Video Game Design Method for Novice

Wei-Da Hao[1] and Akash Khurana[2]

Department of Electrical Engineering and Computer Science,
Texas A&M University-Kingsville
700 University Blvd, Kingsville, TX 78363, USA
[1]wei-da.hao@tamuk.edu
[2]akash.khurana@students.tamuk.edu

Abstract—**This paper shows how college students without prior experience in video game design can create an interesting video game. Video game creation is a task that requires weeks if not months of dedication and perseverance to complete. However, with Alice, a group of three sophomore students who never designed a game can create a full-fledged video game from given specifications. Alice is 3D graphics interactive animation software, which is well-tried and proven to be an enjoyable learning environment. At the start of this project, students are given guidelines that describe expected outcomes. With minimum supervision, in three days, a working program that matches the guidelines is accomplished. In additional two days, students enhance the quality with better graphics design and music. With this experience, 3D graphics interactive animation software, like Alice, is demonstrated to be a useful teaching tool in education for academic courses of game development and design. This paper not just discusses how the video game was created, but also speaks of the difficulties the team overcomes easily with Alice.**

Keywords—**Video game, 3D Interactive Animation, Alice.**

1. INTRODUCTION

The Computer Science Department at the Texas A&M University-Kingsville, USA has been dedicated to promote higher education for Hispanic community in USA. This is an effort shared by many other minority universities nationally, and a number of different teaching and learning methods have been proposed and tried, including development of summer camp and May-mester program. Steve Cooper, Wanda Dann and Barbara Ericson have published a book "Exploring Wonderland, Java Programming Using Alice and Media Computation," [1] which is an output from NSF grants 0618461, 0618380, 0618531 and 0618562. Alice is a programming environment that students can import 3D objects freely downloadable from public domain, and create their own 3D virtual world. Many literatures have reported remarkable and encouraging results in teaching introductory programming CS course [2-4]. However, Alice, as well as other 3D interactive animation programming software, is a useful teaching tool in other courses, too.

This paper intends to emphasize the role of Alice in video game design, especially for students new to this field by reporting an experience using Alice to develop and teach video game design short course in May-mester program supported under NSF grant DUE- 0525496.

To program in Alice for video game, in addition to the well-known drag-and-drop editor to avoid syntax error, students use many of the 3D objects and manipulating methods resulting from decades of progress in computer graphics and virtual reality in a user-friendly environment; for example, modelling 3D objects are avoided because a wealth of 3D objects needed in the design are accessible from web gallery linked to Alice. Moreover, Alice is a gift from Carnegie Mellon University, USA, which is freely downloadable from website www.alice.org. Next version of Alice will be developed by collaboration between academic communities and Sun-Micro system, Inc., who originates Java programming. The graphical, animation and interactive features of Alice enable students to streamline the game design and test process without even first looking at user manual. S. Cooper, W. Dann, and R. Pausch observe that students new to video game design or programming in general find this environment particularly appealing [5]. The same kind of attraction exists to students who are less prepared in science, technology, engineering and mathematics education.

Figure 4 in Appendix I shows the integrated development environment of Alice. The World View panel located at the center of the window is where virtual world is built. By clicking on the Add Object button located immediately below the World View, one can access the gallery for importing new 3D object. Students can click on an object in the object tree in the upper left panel, and then select a method of the selected object displayed in the lower left Detail panel. Programming in Alice is to drag-and drop the selected method to Editor Panel located at lower right of the window.

In section II, we describe the specification of the video game for student to work on. In section III, we show the students' learning outcome, which dictates how the video game was created by a team of three sophomore students without prior game design experience, and the difficulties the team overcomes easily with Alice. In section IV, we address the expectation that students using Alice will decide to take another course in game design and make transition from Alice to other video game development software.

2. SPECIFICATION OF STUDENT PROJECT

A. Project Description

The project assigned to the team is about a frog crossing the pool. This is a project from the book

DOI: 10.5176_2010-2283_1.1.04

"Programming with Alice," authored by Wanda P. Dann, Stephen Cooper and Randy Pausch [6]. The team is composed of three sophomores from a college in south Texas. They have taken the course "Object-oriented programming" in school, but no experience in video game design.

A graduate student works as assistant in May-mester program is the interface between instructor and the team. After instructor briefs on the project, the team is on their own to get the project done. When there is unresolved issue, assistant will inform instructor for directions.

