Urban Ecosystem Design

Bedřich Beneš* Michel Abdul Massih Philip Jarvis Daniel G. Aliaga Carlos A. Vanegas
Purdue University

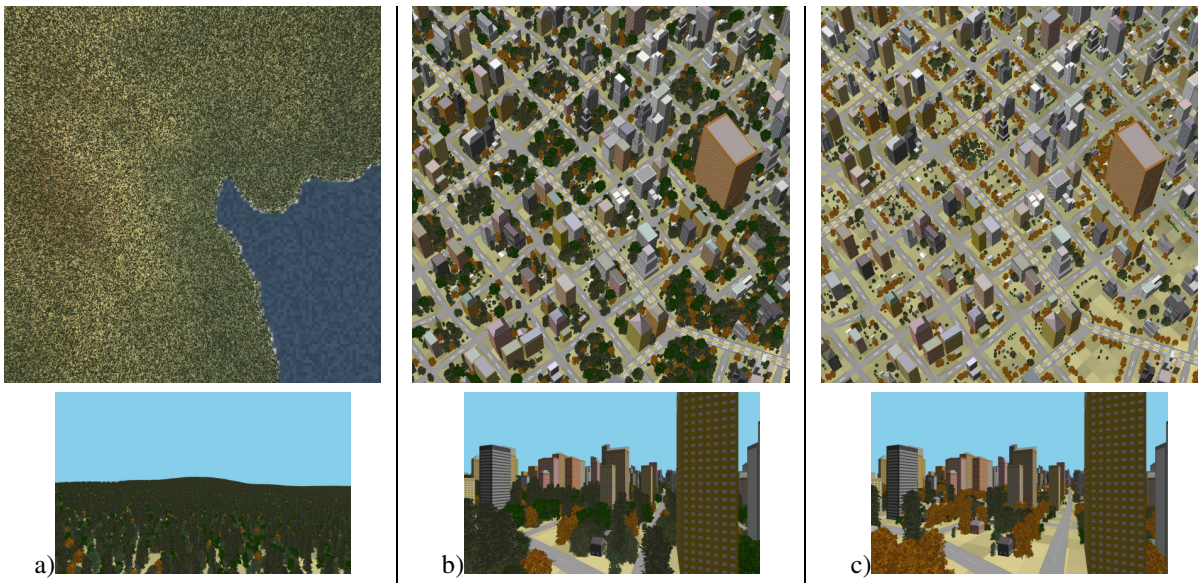

Figure 1: *This example demonstrates the need for urban ecosystems. The image in a) shows a terrain occupied by a wild ecosystem and b) displays the same ecosystem grown over the city layout, where the vegetation invades all areas and attempts to fill them entirely and the ecosystem is chaotic with no control. The image c) shows the managed urban ecosystem that has areas with wild plant growth but also areas controlled by our plant management system.*

Abstract

We address the open problem of spatial distribution of vegetation in urban environments by introducing a user-guided simulation and procedural system for integrating plants into the interactive design process of 3D urban models. Our approach uses as input 3D geometry of an urban layout from which it infers initial conditions and parameters of procedural rules. A level of manageability is calculated for each area of the urban space. The manageability level defines the amount of influence between the wild ecosystem simulation, where the plants compete for resources and seed freely, and the managed ecosystem, where nearly no seeding is allowed and the plants grow only under well-defined conditions. The wild ecosystems are handled by a simulation of plant competition for resources, whereas the procedural generation is based on an expandable set of behavioral rules of owners and typical plant management. Our system provides an interactive semi-automatic method to calculate a spatial plant distribution and to create an urban model with plants covering an area of several square kilometers in less than a minute. It provides a high degree of controllability and works tightly with an urban simulation system. We show various examples, such as plant development over time in managed and unmanaged areas, effect of procedural rules on the plant distribution, and the effect of changing the level of manageability and the plant distribution.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Geometric algorithms, languages, and systemsI.3.6 [Computer Graphics]: Methodology and Techniques—Interaction techniques

Keywords: virtual ecosystems, urban simulation, procedural modeling

*e-mail: bbenes@purdue.edu

1 Introduction

We present an interactive procedural system for integrating plants into the design process of 3D urban models. Currently, in 3D urban modeling vegetation is generally not the main focus, despite it serving as an important visual cue and a necessary background. If plant models (i.e., trees and bushes), and their arrangement, are not generated with proper quality, they can rapidly trigger unwanted attention and cause severe visual disruption. A variety of methods exist for plant generation that range from the modeling of individual plant organs to large plant populations. Nonetheless, plant modeling is very complex and thus their use in an interactive modeling process is not common.

Several methodologies have separately been proposed for the modeling of urban areas and for the modeling of plants. On the one hand, urban modeling algorithms exploit procedural techniques [Merrell and Manocha 2008; Müller et al. 2006; Parish and Müller 2001], integrate with urban simulation processes [Aliaga et al. 2008a; Vanegas et al. 2009a; Vanegas et al. 2009b], or use

example-based synthesis [Aliaga et al. 2008b; Merrell 2007] to create models of entire cities. Further, various individual aspects of urban model authoring have been also addressed in more detail, such as façades [Müller et al. 2007], road generation [Galin et al. 2010], and even tourist maps [Grabler et al. 2008]. On the other hand, various techniques and algorithms for the generation of ecosystems have also been published. In addition to various procedural techniques based on L-systems [Prusinkiewicz and Lindenmayer 1990], one of the most interesting is an automatic spatial plant distribution algorithm based on plant competition for resources [Lane and Prusinkiewicz 2002]. The algorithm populates large areas with widely distributed plants of different species [Deussen et al. 1998; Lane and Prusinkiewicz 2002]. Typically, there is a little control of where and how plants appear and grow. However, in a city plants are not allowed to grow wildly, residential areas are repeatedly pruned, trees are planted at exact locations and protected from damage, only certain species are allowed to grow around roads, etc. To our knowledge, creating a framework for the interactive design of plant ecosystems in urban areas has not been addressed by previous works, despite the visual importance of including plants in 3D city models. A fast, easily controllable, and intuitive integration of plant and urban layouts remains an open problem.

