38

Intervisibility on terrains

L DE FLORIANI AND P MAGILLO

Many interesting application problems on terrains involve visibility computations, for example, the search for optimal locations of observation points, line-of-sight communication problems, and the computation of hidden or scenic paths. The solution of such problems requires techniques to answer visibility queries on a terrain efficiently, as well as the development of data structures to encode the visibility of a terrain from one or several viewpoints. In this chapter, visibility structures such as the viewshed (representing the portions of a terrain visible from a single viewpoint) and the horizon (describing the distal boundary of the viewshed) are introduced, as well as visibility structures related to several viewpoints. All these structures admit continuous and discrete encodings (mainly used on triangulated irregular networks and regular square grids, respectively). An overview is provided of algorithms for building visibility structures on triangulated irregular networks and regular square grids as well as for answering visibility queries on a terrain. Finally, some application-specific problems involving visibility computation are illustrated.

1 INTRODUCTION

Many interesting application problems concerning terrains involve visibility computations, for example, the placement of observation points according to suitable constraints, line-of-sight communication problems, the computation of paths with certain visibility properties, etc. Applications include the location of fire towers, radar sites, radio, TV or telephone transmitters, path planning, navigation and orientation, and so on. An exhaustive survey of visibility-related problems is provided by Nagy (1994). The solution of such problems needs methods to answer visibility queries efficiently, through the development of structures for encoding the visibility situation of a terrain. Visibility queries consist of determining whether a given object located on a terrain is visible from a viewpoint and possibly how much of the object is visible. Visibility structures provide information about the visibility of the terrain itself; the knowledge of suitable visibility structures for a terrain helps answer visibility queries.

After recalling some basic notions about digital elevation models (DEMs) in section 2, section 3 defines various kinds of structures encoding visibility information for a terrain. Relevant visibility structures for a single viewpoint are the *viewshed*, which represents the location and extent of the portion of a terrain visible from a given viewpoint, and the *horizon*, which describes the distal boundary of the viewshed. To model the visibility of a terrain with respect to a set of viewpoints, structures based on combinations of the viewsheds of single points are used; depending on the specific problem, it can be useful, for instance, to consider the union or the intersection of viewsheds, or to count how many viewpoints can see each point on a terrain. Visibility structures admit both *continuous* and *discrete* encodings. In general, continuous encodings of the viewshed (called *continuous visibility maps*) are used for triangulated irregular networks (TINs), while discrete encodings (called *discrete visibility maps*) are used on dense regular square grids (RSGs). Visibility information related to multiple viewpoints is commonly represented in a discrete way, because of its size, leading to structures such as *visibility graphs, intervisibility maps* and *visibility counts*; continuous encodings reduce to overlays of continuous visibility maps.

Section 4 provides a survey of algorithms for computing visibility structures (visibility maps, horizons, etc.) on terrain models and for solving visibility queries. RSGs and TINs have very different structures and are usually handled using very different methods. Visibility computation on RSGs is based on line-of-sight processing (Blelloch 1990; Shapira 1990). This is an expensive process because of the size of the grid; parallel algorithms have been defined, which take advantage of the regular spatial structure of an RSG (Mills et al 1992; Teng et al 1993). TINs have deserved strong consideration from both the computational geometry and GIS communities; practical algorithms are reported (Boissonnat and Dobrindt 1992; De Floriani et al 1989; Lee 1991a), as well as algorithms of theoretical interest for their good asymptotic complexity (Edelsbrunner et al 1989; Katz et al 1991; Overmars and Sharir 1992; Preparata and Vitter 1992; Reif and Sen 1988). Finally, *visibility queries* can be efficiently answered based on either visibility maps or on horizons; ad hoc methods have been proposed for solving queries directly, that is, without computation of intermediate data structures (Cole and Sharir 1989).

All visibility computations are sensitive to errors in elevation near the viewpoint, since these are amplified in proportion to the distance. For this reason, various authors suggest that the topography near the viewpoint must be known much more accurately than on the rest of the surface (Cignoni et al 1995; Felleman and Griffin 1990). *Multi-resolution terrain models* play an important role here, since it is possible to obtain from such models terrain representations whose level of resolution in any area of the domain can be specified by the user. Section 5 introduces multi-resolution models of terrains and considers the problem of computing and updating visibility structures on a multi-resolution model, as well as the direct solution of visibility queries on such models.

In section 6 we consider application problems related to visibility, such as viewpoint placement, line-of-sight communication and path problems, and the role of visibility information in solving them. Section 7 contains some concluding remarks.

2 PRELIMINARIES

A *topographic surface* (or *terrain*) can be regarded as the image of a real bivariate function f defined over a domain D in the Euclidean plane. A digital elevation model (DEM) is a model of one such surface built on the basis of a finite set of digital data (see also Band, Chapter 37; Hutchinson and Gallant, Chapter 9). Terrain data consist of elevation measures at a set of points $S \subseteq D$; points in S can either be scattered, or form a regular grid. A DEM built on S represents a surface that interpolates the measured elevations at all points of S. Two classes of DEMs are usually considered in the context of GIS for visibility computation:

- A TIN is defined by a triangulation of the domain D having its vertices at the points of S. Function f is defined piecewise as a linear function over each triangle. Thus, the surface described by a TIN consists of planar patches.
- An RSG is defined by a domain partition into rectangles, induced by a regular grid over D. Functions used on such partitions depend on the degree of continuity desired for the surface; usually, f is either bilinear or constant over each region.

TINs show good capabilities to adapt to terrain features, since they can deal with irregularly-distributed datasets and may include surface-specific points and lines. Often, a Delaunay triangulation is used as a domain subdivision for a TIN, because of its good behaviour in numerical interpolation.
A triangulation is a Delaunay triangulation if and only if the circumcircle of each triangle does not contain any other vertex; the Delaunay triangulation can also be characterised as the dual graph of the Voronoi diagram of a point set (Boots, Chapter 36). More recently, data-dependent triangulations have been proposed, which take into account the z values of points in V instead of simply their x,y coordinates; the idea is either to maximise or to minimise some quantity that expresses certain properties of the resulting surface (e.g. the ‘roughness’ or the ‘thin-plate energy’, or the maximum jump between adjacent patches; see Dyn et al 1990, for a survey, and Mitas and Mitasova, Chapter 34).

3 VISIBILITY STRUCTURES FOR TERRAINS

Measuring visibility requires computing visibility for (portions of) the surface itself or for objects located on the surface (representing, for example, facilities such as towers, buildings, radio transmitters, etc.). The problem of testing the visibility of an object is essentially a query problem, and must be solved on-line. In contrast, visibility information for the surface itself can be precomputed and stored in appropriate visibility structures; such structures also help answer visibility queries. This section introduces the main structures used to represent the visibility of a terrain with respect to a single or multiple viewpoints; for each structure, first an abstract definition is given, then its encodings on RGSs and on TINs are considered.

Two points V and W on a topographic surface are said to be mutually visible if and only if the interior of the straight-line segment joining them lies strictly above the terrain; such a segment is allowed to touch the surface at most at its endpoints V and W. Any point V lying on or above a topographic surface can be chosen as a viewpoint.

