1. PROBLEM STATEMENT

Computing is the foundation of modern society. A proficient computing workforce is essential for maintaining the country's leadership and competitiveness in the global economy [4]. For this reason, the recent decline in enrollments across Computer Science (CS) departments [43, 44] and the decrease in student diversity [42] pose significant challenges to the continuation of the nation's prominent position in the global high technology arena. An immediate solution is urgently needed. For the health of the national economy, and to sustain successful and vibrant software companies like Microsoft, we must build excitement and enthusiasm for our discipline in order to attract a bright new generation of students early in their academic careers. In the recent professional gatherings, numerous CS educators have reported that incorporating computer gaming in programming classes creates high levels of excitement and motivations. This phenomenon is observed firsthand by the PI in his graphics programming classes [9, 36, 38], where gaming components have drastically increased the enrollments and engaged students' interest in challenging physics and mathematics topics [40]. As eloquently described in the RFP for this proposal, integrating computer gaming into CS1 and CS2 (CS1/2), the first programming courses students encounter, is a promising strategy for recruiting and retaining potential students.

We propose to design and implement *self-contained* games-themed programming assignment modules based on the Microsoft XNA framework for CS1/2 courses taught in C#. The modules will be *self-contained* so faculty members currently teaching CS1/2 courses can pick and choose from our games-themed assignments, and combine them with their own non-games assignments. Faculty will be able to incorporate these modules with a minimum of effort, a minimum knowledge on computer gaming, and a minimum investment in computer-gaming specific curricula. We believe that the key to successful, widespread adoption of computer gaming within CS1/2 is to provide educators with easy, pre-packaged modules that can be incorporated quickly and easily into existing courses. Based on our experience developing the games-themed assignments, the second objective of our project is to design and implement template-tutorials to enable faculty members with little or no experience in graphics and gaming to develop their own games-themed assignments.

With the assessment tools developed from an on-going NSF project [33], we will integrate systematic evaluations into every phase of this project. The quality of the assignment modules will be independently assessed for academic merit and gaming content. The effectiveness of the assignments will be assessed based on student learning outcomes from before and after the assignments are implemented.

To the students, the results of our project will be programming assignments that build games for the XBOX 360 console. This will provide an immediate draw to young students excited about being able to build games for their favorite entertainment device in their first programming course. In addition, since these programs will be developed with the same IDE (Visual Studio 2005, XNA Express) based on the same programming language (C#) as their other non-games assignments; students will be able to better appreciate the importance of the tools and the abstract concepts they are learning. In these ways, these assignments can increase students' enthusiasm for the discipline and enhance their learning experience.

To the faculty members, our assignment modules provide a simple pathway for integrating gaming into their classes. Very importantly, these modules will not govern how abstract topics should be presented: the core intellectual identity of each faculty's curriculum. In this way, the assignments are tools for enhancing existing courses and do not require the faculty to make broader modifications to the curriculum or assignments.

As a follow-up to this project, at the invitation of his department chair, the PI plans to design a new programming class based on the assignment modules from this project to teach students with Java-CS1 background about games programming. Many of the colleges in the Pacific Northwest teach CS1/2 in Java, so the newly developed class can serve as a template for departments to integrate gaming into their curricula.

1.1 Context For Our Project: Survey of Related Work

There are many types of "games" that are suitable for teaching CS subjects, including many non-interactive games (e.g., [15]) or games that are based on dedicated devices (e.g., Lego robots [23]). In the following discussion we focus on *interactive graphical computer games*. When we examine recent efforts in integrating gaming into CS classes, we observe three general categories.

- **Games development classes**. These are entire curricula (e.g., [13]), individual classes (e.g., [25, 22]), or capstone projects (e.g., [20, 26]) designed specifically to develop new games as an end product. When evaluated against the curriculum framework proposed by the IDGA education committee [2], we see that these classes cover all the major core topic areas. Students in these classes must be concerned with all aspects of a real game production including entertainment value, visual quality, audio effects, etc.
- **Games programming classes**. These are classes (e.g., [41, Kuffiner's CMU course]) designed specifically to study technical aspects of building games. For example, topics covered may include path planning algorithms, terrain representation, etc. The topics covered in these classes are general and typically can be applied to different domains. These classes concentrate on covering the games programming topic area in the IDGA curriculum framework.
- **Games client classes**. These are existing CS classes that creatively integrate gaming into their existing curriculum. Typically, games are used as programming assignments (e.g., [8, 16, 41, 10, 21, 37]), or to teach abstract concepts (e.g. [11, 12, 14, 18, 19]), or as an example application area to teach the concepts involved in an entire topic area (e.g., [7, 3, 40]). These are traditional CS classes that exist independent from games programming. These classes are actually *clients* of games development because they use games development as a vehicle to deliver specific abstract concepts. After these classes, students are expected to understand the abstract concepts, and not games development.