The key observation behind our design approach is that plants in urban areas are not distributed arbitrarily but are affected by the structure and inhabitants of the city. Urban plant ecosystems can be interpreted as being *managed* directly by the city residents and employees and indirectly by the geometry of the urban surroundings. The management of the plants can be expressed as a set of procedural rules for use by an automatic algorithm and for use in an interactive design process. For example, plants around the roads usually form alleys, trees are planted between buildings and roads, residential houses usually have large lawns in the backyards, skyscrapers are surrounded by trees, and so forth. Using a provided urban geometry, we can (automatically) infer the aforementioned rules and then use these rules in an algorithm for generating a plant distribution. Additional control and flexibility of the design process can be achieved by user-specified modifications of the rules and their parameters. Moreover, quick and automatic plant generation can be accomplished by a GPU-based plant distribution algorithm.

We present an interactive solution which simultaneously designs a 3D urban model and solves the open problem of generating a plausible spatial plant distribution in the same urban area (Figure 1). Our algorithm works in a closed loop fashion with an interactive urban modeling system (e.g., we use [Vanegas et al. 2009b] but other systems could be used) and adds a component for generating a spatial distribution of managed and unmanaged (i.e., wild) plants. Users can interactively alter multiple parameters of the urban model and plant model creation process, each time resulting in a new 3D model. In [Vanegas et al. 2009b], the urban model creation process is controlled by user-specified alterations to the distribution of population, jobs, and roads as well as by geometry and style parameters. The plant model creation process uses a set of procedural rules to calculate an initial plant distribution and then a competition for resources algorithm is executed to produce a final plant distribution for use in the city model. Optionally, the parameters of the procedural rules can be edited via a simple user-interface.

To enable the aforementioned plant creation process, our method automatically extracts semantic information from the geometry and calculates a level of plant manageability for different areas of the city. Each city block is assigned a level of plant manageability. Expensive areas are fully controlled, whereas low-cost areas are mostly influenced by plant competition growth. For instance, areas such as parks , the vicinity of roads and arterials, and backyards, are assigned a high level of manageability. These areas are seeded by grown trees, similar to what city gardeners or house owners do. Further, they are protected from the influence of other plants, old trees are replaced, and no other plants are allowed to grow there. The plants in areas with low manageability are governed by a competition for resources algorithm - they approximate a wild ecosystem. On the boundary of high and low manageability areas, the wild ecosystems attempt to invade the managed areas, and the managed vegetation can freely send their seeds into the wild ecosystems.

We show various examples, such as plant development over time in managed and unmanaged areas, effect of procedural rules on different areas, and the effect of changing the level of manageability.

2 Previous Work

Urban layout generation can be roughly classified into two main categories: procedural modeling and simulation-based modeling. One of the first papers for a procedural urban layout generation is the seminal work of [Parish and Müller 2001] where the street layout was generated by Open L-systems [Měch and Prusinkiewicz 1996], blocks and parcels were generated from the street graph, and the parcels were completed by procedurally generated buildings. Procedural generation of street layout using tensor fields was introduced in [Chen et al. 2007] and a technique for interactive editing of existing urban layouts was introduced in [Aliaga et al. 2008a]. Complete buildings can be generated by CGA, a procedural model introduced in [Wonka et al. 2003]. In our previous work we have presented an approach for creation an urban layout by procedural completion of urban layout examples [Aliaga et al. 2008b]. One of the principal disadvantages of the procedural models is their low controllability. The pure procedural models can be controlled by defining the generating rules, the example-based techniques rely on the input data, but their combination is again controlled only at a very high level. Semi-interactive techniques, such as [Lipp et al. 2008], allow for a high level of control, but the building generation can take very long time.

The second class of algorithms for urban model generation is based on urban simulations. Two computer graphics approaches presented so far include [Vanegas et al. 2009b; Weber et al. 2009] which use simplified rules for jobs and population distribution to generate underlying urban layouts. Secondary values, such as zone accessibility, land value, amount of people commuting to their jobs, are provided and these values are used to generate streets, blocks, parcels, and buildings. The system presented in this paper uses our previously published system for urban model generation by simulation [Vanegas et al. 2009b]. However, in order to provide a general-purpose method that is not tied to a single system, our system uses only urban geometry and can be seamlessly merged with virtually any city generation application (e.g., CityEngine).

Techniques for the modeling and visual simulation of plants and plant ecosystems have been presented for nearly thirty years. The algorithms can be classified according to their level of detail. On the lowest level are the algorithms for individual plant organs [Zhang et al. 2006], or even cellular subdivision [Lindenmayer 1968]. These techniques are typically used for detailed close-ups. Our focus is a large area of the city that leaves these algorithms out of the scope of this paper.