The basic visibility structure for a terrain is the viewshed. Given a viewpoint V on a terrain, the viewshed of V is the set of points of the surface which are visible from V, that is, viewshed $(V)=\{W\in D \mid W$ is visible from $V\}$.

Another relevant form of visibility information is the horizon of a viewpoint V, which corresponds to the ‘distal’ boundary of the viewshed. Such reduced information can replace the viewshed in some applications, with the advantage of lower storage costs. The horizon determines, for every radial direction around V, the farthest point on the terrain which is visible from V: horizon $(V) = \{W\in D \mid W$ is visible from V and $\forall Q\in D$ if $W\in \overline{VQ}$ then Q is invisible from $V\}$.

Visibility structures for several viewpoints can be defined by combining the viewshed of such points according to some operator; common combination operators are:

- *Overlay*: the terrain is partitioned into regions, in such a way that each region is visible from a given set of viewpoints.
- *Boolean operators* (see Eastman, Chapter 35), such as the intersection of the viewsheds (which gives the portions of a surface visible from all viewpoints), the union (which gives the portions visible from at least one viewpoint), etc.
- *Counting operators*: for example, the surface may be partitioned into regions, such that all the points of a region are visible from the same number of viewpoints.

The set of viewpoints considered is usually restricted to be a subset of the vertices of a TIN, or a subset of the cells of an RSG. In the remainder of the chapter, a visibility structure related to multiple viewpoints will be called a *multi-visibility structure* for brevity; if necessary, the operator used to obtain it will be specified. Note that the overlay of viewsheds contains more information than any other multi-visibility structure.

Any visibility structure can be encoded in either a continuous or a discrete way. For the viewshed (see Figure 1) a continuous encoding subdivides each cell of the DEM; this form is called a *continuous visibility map*, and it is mainly used for

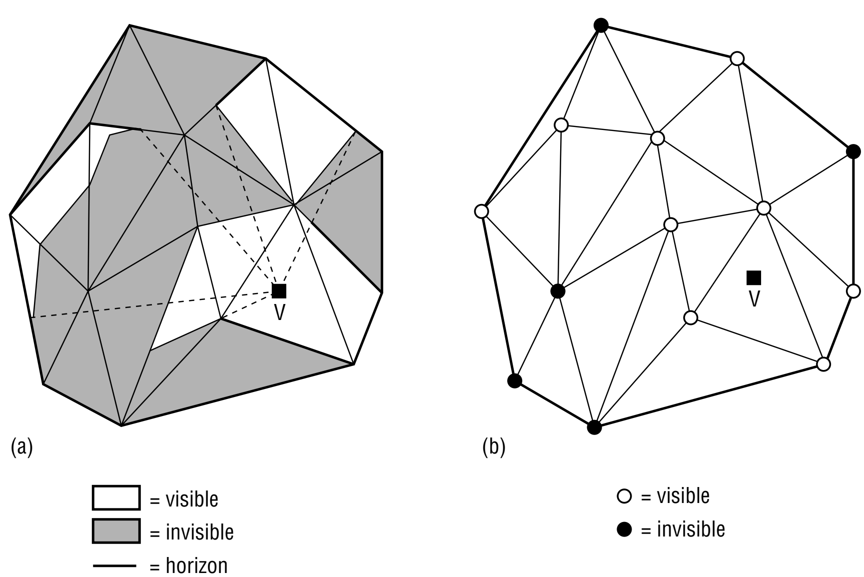

Fig 1. (a) The continuous-visibility map and the horizon; (b) the discrete visibility map on a TIN.

TINs. The continuous visibility map of a TIN with *n* vertices has a worst-case space complexity in $O(n^2)$. On RSGs, the viewshed is usually represented in a discrete way, by marking each grid cell or each vertex as visible or invisible. The resulting array of Boolean values is called a *discrete visibility map*. RSGs are usually dense, so that discrete visibility maps are accurate enough for application needs; on the other hand, for an RSG a continuous visibility map would be difficult to compute (since RSGs do not support linear interpolants) and huge in size. The discrete visibility map has an $O(n)$ worst-case space complexity for a $\sqrt{n}$ by $\sqrt{n}$ regular grid. A discrete visibility map for a TIN is sometimes considered (e.g. Lee 1991); such a map is obtained by marking each triangle or each vertex as visible or invisible; the spatial complexity is $O(n)$ for a TIN with *n* vertices.

A continuous representation of the horizon of a viewpoint *V* consists of a sequence of portions of terrain edges, radially sorted around *V*. As the continuous visibility map, this form is used for TINs. The size of the horizon on a TIN with *n* vertices is $O(n\,\alpha\,(n))$ (Cole and Sharir 1989), where α is the slowly growing inverse of Ackermann's Function. On an RSG, the horizon can be represented in a discrete way as a collection of grid cells; however, since there is no gain in space complexity with respect to a discrete visibility map, discrete horizon representations are not used in practice.

Visibility structures related to multiple viewpoints can be encoded in a variety of different ways. Discrete encodings are mainly used, due to the huge dimensions of a continuous encoding when many viewpoints are considered. Examples of such encodings are the visibility graph and visibility counts. The visibility graph (Puppo and Marzano 1996) represents the overlay of the discrete visibility maps of several viewpoints. It consists of a graph in which each node corresponds to a vertex or to a cell of a DEM and every pair of mutually visible nodes is joined by an arc. The spatial complexity of a visibility graph is $O(n^2)$ for a DEM with *n* vertices; visibility graphs are used both for RSGs and TINs, and are fundamental in solving several application problems (see section 6).

Visibility counts are discrete encodings of multi-visibility structures obtained through counting operators. A visibility count is obtained by labelling each vertex or each cell of a DEM with the number of viewpoints from which it is visible. This information is mainly considered on gridded models. A special case is the *intervisibility map* between two regions (Mills et al 1992); given a source region and a destination region (e.g. two rectangular blocks of cells of an RSG), each point of the destination region is labelled with the number of points of the source region from which it is visible. The size of an intervisibility map is determined by the number of cells of the destination region. Discrete representations of multi-visibility structures based on Boolean operators (e.g. union, intersection) reduce to arrays of Booleans.

4 ALGORITHMS FOR VISIBILITY COMPUTATION

In this section, an overview is given of algorithms for visibility computation on terrains. Two subsections are devoted to the computation of visibility structures, in continuous and discrete encodings, respectively; the last subsection deals with visibility queries on a DEM.

4.1 Computation of continuous visibility structures

As mentioned in section 3, continuous encodings of visibility structures are used only for TINs. Thus, the algorithms considered in this subsection are algorithms for TINs; they exploit the fact that a TIN describes a polyhedral surface. In general, polyhedral surfaces have deserved interest both from the GIS community and from computational geometers working on GIS. Some algorithms have a theoretical interest for their good asymptotic complexity, but are difficult to implement; other algorithms have been successfully implemented and show a good practical performance, whereas they exhibit a poor worst-case complexity, or they even lack a precise theoretical analysis.

