# Design and Implementation of Project-Based Courses on Cutting-Edge Computer Technologies

**Dr. Wenbing Zhao, Cleveland State University**

Dr. Zhao is a Full Professor at the Department of Electrical Engineering and Computer Science, Cleveland State University (CSU). He earned his Ph.D. at University of California, Santa Barbara in 2002. Dr. Zhao has a Bachelor of Science degree in Physics in 1990, and a Master of Science degree in Physics in 1993, both at Peking University, Beijing, China. Dr. Zhao also received a Master of Science degree in Electrical and Computer Engineering in 1998 at University of California, Santa Barbara. Dr. Zhao joined CSU faculty in 2004. He is currently serving as the director of the Master of Science in Electrical Engineering, and the Chair of the Graduate Program Committee in the Department of EECS, the ABET coordinator for the BS in Computer Science Program, and a member of the faculty senate at CSU. Dr. Zhao has authored a research monograph titled: "Building Dependable Distributed Systems" published by Scrivener Publishing, an imprint of John Wiley and Sons. Furthermore, Dr. Zhao published over 150 peer-reviewed papers on fault tolerant and dependable systems (three of them won the best paper award), computer vision and motion analysis, physics, and education. Dr. Zhao's research is supported in part by the US National Science Foundation, the US Department of Transportation, Ohio State Bureau of Workers' Compensation, and by Cleveland State University. Dr. Zhao has served on the organizing committee and the technical program committee for numerous international conferences. Dr. Zhao is an Associate Editor for IEEE Access, an Academic Editor for PeerJ Computer Science, and is a member of the editorial board for International Journal of Parallel Emergent and Distributed Systems, International Journal of Distributed Systems and Technologies, International Journal of Performability Engineering, International Journal of Handheld Computing Research. Dr. Zhao is a senior member of IEEE.

**Prof. Xiong Luo, University of Science and Technology, Beijing and Beijing Key Laboratory of Knowledge Engineering for Materials Science**

Xiong Luo received the Ph.D. degree from Central South University, China, in 2004. He currently works as a Professor in the School of Computer and Communication Engineering, University of Science and Technology Beijing, China. His current research interests include machine learning, cloud computing, and computational intelligence. He has published extensively in his areas of interest in journals, such as the Future Generation Computer Systems, Computer Networks, IEEE Access, and Personal and Ubiquitous Computing.

**Prof. Chaomin Luo, University of Detroit Mercy**

Dr. Chaomin Luo received his Ph.D. in Department of Electrical and Computer Engineering at University of Waterloo, Canada in 2008, where he was awarded Postgraduate Scholarship (PGS) from the Natural Sciences and Engineering Research Council (NSERC) of Canada; received the Best Student Paper Presentation Award at the SWORD'2007 Conference, earned his M.Sc. in Engineering Systems and Computing at University of Guelph, Canada, and his B.Eng. degree in Radio Engineering from Southeast University, China. He is currently an Associate Professor, Department of Electrical and Computer Engineering, at University of Detroit Mercy, Michigan, USA. He was awarded Faculty Research Awards in 2009, 2010, 2014, 2015, and 2016 at University of Detroit Mercy, Michigan, USA. His research interests include engineering education, robotics and automation, control, autonomous systems, computational intelligence and machine learning.

Dr. Luo was the General Co-Chair of the 1st IEEE International Workshop on Computational Intelligence in Smart Technologies (IEEE-CIST 2015), and Journal Special Issues Chair, IEEE 2016 International Conference on Smart Technologies (IEEE-SmarTech), USA. He was the Publicity Chair in the 2011 IEEE International Conference on Automation and Logistics. He was on the Conference Committee in the 2012 International Conference on Information and Automation and International Symposium on Biomedical Engineering and also the Publicity Chair in the 2012 IEEE International Conference on Automation and Logistics. Also, he was Chair and Vice Chair of IEEE SEM - Computational Intelligence Chapter and is currently a Chair of IEEE SEM - Computational Intelligence Chapter and Chair of Education Committee of IEEE SEM.

