NORTHWESTERN
UNIVERSITY

Computer Science Department

Technical Report
NWU-CS-04-38
August 20, 2004

Procedural Modeling of Land Use in Cities

Thomas Lechner, Ben Watson, Pin Ren, Uri Wilensky, Seth Tisue, and Martin Felsen

Abstract

Cities are the largest human artifacts and appear often in digital entertainment. Building digital models of them is extremely difficult, yet there are few tools that can assist artists in this task. We describe a method for procedurally generating city models using agent-based modeling, an established simulation technique in the social, biological, and physical sciences. Our system accepts a terrain description as input and produces a land usage map that portrays roads and residential, commercial, and industrial land use. The results conform well to modern patterns of land use, but each generated city is unique. Our system is controlled by a set of parameters that users can alter numerically or gesturally in order to constrain and guide the results to fit a particular application's needs.

NSF 0326542

Keywords: Urban Development, Cities, Procedural Modeling, Agent-Based Simulation, Complexity

Procedural Modeling of Land Use in Cities

Thomas Lechner Ben Watson Pin Ren Uri Wilensky Seth Tisue

Northwestern University

Martin Felsen

Illinois Institute of Technology

Abstract

Cities are the largest human artifacts and appear often in digital entertainment. Building digital models of them is extremely difficult, yet there are few tools that can assist artists in this task. We describe a method for procedurally generating city models using agent-based modeling, an established simulation technique in the social, biological, and physical sciences. Our system accepts a terrain description as input and produces a land usage map that portrays roads and residential, commercial, and industrial land use. The results conform well to modern patterns of land use, but each generated city is unique. Our system is controlled by a set of parameters that users can alter numerically or gesturally in order to constrain and guide the results to fit a particular application's needs.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling

1. Introduction

Computer graphics research on procedural modeling has produced a number of solutions for generating models of natural objects and phenomena, including terrains, fire and explosions, water, clouds and plants. Interestingly, there are few methods for procedural modeling of human artifacts.

The procedural generation of manmade artifacts is of particular relevance to several audiences. With the capabilities of gaming hardware continually improving, the entertainment industry is under growing pressure to provide more content in its products, while still maintaining the same development cycle and cost. Film companies are starting to utilize virtual sets more frequently. Urban planners and architects are constantly looking for innovative ways of visualizing their ideas both as a creative tool and to communicate their ideas to students and clients.

In this paper, we describe a system for the procedural modeling of cities. Our system differs from previous systems in its effort to model *land usage*. For instance, [PM01] devised a method to generate road networks of various styles, like those of Manhattan or Paris. However, they populated land spanned by their road networks with a city wide arrangement of skyscrapers based upon a density distribution input by the user. Our model *automatically* simulates several different land uses as the city develops, resulting in a more realistic distribution of buildings across the urban landscape. The simulation is *continuous*, allowing the user to halt it at any time, or to *modify* and constrain it before further development takes place.

2. Background

Previous attempts of the graphics community to model cities have primarily focused upon road networks. The input to these methods is usually a terrain specification and a population distribution from existing city data. An L-system then grows the road network to service the population. Often the road networks have a structural design option, such as a gridded versus radial layout [SYBG02]. In the case of [PM01], a height map was added as an extra constraint, and simple building geometry was generated as a function of the density of the population over a given plot. The shortcomings of these methods were their dependence on data input by the user, and the lack of a smooth transition between different road parameters and layouts. In the case of [PM01], the automated architecture did not simulate land use, resulting in entire cities that looked like downtown Manhattan.

Figure 1: *Here is a city generated by our simulation and visualized using SimCity. Buildings are distributed throughout the city using the zoning allocations and density distributions generated by the developer agents. This particular simulation demonstrates how our system utilizes local parameter specifications to achieve different development strategies. The center of the city demonstrates gridding behavior, while the outer perimeter follows an organic strategem.*

Other research has been focused on the architecture of the city rather than its urban layout. [WWSR03] devised a method to generate detailed facades using L-systems. The resulting facades exhibit cultural variety and respond to the influences of population and material. Yet this method can be quite complex and difficult to control. As different types of buildings are added, valid representations in the L-system encoding do not make sense when converted into 3D architecture. As a result, this method requires another set of constraints to monitor all the possible contradictions and prevent them from occurring. Such a solution rapidly grows in size and complexity. Finally, since their system does not account for an urban layout, they depend on detailed data from already existing cities, including building footprints as well as population densities.

