Easing the Pain of Urban Modeling

John Mann
Applied Research Associates, Inc.
Orlando, FL
John.Mann@ara.com

Latika (Bonnie) Eifert
US Army RDECOM-STTC
Orlando, FL
Latika.Eifert@us.army.mil

ABSTRACT

An increase in Urban Operations training in the last few years has led to a greater reliance on highly detailed urban terrain databases for training simulations. These databases usually include buildings with realistic room layouts, and doors and windows for the ingress and egress of semi-automated forces. Training simulations that use dense urban terrain models require a variety of building types with differing layouts, ranging from small homes with simple room layouts to large office buildings with many floors and different layouts on each floor. Depending on the training objectives, these models may contain either geotypical or geospecific room layouts. Generating models of these buildings takes days or weeks. Not only is this costly, but it slows terrain database production, preventing fast turn-around of urban databases. In this paper, we introduce a new tool for rapid generation of urban terrain models. We describe the algorithms used to generate building interiors and discuss potential applications for these models. In some cases, models that used to take a few weeks to build can now be created in less than a few minutes. These models are created parametrically, permitting an unlimited variety of building shapes, sizes and types to be generated. An important feature of these models is that they contain structural properties. This enables modeling of physics-based interactions with the building. Terrain databases that include these models are supporting Future Combat Systems exercises using OneSAF Testbed (OTB) and the OneSAF Objective System (OOS).

ABOUT THE AUTHORS

John Mann is a Principal Computer Scientist at ARA. For the last nine years he has developed and integrated physics-based urban modeling and weapon-target interaction modeling tools. He is the project manager for the Urban and Underground Model Generator (U2MG) project and the Combat Training Center Objective Instrumentation System (CTC-OIS) Area Weapons Effects Simulation (AWES) project. Mr. Mann has a B.S. in Computer Science from the University of Maryland

Latika (Bonnie) Eifert is an engineer at RDECOM. She is focused on the development and integration of technology for synthetic natural environments in the Modeling and Simulation arena. Latika is a graduate of Louisiana State University, with a B.S. in Electrical Engineering, and an M.S. in Computer Engineering from the University of Central Florida. Latika worked at Lockheed Martin and SAIC supporting a wide variety of hardware and software testing. She has also supported various efforts in modeling and simulation and system engineering efforts within the DoD.

Easing the Pain of Urban Modeling

John Mann
Applied Research Associates, Inc.
Orlando, FL
John.Mann@ara.com

Latika (Bonnie) Eifert
US Army RDECOM-STTC
Orlando, FL
Latika.Eifert@us.army.mil

INTRODUCTION

Urban simulation holds great promise for helping address needs for civil and military planning, analysis, training and experimentation. The city planner flies through an urban model to view proposed new buildings in a virtual environment (Maese, 2003). Military planners perform course of action (COA) analysis using urban simulation. Soldiers engage in urban operations training using virtual weapons and visualizing simulated weapons effects.

In each of the examples above, activities that are impossible in the real world – visualizing proposed changes to the urban environment, inserting simulated troops and vehicles into an urban environment for COA analysis, and showing simulated damage to buildings for training exercises (Figure 1), are easy to do in a simulated urban environment. The ability to model these activities makes urban simulation an attractive alternative to traditional methods of planning, analysis, training and experimentation.

Figure 1. A simulated F16 bomb drop visualized in MetaVR's Virtual Reality Scene Generator.

Terrain modelers rely on high quality GIS data and automated terrain database generation tools to create urban databases. These automated tools facilitate the production of databases covering thousands of square kilometers, containing hundreds of thousands of buildings. These tools use automated processes to extrude buildings, create roads, add foliage, and place urban clutter (signs, telephone poles, fire hydrants).

BACKGROUND

In 2003, the Army Research Development and Engineering Command (RDECOM) in Orlando, FL in collaboration with the Defense Threat Reduction Agency (DTRA) funded a small effort to develop a building generator prototype. The building generator is based on a tool used to characterize urban models for weapons effects simulation (Harman and York, 2003). The objective of this prototype was to develop an automated process to create geotypical buildings with interiors and export them to visual and Semi-Automated Forces (SAF) terrain database formats.

