Can Ants Build Urban Street Networks ?

Xavier Marsault

Can Ants Build Urban Street Networks ?

Xavier Marsault[1]

[1]MAP-ARIA Lab, Architecture School of Lyon, 69512, France (xavier.marsault@aria.archi.fr)

ABSTRACT

Our paper deals with the detection and automatic extraction of a hierarchical network of urban streets from maps containing only building footprint data. We develop a new approach for extracting, locating and labelling plausible street networks in a given city, based on geometrical an functional considerations. Using some basic tools from the « Mathematical Morphology » field, we propose simple, robust and efficient techniques for extracting homotopic skeletons of the ground zones and a street-width map. Our method leads to the construction of open or closed connex graphs that we encode and save. Finally, we develop some ant-based techniques to identify plausible elements (streets, boulevards, avenues, lanes, water streams) in this graph.

Keywords: street network, skeleton, periphery, path, mathematical morphology, ant colony optimization (ACO)

1. INTRODUCTION

Due to the rising number of applications in urban development and entertainment industries (video games, animation), automatic generation of complex urban scenes is a fundamental challenge. Modelling complex and realistic cities is still an important research field in urban planning and computer graphics, well covered by recent international publications [2, 3, 8, 10, 11, 14].

There is virtually no semi-automated system to support or help modelling and planning in very large metroplitan areas [1]. Particularly, there is a need for original techniques producing complex 3D cities with automatically generated hierarchical street - and even underground - networks. Taking this into account has received very little attention [5], mainly because of the high complexity of the arrangement and organization that need to be computed.

We can define a city as a set of disparate objects (buildings, furniture, miscellaneous roads), whose coupling can be modelled by networks [2, 9]. The relationship between street network and buildings usually results from a nested rationale. On one hand, for historical reasons, and in the vast majority of cases, the structure of the streets and its extensions define the city shape and govern the filling of the plots and parcels. Such a configuration naturally accepts an underlying graph description, which is the very purpose of the latest research. On the other hand, some recent research [7, 10] give priority to the buildings implementation. Indeed, whenever we have to study the design or simulate urban planning, we have to rely on real data. And buildings are robust ones, better than streets. In such a context, automated generation of street networks from urban footprint maps is an interesting research topic. This “raster approach” of the problem - rather new comparing to the vectorial one [5] - avoids the difficult task of shape vectorization in noisy environments.

The framework of our paper, therefore, exceeds given existing road data in order to run simulations, traffic calculation, maintenance and adaptation of existing routes, and thus requires in any case the computation of a multi-potential graph, which justifies the approach of section 3. The next section explores some recognition techniques and automatic classification of road elements. Designed to run on the resulting graph from section 3, it can be used on any existing urban graph. Finally, we suggest a reinterpretation of our work under the theory of ant stigmergic behavior, followed by an opening towards more ambitious optimization computations.

2. RELATED WORK AND GENERAL APPROACH

Our work is similar to the vectorial approach of Decoret [5], which computes a non-hierarchical road network from a city map of building outlines by calculating the Voronoï diagram of the ground. This allows to get a wired network of roads for which there is always a path between any couple of buildings. This purely geometric approach is insufficient in our view, because it lacks a road classification, certainly difficult to obtain, which forbids its use for simulating dynamic problems of the real world (traffic, transport, urban planning).

Instead, we propose a dual raster approach, where the the Voronoï diagram is replaced by a morphological skeleton computation. The unique input data is the the building footprint image *I* (Figure 1). The ground is in white and the buildings are in grey levels which may be used to encode various properties, such as building height or housing density. This approach let us link our work to the stigmergic behavior of social insects (see section 5.3), although this is not evident in a purely vectorial approach like [5]. We expect this metaphor to be further exploited to automatically produce an urban hierarchical network, using "ant colony optimization (ACO)” [6, 13], a recent metaheuristic which provides very good results for graph optimization problems.

978-1-4244-4136-5/09/$25.00 ©2009 IEEE

Figure.1 – Building footprints in Vénissieux (France)

3. COMPUTING A GOOD STREET SKELETON

3.1 Morphological skeleton of the non-built area

Our first objective is to identify plausible routes in a city, given the image *I* of its building footprints. Considering that everything which is not a building has the potential to become a road, we define the set I_s as *I* amputated of its buildings. This gives a new connex object, which can be considered as the skeleton of all potential roads at first approximation. That we choose to calculate is obtained by a homotopic process which simply consists in sequentially applying the Golay thinning masks of Figure 2 up to image stabilization [6]. In order to improve the morphological processing quality which highly influences the skeleton precision, we first super-scan original images by a factor of 4.