Procedural modeling has long made use of agent-based simulations. Typical examples include Reeves' particle systems [Ree83] for fire and sparks and Reynolds' Boids [Rey87] for flocking. Both involved interdependent simulated agents or entities, each able to sense their local environments and make decisions based on what they perceive. The collective behavior that emerges from their relatively simple local behaviors can be quite complex.

The real world contains many such complex systems, including economies, ecosystems, fluids, gases and societies. The programming language and multi-agent modeling environment, NetLogo [Wil99], was developed to simplify simulating complex systems on the computer. NetLogo was designed for use in both research and education and has become widely accepted in both communities. Though we are not novice programmers, our model-building effort derives advantages from Netlogo's simplicity. The high-level nature of the language enables rapid development, aids low-cost experimentation with different designs, and facilitates the collaborative nature of our work. User extensions to NetLogo can be written in Java; we may make use of this capability as the project develops.

3. The Model

Our simulation makes use of several agent types to generate the most common components of cities: its residential, commercial, and industrial constituents. However, ultimately it

will need to support still more specialized agents types to increase detail, including government buildings, squares and schools. An L-system seemed inappropriate because in the face of such complexity, it might result in "parameter bloat" like that described by [WWSR03]. On the other hand, agent-based simulations describe behaviors and semantics that are much more local, with agents interacting directly with their environment and only indirectly with each other. Thus parameter bloat is not as serious a concern (though still quite important).

In NetLogo the environment in which the agents act is represented as a rectangular grid of *patches* called a *world*.

3.1. The Output

Our system generates an abstract land usage description consisting of *residential*, *commercial*, *industrial*, and *road* land uses. The patches of the world are grouped into *parcels* under the ownership of a *building* agent. The building agent determines the zoning information of the parcel and manages the attributes of the building, such as population, population density, age, and value. Building agents also keep track of neighboring buildings, as the building eventually placed on a parcel may depend upon what other building types exist around it.

Patches may also be occupied by roads. Our system models two types of roads from the real-world road hierarchy: *tertiary* and *primary* roads [EGE93]. Tertiary roads are the lowest in the hierarchy, being the smallest and the most prevalent. Their function is to provide access to nearby property and primary roads. Residential areas and industrial parks tend to utilize these roads the most. Primary roads are the highest in the hierarchy, being the widest and most commonly traversed. Their role is to facilitate travel between high density areas, and support the major portion of traffic flow across the city. In practice, secondary roads fall between these two extremes in the road hierarchy, but these types of roads have not yet been added to our simulation.

3.2. The Input

The primary source of input to our simulation is a terrain height map. A number of global parameters, such as water level, can either be set directly by the user, or left to the default configuration. The types of parameters the user can set predominantly influence the development of roads. Attributes such as road density, grid spacing, allowable deviation from the grid, and desired road interconnectivity can either be applied to the whole world or to a specific portion of it.

Our system is designed to be as self sufficient as possible, even without much user interaction. Yet certain constraints of game or urban design may be unique to the applied context and require user control. For this reason, we have extended our model to allow user specification of parameters

