Virtual Office Walkthrough
Using a
3D Game Engine

Mohd.Fairuz Shiratuddin (fairuz@vt.edu) & Walid Thabet (thabet@vt.edu)

Department of Building Construction
(http://www.caus.vt.edu)

Virginia Tech
VIRGINIA POLYTECHNIC INSTITUTE AND STATE UNIVERSITY
Blacksburg, VA 24061, USA
Tel: +1 (540) 818-4604 · Fax: +1 (540) 231-7339
1 Introduction

Currently, the common method for representing and distributing information about a construction facility to be built is via two-dimensional (2D) drawings and relevant specifications (Figure 1). Recipients of 2D drawings and specifications will extract the necessary information and interpret it based on their previous experience, background and knowledge. Each may have different understanding on how the facility will look like when completed, thus may affect the way the construction process will proceed. The misunderstanding may lead to mistakes which may further lead to additional time and costs. This is the challenge of delivering a facility today.

Construction projects are also characterized by having complex structures consisting of many parties with differing levels of interests, knowledge, expertise and resources. Each party is complementary of one another and has the responsibility to ensure that their designated tasks are completed in time so that subsequent work by others can follow until the final product is in place. Thus the multi-participant nature and the need for collaboration in construction projects are inevitable.

Figure 1: A typical 2D drawing used to relay building information
(Courtesy of Pulte Homes)
Virtual walkthrough applications provide great benefits for visualizing of construction projects during design and actual construction. Virtual walkthroughs can allow participants to perform design/construction review tasks collaboratively, while locally present, or remotely connected. Collaboration among participants from the initial design stage is important because critical decisions can be made as many and as early as possible to lessen disputes, delays, cost overrun etc. at later stages (Waly and Thabet, 2001). Visualization of the desired end product in a virtual environment with real-time walkthrough capabilities will also allow for such collaboration among project participants and will assist each party to decide on the means and methods of achieving the project goal.

This paper describes an approach for developing an affordable and realistic three dimensional (3D) visualization walkthrough application using a game engine. 3D Game Engines have the capability to represent a virtual environment in real-time and realistically. Its walkthrough ability allows user to navigate a virtual environment as if in the real world. Other major advantages offered by present 3D Game Engines are low-cost, networking support, collision detection, support for high frame rates per second, and entry level hardware requirement. The 3D Game Engine technology is currently being used in an ongoing research at the Department of Building Construction at Virginia Tech, USA.

1.1 The 3D Game Engine

The 3D game engine that is chosen for this research is the Unreal Tournament, a currently popular 3D First-Person-Shooter (or sometimes referred as a first person perspective) game engine. The game engine was chosen because it encompasses good quality built-in features provided by its developer, Epic Games (http://www.epicgames.com). The built-in features allows for producing realistic virtual environments, thus to create a new system will mean 'reusing of the already existing invented wheel'. The game engine has built-in tools which also allow alterations and extensions to be made to the game.

Unreal Tournament was released in December 1999. It adopted the 'open architecture' approach where partial codes to manipulate the game engine and updates are made available by its developer via the internet. The concept allows on-line communities (typically game enthusiasts) to build up large knowledge base for the game playing, editing, level (or map) building and distribution. The community also provides free support and tutorials for code alterations, hints and tips for the larger benefits of each other. The Unreal Tournament game comes bundled with its own Games Development Kit (GDK). To access and make customization to the Unreal Tournament engine, a friendly graphical-user-interface (GUI) level editor, known as the UnrealEd 2.0 is provided with the GDK (see Figure 2). The game engine application is used on a Dell Precision Pentium III Xeon computer with Microsoft Windows XP Professional installed.
1.2 Other Hardware