The goal of this project is to design a virtual world, which contains frog, pool and lily pads as 3D objects, and make the frog jump from lily pad to lily pad until it gets all the way across the pool. If the frog jumps and lands on the water (not on the lily pad) the frog sinks into the pool and the game is over. If the frog jump all way across the pool and lands on the other side of the pool, the player wins. Figure 1 shows the example snapshot of the game.

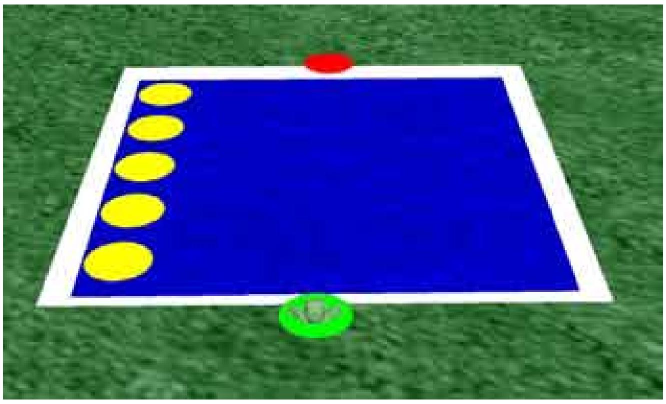


Fig. 1. Example snapshot of the game to be developed by students

B. Project Specification and Instructions

The team is expected to consider the following set of specifications to satisfy the game design:

1) Use a rectangle in blue color to serve as swimming pool.
2) Use five yellow circles deployed along the long side of the pool to serve as lily pads (each one meter apart) floating back and forth across the pool.
3) Place the frog on one green circle aligned with one short side of the swimming pool.
4) Place a red circle on the other short side of the swimming pool.

To help the team to get started smoothly, a few computer science concepts and programming techniques are briefly explained to the team. The topics include:

1) Use data structure "list" to store the five yellow circles.
2) Pass a random number to each lily pad as the duration it floats across the swimming pool to create different speeds.
3) Use an infinite loop to repeatedly move the five lily pads across the pool back and forth at different speeds.
4) Create an event handling event to have the frog jump one meter forward, when player presses the ↑ key.
5) Create an event handling event to have the frog move left, when player presses the ← key.
6) Create an event handling event to have the frog move right, when player presses the → key.

The above instructions are delivered to students in handouts. Each student has a tutorial on using Alice and has access to Internet to find out more about Alice and related information. After instructor conducts thirty minutes lecture on the handouts, students get started.

3. STUDENT LEARNING OUTCOME

A. Use of Alice

The feedback, tabulated in Table 1, from our students shows their experience in using Alice.

TABLE I
FEEDBACK FROM STUDENTS IN PROGRAMMING WITH ALICE

Advantages	Drawbacks
• Eliminate the problem of syntax error. • Tutorials available for downloading • Users can visualize their creation in 3D. • Users can execute to test their programs before completed.	• No tool available to convert 3D objects in Alice to 2D flash. • Creating if statements, using math functions and using relational operators are not that easy as other coding in Alice. • 3D animation is not that good as other commercial 3D animation software.

Although some drawbacks are reported, they agree that Alice has been a great tool in overcoming obstacles to programming [7]. Since Alice is adapted from more complicated virtual reality software for the purpose of teaching and learning programming, its animation is believed to have been simplified.

B. Go beyond the Specification

When students start their project, they have decided to first create a game following the given specification, and then enhance it to a better one. According to their project report [7], they recognize that Alice is program that allows users to program whatever one desires in 3D, so, they have an idea in mind from the start to use this exciting tool making something they will be proud of.

To achieve the goal of the project and beyond, students have set the following objectives:

1) Create a game following the given specification.
2) Enhance the design by:
   - Adding waterfall to the virtual world.
   - Letting the frog swim across the pool into the waterfall, when not landing on lily pad.
   - Beautifying the environment around the swimming pool.
   - Adding music to the game.

C. Learning Outcome

To truly reflect what the students have achieved and learned with the use of Alice in designing their first video game, the major content in their report is duplicated below with minor revision:

The process of creating a new and better game is going to be a process containing lots of detail. Even the smallest detail requires much attention. The first step involves importing 3D objects to virtual world to set up a scene better in graphics than the original one.

