The Investigation on Using *Unity3D* Game Engine in Urban Design Study

[1]Aswin Indraprastha & [2]Michihiko Shinozaki

[1]Graduate Student, Architecture and Urban Design Program
Department of Design and Engineering, Shibaura Institute of Technology
Shibaura 3-9-14 Minato-ku Tokyo 108-8548 Japan
Email aswinindra@gmail.com

[2]Professor, Architecture and Urban Design Program,
Department of Design and Engineering, Shibaura Institute of Technology,
Shibaura 3-9-14 Minato-ku Tokyo 108-8548 Japan
Email sinozaki@shibaura-it.ac.jp

Abstract. Developing a virtual 3D environment by using game engine is a strategy to incorporate various multimedia data into one platform. The characteristic of game engine that is preinstalled with interactive and navigation tools allows users to explore and engage with the game objects. However, most CAD and GIS applications are not equipped with 3D tools and navigation systems intended to the user experience. In particular, 3D game engines provide standard 3D navigation tools as well as any programmable view to create engaging navigation thorough the virtual environment. By using a game engine, it is possible to create other interaction such as object manipulation, *non playing character* (NPC) interaction with player and/or environment. We conducted analysis on previous game engines and experiment on urban design project with *Unity3D* game engine for visualization and interactivity. At the end, we present the advantages and limitations using game technology as visual representation tool for architecture and urban design studies.

Keywords: *architecture; game engine; interactive virtual environment; urban design study; visual representation.*

1 Beyond Architectural Visualization

Architectural and urban design is always about idea of visualization and materialization. The importance of visualization is so crucial that two-dimensional and three-dimensional graphics take the most part in design stages. Throughout history, architectural drawings became major media for architects to present ideas and visions to society. It is not surprising that information of ideas and visions are more important for the society (or, clients) than technical or engineering drawing references. The term architectural visualization refers to the media in which architects used to communicate the ideas to the clients.

Received April 21st, 2009.

The trend of architectural visualization comes up to the point that makes depiction of real world as realistic as possible or even hyper-realistic. By the increasing speed of computer processor and amount space in memory, this trend also demands huge computer resources to render and to animate visualization sequences. The basic role of architectural visualization is no longer for the communication purpose only but also for broader scope and functions such as collaborative design process and marketing. At the same time, design process itself transformed into more parallel and multi-tasking activities. Design visualization is needed not only as the culmination of design development but as the input for design strategy as well.

2 Objectives

The objective of this study is to identify the advantages and limitations of using game engine-based application for visual representation tool in urban design study. By understanding its potentials and limitations, we are able to develop new media for visual representation that integrates CAD applications with real-time interactive animation.

3 Related Researches

Game engine-based environment on a particular urban area could be a tool to assist 3D visual simulation for policy maker and every stakeholder in design study and process [1, 2]. Oxman noted [3] that as the basis for visualization tool, game engines have basic advantage to save implementation time by reusing its built-in functionalities such as rendering engines, navigations and interaction tools. Modern game engines offer a high speed, rendering quality, interactivity and multi-user support that are difficult to obtain using any existing visualization tools.

3.1 Game Engine for Visualization Tool

The use of game engine has been developed under various natures of projects. Commonly, the game world is built by importing its entire object from particular CAD/3D applications. In this case, a particular game engine that usually comes with the game itself becomes a platform on which modification or *mod* process is taking place. Modifying game level for architecture visualization purpose is stimulating because it opens up the game engine for real time data rendering and manipulation. Fairuz et.al [4] has studied that in the area of visualization, game engines could quickly develop models and deploy it with texture and lighting effect in game engine-based application with no further demand on high-end computer specification.

In this study, we examined findings of three previous researches that made attempt to use three different game engines as visualization tool. Despite a lot of technical specification on each engine, we focused on the findings of primary function such as 3D geometry data input/output as well as lighting and texture mapping. This consideration was taken mainly because those basic functions are the most significant functions that take advantage from the 3D rendering and animation software.

The summary of this study is depicted by Table 1.

Table 1 Summary of game engines for visualization.

