An interactive and pen-based simulator to enhance education and research in computer systems: An experience report

Vincent Tam, Johnny Yeung,
C H Leung, Edmund Y Lam and Diane Salter
University of Hong Kong
Hong Kong SAR, China

ABSTRACT

The active uses of simulators to facilitate and/or promote learners' experience in many applications has significantly reshaped the latest educational technology or training methodologies in the past decades including the training of engineering students to understand the actual working mechanisms of specific engineering principles, or the military officers on tactic planning in a simulated combat environment. In many cases, it was clearly revealed that the appropriate uses of simulators not only avoids the indispensable costs of human lives or money lost in the hostile combat or investment field, but also effectively motivates and/or enhances the learners' interests in the relevant fields of study, thus fueling significant impacts on their actual performance. However, many conventional simulators often require the users to input a formal specification file such as a *script* or program to specify about the simulation settings. Besides, even in many Window based simulators, the users may need to explicitly memorize about the meanings of various system variables and their proper settings before running a simulation to observe the imparted changes. All these unnecessary hassles will drastically reduce the interactivity of simulators, and also lower the users' interests in using them. With the fast developing tablet and ultra-mobile PCs, we have seen ample opportunities of employing sophisticated pen-based computing technologies to improve the interactivity of simulators in order to enhance the learners' experience to learn, reason or visualize with simulators in more effective ways. Therefore, in a recent pen-based simulator development project awarded by the Microsoft Research Asia (MSRA), we proposed to use the Microsoft digital ink library to support fast symbol/character recognition and the XML technologies to flexibly define various models of computer architectures so as to build an innovative and pen-based simulator for mobile computing devices. With pen-based or other inputs, our simulator allows the instructors/students to flexibly add or modify instructions that will generate *live* animations to facilitate interactive discussion for teaching undergraduate to postgraduate courses. Besides, our simulator has the full potential to support research on computer systems through visualization of new results generated out of new computational models or optimization strategies. A prototype of our simulator was completed and then released to all our Year-1 students for trials in the last month in which we collected some initial and positive feedbacks. A more vigorous evaluation was planned and would be conducted by the end of this spring semester. All in all, there are many interesting directions for further investigation including the integration of relevant course materials in the form of digital resources or pointers to online databases into our simulator, and a careful study of the pedagogical changes brought by our innovative and pen-based simulator.

1. Introduction

Basically, most educators agree that the use of interactive simulation tools (FSim, 2009; SimBusiness, 2009; SimCity, 2009; Sims, 2009) can help students better understand the underlying working principles of many biological or physical systems, and therefore motivate the learners' interests for further investigation that may lead to improved performance in a specific subject. For instance, the use of a computer-aided simulation game named "The Incredible Machine" (Ward, 1998) was found to enhance the mechanical reasoning capabilities of transition year students, including 17 females and 27 males with their age between 15 and 16 years, in an exploratory study conducted in a vocational school in Ireland. The study strictly followed the pre-test/post-test control group design, with the students' mechanical reasoning skills evaluated by the Differential Aptitude Test for Guidance. There were significant differences in favor of male students in the experimental group with training using "The Incredible Machine" simulation game as shown in the t-test analysis of their pre-test and post-test raw scores. Besides, there are many other successful applications of simulation tools or games in other sectors of education (Kent, 2007; SCS, 2007; SuperKids, 2007).

In Hong Kong, where commercial and financial activities are overwhelmingly active in recent years with a lack of long-term development infrastructure for Engineering related disciplines, many tertiary educators in Engineering are faced with the core problems of motivating the students' learning interests. As revealed in many aforementioned cases, the appropriate uses of simulators (Wikipedia, 2009) can effectively motivate and/or enhance the learners' interests in the relevant fields of study, thus bringing significant impacts on their actual performance. However, many conventional simulators often require the users to input a formal specification file such as a *script* or program to specify about the simulation settings. Besides, even in many Window based simulators, the users may need to explicitly memorize about the meanings of various system variables and how to properly set these variables before running a simulation to observe their influences. All these unnecessary hassles will significantly lower the users' interests in using the simulators, and also reduce their interactivity.

