Dynamic Computation of Levels of Detail in Complex Outdoor Environments

Charles E. Hughes[1 2], J. Michael Moshell[2], Valerie K. Sims[2], and Qingyi Yu[2]

Abstract--Modeling and rendering of natural scenes based on Lindenmayer-systems (L-systems) has been employed in graphical simulations for several decades. Its main use has been in generating photo-realistic images of plants. However, L-plants (plants that are generated using L-systems) have not been used in real-time environments due to their rendering complexity. The goal of our research is to develop and implement algorithms that dynamically generate Level-of-Detail (LOD) models for L-plants. The recursive coalescence of stems and leaf clusters used to generate LOD models is computed online based on user-specified LOD ranges and reduction percentages, as well as plant characteristics such as height and radius. The prototype implementation reported here simulates a complex, interactive scene consisting of L-oaks and L-palmettos.

Index Terms--real-time simulation, virtual environments, plant models, levels-of-detail, Lindenmayer systems

I. Introduction

L-systems originally provided a class of formal models for simulating the development of multi-cellular organisms. Their use was then extended into modeling of higher plants and complex branching structures. Since the introduction of its turtle interpretation, this formalism has further facilitated specification of models for graphical rendering. Recent uses include the generation of realistic models of entire plant ecosystems, such as forests and grasslands. Models of these ecosystems have a wide range of existing and potential applications, including visualization of ecosystems for research, training and educational purposes, and synthesis of scenes for computer art and animations.

One problem with L-systems modeling is that the creation of a realistic densely vegetated scene can take several days just to define the plant models and then an hour or more compute time on a high-end graphical system to synthesize the scene. The implications of this are that the amount of geometric data needed to accurately depict a detailed outdoor scene is far more than can be represented on modern computers. Many approaches have been developed to obtain a good tradeoff between the realism of the images and the computer resources (e.g. CPU time and memory) needed to generate them. One particularly useful technique is that of controlling the Level-of-Detail (LOD) of objects in the scene.

Since the exact shape of the plant modeled by an L-system is unknown until its generation, any LOD method must be automatic and operate online. These requirements give rise to several technical challenges. The first one is how to automatically generate LOD models for trees. Because of speed requirements, the most common solution for LOD modeling is to use static two-dimensional texture-mapped trees drawn as rotating billboards. Unfortunately, the appearance produced by billboards can be objectionable. This is especially evident when a viewer is in motion. The second challenge is the management (selection) of the various levels of detail. Most work on this issue has focused on maximizing the computable visual benefits while limiting the rendering cost. This kind of LOD management ignores an important metric, the non-computable visual benefits, i.e. the subjective feeling about the scene by human observers. One aspect along this subjective axis can be measured by how well the user does in the performance of tasks in the virtual scene, for example, in the performance of search-and-rescue missions in dense forests.

The work reported here builds a natural scene simulating a real wooded area (approximately 20m × 10m). The scene is generated using an L-system, which has been given plant growth patterns to produce a forest with palmettos, a single species of oak and an appropriate synthetic background. The primary contribution is the development and implementation of a novel method of automatic LOD modeling of the trees, in which a user can specify successive reduction levels in the number of stems, while the basic shapes of the trees always remain unchanged.

A secondary contribution is the design and implementation of a user interface that may be employed for subject tests on human performance in the simulated natural scene. The intended use of this is to provide a means by which human factors researchers can run tests to study the impacts of LOD parameters on human perception and performance.

In Section 2 we briefly revisit the development of L-systems, LOD modeling for L-plants and several methods

Charles E. Hughes, School of Electrical Engineering and Computer Science, University of Central Florida, Orlando, FL 32816.

[1] Research was supported by a grant from the National Science Foundation (EIA 9986051).
[2] Research was supported by a grant from the Army Simulation, Training and Instrumentation Command (STRICOM).

of LOD selection. In Section 3 we discuss the formal definition of our parametric L-system and the parameters used in the modeling of our L-plants. In Section 4 we discuss how to recursively build LOD models online given an L-oak with the highest detail. In Section 5 we show the simulated woods and a background generated from the woods. Finally, in Sections 5 and 6, we discuss some advantages and the limitations of our LOD method, and outline our plans for future work.