Engine Name	Release Year	Geometries I/O	Lighting	Texture Mapping
Unreal Engine 2	2003	Built-in editor, any CAD data must be converted	Dynamic lighting and shadow, HDR (High Dynamic Range) Rendering	3D vegetation generator, shader management
Source-Half-Life 2	2004	No built-in editor, any CAD data must be converted	Dynamic lighting and shadow, HDR (High Dynamic Range) Rendering	Optimization for large open area
Gamebryo-Oblivion	2006	Built-in editor with libraries, any CAD data must be converted	Dynamic lighting and shadow	3D vegetation generator
CryEngine 2	2007	Natively read most of CAD file format	Dynamic lighting and shadow, Time of day lighting	Heightmap and polygon reduction

Although all of these engines exhibit the capability of producing dynamic visualization in less of production time than 3D rendering and animation software, the main limitation of using these game engines as visualization tool is laid on its flexibility [5, 6]. 3D geometries data must be converted when in use in game engine. Therefore, there is no feedback or interactive mechanism with 3D/CAD software, which architects used to produce design objects.

3.2 Game Engine for Interactivity Tool

In addition to visualization, virtual environment also has the potential to engage users with interactive and communication systems. The virtual environment will only be useful if it supports physically realistic behavior [7]. The types of realistic behaviors that are important depend on the specific application. In architecture and urban design, usually we demand realistic representation of spatial environment with naturalistic effect as well as dynamic elements that inhabit it. Moreover, for interactive tool that could enhance spatial awareness and comprehension to further make better design decision, it is important to

consider how user can interact with game objects then how we can examine on what interaction that has occurred.

We examined findings from two previous projects that use built-in interactivity tool in game engines [8, 9]. Since interactivity is the element that is not provided by any 3D rendering and animation software, it is important to consider the ease of use of this function to enhance the quality of architecture visualization. Table 2 provides the summary of previous findings.

Table 2 Summary of game engines for interaction.

Engine Name	Interactivity function
HalfLife 2	Built-in interaction editor, limited only for non-playable objects. Basic navigation provided.
Renderware	C++ script to build interaction with objects and users

Both findings resulted in the ease of use or learning curve. Scripting mechanism is the main issue to build interactivity using game engines. For the use in architectural or even urban design study, interaction system is based on the assumption that we can observe, examine as well as explore in three dimensions in real-time whatever objects we want to interact with. This kind of demand may require script programming in order to get high flexibility and freedom to achieve those goals.

4 Constructing Virtual Urban Using *Unity3D* Game Engine

In our research, we use *Unity3D* game engine as the fourth generation of game engine that combines visual simulation capabilities with interactive functions and ease of use in the context of geometry data input and output. We pointed out that architectural visualization done by computer has disadvantages as follows:

a. *Functional deficiency*: it tends to separate the process of designing and the process of visualizing. Even further, graphical projections nowadays have no benefit with the design process. The visualizations are the final products and have no functions as design tool as it was.
b. *Resources deficiency*: there is no limit in pursuing realism of computer rendering or animation as long as there are enough resources. This dependency trend is ineffective and inefficient. Focus of design solutions may have been overlooked by visual amenities.
c. *Technical deficiency*: CAD may already be left behind by other technologies in 3D applications. Game engines are uncultivated

technology that offers more productive and effective aid for design process.

By constructing virtual urban using *Unity3D* game engine we explored its capabilities as well as its limitation with the focus on the design process.

The case study is a developed urban area at central district of Tokyo, Yaesu district. This area is about 220 000 square meters and planned to become new development area. Yaesu area consists of mostly offices and commercial middle rise buildings and many of them are registered as old buildings.

The development process of the virtual urban environment follows the sequence shown in Figure 1. The 3D geometry data obtained from GIS application (*ArcGIS*) were prepared by private company [10] in the *.obj format to be interpreted in Blender application.