With the fast developing tablet and ultra-mobile PCs, we have seen ample opportunities of employing sophisticated pen-based computing technologies to improve the interactivity of simulators in order to enhance the learners' experience to learn, reason or visualize with simulators in more effective ways. Therefore, in a recent pen-based simulator development project awarded by the Microsoft Research Asia (MSRA), we proposed to use the Microsoft digital ink library to support fast symbol/character recognition and the eXtended Markup Language (XML) technologies to flexibly define various models of computer architectures so as to build an innovative and pen-based simulator for mobile computing devices. With pen-based or other inputs, our simulator allows the instructors/students to flexibly add or modify instructions that will generate *live* animations to facilitate interactive discussion for teaching undergraduate to postgraduate courses. Besides, our simulator has the full potential to support research on computer systems through visualization of the latest results generated out of new computational models or optimization strategies. A prototype of our simulator was completed and then released to all our Year-1 students for trials in the last month in which we collected some initial and positive feedbacks. A more vigorous evaluation was planned and would be conducted by the end of this spring semester. All in all, there are many interesting directions for further investigation including the integration of relevant course materials in the form of digital resources or pointers to online databases into our simulator, and a careful study of the pedagogical changes brought by our innovative and pen-based simulator.

This paper is organized as follows. Section 2 reviews some previous works relevant to our proposal including the use of computer simulation games in education and pen-based computing. Section 3 details the system architecture design of our interactive and pen-based COMPAD simulator to enhance learners' experience on mobile devices. We give an empirical evaluation of our proposed simulator on various criteria in Section 4. Lastly, we summarize our work and shed lights on future directions in Section 5.

2. Related Works

In this section, we would firstly consider some existing command-line or windows-based emulators that are customized for specific computer architectures. Later, we would review some previous work on pen-based computing that is relevant to the subsequent discussion of our proposal.

- Existing emulators: Emu8086 (8086 emulator, 2009) is a commercially available windows-based emulator, mainly customized for the study of the 8086 family of microprocessors including the Intel's Pentium and Athlon (AMD), with an integrated 8086 assembler. The source program written in 8086 assembly instructions is compiled by the assembler and then executed on the emulator step-by-step to monitor the registers, flags and memory during program execution. Besides, the MC68HC11 is another windows-based emulator, providing an educational version with limited functions, for assembly programming on the Motorola MC68HC11 micro-controller. Furthermore, the MicroAsm is a free integrated development environment (IDE) for assembly programming on 8086 instructions. It includes a source editor and 8086 assembler. After all, most of the existing emulators are customized for specific computer architectures.
- Pen-based computing: MathPad^2 (LaViola, 2004) allows users to flexibly sketch their mathematical expressions for calculation through intelligent handwriting recognition techniques. Besides, the Java applets available in MathPad^2 provide both interactive and dynamic illustrations for students to explore about the underlying mathematical principles, perform symbolic manipulation, solve equations, and also plot graphs. Other than applying pen-based computing for mathematical reasoning, NuSketch (Forbus, 2001) is designed as a general-purpose multimodal architecture to support sketching. The ink processor of NuSketch accepts any pen-based input to extract useful data through simple signal processing techniques for the multimodal parser to work on. The extracted data can be used for military planning to schedule for any operation. Intrinsically, it can be a good alternative to replace the sketching of military remarks traditionally written on papers or maps. Specifically, a carefully designed vocabulary of visual symbols in NuSketch can be used to represent terrain features, military units, and tasks assigned to units for military planning.

3. System Design

The COMPAD is a visual simulation tool that clearly shows the internal registers, memory and more importantly the data flow of the computer architecture. Essentially, it is a very flexible, model-based and also pen-based simulator that supports the program execution on various computing systems. Given an assembly source program, possibly inputted via the stylus pen of tabulate PC or selection of commands from its drop-down menu, and the underlying execution model specified in several configuration (or schema) files, the simulator can quickly show the key events such as changes in registers or memory, and lastly the computed results through a dynamically generated sequence of animation. The speed of the animation can be flexibly adjusted by the end users. The pen-based input may naturally increase the interactivity of the system with the users since users can easily modify any part of an assembly program on the screen, and instantly visualize about the generated results.