The prototype evolved into a robust building generator capable of automatically generating a variety of building types for large terrain databases. These databases, covering thousands of square kilometers of terrain, are being used by the Army for Future Combat Systems (FCS) exercises. This tool, called the Urban and Underground Model Generator (U2MG) (Mann and Pigora, 2004), has been used to develop terrain databases containing hundreds of unique building layouts (Figure 2).

The purpose of this paper is to discuss the problem U2MG was developed to address and describe the algorithms used to create the building models. We will discuss data sources used to create the building models, the process for creating the models, and areas of current development.

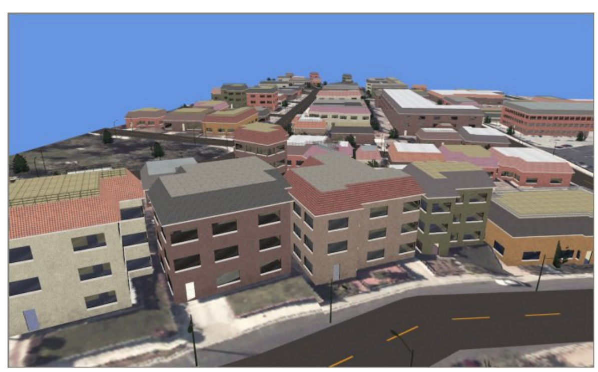

Figure 2. This visual database was created from building footprints derived from LIDAR data and processed by the U2MG building generator.

PROBLEM OVERVIEW

Although many parts of the terrain database generation process have been automated, some steps in the process typically require human intervention. One of those steps is the creation of building models. The factory shown in Figure 3 is an example of a model created manually using a polygon editor. This model took 4 days to build using MultiGen Creator (source: email from Joe Burns, Mounted Warfare Test Bed on June 12, 2006). Models usually take a few days to a few weeks to create, depending on the complexity of the model.

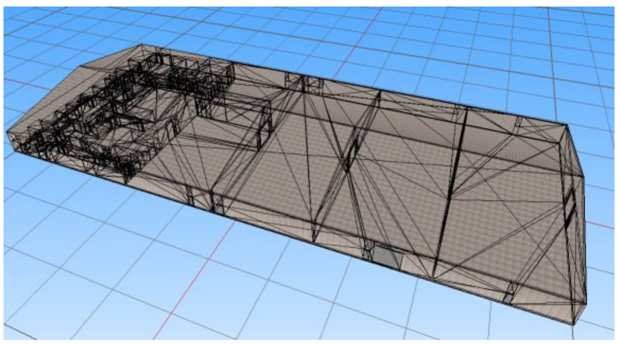

Figure 3. This factory was modeled using MultiGen Creator, a polygon editing tool.

It is not practical to model a handful of buildings and duplicate those models to populate a database if geospecific building footprints are required. The models must have interiors that are tailored to fit within the geospecific footprint. Attempting to manually create these building models for a large database is very time consuming and expensive. The alternative is to use an automated process for generating models.

The Challenge

The challenge with automating interior layouts lies with creating a reasonable layout when little is known about the building. GIS source data contains a limited amount of data about a building, usually supplying only the footprint dimensions and the elevation. Location of exterior doors and windows may be extracted from imagery, though usually only a fraction of the apertures will be visible.

There have been a few tools developed to address this challenge (e.g. Rapid Building Generator, ERDC (Pace and White, 2003) and Automated Building Generation System (ABGS), Lockheed Martin). These tools have been used with some success to build databases for SAF simulations.

The approach we are using goes beyond these previous efforts in several ways. We have developed a tool that generates building layouts for nearly any building shape, with one restriction – wall intersections must be orthogonal. The generated buildings are structurally correct and contain material properties that can be used to model weapon-target interactions. The building generator is exposed through an API interface, so it can be integrated into other applications. Finally, the interior layout generation algorithm is data driven. Users can modify the building layout by editing XML templates that contain the layout parameters.