Entire plants can be generated by sketching [Chen et al. 2008; Ijiri et al. 2006b; Ijiri et al. 2006a], interactively [Lintermann and Deusen 1996], by procedural techniques [Prusinkiewicz and Lindenmayer 1990; Měch and Prusinkiewicz 1996], by a biological simulation [Benes and Millán 2002; Palubicki et al. 2009], or by hybrid methods [Benes et al. 2009]. In our implementation we aim to model thousands of plants and thus their individual modeling is not a viable option. Because of this limitation we have decided to

use a fixed number of predefined plants in different stages of development and each with various levels of geometric detail. Our trees were generated using the XFrog system [Lintermann and Deusen 1996; Lintermann and Deussen 1999].

Techniques dealing with plant ecosystems focus mostly on the spatial plant distribution in the scene. Existing works [Alsweis and Deussen 2005; Deussen et al. 1998; Lane and Prusinkiewicz 2002] focus on the idea of plant competition for resources. Chances of a plant surviving vary according to its age, specie, and local resources. If more plants appear in the same area they compete for resources and some plants survive whereas others die. Entire plant ecosystems are simulated over long periods of time (e.g., tens and hundreds of years) and the result is the spatial distribution of the plants and a record of their evolution over time. The ecosystem itself tends to reach equilibrium but it can be disturbed by external influences such as gardeners presented as autonomous agents [Benes and Espinosa 2003]. Even though these approaches are based on simulation, they share a common problem with procedural methods, namely they suffer from low controllability. In our work, we introduce a new algorithm that exploits the information of an urban layout and provides varying levels of controllability over the spatial plant distribution. Some areas, such as shores, or borders of the city, are kept entirely wild and controlled by the plant competition. Other areas, such as parcels and downtown, are controlled by a set of procedural rules. The continuous mixture of both techniques together with the values derived from the urban model enable a high level of control and provide a fast design tool for the spatial plant distribution of an urban ecosystem.

The paper continues with the description of urban ecosystems. Afterwards, Section 4 describes implementation and the next section discusses example applications and results. The last section 6 concludes the paper with thoughts about limitations and future work.

3 Urban Ecosystems

3.1 System Overview

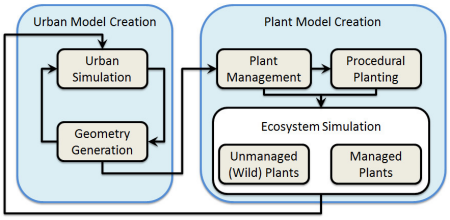

Figure 2: *Spatial plant distribution is generated for a given urban layout that is provided by the system on the left. Plant management computes a manageability value for each city block and procedural planting uses this value to seed plants. An ecosystem simulation is then used to develop managed and unmanaged (wild) plants in the city by simulation of plant space colonization and competition for resources.*

Our method for the interactive design of urban ecosystem consists of two main processes, as shown in Figure 2. Although any system which enables quickly producing 3D urban models is suitable, our urban model creation process is largely based on the work of [Vanegas et al. 2009b] and we include its brief summary for completeness. The input data to this process includes an initial spatial distribution of jobs and population as well as the terrain and the main highways. The process executes a socio-economical and geometrical simulation in order to quickly generate a city model: road network, city blocks, parcels, building envelopes (i.e., just the exterior of buildings), a new distribution of population and jobs, and land value estimates. The created geometry of the 3D urban model is passed to the second main process.

The plant model creation process provides a way to generate a distribution of plants in the city. The process can be used to re-create an urban plant distribution after any change to the urban model or to simulate the growth of the plants over time for a given (fixed) urban model. In both cases, the computed distribution specifies the locations and ages for plants of several species. With the creation process, an ecosystem simulation is run for many iterations that correspond to tens or hundreds of years of the ecosystem in order to reach a "stable" configuration. The plant creation process is subdivided into the following three components.

1. A *plant management algorithm* infers from building, city block, and street geometry the level of manageability of each city block. On one extreme of the continuously-valued manageability level are areas with no control at all, such as areas near the border of the city, that obey the rules of an entirely wild ecosystem [Deussen et al. 1998]. On the other extreme of the spectrum are areas deep within the city limits, such as green zones around skyscrapers in the downtown area, which have a highly controlled ecosystem. The areas in between these two extremes provide ecosystems, such as parks, backyards, or areas around roads, which have some amount of plant manageability.
2. A *procedural planting algorithm* uses the manageability levels, the city geometry, and an expandable set of procedural rules to spread plant seeds. Highly managed areas are seeded (i.e., virtual plants are grown from their seeds) in a very controlled fashion [Sukopp et al. 1990]. Areas with low or no manageability over the plant distribution are seeded with random plants.
3. An *ecosystem simulation* based on competition for resources is executed. In each step of this simulation, the algorithm checks plants for seeding, eliminates old plants, determines the winners of plant competition and calculates the fate of the losers. This algorithm is extended beyond its basic formulation by also taking into account managed plants that are not significantly modified. For example, if a managed plant is old and should be eliminated, then it is replaced by the same species but younger. Competition between managed and unmanaged plants is also considered.

The user can optionally control all steps of the plant creation process. Further, the resulting plant distribution is converted to a large set of 3D tree models which are added to the urban model and rendered interactively using level of detail techniques.