Kalawsky (1996) stated that Virtual Reality (VR) systems need a device that can do at least a 6-DOF (degree of freedom). In supporting Kalawsky's view, an affordable motion sensing gaming device (the Microsoft Sidewinder Freestyle Pro gamepad - see Figure 3) that supports a 6-DOF movement is used for this research to assist the user. The device allows the user the freedom to move, turn and look in real-time in any direction in the virtual environment.
1.3 First and Third-Person Perspective

The interface to Unreal Tournament game is based on the first person perspective principle where the user sees the world through the eyes of the game's main character (Tabor, 1997). The user and other characters in the game environment are known as 'players' a concept that is similar to 'avatars' in a virtual environment (see Figure 4). The user can interact with other avatars in the virtual environment through text-chatting, built-in 'taunt', and recently through voice as well. Using mouse clicks, a keyboard or a joystick, users can freely roam around and make head movements, which enable a large degree of visibility in the virtual environment.

The Unreal Tournament by default displays the First Person Perspective view. However, through a built-in console type command third person perspective view can be enabled (see Figure 5). Console command can be issued through the console window.

To type specific console command in the console window user can press the ` key and then type it in. The command to enable third person perspective is `behindview 1'. Unreal Tournament has various types of console commands. Third person perspective is useful where user can see one-virtual-self with respective to the virtual world.

Figure 4: Samples of `avatars' in Unreal Tournament based engine

Figure 5: The third person perspective view
1.4 Games Customization

Until a couple of years ago, game developers had two basic choices: develop their own in-house engines, or license those that are game-specific (Cramblitt, 1999). Independent graphics software developers saw an opportunity to provide more general-purpose tools that were much less expensive, could be used for different styles of games, would be supported by developers, and would evolve in concert with new technologies and the needs of users. These tools can be divided into two basic categories:

- Real-time renderers and graphics software development kits (SDKs) designed for games and other interactive applications (e.g. OpenGL and Direct-X).
- General-purpose 3D game engines, which go beyond rendering to offer a range of tools designed specifically for game development. This includes high-level object databases with culling, collision detection, animation, 3D sound, scene management, and interfaces to 3D modeling and animation programs.

The Unreal Tournament game can be customized, through extension or alteration of the games elements. This includes:

1. **Creating new maps or levels using the tools provided by the developer.**
2. **Altering the code of the game (i.e. to change, add or remove items, weapons, monsters, avatars and some other aspects of the game).**
3. **Changing the appearance of the avatar either using the template 3D character model files provided, downloading ready-made models or using modeling tools provided by the on-line community.**

The following section discusses the development of the virtual office. This is followed by a section on some real world applications using the Unreal Tournament engine, the advantages and current shortcomings of the game engine and finally a conclusion.
Figure 6: The overall process to develop a real-time virtual walkthrough application from an existing 2D floor plan
2 Developing the Virtual Office

The development process of the virtual office environment follows the sequence shown in Figure 6. It begins with a 2D AutoCAD *.DWG drawing file format (see Figure 7) which was then transformed into a 3D model using 3D Studio VIZ. To import the *.DWG file into 3D Studio VIZ, the settings shown in Figure 8 have to considered to ensure that no model information is lost or misrepresented. It is important to enable ‘Unify Normals’ as it tells VIZ to align all of the models ‘normals’ to point outward.

Figure 7: The 2D floor plan of the office
2.1 From 2D to 3D

Once the *.DWG file is in 3D Studio VIZ, the next step is to extrude the walls. The polyline outlines that have been created in AutoCAD are converted to `closed splines' in VIZ. Then the `Extrude' modifier is applied to the wall splines to extrude it in the z-axis (heights) to create the 3D walls. Doors and windows are then created by “subtracting” them from the 3D walls. The 3D walls with doors and windows are shown in Figure 9.

