Development of Southern Miss's Innovation and Commercialization Park Virtual Reality Environment

Mohd Fairuz Shiratuddin Desmond Fletcher
University of Southern Mississippi
Hattiesburg, MS 39406, USA
{mohd.shiratuddin, desmond.fletcher}@usm.edu

Abstract
This paper discusses the development of a real-time Virtual Reality (VR) application for the University of Southern Mississippi's proposed university-related research and technology park. The park is known as the Hattiesburg "Innovation and Commercialization Park" (ICP), located in the City of Hattiesburg. The VR application is developed based on real-world CAD data, then processed using 3D modeling software, and implemented in BuildIT. BuildIT is a modified version of Garagegames' Torque Game Engine (TGE) which has been modified for architecture and construction visualization purposes. BuildIT maintains all the features currently available and adds database functionality in addition to Digital Elevation Map (DEM) terrain generation tools, quick project creation tools and compression tools. The development of the ICP's VR environment involves four main stages: terrain generation, development of road systems, development of parking lots and modeling of buildings. In each stage, techniques employed and the challenges encountered are discussed. The intended main goals of the ICP's VR environment have been accomplished which are: 1) the VR application to be able to run on average desktop computer, and 2) the VR application to maintain an acceptable minimum of 30 frames per second of image generation at any given time.

Keywords
BuildIT, game engine, terrain, virtual reality

1. INTRODUCTION

In this paper, we discuss our experience in developing a real-time Virtual Reality (VR) application based on given real-world CAD data. The VR application displays The University of Southern Mississippi (Southern Miss) proposed university-related research and technology park in a flythrough mode. This park is known as the Hattiesburg Innovation and Commercialization Park (ICP), located in the City of Hattiesburg, Mississippi. The site is a 500-acre parcel of land located in the northwest quadrant of the city. The park is owned by the Institute of Higher Learning, an arm of the State of Mississippi, and is controlled and managed by Southern Miss. It is located just southwest of the I59/US49 interchange and six miles to the northwest of the Southern Miss campus.

The ICP site has a gently rolling topography and is largely undeveloped. It is bisected by Classic Drive, a two-lane east-west road that provides access from I59 via US Highway 49. The area to the South of Classic Drive is occupied primarily by the Van Hook Golf Course, which was operated by Southern Miss, and has a relatively open character dominated by views of Lake Sehoy and the surrounding golf fairways. A portion of the Lake's northeast shoreline is occupied by a small park that provides outdoor recreation and picnicking. In the eastern portion of the site, south of Classic Drive, the university maintains some field of operations, including a geology park, some research space, and an observatory. The land to the north of Classic Drive is heavily forested, and in contrast to the area to the south, has a "closed" character, with no long views (Figure 1).

This paper introduces BuildIT, a modified version of Garagegames' Torque Game Engine (TGE). Next, the paper describes the development of the VR environment, which involved four main stages: terrain generation, development of road systems, development of parking lots and, modeling of buildings. In each stage, techniques employed and the challenges encountered are discussed. This paper then concludes with some proposed future work to improve Southern Miss' ICP VR environment.

2. BUILDIT

The current state of 3D game industry has evolved to be not only applicable to video games and entertainment industry, but also education and real world visualization applications. The Torque Game Engine (TGE) is a commercial grade game engine which is seeing enormous success in bridging these multiple industry sectors. TGE was originally developed by a computer game development company called Dynamix. Dynamix created computer games titles such as Earthsiege, Starsiege, Tribe and Tribes 2 (Maurina III 2006, Finney 2005). The founders of Dynamix then created GarageGames and currently distribute TGE as a product for independent game development.