Dr. Luo serves as the Editorial Board Member of International Journal of Complex Systems – Computing, Sensing and Control; Associate Editor of International journal of Robotics and Automation (IJRA); and Associate Editor of International Journal of Swarm Intelligence Research (IJSIR). He has organized and chaired several special sessions on topics of Intelligent Vehicle Systems and Bio-inspired Intelligence in IEEE reputed international conferences such as IEEE-IJCNN, IEEE-SSCI, etc. He was the Panelist in the Department of Defense, USA, 2015-2016, 2016-2017 NDSEG Fellowship program, and National Science Foundation, USA, GRFP program, 2016-2017.

**Prof. Yonghong Peng, University of Sunderland, St. Peters Campus**

Professor Yonghong Peng is a Professor of Data Science, and the Leader of Data Science Research at the University of Sunderland. His research include Data Science Foundation and Big Data Innovation including Education Data Analytics, Smart Cities, Business Intelligence, Intelligent Manufacturing and Industry 4.0, Creative industries, Sports, Social Sciences etc. His team developed innovative integrative analytics approaches that have been successfully applied for the discovery of key signatures of human cancers e.g. melanoma and triple-negative breast cancer (TNBC). He is currently acting the Chair for Big Data Task Force (BDTF) of IEEE Computational Intelligence Society (IEEE CIS) and is a member of Data Mining and Big Data Analytics Technical Committee of IEEE CIS. Prof Peng is an associate editor for IEEE Transaction on Big Data, a member of the editorial board of the International Journal of Big Data Intelligence, and an academic editor of the PeerJ and PeerJ Computer Science. Prof Peng has served on the organising committee and the technical program committee for numerous international conferences.

# Design and Implementation of Project-Based Courses on Cutting-Edge Computer Technologies

## 1. Introduction

In this paper, we describe the design and implementation of two project-based courses, one on Apple iOS application development, and the other on Microsoft Kinect [1] application development, and report the lessons learned in teaching these non-traditional courses. This type of non-traditional courses on cutting edge computer technologies was pioneered by Stanford University, where they created the first ever iPhone application development course (as CS 193P) during the 2009-2010 academic year with a huge success [2]. Our two courses were offered as technical elective courses. The two courses are the only two project-based courses on cutting-edge computer technologies in our curriculum. These courses provided students with the opportunities to learn and practice real-world software engineering, and gain experiences in solving multidisciplinary practical problems. Furthermore, these courses help students to attain several ABET student outcomes that are difficult to accomplish via traditional lecture-based and lab-based courses, such as (f) an understanding of professional and ethical responsibility, (i) a recognition of the need for, and an ability to engage in life-long learning, and (j) a knowledge of contemporary issues.

These courses differ from traditional Electrical and Computer Engineering courses both in the content covered and in the way that they were taught. The iOS course covers topics such as object-oriented programming, the Objective-C programming language, various application programing interfaces (APIs) for graphical user interface design, touch-based human-computer interaction, inertial sensors, and computer networking. The Kinect course covers fundamental computer vision technologies that made Kinect possible to perform human motion tracking, the C# programming language, the rich APIs provided by the Kinect Software Development Kit, the Unity 3D game development and visualization platform [3], and computer vision programming with OpenCV, which empowers students to extend the current Kinect APIs for tasks such as object detection. Furthermore, both courses heavily depend on student active learning with lab exercises and team-based projects. However, this does not mean that we completely abandon traditional instruction in classrooms. We believe that it is necessary and beneficial to students for instructors to dissect hard-to-understand subjects and elaborate their insight to the relationship between different concepts and language constructs and how to effectively solve problems. Hence, we do incorporate lecture-based instructions. What is different from traditional courses is that the lecture-based instruction is limited to 30-60 minutes in each 1-hour-and-50-minute session. The remaining time is given to students to explore and learn on their own under the guidance of the instructor.

## 2. Course Design

For both courses, in addition to covering technical content, we arranged at least one guest speaker session in each course. The speakers were invited from the industry who are seasoned software developers for the respective technologies. For the Kinect application development course, we invited an additional guest speaker to talk about career development and entrepreneurship. Our intention was to inspire our students to not only be a life-long learner, but also become an entrepreneur to create something useful for the humanity using their technical knowledge and skills.

