Hierarchical Path Planning for Situated Agents in Informed Virtual Geographic Environments

Mehdi Mekni
Department of Computer Science and Software Engineering
Laval University
Quebec, Canada
mehdi.mekni@ift.ulaval.ca

ABSTRACT

Multi-Agent Geo-Simulation (MAGS) is a modelling and simulation paradigm which involves a large number of autonomous situated agents of various extents evolving in, and interacting with, an explicit description of a geographic environment called a *Virtual Geographic Environment* (VGE). One of the most important skills of autonomous situated agents is their ability to navigate and plan a path inside a VGE. Path planning in MAGS has to be solved in real time, often under constraints of limited memory and CPU resources. Moreover, the computational cost of path planing increases in complex and large-scale VGEs. In addition, most current planners only provide agents with obstacle-free paths and do not take into account the environments' topologic and semantic characteristics nor the agents' capabilities. In this paper, we extend the automated approach to build a semantically-enhanced and geometrically-accurate VGE called an *Informed VGE* (IVGE) that we proposed in [21]. Then, we propose our Hierarchical Path Planning (HPP) algorithm which relies on the topologic graph of the IVGE, and takes advantage of this IVGE's semantically-enriched description in order to provide autonomous situated agents with optimised paths with respect to both the environment's and the agents' characteristics.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: E.1Data Structure; D.2.8 [**Software Engineering**]: H.1.1Systems and Information Theory [Design Tools and Techniques]

General Terms

Design, Algorithms, Optimisation

Keywords

Multi-Agent Geo-Simulation (MAGS), Geographic Information System (GIS), Informed Virtual Geographic Environment (IVGE), Autonomous Situated Agents (ASA), Situated Reasoning (SR), and Hierarchical Path Planning (HPP)

1. INTRODUCTION

Multi-Agent Geo-Simulation (MAGS) is a modelling and simulation paradigm used to study complex systems in a variety of domains including traffic simulation, crowd simulation, and urban dynamics, to name a few. Such complex systems (i.e. car traffic, crowd behaviours, etc.) involve a large number of simulated actors (implemented as software agents) evolving in, and interacting with, an explicit description of the geographic environment called a *Virtual Geographic Environment* (VGE). VGEs are usually oversimplified and represented as being composed of only free spaces and obstacle regions. However, real geographic environments may be complex and large scale which makes the creation of a VGE difficult and needs large quantities of geometrical data originating from environmental characteristics (terrain elevation, location of objects and agents, etc.) as well as semantic information that qualifies space (building, road, park, etc.). The problem of path planning in MAGS involving complex and large scale VGEs has to be solved in real time, often under constraints of limited memory and CPU resources. Classic path planners provide agents with obstacle-free paths between two located positions in the VGE [12]. Such paths do not take into account the environment's characteristics (topologic and semantic) nor the agents' types and capabilities. For example, classic planners assume that all agents are equally capable of reaching most areas in a given map, and any terrain portion which is not traversable by one agent is considered to be not traversable by the other agents [4]. Such assumptions limit the applicability of these planners to solve only a very narrow set of problems: *path planning of homogeneous agents in a homogeneous environment.* Our goal is to address the issue of path planning for agents with different capabilities evolving in complex and large scale geographic environments of various extents.

In order to achieve such a goal, a VGE must precisely represent the geometrical information which corresponds to geographic features. It must also integrate several semantic notions about various geographic features. To this end, we propose to enrich the VGE with semantic information that is associated with the geographic features. Since we deal with large scale geographic environments, the VGE must be organised in a way that reduces the search space for path planning. Hierarchical search is acknowledged as an effective approach to reduce the complexity of such a prob-

SIMUTools 2010 March 15 –19, Torremolinos, Malaga, Spain.
Copyright 2010 ICST ISBN 78-963-9799-87-5 .

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2010.8811
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2010.8811

lem. A number of challenges arise when creating such a semantically-enriched and geometrically-accurate hierarchical VGE, among which we mention: 1) to automatically create an accurate geometric representation of a 3D VGE; 2) to integrate several types of semantic information into the geometric representation; and 3) to structure the environment representation as a hierarchy and use it in *situated reasoning* algorithms (such as obstacle detection and path finding) which are required for MAGS.

In this paper, we present our approach that addresses these challenges toward the creation of such a semantically-enriched and geometrically-accurate hierarchical VGE, which we call an *Informed VGE* (IVGE). This approach extends the IVGE generation model that we presented in [21] by geometrically abstracting the IVGE description for the purpose of the qualification of terrain elevation. Figure 1 presents an overview of our IVGE extended approach which aims at producing an exact representation of the geographic environment based on realistic data provided by a *Geographic Information System* (GIS), and which uses the *Constrained Delauney Triangulation* (CDT) technique for an accurate spatial decomposition. This representation is organised as a topological graph enhanced with data integrating both quantitative information (like the geometry) and qualitative information (like the types of areas such as roads and buildings). In addition, the topological graph is abstracted in order to reduce its size and to enable fast hierarchical path planning processes. Moreover, we propose a Hierarchical Path Planning (HPP) algorithm which relies on this topologic graph structure and which takes advantage of the semantically-enhanced description of the VGE in order to provide autonomous situated agents with optimized semantically-constrained paths. Finally, the complexity analysis of our hierarchical path planning algorithm demonstrates its capacity to optimise path finding in large-scale and complex geographic environments.

The remainder of this paper starts with a discussion of related works on virtual environments and path planners. In Section 3, we provide an overview of geographic environment representation using data provided by Geographic Information Systems (GIS). Section 4 presents our approach to automatically create an Informed VGE. Section 5 outlines a method to enhance the IVGE description using a topologic abstraction that reduces the size of the topologic graph and enables building a hierarchical topologic graph; Section 6 presents how we leverage the hierarchical graph structure of the IVGE model in order to support situated reasoning algorithms such as hierarchical path planning. Section 7 highlights some results obtained by applying our approach to a urban geographic environment in order to address the issue of path planning with respect to both the environment's and the agents' characteristics. Finally, we conclude and present future works.