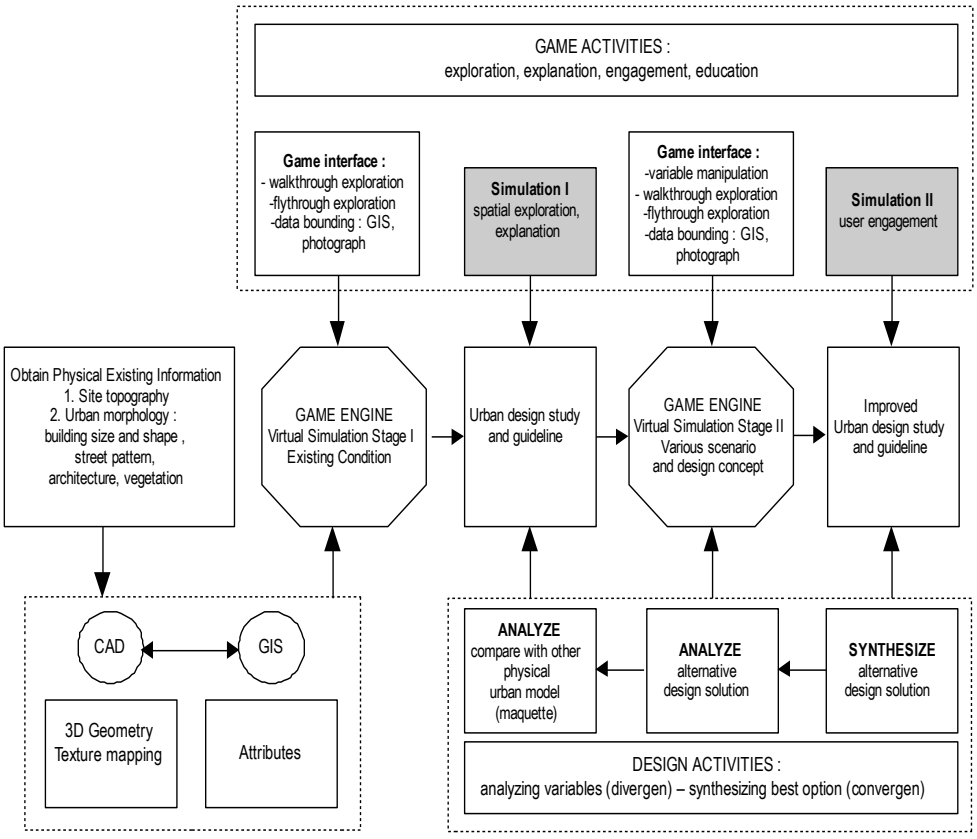


Figure 1 Framework in constructing virtual urban.

At the scale of urban, the complexity of polygon and texture mapping system becomes the main consideration. GIS applications provide 3D polygon exporter but not including any texture mapping system. In the case study, only building has its 3D geometry data in *ArcGIS*. The orthographic photograph of each building facade is obtained from a mapping-service company. This research uses *poly reduction* technique and using *un-wrap* technique for texture mapping in Blender3D software to solve the polygon complexity and mapping system issues. Figure 2 shows the detailed process of overall 3D construction.

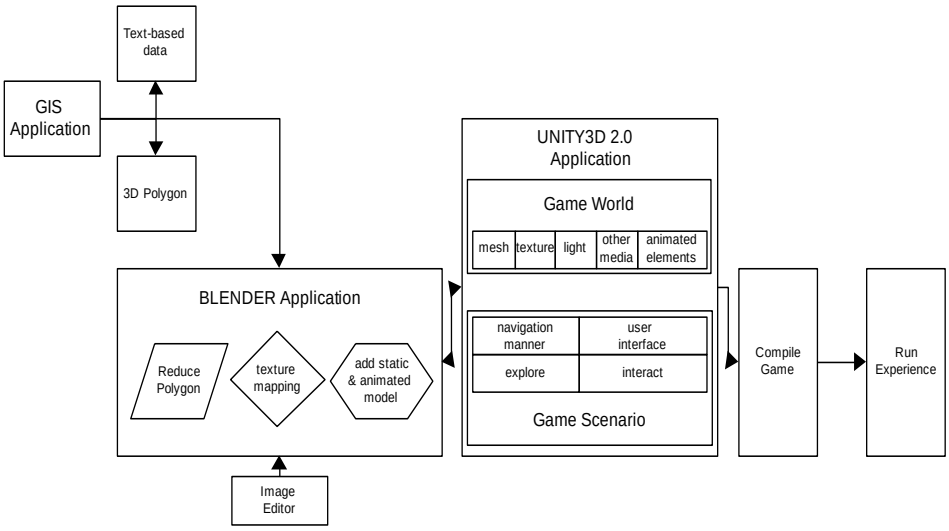


Figure 2 Overall process.