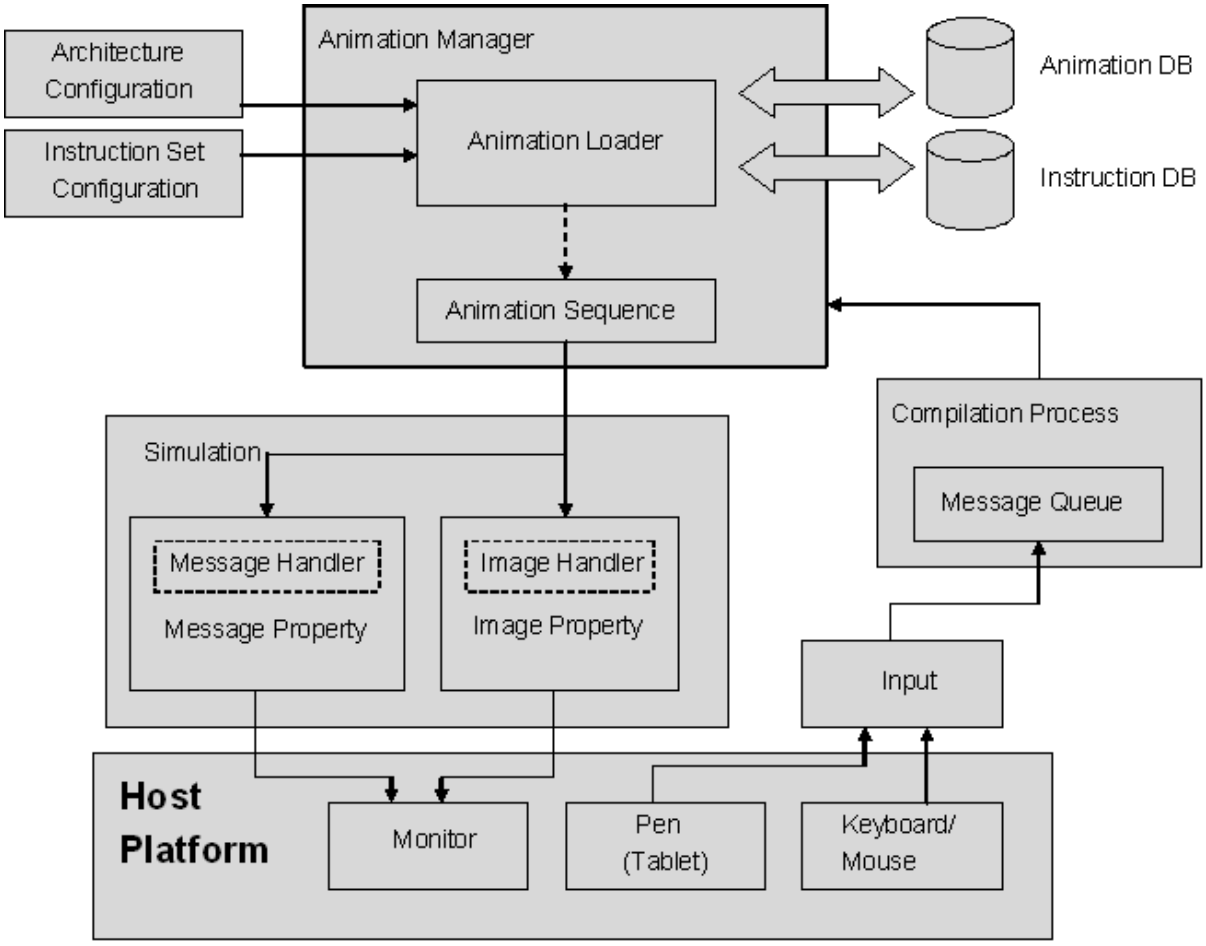

Fig. 1 – The System Design of the COMPAD Simulator

Figure 1 gives the system design of our proposed COMPAD simulator in which the core function is provided by the COMPAD simulation engine. The simulation engine reads in the 3 configuration files as the architecture configuration, the instruction-set configuration and the execution model as required for any specific computer system, and then interpret the inputted source program to generate the sequence of attractive animation to highlight the significant events occurred for registers or other key components of the underlying computer system during the program execution. Whenever a part of the program is updated, an interrupt request will be produced to enforce the COMPAD simulation engine to regenerate