2. RELATED WORK

Virtual environments and spatial representation have been used in several application domains. For example, Thalmann *et al.* [6] proposed a virtual scene for virtual humans representing a part of a city for graphic animation purposes. Donikian *et al.* [15] proposed a modelling system which is able to produce a multi-level data-base of virtual urban environments devoted to driving simulations. In this work, authors proposed a rich virtual environment representing the shopping mall including layout and positions of the stores, kiosks, doors, as well as the atmosphere such as music, lighting, odor, temperature, etc. More recently, Shao *et al.* proposed a virtual environment representing the New York City's Pennsylvania Train Station populated by autonomous virtual pedestrians in order to simulate the movement of people [22]. Paris [20] also proposed a virtual environment representing a train station populated by autonomous virtual passengers, in order to characterise the levels of services inside exchange areas. However, the focus of these approaches is computer animation and virtual reality, so the virtual environment usually plays the role of a simple background scene in which agents mainly deal with its geometric characteristics. Indeed, the description of the virtual environment is often limited to the geometric level, though it should also contain topological and semantic information for other types of applications. Therefore, most interactions between agents and the environment are most of the time simple, permitting only to plan a path in a 2D or 3D world with respect to free space and obstacle regions [5].

Figure 1: Global architecture for IVGE generation; Green: GIS data processing; Red: the topologic graph abstraction and hierarchical path planning processes.

The path planning issue, which consists of finding an obstacle free path between two distinct positions located in a VGE, has been extensively studied. An excellent survey of this topic is available in [12]. The computational effort required to find a path, using a search algorithm such as A* [18] or Dijkstra [13], increases with size of the search space [2]. As a consequence, path planning on large scale geographic environments can result in serious performance bottlenecks. However, representing the virtual environment using the hierarchical approach allows a reduction in the size of the search space as well as the problem complexity in path planning [8]. Two recent hierarchical triangulation-based path planning approaches are described in [4], namely *Triangulation A** and *Triangulation Reduction A**, which are relevant to our work. *TA** makes use of the *Delaunay Triangulation* (DT) technique to build a polygonal representation of the environment without considering the semantic information. This results in an undirected graph connected by constrained and unconstrained edges, the former being traversable and the latter not. *TRA** is an extension of *TA** and abstracts the triangle mesh into a structure resembling a roadmap. Like our method, both TA* and TRA* are able to accurately answer path queries for agents since they make use of the DT technique. However, the abstraction technique used by TA* and TRA* is notably different from our work. They aim to maximise triangle size whereas we aim to topologically abstract the IVGE by merging triangles into convex polygons. We also handle semantically enriched environment descriptions including qualification of space and terrain characteristics while both *TA** and *TRA** assume a homogeneous flat environment. Logan and colleagues pro-

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2010.8811
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2010.8811

posed a bounded A* algorithm based on a 2D raster-based representation of the virtual environment [14]. However, in contrast with Logan's approach, our method relies on a 3D exact space representation of virtual geographic environments informed with semantics. Moreover, Jagadeesh and colleagues *et al.* proposed a hierarchical path planning algorithm specific to road networks [9]. This algorithm can hardly be used to support the navigation of autonomous situated agents of various extents such as pedestrians, bikes, boats in virtual geographic environments.

The lack of efficient planners which are able to deal with large scale geographic environments while taking into account the geometric, topologic, and semantic characteristics of the space motivated us to propose a novel approach based on the concept of *Informed VGE* (IVGE).

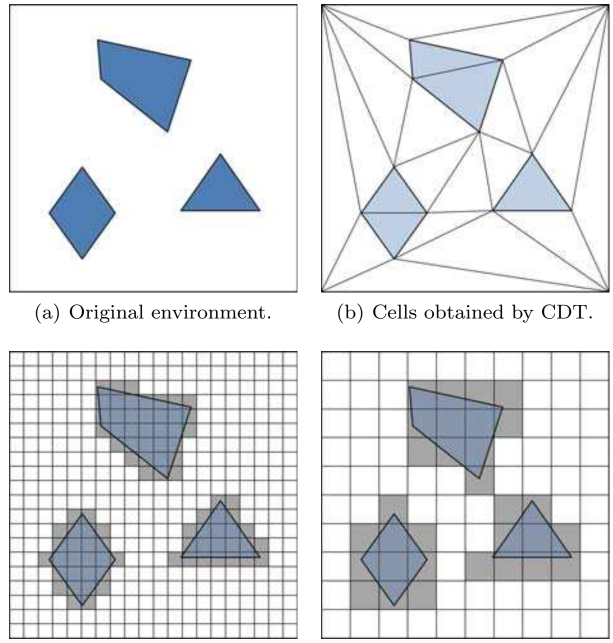
3. SPATIAL REPRESENTATION AND GIS DATA

GIS data are mainly represented in two forms [19]: raster and vector formats. The *raster* format subdivides semantic information into regular *squares* or *square regions* representing discrete, contiguous land areas. This approach generally presents averaged quantitative data, whose precision depends on the subdivision size. The *vector* format exactly locates semantic information with arbitrary complex geometric shapes. This approach generally presents one qualitative object per defined shape.

The VGE exploitation [7] of these data is generally done in two ways. First, the *grid* method [16] is the direct mapping of the raster format, and can also be applied to the vector format (Figure 2(c)). The advantage of this discrete method is that multiple semantic data layers are easily merged in the same geometric representation [23]: the locations where data can be stored are predefined by the grid cells. The main drawback of this method is the problem of localisation accuracy [1], which makes it difficult to position information that is not aligned with the subdivision. Another disadvantage of the grid approach is that its memory complexity depends on the chosen cell resolution, which makes it difficult to represent large environments with fine precision. This method is mainly used for animation [22] or large crowd simulation [17] because of the fast data access it provides.

Second, the *exact geometric subdivision* method consists of subdividing the environment in convex cells defined by the original vector format. The convex cells can be obtained by several algorithms, among which the most popular is the *Constrained Delaunay Triangulation* (CDT) [10]. The CDT produces triangles while keeping the original geometric shapes whose boundaries are named *constraints* (Figure 2(b)). The first advantage of the exact subdivision is that it preserves the input geometry, allowing accurate visualisation of the environment at different scales. Another advantage is that the memory complexity of this approach only depends on the number of shapes, not on the environment's extent and subdivision as is the case for grids. The main drawback of this approach is the difficulty of merging multiple semantic data for partially overlapping shapes. This method tends to be used for crowd microscopic simulation [11] where the motion accuracy is fundamental.

Both VGE representations can be enhanced by an abstraction process [21]. The first goal of an abstraction is to improve the performance of the algorithms based on the en-



(a) Original environment. (b) Cells obtained by CDT.