Once 3D modeling is completed the model is then exported to the AutoCAD *.DXF file format. This will allow for importing the 3D model into the game engine’s environment since the engine’s editor i.e. UnrealEd 2.0 supports *.DXF file format. Once in UnrealEd 2.0 environment adding the game engine’s features are possible.
2.2 Importing into UnrealEd

Shiratuddin and Zulkifli (2001) indicated that the UnrealEd 2.0 can only import not more than 500 polygons of *.DXF model at any one time. Hence to import an entire 3D model with a high number of polygons into the game engine’s environment is not possible. A technique that can be used to overcome this limitation is break down the model into smaller components (see Figure 10). In 3D Studio VIZ, the components with the same properties (e.g. windows, pillars, doors, walls etc.) are grouped together and saved as *.DXF file format. The groups are then imported one at a time into the UnrealEd 2.0 engine’s editor to reconstruct the model and to add features of the Unreal Tournament engine. Importing the components of the model into segments (or layers) into the UnrealEd 2.0 does not pose any geometrical misplacement problems. This is because UnrealEd 2.0 maintains the relationship between the X, Y and Z-coordinate of each component and a fixed reference point in the model.
2.3 Adding Features

Once the 3D modeling is completed and imported into the Unreal Tournament game engine, the UnrealEd 2.0 is used to add features. The Unreal Tournament game engine supports several features to enhance the virtual office walkthrough application.

2.3.1 Texturing

Texturing or texture mapping is a process where a 2D surface (called a texture map) is "wrapped around" a 3D object. Thus the 3D object acquires a surface texture similar to that of the 2D surface applied. Texture mapping to some extent is similar to applying wallpaper, paint, or veneer to a real object. Texturing for the virtual office building includes wall, carpeting, ceiling, outdoor, grates etc. Figure 11 shows some of the texture maps used for virtual office.

Figure 11: Textures applied to the virtual office
2.3.2 Real-World Effect

Incorporated within in the Unreal Tournament game engine are believable imitations of real-world elements that can be applied to the virtual office environment. This includes real-time graphics, movement, and sound "on the fly" while the game is played (Elin, 1999). The game engine can produce real-time special effects such as blazing fire, hot flowing larva, shimmering water, moving clouds etc. With all of these elements already built into the game engine, the requirement of coding lengthy new scripts are no longer required (see Figure 12 for animated effects).

![Figure 12: Real-time effects incorporated into the Unreal Tournament engine](image)

2.3.3 Audio Effects

In games, audio (foreground and background) plays a big role in providing good atmospheric sound effects and music to compensate all the actions in the game (Shiratuddin et al, 2000). For the virtual office environment prototype, audio effects were used for user’s footsteps and as background music.

2.3.4 Real-Time Lightings

Good lighting effects can provide realism and atmospheric effect to a virtual environment. Without proper lighting conditions, a virtual environment will be insipid and monotonous.

The Unreal Tournament engine supports a variety of lighting conditions. It has Point, Dynamic, Fog, Spotlights (Search), Shimmering, Coronas and Wavering lighting effects. All the lighting effects in the ‘Unreal Tournament game’ are all built-in into its game engine. Table 1 shows the various types of lighting present in the Unreal Tournament engine. Figure 13 shows some samples of lighting effects and Figure 14 displays the virtual office with several of the lighting options implemented.
Types of Lighting

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Dynamic</td>
</tr>
<tr>
<td>2</td>
<td>Colored</td>
</tr>
<tr>
<td>3</td>
<td>Point</td>
</tr>
<tr>
<td>4</td>
<td>Radiosity</td>
</tr>
<tr>
<td>5</td>
<td>Spotlight (Search)</td>
</tr>
<tr>
<td>6</td>
<td>Coronas</td>
</tr>
<tr>
<td>7</td>
<td>Fog</td>
</tr>
<tr>
<td>8</td>
<td>Shimmering</td>
</tr>
<tr>
<td>9</td>
<td>Wave</td>
</tr>
</tbody>
</table>

**Table 1: The 3D game engine lighting effect support**

**Figure 13: Lighting effects in Unreal Tournament**

**Figure 14: The fully lighted virtual office**

Once all the features applied to the newly created virtual office environment, it was compiled and executed.
3 Real-World Examples