The building in Figure 4 was created using this tool. This model is nearly identical to the building in Figure 3, but the creation time was significantly less.

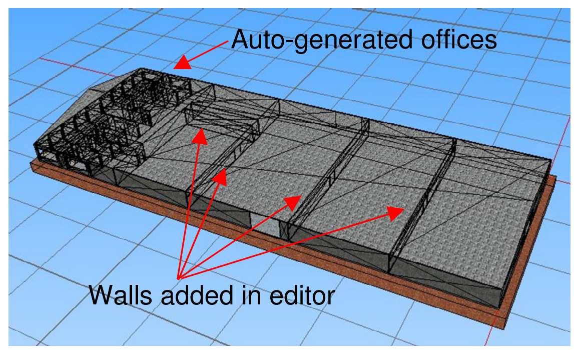

Figure 4. This building was auto-generated using the U2MG factory template and modified in the editor to add walls enclosing manufacturing bays. Total creation time, including editing, was less than 5 minutes.

URBAN MODELING PROCESS

Figure 5 compares the manual process to the automated building creation process.

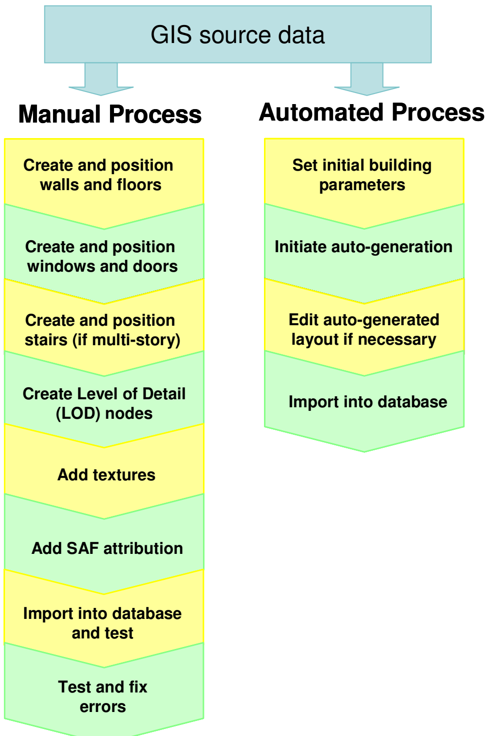

Figure 5. This comparison illustrates the amount of effort required to generate a model manually and the steps saved in the automated process.

The advantage of the automated process is the short setup time and small amount of training required to use the tool. The manual model creation process requires expert knowledge of the modeling tool. Based on conversations with users, the manual process takes several days versus minutes for the automated process.

The next several sections describe the automated building creation process in detail.

GIS Source Data

The modeling process starts with GIS source data. There are several sources of data for creation of urban terrain. Shape files are normally used to create building footprints. Shape files are derived either from footprints generated from LIDAR data (Figure 6) or by extracting building footprints from high-resolution imagery (Figure 7).

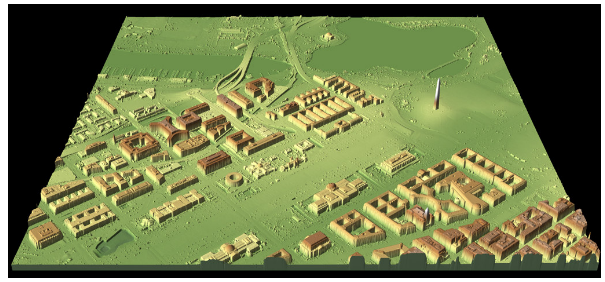

Figure 6. LIDAR capture of the Washington, DC mall area.

Building elevations are derived from Digital Elevation Models (DEMs). DEMs must have high enough resolution (generally 1 meter or better) to derive accurate building elevation data. This is especially true when modeling dense urban areas. The DEM is also used to model the terrain surface surrounding the buildings.