Ours is a *client of games* approach where we propose to design games-themed assignments to teach foundational programming concepts. The earliest work in this area (e.g., [8, 16]) adapted games almost "anecdotally" without holistic considerations. Other pioneering work (e.g., [12, 14]) implemented new "games or graphics"-based problem solving paradigms to facilitate learning of algorithmic design. To adopt these approaches, faculty members must forfeit the intellectual investment of their own curriculum and learn to teach within the new paradigms. Some more recent work (e.g., [10, 21]) described efforts in organizing CS1/2 as games development classes where fundamental programming concepts are presented in the context of building interactive games. These approaches demand prior gaming and/or graphics knowledge which render the adaptation challenging for many faculty. Our proposed work is most similar to [41, Sweedyk's CS121 course], where games-themed assignments are designed with no assumptions on how concepts are presented. However, instead of a sophomore-level software engineering course, our focus will be CS1/2 courses. More importantly, Sweedyk never intended for her assignments to be *self-contained* modules suitable for adaptation by others.

1.2 Details of Our Project

When considering games-themed assignments in the context of CS1/2 courses, the following are important factors.

- **Academic integrity:** CS1/2 are the introductory foundation courses. It is crucial that students understand basic concepts and the context of that knowledge. Confusing these courses with "*gaming courses*" will create disappointment and resentment in later CS courses. In addition, this can limit students' potential career outlook.
- **Gaming content:** The inclusion of computer gaming must enhance specific and relevant learning objectives. Including games solely for the purpose of engaging students will cause distraction and will waste time. The choice of games is also

important. Many commercial games are created for stereotypical gamers (e.g., violent games, etc.). Without careful consideration, gaming content may repel under-represented groups (e.g., women).

- **Intellectual investment and implementation barriers:** CS1/2 are well established classes where faculty have significant intellectual investment in their implementation of the curriculum. In addition, a significant percentage of these faculty members did not grow up playing computer games or may not be familiar with the involved graphics programming. Both of these factors contribute to faculty's hesitation to adopt gaming in their courses.

Existing games-themed programming classes tightly couple their curricula with specific approaches. These are interesting and exciting pioneering work, however they also tend to be difficult to customize and challenging to adopt. Our project proposes to design, implement, and assess *self-contained* games-themed programming assignment modules based on the Microsoft XNA framework for CS1/2 courses taught in C#. To ensure ***academic integrity***, each assignment module will include clearly defined learning objectives; assessment tools; and sample student learning outcomes. For example, a 2D cell-based game (e.g., traversing a 2D map for scavenger hunt) can be used to enhance 2D array concepts. The associated assessment tools would include student self-tests (e.g., traversal and manipulation of 2D arrays, etc.); and sample evaluation questions for the faculty (e.g., questionnaire for student self-reflection, questions for quiz, etc.). The sample student learning outcomes will be results from this project showing student learning with/without and before/after games-themed assignments. In this way, faculty members can align expectations with likely outcomes before adapting an assignment module. When designing the ***gaming content***, we will work with advisors from the local games industry. To maintain gender impartiality, we will especially consult our contacts with local studios that design games for girls (e.g., we have existing relationship with, including graduates working at Her Interactive Inc.). To preserve and support ***intellectual investment*** in curriculum design, the assignment modules will be *self-contained*. For example, if deemed appropriate, a faculty member can design her own non-games assignments for enhancing looping concepts while still choose our scavenger hunt game as the assignment for 2D arrays. To ***encourage adaptation*** and to ***lower implementation barriers***, each module will be accompanied by a detailed laboratory manual describing the mechanics and procedures. A sample solution would also be provided for the faculty. Finally and very importantly, we will develop a set of template-tutorials for interested faculty to design and implement their own games-themed programming assignments.