II. PRIOR WORK

A. L-systems

L-systems are sets of rules and symbols (formal grammars) that model growth processes. This family of grammars was originally introduced to model the development of simple multi-cellular organisms (for example, algae) in terms of division, growth, and death of individual cells. It has subsequently been extended to higher plants and complex branching structures described as configurations of modules in space. These application gained momentum after 1984, when Smith [1] introduced state-of-art computer graphics techniques to visualize the structures and processes being modeled. From the perspective of our research, the most relevant work includes the development of parametric L-systems, a particularly convenient programming tool for expressing models of plant development [2,3], and the introduction of turtle interpretation (a means of using the output of the grammar to draw images) of L-systems [4,5].

B. LOD Modeling for L-plants

Traditionally, the most popular *Level-of-Detail* method for use with trees is billboarding. In billboarding, trees are drawn with simple planar texture mapped geometry that is transformed to always face the viewer. Since a tree has a roughly cylindrical symmetry, an axial rotation is used about the axis running parallel to the tree trunk.

In addition to billboarding, there is another heuristic multiresolution representation specific to trees (Weber and Penn [6]), which allows for reducing the number of geometric primitives in the models that occupy only a small portion on the screen. This method is only effective when used to optimize the display of large quantities of trees. In contrast, rendering at close range using this technique is not quite fast enough to meet the needs of real-time simulation.

C. LOD Modeling for L-plants

In addition to modeling, LOD-based systems must implement a strategy to select coarse level representations as the model moves farther away from the viewer. Traditional LOD selection is based only on the distance between the object and the viewpoint. The problem is: how can we get the best overall image quality while limiting the number of polygons and maintaining a guaranteed frame rate?

Consideration of this problem leads to several criteria for the effective selection of LOD. Two common ones are *frame rate oriented LOD selection* as exemplified in Funkhouser and Sequin [7] and *perceptual oriented LOD selection* as exemplified in Watson, et al [8,9], and Reddy, et al [10]. Additional schemes focus on *visibility*, discarding polygons that are off-screen, oriented away from the viewer or occluded (Law [11]). Cohen-Or et al. [12] and Nadler et al. [13] outline a visibility preprocessing method for an outdoor environment that partitions the view space into cells. Using this, they formulate the probability for a given object to be visible from a given view cell as a function of distance from the view cell and density of the intervening scene. In comparison to Euclidean (pure distance-based) techniques, this visibility measure provides a better criterion of LOD selection for densely occluded scenes.

III. DEFINITION OF PARAMETRIC L-PLANTS

Since our modeling of specific species of oak and palmetto is based on a parametric L-system, we will discuss the definition of the system first. Then the modeling of L-palmettos and L-oaks will be presented.

A parametric L-system is an L-system operating on *parametric* words, which are strings of *modules* consisting of *letters* with associated *parameters*. The letters belong to an alphabet *V*, and the parameters belong to the set of real numbers ***Â*** A module with letter $A \in V$ and parameters $a_1, a_2, \ldots, a_n \in$ ***Â*** is denoted by $A(a_1, a_2, \ldots, a_n)$. Every module belongs to the set *M = V ´* ***Â*****, where ***Â***** is the set of all finite sequences of parameters. The set of all strings of modules and the set of all such nonempty strings are denoted by *M* = (V ´* ***Â*****)** and $M^+ = (V$ ´ ***Â*****$)^+$ respectively.

The real-valued *actual* parameters appearing in the words have a counterpart in the *formal* parameters, which may occur in the specification of L-system productions. If ***S*** is a set of formal parameters, then *C(**S**)* denotes a logical expression with parameters from ***S***, and *E(**S**)* is an arithmetic expression with parameters from the same set.

A *parametric 0L-system* is defined as an ordered quadruple G = <*V*, ***S***, *W*, *P*>, where:

- *V* is the *alphabet* of the system,
- ***S*** is the *set of formal parameters,*
- *W* ***Î*** *(V ´* ***Â*****)*$^+$ is a nonempty parametric word called the *axiom*,
- *P* ***Ì*** *(V ´* ***S*****) ´ C(**S**) ´ (V ´ E(**S**)*)** is a finite *set of productions.*

The symbols **:** and ® are used to separate the three components of a production: the *predecessor*, the *condition*, and the *successor*. A production has the format "*pred: cond* ®*succ"*.

A production in a 0L-system matches a module in a parametric word if the following are met:

- The letter in the module and the letter in the production predecessor are the same.
- The number of actual parameters in the module is equal to the number of formal parameters in the production predecessor.
- The condition evaluates to *true* if the actual parameter values are substituted for the formal parameters in the production.