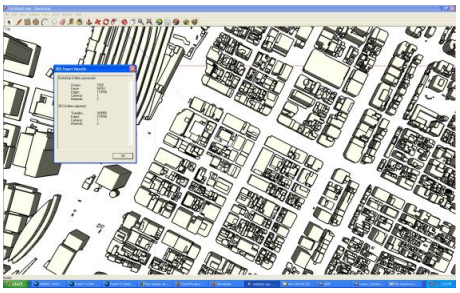
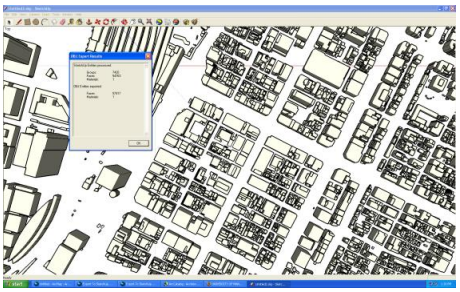
The method for constructing virtual environment for this research can be derived into the following stages:

1. Game world stage: including 3D data acquisition and development, other static game objects as urban furniture such as vegetation, street lights and others as well as animated game objects as urban elements such as vehicles, humanoid characters.
2. Game scenario development stage, including navigation system and interaction system.

4.1 Data Acquisition

Game world construction begins with 3D data acquisition. For such purpose, this research use GIS-based 3D data from *ARC GIS* provided by Tokyo Digital Map Corporation. Automatic 3D data acquisition is employed by using several file format exporter provided by *ArcGIS*. Table 3 shows features of each file format and size of file it produced.

Table 3 File types and data size.

File Format	Properties and file size
	file format : 3DS filesize : 28 308KB proprietary format of Autodesk (Autodesk 3DS Max, AutoCAD) basically it store information about mesh (location and dimension), material, camera and light
	file format : DAE filesize : 46 613KB Digital Assest Exchange is from Collaborative Design Activity for establishing an interchange file format for interactive 3D application. Basically it is XML file and used in various 3D application because it is not proprietary based. store information about mesh, material and shader, physics content
	file format : FBX filesize : 49 286KB proprietary format of Autodesk (Maya) it store information about mesh, image maps, UV mapping.
	file format : OBJ filesize : 11 465 KB proprietary format of Autodesk basically it store information about mesh and material

From the table it can be concluded that the minimum file size is achieved by using .obj file format. Since 3D data mostly consist of simple polygons that represent buildings, the next step is to examine whether these polygons are constructed by using minimum vertices.

Figure 3 shows three characteristics of 3D data opened in Blender: number of object, number of vertices and memory usage. Result shows that although 3D data came from the same objects in *ArcGIS*, the number of objects varies. Number of vertices and memory usage follow accordingly. This issue is due to the file format specification.

	OBJ	3DS	DAE
Num.of object	7421	14941	22361
Num. of vertex (K)	79.8	838	159
Memory usage	83.66	241.65	161.12