Platform for implementation: the existing games engines suitable for academic uses (e.g., [5, 6, 24]) do not support the XNA framework and do not allow XBOX 360 based programming assignments. Although the Torque X engine [45] appears promising, the on-going effort is currently in pre-beta stage. In all cases, these engines are complex and typically dictate the structure of the programs one could develop. Recall that one of the most important aspects of our work is that the assignment modules can *replace* existing ones and thus allow *gradual* integration of gaming with *minimum* interruption to existing courses. To accomplish this goal, we need a supporting *lightweight* library for rapid prototyping of interactive graphical programs. UWBGL [39], the results from an on-going NSF funded project [33], is such a library. This library is designed to support rapid development of games-themed Concept Demonstration Applications (CDAs) [31] for teaching computer graphics [9, 36, 38]. Based on this library, we have successfully developed games-themed programming assignments for graphics [40] and algorithm analysis classes [37]. Currently, UWBGL is based on C++ and Microsoft Direct3D API. However, this library is highly portable and earlier versions have been successfully ported to C# and OpenGL. We will port the current version of UWBGL to the XNA framework and develop our games-themed assignments based on the ported UWBGL. Because the CDAs are designed to *demonstrate* concepts, we fully expect that porting these applications over to the XNA framework will contribute significantly towards the development of the template-tutorials.

Courses for testing: we must verify students are learning the same concepts before and after the implementation of games-themed assignments. In addition, we must demonstrate the assignments are indeed *interchangeable* with existing

CS1/2 assignments taught in C#. Professor Michael Panitz of Cascadia Community College has graciously offered his well established C# based BIT 142/143 (CS1/2) courses as the platform for our study. By the time our project commences in fall 2007, Professor Panitz will have taught these courses for 7 years (the most recent 2 years in C#). These classes are well established with many successful graduates, perfectly suited as a platform for testing our assignments.

2. Expected Outcome

We expect to deliver 6-8 XNA based programming assignment modules. Each module would include learning objectives, assessment tools, sample solution source code, sample student learning outcomes, and a detailed lab manual. In addition, we will deliver a set of template-tutorials providing a step-by-step guide for faculty to develop their own games-themed programming assignments. As discussed in Section 5, we also expect scholarly publications resulting from this work.

We believe the most significant outcome from this work would be ***excitement from empowerment***: for both students *and* faculty. For students, the XNA based assignments empower them to apply the mundane programming construct and abstract programming concepts in implementing games that they can play on their favorite gaming console. For faculty members, the XNA based assignments provide a simple and clear pathway and thus empower them to integrate games into their courses. Pedagogically, from the programming language perspective, C# and Java are similar languages. Currently, many CS1/2 courses are based on Java. With the success of XNA-based games-themed programming assignment modules, we expect an increase in interest from faculty members switching from Java to C# in teaching their CS1/2 courses.