Although there are a number of free and commercial game engines available in the market, without access to the entire source code, it is almost impossible to further expand and modified the engine, to include features and requirements that can be used by the design and construction industry (Shiratuddin and Thabet, 2003). Licensing cost of many commercially available game engines is still high and to date it is not cost effective for many academic institutions. However, in the case of TGE, licensing only costs $100 per programmer. The license includes access to the entire source code that will allow for further modifications, and online community support. This is the premise of BuildIT and it is currently being used for the architectural visualization purposes, both in class and research. BuildIT is primarily intended as a real-time VR scene design application for design students and architects. While BuildIT maintains all the features of TGE (GarageGames, 2006), some additional implementations have been made to it. BuildIT has preliminary support for database access, has a Digital Elevation Map (DEM) terrain generation tool, and a quick project creation and compression tools. The current implementation of the DEM terrain generation tool for BuildIT allows a satellite image to be imported, processed to produce the required heightfield bitmap image, and then imported into BuildIT for the 3D terrain to be generated. SQL database access is still under development, and once completed can allow many other design and construction related applications to be developed. Another feature that has been developed for BuildIT is the quick project creation and compression tools. The tools are meant for portability and easy sharing purposes. In BuildIT, a user can create a VR scene (or a project) and once completed, can simply zip-it up and give it to another user. The user does not need to know the directory structure, and where each file is located. This is one of the valuable features of BuildIT. BuildIT has been tested and supports VR devices such the eMagin Z800 head-mounted-display (HMD), the Intersense tracking device, the p5 data glove, the Logitech Spaceball 3D navigation input device, and also a few game input devices. 3D stereo display is still being tested utilizing third-party stereoscopic API.

3. DEVELOPMENT OF SOUTHERN MISS' ICP VIRTUAL REALITY ENVIRONMENT

In this project, we use BuildIT to develop the VR environment of Southern Miss' proposed ICP. Southern Miss' ICP includes an area of approximately 1 square mile. To develop such a VR environment, proper planning and work-flow is required. We were provided with a CAD file that represents the ICP site and surrounding areas including the topographical layout/contour lines, the footprints of the buildings, and the road systems (Figure 2), a document containing satellite images, CAD printouts, and descriptions of the ICP.

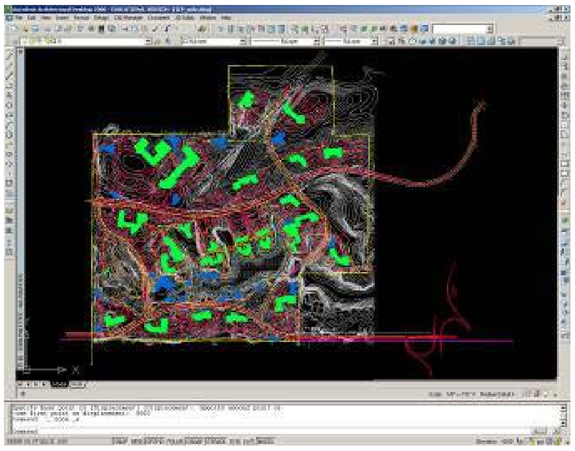

Figure 2: A screenshot of the CAD drawing provided

In the development process, AtLast's SketchUp, Autodesk Architectural Desktop, Autodesk Viz, and BuildIT were used. Autodesk Architectural Desktop was used to open and export the CAD file for use in SketchUp and Viz. The components in the CAD file such as the road systems, buildings' footprints, etc. were drawn in separate layers, thus making it simpler to process the required components by simply locking onto them, and hiding or deleting any unwanted layers. The development of the ICP's VR environment is divided into 4 main development stages; terrain, road systems (both highways and small roads), the parking lots and the buildings.

3.1 Large Terrain Real-Time 3D Visualization

Terrains generated from CAD software e.g. Autodesk Map, Max or Viz, although very realistic, may prove impossible to be processed and rendered in real-time due to the high number of polygons and the rendering capability of the average video card. TGE uses an internal terrain rendering engine with level of detail (LOD) correction that overcomes this challenge and provides a very fast, real-time terrain rendering that works on very low-end video cards. There are numerous techniques to generate terrains in TGE. After testing the use of a complete 3D mesh model representing the terrain, the approach taken in this project was to use a heightfield image map derived from the provided CAD files.

First, the original CAD file was referred to and opened using Autodesk Architectural Desktop (ADT). There were two layers of the 3D contours. In ADT, the contour layers were locked and the rest of the layers were deleted; the file was then saved as a new CAD file. The ICP's project boundary is approximately 1.034 square mile. Using the contour CAD file, a 5,459.52 square foot/1.034 square mile polyline was drawn to represent the project boundary. The two contour layers were grouped together into one single layer and edited to remove any unwanted contour lines, contours that extended beyond the project boundary, and any other unnecessary components that still existed.