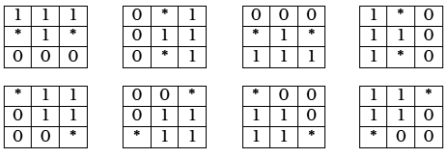

Figure.2 – The Golay thinning masks on a square grid
(0 = ground ; 1 = building ; * = 0 or 1)

We can see the results of the algorithm in Figure 3. We obtain two types of graph, depending on the possibility to connect the city to its environment (opened city), or not (secluded city). The obtained skeleton is sparseness of barbs because of the noise sensitivity of the skeletonization processing. These barbs (Figure 5a) consist of small arcs that have a free end (no neighbor), and have no major interest in a road network (at most, some may be interpreted as small impasses).
Moreover, as the city is surrounded by an infinite ground surface, skeleton (a) is "open". In fact, the used homotopic procedure preserves the image topology. The "closed" skeleton (b) is obtained by adding a 1 pixel "virtual building" around the image before thinning.
There just remains four barbs linking the city to the corners of the image. Therefore, the image closure on its edges introduces a kind of artificial suburb that does not match enough to the city content. So, at this step, two issues remain to be solved: the outskirts of the city (in section 3.3, we will provide a much more clever solution for this problem) and barb removal. One more time, barb elimination is performed using a Golay morphological filter (Figure 4). This cleaning step is done with 8 other masks applied until idempotence.

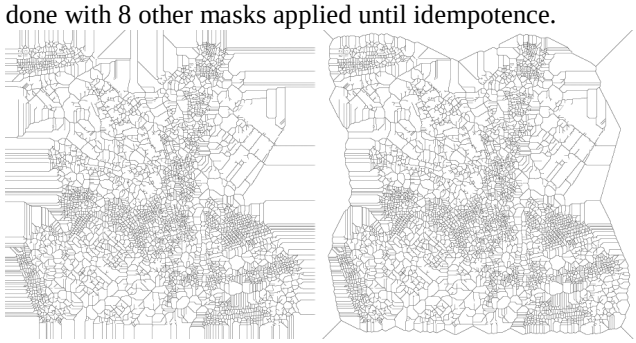

Figure.3 – Open (a) and close (b) skeletons

We now have a clean skeleton (Figure 9), made of cells containing each one a unique building, as it happens in a Voronoï diagram. However, apart from the fact that this object represents all the "center lines" of the original shape, we have no other information, including the width of potential streets. It would be interesting to know the maximum space available at each pixel of the image, and thus at each pixel of the skeleton. In the Mathematical Morphology's terminology, this is called the "maximum-ball map" or "distance-map".

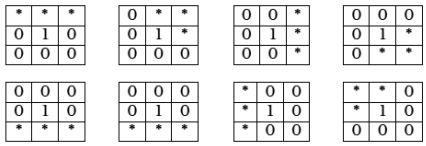

Figure.4 – The Golay barb-removal masks
(0 = ground ; 1 = building ; * = 0 or 1)

3.2 Distance map

For each pixel *p* of *I*, we wish to calculate the largest ball centered in *p* and which does not intersect any building (figure 5b shows several examples). The image of the radius for each pixel is called the "distance map". Mathematical Morphology provides a very elegant solution for computing the maximum-balls radii with the use of neighborhood-based operators. Let *B(p)* denote the ball surrounding *p* (see next section). The algorithm is described in the following pseudo code:

Each building-pixel becomes a full pixel
Each pixel *p* is assigned a count variable *c(p)* initialized to zero
While there remains a ground-surrounded pixel
 For each pixel *p* in the image
  if *B(p)* does not contain a full pixel
   $c(p)+1 \rightarrow c(p)$
  end if
  else
   p becomes a full pixel
  end else
  image update with new full pixels
 end For
end While

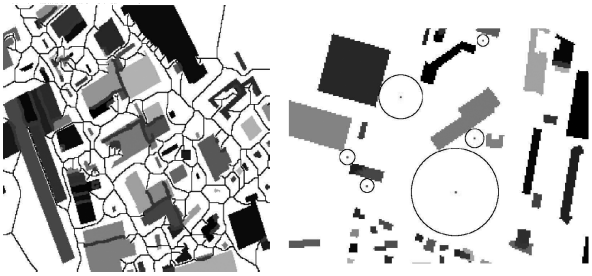

Figure.5 – Skeleton barbs (a), maximum-ball examples (b)