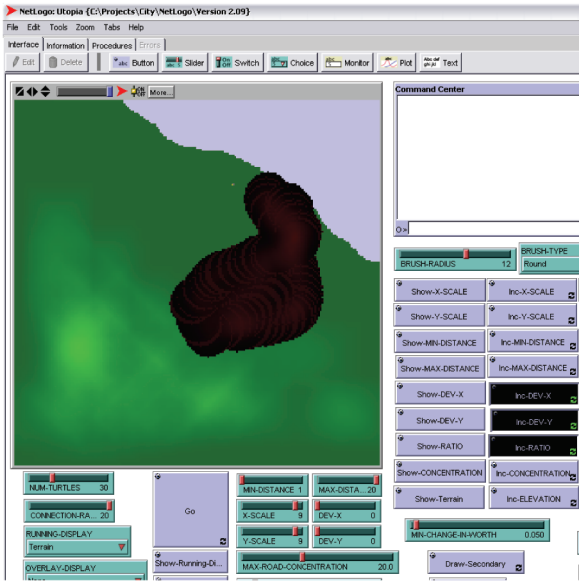

Figure 2: *Users can specify local parameters through a painting interface.*

at certain locations in the world by "painting" those locations with the needed parameter values 3.2. In short, we endeavor to allow the user to specify as much or as little as he or she wants in order to obtain a useful and/or realistic looking city.

3.3. The World

Each patch in the world stores information regarding the interaction between agents and itself. Most important is *value*, which describes the price and thus attraction of building agents to patches. In order to speed up the simulation, we use a lazy update, recalculating patch information only when an agent requests it.

3.4. The Agents

Our agents approximate the actual behavior of urban developers and development [EGE93]. A developer will make a land speculation and choose a development site. He will then draw up a proposal that satisfies his clients' needs and the city's restrictions. The city ultimately decides whether or not the development occurs. This is dependant on whether the proposal matches the city regulations, and if it makes some net positive impact in the community, e.g. raising land value or fulfilling a public service.

Each agent roughly follows the same algorithmic paradigm. The agent spends a certain period of time searching through the world until it discovers a suitable place to start development. Once that place is found, the agent generates a hypothetical solution, and proposes it to the city. The

Developer	Quality Vector								
	R	W	E	1/E	D_r	$1/D_r$	D_c	D_i	$1/D_i$
Residential	0.15	0.25	0.25	0.00	0.25	0.00	0.00	0.00	0.10
Commercial	0.41	0.17	0.17	0.00	0.00	0.00	0.25	0.00	0.00
Industrial	0.20	0.10	0.00	0.05	0.00	0.40	0.00	0.25	0.00

Quality	Symbol	Description
road density	R	Surrounding road concentration
water value	T	Value accruing from proximity to water
elevation value	E	Value accruing from proximity to desired elevation
residential density	D_r	Surrounding residential density
commerical density	D_c	Surrounding commercial density
industrial density	D_i	Surrounding industrial density

Figure 3: *Value functions for residential, commercial, and industrial developer agents.*

city tests the proposal against its constraints, and should the proposal pass, the agent purchases the land and builds or improves the structure on it. Afterwards, the agent resumes its searching phase.

3.4.1. Developers

Developers are those agents that generate parcels and use land. Urban developers identify at least nine different types of land usages. Our model currently only implements the *residential*, *commercial*, and *industrial* categories since these three categories make up the majority of land use in most cities. Other categories include government buildings, cemeteries, and water/sewage.

Each type of developer has a different method of evaluating land value [EGE93]. *Residential* developers prefer regions where the road network is less busy. They avoid industrial zones when possible, and prefer to be close to water. On the other hand, *industrial* developers often choose land of lesser quality and avoid residential zones if there is another option. Depending on the type of industry, certain developers may want to seek locations with access to heavy transport. *Commercial* developers are amicable towards both residential and industrial zones, and are particularly interested in seeking high traffic areas of the road network.

In their search phase, developer agents are attracted to patches with high value. If any of their proposed developments improves value at a certain location, the city grants permission to develop. Each type of developer agent has its own method for valuing a patch in accordance with the type of development it builds. Values are determined using a weighted sum of local patch attributes. These attributes are normalized by their respective means, so that the attributes that are above patch average become more attractive, and those that are around or below average are less so. The weights sum to one but are different for each type of developer. Thus an average patch will have a value of 1, while a more interesting patch will have a higher value.