The newly created CAD file now contains only the 3D contour lines and the project boundary. This new contour CAD file was imported into Autodesk Viz and a 3D mesh model of the terrain was developed using Autodesk Viz. The resulting 3D terrain mesh had too many polygons (approximately 980,000) hence was impossible to be used and rendered in real-time.

Alternatively, a heightfield image map can be created from existing 3D contours using either Autodesk Viz or AtLast SketchUp. A heightfield image map creates vertically deformed surfaces based on the light and dark values of a bitmap image which is generated from the contours. The goal of using this technique is to create a true heightfield image map representation of the ICP's terrain, import it into BuildIT, and then let BuildIT generate the corresponding 3D terrain.

In producing the heightfield image map, two more workflows were tested; one with AtLast SketchUp and the other with Autodesk Viz 2006. First, the CAD contours were imported into SketchUp. The SketchUp "sandbox contour" tool was used to create the surface of the terrain. However, before the sandbox tool can be used, the contour lines had to be exploded into individual polylines. A new problem occurred because once exploded, it was impossible to work with all the individual polylines. There were simply too many polylines for SketchUp to process since all the individual polylines then had to be manually pieced and connected to one another in order to create a surface. The workflow using SketchUp's sandbox tool ended up with the CAD file exploded into 250,000 very small polylines (Figure 3) and a heightfield map image was not able to be efficiently created.

However, using Viz to create the heightfield image proved very efficient and the process is described below. The CAD file that contains only the contour lines in was opened in ADT, and ADT was directly linked to Viz through the 'File Link Manager'. With this feature, any changes made in ADT will be reflected in Viz. (This file linking feature exists in many of Autodesk's software products). Based on the linked contours, Viz then generated the heightfield image which was exported in a BMP format. The BMP was then converted to PNG format for use in BuildIT.

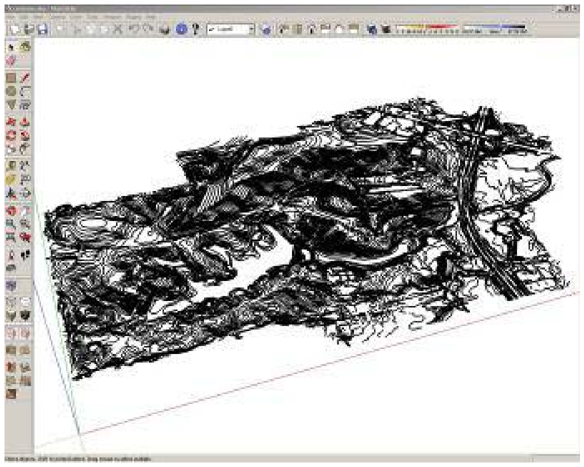
Figure 3: The exploded polylines in SketchUp

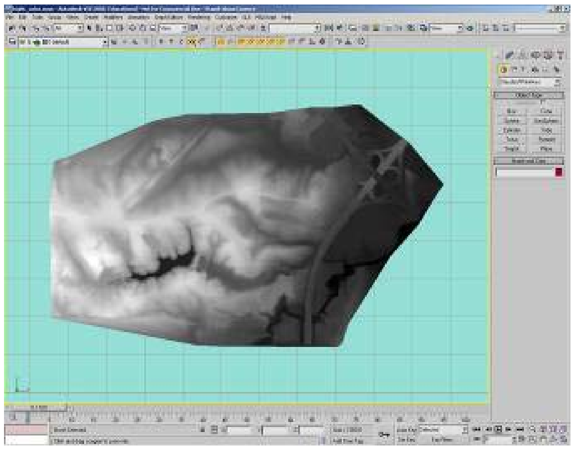
Figure 4: The 8-bit grayscale displacement map of the ICP's terrain

Based on the given CAD data, the difference in elevation between the highest (262') and lowest (150') point of the ICP terrain is 112'. In BuildIT, to create a terrain from a displacement map, an 8-bit grayscale is required. Since the ICP's terrain is mostly flat, the 8-bit grayscale bitmap was used, divided into 16 gray color-shifts with each shift differing at 7'. The terrain was colored with a 7' shifts starting at 150' with RGB 0,0,0 color value (black) and going up to 262' with RGB 255,255,255 color value (white) (Figure 4).