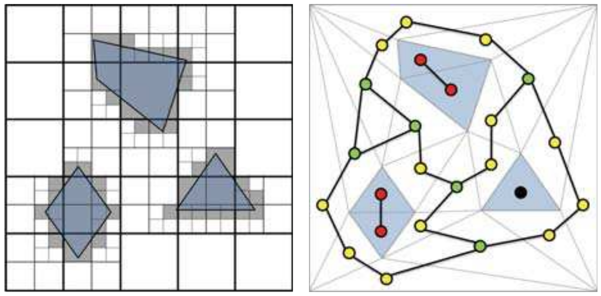
(c) Cell decomposition by uniform grids with two resolutions.

Figure 2: The two common cell decomposition techniques used to represent environments.

vironment description, such as path planning, by reducing the number of elements used to describe the environment. The usual abstraction model for grids is mainly geometric (Figure 3(a)): the quadtree groups four boxes of the same kind to create a higher-level cell [22]. When considering the exact decomposition, an abstraction is generally based on topological properties rather than on purely geometric ones. Indeed, the exact cell subdivision generates connected triangles which can be manipulated as the nodes of a topological graph. This graph can then be abstracted by grouping the nodes, producing a new graph with fewer nodes [21]. For example, Figure 3(b) shows an abstraction which is only based on the nodes' number of connections c: isolated ($c = 0$), dead-end ($c = 1$), corridor ($c = 2$), and crossroad ($c \geq 3$). A topological graph can be used for spatial reasoning, like path planning, thanks to traversal algorithms. These algorithms benefit of the abstraction by traversing first the more abstracted graph, and then by refining the computation in the sub-graphs until reaching the graph of the finest spatial subdivision. This exploitation creates a new need for an abstracted graph which is less prevalent in literature: the minimal information necessary to make a decision must be available at all levels. For example, if the width of a path is relevant for a path planning algorithm, this information must be accessible in all the abstracted graphs; if not, the evaluation would be greatly distorted compared to a non-abstracted graph.

Two kinds of information can be stored in the description of an IVGE. Quantitative data are stored as numerical values which are generally used to depict geometric properties (like a path's width of *2* meters) or statistical values (like a density of *2.5* persons per square meter). Qualitative data

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2010.8811
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2010.8811



(a) Grid abstraction using a three levels quatree. (b) topological graph representation of a CDT.

Figure 3: Abstraction examples for two kinds of environment descriptions.

are introduced as identifiers which can be a reference to an external database or a word with arbitrary semantics, called a label. Such labels can be used to qualify an area (like a *road* or a *building*) or to interpret a quantitative value (like a *narrow* passage or a *crowded* place). An advantage of interpreting quantitative data is to reduce a potentially infinite set of inputs to a discrete set of values, which is particularly useful to condense information in successive abstraction levels to be used for reasoning purposes.

The resulting topological graph can be improved in two ways. First, qualitative information from the arcs of the graph are propagated to the nodes, which allows, for example, deduction of the internal parts of the buildings or of the roads in addition to their outline. Second, we propose a novel approach of information extrapolation using a one-time spatial reasoning process based on a geometric abstraction. This second technique can be used to fix input elevation errors, as well as to create new qualitative data relative to elevation variations. These data are stored as additional semantics bound to the graph nodes, which can subsequently be used for spatial reasoning.

4. COMPUTATION OF IVGE DATA

We propose an automated approach to compute the IVGE data directly from vector GIS data. This approach is based on five stages which are briefly described in this section (Figure 4): *input data selection*, *spatial decomposition*, *maps unification*, *geometric abstraction*, and finally the *informed graph generation*. In this section we briefly introduce these stages since a more detailed description of our IVGE model is provided in [21].

4.1 Input data selection

The first step of our approach is the only one requiring human intervention. It consists of selecting the different vector data sets which are used to build the IVGE. The only restriction concerning these data sets is that they must respect the same scale. The input data can be organised into two categories. First, *elevation layers* contain geographical marks indicating absolute terrain elevations. As we consider 2.5D IVGE, a given coordinate cannot have two different elevations, making it impossible to represent tunnels for example. This said multiple elevation layers can be specified, and the model can merge them automatically. Second, *semantic layers* are used to qualify various types of data in space. Each layer indicates the physical or virtual limits of a given set of features with identical semantics in the geographic environment, such as roads or buildings. The spatial extents can overlap between two layers, and the model is able to merge the information.

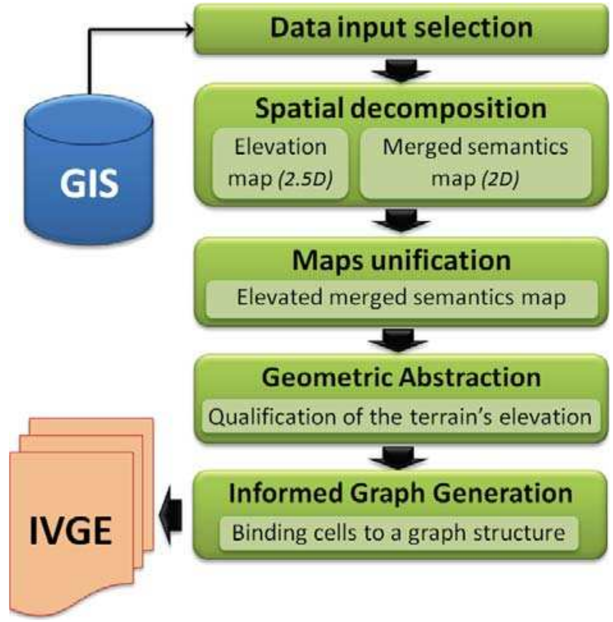


Figure 4: The five stages to obtain an IVGE from GIS data. All the stages are automatic but the first.

4.2 Spatial decomposition

The second step consists of obtaining an exact spatial decomposition of the input data in cells. This process is entirely automatic, using Delaunay triangulation, and can be divided into two parts in relation to the previous phase. First, an elevation map is computed, corresponding to the triangulation of the elevation layers. All the elevation points of the layers are injected in a 2D triangulation, the elevation being considered as an attribute. This process produces an environment subdivision composed of connected triangles (Figure 5(a)). Such a subdivision provides information about coplanar areas: the elevation of any point inside a triangle can be deduced using the elevation of the three measured verticies. Second, a merged semantics map is computed, corresponding to a constrained triangulation of the semantic layers. Indeed, each segment of a semantic layer is injected as a constraint which keeps track of the original semantic data using additional attributes. The obtained map is then a constrained triangulation merging all input semantics (Figure 5(b)): each constraint represents as many semantics as the number of input layers used to build it.