With this simple algorithm, we get the radius of the maximum-ball for each pixel of the image. On a square grid, the concept of ball (circle / disc) is approximate. Two 3x3 masks are available to fit a circle: either the cross (called ball V_4), or the filled square (called ball V_8). Experiments showed that the best approximation (in terms of euclidian distance) of the circle is to iterate alternately V_4 and V_8 balls. Moreover, the $V_8V_4V_8$... ball (Figure 6b) provides the best results in relation to the skeleton valuation. Once this procedure has been performed, if we turn the counter variable of each pixel into a grey level, we obtain “visual results”, as shown in Figure 7. We clearly recognize broad zones: the periphery but also some areas within the city.

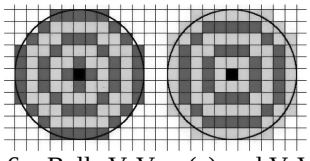

Figure.6 – Balls $V_4V_{8...}$ (a) and $V_8V_{4...}$ (b) after 5 dilatations

3.3 Computation of a plausible periphery

Figure 3 shows that the raw skeletonization is hard to use on the edges. By adding a "virtual building" on the edges of the image, the skeleton is enhanced with a periphery, but it is a too distant and unrealistic one. An improved solution consists in simply setting a virtual adaptive frontier all around the city to shape a periphery.

Figure.7 – “Maximum-ball map” or “distance map”

Thanks to the distance-map, we can calculate it by specifying the minimum distance *D* to the closest buildings. This "pseudo-expansion" of the city edges is not a new mathematical morphology tool, and its process is rather simple. The distance-map is used to draw in the image all the maximum-balls whose centers are located on the image borders, respecting the distance *D*. The first iteration already provides a rather fine periphery. We can iterate the process by considering all the pixels located in the demarcation line resulting from previous drawn circles, and so on until satisfaction.
Figure 8 illustrates this process, where each color corresponds to an algorithm iteration. Finally, starting from the city image completed with a periphery, we get a closed, cleanned and manageable skeleton (Figure 9). This is achieved by combining a topological information provided by the skeleton with a measure information given by the maximum-balls radii, and also by filtering out small sections that can not correspond to any road element. The next step introduces some image analysis in order to extract a graph from the skeleton.

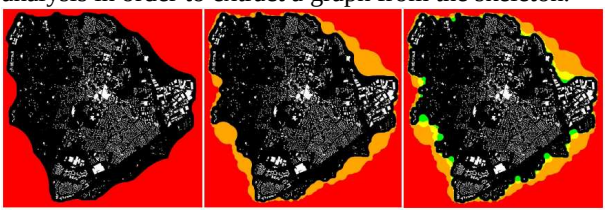

Figure.8 – Computed iterative peripheries 1(a),2(b),4(c)

4. GRAPH COMPUTATION FROM SKELETON

By construction, the skeleton object has a planar graph structure with junctions (edges) and nodes. We use image analysis to convert it into a graph, and proceed in two steps: nodes detection and junctions extraction.

4.1 Node detection

Let’s define a node as the intersection point of at least three junctions. To detect a “node pixel”, we may test at most 33 masks coding 3x3 neighborhood configurations represented in Figure 10. We sequentially apply these filters to each pixel until one of them meets one of the above configurations. If so, corresponding pixels are marked as nodes. Once all the nodes have been recognized and stored, we can proceed with junctions.

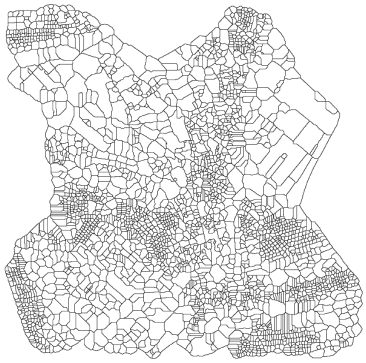

Figure.9 – Homotopic close skeleton (cleaned, filtered)