the part of animation codes being affected so as to ensure a faster response time and therefore higher interactivity with the end users. Besides, depending on the range of input devices available on the host (desktop, tablet or ultra-mobile PC) platform, the user can choose to use various input devices including the stylus pen, keyboard or mouse to interact with our COMPAD simulator through selecting a function or writing an instruction to modify the concerned program.

Lastly, it is worth noting that the behavior of each computer system for program execution can be essentially captured by the 3 configuration files as mentioned above. Clearly, our proposed COMPAD simulator is very adaptive and flexible since our simulation engine can actually simulate a completely different computer system after a new set of configuration files is loaded into our COMPAD simulator. For instance, the first version of the simulator COMPAD-PRO is targeted to simulate the computer architecture of the Motorola MC68HC11 micro-controller. The file 68HC11.xml specifies the configuration of the underlying computer architecture that consists of the configuration of the random-access memory (RAM), read-only memory (ROM), various registers and the 8-bit accumulator, etc. All in all, other architecture models such as pipelined computers can be readily integrated into our proposed COMPAD simulator by updating the concerned configuration files.

4. Prototype Implementation and Evaluation

To demonstrate the feasibility of our proposal, we implemented a prototype named COMPAD-PRO using the C# programming language on the Microsoft .NET development platform since our prototype was intended to be executed on the pen-based mobile computing devices such as the tabulate or ultra-mobile PCs. The current prototype implementation consists of approximately 3,800+ lines of source codes, including around 2,300 lines of C# codes for the COMPAD simulation engine and the interface for the assembly program input and animation, and the remaining 1,500+ lines for the 3 configuration files involved. It took 4 man-months for the design and implementation of our COMPAD-PRO simulator.

To allow users to easily check the involved operations and computed results of the simulation, our COMPAD-PRO simulator has 4 execution modes available as E1–E4, supporting 3 different types of animation and a fast mode without any animation. Basically, E1 is the mode of executing all instructions in the input area. It will activate the compiler to convert to the corresponding machine language for storage in the main memory. The animation will be displayed with some messages to highlight the key event occurred for each step of the simulation. The simulation will stop when the last instruction is executed. E2 is the mode to execute a single instruction for speculation. E3 is the commonly available debugging mode to monitor on the execution of the involved instructions in a step-by-step manner. The user can stop the animation at any step to check on the computed results of the current simulation. E4 is the mode of execution without any animation. It can be used to compare the difference in the computed results after a small part of the program is modified.

Figure 2(a) shows the general user interface of our COMPAD-PRO simulator ready for pen-based input of program instructions. It consists of tabbed window display on the left-hand side for the input of assembly program and the display of register values. Users can flexibly modify any instruction inside the assembly program by handwriting through the pop-up window for command recognition on the right-hand side. The middle area is to display the generated animation with a “speed” button for end users to flexibly adjust an appropriate speed of the animation. The bottom part is the message window that shows the key events or operations happened during the animated program execution. Figure 2(b) shows the resulting

value stored in each memory cell, namely the memory content, after executing the concerned assembly program.