We first created the iOS application development course in fall 2010, soon after Stanford University offered theirs. Our course covers the following topics:

- iPhone application development environment (XCode)
- The Objective-C programming language
- The Model-View-Control design pattern
- Basic user interface
- Autorotation and autosizing
- Multiview applications
- Tab bars and pickers
- Categories and table views
- Navigation controllers
- Data persistence
- Networking with Bonjour
- How to program with GameKit
- Taps, touches, and gestures
- Location and MapKit
- Accelerometer
- How to program with core graphics and quartz 2D
- Unit testing
- Consuming Web services
- Access build-in applications
- Concurrency and multithreading

The iOS course was designed only for computer engineering majors. Hence, at the beginning of the semester, a screening test was administered and students who failed the test were advised to withdraw from the course.

We first designed the Kinect application development course in spring 2014. Unlike the iOS course, we intentionally made the course open to all electrical and computer engineering students who are interested in taking it regardless of their background. This course covers the following topics:

- Kinect application development environment (Visual Studio)
- The C# programming language
- Kinect color image stream
- Kinect depth image stream
- Kinect skeleton tracking
- Kinect microphone array
- Speech recognition with Kinect
- Kinect 3D game development with Unity
- Gesture recognition
- OpenCV with Kinect

In the Kinect course, to accommodate the diverse background of the students, we designed two sets of tasks for each session. The basic set was catered to those who do not have extensive programming background, while the advanced set was designed to challenge those who do. As an example, one of the lab exercises in the Kinect application programming course required students to build a hand-tracking Kinect application where the right hand of the user is tracked and displayed on the screen with a circle shape. The advanced tasks for this lab would make the hand tracking application a drawing application, as specified below:

- Task#1: Display the traces of the hand movement. This task would require students to explore APIs that enable the display of hand traces.
- Task#2: Add a reset button to clear existing traces so that a new drawing can be made. In addition to learning the API to clean up the traces, this task would require students to develop a scheme that the button can be pushed without disrupting the current drawing, for example, to use another hand or to use voice command.
- Task#3: Add a palette chooser to enable drawing using different colors. Note that the color of existing traces should not be changed while drawing color is changed, i.e., only new traces will use the newly set color from the chooser. This is the most challenging task because it involves the construction of a group of graphical controls representing different colors, in addition to the mechanism developed in the previous tasks.

## 3. Teaching Methodologies

The nature of project-based courses calls for a hybrid teaching methodology that involves short lectures, active learning, lab sessions, and team-based projects. Typically, each class starts with a short lecture, which is followed by a set of lab exercises. The lab exercises were designed for student to learn how to build simple applications using the APIs covered in the lecture.

The flipped classroom active learning methodology [4,5] was applied throughout these courses because it fits very well with the project-oriented nature of the courses. Specifically, for the advanced tasks in the lab exercises and for the term projects, students were expected to search for and learn additional APIs and software libraries that are not covered by the short lectures.

Of course, employing flipped classroom active learning doesn't mean that we totally leave the students alone. During the lab exercises, we proactively checked the progress of students and discussed with students when they had doubts on how to proceed. This method facilitated us to engage and to work with individual students in every session throughout the entire semester. Hence, it created much stronger instructor-student interaction than traditional courses.

## 4. Course Outcome and Evaluation

The iOS course was not evaluated formally using surveys. The outcome of the course was evaluated based on the student performance on their term projects. Of the 19 students who enrolled in the course, only 1 student failed to complete the term project due to medical reasons. All other 18 students successfully finished their projects, which must be a working iOS application. Below is a list of iOS applications developed by students:

- *CityFit*: an application designed for people to sense locations in their environment where they can help achieve their fitness goals both physically and mentally. CityFit is the most outstanding iOS app produced by the students in the class. The CityFit app's screenshots are shown in Figure 1. As a side note, soon after taking the course, the student who developed the CityFit app was hired by the biggest software company in town to develop mobile app professionally.