3.2 Plant Management

The plant management algorithm determines for each city block (i.e., a collection of parcels and lots closed by streets into a logical urban entities) a manageability level of the contained plants. The block *manageability level* $0 \leq m_i \leq 1$ is a normalized value describing how well the owners of the city block take care of the contained plants - often, it is related to the value of the land. For city blocks containing very high buildings, such as skyscrapers, we set $m = 1$. We define such city blocks as those containing buildings whose height is within the top 10% of the range of building heights in the city. For all other city blocks, the manageability level is computed as a function of city block size and of the size of buildings/houses within the block. More precisely, a block's manageability level is defined as

$$m_i = w_b b'_i + w_e(1 - e_i) \quad (1)$$

where w_b and w_e are user-specified weights (e.g., $w_b = w_e = 0.5$ in our experiments) and $0 \leq w_b, w_e \leq 1$ and $w_b + w_e = 1$, b'_i is the normalized height of the buildings in the block, and e_i is the effective area of the block

$$e_i = \frac{1}{a}\left(a - \sum_j b_j\right), \tag{2}$$

where a is the area of the block and $\sum b_j$ is the area of all the buildings in the block.

Intuitively, plant manageability is high in areas with high values of m. Areas with $m = 1$ are not allowed to contain any wild plants whereas areas with $m = 0$ do not contain any plants that are managed. The manageability level value for other areas determines the percentage of wild plants allowed in each such area and that will be used in the later described ecosystem simulation. For nonzero values of m, the ecosystem simulation will ensure a proper number of wild plants are maintained in the city block (e.g., simulating plant growth and dispersion as well as the management task of removing unwanted plants). By default, urban areas that do not belong to any city block are seeded by the wild ecosystem.

Figure 3 shows an example of an urban layout, where the manageability is changed from a small number of blocks on the left, to high manageability on the right. As the result, with increasing manageability the number of wild ecosystem zones decreases. Over-controlled ecosystem leaves few spots for parks and recreational areas. The area of 3×3 km includes about $250,000$ plants and was simulated in 30 seconds.

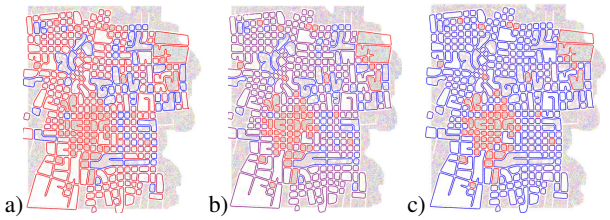

Figure 3: *To show the effect of the level of manageability, blocks with high manageability have blue borders with decreasing manageability turning red. Fully managed blocks with $m = 1$ are covered with plants entirely. a) Low level of manageability leaves many areas dominated by wild ecosystems. b) A balanced urban ecosystem. c) An over-controlled urban ecosystem with few parks in the middle and most of the zones fully controlled.*

3.3 Procedural Planting

Figure 4: *All plant species used in our system.*

Theoretically, each individual plant could have manual manageability. However, controlling thousands of plants would be overwhelming. Instead we control groups of plants inside individual city blocks. Once the manageability level of each city block is determined, plants must be seeded (i.e., planted) prior to executing the (controlled) ecosystem simulation. There are two primary approaches for seeding a plant. The plant can be seeded i) by another plant or ii) by a set of procedural rules which imitate managed planting by the city's inhabitants. Seeded plants that are not managed straightforwardly lead to wild ecosystems, whereas managed plants are seeded once using our procedural rules and then managed during the ecosystem simulation. In the following, we describe our initial set of procedural roads that algorithmically implement plant management tasks that result from observing urban spaces [Sukopp et al. 1990]. Our proposed rules can be applied to a variety of urban configurations but the rule set is clearly extensible.

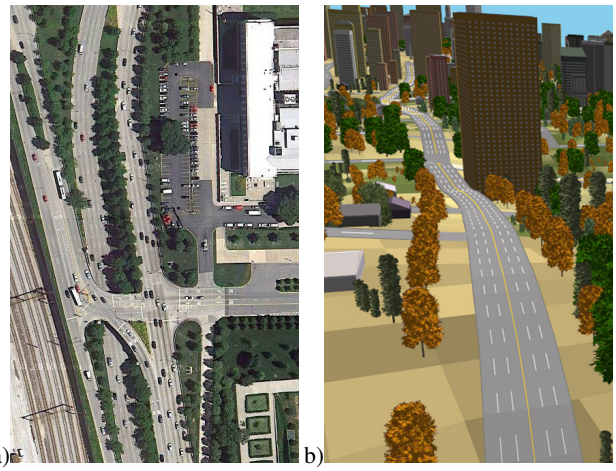

Figure 5: *Alleys formed around roads and arterial streets in a) aerial photograph of a neighborhood in Chicago, and b) our simulated neighborhood.*

Roads and arterial roads are often accompanied by trees on their sides as can be seen in Figure 5. The road classification is provided by the urban simulation module, but could be also calculated from the road geometry using algorithm from [Aliaga et al. 2008b]. Our system plants trees procedurally around a user-controlled percentage of the main roads. The plants are fully controlled, no wild trees are allowed, and trees are distributed in jittered distances around the main axis of the road.

Blocks are classified and planted according to the number of buildings. For residential blocks with a single building, the side of the building closest to the road is assumed to be the front side of the building. Several trees are planted between the road and the front side as people like privacy in their residences. Similarly, the opposite side of the block is seeded with a jittered row of plants since people also prefer to have privacy in their backyard. The area between the house and the end of the backyard is left empty as a lawn.

In blocks with multiple houses, our method automatically calculates the main axis of the block and seeds several plants along the main axis. This is a common pattern of typical US cities where people tend to separate their backyards from their neighbors. The rest of the area is occupied per building in the manner analogous to the case of a single house. Figure 6 compares a real neighborhood to a simulated neighborhood produced by our system.