Hyperspectral and multi-spectral data sources provide attribution that add value to urban terrain databases. These data sources are derived by capturing multiple light bands from the terrain surface, By matching the values of the light bands with known values for objects found on the terrain surface, we can derive a rich source of attribution. Spectral data may be used to determine roof types and building materials. With this data, we can assign the appropriate texture maps to models and assign structural attributes to buildings.

Many types of GIS data are available for free or for very little cost online. For example, we were able to download freely available digital ortho photos and elevation data of Orange County, Florida. We requested commercial building footprints from the local city government office and were provided with this data at no cost (Figure 7 and Figure 8). Similar data sets are available for most metropolitan areas for little or no cost.

Figure 7. We downloaded this digital orthophoto of the Orlando Executive airport and surrounding area at a public domain website at no cost.

Figure 8. We obtained these shapefile footprints of Orlando commercial buildings from the local city government.

Automatic generation of interior layouts

The GIS source data provides the inputs for the building footprint and the building height. Roof and exterior wall properties may also be derived from GIS data, if this type of data is available. These inputs are fed into the automated interior layout algorithm.

The automated interior layout process uses a rule-based algorithm. The rules were derived from observations of typical building layouts for each building type. For example, there are certain rules that apply in general to an office building that permit only a finite number of possible office layouts. Elevators are usually located near the center of the building and close to the main entrance for accessibility reasons. Most offices line the exterior walls next to windows, while bathrooms are usually in common areas.

Our building layout rules take into consideration several factors, including the number of plausible rooms and room types, placement priority of room types, proximity to other building elements, and room dimensions. Some elements such as elevators and stairs have restrictions on placement, dictated by building codes and safety regulations. Other elements are placed according to the preferences of the tenant. Placement of these items is generally ruled by accessibility, maximum use of space and/or aesthetics.

The rule sets vary depending on the type of building. For an office building, the algorithm places the main entrance, hallways, elevators and stairways before any other elements. The algorithm divides the remainder of the building into sub-areas and applies the rules for placement of remaining building items, maximizing the use of space while maintaining accessibility.

For large buildings, the algorithm will consider hundreds of possible combinations of rooms, selecting a final layout for the sub-areas that makes the best possible use of space by maximizing the number of rooms placed. The algorithm will resize individual rooms within the dimension constraints specified by the rules. After selecting an optimum layout, the algorithm finishes by placing doors on offices and at the ends of hallways, and windows on exterior walls.

The key to creating a realistic layout is dividing up the building into logical units. This begins with the initial hallway placement and the selection of the sub-areas of the building (Figure 9). The main hallway in an office building is placed in the center of the building. Additional hallways (how many will depend on the building dimensions) are placed so that all offices will have access to a hallway. Elevators and bathrooms are placed near the main hallway and stairs are placed next to the smaller hallway.

The remaining open spaces are divided up as shown in Figure 10. These sub-areas are further divided up as shown in Figure 11. It is much easier to arrange rooms in a logical pattern by this iterative process of subdividing spaces as opposed to finding the optimal room arrangement for a large area. The final step is to generate all possible office layouts and select the layout that maximizes the available space in each sub-area (Figure 12).

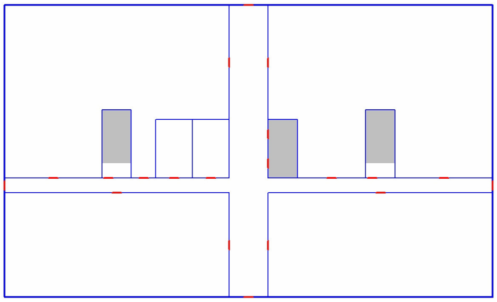

Figure 9. Create entrances and common areas (e.g. hallways, elevators, staircases and bathrooms)