The problem of computing the continuous visibility map of a TIN is connected with the more general hidden surface removal (HSR) problem for a 3-dimensional scene. Given a viewpoint and an image plane, HSR algorithms build the visible image of a scene, that is, a subdivision of the image plane formed by collecting the projections (images) of the portions visible from the viewpoint of each face of the scene (see Figure 2). Thus some algorithms are reported here for HSR, which can be used to compute the continuous visibility map of a TIN.

A common approach to visibility computation on a TIN consists of processing the faces in front-to-back order from the viewpoint. Given a viewpoint V, a cell c_1 of a DEM is said to be in front of another cell c_2 if a ray emanating from V intersects c_1 before intersecting c_2. A front-to-back order of a DEM is any total order of its cells consistent with the 'in front' relation; if $c_1 < c_2$ then c_1 may be in front of c_2, but not vice versa. A DEM is sortable if a front-to-back order exists. Because of their irregular structure, sortability is not guaranteed for all TINs. Delaunay-based TINs have been shown to be always sortable (De Floriani et al 1991); a non-sortable TIN can always be made sortable by splitting some of its triangles (Cole and Sharir 1989).

The front-to-back approach exploits the fact that a triangle t can be hidden only by triangles in front of it. At each step, a current horizon is maintained and used to determine the visibility of new triangles (see Figure 3). This method was developed and implemented by De Floriani et al

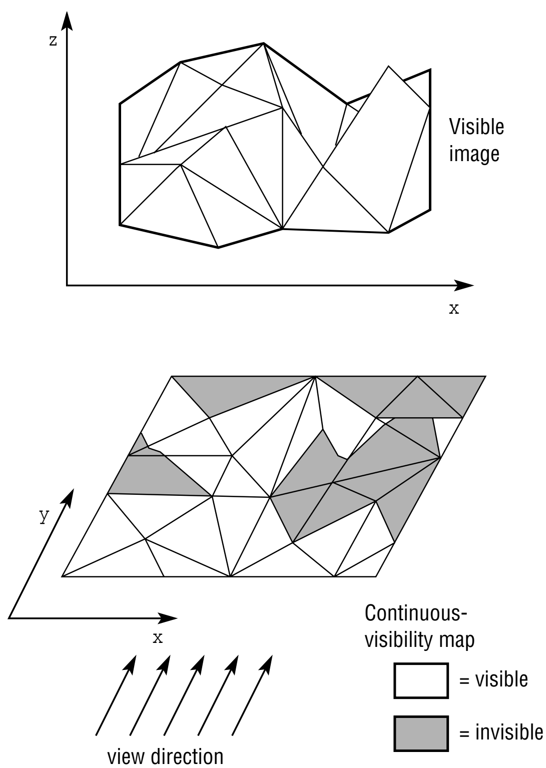

Fig 2. The continuous-visibility map and the visible image of a TIN. The viewpoint lies at infinity in the negative *y* direction, and the *x, z* plane is used as the viewplane.

(1989) and experiments show a nearly-linear time complexity, even though the asymptotic time complexity is $O(n^2 \alpha(n))$ in the worst case. The algorithm was parallelised by De Floriani et al (1994b) by using a data partitioning strategy. The domain is subdivided into radial sectors, which have the viewpoint as their common vertex, and each sector is assigned to a processor. The algorithm has been implemented on a hypercube machine nCUBE-2, a coarse-grained multiple instruction, multiple data (MIMD) architecture.

More sophisticated algorithms, still based on a front-to-back traversal, achieve an output-sensitive time complexity by storing the current horizon into some special data structures (Preparata and Vitter 1992; Reif and Sen 1988). The time is $O((n+d) \log^2 n)$, where n is the TIN size and d is the size of the computed visible image. Interest in these algorithms is mainly theoretical. Reif and Sen (1988) also propose a parallel algorithm, based on a variation of the front-to-back approach.

Still in the realm of theoretically interesting algorithms, the worst-case optimal algorithm by Edelsbrunner et al (1989) can be mentioned. It computes the visible image of a TIN in $O(n^2)$ time, based on a divide-and-conquer strategy. Theoretically efficient output-sensitive HSR algorithms, which can be applied to a TIN, have been proposed by Katz et al (1991) and by Overmars and Sharir (1992). Both algorithms exploit a front-to-back traversal of the scene; a parallel version of the latter algorithm has been proposed by Teng et al (1997).

An on-line algorithm has been proposed by Boissonnat and Dobrindt (1992). This algorithm computes the visible image of a generic set of triangles in the space (and thus the visibility map of a TIN), and is based on an incremental update of the visible image by inserting triangles one at a time. The kernel of the algorithm is a special data structure which provides bounds for the expected time and space complexity when averaging on all possible permutations of the input data. This algorithm was successfully implemented. The algorithm has also been extended to a fully dynamic method, which allows both insertions and deletions of triangles in expected $O(n \log n)$ time (Bruzzone et al 1995; Dobrindt and Yvinec 1993). Dynamic algorithms are of special interest because they are useful to update visibility information for a terrain when the underlying elevation model changes

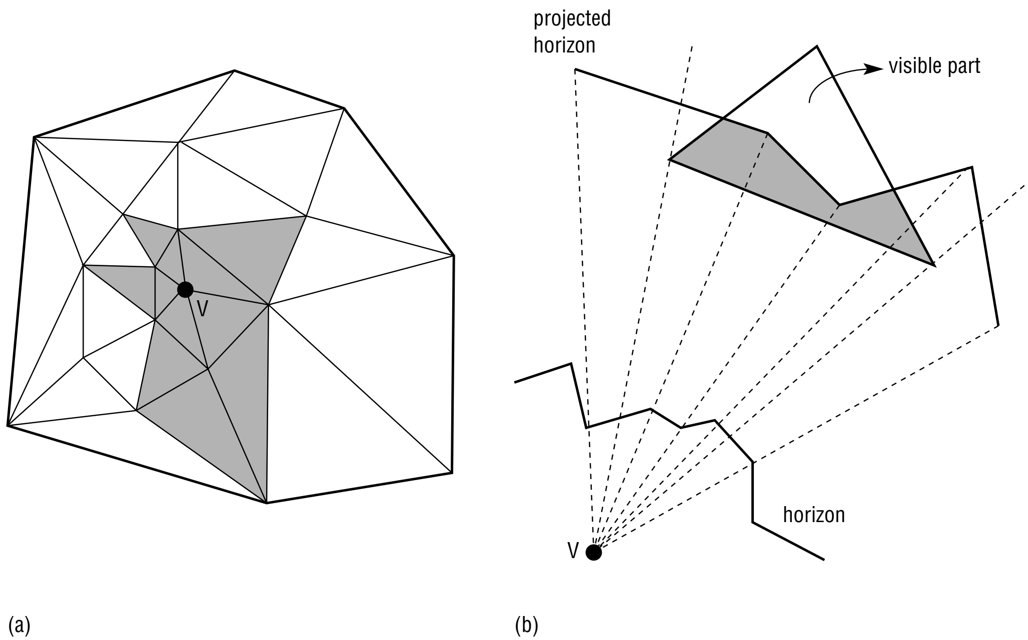

Fig 3. A generic step of a front-to-back visibility algorithm: (a) the set of already-processed triangles (shaded) and the current horizon (thick); (b) determination of the visibility of a triangle by projecting the current horizon on the triangle itself (the portions of the triangle lying above the projected horizon are visible).