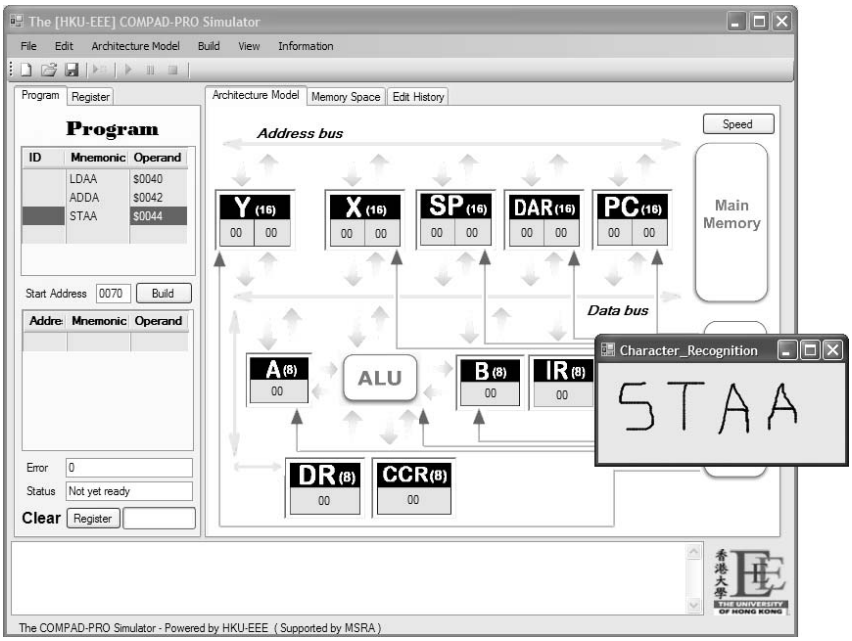

Fig. 2(a) – The User Interface for the Pen-Based Input of our COMPAD Simulator

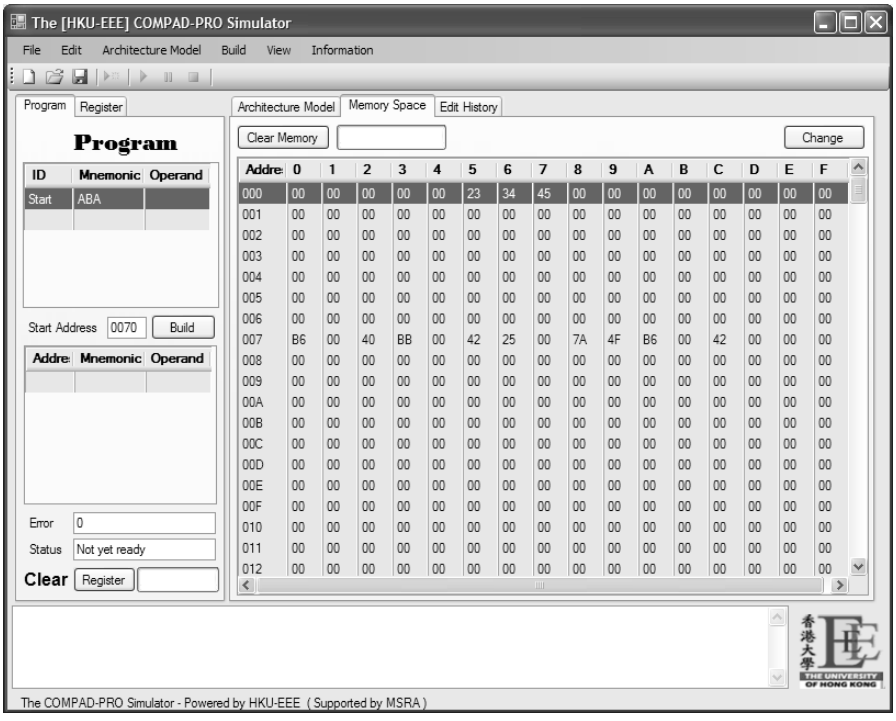

Fig. 2(b) – The User Interface for the Memory Content of our COMPAD Simulator

Basically, our COMPAD-PRO simulator provides two methods for users to input their assembly program. The first one is a writing pad that provides character recognition for any hand-written instruction inputted by a stylus pen, mouse or other pointing devices. The result

of the recognition will be displayed in the corresponding tabbed window for program input. Besides, in case where handwriting is inconvenient, users can still use the fast input method of drag-and-drop to choose an appropriate instruction from the drop-down list of instructions available on the left column of the window, and insert the selected instruction into the current assembly program being edited.