4.2 Extracting junctions

In order to follow the junctions from node to node, we use the Pavlidis algorithm, whose description can be

found in [6]. The existence of a path from a given pixel is determined using a neighborhood configuration, as the calculation of the new direction. Once junctions are extracted, we also store meaningful statistics, such as:
– the number of pixels of junction j : N_j
– the distance between extrem nodes : L_j
– minimal, maximal and mean values (R_MIN_j, R_MAX_j, R_AVG_j) for the ball radius along the junction
– a straightness coefficient: $Rect_j = L_j / N_j$

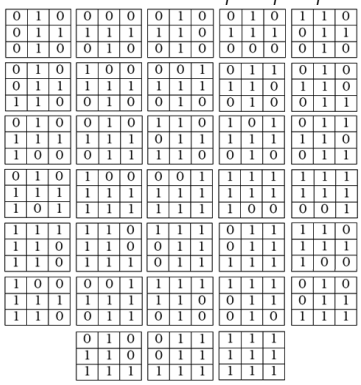

Figure.10 – All types of nodes in a 3x3 neighborhood in a square grid

4.3 Formatting the graph

Once the entire skeleton has been stored both in a node-list and in a junction-list, we can build the associated graph. On the skeleton of a city like Vénissieux, there are about 5700 nodes and 8500 junctions. This is generally a 1-graph (there is at most one edge between any arbitrary two nodes), whose density (ratio of the number of edges on the high number of nodes raised to a power of two) is very low. Encoding it by an adjacency matrix is very effective in memory. If we denote by N the number of nodes of the graph, and M the number of edges, it consumes at least N^2 memory locations. The encoding scheme using adjacency list is better in the case of low-density graphs, because it requires only $(M + N + 1)$ memory locations. The advantage of a matrix representation is the simplicity for the construction of predecessors and successors, while the representation list does not include immediate test for the presence of an arc - a test that we may need. A compromise was made with a matrix structure using C++ *map* and *vector<bool>* types, whose size is N^2. The standard C++ ensures that the vector template for Boolean types uses only one bit per element. The graph is rebuilt from the skeleton data stored in an XML file using the TinyXML library; and any application can use this data file thereafter. Reading and formatting the graph does not take more than 0.4 s on our computer (dual Core^2, 2.93 GHz).

5. CAN ANTS BUILD STREET NETWORKS ?

5.1 Methodology

We suggest a clever exploration of the multi-potential graph that we have extracted to provide a qualitatively and quantitatively relevant solution to the problem of automatic road element recognition. First let's consider that in real life, the most often used roads are the widest. Our goal is to perform some graph simplifications, by extracting a hierarchical classification of the longest, widest and as possible the most uniform road parts that have an identity within the body made up of small alleys, streets, avenues, boulevards, roads and rivers. In a comprehensive manner, we wish to minimize as possible the number of such elements. However, they are composed with connex junctions of the graph, and the first difficulty is to find their most relevant extrem nodes, from a geometric and functional point of view. Of course, the node valence is small (≤ 4), but the skeleton is very sensitive to noise, and many nodes are connected with very short junctions, which may imply a merging task, increasing the resulting node valence. In addition, any graph junction is not necessarily a plausible way: those resulting from the skeletonization of vast open areas (wasteland interstitial spaces, underserved ones) are such examples.
A first step to determine the most traversed roads uses a variant of the Dijkstra's shortest path algorithm [8] which is applied to all graph junctions and coupled with an ant-marking technique. We analyze this procedure and its limitations in details, before seeing how we could simulate street network emergence with ACO.

5.2 Automatic determination of major axes

By following the widest junctions of the graph, it is already possible to identify rivers running completely through a city, if any, and provide bridges at intersections with the possible streets. But this strategy can only be applied for rivers. For other road type, the computation is done in three steps. First, we borrow from graph theory the "shortest path algorithm" of Dijkstra, whose junction valuation is modified to take account for R_MIN_j and N_j parameters. It can exhibit a best path between any pair of nodes, according to those parameters (Figure 11). Then, results from the first step are used to automatically obtain the classification of the main and secondary roads, which is only affected by parameters S and α (see next section). The underlying idea is based on "pheromone deposit" made by ants, only in order to mark the most traversed axes. In this purpose, we attach a counter to each junction and launch an overall calculation of shortest paths between all pairs of nodes. We obtain a traversal frequency which is a pre-classification of all junctions (Figure 12). In a last step, which is already under development, we statistically process this map with a rectitude criteria in order to connect junctions belonging to the same axes, and then classify all recognized axes (rivers, boulevards, avenues, streets, lanes,...).