(a problem occurring, for instance, with multi-resolution terrain representations, see section 5).

The horizon of a TIN is obtained as a side-product of algorithms for computing visibility maps (for example, front-to-back algorithms build the horizon as an auxiliary structure). The horizon can also be directly computed as the upper envelope of the set of segments obtained by projecting the terrain edges on an image plane. Existing upper-envelope algorithms are of two types: divide-and-conquer methods run in $O(n\,\alpha(n)\log n)$ (Atallah 1983) or $O(n \log n)$ (Hershelberg 1989) time (the latter is worst-case optimal since the inherent complexity of the problem is $O(n \log n)$); an incremental approach leads to a sub-optimal $O(n^2\,\alpha(n))$ time in a straightforward implementation, while a randomised version (De Floriani and Magillo 1995) has an expected running time of $O(n\,\alpha(n)\log n)$.

Though not very relevant for practical use, continuous encodings of multi-visibility structures for TINs can be obtained starting from the overlay of continuous visibility maps. Efficient algorithms exist which compute the overlay of two maps, the most common of which are based on a sweep-line technique (Bentley and Ottmann 1979; Chazelle and Edelsbrunner 1992; Clarkson and Shor 1989; Guibas and Seidel 1987; Mairson and Stolfi 1988). The time complexity is sensitive to the size of the final result (which may be quadratic in the size of the two input maps, depending on the number of intersections).

4.2 Computing discrete-visibility structures

Algorithms for discrete-visibility maps are based on the computation of the intersection between lines-of-sight from the viewpoint and edges of the DEM cells. A straightforward approach requires $O(n^2)$ time on a TIN and $O(n\sqrt{n})$ time on an RSG with n vertices. For RSGs, the elevation of each edge is assumed to vary linearly between the two endpoints, and the elevation of the terrain inside each grid cell is usually not considered.

Lee (1991) computes the discrete-visibility map of a TIN by using a front-to-back method, similar to the one used by De Floriani et al (1989) for continuous-visibility maps; here, the discretisation is achieved by considering a triangle as visible if all its three edges are completely visible.

Discrete-visibility maps are mainly used for RSGs. The method of Shapira (1990) traces a line-of-sight from the viewpoint *V* to any other point *P*, and starts walking along on the line-of-sight from *V* to *P*. The walk terminates when either an intersection between the line-of-sight and a terrain edge is found before reaching *P*, or when *P* is reached. This method performs redundant computations, since rays to different points of the grid may overlap partially. Other methods (e.g. Blelloch 1990) consider only rays joining the viewpoint *V* to a boundary point of the destination region, and determine the visibility of each cell along a line-of-sight by using the current tangent from the viewpoint to the terrain (see Figure 4). The complexity can be further reduced by sampling a subset of line-of-sight directions around a viewpoint, thus leading to approximate discrete-visibility maps.

Visibility computation on RSGs is an expensive process because of the size of the grid, especially when multi-visibility structures (e.g. a visibility graph) are computed. On the other hand, because of their regular topology RSGs are especially suitable to be handled by parallel methods developed for the architecture of massively-parallel computers (see Openshaw and Alvanides, Chapter 18). Parallel algorithms exploit the fact that the regular spatial structure of an RSG can be directly embedded into a parallel single instruction, multiple data (SIMD) architecture, such as the mesh or the hypercube. Mills et al (1992) and Teng et al (1993) have proposed parallel algorithms for computing intervisibility maps on an RSG.

The method of Mills et al (1992) uses a parallel version of the algorithm of Shapira. Every line-of-sight from every viewpoint of the source region is processed in parallel. Elevation data are communicated from one sight-line to an adjacent one, in order to reduce global communication between processors. The algorithm has been implemented on a Connection Machine CM-2.

The method of Teng et al (1993) performs a sweep traversal of the source region, and exploits the coherence at adjacent viewpoints in order to reduce global communications. They consider only lines-of-sight from a point of the source region to a boundary point of the target region, as does Blelloch (1990). If w is the maximum length (in cells) of a line-of-sight, and l is the side of source and destination regions, the time complexity of the algorithm is estimated as $O(l \log w)$, when using $O(l^2 w)$ processors. Experimental results obtained on a Connection Machine CM-2 match this estimate.

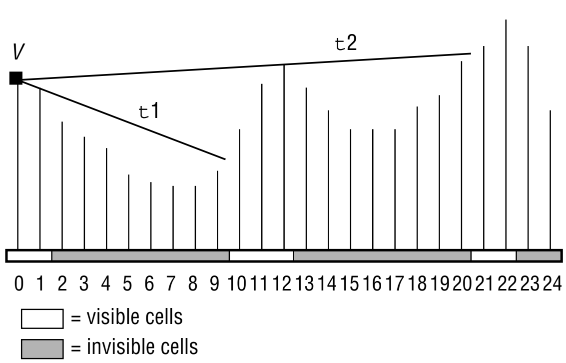

Fig 4. Walking along a line-of-sight in Blelloch's approach: a vertical section of the DEM in the direction of the line-of-sight is shown; *t1* is the current tangent for cells 2 to 10; *t2* for cells 13 to 21.

4.3 Answering visibility queries

Visibility queries concern the computation of visibility for objects located on the terrain rather than for the surface itself. Given a viewpoint *V*, a visibility query is defined by providing a query object. The problem consists of determining the visibility of the object from *V*. A visibility query for a point *Q* simply requires a test of whether *Q* is visible from *V*. As for the visibility of the terrain itself (see section 3), the visibility of a non-point object can be encoded either in a continuous way or in a discrete way. In the continuous approach, the query is answered by computing a partition of the query object into visible and invisible subsets. A discrete answer consists of marking the object with a Boolean value, according to some convention; the answer can be true either when the object is entirely visible, or when it is at least partially visible, or when it is visible for more than a certain percentage of its extent, etc.

Finding the visibility of a query object from a given viewpoint *V* can be solved by examining the lines-of-sight joining *V* to a point on the object. The simplest case occurs when the query object is a point *P*. In this case, a 'brute force' approach that searches for the intersection of segment $\overline{PV}$ with the edges of the DEM takes $O(n)$ time on a TIN and $O(\sqrt{n})$ time on an RSG. If several queries must be solved for the same viewpoint, some kind of preprocessing can reduce the query time.

If a visibility map of the terrain is available, then testing the visibility of a query point lying on the surface reduces to a *point location* in such a map. If the query point has a non-null height, then the approach is different depending whether we are considering continuous visibility on TINs or discrete visibility. The latter case reduces to point location in an enriched version of the discrete-visibility map, storing, for each cell, the minimum height which should be added to the cell in order to make it visible (zero if and only if the cell is visible). Such an enriched map can be computed with the same techniques. Point location within the simple or enriched discrete-visibility map can be done in constant time by computing the indices of the row and column containing the query point. As far as continuous visibility on TINs is concerned, testing whether a given point is visible reduces to locating the projection of such a point in the visible image of the TIN. Both the visible image and the continuous-visibility map are plane subdivisions; thus, existing techniques for point location within a plane subdivision can be used (see Kirkpatrick 1983; Lee and Preparata 1977; Preparata and Shamos 1985; Sarnak and Tarjan 1986), typically resulting in logarithmic query times.