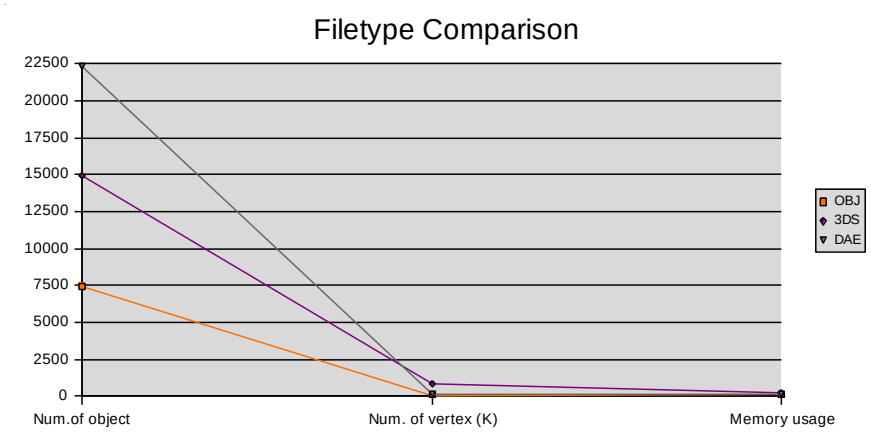


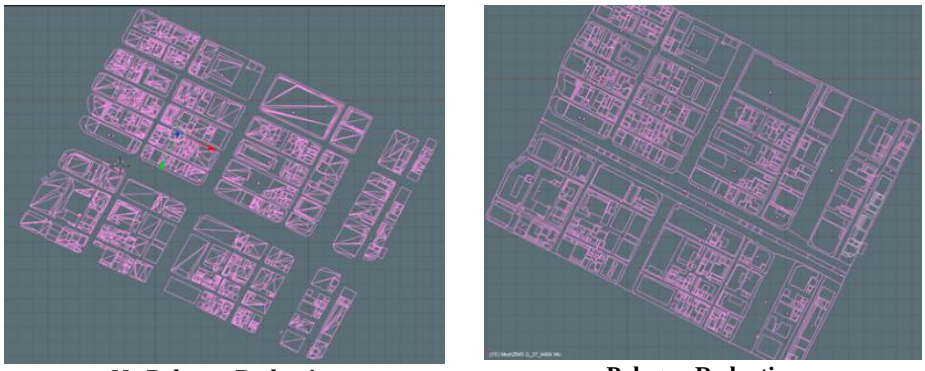
Figure 3 File-types comparison.

The examination of each 3D polygon to reach minimum vertices usage is done by using semi-automatic method in Blender called *Poly-Reduction*[1]. Basically, it will count vertices of each selected object, reduce the number of polygonal faces in particular mesh and scan if there is any vertex duplication. It is semi-automatic because it needs to do fix-up editing afterwards.

After applying *Poly-Reduction* method and performing some fix-up editing, it is clear that the number of vertices decreases. It also affects the file size and

[1] *Semi automatic method in Blender, based on Python script*

memory usage. Figure 4 shows the difference between scene before Poly-Reduction and after Poly-Reduction.



No Polygon Reduction
-Num. of objects : 1018
-Memory usage : 54.14 MB

Polygon Reduction
-Num. of objects : 318
-Memory usage : 51.83 MB

Figure 4 Polygon reduction and data comparison.

4.2 Visualization Techniques

In order to achieve graphic performance as appropriate as the most 3D rendering applications, we explored *Unity3D* capability to handle basic function such as large scale texture mapping and dynamic lighting (see Figure 5). The main factor taken into consideration is the balance between graphic performance and real-time simulation.

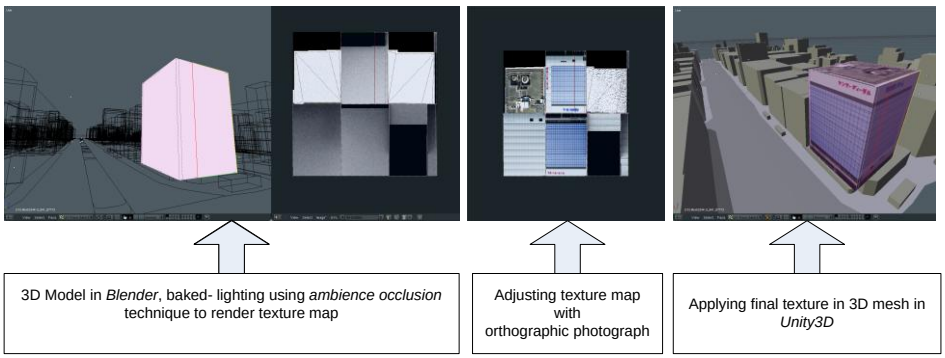
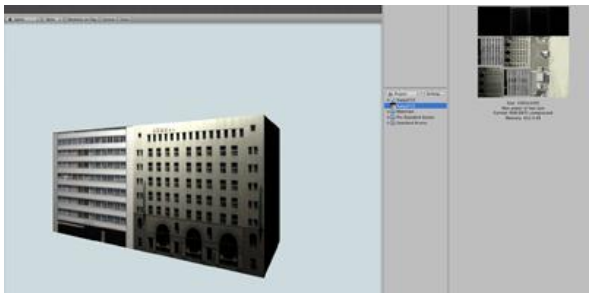


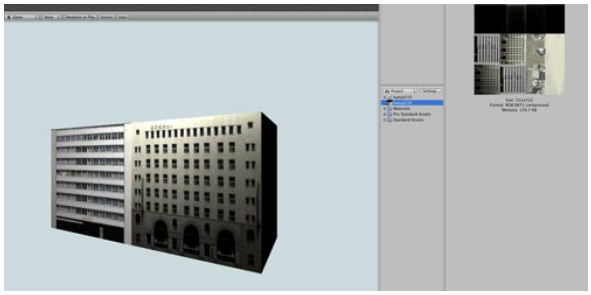
Figure 5 Procedure of texture mapping and baked-lighting.