High-value blocks usually correspond to the zones with the highest buildings in the city. These buildings usually occupy a significant part of the block. Further, as can be seen in Figure 7, the non-occupied areas are typically filled with dense plants. We detect these zones as blocks with $m = 1$. In these areas we apply regular plant distribution over the available zones using random jittering.

Finally, *parks* are simulated as nearly wild ecosystems with several managed plants that define their overall appearance and cannot be

Figure 6: *Blocks with multiple houses usually have row of trees that separate the buildings. Each house can have some trees in front to hide the house from the street as shown in a) which is an aerial photograph of West Lafayette, IN and b) is our simulated neighborhood.*

changed. The dominant trees are situated near streets in order to protect the park visitors and the park interior. Further, a large open area inside the park is protected from plants.

Figure 7: *a) Blocks with the highest land value, such as this part of Manhattan, are efficiently filled with green areas. The image in b) shows our simulated neighborhood.*

3.4 Ecosystem Simulation

The rules controlling development of plants are inspired from those of a wild ecosystem and are based on symmetric and asymmetric plant competition, similar to that described in [Alsweis and Deussen 2005] and [Deussen et al. 1998; Lane and Prusinkiewicz 2002]. However, in our system we extend the standard ecosystem simulation to consider managed plants. In particular, the simulator attempts to erase in regular intervals excessive amount of wild plants in controlled blocks, similarly to a housekeeper or to city-wide plant management. The amount of eliminated plants depends on the desired plant density. We have determined experimentally the spatial density of plants in the wild ecosystem as $\bar{d} = 0.65$ plant/m^2. In each iteration we calculate the number of plants p_i in the $i-$th block and calculate its the plant density as $d_i = p_i/e_i$, where e_i is the effective area of the block (2). The normalized plant density in the block is $d'_i = d_i/\bar{d}$. The simulator compares the normalized plant density with the desired level of manageability m_i. If the density is higher, it randomly eliminates the excess of plants. It is important to note that if the density of a block is lower than this desired value, we do not add new plants.

The ecosystem simulation represents each plant by its field-of-neighborhood (FON) and computes the interactions between FON's over time, as well as random and controlled seeding. The FON is a circular zone of influence with the plant in its center. The FON's radius depends primarily on the plant size; further, as the plant develops its FON grows as well. When two FONs collide, the *viability* of each plant involved in this collision is evaluated to determine their survival. The viability function $v_x(t)$ of the plant x at age t determines the fate of each plant in the collision. The smaller, weaker, and more frequent plants have a higher chance of elimination than the larger, stronger, and less frequent plant species. We simplify the viability function to:

$$v_x(t) = \begin{cases} p_a t & \text{if } t < 0.5 \\ p_a & \text{otherwise} \end{cases}$$

where the t is the normalized plant age and p_a is the inverse statistical distribution of the plant in the ecosystem. Further, we define

$$p_a = 1 - \frac{\eta_a}{\sum_i \eta_i}$$

where η_i is the number of occurrences of plant species i. In other words, p_a increases when the plant is not present in the ecosystem and therefore increases the plant's chances for survival in a collision. This is a simple global control that protects plant species against extinction.

Each plant seeds in the fall of each year. Seeds are planted around the plant's FON in a random circular area of approximately 2-3$\times$ the FON's radius. Seeding leads to clusters of similar plants as can be seen in Figure 8 that provides visual plausibility that could not be achieved only with random seeding. The example shows 2D spatial distribution of the plant ecosystem after 25, 75, 100, and 125 years and the number of plants was around 2,000. The overall time of the calculation was 35 seconds with $\Delta t = 100$ days.

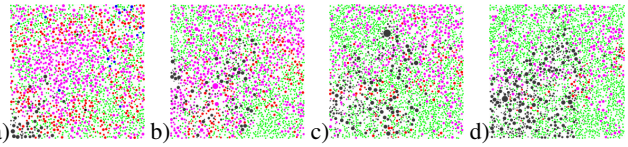

Figure 8: *Four frames of the spatial plant distribution generated by the wild ecosystem competition for resources. The area of 250×250 m is occupied by seven different plants with a cluster of dark plants seeded in the lower left corner. The figures show the development of the plant ecosystem after a) 25, b) 75, c) 100 and d) 125 years. Plants tend to exhibit emergent clustering phenomena that would be difficult to achieve by random seeding.*

In our method, a special case is a collision between managed and unmanaged plants. If an unmanaged plant and managed plant collide, the unmanaged plant is always killed. If two managed plants or two unmanaged plants collide, the previously-described algorithm calculates their viability. This competition provides a smooth transition between the boundary of a managed and an unmanaged areas.

4 Implementation

Our system is implemented in C++ with support of OpenGL for rendering and CUDA for ecosystem simulation. We have tested all examples on an Intel i7 920 CPU at 2.67 GHz. The computer was equipped with NVidia GeForce 480 with 1.5GB of memory.