Cole and Sharir (1989) proposed a data structure which allows a logarithmic query time by using an amount of memory less than the $O(n^2)$ space required by a continuous-visibility map or a visible image. They reduce a visibility query on a TIN to a ray-shooting query, that is, to the problem of determining the first face of the terrain hit by a ray emanating from the viewpoint and passing through the query point. For answering such queries, Cole and Sharir build a balanced binary tree, which stores a set of partial horizons. The space complexity of the whole tree is only $O(n\,\alpha(n) \log n)$.

5 VISIBILITY ON MULTI-RESOLUTION TERRAIN MODELS

Often, huge sampled datasets are available for a topographic surface. This allows accurate DEMs at the cost of storage space and access times. Since not all tasks require the same level of detail, the use of high-resolution models may affect applications for which many of the details are not relevant. Multi-resolution terrain models have been developed to provide a compact representation of a surface at different resolutions. A multi-resolution model avoids redundancy of information and supports an easy extraction of terrain representations at any level of detail. This section first introduces multi-resolution terrain models and then briefly reviews the available methods for visibility computation on such models.

5.1 Multi-resolution terrain models

Multi-resolution terrain models are built from a given (huge) set of data by using iterative procedures of two kinds: simplification methods start from the complete dataset and progressively eliminate points, while refinement techniques progressively include points into an initial minimal set. At appropriate steps of the process, fragments of the current DEM are selected to be stored in the model.

A generic multi-resolution model of a terrain is based on a collection of fragments of DEMs, each characterised by a certain degree of accuracy, which can be combined in different ways to provide a description of the whole surface. A multi-resolution model encodes such components together with information needed to combine them into a single DEM satisfying any given level of accuracy. Such additional information typically includes:

- relationships of spatial interference between fragments (two overlapping fragments will provide two representations of the same area at different accuracies);
- information about adjacency and boundary matching between fragments (used to determine whether the union of two fragments forms a proper DEM).

Such information is represented in different ways depending on the structure of the multi-resolution model, which in turn depends on the strategy used for its construction.

A large subclass of multi-resolution models proposed in the literature is characterised by a nested subdivision of the domain; such models are usually termed hierarchical. A hierarchical model can be effectively described by a tree where nodes are the fragments (local DEMs), and arcs correspond to containment of a DEM into a cell of another DEM (see Figure 5). *Quadtrees* (Chen and Tobler 1986; Samet and Sivan 1992) and *quaternary triangulations* (Gomez and Guzman 1979) are regular hierarchical models based on the recursive partition of a square

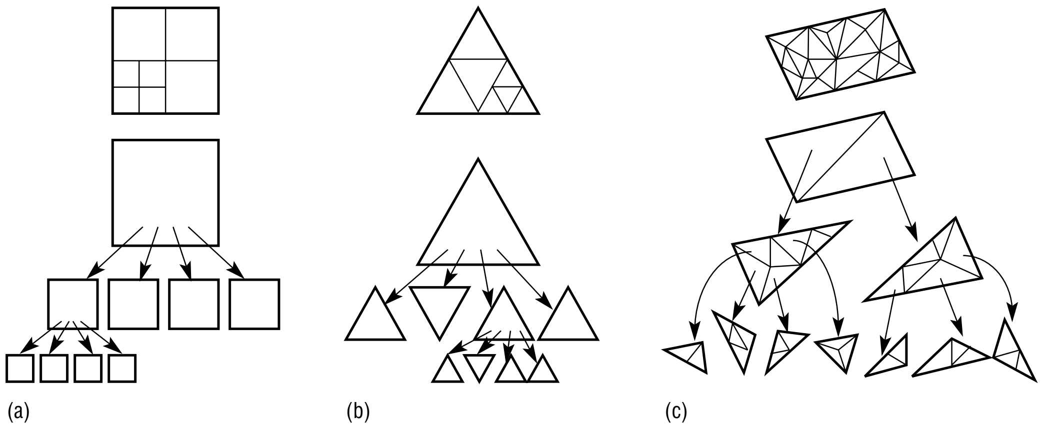

Fig 5. Hierarchical terrain models: (a) quadtree; (b) quaternary triangulation; (c) hierarchical TIN.

or an equilateral triangle, respectively, into four equal parts. In *restricted quadtrees* (Herzen and Barr 1987) the problem of preserving the continuity of the surface is solved by imposing the condition that adjacent squares cannot differ by more than one level in the refinement process, and by triangulating the final cells. Regular hierarchical models can be built both by refinement and by simplification. Other hierarchical terrain models are based on TINs; hierarchical TINs rely on a top-down refinement process, driven by various criteria (e.g. random or accuracy-driven strategies for the insertion of points, Delaunay or heuristic triangulation, etc.). The continuity of the surface is guaranteed through a consistent refinement of edges. Adaptive hierarchical triangulations (Scarlatos and Pavlidis 1990) and hierarchical Delaunay triangulations (HDTs) (De Floriani and Puppo 1995) are the two major examples of such models.

In more general multi-resolution models the spatial interference between two fragments does not necessarily reduce to a containment of one fragment by the other. All existing proposals of non-hierarchical multi-resolution models are based on TINs (see Figure 6). The first proposal is the *Delaunay pyramid* (De Floriani 1989), which encodes fragments from a sequence of Delaunay TINs describing a terrain at a sequence of increasing resolutions. This model does not rely on a special construction technique, and can be built by simplification as well as by refinement. Interference links are stored between pairs of consecutive triangles which have a proper intersection. The model proposed by Berg and Dobrindt (1995) is built through iterative simplification of a Delaunay TIN; at each step, a set of independent vertices (i.e. vertices that are not endpoints of the same edge) of

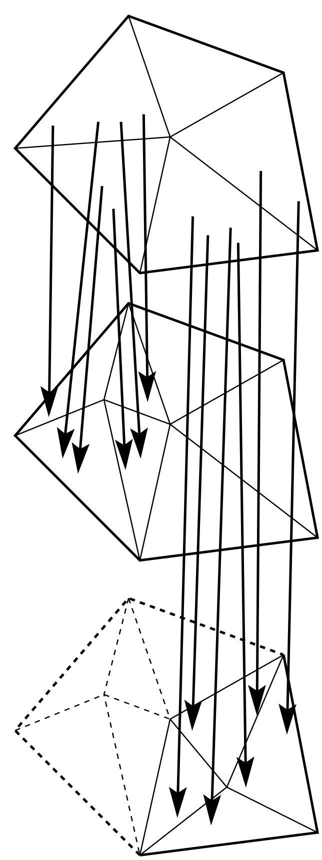

Fig 6. A pyramidal terrain model.

small degree is removed, and the ‘holes’ left by those vertices are re-triangulated. Interference links are maintained between the triangles incident in a removed vertex and those created to fill the hole.