In BuildIT, a new scene was created and the 8-bit grayscale heightfield/displacement map was imported into BuildIT. Using BuildIT's terrain builder, a 3D terrain was generated and the scene file was saved (Figure 5). Since BuildIT supports terrain generation from a heightfield/displacement map, it was concluded that this technique was more appropriate for this purpose. There was no noticeable reduction in frame-per-second rendering.

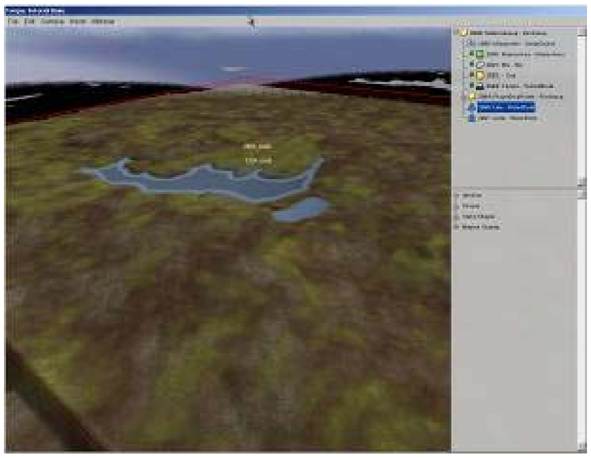

Figure 5: Terrain generated in BuildIT

3.2 Road Systems

The next step in developing the ICP's VR environment was to model and place the road systems. Two methods were tested whereby a CAD file that contains only polylines of the roads was used (Figure 6). In the first method, the polylines were imported into SketchUp, and once in SketchUp an individual roadway was selected. A profile of each road segment was created by utilizing the center line of the selected roadway and one end. Using the 'follow-me' tool in SketchUp, the profile was pulled along the length of the entire road system (Figure 7). This method worked out quite well, however it created much too high a polygon count to be useful enough for real-time rendering purposes. The entire road system had approximately 90,000 polygons due to all the road's curvatures and the changes in elevation. The resulting 3D mesh model of road was unusable, because the polygon was just too high for real-time rendering purposes.

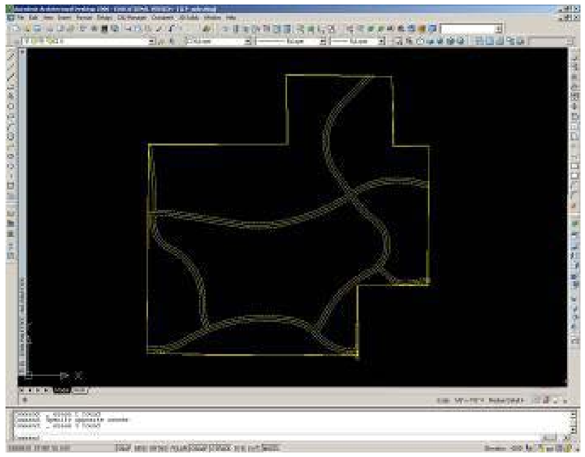

Figure 6: Polylines of the road systems

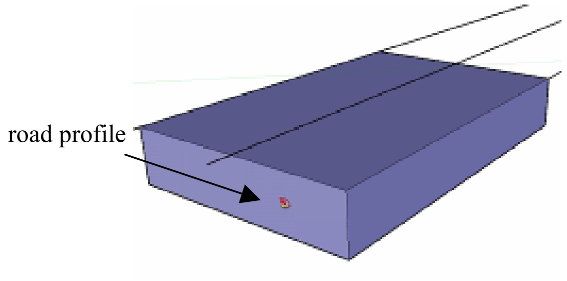

Figure 7: SkecthUP 'follow-me' tool to generate the road profile

Again using SketchUp, a second approach was used by creating a totally flat road system that consists of only 2D planes/faces, and ignoring the changes in elevation. In this case, it was achievable since the ICP's landscape is almost flat with very minimal change in elevation. For landscape that has dramatic changes in altitude, this technique may not be suitable.