Developer agents can build parcels using three different methods. The first is simply to introduce a new parcel on previously undeveloped land. A developer will select a patch adjacent to the road network and grow a parcel from it patch by patch. The depth of the parcel perpendicular to the road network is determined by the grid size selected by the user. The number of patches in the parcel depends on the type of developer. In our simulation, residential parcels are kept to about a fourth of the size of industrial parcels, and commercial parcels are about half as large as residential parcels. If the number of available patches is too small, the developer searches for any adjacent parcels with a matching land use. If any such parcel is found, the developer chooses one that can accept the patches in the incomplete parcel without becoming too large.

Developers can also improve an existing parcel. The more the value of a parcel increases, the more likely it is that developers will want to upgrade it. Our system defines an upgrade as an increase in the parcel's population density. Note however in the above value equations 3.3 that huge disparities in population among neighboring parcels reduces value. In addition, in most real-world cities the density of certain types of buildings is legally limited. We simulate this by heavily penalizing parcel value when the simulated limit is exceeded.

Finally, developers can change the land use of an existing parcel. As cities develop and land uses change, the land values of the parcels themselves begin to change. Parcels which may have originally been valuable to one type of developer may no longer be considered valuable, or perhaps may be considered to be more valuable by other developers. Our system allows a developer to change land usage when an increase in value would result. This comparison of values

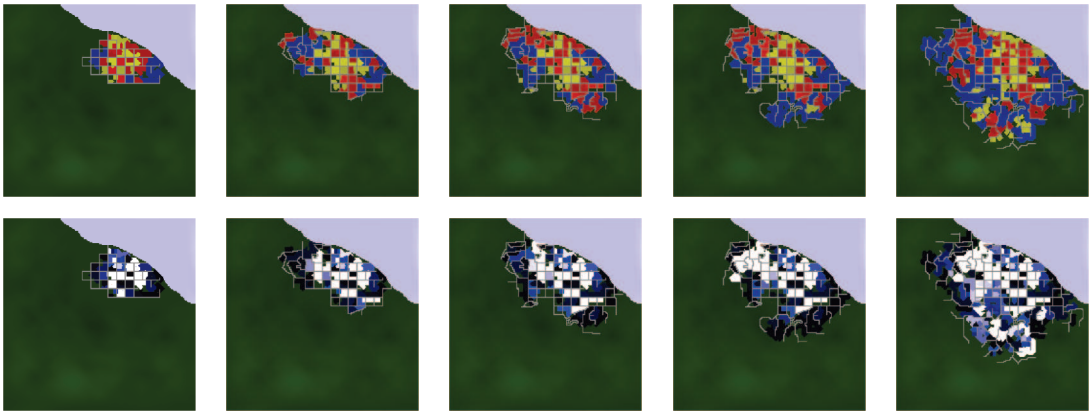

Figure 4: *A sequence of development stages. Top row represents land use, bottom row represents population density.*