**Figure 1. Screenshots of the CityFit iOS app.**

- *RecipeBrowser*: an application that can store and search recipes for one's favorable meals.
- *Quest*: a game based on OpenGL.
- *XYZLogger*: an application designed for people who prefer listening to music while working out. The application can sense the movement of the person who are holdig the phone and find a song in the database that matches the rhythm of the workout.
- *AlcoaApp*: an application that provides a mobile interface to inventories for the student's workplace.
- *WeatherStation*: an application that allows the user to check the current weather conditions.
- *Mingling*: an application that mingles social networks with geo-locations.
- *iCallForward*: an application that enables call forwarding for Vonage VoIP users.
- *iFlashCards*: an application that allows the users to create, view, and share electronic flash cards.
- *iControl*: an application that enables users to control their home appliances and multimedia facilitates using their iPhone.
- *iPassport*: an application designed to make travel easier. The application includes a currency converter, a language translator, a flight status and itinerary checker, and performs auto-messaging updates depending on the user's location.
- *OnRoadAlert*: an application that senses the driving behavior of the user and alerts the user if the car deviates from the current lane.
- *HomeCal*: a custom calendar application for iPhone.

- *TheWord*: a Bible application for iPhone.
  *CityTourGuide*: an application that allow users to embark upon a self-guided tour of the city where our institution is located. It will alert users when points of interests are in close proximity to their current location and provides a brief history of all points of interests via a text description and an audio file.

The Kinect course was evaluated via both project-based objective assessment, and survey-based assessment. The survey was administered at the end of the course. The Kinect course attracted a lot more students with 72 students enrolled because we intentionally made the course available to all students without screening on their programming skills. As such, students were divided into 2-3 person teams to work on their projects. Students completed a total of 25 projects. The following is a list of outstanding projects:

- *Smart Presenter*: This application was designed to make PowerPoint presentations using hand gestures. Slide transition can be controlled also by voice commands. This is the best Kinect application developed in the class. The screenshots for the application are shown in Figure 2.