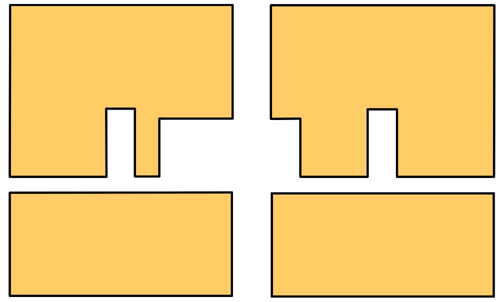

Figure 10. Determine the remaining areas (areas left over after subtracting common areas)

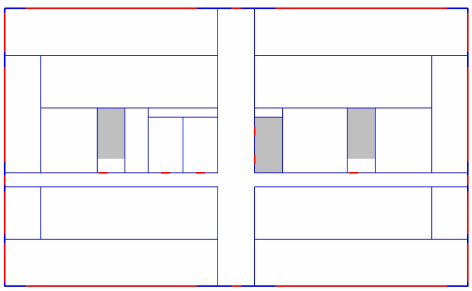

Figure 11. Subdivide remaining areas into rectangles (multiple options)

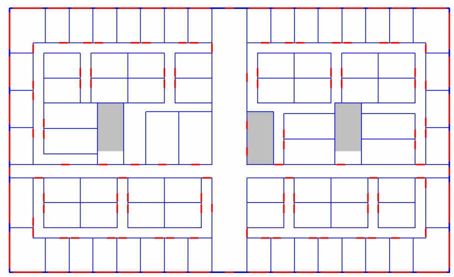

Figure 12. Subdivide rectangles into offices (maximize number of offices) and add windows and doors

Building templates

One requirement for this project was to provide the user with building templates that could be modified to create variations of standard building types. We did this by creating an XML file for each building type. This file contains a list of room types, room dimensions and placement priority within the building. Rooms with the highest placement priority are located after hallways are placed.

We created templates for seven building types: offices (Figure 13), hotels (Figure 14), factories (Figure 15), apartment complexes (Figure 16), aircraft hangars, farm storage and houses. The algorithm for each building type reads the parameters from the XML template and creates a layout that satisfies the constraints set in the template. Each building type has a rule set that uses the parameters in the XML template to create the layout.

Algorithms control the logic for arranging rooms. The XML template specifies parameters that the algorithm uses to define the rooms. In the future, we plan to move some of the room placement logic into the XML file. This will allow users to create new building types by combining rules from two or more XML files.