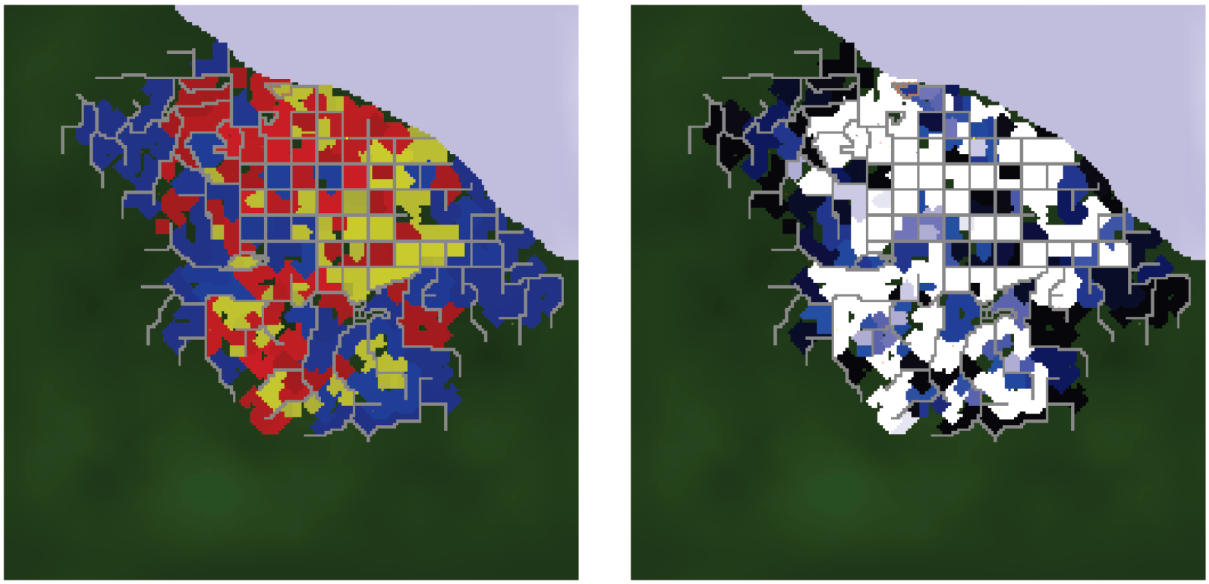

Figure 5: *The left image represents the final simulated land use. Yellow is residential, red is commercial, blue is industrial. The right image represents the final simulated population density. Blue is low density, white is high density.*

for different land uses and across different developers is one of the major reasons for normalizing patch attributes.

3.4.2. Roads

Tertiary roads are designed to service land within a certain limited distance, and they generally tend to be private, resulting in smaller road diameter and a variable level of interconnectivity. They are generated through the efforts of two types of agents - *extenders* and *connectors*. When a road segment is laid down, these agents mark each of the patches within a given radius with the distance to the nearest road. An extender agent roams the terrain until it finds a piece of land that is not serviced by the existing road network. Once that area of land has been discovered, it follows the declining distance values back to the road network, and keeps a record of all the patches it passes through during its journey. The terrain has some influence on the agent's decision, with the agent trying to avoid large changes in elevation. In addition, when the road is not strictly gridded, the extenders attempt to choose paths that go along parcel boundaries. Once the agent reaches the tertiary road network, it computes some tests to ensure that the resulting network is not too dense, the new intersection is not too close to another, and that the segment travels a reasonable distance. These tests involve issues like road density, proximity to existing intersections, and deviation from the starting position. If these tests are passed, the