The multi-triangulation proposed by Puppo (1996) is a general model which includes all previously mentioned models as special cases. The basic idea is a partial order of fragments, which drives their combination. The model can be described as a directed acyclic graph, where the nodes are the fragments, with the properties that two fragments connected by an arc have a spatial interference, and every cut of the graph corresponds to a TIN describing the whole terrain; larger cuts correspond to more detailed representations of the surface (see Figure 7). Cignoni et al (1995) use a different and less complex data structure for encoding a multi-triangulation.

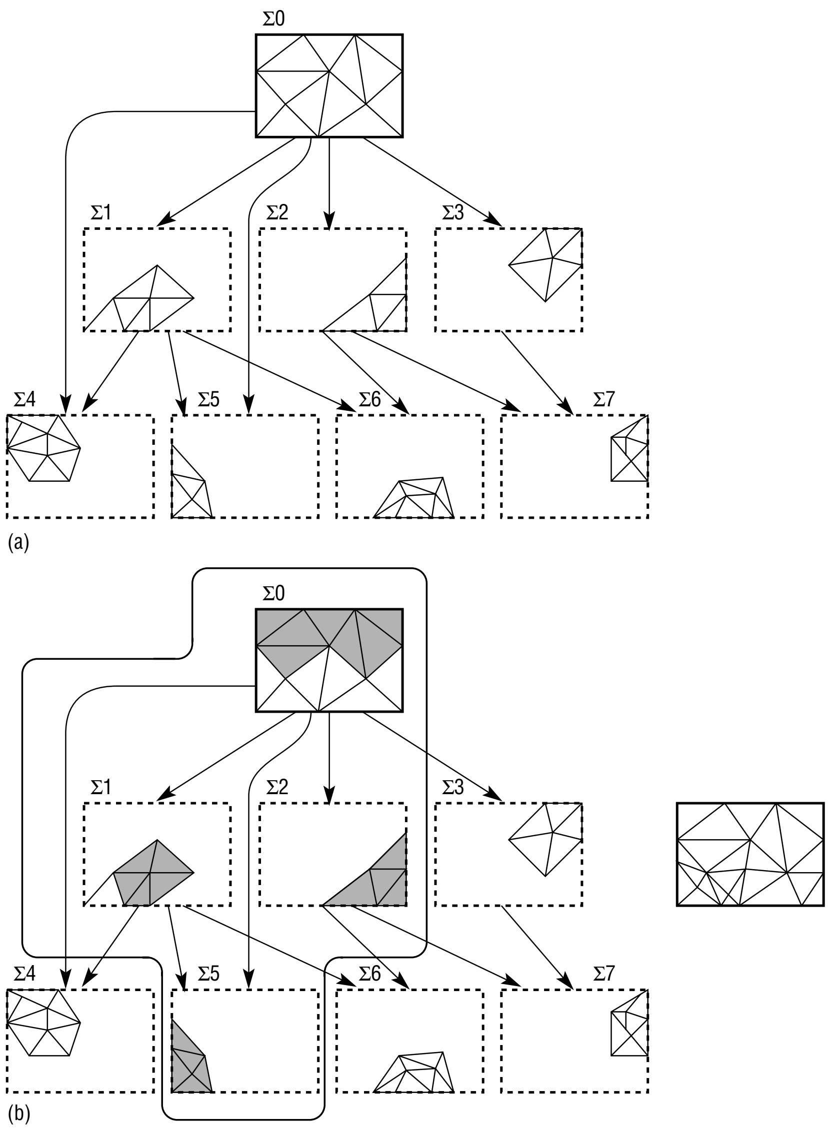

Fig 7. (a) A graph representing a multi-triangulation (the fragment stored in each node is shown, the dashed rectangle is the domain); and (b) a cut and the corresponding TIN defined over the domain (the TIN is obtained by collecting the ‘exposed’ triangles of the cut, shown shaded).

5.2 Visibility algorithms on multi-resolution terrain models

Since multi-resolution terrain models do not provide an explicit terrain representation, a DEM, describing the surface at a given user-defined level of resolution, must be extracted to be used for visibility computations. The level of resolution may be constant (i.e. a single threshold value is defined for the error over the domain), or variable (i.e. the maximum error over each cell is defined according to a threshold function). In visibility applications, an accuracy that decreases according to the distance from the viewpoint is especially interesting; because errors in elevations near the viewpoint are amplified in proportion to the distance, it is convenient to represent the topography more accurately near the viewpoint than on the rest of the surface (Felleman and Griffin 1990).

A variable-accuracy DEM will be made up of cells from different levels of a multi-resolution model. The main difficulty here is ensuring the continuity of the surface described by the extracted model. Cignoni et al (1995) propose an extraction algorithm for the case of accuracy decreasing with the distance from a viewpoint, which guarantees continuity. The method starts from the triangle having the maximum resolution and enlarges the model by including adjacent triangles; the approach applies to their own model and to any hierarchical TIN. Puppo (1996) proposes a general approach for extracting a DEM at variable resolution, based on interference information only. The method performs a traversal of the graph encoding interference among fragments, and identifies a minimal cut of the graph which satisfies the desired accuracy. This method, with some additional care to avoid cracks in the extracted surfaces, applies also to any hierarchical model (see De Floriani and Magillo 1996).

Once a DEM at the desired resolution has been obtained, any algorithm for visibility computation can be applied to it. On-line algorithms (e.g. Boissonnat and Dobrindt 1992; De Floriani and Magillo 1995) seem to be the most suitable because they can be run in parallel with the construction of the extracted DEM. Magillo and De Floriani (1994) have presented algorithms which compute the visibility map on a hierarchical TIN by navigating the tree-like structure of the model, and thus do not need the explicit construction of a TIN at the given accuracy.

A further problem connected to visibility on multi-resolution terrain models is the update of already-computed visibility structures when changing the resolution in some portion of the domain. This is required in applications such as flight simulation, where the focus of attention is rapidly moving, and thus the maximum resolution is required for different areas at different times. The visibility-update problem can be solved by recomputation or by applying dynamic algorithms. Such algorithms update a visibility structure after the deletion of old DEM cells and the insertion of new ones. Dynamic algorithms have been proposed for horizon computation (De Floriani and Magillo 1995) and for continuous-visibility maps on a TIN (Bruzzone et al 1995; Dobrindt and Yvinec 1993). The core of all these algorithms is a special data structure which helps in locating the parts of the structure affected by an update, providing a good expected performance for a random sequence of updates.

Finally, the brute force approach to visibility queries (see section 4.3) can be combined with a traversal of the tree encoding a hierarchical terrain model to answer visibility queries efficiently at a certain level of resolution. The aim is locating those cells, at the given accuracy, which may hide a query object; the visibility of the object is then tested only against those cells. The problem reduces to an interference query on the extracted DEM, for which the tree-like structure acts as a spatial index (see Figure 8). The major advantages are obtained with regular hierarchical models.

