init}$ with the initial velocity $\mathbf{v}_{init}$. Those parameters are chosen so that the particles will move inside the tree-shaped object in image I.

3 HIERARCHY TREE

One of our goals is to obtain the oriented skeleton of objects in the image I. In order to complete this task we generate a hierarchy tree during the particles' evolution. In addition, this tree is used to allow our system to change its topology. Using it, we can determine the neighborhoods $V(M_i, k)$ which are useful for the computation of $\mathbf{f}_{int}^{(i,k)}$ and $\mathbf{f}_{reg}^{(i,k)}$. In fact, these neighborhoods are necessary to know which particles have an effect on a given particle, depending on the object shape. For example, particles can be close whereas they must not interact. Figure 1 shows such a case.

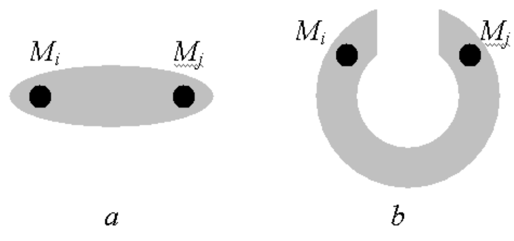

Figure 1: a) M_i and M_j must interact - b) M_i and M_j must be prevented from interacting.

A solution that uses the comparison between euclidian and geodesic distances is proposed in [7]. However in our case this solution can't be applied because of the large computational time needed.

3.1 Tree Generation

We propose two methods for the generation of the hierarchy tree. Both take advantage of the trajectories of all particles. This information is directly related to the orientation of the tree.

3.1.1 Post computation generation

The first method builds the hierarchy tree by directly using the trajectories of the particles. Let $C_i^{(k)}$ be the complete path of particle M_i from $\mathbf{r}_{init}$ to $\mathbf{r}_i(k)$. Our goal is to find the particle M_j linked forward to M_i so that M_iM_j is a part of the skeleton that we are searching for: M_j is the particle which is the closest to M_i in time and in space. In fact we scan the entire trajectory, beginning at step k, trying to find M_j as given by equation (13). If no particle satisfies the relation, we move back to step $k-1$ and so on until step 1.

$$j = \arg \min_{l \in \left[1; N_{part}^{(k)}\right] \setminus \{i\}} \{\|\mathbf{r}_l(k) - \mathbf{r}_i(m)\| \mid \|\mathbf{r}_l(k) - \mathbf{r}_i(m)\| \leq D\} \quad (13)$$

where D is used to enlarge the trajectory $C_i^{(k)}$ and m is the current observed iteration varying from k downto 1. This method gives us the closest neighbor (in time) which is the closest to a given curve (in space). That way, we can build the complete hierarchy tree after running the evolutionnary algorithm.

3.1.2 Step-by-step generation

For the second method we consider that the tree can be generated by modifying its structure at each step of the evolution. Links between particles are changed according to the geometric relations with their neighbors. We study different cases corresponding to the different configurations of each node: additions and convergences, deletions, divergences and shifts.

·*Addition and convergence:*

The technique is exactly the same as the one used for the first method (using equation 13). A link can be created from node M_i to node M_j if M_i has no neighbor or if this link has been cut before. We can speed up the algorithm by searching the neighbor in a restricted list which contains the parent nodes of M_i. By rebuilding the tree during the system evolution, this list can easily be generated.

·*Deletion:*

A link is removed when the inter-distance between its nodes is longer than a given threshold. This case often occurs for the leading particle of a branch.

·*Divergence and shift:*

This situation happens for junctions. Considering three nodes M_1, M_2 and M_3, if the angle $\widehat{M_1M_2M_3}$ becomes acute (which is determined by comparing its value to a given threshold), all links to M_2 are cut off. Then, new branches are added using the *addition and convergence* method.

3.2 Tree Use for the System Evolution

Rather than using a static method to solve the topology changes issue [8], we take advantage of the trajectories of all particles to link them dynamically. Given the tree, we know which are the links from one particle to another. If a link exists from M_i to M_j, then M_j is a part of $V(M_i, k)$ at step k. These neighborhoods are used to compute the regularization forces and the intereaction forces. In order not to compute the whole tree at each iteration, a test is used to start a new computation. In fact, as the tree structure changes during the evolution, neighborhoods have to change too because they stop revealing the real topology of the image and lead particle onto wrong paths. These mistakes can be detected by studying the links between particles: if a link crosses an object boundary, then this link must be cut off. Thus, during the evolution the algorithm scans the image I along each link. If a boundary is detected (using a comparison with a given threshold), then a new computation of the tree is needed.