After creation of the scene, the next step would be making the already in place lily pads move back and forth across the pool at different speeds. This step is very detail oriented, since the shape of the pool of the waterfall is not a rectangle. Dozen lines of codes are implemented for the task in this step.

When modifying the existing code or adding a new line, extensive testing must be carried out to ensure that the software executed as expected. This is because the change of codes in one place could affect several other modules in the coding.

The actions of jumping, sinking and dancing of the frog are coded into methods. These methods are invoked in an event handling routine connected to an event when player presses ↑ key.

For example, when player presses ↑ key, method jumping is invoked to let the frog jump forward. Then, if the frog does not land on the lily pad, method sinking is invoked. When the frog lands on the circle deployed along the side of the pool of the waterfall, method dancing is invoked.

All of the simple and complex actions have to be coded and invoked in certain instant. If an error exists in the coding, the program shall not work properly. Hundreds of lines of codes are created in this project, which is a time-consuming task. Thanks to Alice, the team is able to accomplish most of the objectives, and learn to program in an advanced level.

Figure 2 shows the snapshot of the enhanced game accomplished by the team. The picture in Figure 2 depicts the final version of the enhanced game design. The image portrays the scene of the imagination in students' mind when they decide to make a difference. Compared to the original design, this picture has better graphics and realistic touch.

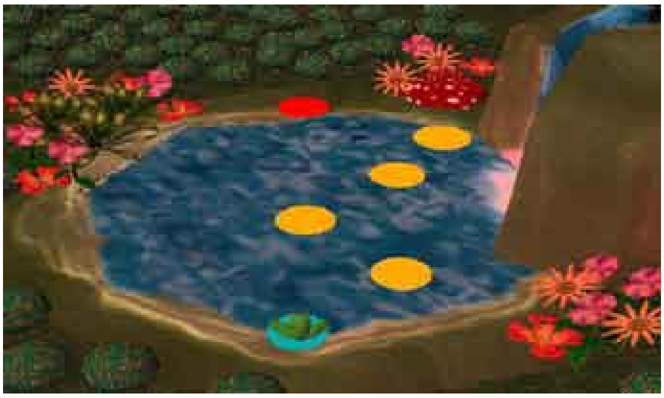


Fig. 2. Snapshot of the enhanced game by the team

D. Overcome Major Difficulty

During the five days of project development, most of the time students try to figure out how to move each lily pad at different speeds in the non-rectangle shaped pool. The drag-and-drop coding style in Alice allows them to try different methods in a fast pace. Otherwise, mistakes are very likely to occur when they revise the code.

Although students did not accomplish all the goals set at the beginning for the enhanced design (for example, 2nd item in object 2) in students' objectives is not achieved.), and there are bugs existing in the coding, it has been observed that Alice fosters an environment for students to carry out their imagination and to outperform themselves. In the aspect of programming, with drag and drop design, Alice keeps the frustration of syntax errors to the minimum, so students can focus more on the core logic of their design. The wealth of 3D objects in a variety of categories freely downloadable from Alice gallery is another major factor that enables its tremendous impact on programming for students, including the programming on video game.

4. WHAT'S NEXT?

Currently, Alice is not professional game design development software, so we expect students using Alice can decide to take another course in game design to make transition to other software. Once students are motivated at the beginning stage of their video game design, the success rate to manage more complex design tools is much higher. In addition to commercial game design software from Microsoft (www.microsoft.com) and Autodesk (www.autodesk.com), readers can find the list of many open source game design software in appendix II.

This paper is about the first video game design lesson for beginners. Before one wants to design a professional video game, however, he needs to think about its design from different aspects, including modelling, rendering, animation, and its impact to the society. Although it is a team work to develop a professional game, academic curriculum is supposed to include all of the necessary subjects to foster leaders of the century. Figure 3 shows before conducting a professional game design or taking computer graphics course, students had better acquire the knowledge of digital art, modelling, rendering and animation, as well as the understanding of the impact of video game to the society [8]. Before the long trek, Alice could be the valuable vehicle to motivate students at the very early stage.