Although the use of game engines for developing virtual environments is still not very common, several good examples of successful real-world projects that have utilized this technology exist (Shiratuddin & Thabet, 2001). Since 1998, companies and researchers throughout the world have undertaken some major successful projects. The projects utilized 3D Game Engines to create highly realistic, believable and fascinating virtual worlds. Table 2 lists some of the major VE projects that utilized a mixture of 3D CAD models and 3D Game Engines.
<table>
<thead>
<tr>
<th>Year</th>
<th>VE Project</th>
<th>GDK</th>
<th>Developer</th>
<th>Description/Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1998</td>
<td>Virtual Florida Everglades National Park</td>
<td>Unreal</td>
<td>Project leader: Victor DeLeon</td>
<td>A project to educate the public and also promote ecological awareness</td>
</tr>
<tr>
<td>1999</td>
<td>Long Island Technology Center</td>
<td>Unreal</td>
<td>Perillith Industrielle for Rudin Management</td>
<td>Demo can be downloaded at <a href="http://www.unrealty.net">http://www.unrealty.net</a>.</td>
</tr>
<tr>
<td>1999</td>
<td>HypoVereins Bank</td>
<td>Unreal</td>
<td>Perillith Industrielle for Turbo D3</td>
<td>Virtual bank in Germany. Demo can be downloaded at <a href="http://www.unrealty.net">http://www.unrealty.net</a>.</td>
</tr>
<tr>
<td>2000</td>
<td>Virtual Graz of Austria</td>
<td>Unreal</td>
<td>Bongfish</td>
<td>Graz is the second largest city in Austria. Funded by UNESCO.</td>
</tr>
<tr>
<td>2000</td>
<td>Cambridge University and Microsoft Science and Technology site in West Cambridge</td>
<td>Quake 2</td>
<td>Martin Centre for Architectural and Urban Studies, Cambridge University</td>
<td>Part of a project on using electronic communication between buildings' architects and their eventual users.</td>
</tr>
<tr>
<td>2000</td>
<td>CAVE Quake 3</td>
<td>Quake 3</td>
<td>Quake 3 Visualization and Virtual Environments Group, NCSA.</td>
<td>A CAVE system based on the Quake 3 Arena engine. Web-site at: <a href="http://www.visbox.com/cq3a/">http://www.visbox.com/cq3a/</a>.</td>
</tr>
<tr>
<td>2001</td>
<td>CAVE UT</td>
<td>Unreal Tournament</td>
<td>Medical Virtual Reality Center, Department of Otolaryngology, University of Pittsburgh</td>
<td>A CAVE system based on the Unreal Tournament engine. Web-site at: <a href="http://www2.sis.pitt.edu/~jacobson/ut/CaveUT.html">http://www2.sis.pitt.edu/~jacobson/ut/CaveUT.html</a>.</td>
</tr>
</tbody>
</table>

Table 2: Major VE projects utilizing 3D Game Engines
4 Advantages and Disadvantages

Game engines in general are characterized by their low-cost and their availability off the shelf from many computer stores, hence making these applications affordable for developing VR applications. The following section discusses the advantages of utilizing 3D game engines, their current shortcomings, and some thoughts on how to overcome them.

4.1 Advantages

The ability of the game engine to allow for the development of real-time walkthrough application can provide the owner of a project the ability to freely inspect the virtual facility prior to complete design or construction and can better set realistic expectations on the final product, rather than just viewing representations in the forms of 2D drawings, static image rendering or fixed-path animation.

The walkthrough application allows users and all participants to feel as though they are there, walking through space, able to move up stairs, peering out windows etc. This application is also able to give them a real scale of the facility, space, and furnishing.