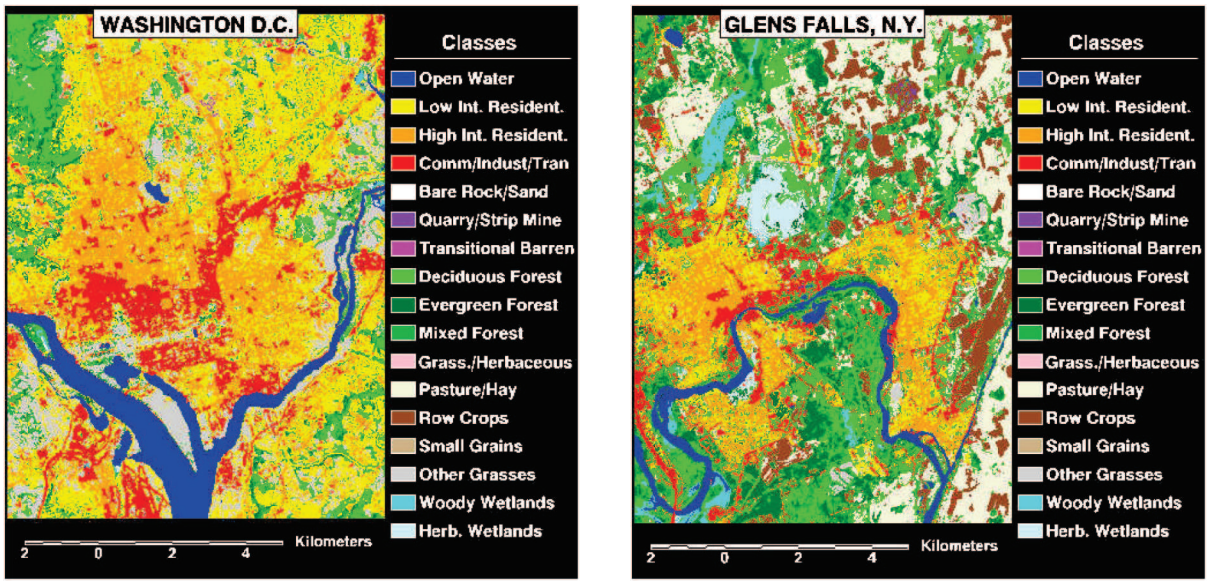

Figure 6: *Land use in two real world cities. Note the separate commercial and residential clustering. (Images taken from* http://landcover.usgs.gov/sample.asp*)*

road segment offered by the agent is confirmed and added to the existing road network, and the extender continues to roam in search of more land to pave. To prevent the road network from developing faster than the building construction, extenders are forbidden to roam too far from developed land.

A connector agent is constrained to wander over the existing road network. As it does so, it samples a random patch in the road network within a given radius. It then proceeds to try to reach the selected patch via the road network with a breadth first search. If it cannot reach its target within the original sampling radius, or if it must go too far out of its way, it will attempt to build a road segment between its current patch and the one it cannot reach. The suggested road segment is subjected to the same tests used by the extender agents before it is added to the network.

The constraints imposed on our road-building agents can be used to simulate a range of differing behaviors. The user can specify that the road segments must lie directly on a pre-established gridded pattern, or give the road segments permission to be located within a certain deviation from the grid. When the deviation is set very large in proportion to the length of the submitted road segments, road layout becomes curvilinear and organic. As this deviation is set towards zero, the roads become more rectilinear.

The second type of road simulated by our model is the primary road. It is important to note that primary roads are not expressways. They are merely larger roads carrying more traffic and containing several lanes. Primary roads service the city by supplying a means of transit between dense areas of the city. In our model, we distinguish primary roads by coloring them darker and making them twice as wide as tertiary roads.

The primary road agent always tries to link dense regions of the city with the city center (of its population density), or the existing primary road network. These agents search the world for regions with both high density and large distance from the primary network. Once an agent reaches a local maximum in population density and distance over a certain threshold, it builds a primary road from its current location to the city center. In order to prevent the roads from merely accumulating at the city center, the agent searches the local area around the direct path between it and its target. It will then choose the closest primary road that falls in this region as its new target location. Once the target location has been chosen, the agent follows a simple path algorithm with constraints designed to discourage the primary road from crossing parcel boundaries and encourage upgrading of the existing road network. After the agent reaches its target patch, it submits the potential road to the city jurisdiction for approval, using criteria similar to the connector agents.

4. Results

In Figures 4-5, terrain and local road parameters were input via a simple paint interface. Higher elevation is represented by increased luminance. Residential parcels are denoted by yellow, commercial are denoted by red, and industrial are represented by blue. Primary roads are distinguished from tertiary roads in that they are twice as wide and a deeper shade of gray.

Figure 7: *Two close-up SimCity visualizations of the town in Figure 5. Notice the transition from gridded to ungridded layout.*

4.1. Comparison to Real World Data

In comparing our results to real world city data sources (Figure 6), we are pleased to discover that our simulated images show similar patterns of land use. Residential districts tend to develop near waterfronts and segregate themselves from industry. Commercial districts are drawn to areas of high road density, and mingle well with the residential and industrial regions. Industry by contrast attempts to develop along the city edge, and generally gives way to residential and commercial influence as the city develops. Tertiary roads adhere to gridding parameters and avoid crossing parcel boundaries. Primary roads demonstrate connectivity among the denser parts of the city and its center.

The density and development of the city also demonstrate encouraging behavior. At the founding of the city, the residential district primarily remains situated near the city center. The city has a highly populated core with density that drops off toward the outer edges. But as the city develops, small suburbs begin to form in less dense regions along the city outskirts, eventually drawing more of the city density toward them. It is interesting to note that these behaviors have emerged without any explicit rules designed to generate them.

Some of our results show more industrial use than many of the real world datasets we received. However, once we diminished the number of industrial agents, our results started conforming once more to land use distributions of our real world data. In the future, we may integrate a simple economy model into our simulation to achieve more realistic industrial land use.