A matching production can be applied to the module, creating a string of modules specified by the production successor. The actual parameter values are substituted for the formal parameters according to their position.

After a string of modules has been generated by our L-system, it is scanned sequentially from left to right, and the consecutive modules are interpreted as commands that maneuver a LOGO-style turtle in three dimensions just as in the traditional use of turtle geometry introduced in previous work [4,5]. The turtle is represented by its state, which consists of the turtle position and orientation in the Cartesian coordinate system. Changes in the turtle's state are caused by interpretation of specific modules, each of which may be followed by parameters.

In the modeling of plants, the amounts by which the parameters associated with modules are incremented and decremented follow normal distributions (e.g. the length of the stem is expected to decrease by 10%±5%, so the mean decrement is 10% and the mean square deviation is $5\%^2$). We used the Box-Mueller method to generate sample values of a random variable following a normal distribution.

The parameters for palmettos are as follows with a resulting plant illustrated in Figure 1:

- Separation angle (α): angle between the adjacent palmettos in a cluster
- Angle of stem curvature (β): angle between the first segment and the normal of the ground
- Angle of leaf separation (γ): angle between the adjacent leafs in a palmetto
- Angle of leaf position (δ): angle between a leaf and the average plane that leafs are on
- Number of palmettos in a cluster

The parameters for oaks are as follows with a resulting tree also illustrated in Figure 1:

- Number of side branches
- Angle of side branches (α): angle between the trunk and the side branch
- Angle of binary branches (β): angle between the binary branching stems
- Binary branch probability: probability with which a stem will split into binary stems instead of continuing to grow as a single stem

Figure 1. Palmettos and oak, plus modeling parameters

IV. AUTOMATIC LOD MODELING OF L-OAK

We refer to the original oak models produced by traditional L-systems as Lindenmayer models or P-models. The choice of the symbol P is intended to emphasize that these models are going to serve as our "ground truth" about the oak. This is the model upon which our LOD algorithm will be working.

In the P model, rectangular prisms model trunks, stems and twigs. Collectively we refer to these elements as sticks; they contain no bends. Sticks alternate with nodes that represent both bends in a limb, and branching-points. The initial stem is called the trunk. Normally it is drawn vertically. The final stems are joined to leaf clusters, which are transparent polygons bearing two-dimensional texture maps, with an image of appropriate color and texture to represent a cluster of leaves of the oaks being modeled.

We consider that the development from P_0 (the base model) to P_n (the n-th generation) represents the normal growth of the plant. In Figure 2, the first three generations are shown, with the sticks and nodes labeled in P_2.

Sticks in P_n are numbered $S_{n,0}$, $S_{n,1}$, $S_{n,2}$…, where the sticks $S_{n,e}$ and $S_{n,e+1}$ denote the outbound sticks from node $N_{n,\lfloor e/2 \rfloor}$. Thus, $S_{2,2}$ and $S_{2,3}$ are outbound from $N_{2,1}$. For simplicity, we have not shown them in the diagram of P_2, but there are four leaf clusters $C_{2,0}$ through $C_{2,3}$ attached to the sticks $S_{2,0}$ through $S_{2,3}$ like those in P_1 and P_2.

If P_n is not an idealized plant (that is, some of its nodes do not have two out-sticks), we still preserve the numbering system, which means that some of the number pairs <n, i>

have no corresponding sticks. This makes it easy to locate the parent node for any stick.

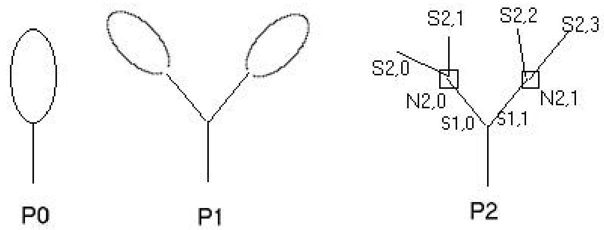

Figure 2. First three generations of P model

First, we define two parameters that are used in our algorithm. *Height* of a stick is the distance from the connection point between the stick and its parent stick to the ground (e.g. in Figure 3, Height of sticks $S_{2,2}$ and $S_{2,3}$ are the same. Actually all the sticks growing out of a same node have the same Height). *Radius* is the width of the stick. We refer to $\frac{Height}{Radius}$ as *HR* for each stick. Assume that Height of the initial stem of the oak is 0. Since the sticks become higher and thinner with the applications of productions, those produced in later iterations have greater values of *HR*.