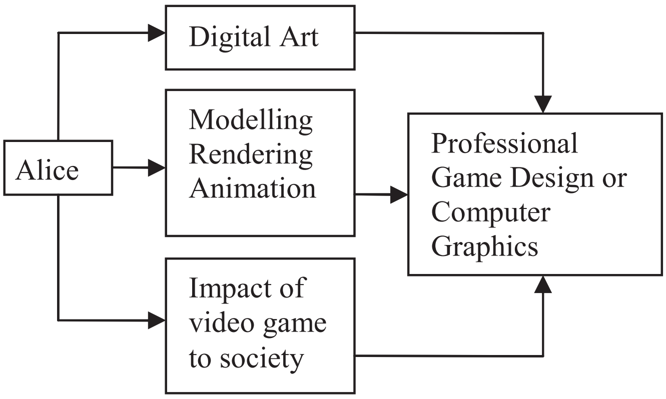


Fig. 3. Illustration of "what's next" after Alice

ACKNOWLEDGEMENT

Our thanks to the support of May-mester at Texas A&M University-Kingsville through a cooperative agreement (DUE#0525496) with the National Science Foundation (NSF)

REFERENCE

[1] Steve Cooper, Wanda Dann and Barbara Ericson, *Exploring Wonderland: Java Programming Using Alice and Media Computation*. Prentice Hall 2009.

[2] Stephen Cooper, Wanda Dann, Randy Pausch, *Teaching Objects-first In Introductory Computer Science*, SIGCSE 2003. http://www.alice.org/index.php?page=publications/publications

[3] Matthew Conway etc., *Alice: Lessons Learned from Building a 3D System for Novice*, CHI 2000. http://www.alice.org/index.php?page=publications/publications

[4] B. Moskal, D. Lurie, S. Cooper, *Evaluating the Effectiveness of a New Instructional Approach*, Proceedings of the 35th SIGCSE Technical Symposium, pp. 75-79, March 2004, Norfolk, VA .

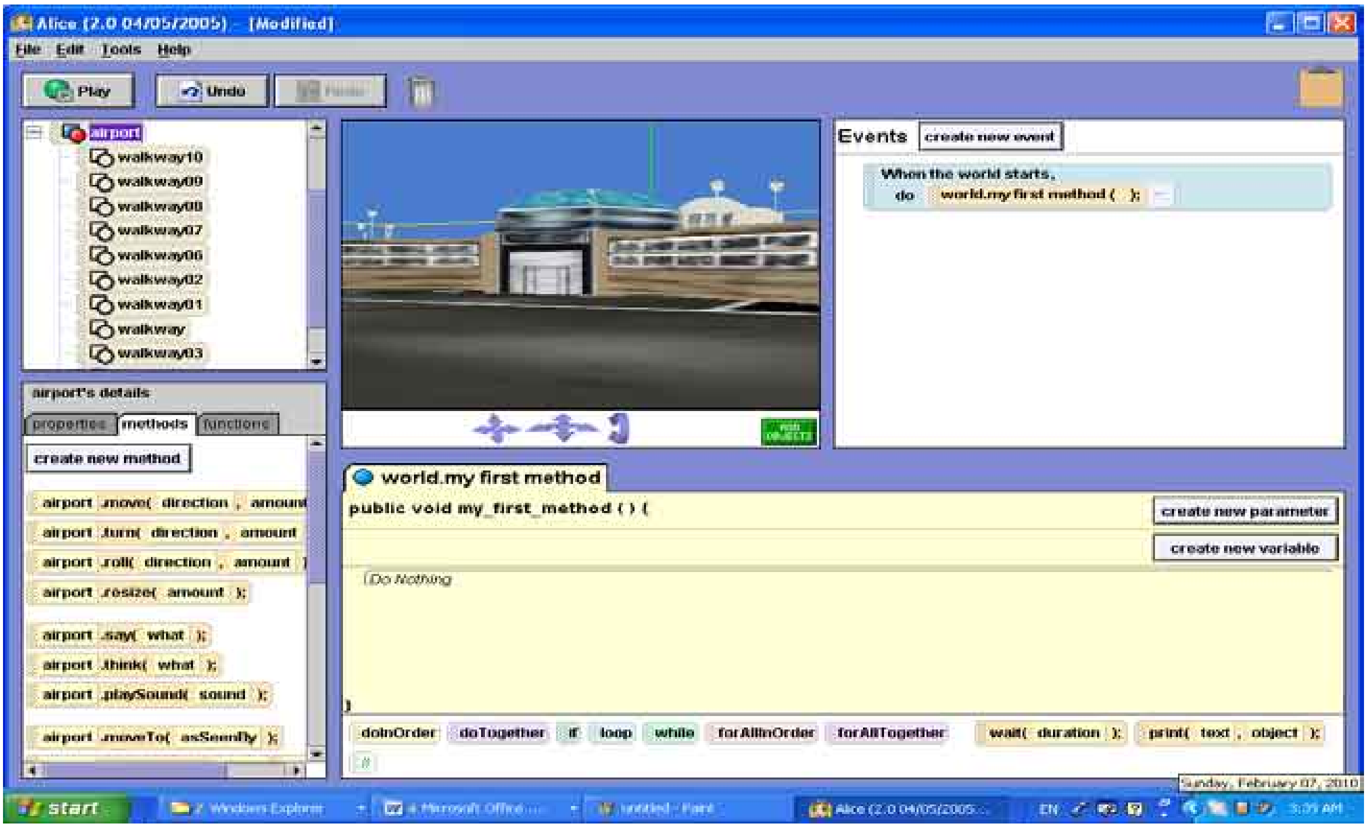
[5] S. Cooper, W. Dann, and R. Pausch, *Using Animated 3d Graphics to Prepare Novices for CS1*, Computer Science Education, 13(1), pp. 3-30.