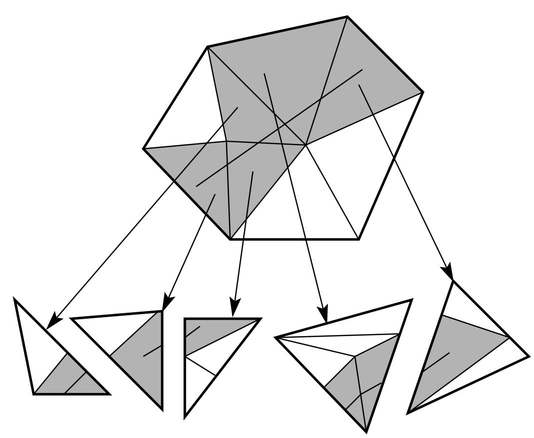

Fig 8. Processing a segment interference query on a hierarchical terrain model. First, the triangles of the root intersected by the segment are found; then the search is repeated on the nodes defining such triangles.

6 VISIBILITY-RELATED PROBLEMS AND ALGORITHMS

Interesting application problems on a terrain, which can be solved based on visibility information, can be classified into the following major categories:

- problems which require the placement of observation points on a topographic surface, according to suitable requirements;
- line-of-sight communication problems;
- problems regarding the computation of paths on a terrain, with certain visibility properties.

Viewpoint placement problems require the placement of several observation points on a terrain, in such a way that a large part of the surface is visible from at least one point. Applications include the location of fire towers, artillery observers, radar sites, etc. In general, the aim is either to minimise the number of viewpoints to cover a target area, or, in a dual formulation, to select a fixed number of points in such a way that the visible area is maximised.

For the placement of a single observation point, algorithms running in polynomial time are known. If the height is fixed, a point that can see the whole surface (if one exists) can be determined in $O(n \log n)$ time, while the point with the lowest elevation, from which the entire terrain is visible, can be determined in $O(n \log^2 n)$ time on a TIN with n vertices (Sharir 1988).

The problem of determining an optimal set of viewpoints is usually addressed in a discrete version, by allowing viewpoints to be located only on the vertices of a DEM. Even in this case, the complexity of the problem is exponential in n (Puppo and Marzano 1996), and thus heuristic solutions are used. Several heuristic algorithms are discussed by Lee (1991); Lee considers both the case when the heights of the viewpoints are fixed, and the case when heights are variable and must be minimised. The methods presented are based on a greedy approach; they iteratively add viewpoints to an initially empty set, or eliminate viewpoints from an initial set containing all vertices. Bose et al (1996) address the problem of placing vertex-guards and edge-guards on a terrain as a graph matching problem, and propose simple polynomial-time algorithms to place a worst-case optimal number of guards.

Line-of-sight communication problems consist of finding a visibility network, and connecting two or more sites, such that every two consecutive nodes of the network are mutually visible. Straightforward applications are in the location of microwave transmitters for telephone, FM radio, television, and digital data networks (see Fry, Chapter 58). A typical problem is finding the minimum number of relay towers necessary for line-of-sight communication between a set of sites. The given sites and the relay towers are usually restricted to the vertices of a DEM.

Puppo and Marzano (1996) show that the only visibility information necessary for solving problems of this kind is the visibility graph, and reduce them to classical graph problems. Connecting two sites reduces to a shortest path search on the visibility graph. The construction of a line-of-sight network between several sites reduces to finding a minimum connected sub-graph of the visibility graph, containing the given sites, also known as a *Steiner problem* in graph theory, which is known to be NP-complete. Thus, heuristics developed in the graph literature can be applied. De Floriani et al (1994a) propose a dynamic approach which reduces memory requirements by computing on-line only those portions of the visibility graph that are needed for the computations.

Paths can be defined on a terrain, with application-dependent visibility characteristics. A smuggler's path is the shortest path, connecting two given points, such that no point on the path is visible from a predefined set of viewpoints. Conversely, a path where every point can be seen from all viewpoints is known as a scenic path. Path problems are usually addressed on a DEM by restricting the viewpoints to be vertices, and the path to passing along edges. A solution (if one exists) can be determined by first computing the vertices which are visible or invisible from all the viewpoints, and then applying a standard shortest path algorithm to the edges connecting them. The problem of finding a path 'as hidden as possible' is considered by Puppo and Marzano (1996); a shortest path is computed after having weighted every edge with the number of viewpoints that can (or cannot) see it.

A problem of high practical interest is that of updating visibility for a viewpoint moving along a trajectory. Such a problem has received little attention, perhaps because of its intrinsic difficulty. Existing results are restricted to straight-line trajectories, and, in particular, vertical trajectories (Bern et al 1994; Cole and Sharir 1989); they have mainly a theoretical interest. They are related to the computation of the points along the trajectory at which topological changes occur in the visible image, or to the processing of ray-shooting queries (i.e. finding the terrain face 'seen' by a given visual ray) from arbitrary viewpoints lying on the path.

7 CONCLUSIONS

Many visibility-related problems on terrains still lack practically satisfactory solutions; this group includes such problems as the update of visibility after modifications in the underlying elevation model (e.g. in the context of multi-resolution terrain modelling; see section 5) or for a moving viewpoint, the computation of optimal paths according to given visibility criteria, and several other optimisation problems (see section 6). This lack of efficient solutions is partially attributable to the fact that such problems have received little attention from the research community, both because most of them are intrinsically hard (several optimisation problems have been shown to be NP-complete), and because some of them have come to the attention of the GIS community only recently (for instance, problems on multi-resolution terrain models). In many cases (including path and communication problems), a good definition of the problem is still missing, thus making the work of finding algorithmic methods even more difficult. More research effort should be spent in investigating the problems mentioned above, since they have a high impact on applications and a fundamental importance in the development of information systems of the future.

References

Atallah M 1983 Dynamic computational geometry. *Proceedings, 24th IEEE Symposium on Foundations of Computer Science*. IEEE Computer Society: 92–9

Bentley J L, Ottmann T A 1979 Algorithm for reporting and counting geometric intersections. *IEEE Transactions on Computers* 28: 643–7

Berg M de, Dobrindt K 1995 On levels of detail in terrains. *Proceedings, Eleventh Annual ACM Symposium on Computational Geometry*: C26–C27

Bern M, Dobkin D, Epstein D, Grossman R 1994 Visibility with a moving point of view. *Algorithmica* 11: 360–78

Blelloch G E 1990 *Vector models for data-parallel computing*. Cambridge (USA), MIT Press

Boissonnat J D, Dobrindt K 1992 On-line construction of the upper envelope of triangles in $\mathbf{R}^3$. In Wang C A (ed.) *Proceedings, Fourth Canadian Conference on Computational Geometry*: 311–15

Bose P, Kirkpatrick D, Li Z 1996 Efficient algorithms for guarding or illuminating the surface of a polyhedral terrain. In Fiala F, Kranakis E, Sack J R (eds) *Proceedings, Canadian Conference on Computational Geometry*: 217–22

Bruzzone E, De Floriani L, Magillo P 1995 Updating visibility information on multi-resolution terrain models. In Frank A U, Kuhn W (eds) *Proceedings, Conference on Spatial Information Theory (COSIT 95)*. Lecture Notes in Computer Science 988. Berlin, Springer: 279–96