For such condition, there are two constraints of texture mapping to obtain greater performance both in graphics and game play (see Figures 6 & 7):

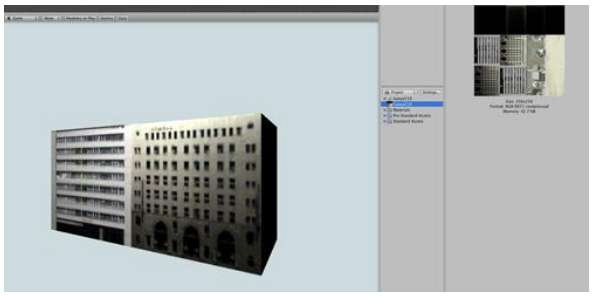
1. Texture size must be as small as possible and visually appropriate.
2. Texture must be layered with baked-lighting information to get lighting effect (*ambient occlusion*) during game play.



-image size : original 1024 x 1024 pixel
-memory consume : 652 KB
-visual quality : high quality, antialiasing, detailed relief



-image size : 512 x 512 pixel
-memory consume : 170.7 KB
-visual quality : appropriate quality, nearly no-difference from 1000 x 1000 pixel



-image size : 256 x 256 pixel
-memory consume : 42.7 KB
-visual quality : poor and loose detail

Figure 6 Comparison of texture size.

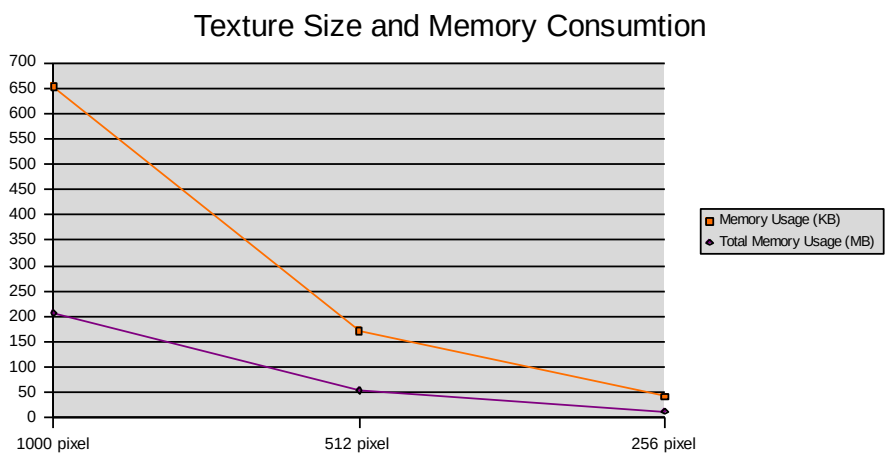


Figure 7 Memory size comparison.

As result, for further development in *Unity3D* we mainly use 2D texture in 512 x 512 pixel resolution.

4.3 Interaction Technique

Virtual reality in game-based environment consists of two parts: game world (game environment and all game objects) and game design (game scenario that determines how the game will be played and how the game world would interact with the user or with other game objects).

Our first intention on investigating *Unity3D* is to explore its ability as a tool for design study, how it differs from any other available 3D applications or any 3D viewer and presentation authoring applications and what is the advancement (see Table 4).

Therefore before we expand our investigation, we start with architecture design point of view. We regard game engine as an engine to develop our tool therefore it is important to consider aspects of design study to be incorporated in this application and at the same time investigating the method and technique to accommodate such requirements.

Table 4 Game design and requirements.

Game Design	*Unity3D*- based application	Other applications
create 3D environment which visual quality balance with the game performance	maximum visual quality for real world representation, i.e : light and shadow, alpha channel, texture map, time-frame independent animation	less visual quality features (VRML, Flash-based)
create system of navigation which allow user to explore with maximum degree of freedom	allow maximum degree of freedom to explore and navigate in virtual environment	pre-programmed animation sequence (3D animation), less flexible freedom of movement (QuickTime VR) view direction, high degree of navigation system (Bentley-3D PDF)
create system of navigation which allow user to examine particular object of interest in various way of perspectives	method to enhance spatial comprehension	view direction, high degree of navigation system (Bentley-3D PDF)
create method to incorporate various type of information	rich content and other data bounding technique	require script language and server-based interaction (Flash-based)
create method to accommodate flexible 3D data exchange	method for 3D data interchangeable to external tool	less flexible to perform data synchronization (Flash-based and other animation packages)
create a method to translate conventional analysis study	method to perform some basic visual-based analysis	less flexible
flexibility to expand game design	Object-oriented programming	