[6] Wanda P. Dann, Stephen Cooper and Randy Pausch, *Learning to Program with Alice*. Prentice Hall 2006. p. 248

[7] Garza Jose, Velasquez Jorge, Villarreal Herald, Akash Khurana and Wei-Da Hao, *Develop Video Game Using Alice 2.2*. Presentation. STEM* Talent Expansion Program, May 18-29 2009.

[8] Richard Sandford and Ben Williamson, *Games and Learning*, Futurelab October 2005. http://www.futurelab.org.uk/resources/publications-reports-articles/handbooks/Handbook133

APPENDIX I



airport .move(direction , amount
airport .turn(direction , amount
airport .roll(direction , amount)
airport .resize(amount);
airport .say(what);
airport .think(what);
airport .playSound(sound);
airport .moveTo(asSeenBy);

When the world starts,
do world.my first method ( );

```
public void my_first_method ( ) {
  (Do Nothing)
}
```

Fig. 4. The Alice User Interface

APPENDIX II

TABLE II
LIST OF OPEN SOURCE GAME ENGINES

(Remark: Readers are referred to http://en.wikipedia.org/wiki/Game_engine for a more complete list of open source game engines.)

1	Aleph One (formerly known as the Marathon Open Source Project)	**Wiki**: http://en.wikipedia.org/wiki/Aleph_One_(game_engine) **Website**: http://source.bungie.org, http://sourceforge.net/projects/marathon/ **Vendor**: Marathon
		Note: 1. download Aleph One 2.download all new maps, physics files, scenarios.3.Create content using tools like Forge(Map Editing), Anvil(Physics modelling, Shape file editing).
2	Allegro library	**Wiki**: N/A **Website**: http://www.liballeg.org **Vendor**: Marathon
		Note: Uses C programming.
3	Axiom Engine	**Wiki**: http://en.wikipedia.org/wiki/Axiom_Engine **Website**: http://www.devmaster.net/engines/engine_details.php?id=81 http://axiomengine.sourceforge.net/forum **Vendor**: Open source community
		Note: Fully object oriented game engine development effort using C# and the Microsoft.NET platform to create an easy to use, flexible, extendable, and powerful engine that allows for rapid development of games
4	Build engine	**Wiki**: http://en.wikipedia.org/wiki/Build_engine **Website**: http://advsys.net/ken/build.htm **Vendor**: Open source community
		Note: The Build engine represents its world on a two-dimensional grid using closed 2D shapes called sectors, and uses simple flat objects called sprites to populate the world geometry with objects. It is generally considered to be a 2.5D engine as the basic world geometry is two-dimensional.
5	Crystal Space	**Wiki**: http://en.wikipedia.org/wiki/Crystal_Space **Website**: http://www.crystalspace3d.org/main/Main_Page/ **Vendor**: Open source community
		Note: Crystal Space is a framework for developing 3D applications written in C++. It is very modularly built with a number of more or less independent plug-ins. The client programs use the plug-ins, like the OpenGL 3D renderer, by registering them via Crystal Space's Shared Class Facility (SCF).
6	Cube	**Wiki**: http://en.wikipedia.org/wiki/Cube_(video_game) **Website**: http://cubeengine.com/cube.php4/ **Vendor**: Free and open source software
		Note: Cube is a landscape-style engine that pretends to be an indoor FPS engine, which combines very high precision dynamic occlusion culling with a form of geometric mipmapping on the whole world for dynamic LOD for configurable fps & graphic detail on most machines. Uses OpenGL & SDL.
7	Cube 2: Sauerbraten	**Wiki**: http://en.wikipedia.org/wiki/Cube_2:_Sauerbraten **Website**: http://sauerbraten.org **Vendor**: Free and open source software
		Note: The aim of this game is not necessarily to produce the most features & eye candy possible, but rather to allow map/geometry editing to be done dynamically in-game, to create fun game play and an elegant engine.
8	Delta3D	**Wiki**: http://en.wikipedia.org/wiki/Delta3D **Website**: http://delta3d.org// **Vendor**: Open Source
		Note: Delta3D's modular design integrates other well-known Open Source