5.2.1 Results in interactive mode

To calibrate different graph junction valuation types, an interactive tool was developed. It allows the user to

select with the mouse arbitrary couples of nodes in the skeleton, and display the shortest path joining them. The algorithm used is that of Dijkstra with a heap data structure [4], whose complexity is *O(n.log(n))*. In such a minimization problem, the native valuation V_1, which is the length of the junctions, provides the optimal pedestrian path, which is not concerned with the street radius and can go through narrow places.

So, we introduce a new valuation to take radius values into account. Remember that R_MIN_j junction value is the maximum width of a candidate-road linking the two extrem nodes. Suppose we pre-compute the largest minimum-balls $max\ \{\ R\_MIN_i\ \}$ for all graph junctions. From there, we valuate each junction *j* as the difference of this value and the R_MIN_j value. Thus, the lowest valuation is given to the junction whose R_MIN_j value is highest. And this gives more chance to the shortest path to be found where there is more space.

In figure 11, green dots represent departure and arrival nodes chosen with the mouse. The red path corresponds to the pedestrian one, and the blue one is for V_2 valuation. A large majority of blue and red paths are the same when the path is short. If we select some appropriate distant nodes, the algorithm quickly retrieves paths visually matching those of reality. The blue route tends to avoid places with a multitude of small branches (where the space between buildings is small), which validates our tool. Two parameters are introduced in order to control the algorithm behavior: a threshold *S*, which prohibits going through a junction with a radius less than *S*, and a coefficient α to balance the weight of R_MIN_j. This leads to the following valuation V_{2j}:

$$V_{2j} = N_j * (max\ \{\ R\_MIN_i\ \} + 1 - R\_MIN_j)^{\alpha} \quad (1)$$
$$\text{with } 0 \leq \alpha \leq 2$$

We add 1 to avoid a zero valuation for the junction. The more α decreases, the more we get closer to the pedestrian path if there is no minimum threshold. If $\alpha = 0$, then $V_{2j} = V_{1j}$. Otherwise the path is forced to go through junctions with higher radius values. In our experiment, 2 appears to be a sufficient limit for α values. We must now find an automatic method to extract the main routes, then medium and small ones.

5.2.2 Parsing the graph with virtual ants

We start by filtering the junctions whose maximum width is less than a threshold set by the user. Then we provide each junction with an ant crossing counter. Depositing pheromone is only used to increment the counter, and does not influence the shortest path algorithm behavior. Then, we launch an overall process computing the minimal paths for any couple of nodes within the graph. These paths are not saved, but each time a connection is used, its counter is incremented. With an elaborate implementation, the whole computation takes 30 seconds on our computer, for the graph shown in Figure 9.

Paths that appear in dark blue are the most visited, and those that are pale blue the least visited. Some major axes of the city appear very clearly, others less. There are very few routes heavily marked in the dense residential areas, which means that the algorithm gives priority to the broader zones.

Figure.11 – Search path simulation in the graph

The last step, currently in progress, is to take into account a directional pheromone, by creating for each node a routing table (storing conditional statistics of order 1). Each table records the probability for ants to follow the junctions connected to this node, when they arrive from a given junction. Local statistics on the flow should enable us to improve the monitoring of junctions (using the maximum values of the routing table) and the detection of extreme nodes (when routing is equivalent in several directions). Once both extremities are found for a path, we replace all the R_MIN_j junction values by their minimum value computed along the path. This results in lines of constant width, easy to classify according to the characteristics of the “road body” used. Starting and ending nodes are either street intersections, or place centers, or bridge extremities.

Figure.12 – Automatic results (most visited axes are in dark blue)

5.3 Reinterpreting Mathematical Morphology *(MM)*

MM is based on neighborhood transformations (erosion, dilation, thinning, barb removal...) repeatedly applied on a form. Selected size for neighborhood always being limited and small, we can interpret the *MM* operators as virtual ants performing stigmergic and uncoordinated actions, ie local and only guided by previous work configurations (the state of the form). So, it can be legitimately argued that virtual ants should be able to extract the non built urban space skeleton.