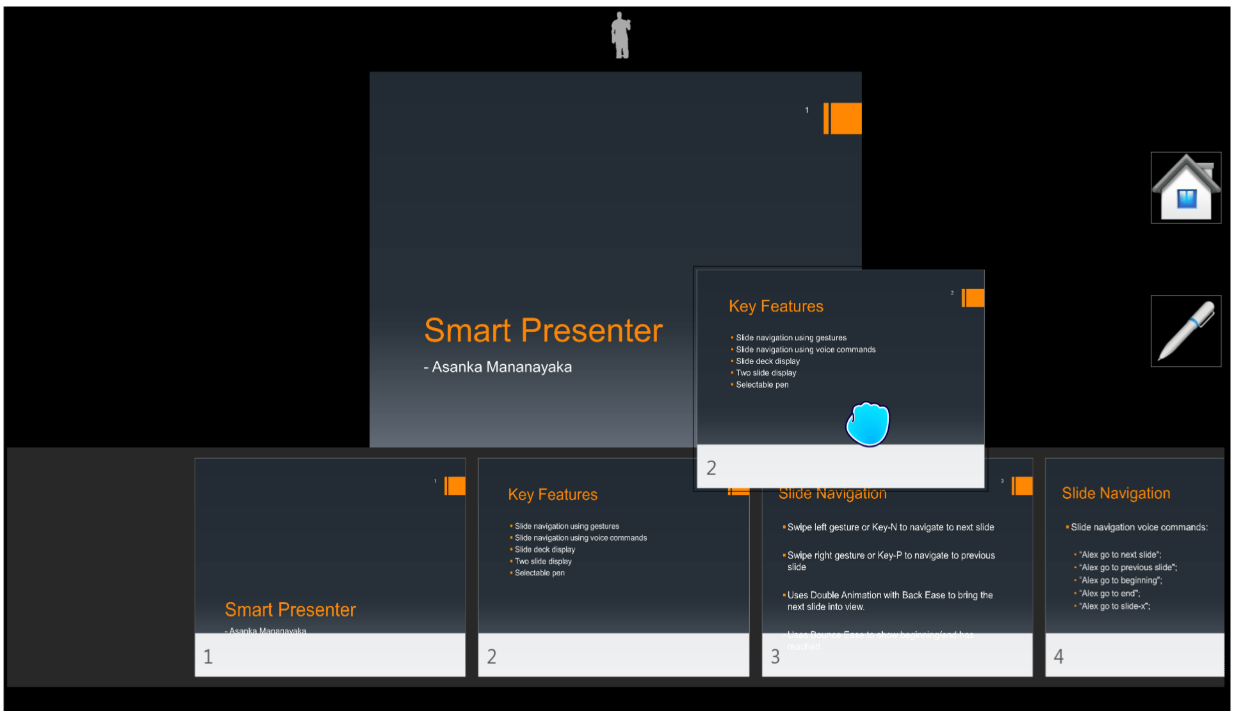


**Figure 2. A screenshot for the Smart Presenter Kinect application.**

- *Presentation Kinesis*: This is also a presentation application that allows a user to control PowerPoint slides and to make annotations using gestures.
- *Fun Learning Games*: This application consists of three games designed for preschool or kindergarten children to learn shapes, spelling, and simple math using hand gestures.
- *Match Me*: This is also a Kinect game designed for young kids. The user is expected to find a shape that matches what is displayed from a pile of shapes using his/her hand.

- *A Kinect Painting Application*: This is a hand painting application. A user can change colors, stroke sizes, etc.
- *A Virtual Sword Combat Game*: A Unity 3D game that allows two players to play a virtual sword contest using Kinect.
- *A Virtual Boxing Game*: This is another Unity 3D game that allows two players to play a virtual boxing game using the Kinect sensor.
- *A Kinect Map Application*: A Kinect application that allows a user to traverse the Google Map via hand gestures.
- *A Family Media Player*: This Kinect application presents a media-center-like interface for a user to choose media to play, and to pause, stop, or play the selected media file using hand gestures.
- *Kinect Softball Pitch Analyzer*: This Kinect application was designed to capture the pitching motion of a baseball player and provide real-time feedback regarding the quality of the pitching based on predefined rules.

Because students were asked to work in a team, additional measure was necessary to ensure that all students in the team were contributing to the project. We decided to adopt the individual evaluation rubric form [6] for the Engineering Projects in Community Service (EPICS) at Purdue [7]. Since the starting of the term project, every member in a team was asked to complete and turn in a self-evaluation report based on the individual evaluation rubric regarding the his/her role in the team and his/her contributions made to the project every other week.

For the Kinect course, we conducted an informal survey regarding their previous programming experiences. Only 20 students out of 72 had extensive programming experiences and virtually none had programmed with the C# programming language before. At the end of the course, 25 out of 30 teams successfully completed their projects and demonstrated a quality Kinect application. This shows that 83% of students had meet the objective of this course.

Additionally, we designed and administered a formal survey toward the end of the course. We received 68 valid surveys. The survey solicits input from students regarding the following three items:

- Item 1: How various subjects we taught in the class have been utilized successfully in students' term projects. For each subject covered, three options were given: attempted successfully; attempted with some success; never attempted.
- Item 2: Student opinion regarding the course design and implementation, including the value of learning cutting-edge technology, whether or not this course has enhanced their programming skills, and the benefit of the guest seminar.
- Item 3: The overall rating of this course.

The survey result on item 1 is illustrated in Figure 3. Based on the survey result, we computed the success rate for each major subject covered in the course, and the utilization rate of the subject in student's projects. The success rate for each subject is determined as the success count (i.e., the count for "attempted successfully") divided by the total number of attempted count. The subject utilization rate is computed as the number of attempted count (the total of the count for the "attempted successfully" and "attempted with some success") divided by the total number of students in the class. For several advanced subjects, we consider a low utilization rate normal.

We are more concerned about the success rate because a low successful rate means that we should spend more time and effort to cover the subject in future offering of the course.