4.3 Maps unification

The third step to obtain our IVGE data consists of unifying the two maps obtained in sub-section 4.2. This phase can be depicted as the mapping of the 2D merged semantic map (Figure 5(b)) on the 2.5D elevation map (Figure

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2010.8811
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2010.8811

(a) Triangulated elevation map (2.5D). (b) Merged semantics map (2D). (c) Unified map (2.5D).

Figure 5: The two processed maps (a, b) and the unified map (c).

5(a)) in order to obtain the final 2.5D merged semantics map (Figure 5(c)). First, preprocessing is carried out on the merged semantics map in order to preserve the elevation precision inside the unified map. Indeed, all the points of the elevation map are injected in the merged semantics triangulation, creating new triangles. Then, a second process elevates the merged semantics map. The elevation of each merged semantics point P is computed by retrieving the corresponding triangle T inside the elevation map, i.e. the triangle whose 2D projection contains the coordinates of P. Once T is obtained, the elevation is simply computed by projecting P on the plane defined by T using the Z axis. When P is outside the convex hull of the elevation map, then no triangle can be found and the elevation cannot be directly deduced. In this case, we use the average elevation of the points of the convex hull which are visible from P.

4.4 Geometric abstraction

Geographic environments are seldom flat. Therefore, it is important to consider the terrain shape elevation when describing a geographic environment. Quantitative elevation data are stored in the GIS which is suitable for calculations. However, spatial reasoning such as path planning often needs to manipulate qualitative information. For example, when considering a slope, it is obviously simpler and faster to qualify it using an attribute such as *gentle* and *steep* slope rather than using numerical values. Hence, it is easy to decide that *gentle slopes* are *crossable* and *steep slopes* are not. However, when dealing with large scale geographic environments, qualifying the terrain's shape, including its light variations, may be a complex task. To this end, we propose a geometric abstraction process that uses geometric data to group cells and to extract the terrain's elevation information from spatial areas. The geometric abstraction relies on a coplanarity criterion which is assessed by computing the difference between the *normal vectors* of two neighbouring cells or groups of cells. In order to compute the normal vector of a group, we adopt the *area-weight normal vector* [3] which takes into account the unit normal vectors of its composing cells as well as their respective surfaces.

4.5 Informed graph generation

The unified map now contains all the semantic information of the input layers, along with the elevation information. This map can be used as a topological graph, where each node corresponds to the map's triangles, and each arc to the adjacency relations between these triangles. Then, common graph algorithms, especially graph traversal, can be applied to this topological graph. One of these algorithms retrieves the node, and so the triangle, corresponding to given 2D coordinates. Once this node is obtained, it is possible to extract the data corresponding to the position, such as the elevation, using the 2.5D triangle and the semantics information. Many other algorithms can be applied, such as path planning or graph abstraction, but they are out of the scope of this paper and will not be detailed here.

5. TOPOLOGIC ABSTRACTION

In Section 4, we presented our work on the generation of informed virtual geographic environments using an exact spatial decomposition scheme which subdivides the environment into convex cells organised in a topologic graph structure. However, inside large scale and complex geographic environments (such as a city for example), such topologic graphs can become very large. The size of such a topologic graph has a direct impact on paths' computation time. In order to optimise the performance of path computation, we need to reduce the size of the topologic graph representing the IVGE. The aim of the topologic abstraction is to provide a compact representation of the topologic graph suitable for situated reasoning and enabling fast path planning. However, in contrast to the geometric abstraction which only enhances the description of the IVGE with elevation semantics, the topologic abstraction extends the topologic graph with new layers. In each layer (except for the initial layer which is called level 0), a node corresponds to a group of nodes of the immediate lower level (Figure 6). Indeed, the topologic abstraction simplifies the IVGE description by combining cells (triangles) in order to obtain convex groups of cells. Such a hierarchical structure evolves the concept of *Hierarchical Topologic Graph* in which cells are fused in groups and edges are abstracted in boundaries. To do so, convex hulls are computed for every node of the topologic graph. Then, the coverage ratio of the convex hull is evaluated as the surface of the hull divided by the actual surface of the node. The topologic abstraction finally groups a set of connected nodes if and only if the group ratio is close to one. Let G

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2010.8811
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2010.8811

be a group of cells, C be the convexity rate, and $CH(G)$ be the convex hull of the polygon corresponding to G. C is computed as follows:

$$C(G) = \frac{Surface(G)}{Surface(CH(G))} \quad and \quad 0 < C(G) \leq 1 \qquad (1)$$

Indeed, the convex property of groups needs to be preserved after the topologic abstraction. This ensures that an entity can move freely inside a given cell (or group of cells), and that there exists a straight path linking edges belonging to the same cell (or group of cells). Figure 7 illustrates an example of the topologic abstraction process and the way it reduces the number of cells representing the environment. Figure 7(a) depicts the initial exact spatial decomposition of a complex building which yields 63 triangular cells. Figure 7(b) presents 28 convex polygons generated by the topologic abstraction algorithm. The optimisation rate of the number of cells representing the environment is around 55%.

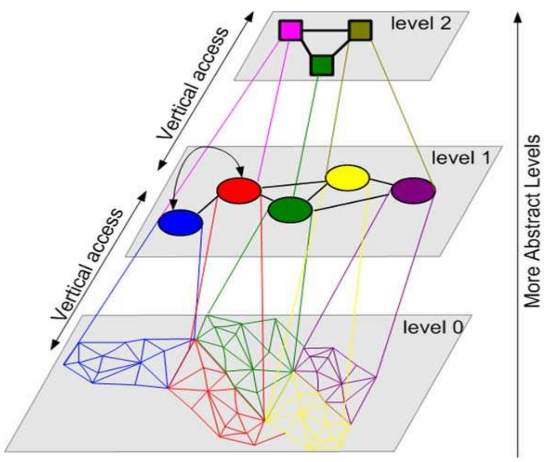
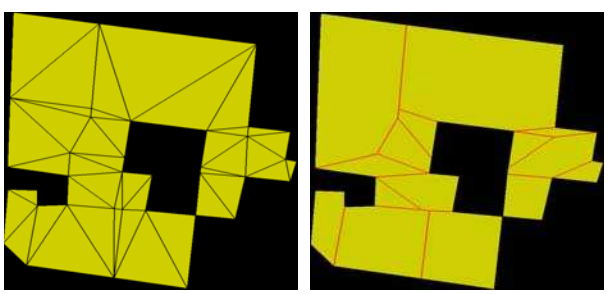


Figure 6: The topologic graph extraction from space decomposition and extension into different levels using the topologic abstraction.