The main product of our COMPAD-PRO simulator is the generated animation as display in the middle window. This is the area where the whole emulated computer system with its registers, main memory and buses can be clearly seen. The animation shows the most information for all the detailed steps of operations including the communication between different components inside the concerned microprocessor and changes of values stored in the registers or main memory.

The message displayed in the bottom window summarizes the latest information involving the key events occurred, and also the important changes in the contents of the registers. The displayed message closely follows the pace of animation to facilitate the students' learning or revision. In addition, when there is any change in the computer architecture used, a message will also be shown in this textbox to notify the user.

During the development of our prototype, one of the major challenges is to formulate a flexible encoding scheme for the various assembly instructions involved for different computer systems. The scheme should be adaptive and easy-to-use so that users can easily add new instructions into our simulator without much coding effort. In view of this requirement, we decided to use a schema file in the form of the Microsoft Access file so as to allow users to specify about their new instruction to be added by simply updating the schema file. In this way, even a novice user can add new instruction into our simulator without modifying any part of our program.

About the evaluation, we have carefully planned to conduct the preliminary evaluation of our prototype on in a Year-1 core course, namely the ELEC1401 – Computer Organization and Microprocessor, with around 100 students in two phases. In the first phase, around the beginning of the 2nd semester, we would allow all the students to download our prototype and try their own laptop PCs installed with mouse pointing devices or pen-based tabulate PCs provided by our Department. Before the release, we would have already thoroughly tested our COMPAD-PRO simulator on various computing devices including the Samsung Q1 UMPC, Fujisu U2010 Lifebook, or the IBM X40 notebook PCs commonly used by our Engineering students. A demonstration session will be provided to our students after the relevant topic on program execution is taught so that students can later explore about the various features of our simulator at their own pace. After a trial period of 2 – 3 weeks, we would conduct a preliminary survey in the form of questionnaire mainly to study about whether the animation features available in our prototype is effective or sufficient to *motivate* the learners' interests or not, and also whether our simulator is easy-to-use or not. By the end of the semester, we would come back to conduct the evaluation in the 2nd phase with both questionnaire and interviews to investigate on whether our simulation tool can really promote the learners' interests and possibly their performance as well after a longer period of uses. After some careful analysis, a detailed report about the evaluation results will be published in some international conferences on educational technologies, and also available under our project website around October, 2009.

5. Concluding Remarks

In this paper, we proposed an innovative and pen-based COMPAD simulator to enhance both education and research in computer systems. Being model-based, our proposal is adaptive and different from many commercially available Windows-based emulators that are

customized for specific computer architectures. Besides, our COMPAD simulator allows learners to flexibly modify any part of a program through pen-based inputs, and instantly visualize about the computed results. In this way, the COMPAD simulator can support not only education but also research such as instruction scheduling in computer systems.

The simulator has been tested by different users who gave us fairly positive feedbacks. Around the mid of February 2009, the simulator will be released to motivate the students' learning interests, and facilitate their understanding on the operations of modern computer systems in the Department of Electrical and Electronic Engineering, the University of Hong Kong. More importantly, our proposed COMPAD platform will be useful not only to support undergraduate or postgraduate teaching, but also to facilitate research discussion. We had spent 4 man-months to work on the simulator and thoroughly tested the prototype. A project website is being set up for the students to download the simulator software or any updated schema files for the concerned computer architectures.

Clearly, our proposal of adaptive and pen-based simulator for computer systems has opened up many interesting directions for further investigations. First, the current implementation of our COMPAD simulator to generate attractive animation is still computationally intensive, thus with fairly poor performance when our simulator is executed on sub-notebook or ultra-mobile PCs with slower processors running at 800 MHz or below. Hence, it is definitely worth investigating on how to optimize our implementation so that our simulator can be executed with reasonable performance on other mobile computing devices such as the Apple™ iPod Touch or even smart-phones. Besides, to facilitate the teaching of more advanced undergraduate or postgraduate courses in computer systems, it is interesting to study how to include other new or more sophisticated computer architectures into our existing prototype. Furthermore, it is exciting to explore on how to improve our COMPAD simulator on providing better animation and visualization so as to support the research study such as instruction scheduling in computer systems. Last but not the least, the integration of relevant course materials in the form of digital resources or pointers to online databases into our simulator is interesting. All in all, a careful study of the pedagogical changes brought by our innovative and pen-based simulator is definitely important and worth studying.