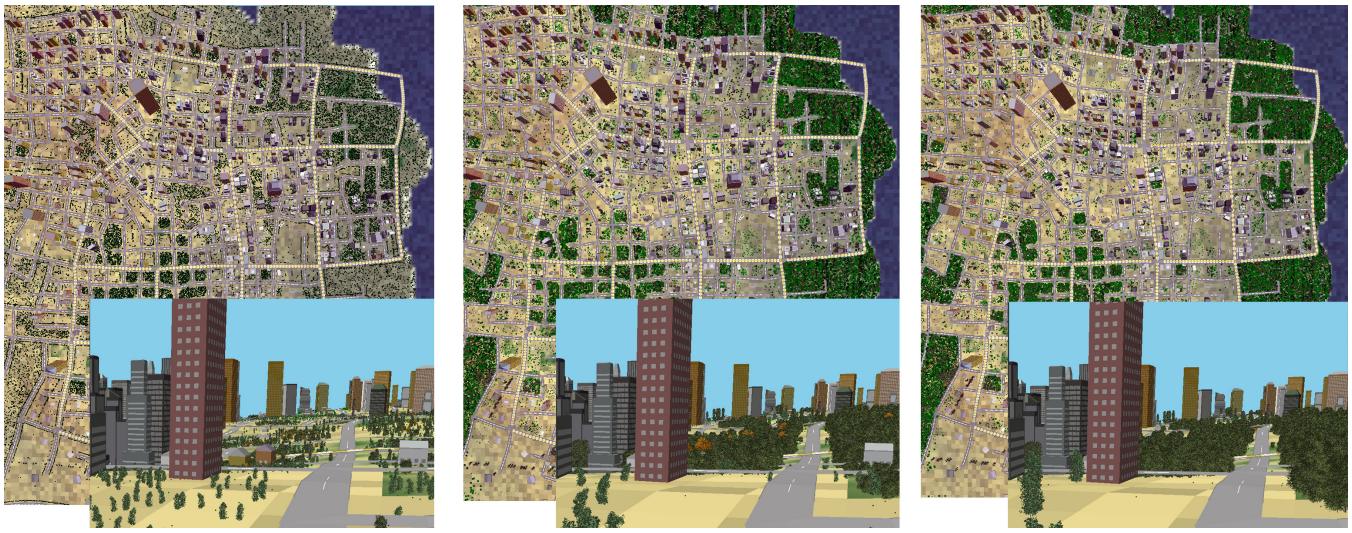

Figure 9: *An example of an urban ecosystem development over time, starting with image a) until image c). The insets (from left to right) show young and small trees becoming larger trees that eventually get removed by the plant management. The cluttered areas in the back are parks.*

As we aim to model thousands of plants where no significant detail will be displayed, we have decided to use a fixed number of pre-generated plant models in different stages of development and with seven levels of detail. Our trees were generated in the XFrog system [Lintermann and Deusen 1996; Lintermann and Deussen 1999] and the level of detail were generated using XFrog Xtune. We use seven different plants that are shown in figure 4 each in three different stages of development. In order to provide a "continuous" transition of the plant development, we quantize its life span into three intervals. We start with the smallest model which we continuously age until it is replaced by a next existing stage.

The most time-demanding operation is the collision detection of plant's FON together with the viability evaluation. As this task is evaluated for hundreds of thousands of plants and is essentially the same procedure called repeatedly, it was a great candidate for parallelization in CUDA. The CUDA kernel loads the plant locations as a 1D array, the city footprint as a 2D texture, and outputs a flag for each plant if it survived the test. The kernel first checks all plants that are seeded or procedurally initialized at incorrect locations such as roads, or buildings. This is achieved by a texture lookup into the city footprint and is provided very quickly. In this way, we make sure the plants do not collide with urban geometry. The plants are sorted into overlapping bins and we check for the collision and viability of each plan in the bin. As each plant is checked against each plant in the bin, it has a theoretical complexity $O(n^2)$. However the high plant density and the subdivision into bins helps to speed this test significantly, it allows to keep the GPU busy, and it avoids OS time-out problems with the graphics driver. As each kernel runs evaluation only for a single plant it writes only once at the end into the main memory, so the memory writes are without bank collisions. We have achieved a performance of 50M-70M collision tests per second allowing for 250,000 plants being tested at 5-6 fps.

The visualization engine implements kd-tree subdivision of the space with view frustum culling and automatic level of detail (LOD) selection. Each plant has seven LOD levels generated with XFrog Xtune. The plants are preloaded and instantiated on the GPU.

5 Results

Figure 1 justifies the need for managed plant design. The first figure shows a wild ecosystem. In the second figure an urban layout has plants distributed naïvely using just the plant competition algorithm. This results in all available areas being invaded with all kinds of plants wildly competing for resources with no consideration of other urban layout aspects. The last image shows the same urban layout but with plant management. Expensive areas have their plants managed better than parks and areas with low land value. The city border is still controlled by the wild ecosystem simulation as well as some areas close to the shore.

The example in Figure 9 shows an urban area of 3×3 km populated with $250,000$ plants that shows the plant distribution development over time. The left image shows mostly new trees that quickly occupy the allowed areas. The trees grow as can be seen in the image insets from the ground perspective. The sequence of images demonstrates the changes in urban ecosystem as a city evolves.

The second sequence in Figure 10 shows a case for altering city geometry. Starting from the same urban area and the same plant distribution of Figure 9a), the city geometry has been slightly modified and the urban ecosystem has been recalculated correspondingly. The simulation time of each examples from Figures 9 and 10 was about 90 seconds for the urban layout and 120 seconds for the urban ecosystem. The ecosystem in the latest stages shows plants with an age of up to 70 years. The simulation step $\Delta t = 1$ month.

Figure 11 shows changes of the plant spatial distribution due to changes in procedural planting. The procedural parameters have been modified to produce an alternate spatial distribution under user-control.