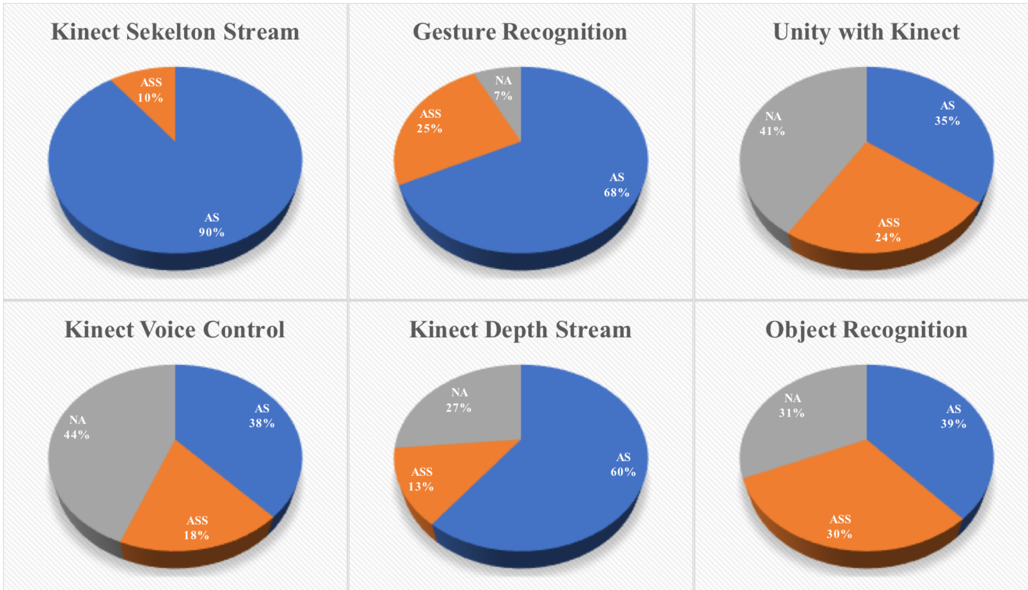


**Figure 3. Survey result for item 1 on the subjects covered in the Kinect course. Here AS stands for Attempted with Success, ASS stands for Attempted with Some Success, and NA stands for Never Attempted.**

The survey result for the success rate and utilization rate for subjects covered in this course are shown in Figure 4. The advanced subjects such as the integration of Unity with Kinect, and object recognition using OpenCV, have less utilization rates. Furthermore, for those who have attempted to incorporate these harder subjects in their projects, the success rates were less than that of other subjects.

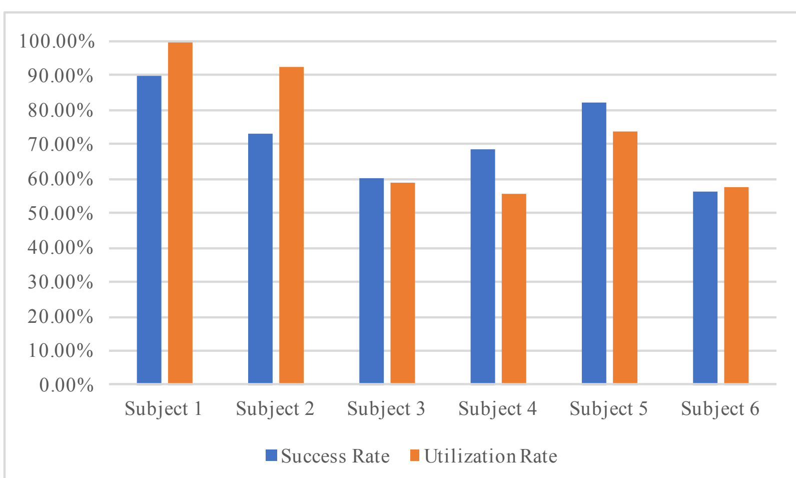


**Figure 4. Success rate and the utilization rate for item 1. Here subject 1 through 6 refer to Kinect skeleton stream, gesture recognition, Unity, voice control, Kinect depth stream, and object recognition.**

For item 2, we included the following questions in our survey:

- Q1. This course makes the degree program more attractive to me.
- Q2. I had a more engaged learning experience in this course than other traditional lecture-based courses.
- Q3. I have improved my programming skills in this course significantly.
- Q4. I became more interested in learning computer programming.
- Q5. I enjoyed the guest seminar on entrepreneurship and career development.
- Q6. I think it is a good idea to add a module on entrepreneurship & career development in our curriculum.

For each question, five choices were provided: strongly agreed (SA), agreed (A), neutral (N), disagreed (D), and strongly disagreed (SD). To compute the overall score for each question, SA is assigned to have 5 points, A is assigned to have 4 points, N is assigned to have 3 points, D is assigned to have 2 points, and SD is assigned to have 1 point.

For item 3, only a single question is asked: how will you rate this course overall? Four choices were provided for this question: excellent (4 points), good (3 points), fair (2 points), and poor (1 point). Figure 5 illustrates the survey result for item 2 and item 3.