5.4 Simulating street network emergence with ACO

We are undoubtedly facing a combinatory graph problem, with multiple local and global constraints, as mentioned in previous sections. If we can exhibit some "objective functions", we will be able to use metaheuristic techniques to solve this problem [13]. In this case, ACO is likely to help us finding good solutions, as it has been demonstrated in the context of automatic generation of architectural plans [7].
It has been studied for some years that ants perform all their work using two strategies: stigmergy and recruitment. Stigmergy is a "whole communication" using a collective memory: pheromone deposit. When ants choose their path, they tend to select the one which has the highest pheromone concentration. In this way, they are ensured to find their way back to their nest. Each ant is heading by following the pheromone tracks which are left by other members of the colony. Ants choose their path in a probabilistic way. As pheromones gradually evaporate, the probability that takes an ant to choose its path is changing over time.This strategy has inspired researchers and given birth to metaheuristics to solve problems known to be difficult (NP-complete), ranging from graph exploration to task planning [4, 8]. Readers are invited to consult reference book [13] for more details.
Simulating street network emergence with ACO is the next step of our work, which is still in progress and cannot be currently presented in this paper. The idea we follow is to let ants deposit a second type of pheromone at the starting or ending nodes of potential paths, while maintaining the shortest path approach described above.

6. CONCLUSION AND FUTURE WORK

We have put forward the first steps of a modular and robust protocol for extracting a hierarchical network of urban roads, drawing on Mathematical Morphology tools, graph exploration and ant behavior. Our contribution includes:
- automatic and parameterizable periphery calculation,
- skeleton calculation that represents a set of potential routes that can be used to connect a city to its neighbors,
- a good heuristic injected in the shortest path algorithm suited to search for wider and regular streets.

We have suggested a reinterpretation of Mathematical Morphology as a result of a stigmergic process achievable by an ant colony. This allows us to unify our work and to positively respond to the question: « Can ants build urban street networks ? ».
From a modelization point of view, the geometric aspect of road generation has not been addressed. We could have simply created road sections for each junction of the skeleton, but the whole result would not have looked like a real urban fabric. In addition, our work raises interesting detection and modeling questions about automatic placement of urban squares. Furthermore, the skeletonization process should be adapted in the future in order to allow roads to be closer to buildings, as it often happens in a real city.
Thus, after many experiments, our work leads to the production of innovative solutions able to consider soon:
- the development of a visualization tool for our street network 3D models, gathering all extracted objects (roads, rivers, intersections) and their fittings. We will use geometric primitives employed by civil engineers for road construction (line, circle, clothoid).
- the extension of our algorithm to deal with other decision-making problems, including urban planning. Our generic approach is very open.
- the design of specific algorithms to generate road and underground networks, adapted to terrains with relief.
- the integration as an overall constraint of the fractal "inverse power law", often called "the human range of scales" [12], which characterizes man-made structures that exhibit a network of "living connections".

REFERENCES

[1] Batty M, "Model Cities", Center for Advanced Spatial Analysis, paper 113, 2007.
[2] Batty M, Longley P.A., "Fractal Cities: a Geometry of Form and Function", Academic Press, London and San Diego, 2004.
[3] Chen G, Esch G, Wonka P, Muller P, Zhang E, "Interactive Procedural Street Modeling", TOG Vol 27, Issue 3, Siggraph 2008.
[4] Cormen T, Leiserson E, Rivest R, Stein C, "Introduction à l'algorithmique", Ed.Dunod, 2004.
[5] Decoret X, Sillion F, "Street Generation for City Modelling", ARTIS - GRAVIR – INRIA, 2002.
[6] Graffigne C., Zerubia J., "Analyse d'images : filtrage et segmentation", Masson, 1995.
[7] Ireland T, "Sniffing Space", *11th Generative Art Conference*, 2008.
[8] Larive M, Gaildrat V, Dupuy Y, "Génération Automatique de Zones Urbaines: un Etat de l'Art", AFIG, 2004.
[9] Mangin D, Panerai P, "Projet urbain", Editions parenthèses, 1999.
[10] Marsault X, "Generation of textures and geometric pseudo-urban models with the aid of IFS", in *"Chaos in Art and Architecture"*, Int. Journal of Dynamical System Research, vol I, number 3, 2005.
[11] Muller P, Wonka P, Haegler S, Ulmer A, Van Gool L, "Procedural Modeling of Buildings", TOG Vol 25, Issue 3, Siggraph 2006.
[12] Salingaros N, "A universal rule for the distribution of sizes", *Environment and Planning B : planning and Design*, 26:909-923, Pion Publications, 1999.
[13] Solnon C, "Optimisation par colonies de fourmis", Hermès, 2008.
[14] Yoav I, Parish H, Müller P, "Procedural modeling of cities", *SIGGRAPH,* ACM Press New York 2001.