6 Conclusions and Future Work

We have presented an interactive simulation and procedural system for integrating plants into the design process of 3D urban models. Our approach uses as input 3D geometry of an urban layout. It infers initial conditions and parameters of procedural rules and calculates the level of manageability for different areas. This pa-

Figure 10: *The city from Figure 9a) where geometry has been changed to that in (top) and then (down) and the plants adapted to the new city.*

rameter defines the amount of influence between the wild ecosystem simulation, where the plants compete for resources and seed freely, and the managed ecosystem, where nearly no seeding is allowed and the plants grow only at strictly defined locations. The wild ecosystems are handled by a simulation of plant competition for resources, whereas the procedural generation is based on an initial set of behavioral rules of owners and on plant management of typical US cities. Our system enables designing, distributing, and outputting a 3D model contains plants and urban structures in an average area of 10 square kilometers in less than a few minutes, offering a high-level of controllability, and working tightly with an urban simulation system. We have shown various examples, such as plant development over time in managed and unmanaged areas, the effect of procedural rules on different areas, and the effect of changing the level of manageability.

Our approach is not without limitations. First, there is a set of rules that must be defined by the system designer. We use a fixed set of embedded procedural rules in our implementation. In order to address this limitation, an open system which would allow user-defined rules should be implemented. Second, our choice of rules is based on our personal preference. We sought to demonstrate the concept of merging urban design and plant design. In practice, different rules are probably appropriate for different cities. Thus, the way to select rules needs to be improved.

Several avenues exist for specific future work. The plant management is based on the derived land value. Nonetheless, the urban simulation provides job distribution, land value, and accessibility that could be use for a more sophisticated derivation of plant distribution. Another work would include the development of an interactive design system, where all the above described rules would be implemented as procedural brushes or interactive design tools. Our system works on the level of entire plant models. However, real plants interact with each other on a much finer level, so future work should focus on simulation of collision of tree branches, interaction with buildings, illumination and other environmental aspects.

Acknowledgements

We would like to thank NVIDIA for providing free graphics hardware, Oliver Deussen for free version of XFrog, and Ondřej Šťava for help with CUDA. This work has been supported by NSF *IIS-0964302*, NSF *OCI-0753116*, and Adobe Inc. grant *Constrained Procedural Modeling*.

References

ALIAGA, D. G., BENES, B., VANEGAS, C. A., AND ANDRYSCO, N. 2008. Interactive reconfiguration of urban layouts. *IEEE Computer Graphics and Applications 28*, 3, 38–47.

ALIAGA, D. G., VANEGAS, C. A., AND BENES, B. 2008. Interactive example-based urban layout synthesis. *ACM Trans. Graph. 27*, 5, 1–10.

ALSWEIS, M., AND DEUSSEN, O. 2005. Modeling and visualization of symmetric and asymmetric plant competition. *Eurographics Workshop on Natural Phenomena*, 1–11.

BENES, B., AND ESPINOSA, E. D. 2003. Modeling virtual ecosystems with the proactive guidance of agents. *International Conference on Computer Animation and Social Agents*, 126.

BENES, B., AND MILLÁN, E. 2002. Virtual climbing plants competing for space. In *IEEE Proceedings of the Computer Animation 2002*, IEEE Computer Society, 33–42.

BENES, B., ANDRYSCO, N., AND STAVA, O. 2009. Interactive modeling of virtual ecosystems. In *Eurographics Workshop on Natural Phenomena*, Eurographics Association, 9–16.

CHEN, G., ESCH, G., WONKA, P., PASCAL, AND ZHANG, M. E. 2007. Interactive procedural street modeling. *ACM Trans. Graph. 27*, 3, 35.

CHEN, X., NEUBERT, B., XU, Y.-Q., DEUSSEN, O., AND KANG, S. B. 2008. Sketch-based tree modeling using markov random field. *ACM Trans. Graph. 27*, 5, 1–9.

DEUSSEN, O., HANRAHAN, P., LINTERMANN, B., MĚCH, R., PHARR, M., AND PRUSINKIEWICZ, P. 1998. *Realistic modeling and rendering of plant ecosystems*. ACM, 275–286.

GALIN, E., PEYTAVIE, A., MARÉCHAL, N., AND GUÉRIN, E. 2010. Procedural generation of roads. *Computer Graphics Forum (Proceedings of Eurographics) 29*, 2, 429–438.

GRABLER, F., AGRAWALA, M., SUMNER, R. Q., AND PAULY, R. 2008. Automatic generation of tourist maps. *SIGGRAPH '08: ACM SIGGRAPH 2008 papers*, 1–11.

IJIRI, T., OWADA, S., AND IGARASHI, T. 2006. Seamless integration of initial sketching and subsequent detail editing in flower modeling. *Comput. Graph. Forum 25*, 3, 617–624.