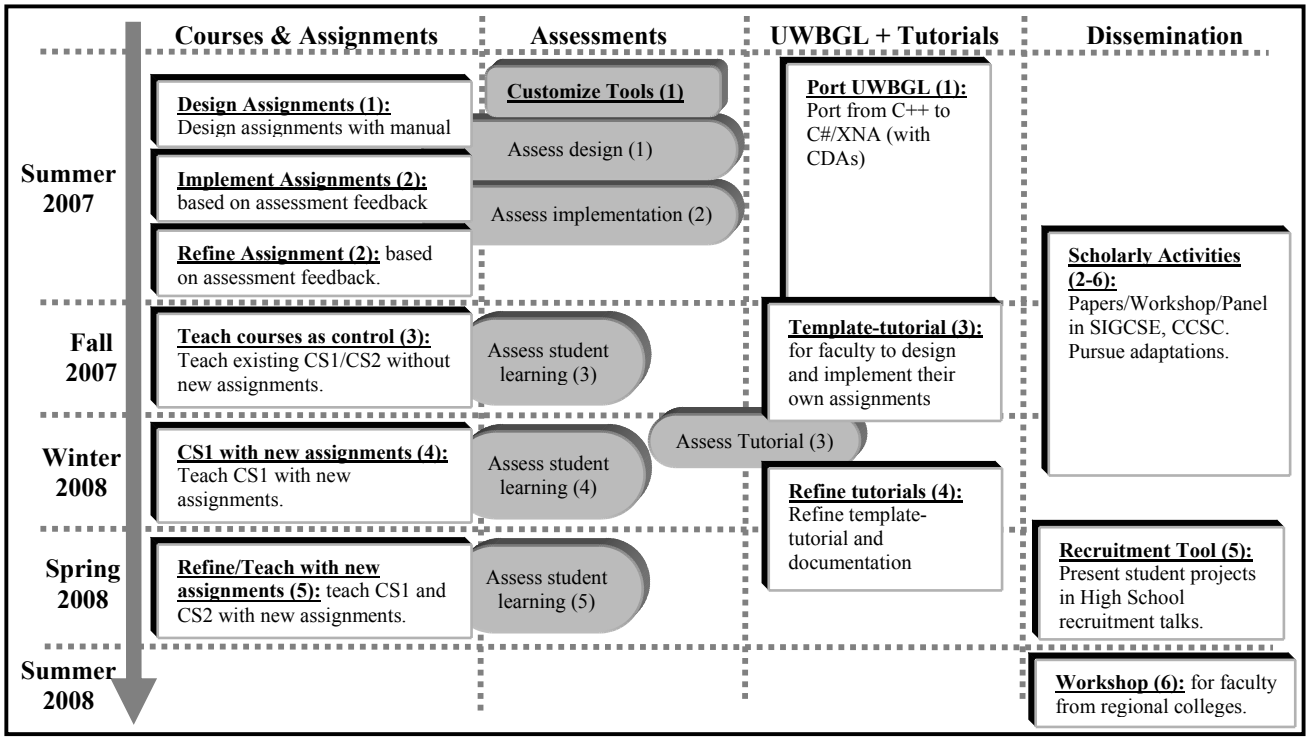

Figure 1: Project Schedule

3. Schedule

Figure 1 is an overview of the schedule for our project from summer 2007 to summer 2008. Our one-year schedule includes two summers to facilitate initial preparation and wider dissemination of the results at the conclusion of the project.

Courses and Assignments: The games-themed assignment modules will be designed and implemented over summer 2007 (Aug-Sep). The BIT 142/143 (CS1/2) courses will be taught *without* the new games-themed assignments in fall 2007

(Oct-Dec) and will serve as the control group. To ensure quality and integrity, we will introduce the new assignments gradually and with careful monitoring. For this reason, in winter 2008 (Jan-Mar) we will only introduce the new assignments in BIT-142. We will refine all assignment modules based on assessments results from this class. In spring 2008 (Mar-June), we will introduce the new games-themed assignments to both BIT 142 and 143. **Assessments:** We will adopt and customize the tools from our prior [30, 34] and currently on-going NSF funded [9, 33, 40] course work and student learning assessment projects. These tools will be used to assess the assignment modules and student learning outcomes of all BIT 142/143 classes. **UWBGL and Tutorials:** Although only the UWBGL library is required for this project, the PI's department has committed in-kind support for porting all of the CDAs to C# and XNA. We will begin the porting effort in early summer 2007. By the August time frame, the XNA-based UWBGL will be ready for supporting the new assignments. We will begin designing tutorials to help educators build their own assignments after we have firsthand experience and understand the procedures involved. It is expected that the initial tutorials will be ready for testing in late fall 2007. We will engage a senior graduate teaching assistant to design and implement additional games-themed assignments for CS1/2 courses using our tutorials. Based on the feedback from the testing, we will refine the tutorials to be re-tested in winter 2008. **Dissemination:** Please refer to the discussion in Section 5.

The following are the measurable milestones for the project (the numbers behind each task in Figure 1 correspond to the milestone number):