4 RESULTS

The particle system algorithm is applied to synthesized images in order to find the skeleton of objects. Figure 2 presents the results for tree generation using our method. The tree generation is computed directly from the trajectories as described in section 3.1.2.

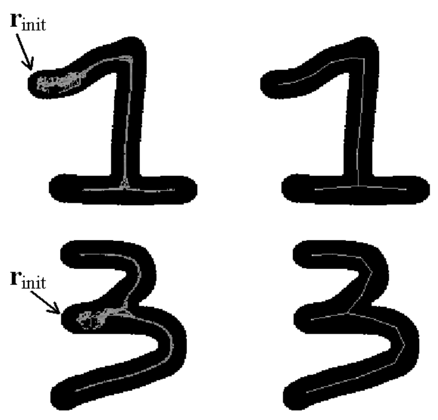

Figure 2: On the left, particle trajectories, on the right, final trees (arrows show initialization points).

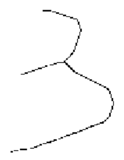

Figure 3: Skeleton extraction using Marthon morphological method.

These results show that the algorithm, which is a totally new approach, produces a better skeleton than does a morphological operator for simple binary images (see figure 3).

In addition, this method can find the skeleton of grayscale objects as shown in figure 4.

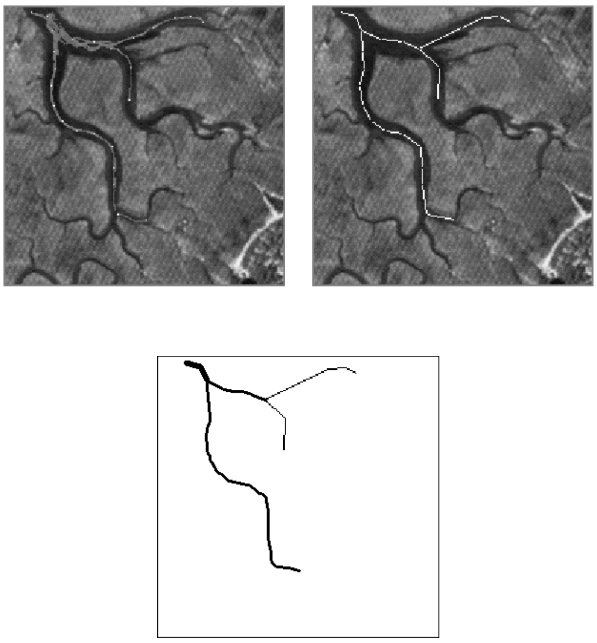

Figure 4: Particle trajectories, main ramifications of the skeleton and hierarchical tree for a satellite image. Particles were generated at the top left of the picture. In the third figure, the line width expresses the hierachical position of the ramification.

As the particles first move through large ramifications, the final tree reveals information about the hierarchy of the objects. Furthermore, in the case of complex shapes the hierarchy involved in the skeleton graph leads to a simple burring method.

5 CONCLUSION

The simultaneous use of particle systems and tree generation allows us to extract skeletons. Results are better than those obtained by mathematical morphological approaches because the method returns a linear piecewise and connected skeleton and gives additional information on the hierarchy of structures.

We are currently working on particle leakages detection. This problem occurs when the particle density reaches a too high level, creating local explosions. We also study the 3-D extension which leads neither to new issues nor to computation time excess.

References

[1] J. Serra, *"Image Analysis and Mathematical Morphology"*, Academic Press, Vol. 1.

[2] M. Kass, A. Witkin and D. Terzopoulos, *"Snakes: Active Contour Models"*, International Journal of Computer Vision, 321-331, 1988.

[3] R. Szeliski and D. Tonnesen, *"Surface Modeling with Oriented Particle Systems"*, in SIGGRAPH'92 Conference Proceedings Computer Graphics, Chicago, IL, pp. 185-194.

[4] M. Dorigo and L.M. Gambardella, *"Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem"*, IEEE Transactions on Evolutionary Computation, Vol. 1, No. 1, April 1997.

[5] J. Liu, Y.Y. Tang and Y.C. Cao, *"An Evolutionary Autonomous Agents Approach to Image Feature Extraction"*, IEEE Transactions on Evolutionary Computation, Vol. 1, No. 2, July 1997.

[6] H. Yahia and I. Herlin, *"Image Processing of Meteorological Images with Implicit Functions"*, ICAOS'96, Paris.

[7] O. Lavialle and P. Baylou, *"Morphologie mathématique adaptative d'inspiration coulombienne"*, RFIA'98, Clermond-Ferrand, Jan. 1998.

[8] T. McInerney and D. Terzopoulos, *"Topologically Adaptable Snakes"*, Proceedings of the Fifth International Conference on Computer Vision, Cambridge, MA, June, 1995.