(a) (b)

Figure 7: Illustration of the topologic abstraction process with a strict convex property (*C(gr) = 1*); (a) the exact space decomposition using CDT techniques (63 triangular cells) ; (b) the topologic abstraction (28 convex polygons)

6. HIERARCHICAL PATH PLANNING

In this section, we present our hierarchical path planning algorithm (HPP for short). We then provide a computation analysis of the algorithm complexity which aims to point out the contribution of our algorithm. Finally, we propose a path enhancement method in order to optimise the computed paths for more realistic moving agents.

6.1 Algorithm

Let us consider the topologic graph extracted from the exact spatial decomposition before highlighting the usefulness of the topologic and semantic abstractions. Since cells are convex, it is possible to build an obstacle-free path by linearly connecting positions located at two different borders belonging to a given cell. Thus, it is also possible to use borders, represented by edges in the graph, to compute obstacle-free paths between different locations in the environment. Since the topologic graph structure is hierarchical, each node at a given level i (except at level 0) represents a group of convex cells or abstract cells of a lower level $i-1$. Hence, our approach can be used to compute a path linking two abstract nodes at any level.

Let us consider a hierarchical topologic graph G composed of i levels. Nodes belonging to level 0 are called *leaves* and represent convex cells produced by the exact spatial decomposition. Nodes belonging to higher levels ($i > 0$) are called *abstract nodes* and are composed of groups. Given a starting position, a final destination, and a hierarchical topologic graph G composed of i levels, the objective of our algorithm is to plan a path from the current position to the destination using G. The algorithm starts from the highest level of the hierarchy and proceeds as follows:

- **Step 1:** Identify the abstract nodes to which the starting position and the final destination belong.
  Two cases need to be considered:
  - Case 1: Both are in the same abstract node k at level i.
    Proceed to *step 1* with the groups (at level $i-1$) belonging to node k.
  - Case 2: They are in different abstract nodes k and j at level i. Proceed to *step 2*.
- **Step 2:** Compute the path from the abstract node k to the abstract node j.
  For each pair of consecutive nodes (s, t) belonging to this path, two cases are possible :
  - Case 1: Both are leaves. Proceed to *step 4*.
  - Case 2: Both are abstract nodes. Proceed to *step 3*.
- **Step 3:**
  - If the starting position belongs to s then identify to which group gs of s it belongs and proceed to *step 2*, in order to compute the path from the abstract node gs to the closet common boundary with the abstract node t. Else proceed to *step 2* in order to compute the path from the center of the abstract node s to the closet common boundary with the abstract node t.

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2010.8811
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2010.8811

– If the final destination position belongs to t then identify to which group gd of t it belongs and proceed to *step 2*, in order to compute the path from the closet common boundary with the abstract node s to gd. Else proceed to *step 2* in order to compute the path from the closet common boundary with the abstract node s to the centre of the abstract node t.

- **Step 4:** Once in a leaf, apply a path planner algorithm (we used the Djkstra and A* algorithms) from the starting position to the final goal using the convex cells which belong to the informed graph.

The strategy adopted in this algorithm is to refine the path planning when getting closer to the destination. The algorithm starts by planning a global path between the start and the destination abstract nodes (step 1). Then, for each pair of successive abstract nodes, it recursively plans paths between groups (of lower levels) until reaching leaves (steps 2 and 3). Once at leaves (convex cells at level 0), the algorithm proceeds by applying a path planning algorithm such as Dijkstra and A* (step 4). Hence, at level i, the path planner exploration is constrained by the nodes belonging to the path computed at level $i+1$.

Moving agents can use this algorithm in order to plan paths within the IVGE. The path computed in step 2 is actually a coarse-grained path whose direction is only indicative. Since the path is refined in a *depth-first* way, agents can perform a local and accurate navigation inside an abstract node without requiring a complete and fine-grained path computation towards the final destination. The lower levels' sub-paths (related to other abstract nodes) are computed only when needed, as the agent moves. Such a *just in time* path planning approach is particularly relevant when dealing with dynamic environments. Classic path planning approaches use the entire set of cells representing the environment and compute the complete path between a start and a final positions. These classical approaches suffer from two major drawbacks : 1) the computation time of a path is considerable since it involves all the cells composing the environment; 2) the planned path may become invalid as a consequence of changes in the environment. An interesting property of our hierarchical path planning approach is the optimization of calculation costs over time. Indeed, the entire path is only computed for the most abstracted graph, which contains a small number of abstract nodes compared to the informed graph (convex cells at level 0). In addition, our approach provides a *just in time* path planning which can accommodate a dynamic environment. Furthermore, this hierarchical path planning is adapted to any type of agents, whenever we are able to generate the abstracted graphs taking into account both the geographic environment and the agents' characteristics.

6.2 Complexity analysis

In order to highlight the outcomes of our approach, let us compare the computation cost of our hierarchical path planning with the standard path planning. Let $G_0(V_0, E_0)$ be the graph representing the virtual environment at level 0, which corresponds to cells produced by the spatial decomposition process. Let V_0 correspond to the set of vertices and E_0 correspond to the set of edges at level 0. Let $|V_0| = N$ be the number of nodes of the graph G_0. Let us consider a starting position s and a destination position d located in the virtual environment. The computation cost of the shortest path between s and d at level 0 (represented by the graph G_0) is denoted by $C_0(N)$ and is given by the following equation:

$$C_0(N) = O(N * ln(N)) \quad (2)$$

Let us now compare $C_0(N)$ with the computation cost of our hierarchical path planning algorithm which relies on the hierarchical topologic graph with k levels. To this end, we need to raise some assumptions for the sake of simplification. First, let us assume that the topologic abstraction process may be thought of as a function h which abstracts a topologic graph G_{i-1} and builds a new topologic graph G_i. The function h can be written as follows:

$$h(G_{i-1}(V_{i-1}, E_{i-1})) = G_i(V_i, E_i) \quad with \;\; 0 \le i \le k-1 \quad (3)$$

Let l_i be the *abstraction rate* between two successive levels $i-1$ and i (with $0 \le i \le k-1$). Since the abstraction process aims to reduce the number of nodes at each new level, we have $l_i > 1 + \epsilon$ (with $0 \le i \le k-1$) as illustrated in equation 4.