Funding commence: July 15th 2007 **Milestone 1:** **By** Aug 15th 2007	• Assessment tools designed and customized (*Sung* + *Reed-Rosenberg*) • XNA based UWBGL ported (*Sung* + *Student Intern*) • Topic & learning objectives for assignment selected (*Sung* + *Panitz*) • Assignments designed and lab manual outlined (*Sung* + *Panitz*) • Assignment design assessed for academic and games integrity (*TBN Consultants*)
Milestone 2: **By** Sep 15th 2007	• Assignment implemented/refined and lab manual completed (*Sung* + *Panitz*) • Assignment implementation assessed for academic/games integrity (*TBN Consultants*) • SIGCSE: Games-Themed Assignment Modules (*Sung/Panitz/Reed-Rosenberg/MSR*)
Milestone 3: **By** Dec 2007	• Taught CS1/2 (BIT 142/143) as control (*Panitz*) • Assessed student learning outcome (*Sung* + *Panitz* + *Reed-Rosenberg*) • UWBGL tutorials designed/implemented/tested (*Sung* + *Student Intern* + *TBN TA*) • CCSC Workshop proposal + Project Summary for MSR Academic Days
Milestone 4: **By** Mar 2008	• Taught CS1 (BIT 142) with new assignments (*Panitz*) • Assessed student learning outcome (*Sung* + *Panitz* + *Reed-Rosenberg*) • UWBGL tutorials refined/retested (*Sung* + *TBN TA*) • Attend/Feedback from MSR Academic Days + SIGCSE Conference (*Sung* + *Panitz*)
Milestone 5: **By** June 2008	• Taught CS1/2 (BIT 142/143) with new assignments (*Panitz*) • Assessed student learning outcome (*Sung* + *Panitz* + *Reed-Rosenberg*) • Local High School recruitment trips (*Sung* + *Panitz*)
Milestone 6: **By** July 2008	• Organized workshop for faculty members from regional colleges (*Sung/Panitz/ Reed-Rosenberg/MSR*) • Package modules for MSDN Academic Alliance Repository (*Sung*)
After project: By Sep 2008	• SIGCSE Paper: Experience in Integrating Games-Themed Assignment Modules in CS1/2 (*Sung/Panitz/ Reed-Rosenberg/MSR*) • SIGCSE workshop/panel proposal (*Sung/Panitz/ Reed-Rosenberg/MSR*) • Design a new XNA-based games programming class based on the assignments (*Sung*) • Analyze enrollment impact (*Sung* + *Panitz*)

4. Use of Funds

5. Dissemination and Evaluation

Our dissemination plan is organized around three main objectives: sharing of intellectual learning (conference/journal publications), sharing of materials (workshop/panel to encourage adoption of the material), and student recruitment. Based on our experience [9, 32, 34-40], we believe the results from this project warrant at least three conference papers: one on the design and implementation of the self-contained and *replaceable* assignment modules (to be written in Sep 2007 for SIGCSE 2008), one on the student learning assessments (to be written in summer 2008 for SIGCSE 2009), and one on approaches used to encourage adoption of the assignments (suitable for CCSC-NW type regional conferences). A potential fourth journal paper would summarize the entire project (e.g., Journal of Games Development). In early summer 2008, at the conclusion stage of this project we will organize a workshop for faculty members from regional institutions to share our results and to pursue adoption. We plan to demonstrate student projects from these courses during High School recruitment trips (which will include the Redmond High School). At the conclusion of the project, we will package and submit our assignment modules to the MSDN Academic Alliance Repository.

6. Other Support

UWB will acquire the XBOX units and loan these units to Cascadia Community College during the project. New graphics card will be purchased for Professor Panitz to upgrade the machine in his office to support more demanding graphics/games applications. (*) Recognizing not all of the CDAs are relevant to this project, if funded, the Computing and Software Systems (CSS) will contribute one of the internship students during summer 2007 for the cost of $4,000.

7. Qualifications of Principal Investigator

The PI of this project, Professor Kelvin Sung, received his PhD in CS from the University of Illinois at Urbana-Champaign. The PI's research interests are in the areas of high quality image synthesis (e.g.,[51]). Before joining UWB, The PI was with Alias|Wavefront (now part of Autodesk) where he was a chief designer of the Rendering module of the academy award winning Maya software system and co-designed a patented motion blur algorithm [52]. More recently, the PI has been funded by both UWB [28-31] and by the NSF [33] to study issues related to CS education [9, 32, 34, 36-40]. Many of these experiences and results are directly applicable to this project. For example, the PI has recent experience in designing games-themed programming assignments [37, 40]; his experience in implementing CDAs [39] are applicable to the development of the template-tutorials; and his experience in assessing courseware and student learning [34, 36, 40] are applicable to the assessment of the quality and effectiveness of the games-theme assignments. After graduation, many of the PI's students have taken jobs at local games companies/studios including: Crankypants, Electronic Arts, Flying Labs, Amaze Entertainment, Griptonite, Handheldgames, Her Interactive, Humongous Entertainments, Wild Tangent. The PI will work with Professor Michael Panitz from the Cascadia Community College (CCC) in teaching and assessing the effectiveness of the XNA assignments. Professor Panitz received his Master of Engineering (CS) from Cornell University, and worked at Microsoft before joining CCC. Now in his 6th year at CCC, Professor Panitz has taught BIT 142/143 (CS1/2) almost continuously, first in C++ and since 2005, in C#. His students have used the knowledge they acquired in his classes to advance their careers at Microsoft, to succeed in CS programs at four year schools, and to succeed at programming positions taken directly after graduation. Dr. Rebecca Reed-Rosenberg, the Director of the UWB Teaching and Learning Center, will be in charged of designing, customizing, and administering the assessment instruments. Dr. Reed-Rosenberg is currently working with the PI on the NSF funded project [33] in assessing the PI's computer graphics courses [40]. Other projects the PI and Dr. Reed-Rosenberg collaborated on include [30, 34].