4.1.1 Real-Time Walkthrough

Realism and details in virtual environments are achieved through the process of adding 3D qualities such as shadows, colors and shade variances. According to Campbell and Wells (1994), the criterion that makes virtual reality closer to reality is the ability to allow "immediate, direct, and more intuitive control over a three-dimensional design".

4.1.2 Realistic Visualization

The game engine meets the criteria suggested through its built-in tools which are capable to enhance realism. Realistic environment visualization is achieved through the use of photo-realistic images for texturing, real-time lighting, shadows, real-time reflective and mirrored surfaces and marble shine. Realism is further enhanced through interactivity and audio effects.
4.1.3 Lighting

Resembling the real world, lighting provides the sense of security and confidence in occupying or maneuvering an enclosed space. The game engine has the 'dynamic lighting' feature. This type of lighting can be seen in the game environment when a weapon is fired; the blast of the fire can cast light off the walls and surrounding objects. The second type of lighting includes the shadow effect, where the engine will automatically show any shadow that should occur in reality when light is blocked by an object.

Other types of lights that the game engine supports are colored lighting. It also has the additional allowance for new modifications on new types of lights e.g. radiosity lighting is used to produce glowing lava effects. Point lighting is the basic form of lighting in the engine and it can also be used to produce other types of lighting effects.

4.1.4 Collision Detection

An outstanding criterion of most 3D game engine (including Unreal Tournament) is the built-in 'collision detection' ability. Current commercial VR tools require the collision detection feature to be manually defined by the user for the specific graphical element or solid object. The Unreal Tournament engine however has a built-in tool for collision detection that automatically prevents users from 'walking through' walls, doors or other object where a collision should occur. The game engine is able to automatically detect when user collides with solid or non-solid objects, thus users will experience the 'bump' effect as in real life.

4.1.5 Interactivity

The term interactivity means the interaction between computer and user which takes place through changes of viewed locations, typed commands, voice commands, mouse clicks, or other means of interfacing. The Unreal Tournament engine accepts and responds to user activity in real-time at an interactive rate of at least 30 frames-per-second (fps). This feature is incorporated and can be visualized and experienced in the virtual environment of the office building.

Interactive features are important for users of the virtual environment as it will relate what they are seeing in the virtual environment to the real world. This feeling of realism is important to convince users that the environment is realistic and represent the real world (Mays, 1998 and Miliano, 1999).
4.1.6 Polygon Count/Frames per Second

Mullen (1998) suggested, to benchmark the performance of a 3D game engine running on a computer is by measuring the number of images it can generate on-screen per second (i.e. fps). The virtual office running on the Unreal Tournament engine achieves a minimum fps of at least 40 frames (See Figure 15).

Normally, using other virtual reality tools, when a model is developed, the complexity increases as the level of realism increases. Increase in real-time rendition will drop the frame-rate of a real-time walkthrough quite noticeably. Campbell and Wells (1994) indicated that the frames per second generated will drop from 15 to 3-4 fps in a 3D model when the polygon counts exceed 10,000. This situation is unacceptable for presentational purposes, and makes inspecting the model disorienting and difficult (Miliano, 1999).

The Unreal Tournament game engine can generally maintain an interactive 30 fps image rendering. It can also handle approximately 60,000 polygons in a single level with full lighting and texturing. To overcome the 60,000 polygons limit per level, interconnected levels are supported and with this feature users can go back and forth between levels.
4.1.7 Avatars

In VE, avatars are user defined geometry forms (Vince, 1998) that can either possess intelligent characteristics (i.e. AI characters or bots) or just simply present a virtual representation controlled by the user's input. Avatars can represent project participants having a tour in a virtual facility in group or individually. Avatars can also be configured to become workers simulating the environment of a real facility when the building is occupied and in use. The environment can also be configured to simulate the event of fire where the avatars represent workers reacting to fire and finding fire escapes. Many other events involving avatars can be simulated using the built-in tools provided by the game engine. This helps the project team to plan ahead on future unseen possibilities regarding the facility.