$$l_i = \frac{|V_{i-1}|}{|V_i|} \quad with \;\; l_i > 1 + \epsilon \; and \; \epsilon > 0 \quad (4)$$

Second, let us suppose that the k^{th} level of our hierarchical topologic graph is composed of m nodes. N which corresponds to the number of nodes of the graph G_0 can be expressed using equations 3 and 4 as follows:

$$N = m * l_{k-1} * ... * l_0 \quad (5)$$

$$N \ge m * (1+\epsilon)^k \quad with \;\; k > 0 \; and \;\; \epsilon > 0 \quad (6)$$

$$N = m * \prod_{i=0}^{k-1}(l_i) \quad with \;\; k > 0 \; and \;\; m > 0 \quad (7)$$

Let l_{Avg} be the average value of l_i (with $0 \le i \le k-1$). Using l_{Avg}, equation 7 becomes:

$$N = m * l_{avg}^k \quad with \;\; k > 0 \; and \;\; m > 0 \quad (8)$$

Let us replace the term N in equation 2 by its value in equation 8:

$$C_0(m) = O(m * l_{avg}^k * ln(m * l_{avg}^k)) \quad (9)$$

Equation 9 can be developed as follows:

$$C_0(m) = O(m * ln(m) * l_{avg}^k + m * l_{avg}^k * ln(l_{avg}^k)) \quad (10)$$

Let Nb_k be the number of nodes composing the computed path at level k. The computation cost of Nb_k is given by the following equation:

$$Nb_k = O(m * ln(m)) \;\; with \;\; k > 0 \; and \;\; m > 0 \quad (11)$$

The hierarchical path planning algorithm involves the computation of the shortest path at level k and the refinement of the path linking each pair of successive nodes at lower levels. Therefore, the shortest path from s to d corresponds to the computation of Nb_k at level k and its refinement through the lowest levels. Such a shortest path is denoted C_k and has a computation cost which can be computed by the following

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2010.8811
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2010.8811

equations:

$$C_k(m) = Nb_k * \sum_{j=0}^{k-1} l_{avg}^j \quad (12)$$

$$C_k(m) = Nb_k * \frac{l_{avg}^k - 1}{l_{avg} - 1} \quad (13)$$

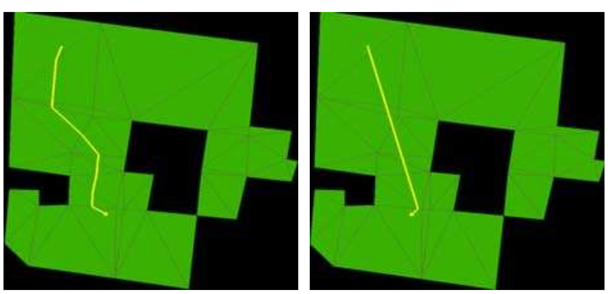
The term Nb_k in equation 13 is replaced by its value expressed in the equation 11 as follows:

$$C_k(m) = O(m * ln(m) + m * \frac{l_{avg}^k - 1}{l_{avg} - 1}) \quad (14)$$

Let us compare the computation costs of standard path planning approaches (equation 10) and our hierarchical path planning approach (equation 14). First, it is obvious that the first term $m * ln(m)$ in equation 10 is inferior to the first term $m * ln(m) * l_{avg}^k$ in equation 14 since the abstraction rate $l_{avg}^k > 1$. Second, in a similar way, the second term $m * (l_{avg}^k - 1/l_{avg} - 1)$ in equation 10 is inferior to the second term $m * l_{avg}^k * ln(l_{avg}^k)$ in equation 14. In conclusion, the hierarchical path planning algorithm along with the hierarchical topologic graph that we propose is at least $ln(l_{avg}^k)$ orders of magnitude faster than standard path planning approaches.

6.3 Path Optimisation

The topological abstraction only groups together adjacent cells or groups of cells with respect to the convexity criterion. While this approach is efficient to reduce the size of the topologic graph, it gives up the optimality of the computed path. Indeed, paths are optimal in the abstract graph but not necessarily in the initial problem graph (informed graph at level 0). In order to improve the quality of the computed path (i.e., length and visual optimisation), we perform a post-processing phase called *path optinmisation* (Figure 8). Our strategy for path optimisation is simple, but produces good results. The main idea is to replace local sub-optimal parts of the computed paths by straight lines. We start from one end of the path (Figure 8(a)). For each node part of the computed path, we check whether we can reach a subsequent node in the path in a straight line. If this is possible, then the linear path between the two nodes replaces the initial sub-optimal sequence between these nodes (Figure 8(b)).



(a) (b)

Figure 8: (a) The original computed path ; (b) The computed path after optimisation.

7. RESULTS

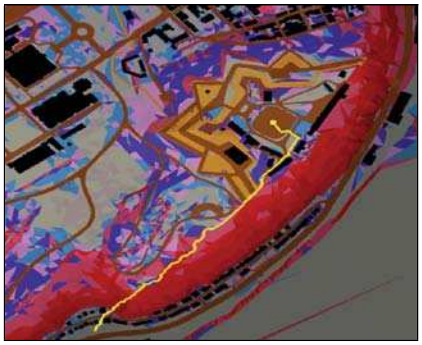
In this section, we present the results of the implementation of our IVGE generation approach. We also show how the multi-level graph structure and the semantically-enriched description of the IVGE can be used to support agents' path planning, taking into account both the environment's characteristics and the situated agent archetypes. Our IVGE generation model is efficient and can process an area such as the center part of Quebec City, with one elevation map and five semantic layers, in less than five seconds on a typical computer (Intel Core 2 Duo processor 2.13Ghz, 1G RAM). The resulting unified map approximately contains $122,000$ triangles covering an area of $30km^2$. The necessary time to retrieve the triangle corresponding to a given coordinate is negligible (less than 10^{-4} seconds). The geometric abstraction produces approximately $73,000$ groups of cells in 2.8 seconds.

Figure 9 illustrates two paths linking two locations situated in the IVGE. Figure 9(a) shows a path planning (coloured in yellow) which avoids obstacles such as *buildings*, *walls* which are coloured in black, but does not take into account the terrain characteristics. Therefore, this path crosses an area coloured in red which represents a steep slope. In Figure 9(b), the algorithm has generated a path which respects both the terrain and the obstacles in the IVGE. Indeed, the steep slopes (initially coloured in red) are avoided since they are now considered as obstacles (coloured in black). This path is longer, but it fully respects the constraints of the environment and the slope of the terrain.