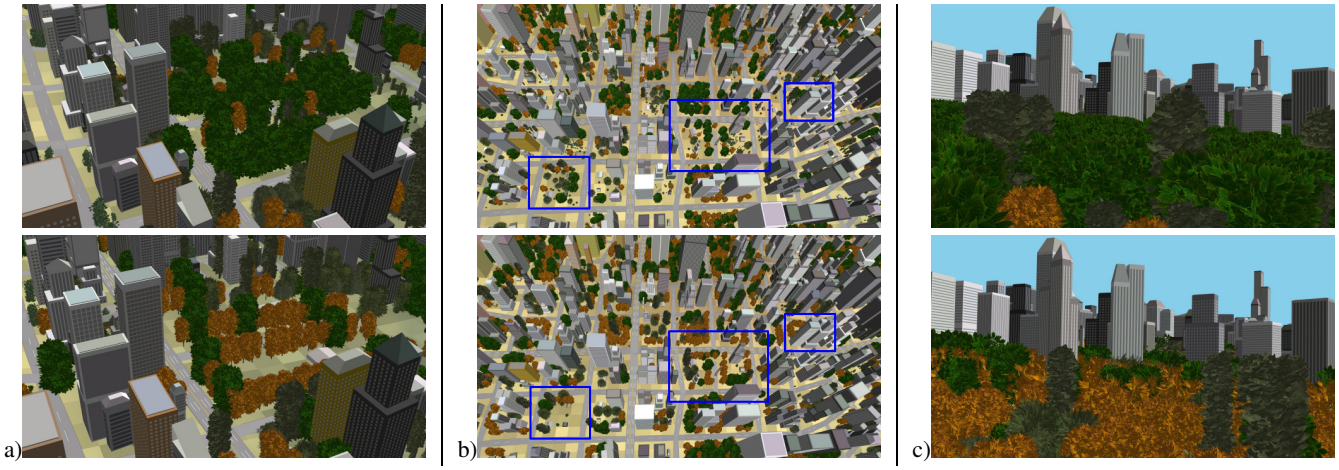

Figure 11: *This figure shows manipulation of the procedural parameters. The upper image in a) was generated by using a low block management value for the block in the center whereas the management value for the lower image in a) is almost $m = 1$. In b), at the top is an image that shows low density of plants seeded along the alleys formed around the streets, rows of trees between houses and egress sites; the bottom image shows what happens if these parameters are set to a high density; we outline with a box several corresponding parts of the city for comparison purposes. A park with mostly chestnut trees can be seen in the image at the top of c) and the same park with mostly ginkgo trees at the bottom of c).*

IJIRI, T., OWADA, S., AND IGARASHI, T. 2006. The sketch l-system: Global control of tree modeling using free-form strokes. *ACM Transactions on Graphics 4073*, 138–146.

LANE, B., AND PRUSINKIEWICZ, P. 2002. Generating spatial distribution for multilevel models of plant communities. In *Proceedings of Graphics Interface'02*, vol. I, 69–80.

LEGENDRE, P., AND LEGENDRE, L. 1998. *Numerical Ecology 2nd Edition*. Elsevier Science.

LINDENMAYER, A. 1968. Mathematical models for cellular interaction in development. *Journal of Theoretical Biology Parts I and II*, 18, 280–315.

LINTERMANN, B., AND DEUSEN, O. 1996. Interactive modelling and animation of branching botanical structures. In *Computer Animation and Simulation'96*, Springer–Verlag Wien New York, Springer Computer Science, 139–151.

LINTERMANN, B., AND DEUSSEN, O. 1999. Interactive modeling of plants. *IEEE Comput. Graph. Appl. 19*, 1, 56–65.

LIPP, M., WONKA, P., AND WIMMER, M. 2008. Interactive visual editing of grammars for procedural architecture. *SIGGRAPH '08: ACM SIGGRAPH 2008 papers*, 1–10.

MERRELL, P., AND MANOCHA, D. 2008. Continuous model synthesis. *ACM SIGGRAPH Asia 2008 papers*, 1–7.

MERRELL, P. 2007. Example-based model synthesis. In *I3D '07: Proceedings of the 2007 symposium on Interactive 3D graphics and games*, ACM, 105–112.

MÜLLER, P., WONKA, P., HAEGLER, S., ULMER, A., AND GOOL, L. V. 2006. Procedural modeling of buildings. In *ACM SIGGRAPH 2006 Papers*, ACM, 614–623.

MÜLLER, P., ZENG, G., WONKA, P., AND GOOL, L. V. 2007. Image-based procedural modeling of facades. In *ACM SIGGRAPH 2007 papers*, ACM, 85.

MĚCH, R., AND PRUSINKIEWICZ, P. 1996. Visual models of plants interacting with their environment. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, ACM, 397–410.

PALUBICKI, W., HOREL, K., LONGAY, S., RUNIONS, A., LANE, B., MĚCH, R., AND PRUSINKIEWICZ, P. 2009. Self-organizing tree models for image synthesis. *ACM Trans. Graph. 28*, 3, 1–10.

PARISH, Y. I. H., AND MÜLLER, P. 2001. Procedural modeling of cities. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, ACM Press, 301–308.

PRUSINKIEWICZ, P., AND LINDENMAYER, A. 1990. *The Algorithmic Beauty of Plants*. Springer–Verlag, New York. With J.S.Hanan, F.D. Fracchia, D.R.Fowler, M.J.de Boer, and L.Mercer.

SUKOPP, H., HEJNY, S., AND KOWARIK, I. 1990. *Urban ecology: plants and plant communities in urban environments*. Balogh Scientific Books.

VANEGAS, C. A., ALIAGA, D. G., BENES, B., AND WADDELL, P. 2009. Visualization of simulated urban spaces: Inferring parameterized generation of streets, parcels, and aerial imagery. *IEEE Transactions on Visualization and Computer Graphics 15*, 2, 1–10.

VANEGAS, C. A., ALIAGA, D. G., BENES, B., AND WADDELL, P. A. 2009. Interactive design of urban spaces using geometrical and behavioral modeling. *ACM SIGGRAPH Asia 2009 papers*, 1–10.

WEBER, B., MUELLER, P., WONKA, P., AND GROSS, M. 2009. Interactive geometric simulation of 4d cities. *Computer Graphics Forum* (April).

WONKA, P., WIMMER, M., SILLION, F., AND RIBARSKY, W. 2003. Instant architecture. *ACM Trans. Graph. 22*, 3, 669–677.

ZHANG, X., BLAISE, F., AND JAEGER, M. 2006. Multiresolution plant models with complex organs. *VRCIA '06: Proceedings of the 2006 ACM international conference on Virtual reality continuum and its applications*, 331–334.