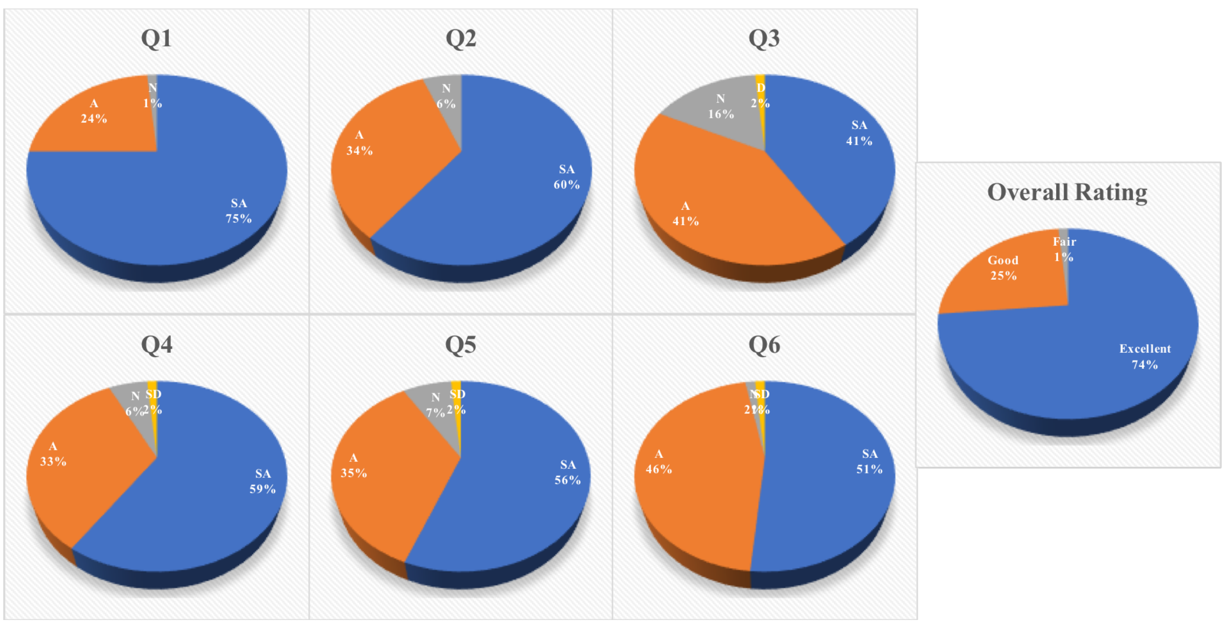


**Figure 5. Survey result for item 2 and item 3.**

The average scores for the six questions in item 2 are shown in Figure 6. The average scores range from 4.2 (for Q3) to 4.7 (Q1) (maximum score is 5.0). The average score for the overall ranking is 3.72 (maximum score is 4.0). The item that received the lowest rating is question 2 regarding if a student has "significantly" improved his/her programming skills. We speculate that this may be due to our use of the word "significantly." It is difficult to determine how much improvement is "significant" improvement. Nevertheless, as shown in Figure 5, 41% of students strongly agreed that they have significantly improved their programming skills and another 41% of students agreed that this is the case. In particular, it is very encouraging to see that 51 students out of 68 (i.e., 75% of the students) who completed the survey strongly agreed that this course made the degree program more attractive.