		projects such as OpenSceneGraph, Open Dynamics Engine, Cal3D, and OpenAL, integrating them in an easy-to-use API. Rather than bury the underlying modules, Delta3D integrates them together, allowing access to the important underlying components. This provides a high-level API while still allowing the end user the optional, low-level functionality.
9	Digital Novel Markup Language	**Wiki**: http://en.wikipedia.org/wiki/Digital_Novel_Markup_Language **Website**: http://hp.vector.co.jp/authors/VA015183/ **Vendor**: Open Source
		Note: DNML is one of the first scripting language game engines for creating visual novels, an interactive fiction game. It was developed using C++. The programming structure is similar to HTML which it made it easy to produce dōjin games. Up until 2001, dōjin games were not yet popular on the Internet except in Japan which caused the developers to cease the development of DNML.
10	dim3 (also called as Dimension 3)	**Wiki**: http://en.wikipedia.org/wiki/Dim3 **Website**: http://www.klinksoftware.com/ **Vendor**: Open source community
		Note: dim3 uses OpenGL for rendering, OpenAL for audio, JavaScript for scripting (using Safari's Nitro implementation), XML for data and Simple DirectMedia Layer for resolution switching and input. Some of the engines features include shader support, per-pixel lighting, bump, and specular mapping, real-time lighting and/or baked lighting, particles and other special effects, skeletal animation and customization through javascript.
11	Exult	**Wiki**: http://en.wikipedia.org/wiki/Exult **Website**: http://exult.sourceforge.net/ **Vendor**: Open source community
		Note: N/A
12	Fore Thought Entertainment QuakeWorld	**Wiki**: http://en.wikipedia.org/wiki/Fore_Thought_Entertainment_QuakeWorld **Website**: http://www.fteqw.com/ **Vendor**: Open source community
		Note: N/A
13	Game Blender	**Wiki**: http://en.wikipedia.org/wiki/Game_Blender **Website**: http://www.blender.org **Vendor**: Open source community
		Note: The Blender Game Engine is a component of Blender, a free and open-source comprehensive 3D production suite, used for making real-time interactive content. The game engine includes support for features such as Python scripting and OpenAL 3D sound.
14	Visualization Library	**Wiki**: http://en.wikipedia.org/wiki/Visualization_Library **Website**: http://www.visualizationlibrary.com **Vendor**: Open source community
		Note: The most recent release is Visualization Library 2010.06.1000, released on the 1st June 2010 and featuring a Simplified BSD Licensing model.

Wei-Da Hao was born in Taiwan in 1964. He received the B.S. degree from National Taiwan University, Taiwan, in 1985; M.Sc. degree from National Chung-Hsing University, Taiwan, in 1990; and Ph.D. degree from Portland State University, USA, in 1996.

He is currently the assistant professor of the Department of Electrical and Computer Science, Texas A&M University-Kingsville, USA. His research interests are in the areas of computer graphics, educational video game design, theoretical computer science and digital signal processing.

Akash Khurana was born in India. He was admitted to the Masters Program of Computer Science, Texas A&M University-Kingsville, USA, in 2009. Currently, he is working on his thesis to fulfil the degree requirement. His research interest includes technology enhanced education, especially in applying 3D interactive animation software as a training tool to engage teaching and learning. His career goal is in the field of software testing automation.