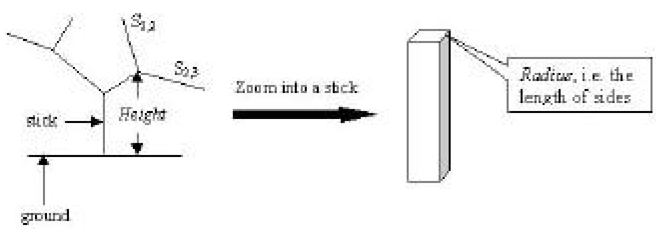

Figure 3. Height and Radius of a stick

Based on the basic P Models and the parameters above, we derive an R model, which incorporates levels-of-detail and is aimed to achieve three goals:

1. To reduce the number of polygons;
2. To keep the height of the oak;
3. To retain the coverage of the cluster.

Since our algorithm works separately with the sticks and the leaf clusters, the two distinct aspects of the algorithm will now be discussed: sticks and leaf clusters.

Sticks. Initially, we can have an ordered thresholds array of HR values, in which $0 < HR\_th_{[1]} < HR\_th_{[2]} < \ldots < HR\_th_{[n-1]} < HR\_th_{[n]}$. We use this array to divide all the sticks in a P model into $n+1$ sets according to their HR values. For each stick in the *i*th set, its HR is between $HR\_th_{[i-1]}$ and $HR\_th_{[i]}$, where $2 \le i \le n$. For those sticks in the 1st set, all of their HRs are smaller than $HR\_th_{[1]}$; For those sticks in the $(n+1)$th set, all of their HRs are greater than $HR\_th_{[n]}$.

Suppose that we want to set N levels of detail (R models) for the original model (P model) of an oak. First, we need to divide all the sticks into N sets following the above principle. Then, to build up R_1 (the model that is just less detailed than the P model), the sticks in the *N*th set should be replaced by their extended parents that are originally in the *(N-1)*th set; for R_2, the sticks in the *(N-1)*th set, which may have already been extended, will be further replaced by their extended parents that are originally in the *(N-2)*th set, and so on.

There is a more complex case. Assume as in Figure 4 that there are four sets of sticks divided by the array *HR_th[]*. According to $HR\_th_{[3]}$, sticks $S_{3,4}$, $S_{3,5}$, $S_{3,6}$ and $S_{3,7}$ will be replaced with the extended sticks $S_{2,2}$ and $S_{2,3}$. The children of stick $S_{3,7}$, i.e. $S_{4,14}$ and $S_{4,15}$, however, are below $HR\_th_{[3]}$, not in the same set as $S_{3,7}$. So in the first simplified model, they won't be replaced, resulting in $S_{3,7}$ not being replaced (to do otherwise would destroy the tree's continuity).

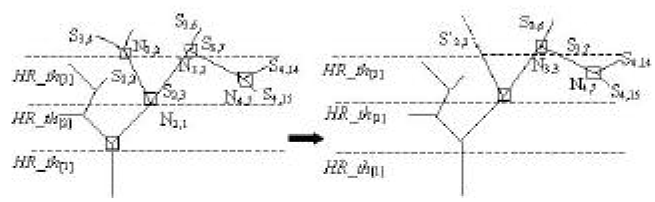

Figure 4. Complex case of recursive merge of sticks

Leaf Clusters. Suppose the centers of the original leaf clusters are O_i (x_i, y_i, z_i) (i=1, 2, …, n), which are the top ends of the latest iteratively generated stems, and the radii are r_i (i=1, 2, …, n). When the root stick of a subtree extends to replace all the other sticks, a merged leaf cluster, centered at the top end of the extended stick, will be generated to replace all the leaf clusters in the previous level, as in Figure 5.