4.3.1 Navigation

The navigation system is the main important to be considered in this application. As for a tool to examine 3D world, the main considerations of developing navigation system are:

1. It must be easy to learn, easy to remember and easy to use with maximum intuitive level as possible.
2. It must have rich features on minimum key input

As navigation system is crucial both for exploration and comprehension of 3D environment, we then elaborate the design of navigation system into the following criteria:

1. There are two main navigation systems: *Human Eye View (HEV) (First Person Camera) and Bird Eye View (BEV)*. Both are triggered by mouse stroke on an icon.

2. On each navigation method, instead of using icon-based navigation panel for pan, zoom, orbit and others (see Figure 8), each method has contextual button using mouse-keyboard stroke combination.
   In BEV:
   - zoom: using mouse scroll wheel
   - orbit: using mouse movement
   - pan: using mouse left click – and drag
   -

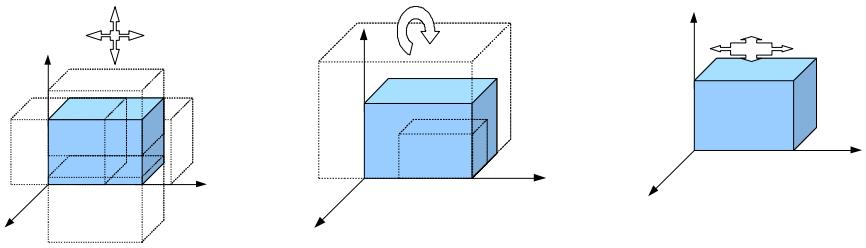


Figure 8 Pan, zoom and orbital navigation using mouse.

3. In FPS, we use standard view navigation for FPS game (see Figure 9) :
   - *look* and turn : using mouse movement
   - *walk* forward : using key "w"
   - *walk* backward : using key "s"
   - *slide* left : using key "a"
   - *slide* right : using key "d"

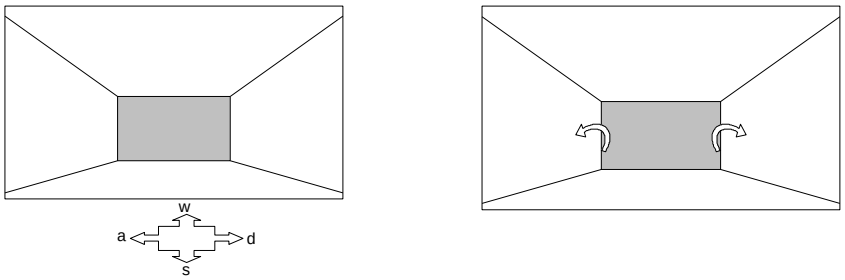


Figure 9 Turn/look and move toward direction using mouse and keyboard combination.

All these navigation system are programmed using *Unity3D* JavaScript since it does not provide any built-in navigation system as available on previous web-based applications.

Snapshot	Function
	HIDE/SHOW Hide selected object and show selected object
	ISOLATE/ SHOW ALL Isolate : hide unselected objects in the same building's lot
	RETURN Back everything to normal

Figure 10 Snapshot of interaction functions.

4.3.2 Interaction

Main considerations behind an interaction system are:
1. object manipulation: select, hide, show, isolate, swap
2. information retrieval : text, image, video

Object manipulation method is a feature that allows user to select any particular objects and perform such task that has effect on those objects. It mainly focuses

on the interaction in such a way that the user has the focus on this particular object. The manipulation method provides *hide/show* selected object or hide/show other objects, and isolates particular object from others (see Figure 10).

In the context of design study and urban design study in particular, virtual environment with this kind of interaction gives user flexibility to examine and to comprehend objects and its context. Combined with a navigation system, this application can be used as a tool for conducting urban-scale visual representation with advancement in flexibility to manipulate objects at real time.

In addition, we develop another feature named: ***Swap Object***. This feature allow user to change selected mass or object with new mass from the preset gallery (see Figure 11). The main reason to put such a feature in is to allow user examines the effect of design transformation on 3D objects and its surrounding environment. This feature is not available in any previous applications.

Figure 11 Snapshot of swap objects.