4.2. SimCity

Our discussion up to this point has focused on a very abstract city model. Electronic Arts and Maxis have very kindly granted us the source code for SimCity 3000 to aid our research effort. The SimCity 3000 graphics engine has been very useful in helping us visualize our results. However, we found it necessary to introduce limitations in our simulation output to replace the SimCity simulation output. For example, the SimCity simulation only generates gridded layouts and and has very regularly shaped parcels. SimCity also simulates only one type of road, while we simulate a road hierarchy. Although the images it generates are more artistic and map-like than strictly realistic, it is still valuable as an interim visualization tool. Eventually, we plan to build our own realistic 3-D city renderer to bring our cities to life.

We begin our SimCity visualization by loading an empty scene and directly mapping our elevation, terrain, and road distributions to the environment. The SimCity road system is very sensitive to small deviations in elevation, which frequently causes the road network to break up. To prevent this from occurring we limit differences in height.

Next we attempt to populate the city with buildings according to the zone types and population densities generated by our simulation. This task proved to be quite difficult. SimCity applies a different land use scheme for its buildings, which doesn't directly correspond to our less regularly shaped parcels. First, we manually generate a subset of the original SimCity building dataset, and categorize each building by its type, footprint, and population density. Then when we load our simulated model, our SimCity extension splits

each parcel into squared plots with evenly distributed density. Our extension then tries to find the best-fit building from our subset in regard to the zone type and density for those slots. Since this does not guarantee that the entire parcel is filled by the building allocation, we supply space fillers such as parking lots for commercial zones, and lawn units for residential zones.

5. Future Work

For games and film, our model will need to be extended to be capable of generating larger, more detailed maps. We will need to build more complete transportation networks, e.g. highways and public transit, and include more land use types, such as government buildings, cemeteries, and public parks. We anticipate that new types of agents will be necessary to reach this level of richness. Performance enhancements to NetLogo may be necessary to accommodate more detail and larger output size.

Currently, our model is culturally specific. As our model for Western culture becomes more complete, we will expand our research to support other urban layouts from different cultures and historical periods and the transitions between them.

Finally, the two-dimensional land use modeling described here is only one constituent of what will need to be a dual approach to generating three-dimentional urban content. The other component will be a new 3-D architectural generator to populate our cities with realistically varied building shapes. When both components are complete, our cities should lack only the people to live in them.

6. Acknowledgements

This project is supported by NSF grant 0326542, and Electronic Arts. We would also like to extend our thanks to Michael Kwartler and George Janes of the Environmental Simulation Center for many interesting discussions.

References

[EGE93] EISNER S., GALLON A., EISNER S.: *Urban Pattern*, sixth ed. John Wiley & Sons, Inc, 1993. 5, 6

[PM01] PARISH Y. I. H., MÜLLER P.: Procedural modeling of cities. In *Proceedings of ACM SIGGRAPH 2001* (July 2001), pp. 301–308. 3

[Ree83] REEVES W. T.: Particle systems: a technique for modeling a class of fuzzy objects. *ACM Transactions on Graphics 2*, 2 (1983), 91–108. 4

[Rey87] REYNOLDS C. W.: Flocks, herds, and schools: A distributed behavioral model. In *Proceedings of ACM Symposium on Virtual reality software and technology 2002* (July 1987), pp. 25–34. 4

[SYBG02] SUN J., YU X., BACIU G., GREEN M.: Template-based generation of road networks for virtual city modeling. In *Proceedings of ACM Symposium on Virtual reality software and technology 2002* (Novermber 2002), pp. 33–40. 3

[Wil99] WILENSKY U.: NetLogo. Center for Connected Learning and Computer Based Modeling, Northwestern University, 1999. http://ccl.northwestern.edu/netlogo. 4

[WWSR03] WONKA P., WIMMER M., SILLION F., RIBARSKY W.: Instant architecture. *ACM Transactions on Graphics 22*, 3 (2003), 669–677. 4, 5