The 2D faces for the entire road system was slightly extruded to give the roads some depth and made them into a 3D object. The 3D road model was then converted into BuildIT's static mesh DTS (Dynamix Three Space) file format. The conversion of any 3D model into BuildIT's static mesh can be done using either Autodesk Viz (a plugin is freely available), Chumbalasoft's Milkshape (native support for DTS) or other 3D modeling software (Maya, LightWave, Blender3D, GameSpace) that support exporting to the DTS file format. The scene that contained the ICP terrain was then opened in BuildIT. The DTS road file was imported into the scene as a static mesh model, and placed at the corresponding georeferenced location. The scene was saved and tested.

At this point, it was discovered that at certain viewpoint, the road system disappeared and reappeared erratically. BuildIT's rendering engine automatically changes the view quality of objects based on distance from the viewer; in this case, the road mesh was so large that a conflict developed within the renderer. In real-time graphics rendering technique, predicting and not rendering parts of the scene that are out of the user's field of view reduces the load of the graphics processor. This prediction is based on the objects location within the "world's" octant space. If an object spans all octant, the rendering can not be optimized. The road system had to be remodeled again and this time the road systems were broken down into smaller sections of 150' to 200' each. The individual sections were converted into different DTS static meshes and reimported into BuildIT. The different sections of the roads were placed into the scene (Figure 8) and the disappearing/reappearing issue of the road system at any viewpoint of the user was no longer an issue.

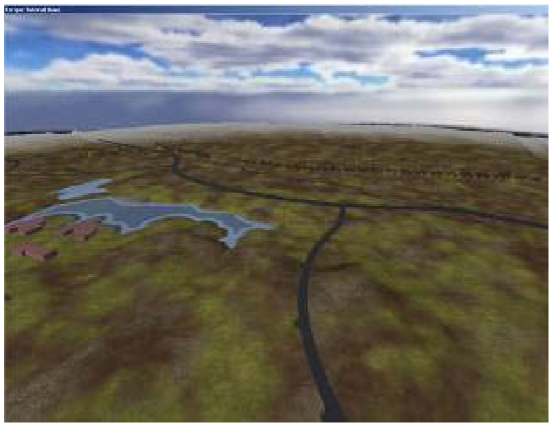
Figure 8: The ICP's road systems rendered in BuildIT

3.3 Parking Lots

The ICP's VR environment is intended for flythrough, and viewed at a certain height only. Based on this, a high level of detail and realistic representation of the parking lots was not a concern. Hence, a different approach was used to develop the parking lots which consist of concrete curbs or various sizes and shapes. Creating 3D models of the curbs can unnecessarily contribute to a high number of polygons to the VR environment that can slow down the real-time rendering process of BuildIT's rendering engine. As such, a transparent texture approach for all the parking lots was used. A rendered image of the surface of the parking lot was created and mapped as a transparent texture map on top of a square block. This dramatically reduced the high polygon count with each lot containing only 12 polygons (2 polygons of each surface of the square block). In order to create the texture map for the parking lots, a new CAD file was created that contained only the parking lots. The parking lots CAD file was opened in SketchUp and all the curbs were created as 3D models (Figure 9).

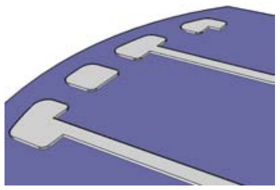
Figure 9: 3D model of the parking lot curbs

In SketchUP, viewing from the top view, each parking lot was rendered into JPEG images. To use a transparent texture image mapped onto a surface of a 3D object in BuildIT, two separate JPEG files were created; one containing the actual image and the other a black and white image representing the transparent alpha-channel (Figure 10 and 11 – shows the parking lot outline jpeg and alpha jpeg respectively). Once rendered in BuildIT, only the lines representing the curbs and parking spaces were displayed (Figure 12).