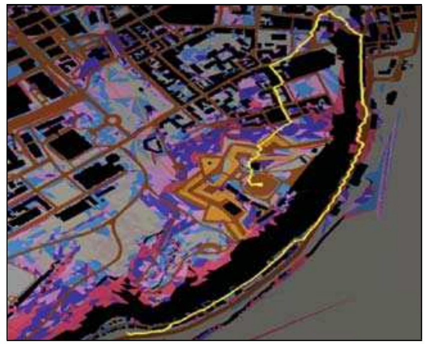
In a multi-agent geo-simulation an agent may not aim at reaching a particular position but rather a particular area in the IVGE. This type of path planning answers the question: *how to find a path to reach a specific area while respecting the environment's constraints?* The semantic information (building, house, marina, wall of the old city, etc.) integrated in the IVGE description helps answer such a question. Thanks to the topological graph and using the *Dijkstra* algorithm the system computes the shortest path to reach a specific area located in the IVGE. If the semantics associated with the visited node correspond to the target area's characteristics, then the algorithm stops and the path is generated. To illustrate path planning towards a target area qualified by one or several semantics (instead of a target position), we propose the following example: a tourist who moves using a wheelchair is located inside the old city of Quebec. This tourist wants to visit an attraction spot called the *marina*. Here, the *marina* is not identified by coordinates (x, y, z), but rather by semantic information. A path that only avoids the obstacles of the environment (buildings coloured in black) but crosses steep slopes areas (coloured in red) is obviously not acceptable for this tourist. Figure 9(c) shows the computed path to reach the marina (the marina is coloured in blue at the top of the figure). This path avoids steep slopes (coloured in black) as well as obstacles situated in the IVGE (buildings coloured in black) and reaches a place identified by the semantic information (marina).

We have formally demonstrated how our hierarchical path planning algorithm allows for the enhancement of the path planning computation cost in our IVGE. Indeed, our hierarchical path planning algorithm, along with our IVGE's hierarchical structure, is faster than standard path planning approaches.

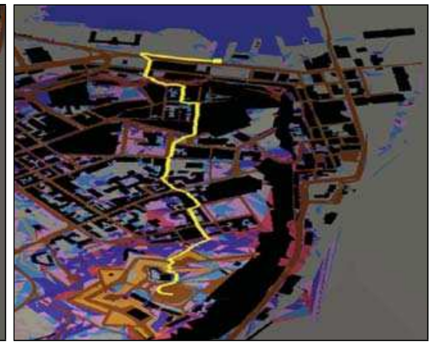
Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2010.8811
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2010.8811



(a)



(b)



(c)

Figure 9: Path planning in the IVGE (the computed path is coloured in yellow). (a) path computed with no regard for the terrain shape; (b) path computed with regard for the terrain shape; (c) Search path to get to a place (marina) in the IVGE (place described by semantics).

Finally, in order to highlight the outcomes of the path optimisation process, we randomly selected 19 starting and destination positions in the IVGE. For each pair of positions, we compared the original computed path length with the optimised path length. Figure 10 depicts the comparison of the non optimised computed path length and the optimised path length. It shows how the optimisation process reduces the computed path length by an average of 16%.

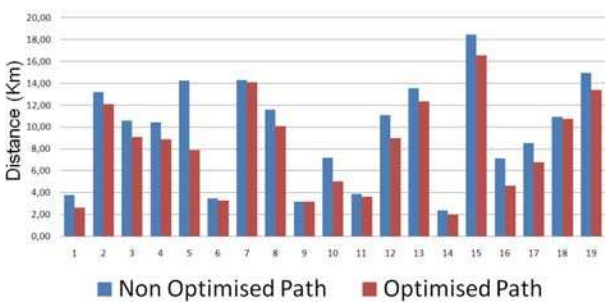


Figure 10: Optimised versus non-optimised paths lengths.

8. DISCUSSION

In order to reduce the search space, we proposed a hierarchical topologic graph that groups convex cells and groups of cells. Hence, each abstract node at level i contains a subset of this graph at level $i-1$, composed by at least one node or abstract node. The extraction of this hierarchical topologic graph only requires an acceptable one-time computation cost and a low memory overhead. Despite the reduction of the number of nodes, this technique creates two application-dependent issues that must be addressed: ***hierarchical traversal cost*** and ***Information richness***.

First, the *hierarchical traversal cost* increases with each grouping, which might limit the performance of the search space reduction brought by the hierarchical representation. Indeed, despite of the number of levels of the hierarchal topologic graph, the path planning process provides moving agents with a set of convex cells (belonging to level 0) to pass through in order to reach the final destination. This means that the path planning process must inevitably traverse the hierarchical topologic graph from its top to its bottom in order to compute such a set of cells.

Second, the *information richness* decreases with each grouping level, which could lead to useless additional abstraction levels that may not improve the decision making of the hierarchical path planning algorithm. Indeed, the more potential sub-paths an abstract node contains, the less its choice influences the path planning process. Therefore, the determination of the number of topologic abstraction levels must be carefully analysed with respect to these two critical issues in addition to the application requirements.

Another important aspect of our IVGE is its capability to represent geographic environments which are distributed in space. Indeed, thanks to the hierarchical structure of the topologic graph, our model is capable of representing portions of geographic environments which are not adjacent in space. For example, consider the problem of traveling by car from Quebec city (QC, Canada) to New York (NY, USA). We need to compute the shortest (minimum distance) path from a given address in Quebec city, let us say 312 *Marie-Louise*, to a given address in New York city, let us say 1213 4*th* Avenue, *Brooklyn*. Given a detailed description of the geographic environment showing all roads annotated with driving distances, a classic planner can compute such a travel route. However, this might be an expensive computation, given the large size of the description of the geographic environment. This problem may be solved in a three steps process. First, we compute the path from 312 *Marie-Louise* to a major highway leading out of Quebec city. Second, we compute the path from Quebec to the boundaries of New York. Third, we compute the path from the incoming highway to 1213 4*th* Avenue, *Brooklyn*. Assuming that the second path is mostly composed of highways and can be quantified (distance and travel time), it is easy to model this path using a *conceptual node* in our hierarchical topologic graph. A conceptual node allows for linking spatially distributed geographic environments and hence allows us to accurately compute optimal paths with respect to these environments' characteristics.