4.1.8 Multi-Participants

According to Sweeney (1998), one of the strength of the game engine is its multi-participant networked capability. The inherent multi-user nature of the game technology lets clients connect to its server using the game's client software over the internet and LAN. Users can thus examine the design and facility in real-time. Using the game engine, a walkthrough-VR application can be leveraged before the final iteration of a design.

Users can represent themselves using the avatars and interact with each other in the virtual facility regardless of the geographical location. A useful means of communication when two or more avatars are within each other's view is to use basic gestures, chatting or talking. Further customization of all these gestures can be achieved with additional programming.
4.1.9 Console Commands

The Unreal Tournament engine has built-in commands to define some useful effects for virtual walkthrough purposes. In the Unreal Tournament game itself, console commands are issued to invoke ´cheat´ modes. These commands can be typed in using the console command window. To reveal the console command window, user has to press the ´`´ on the keyboard. Once the console command window (see Figure 16) is shown, user has to type in ´`iamtheone´ so that further commands will take into effect. Some of the useful console commands that can benefit a virtual walkthrough are:

<table>
<thead>
<tr>
<th>Command</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>ghost</td>
<td>disable the collision detection so users can walk through solid objects</td>
</tr>
<tr>
<td>fly</td>
<td>enable users to defy gravity and fly-through the environment</td>
</tr>
<tr>
<td>walk</td>
<td>goes back to normal walking mode</td>
</tr>
<tr>
<td>behindview 1</td>
<td>third person perspective view</td>
</tr>
<tr>
<td>behindview 0</td>
<td>first person perspective view</td>
</tr>
<tr>
<td>say [message]</td>
<td>broadcast text message to other users</td>
</tr>
</tbody>
</table>

Figure 16: The console type command window in Unreal Tournament to enable various commands
With all the features mentioned above it is therefore acceptable to use the game engine for developing a virtual environment because it has good quality presentation and will make inspecting a virtual environment manageable, controlled, and easy.

### 4.2 Shortcomings & Solutions

#### 4.2.1 Licensing

The Unreal Tournament is one of most popular engine and it can produce high quality virtual environments. However, licensing the Unreal Tournament engine costs approximately USD 350,000. This issue can be a hindrance and not cost effective for potential commercial users who are small sized construction companies or small scale construction projects. Yet for in-house research and development, non-commercial and educational purposes, the engine can be used without the need for licensing.

There are also alternative game engines in the commercial market that can be afforded by prospective users e.g. Torque engine, which is developed by GarageGames.com (Sullivan, 2001). The Torque engine was used to develop the highly acclaimed multi-user game i.e. Tribes 2. GarageGames.com is currently licensing the Torque engine for USD 100. This engine consists of components such as the scripting engine, mesh engine, mission editor, terrain engine, particle engine, interior/building engine etc (Torque FAQ, 2001). With such features, VR developers can easily create virtual environment with more concentration on the design, creativity and not the underlying codes.

Elsewhere, Morfit, Inc. is another company that offers affordable game engine technology to the mass public. It is currently licensing its latest software technologies such as NorthDragon 3D WebMaker v.1.5, and Morfit 3D Developer Studio v.5.0 for: Microsoft Visual C++, Borland Builder, Borland Delphi, and Microsoft Visual Basic. They cost US$ 49 for non-commercial and US$ 399 for commercial uses.

#### 4.2.2 No Official Guide

GDKs released to the public users come with the computer games bought off-the-shelf. These GDKs do not come with any official user's guide from the developers. For new comers it can be overwhelming.

However, those who are interested in harnessing the power of these GDKs can find information from an immense online support from the gamers community themselves throughout the world, who freely shares their experiences in creating new game levels or virtual worlds based on these engines. Therefore, there are many online resources such as unofficial user's guide, manual, tips and tricks etc. available on the Internet. Tim Sweeney, one of the lead programmers of Unreal stated, "As the online community learns
more about building levels, we will see complete online worlds spanning hundreds or even thousands of levels interwoven by teleporters”.