8. Methods of Assessment

As in our previous and on-going coursework evaluation projects [30-34, 40], we will continue with the mixed method approach [50] to assess our project. Informed by the recent assessment scholarship [46-49], we will continuously evaluate each phase of our project and incorporate those feedbacks in order to adjust our ongoing progress. For the assessment of the ***games-themed assignments***, the formative evaluations of academic merit during design and implementation phases will involve outside faculty performing qualitative design document reviews; interviews with the PI; and filling out a quantitative questionnaire. We will design the questionnaire based on the learning objectives and expected competencies found in [1]. The formative evaluations of games integrity will be based purely on qualitative design document reviews and interviews by the consultants from games industry. The summative evaluation of the games-themed assignments will be conducted after the courses are taught. We will submit selected student assignments together with similar assessment instruments, to the independent outside faculty and games industry consultants for evaluations. For the assessment of ***student learning outcomes***, we will reuse many of the instruments and follow the procedures from [34, 40], where we will administer pre/post-tests, collect student perceptions, grade the assignments, and gather faculty self-assessment based on faculty reflections on in-class activities. This data will be collected for both the control and the latter courses with the new assignments for comparison/analysis. For the assessment of ***template-tutorials***, the first stage of formative evaluation will be *implicit* where the qualitative feedback will come from our own experience from developing the games-themed assignments. The second phase of the formative evaluation will be conducted much like a mini "software product release". We will recruit a target user (a senior graduate student in CS potentially from the UW Seattle campus) with minimal to no prior knowledge in gaming and/or graphics programming. This user will be assigned the task of developing a games-themed assignment based solely on the tutorials. After the exercise, the user will be interviewed and will fill out an evaluation questionnaire. We will refine the tutorials based on this feedback. This evaluation cycle will be repeated as the summative evaluation of the tutorials.

REFERENCES

[1] Computing curricula 2001, Computer Science, December 2001. Final Report, The Joint Task Force on Computing Curricula IEEE Computer Society and ACM, *http://www.acm.org/education/curricula.html*.

[2] IGDA curriculum framework report version 2.3 beta, February 2003. International Game Developer's Association, *http://www.igda.org/academia*.

[3] Artificial intelligence for computer games, 2006, University of Sao Paulo (USP/SP), Microsoft Academic Alliance Repository Newsgroup, Object ID: 6210, *http://www.msdnaacr.net/curriculum/pfv.aspx?ID=6210*.

[4] CISE Pathways to Revitalize Undergraduate Computing Education (CPATH). *Request For Proposal, National Science Foundation, Directorate for Computer and Information Science and Engineering*, 2006.

[5] Dxframework: A pedagogical computer game engine library, U. of Michigan, 2006. *http://dxframework.org/*.

[6] Software enginnering for computer games, University of Sao Paulo (USP/SP), 2006. Microsoft Academic Alliance Repository Newsgroup, Object ID: 6211, *http://www.msdnaacr.net/curriculum/pfv.aspx?ID=6211*.

[7] Multi-user programming pedagogy for enhancing traditional study, Rochester Institute of Technology, 2006. *http://muppets.rit.edu/muppetsweb/people/index.php*.

[8] J. C. Adams. Chance-it: an object-oriented capstone project for cs-1. In *SIGCSE '98 Proceeding*, pages 10-14, 1998. ACM Press.