Another function to push the ability of a game engine as tool for interactive visualization is interactive animation. In this case we are about to show an animation of building process within the game environment (see Table 5 and Figure 12). We promote a new way to represent architectural or urban design by combining animation and interactivity in game environment.

In game engine-based environment, animation can be bound by using its object and key frame information. *Unity3D* translates this information and wrap-up

animation at runtime. As a result, the animation can be viewed from any viewpoint.

Table 5 Type of animation techniques.

Type of animation representation	Property
Conventional Animation	static, user become spectator, no interactivity.
Interactive visualization using authoring software	static, user engage with application using point-click method and others. 2D animation
Interactive visualization using game engine	dynamic, user engage with application in 3-dimensional world

Building's lot selected, user can hide the buildings

Animation sequence on selected object

User can examine animation from any view point

Snapshot after animation completed

Figure 12 Snapshots of interactive animation.

5 Conclusion

We found some findings on evaluation of *Unity3D* game engine as a tool for design study by examining the aspects of visualization and interaction. By considering these two factors, we emphasize on the data compatibility between *Unity3D* game engine and CAD/GIS software (*Blender, AutoCAD, ArcGIS*). This compatibility and interoperability of game engine has significant factor for the integration of digital design tools.

From the experiment, we found significant improvement of CAD data compatibility in *Unity3D* as it can natively read most CAD file types. Furthermore, seamless integration between game engine and CAD applications makes file synchronization possible. We experienced that .obj file format is the most effective format to be exported to Unity for meshes as .fbx to animation.

For the interaction mechanism we created scripts and develop basic interaction system that engages users within this virtual environment. From one side of view, *Unity3D* JavaScript-based language gives freedom to create interaction system based on our own preference and ideas. Some of 3D navigation systems that we created are learned from other applications such as *Google Earth, Adobe 3D PDF, Corona VRML* player and others.

Object oriented programming language also gives an advantage in the way that any interaction method came from game object behavior and it responds to user interaction. The script-based interactions also open possibility to create an interaction system that bounds external data into game object. For example, we bound external data of text, image, and video into game interaction.

Although *Unity3D* has the potential to facilitate the development of an application that can be a 3D viewer and 3D simulation with interaction features at the same time, it has some limitations regarding to its function as a general purpose game engine application, which leads to some restraint conditions particularly in non-computer science fields.

In architecture and urban design study, the limitation comes mostly by the fact that *Unity3D* has not been equipped with computer aided design tools that are usually embedded in such applications. As for visual representation tool with object-based interaction system that mostly done by using *Boolean* algorithm, *Unity3D* comes with appropriate features and capability in delivering such simulation.

References

[1] Hunt, E. & Waller, D., *Orientation and Wayfinding*: A Review, in ONR Technical Report N00014-96-0380, Office of Naval Research, Arlington, VA, USA, 1999.

[2] Schnabel, Marc Aurel, *Architectural Design in Virtual Environment*. Published PhD Dissertation, University of Hongkong, 2004.

[3] Oxman, Rivka, *Digital Architecture as a Challenge for Design Pedagogy: Theory, Knowledge, Models and Medium*, Journal of Design Studies 29 (2008), p.99-120, 2008.

[4] Fairuz, Shiratudin & Walid Thabet, *Virtual Office Walkthrough Using a 3D Game Engine*, Journal of Design Computing, 1V, 2002.

[5] Moloney, Jules & Lawrence Harvey, *Visualization and 'Auralization' of Architectural Design in a Game Engine Based Collaborative Virtual Environment*, Proceeding of the 8th International Conference of Information Visualization, 2004.

[6] Eshaq, Ahmad Rafi & Peter Karboulonis, *The Importance of Virtual Environment in The Design of Electronic Games and Their Relevance to Architecture*, Proceeding of the 18th International Conference of eCAADe, 2000.

[7] Bermudez, Julio & Kevin King, *Media Interaction and Design Process: Establishing A Knowledge Base*, Journal of Automation in Construction 9 (2000), p.37-56, 2000.

[8] Calderon, Carlos & Marc Cavazza, *Using Games Engine to Implement Intelligent Virtual Environment*s, Proceedings of the 2nd International Conference on Intelligent Games and Simulation (GAME-ON 2001), 2001.

[9] Andreoli, Roberto, et al., *Interactive 3D Environments by Using Video Games Engines*, Proceedings of the 9th International Conference in Information Visualization. IEEE, 2005.

[10] Tokyo Digital Map Corporation © 2007.