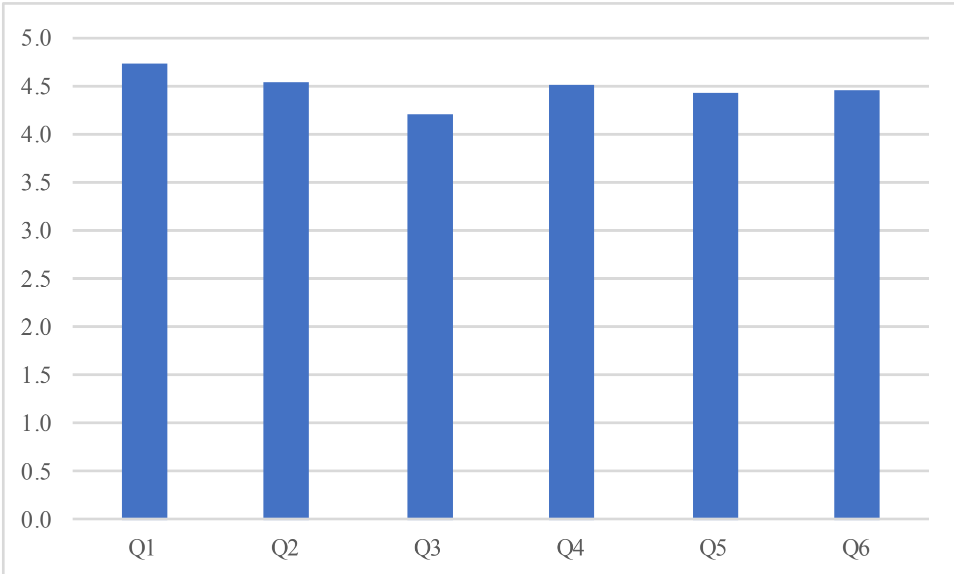


**Figure 6. Average scores for the six questions in item 2.**

## 5. Discussion

For the Kinect course, we intentionally made it available for all electrical and computer engineering students to take it. It is our belief that not only for computer engineering students, electrical engineering students should also be exposed to substantial computer programming training for them to be competitive in the current job market. Our current curriculum for electrical engineering students severely lacks the computer programing component. Our survey result indeed showed that our effort was very well received by students.

As a tradeoff, we had to face the challenge of teaching a class with drastically different preparation background. In the first few lectures, we used a tutorial style to teach students how to build simple applications using the APIs we covered. We provided computer programming code and a step-by-step guide on how to build each application in a set of basic tasks. We did so to ensure all students could complete the tasks and follow the instruction.

Not surprisingly, we observed that a fraction of students could finish the tasks far ahead of others. To accommodate this group of very well prepared students, we started to add a set of advanced tasks in each session, which we call challenge tasks. For these challenge tasks, we provided only guidelines and students would have to search the Web for relevant APIs and to come up with their solutions on how to use these APIs to accomplish these tasks.

Another challenge we faced in teaching project-based courses is that it is difficult to track student progress on their projects. For the Kinect course, we asked students to complete a self-evaluation report every two weeks. All teams reported that they were making good progress on their projects. However, at the end of the semester, five teams failed to complete their projects. Apparently, self-reporting was not reliable for all students. In the future, this issue could be addressed by using public source code control Websites, such as github.com, which could help instructors to track students' actual progress.

Finally, we should note that teaching cutting-edge technologies is demanding both in terms of hardware cost and in terms of the demand on the instructor's time and skills. For the Kinect course, we have purchased over 30 Kinect sensors for students thanks to the support of our institution. For the iOS course, our institution invested more than $30,000 on Apple computers and iPod touch devices. It would have been impossible for us to offer these courses without the such institutional support. Furthermore, the APIs for cutting-edge computer technologies are rapidly evolving. Hence, this requires the instructors not only to learn the APIs, but to update their course material every year to keep up with the changes. Unless such technologies align very well with the instructors' own research, which is indeed the case for us, doing so might not be feasible.

**References**


[1] Lun, R., & Zhao, W. (2015). A survey of applications and human motion recognition with microsoft kinect. *International Journal of Pattern Recognition and Artificial Intelligence*, 29(05), 1555008.

[2] "Welcome to CS193P iPhone Application Development." http://web.stanford.edu/class/cs193p/cgi-bin/drupal/system/files/lectures/01_Introduction.pdf. Accessed: November 6, 2014.

[3] "Unity." http://unity3d.com/. Accessed: February 6, 2017.

[4] Bonwell, C.; Eison, J. 1991. Active Learning: Creating Excitement in the Classroom. *AEHE-ERIC Higher Education Report* No. 1. Washington, D.C.: Jossey-Bass. ISBN 1-878380-08-7.

[5] Bergmann, Jonathan and Sams, Aaron. 2012. *Flip Your Classroom: Reach Every Student in Every Class Every Day*. International Society for Technology in Education.

[6] "Individual Evaluation Rubric." https://sharepoint.ecn.purdue.edu/epics/teams/Public%20Documents/EPICS_Individual_Evaluation_Rubric.docx. Accessed: November 6, 2014.

[7] Coyle, E. J., Jamieson, L. H., & Oakes, W. C. 2005. "EPICS: Engineering projects in community service." *International Journal of Engineering Education,* 21(1), pp. 139 – 150.