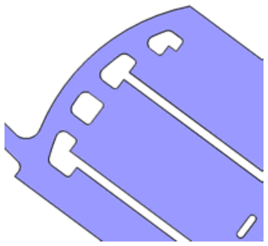
Figure 10: Parking lot outline

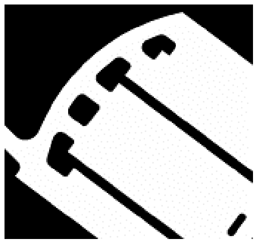
Figure 11: Transparent alpha-channel

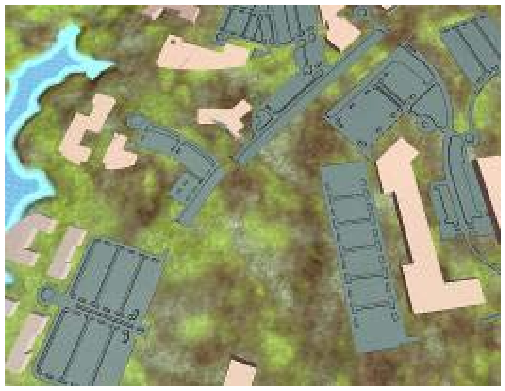
Figure 12: A screenshot of ICP's parking lots

3.4 Buildings

Similar to the initial steps taken in developing the terrain, road systems and parking lots, a new CAD file was prepared that only include the buildings' footprint. The buildings' CAD file was imported into SketchUp and in SketchUp utilizing the 'push/pull' tool, the building's profile was extruded to make it into a 3D mass model (Figure 13). Only 3D mass models were developed because the ICP's architectural style has yet to be developed. Detailed interior floor plans will be developed when clients for the individual sites are determined. Improvement will be made at that time to include more detailed designs of each building where users can freely walkthrough them. With the buildings fully in place, the development of the ICP VR environment was complete.

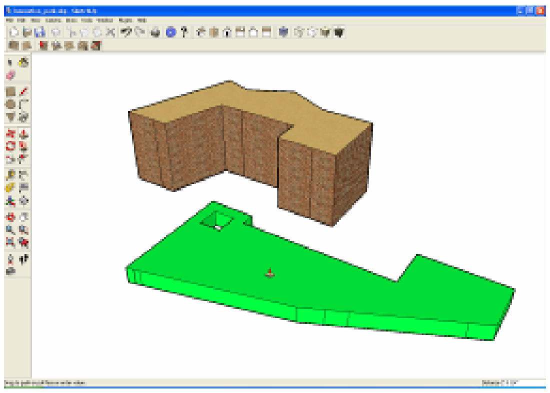
Figure 13: 3D mass model of the buildings

4. CONCLUSION AND FUTURE WORK

Shown in Figures 14, 15 and 16 are some of the screenshots of Southern Miss's ICP VR environment.

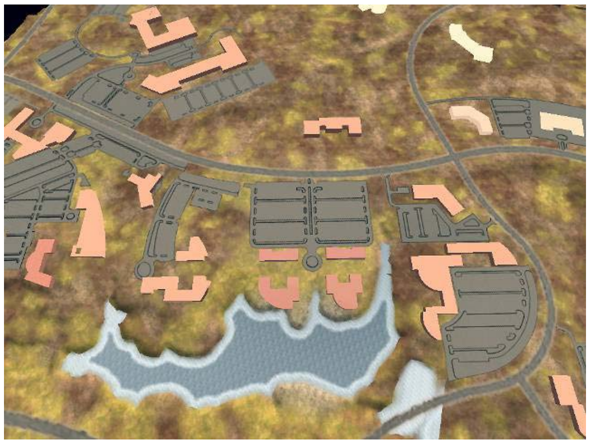
Figure 14: Screenshot of the ICP's VR environment

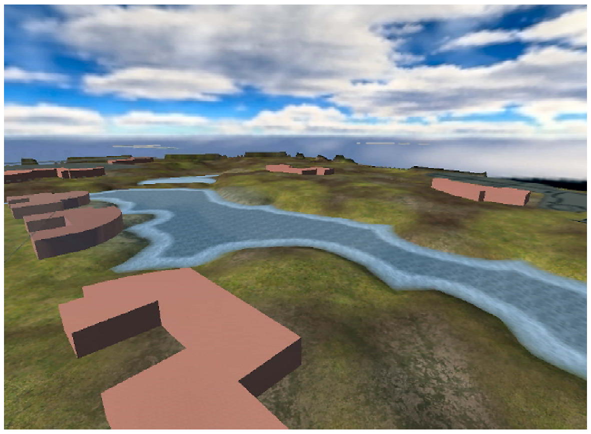
Figure 15: Screenshot of the ICP's VR environment