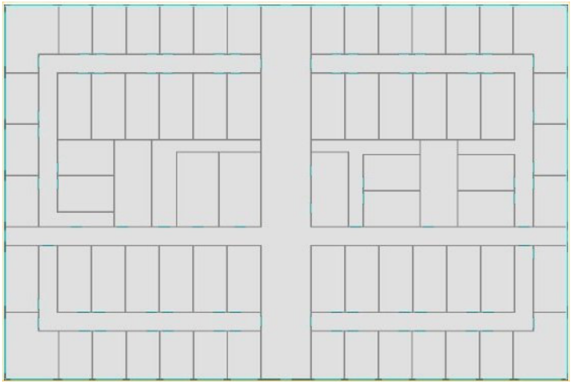

Figure 13. Office building layout

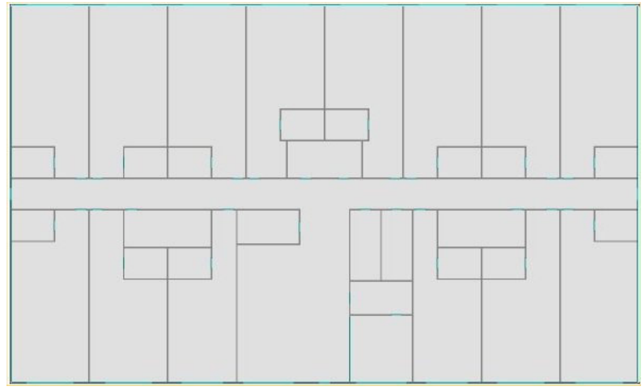

Figure 14. Hotel layout

Figure 15. Factory layout

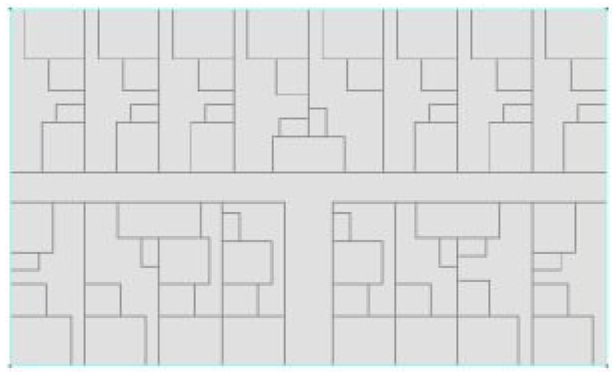

Figure 16. Apartment complex layout

Arbitrary shaped buildings

An arbitrary shaped building is a non-rectangular building footprint with orthogonal wall intersections. The algorithm for generating footprints for arbitrary shaped buildings is similar to the algorithm for rectangular shaped buildings.

The algorithm for finding the optimal room layout for an arbitrary shaped building (Figure 17, Figure 18) starts by dividing the building into rectangular areas. If two rectangular areas are adjacent, the algorithm tags those areas as connected. The rectangular areas are subdivided using the same algorithm as for rectangular buildings. The room layout algorithm populates the subdivided areas with rooms, placing doors at the connection points between adjacent areas.

Figure 17. This arbitrary shaped building was created from a shape file footprint.

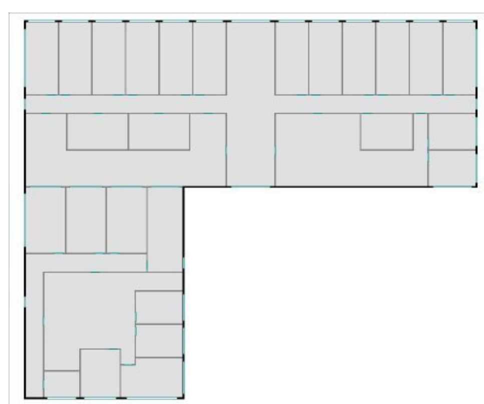

Figure 18. This office building layout was generated to fit a geospecific footprint.

Geospecific room layouts

Mission planning and rehearsal require geospecific rather than geotypical room layouts. A user can modify these geotypical layouts in the building editor to create a geospecific layout. The editor contains tools for each type of building element (walls, doors, and stairs). Note that there is a time savings by editing with higher order building elements such as walls and doors versus editing polygons. With these editing tools, the user can add, remove or change the properties of building elements. Once the building model is complete, the user can import the model into the terrain database or export the model to OpenFlight or XML format.

Creating geospecific room layouts is primarily a manual process. In the future, programs like DARPA Visibuilding (DARPA, 2005) will automatically map building interiors using sensors. Building generators will leverage this data to generate geospecific interiors.

ADDING ADDITIONAL DETAIL TO BUILDINGS

One of the greatest challenges in using auto-generated tools for creating urban models is creating models that look realistic. The usefulness of a building model diminishes significantly if the person using it does not believe the building model represents an actual building in the real world. The capabilities we discuss in this section are either in development or are part of near term development efforts.

Increased realism typically takes the form of improved model textures and additional detail beyond the main structure of the building. Improving the appearance of a model with textures may involve moving beyond basic texture mapping and taking advantage of advances in graphics. Correct mapping of texture sizes to polygons, use of lighting and shading and use of multi-texturing (i.e. combining of multiple textures) can improve the appearance of textures (Figure 19).

Figure 19. U2MG generated building with window sills, soffits and semi-transparent window textures added for realism.

We have added additional features to building models to add more variety to buildings and to increase realism. For example, the automated building models produced can include balconies (Figure 20) and porches. In a hotel, the algorithm locates balconies by associating the balcony with a room and sizing the balcony in proportion to the room width.