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2010.8811
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2010.8811

9. CONCLUSION AND FUTURE WORKS

In this paper, we proposed an accurate and automated approach for the generation of semantically-enhanced and geometrically-accurate virtual geographic environments using GIS data. This novel approach offers several advantages. First, the description of the IVGE is realistic since it is based on standard GIS data. This description is also quite accurate because it is produced by an exact spatial decomposition technique which uses data in a vector format. Hence, this description preserves both the geometric and the topological characteristics of the geographic environment and enables a graph-based description of the virtual environment. The topologic approach goes beyond grid-based techniques by combining semantic information merging with the accuracy of vector-based representations. The main outcome of such a semantically-enhanced and geometrically-accurate virtual geographic environment concerns agents' situated reasoning capabilities such as path planning in large-scale and complex geographic environments. We proposed a hierarchical path planning algorithm (using *Dijkstra* and *A**) which takes advantage of our IVGE model to provide paths which take into account the agents' and environment's characteristics.

We are currently working on further improvements of the IVGE's description by integrating enriched knowledge representations (called *the environment knowledge*) using *Conceptual Graphs* aimed at assisting situated agents' interactions with the IVGE and helping them achieve their goals. The goal of the environment knowledge integration is to extend the agents' knowledge about their surrounding environment. The above-mentioned contributions of our model offer new opportunities for many applications in a variety of application domains including the entertainment industry (games and movies), security planning and crowd management (planning events involving large crowds), and environment monitoring in natural environments using spatially-aware sensor networks.

10. REFERENCES

[1] R. Andersen, J. L. Berrou, and A. Gerodimos. On some limitations of grid-based (ca) pedestrian simulation models. In *First International Workshop on Crowd Simulation (V-Crowds'05)*. VRlab, EPFL, 2005.

[2] A. Botea, M. Müller, and J. Schaeffer. Near optimal hierarchical path-finding. *Journal of Game Development*, 1:7–28, 2004.

[3] S.-G. Chen and J.-Y. Wu. A geometric interpretation of weighted normal vectors and its improvements. *Computer Graphics, Imaging and Vision: New Trends, 2005. International Conference on*, pages 422–425, July 2005.

[4] D. Demyen and M. Buro. Efficient triangulation-based pathfinding. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference (AAAI'06)*, Boston, Massachusetts, USA, July 16-20 2006.

[5] S. Donikian. Vuems: A virtual urban environment modeling system. In *Proceedings of the Conference on Computer Graphics International (CGI'97)*, pages 84–92, Washington, DC, USA, June 23-27 1997.

[6] N. Farenc, R. Boulic, and D. Thalmann. An informed environment dedicated to the simulation of virtual humans in urban context. In P. Brunet and R. Scopigno, editors, *Computer Graphics Forum (Eurographics '99)*, volume 18(3), pages 309–318. The Eurographics Association and Blackwell Publishers, 1999.

[7] J. Gong and L. Hui. Virtual geographical environments: Concept, design, and applications. In *International Symposium on Digital Earth (ISDE)*, pages 369–375, Beijing, China, November 29 - December 2 1999.

[8] D. Harabor and A. Botea. Hierarchical path planning for multi-size agents in heterogenous environments. In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS'08)*, Sydney, Australia, September 14-18 2008.

[9] G. R. Jagadeesh, T. Srikanthan, and K. H. Quek. Heuristic techniques for accelerating hierarchical routing on road networks. *IEEE Transactions on Intelligent Transportation Systems*, 3(4):301–309, 2002.

[10] M. Kallmann, H. Bieri, and D. Thalmann. Fully dynamic constrained delaunay triangulations. *Geometric Modelling for Scientific Visualization*, 2003.

[11] F. Lamarche and S. Donikian. Crowds of virtual humans: a new approach for real time navigation in complex and structured environments. *Computer Graphics Forum, Eurographics'04*, 2004.

[12] S. LaValle. *Planning Algorithms*. Cambridge University Press., Cambridge, 2006.

[13] J. Lengyel, M. Reichert, B. R. Donald, and D. P. Greenberg. Real-time robot motion planning using rasterizing computer graphics hardware. In *SIGGRAPH '90: Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, pages 327–335, New York, NY, USA, 1990. ACM.

[14] B. Logan and N. Alechina. A* with bounded costs. In *AAAI '98/IAAI '98: Proceedings of the fifteenth national/tenth conference on Artificial intelligence/Innovative applications of artificial intelligence*, pages 444–449, Menlo Park, CA, USA, 1998. American Association for Artificial Intelligence.

[15] J.-E. Marvie, J. Perret, and K. Bouatouch. Remote interactive walkthrough of city models. In *Proceedings of the 11th Pacific Conference on Computer Graphics and Applications (PG'03)*, pages 389–393, Oct. 2003.

[16] M. Mekni, N. Sahli, and B. Moulin. A geosimulation approach involving spatially-aware agents a case study on the identification of risky areas for trains. In *Agent-Directed Simulation Symposium (ADS'08), Part of the 2008 Spring Simulation Multiconference (SpringSim'08)*, pages 39–44. ACM/SIGSIM, April 2008.

[17] B. Moulin, W. Chaker, J. Perron, P. Pelletier, H. Jimmy, and E. Gbei. Mags project: Multi-agent geosimulation and crowd simulation. *Lecture notes in computer science*, 2825:151–168, 2003.

[18] J. Nilsson. *Principles of artificial intelligence*. Springer-Verlag, Berlin ; Heidelberg ; New York, third edition, 1982.

[19] Open Source Geospatial Fundation (OSGEO). GDAL-OGR: Geospatial Data Abstraction Library / Simple Features Library Software, Aug. 2008. http://www.gdal.org.

[20] S. Paris. *Characterisation of the levels of services and modeling of the movement of people inside exchange areas*. PhD thesis, Université de Rennes 1, October 2007.

[21] S. Paris, M. Mekni, and B. Moulin. Informed virtual geographic environments: an accurate topological approach. In *The International Conference on Advanced Geographic Information Systems & Web Services (GEOWS)*, pages 1 – 6. IEEE Computer Society Press, 2009.

[22] W. Shao and D. Terzopoulos. Environmental modeling for autonomous virtual pedestrians. *Digital Human Modeling for Design and Engineering Symposium*, 2005.

[23] J. Zhu, J. Gong, H. Lin, W. Li, J. Zhang, and X. Wu. Spatial analysis services in virtual geographic environment based on grid technologies. *MIPPR 2005: Geospatial Information, Data Mining, and Applications*, 6045(1):604–615, 2005.

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2010.8811
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2010.8811