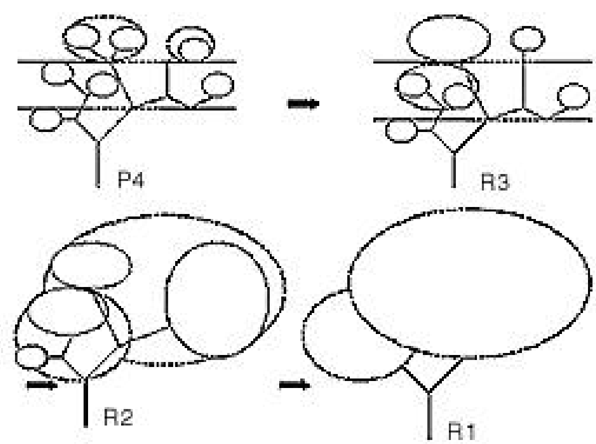

Figure 5. Recursive merge of leaf clusters

Suppose the radius of a merged cluster is R_0, which is at the top of an extended stick *S*. The radius should be derived from the radii of the original leaf clusters that are in the subtree rooted at *S*. In Figure 6, we have three leaf clusters and they will be merged into one. (x_0, y_0, z_0) and (x_i, y_i, z_i) are the centers of the top of the extended root stick and the tops of the sticks that originally have leaf clusters at their tops, respectively. dis_i is the distance between (x_0, y_0, z_0)

and (x_i, y_i, z_i). r is the average radius of the original leaf clusters. R_0 in Figure 6 can be derived by the formula below when n is assigned 3.

$$dis_i = \sqrt{(x_i - x_0)^2 + (y_i - y_0)^2 + (z_i - z_0)^2}\ ,$$

$$r = \sum_{i=1}^{n} r_i / n\ ;$$

$$R_0 = \sum_{i=1}^{n} dis_i / n + r\ .$$

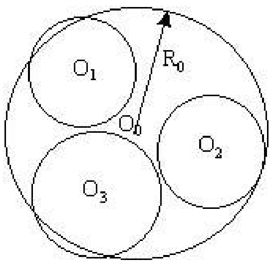

Figure 6. Radius of merged leaf cluster

The above establishes the basis for our level of detail generation. We now discuss the simulation and rendering of a natural scene with oaks and palmettos.

V. TEST ENVIRONMENTS

There are two components in the test environment. One is the simulation of a real scene and the other is an interface that experimentalists can use to do subject tests.

Below (Figure 7) is a snapshot of the simulated scene with a background of a blue and cloudy sky, and a texture-mapped ground, whose image is taken from the real scene.

From this scene we created a 180-degree-panorama (Figure 8) of the simulated scene as the background for the final simulated scene (Figure 9).

After the natural scene has been simulated, a fixed path is set for navigation through it. All the running results have been obtained based on the navigations along this fixed path.

The interface we designed is for use in the human performance experiments. Experimenters can use it to set and get parameters that influence LOD modeling and user navigation.


Figure 7. Simulated natural scene


Figure 8. Panorama of computer rendered scene


Figure 9. Final simulated scene

VI. CONCLUSIONS

The key contribution of this research is the development and implementation of a novel method of automatic LOD modeling of trees, specifically oaks. Although some previous work has been done on the LOD modeling of plants, most of these borrow techniques from the LOD modeling of non-plant objects. Unfortunately, such approaches usually ignore the growth characteristics of plants. The difference between our LOD method and others *first* lies with the fact that we have designed it based on the geometric structure of the tree species (oak) and made use of those characteristics in our implementation of the recursive algorithm. *Second*, our method is more easily user controlled, which is rare in other LOD modeling of plants. One of the goals that we want to achieve with this automatic LOD modeling is to examine the impact of the LOD parameters on human perception and performance. The specific parameters that we have focused on are LOD representations and ranges. Driven by this goal, we have tried to design an intuitive and controllable method that is easily operated by not only computer scientists but also other researchers such as psychologists. Although the advantages of our LOD methods in the context of such a user interface need to be further verified by performance tests, we believe this work represents an important exploration into the interdisciplinary area of advanced graphics rendering techniques and psychological (human performance) experiments.

Our experimental results support the basic principle in computer graphics that the frame rate increases with a decrease in the number of geometry primitives. One particular experiment we ran was a comparison between two schemes of geometry reduction, "50%-25%-12.5%-12.5%" versus "25%-25%-25%-25%". The meaning of the first of these is that we should remove half the geometry to produce the first reduced level-of-detail, 25% more (75% total) for the second, and so on until all geometry is gone and only a texture remains. The second scheme decreases geometry more slowly, specifying a uniform reduction at each level.