### 4.2.3 Lack of Awareness and Acceptance

Numerous academic studies have been undertaken on the potential use of VR but there is still reluctance in using the technology, particularly in the construction industry. A VR Awareness Campaign on UK businesses for the UK Department of Trade and Industry has reported that the country’s construction industry was one of the top 5 key sectors that can benefit from VR (Cydata Limited, 2000). Other sectors were Automotive, Aerospace/Defense, Oil/Gas, and Major Engineering Contractors. 82% of construction industry respondents reported increase of awareness after the campaign. The report also concluded that the general reasons for inactivity of VR in UK businesses were "Not needed or not relevant"; "Lack of information"; "Lack of applications"; "Cost"; "Standards/Compatibility problems"; "Technology problems"; and "Other" (e.g. company too small; not ready; other priorities; no perceived benefit). Table 3 summarizes the findings.

<table>
<thead>
<tr>
<th>Reasons given:</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>No need/not relevant</td>
<td>20</td>
</tr>
<tr>
<td>Lack of information</td>
<td>18</td>
</tr>
<tr>
<td>Lack of applications</td>
<td>18</td>
</tr>
<tr>
<td>Cost</td>
<td>17</td>
</tr>
<tr>
<td>Standards/Compatibility problems</td>
<td>8</td>
</tr>
<tr>
<td>Technology problems</td>
<td>2</td>
</tr>
<tr>
<td>Other (e.g. company too small; not ready; other priorities; no perceived benefit)</td>
<td>44</td>
</tr>
</tbody>
</table>

**Table-3: Reasons for inactivity (Source: Cydata Limited (2000))**

Based on the report, the awareness on the technology and the benefits that it encompass should be made aware to the construction industry. As discussed in this paper, the more affordable alternative for development of VR application offered by the gaming industry should also be made known and widely supported and encouraged.

The challenges outlined above are diminishing and should be overcome as newer technologies are invented.
5 Conclusions

As compared to other industries in the world, the computer industry evolves quickly from time to time. The computing processing power that used to be available in expensive high-end workstations is now affordably available in many homes, small and medium sized business. The enormous number-crunching rendering capability of a high-end graphics workstations are integral in computer games which is made possible to run on entry-level Personal Computers. This advancement in computer hardware has made it possible for games developer to creatively produce highly believable interactive 3D virtual environment with human-like motion 3D characters and yet still fun to play.

Game engines and its design concept has the capability to represent a realistic virtual environment in real-time. In the construction industry alone, it can generate real-time VR applications that can represent architectural walkthroughs, 4D planning, virtual pre-construction planning processes and many more. The 3D game engine also offer low-cost VR solution with very outstanding built-in features such as multi-participant capabilities, collision detection, higher frame rates per second and still only requires entry level hardware.

The integration of CAD and 3D Game Engine to develop a low-cost but still maintaining high performance VE application is beneficial for those involved heavily in the construction process e.g. architects and contractors to visualize construction facilities before the actual and physical construction exist. As discussed in this paper utilizing 3D game engine can greatly improve real-time walkthrough experience needed in a VR application, resulting in higher frame-rates with higher quality resolution images.

In turn, the advancement of the computer games industry should be realized to benefit other sectors e.g. construction, manufacturing etc. In construction, the built-in features and tools provided by the game engine like Unreal Tournament can benefit project team in planning ahead before the actual physical construction begins. Using the engine's multi-participant capabilities can assist the project team from the initial design stage to make early and critical decisions to avoid possible disputes, delays, cost overrun etc. at later stages. Visualization of the desired end product in a virtual environment with real-time walkthrough capabilities will allow for such collaboration among project participants and will assist each party to decide on the means and methods of achieving the project goal.
6 References