[9] E. Angel, S. Cunningham, P. Shirley, and K. Sung. Teaching computer graphics without raster-level algorithms. In *SIGCSE '06: Proceedings*, pages 266-267, ACM Press.

[10] J. D. Bayliss and S. Strout. Games as a “flavor” of cs1. *In SIGCSE '06: Proceeding*, pages 500-504, 2006. ACM Press.

[11] K. Becker. Teaching with games: the minesweeper and asteroids experience. *J. Comput. Small Coll*., 17(2):23-33, 2001.

[12] M. C. Carlisle, T. A. Wilson, J. W. Humphries, and S. M. Hadfield. Raptor: a visual programming environment for teaching algorithmic problem solving. In *SIGCSE '05: Proceedings*, pages 176-180, 2005. ACM Press.

[13] R. Coleman, M. Krembs, A. Labouseur, and J. Weir. Game design & programming concentration within the computer science curriculum. In *SIGCSE '05: Proceedings*, pages 545-550, 2005. ACM Press.

[14] W. Dann, S. Cooper, and R. Pausch. *Learning to Program with Alice*. Prentice Hall, Upper Saddle River, NJ, 2006.

[15] P. Drake. *Data Structures and Algorithms in Java*. Prentice Hall, Upper Saddle River, NJ, 2006.

[16] N. Faltin. Designing courseware on algorithms for active learning with virtual board games. In *ITiCSE '99: Proceedings*, pages 135-138, 1999. ACM Press.

[17] J. Frechtling, L. Sharp, and Ed. User-Friendly Handbook for Mixed Method Evaluations. *Directorate for Education and Human Resources, Division of Research, Evaluation and Communication*, National Science Foundation, 1997.

[18] R. Giguette. Pre-games: games designed to introduce cs1 and cs2 programming assignments. In *SIGCSE '03: Proceedings*, pages 288-292, 2003. ACM Press.

[19] S. Hansen. The game of set®: an ideal example for introducing polymorphism and design patterns. In *SIGCSE '04: Proceedings*, pages 110-114, 2004. ACM Press.

[20] R. M. Jones. Design and implementation of computer games: a capstone course for undergraduate computer science education. In *SIGCSE '00: Proceedings*, pages 260-264, 2000. ACM Press.

[21] M. C. Lewis and B. Massingill. Graphical game development in cs2: a flexible infrastructure for a semester long project. In *SIGCSE '06: Proceedings*, pages 505-509, 2006. ACM Press.

[22] B. Maxim. Game design and implementation 1 and 2, 2006. *Microsoft Academic Alliance Repository Newsgroup, Object ID: 6227, http://www.msdnaacr.net/curriculum/pfv.aspx?ID=6227*.

[23] M. McNally, M. Goldweber, B. Fagin, and F. Klassner. Do lego mindstorms robots have a future in cs education? In *SIGCSE '06: Proceedings*, pages 61-62, 2006. ACM Press.

[24] I. Parberry. Sage: A simple academic game engine, 2006. *University of North Texas, http://larc.csci.unt.edu/sage/*.

[25] I. Parberry, M. B. Kazemzadeh, and T. Roden. The art and science of game programming. In *SIGCSE '06: Proceedings*, pages 510-514, 2006. ACM Press.

[26] I. Parberry, T. Roden, and M. B. Kazemzadeh. Experience with an industry-driven capstone course on game programming: extended abstract. In *SIGCSE'05: Proceedings*, pages 91-95, New York, NY, USA, 2005. ACM Press.

[27] G. Rogers, K. Sung, and W. Kubitz. Combining graphics and windowing standards in the xgks system. *Computer Graphics Forum*, 9:229-237, 1990.

[28] K. Sung. Instructional infrastructure development and student research initiatives based on recent research efforts. *Worthington Distinguished Professor Award* ($3,500), University of Washington, Bothell, June 2001.

[29] K. Sung. The computing and software systems center for integrated teaching, learning and scholarship. *Worthington Technology Award* ($11,380), University of Washington, Bothell, June 2002. (Co-PI: M. Fukuda and C. Jackels and M. Stiber).

[30] K. Sung. Assessment and refinement of computer-base programming course for non-technical students. *Worthington Technology Award* ($10,000), University of Washington, Bothell, June 2004-2006.