Our observation was that the first scheme improved performance over the second, as measured by frame rate, by between 15% and 20%. On the other hand, the visual discontinuities ("pops") in renderings were slightly worse in the first scheme. This was particularly true when we represented the oaks by both sticks and leaf clusters versus just sticks (trees that have lost all their leaves). The poorer performance of the former is caused by the fact that the size of the 2D recursively merged leaf cluster is *approximately* computed and cannot precisely replace multiple leaf clusters positioned in 3D space. A solution that we have employed to relieve the jarring feeling brought about by pops is to use a panorama as the background of the scene (Figures 8 and 9). This works quite well, effectively removing perceived visual discontinuities, when the panorama is formed from a snapshot of the model. The technique was much less effective when we used a photograph from the real-world scene as the basis for the panorama.

VII. Future Work

There are two aspects of this work that our group plans to pursue in the near future. First is the modeling of more species. So far we have modeled one species each of oak and palmetto. Next we will model pine trees, which are usually tall with sparse foliage. These characteristics mean that pine trees do not heavily occlude other plants; that is, in fact, why we were not concerned with pine trees in our prototype.

Our second objective is to implement performance tests for human subjects using the virtual scene. In these tests we want to discover an "ideal" scheme of sticks reduction in our LOD mechanism, so that the simulated natural scene has the same "hiding power" as the real scene. We will do some pilot tests with pictures taken from the real scene and static scenes produced by our program. Subjects will be asked to find prescribed targets and then approximate the distance between the viewpoint and the targets. The purpose of these tests is to exam differences between the perceptions of the virtual scene and the real scene under static conditions. Such differences depend on occlusion, relative size and relative density. After this, we will expand our experiments to include complex tasks such as search-and-rescue that depend on dynamic conditions associated with traversing the scene, first through pre-planned paths and later with freedom of movement, constrained only by the walkways of the natural scene.

VIII. References

[1] R. Smith. "Plants, fractals, and formal language." *Computer Graphics* 18:3, pages 1-10, 1984.

[2] P. Prusinkiewicz and A. Lindenmayer, etc. *The Algorithmic Beauty of Plants*. Springer-Verlag, New York, 1990.

[3] P. Prusinkiewiczv and J. Hanan. "Visualization of botanical structures and processes using parametric systems." In D. Thalmann, editor, *Scientific Visualization and Graphics Simulation*, pages 183-201. J. Wiley & Sons, Chichester, 1990.

[4] P. Prusinkiewicz. "Graphical applications of L-system." *Proceedings of Graphics Interface '86 – Vision Interface '86*, pages 247-253,1986.

[5] A. L. Szilard and R. E. Quinton. "An interpretation for DOL systems by computer graphics." *The Science Terrapin* 4, pages 8-13, 1979.

[6] J. Weber and J. Penn. "Creation and rendering of realistic trees." *SIGGRAPH 95 Conference Proceedings*, pages 119-128, August 1995.

[7] Thomas A. Funkhouser and Carlo H. Sequin. "Adaptive display algorithm for interactive frame rates during visualization of complex virtual environments." *Computer Graphics Proceedings*, pages 247–254, August 1993.

[8] B. Watson, N. Walker, L. Hodges, A. Worden. "Managing level of detail through peripheral degradation: effects on search performance with a head-mounted display." *ACM Transactions on Computer-Human Interaction* 4:4, pages 323-346, 1997.

[9] B. Watson, N. Walker, L. Hodges and M. Reddy. "An Evaluation of level of detail degradation in head-mounted display peripheries." *Presence* 6:6, pages 630-637, 1997.

[10] M. Reddy, N. Walker, B. Watson and L. Hodges. "Managing level of detail in virtual environments: A perceptual framework." *Presence* 6:6, pages 658-666, 1997.

[11] F. Law, T. Tan. "Preprocessing occlusion for real-time selective refinement." *Symposium on Interactive 3D Graphics*, pages 47-53, 1999.

[12] D. Cohen-Or, G. Fibich, D. Halperin and E. Zadicario. "Conservative visibility and strong occlusion for viewspace partitioning of densely occluded scenes." *Computer Graphics Forum* 17:3, pages 243-253, 1998.

[13] B. Nadler, G. Fibich, S. Lev-Yehudi and D. Cohen-Or. "A qualitative and quantitative visibility analysis in urban scenes." *Computers & Graphics* 23, pages 655-666, 1999.