References

Forbus, K., Ferguson, R., & Usher, J. (2001). Towards a computational model of sketching, *Proceedings of* the 6th *International Conference on Intelligent User Interface* (*IUI'01)*, January 14-17, 2001, Santa Fe, New Mexico, United States, pp. 77 - 83.

Kent NGfL. (2007). Simulations – Google Search Kent NGfL website. Retrieved: June 15, 2007, from http://www.kented.org.uk/ngfl/software/simulations/index.htm.

LaViola, J. and Zeleznik, R. (2004). MathPad^2: A System for the Creation and Exploration of Mathematical Sketches, in *ACM Transactions on Graphics*, 23(3):432-440, August 2004.

SCS Publication team. (2007). Simulation: Transactions of the Society for Modeling and Simulation International website. Retrieved: June 3, 2007, from http://www.scs.org/pubs/simulation/simulation.html.

SuperKids Software development team. (2007). *SuperKids Software Review of RockSim – Model Rocket Design and Simulation Software*. Retrieved: June 5, 2007, from http://www.superkids.com/aweb/pages/reviews/science/06/rocksim/merge.shtml

The 8086 emulator development team. (2009). *the 8086 Microprocessor Emulator with Integrated 8086 Assembler website*, Retrieved: January 27, 2009, from *http://www.emu8086.com/*.

The FSim development team. (2009). *The MS Flight Simulator website*. Retrieved: June 5, 2009, from http://www.fsinsider.com/Pages/default.aspx.

The SimBusiness development team. (2009). Aspyr - *The Sims™ 2 Open for Business website*. Retrieved: June 15, 2009, from http://www.aspyr.com/product/info/3.

The SimCity development team. (2009). *The SimCity Societies*. Retrieved: June 15, 2009, from http://simcity.ea.com/.

The Sims development team. (2009). *The Sims Official Site*. Retrieved: June 15, 2009, from http://thesims.ea.com/.

Ward, J., Carroll, P. (1998). Can the Use of a Computer Simulation Game Enhance Mechanical Reasoning Ability: an exploratory study. *Working Paper Series MCE-0998*, 1-20. Retrieved: June 10, 2007 from http://citeseer.ist.psu.edu/431172.html.

The Wikipedia development team. (2009). *Simulation – Wikipedia, the free encyclopedia*. Retrieved: June 1, 2009, from http://en.wikipedia.org/wiki/Simulator

Bibliography

Vincent Tam completed his Ph.D. in computer science from the University of Melbourne, Australia in 1998. Dr. Tam was the winner of Innovative Teaching Award (2000) in the School of Computing, National University of Singapore. He is a senior teaching staff in the University of Hong Kong, the part-time deputy director of an EMB-funded project on e-learning packages, and also the principal investigators of several teaching development projects.

Johnny Yeung received his BSc(Eng) from the University of Hong Kong in 2007. He is currently a full-time postgraduate student working for this interactive and pen-based simulation project for different computer systems.

C.H. Leung received his BSc(Eng) and PhD degrees from the University of Hong Kong, and his MEng degree from McGill University, all in electrical engineering. He has been teaching at the University of Hong Kong since 1986.

Edmund E. Lam is experienced in image and signal processing techniques and computer algorithms. His main research work is in computational imaging, but he is also very interested in fostering the use of technology in teaching and learning.

Diane J. Salter has 15 years of experience internationally related to e-learning initiatives in higher education settings including the Universities of Toronto and currently at the University of Hong Kong as an Associate Professor in the Centre for Advancement of University Teaching (CAUT). Her work has been published and presented internationally and she recently was principal project lead on a TDG at Polytechnic for HK $1,125,000. Her current research is focused on curriculum development and innovative approaches to guiding teaching and curriculum design incorporating pedagogically sound applications of technology to enhance active learning.