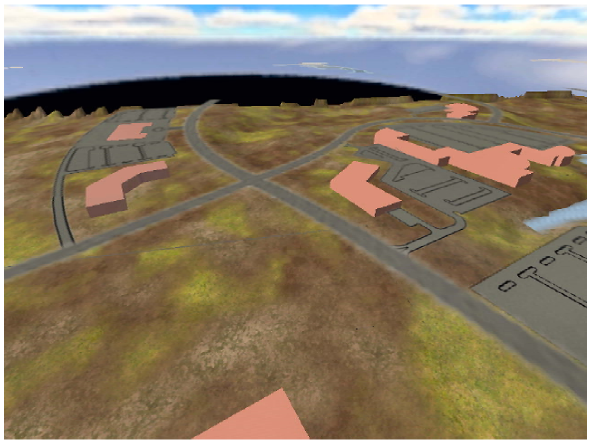
Figure 16: Screenshot of the ICP's VR environment

We have achieved the intended main goals of the development of Southern Miss's ICP VR environment which are: (1) the VR application is able to run on average desktop computer, and (2) the VR application can maintain a minimum acceptable 30 fps of image generation at any given time. The VR environment is able to be efficiently executed on an average desktop computer (with a Pentium 4 2-Ghz processor, 256Mb RAM, GeForce4 MX video card, and Windows XP installed) and the real-time rendering rate is maintained at an average of at least 30 frames-per-second at any given user's field of view. Below 30 fps, user will start to experience lag and notice imperfect renditions of the scenery.

One of the main challenges encountered during the development of the VR environment was in reducing the number of polygons of the 3D models. Given that it was targeted for the VR environment to be able to run on typical desktop computer, various methods were tested until a minimum satisfactory frame rate of 30 fps was achieved, without sacrificing too much of 'the realism' of the real-time scene. Reduction of the number of polygon in a 3D model can greatly speed up the real-time rendering process of a 3D scene in a virtual reality environment as the number of polygons is inversely proportional to the fps rendered in real-time. Utilizing the heightfield map image technique to generate the 3D terrain for the ICP and using simplified modeling techniques for the infrastructure, BuildIT had no rendering issue and drops in frame rates as its' 3D rendering engine and terrain engine was designed for such purpose.

It is planned that a walkthrough feature be implemented for Southern Miss's ICP VR environment. In this walkthrough feature, user will be able to walk into individual buildings, drive through the scene and interact with objects in the VR environment. A multi-participant VR environment is also planned to be developed.

5. ACKNOWLEDGEMENT

The authors would like to express their sincere gratitude to Dr. Cecil Burge, the Vice President for Research and Economic Development of Southern Miss for his support that made it possible for us to develop the VR environment for the Innovation and Commercialization Park. The authors would also like to thank Mr. Shawn O'Keffee, Mr. Robert Thornburg and Mr. Chary Srinivas for their assistance in this project.

6. REFERENCES

Finney, Kenneth (2005). Advanced 3D Game Programming All in One, Course Technology PTR, 1st edition, 2005, ISBN: 1592007333.

GarageGames (2006). The Torque Game Engine.
`http://www.garagegames.com/products/1#features`
Last accessed: May 2006.

Maurina III, Edward F. (2006). The Game Programmer's Guide to Torque: Under the Hood of the Torque Game Engine, AK Peters, Ltd., 2006, ISBN: 1568812841.

Shiratuddin, Mohd Fairuz and Thabet, Walid (2003). A Framework for a Collaborative Design Review System Utilizing the Unreal Tournament (UT) Game Development Tool, The 20th CIB W78 Conference on Information Technology in Construction Waiheke Island, Auckland, New Zealand, 23-25 April 2003.