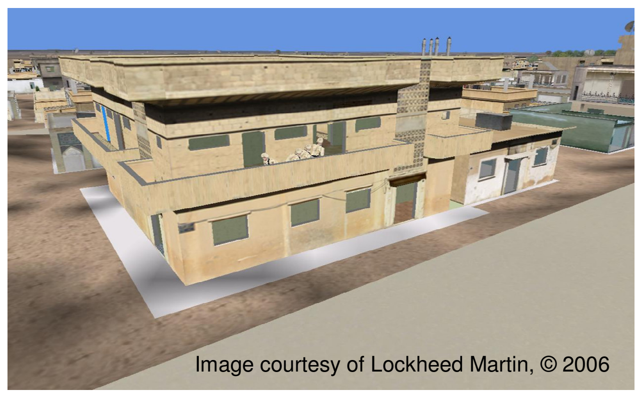

Figure 20. This building has balconies that can be used by SAFs for route planning through rooms.

For a virtual walk-through inside the building, varying the wall texture color increases realism and assists with navigation (Figure 21). A human looks for cues inside a virtual model to assist with spatial orientation. Cues may include color on the interior walls of rooms, pictures, or furniture and fixtures that are tailored to the purpose of the room.

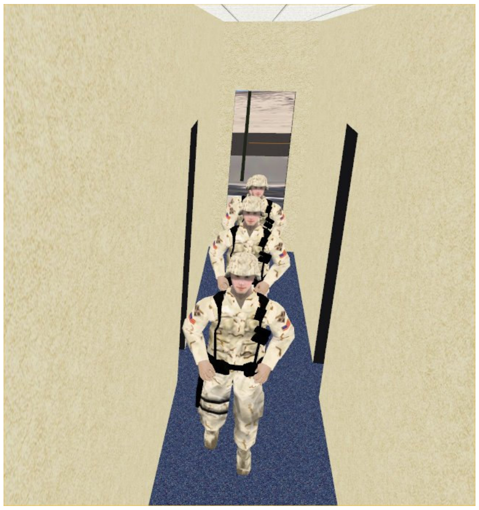

Figure 21. Textures and lighting assist with navigation

USING THE BUILDING GENERATOR

The building generator runs as a server process and is launched from a single client that spawns one or more instances of the building generator running on different servers. For large terrain databases covering hundreds or thousands of square kilometers, building terrain blocks in parallel is essential since it can take several days of continuous processing to generate a large database.

The client process accesses the building generator through an Application Programming Interface (API). Data is passed in XML format from client to server and the return file is a model in either attributed OpenFlight format or OneSAF Ultra High Resolution Building (UHRB) XML format.

Attribution of Model Elements

OneSAF terrain databases require attribution that identifies the location of rooms, connections between rooms, and the direction of stairs. These attributes are used by the SAF route planning algorithm to map out all possible paths that an entity may take through a building. The building generator adds this attribution and saves it with the exported model.

In addition to the SAF attribution, the building generator assigns material properties to each element in the building model. Walls have a thickness and properties that specify the construction material (e.g. concrete, wood, brick). These material properties and structural data are necessary for modeling the interactions of weapons and sensors with the building.

PERFORMANCE

We ran a test with the building generator using a dataset of Barstow, CA consisting of a shape file with 180 building footprints. All of the footprints were non-rectangular. We built a database using TERREX Terra Vista with the U2MG plug-in on a Pentium 4 3.6 GHz with 1 GB of RAM. The database build completed in 14 minutes, or an average of about 4.7 seconds per building. The output from this database build was an OpenFlight model and a SAF database in Compact Terrain Database (CTDB) format.

Most of the buildings in the Barstow database were the size of a typical house or small office building with two or three stories. The square footage of a floor has the most significant impact on performance because the layout algorithm considers more possible layouts as the square footage increases.