[31] K. Sung. Courseware infrastructure development for supporting proposals to the nsf. *Worthington Scholar Award* ($10,000), University of Washington, Bothell, June 2004-2006.

[32] K. Sung. Building interactive graphics applications. In P. Shirley, editor, *Fundamentals of Computer Graphics*, chapter 18, pages 401-449. A. K. Peters, Wellesley, Massachusetts, second edition, 2005.

[33] K. Sung. Essential concepts for building interactive computer graphics applications. *CCLI-EMD, NSF*, DUE-0442420, $60,500, 2005-2007.

[34] K. Sung, A. Leong, and R. Reed-Rosenberg. A programming course for business mis students. *ASEE/IEEE Frontiers in Education 35th Annual Conference*, October 2005. Conference Proceedings (Conference CD).

[35] K. Sung, G. Rogers, and W. Kubitz. A critical evaluation of pex. *IEEE Computer Graphics and Applications*, 10(6):65-75, November 1990.

[36] K. Sung and P. Shirley. A top-down approach to teaching introductory computer graphics. *ACM SIGGRAPH 2003 Educator's Program*, July 2003. Conference CD/DVD-ROM Disc 1.

[37] K. Sung and P. Shirley. Returning adult students for algorithm analysis. *The Journal of Computing Sciences in Colleges*, 20(2):62-72, December 2004. Proceedings of the Sixth Annual CCSC-NW Conference.

[38] K. Sung and P. Shirley. A top-down approach to teaching introductory computer graphics. *Computer and Graphics*, 28(3):383-391, June 2004. Invited full paper based on SIGGRAPH 2003 conference-paper.

[39] K. Sung, P. Shirley, and S. Baer. Essential Concepts for Building Interactive Graphics Applications. A. K. Peters, Wellesley, Massachusetts, 2007. 400 pages draft manual script developed, on target for publication in 2007.

[40] K. Sung, P. Shirley, and R. Reed-Rosenberg. Experiencing aspects of games programming in an introductory computer graphics class. In *SIGCSE '07: Proceedings*, 2007. to appear.

[41] E. Sweedyk, M. deLaet, M. C. Slattery, and J. Kuffner. Computer games and cs education: why and how. In *SIGCSE '05: Proceedings*, pages 256-257, 2005. ACM Press.

[42] J. Vegso. CRA taulbee trends: Female students & faculty, 2005. *http://www.cra.org/info/taulbee/women.html*.

[43] J. Vegso. Interest in CS as major drops among incoming freshmen. *Computing Research News*, 17(3), May 2005.

[44] J. Vegso. Drop in CS bachelor's degree production. *Computing Research News*, 18(2), March 2006.

[45] GarageGames Torque X, *http://www.garagegames.com/products/torque/x*, 2006.

[46] Angelo, Thomas A. and K. Patricia Cross, ''*Classroom Assessment Techniques: A Handbook for College Teachers*,'' San Francisco: Jossey-Bass, 1993.

[47] Banta, Trudy W., ed. ''*Building a Scholarship of Assessment*,'' San Francisco: Jossey-Bass, 2002.

[48] Mentkowski, Marcia, ''*Learning That Lasts: Integrating Learning, Development, and Performance in College and Beyond*,'' San Francisco: Jossey-Bass, 2000.

[49] Nichols, James O. and Karen W. Nichols, ''*The Departmental Guide and Record Book for Student Outcomes Assessment and Institutional Effectiveness*,'' 3rd edition. New York: Agathon Press, 2000.

[50] Joy Frechtling, and Laure Sharp, ed. ‘‘*User-Friendly Handbook for Mixed Method Evaluations*,’’ Directorate for Education and Human Resources, Division of Research, Evaluation and Communication, National Science Foundation, August 1997.

[51] K. Sung, A. Pearce, and C. Wang `"Spatial-Temporal Antialiasing," *IEEE Transactions on Visualization and Computer Graphics, PP. 144-153, Vol. 8, No.2*, April-June 2002.

[52] A. Pearce, and K. Sung, "Analytic motion blur coverage in the generation of computer graphics imagery," *US Patent Number: 6,211,882*, April 3, 2001. *The implementation of this patent can be found in all Alias|Wavefront image generation systems. Images generated based on this patent can be found in movies like Independence Day, or Wing Commander, etc.*