Chazelle B, Edelsbrunner H 1992 An optimal algorithm for intersecting line segments in the plane. *Journal of the Association for Computing Machinery* 39: 1–54

Chen Z T, Tobler W R 1986 Quadtree representation of digital terrain. *Proceedings AutoCarto*: 475–84

Cignoni P, Puppo E, Scopigno R 1995 Representation and visualization of terrain surfaces at variable resolution. In Scateni R (ed.) *Proceedings, International Symposium on Scientific Visualization*. Singapore, World Scientific. *http://www.disi.unige.it/person/PuppoE*

Clarkson K L, Shor P W 1989 Application of random sampling in computer geometry. *Discrete and Computational Geometry* 4(5): 387–421

Cole R, Sharir M 1989 Visibility problems for polyhedral terrains. *Journal of Symbolic Computation* 17: 11–30

De Floriani L 1989 A pyramidal data structure for triangle-based surface description. *IEEE Computer Graphics and Applications* 9: 67–78

De Floriani L, Falcidieno B, Nagy G, Pienovi C 1989 Polyhedral terrain description using visibility criteria. *Technical Report 17*. Genova, Institute for Applied Mathematics, National Research Council

De Floriani L, Falcidieno B, Nagy G, Pienovi C 1991 On sorting triangles in a Delaunay tessellation. *Algorithmica* 6: 522–32

De Floriani L, Magillo P 1995 Horizon computation on a hierarchical terrain model. *The Visual Computer:* 11: 134–49

De Floriani L, Magillo P 1996 A comprehensive framework for spatial operations on hierarchical terrain models. *Technical Report DISI-TR-96-15*. University of Genova, Department of Computer and Information Sciences

De Floriani L, Marzano P, Puppo E 1994a Line-of-sight communication on terrain models. *International Journal of Geographical Information Systems* 8: 329–42

De Floriani L, Montani C, Scopigno R 1994b Parallelizing visibility computations on triangulated terrains. *International Journal of Geographical Information Systems* 8: 515–32

De Floriani L, Puppo E 1995 Hierarchical triangulation for multi-resolution surface description. *ACM Transactions on Graphics* 14: 363–411. *http://www.disi.unige.it/person/PuppoE*

Dobrindt K, Yvinec M 1993 Remembering conflicts in history yields dynamic algorithms. In Ng K W, Raghavan P, Balasubramanian N V, Chin F Y L (eds) *Algorithms and computation*. *Lecture notes in computer science 762*. Hong Kong, Springer: 21–30

Dyn N, Levin D, Rippa S 1990 Data-dependent triangulations for piecewise linear interpolation. *IMA Journal of Numerical Analysis* 10: 137–54

Edelsbrunner H, Guibas L J, Sharir M 1989 The upper envelope of piecewise linear functions: algorithms and applications. *Discrete and Computational Geometry* 4: 311–36

Felleman J P, Griffin C 1990 The role of error in GIS-based viewshed determination – a problem analysis. *Technical Report EIPP-90-2*. State University of New York, Institute for Environmental Policy and Planning

Gomez D, Guzman A 1979 Digital model for 3-dimensional surface representation. *Geoprocessing* 1: 53–70

Guibas L J, Seidel R 1987 Computing convolutions by reciprocal search. *Discrete and Computational Geometry* 2: 175–93

Hershelberg J 1989 Finding the upper envelope of *n* line segments in O(*n* log *n*) time. *Information Processing Letters* 33: 169–74

Herzen B von, Barr A H 1987 Accurate triangulations of deformed, intersecting surfaces. *Computer Graphics* 21: 103–10

Katz M J, Overmars M H, Sharir M 1991 Efficient hidden surface removal for objects with small union size. *Proceedings, Seventh ACM Symposium on Computational Geometry*. New York, ACM Press: 31–40

Kirkpatrick D G 1983 Optimal search in planar subdivision. *SIAM Journal of Computing* 12: 28–33

Lee J 1991a Analyses of visibility sites on topographic surfaces. *International Journal of Geographical Information Systems* 5: 413–29

Lee J, Preparata F P 1977 Location of a point in a planar subdivision and its applications. *SIAM Journal of Computing* 6: 594–606

Magillo P, De Floriani L 1994 Computing visibility maps on hierarchical terrain models. In Pissinou N, Makki K (eds) *Proceedings, Second ACM Workshop on Advances in Geographic Information Systems (GIS 94)*. New York, ACM Press: 8–15

Mairson H G, Stolfi J 1988 Reporting and counting intersections between two sets of line segments. In Earnshaw R A (ed.) *Theoretical foundations of computer graphics and CAD*. NATO ASI Series F40. Berlin, Springer: 307–25

Mills K, Fox G, Heimbach R 1992 Implementing an intervisibility analysis model on a parallel computing system. *Computers and Geosciences* 18: 1047–54

Nagy G 1994 Terrain visibility. *Computer and Graphics* 18: 763–73

Overmars M, Sharir M 1992 A simple output-sensitive algorithm for hidden surface removal. *ACM Transactions on Graphics* 11: 1–11

Preparata F P, Shamos M I 1985 *Computational geometry: an introduction*. Berlin, Springer

Preparata F P, Vitter J S 1992 A simplified technique for hidden-line elimination in terrains. In Finkel A, Jantzen M (eds) *Lecture notes in computer science 577*. Berlin, Springer: 135–44

Puppo E 1996 Variable resolution terrain surfaces. In Fiala F, Kranakis E, Sack J R (eds) *Proceedings, Canadian Conference on Computational Geometry*: 202–10. Also published in longer version as *Technical Report 6/96*, CNR – Institute for Applied Mathematics, Genova, Italy, 1996. *http://www.disi.unige.it/person/PuppoE*

Puppo E, Marzano P 1996 Discrete visibility problems and graph algorithms. *International Journal of Geographical Information Systems* 11: 139–61

Reif J, Sen S 1988 An efficient output-sensitive hidden surface removal algorithm and its parallelization. *Proceedings, Fourth ACM Symposium on Computational Geometry*. New York, ACM Press: 193–200

Samet H, Sivan R 1992 Algorithms for constructing quadtree surface maps. *Proceedings, Fifth International Symposium on Spatial Data Handling*: 361–70

Sarnak N, Tarjan R E 1986 Planar point location using persistent search trees. *Communications of the Association for Computing Machinery* 29: 669–79

Scarlatos L, Pavlidis T 1990 Hierarchical triangulation using terrain features. *Proceedings IEEE Conference on Visualization*. IEEE Computer Society: 168–75

Shapira A 1990 'Visibility and terrain labeling'. Masters thesis. Troy, Rensselaer Polytechnic Institute

Sharir M 1988 The shortest watchtower and related problems for polyhedral terrains. *Information Processing Letters* 29: 265–70

Teng Y A, Menthon D de, Davis L S 1993 Region-to-region visibility analysis using data parallel machines. *Concurrency: Practice and Experience* 5: 379–406

Teng Y A, Mount D, Puppo E, Davis L S 1997 Parallelizing an algorithm for visibility on polyhedral terrain. *International Journal of Computational Geometry and Applications* 7: 75–8