APPLICATIONS FOR URBAN DATABASES

Examples of current applications for these databases include the OneSAF Test Bed (OTB, Figure 22), the OneSAF Objective System (OOS) and the Joint Conflict and Tactical Simulation (JCATS), each of which uses a terrain database to model route planning and line of sight for vehicles and personnel entities. The military has used these simulations for experimentation (Graniela and Siddon, 2003), analysis (Pleban, Eakin and Salter 2000) and training applications (Shimamoto, 2000).

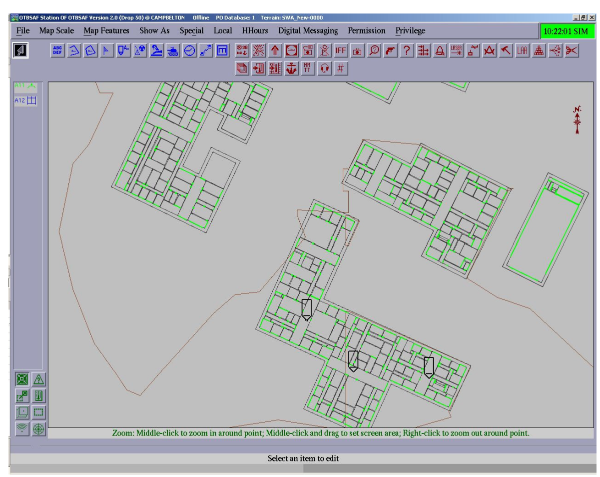

Figure 22. The OneSAF Test Bed (OTB) Plan View Display shows automated interior layouts for buildings in Multi Elevation Structure (MES) format.

Visual urban databases, correlated with a constructive simulation database are often used to present a visual display of simulated entities. These visual databases provide an out of the window view for vehicle simulators or a first person shooter view for dismounted troop simulators.

Urban databases with geotypical building interiors are used for urban operations training. Geospecific building interiors that represent actual building layouts are necessary for mission rehearsal. For both of these applications, the level of realism of the database will affect its usefulness. The visual cues provided by a high resolution terrain database are necessary for recognition and navigation purposes.

The use of urban databases is not exclusive to the military. City planners use synthetic urban terrain databases of metropolitan areas for planning purposes (Maese, 2003*)*. Planners are usually able to create these databases from existing GIS data including vector maps of city streets, digital ortho photos and shape files containing commercial building footprints.

CONCLUSION

Automated building generation is a key part of the solution for producing large scale, realistic urban simulations. Building generators make it possible to rapidly create databases with thousands of unique buildings matching geospecific footprints. Future improvements to automated building generation will include automated creation of geospecific buildings from sensor data and creation of models that more accurately reflect real world urban areas.

The building generator API discussed in this paper is available by request from RDECOM/STTC for government projects. Contact the authors for more information.

REFERENCES

DARPA. VisiBuilding: ISR for Building Interiors (2005). Broad Agency Announcement 06-04.

Graniela, B., Siddon, D. (2003). Development Of Future Combat System Synthetic Natural Environments. Proceedings of the Interservice/Industry Training, Simulation and Education Conference.

Harman, W., York, A. (2003, Spring/Summer). Integrated Munitions Effects Assessment: A Weapons Effects and Collateral Effects Assessment Tool. NBC Report, 30-37.

Maese, Kathryn. (2003). A Virtual Sell: 3-D Imaging Program Makes City Planning a Lot More Real. LA Downtown News, 22-Dec.

Mann, J., Pigora, M., Kraus, M. (2004). A High Resolution Urban and Underground Model Generator. Simulation Interoperability Workshop, Spring 04 Proceedings.

Pace, M., White, B. (2003). Rapid Building Generation for MOUT. Simulation Interoperability Workshop Proceedings.

Pleban, R. J., Eakin, D. E., & Salter, M. S. (2000). Analysis of mission-based scenarios for training soldiers and small unit leaders in virtual environments (Research report No. 1754). Alexandria, VA: U.S. Army Research Institute for the Behavioral and Social Sciences.

Shimamoto, F. (2000). Simulating Warfare is no Video Game. Science and Technology Review.