The Journal of Computing Sciences in Colleges

Papers of the 31st Annual CCSC South Central Conference

April 3, 2020
The University of Texas at Dallas
Richardson, TX

Baochuan Lu, Editor
Southwest Baptist University

Laura Baker, Regional Editor
St. Edward's University

Volume 35, Number 7 **April 2020**

The Journal of Computing Sciences in Colleges (ISSN 1937-4771 print, 1937-4763 digital) is published at least six times per year and constitutes the refereed papers of regional conferences sponsored by the Consortium for Computing Sciences in Colleges.

Table of Contents

The Consortium for Computing Sciences in Colleges Board of Directors

CCSC National Partners

The Consortium is very happy to have the following as National Partners. If you have the opportunity please thank them for their support of computing in teaching institutions. As National Partners they are invited to participate in our regional conferences. Visit with their representatives there.

Platinum Partner

Turingscraft
Google for Education
GitHub
NSF – National Science Foundation

Silver Partners

zyBooks

Bronze Partners

National Center for Women and Information Technology
Teradata
Mercury Learning and Information
Mercy College

Welcome to the 2020 CCSC South Central Conference

On behalf of the 2020 South Central Steering Committee we extend our warmest welcome to the region's 31st annual conference hosted by the University of Texas at Dallas in Richardson, Texas. Our conference chair and host, Shyam Karrah, has prepared another wonderful venue for the conference with support from his department and staff at UT Dallas. The generous time and effort required to host the conference is very much appreciated and recognized by the Steering Committee and the Consortium.

For this year's program we have a full schedule of 8 paper presentations, 3 tutorials, nifty assignments, faculty posters, and a student poster competition for all participants to enjoy. All papers undergo a double-blind review process and 8 of 15 paper submissions were accepted for presentation and publication in the *Journal of Computing Sciences in Colleges* for an acceptance rate of 53%. We had 21 paper reviewers from across the country participate in our paper reviewing and we are especially grateful to all of them for their expertise and comments used in our paper and program selection process.

Each year the Steering Committee extends an invitation to colleagues to host the conference in the future and to join our collegium of computer science educators to provide new input and suggestions for improvement to our conference. We strongly encourage our members of the South Central region to attend the Friday evening business meeting and to join in our efforts to bring in fellow colleagues who wish to contribute to the preparation and planning for the conference next year and beyond.

We extend congratulations to all of our presenters for their accepted work and to all of our student poster presenters for their efforts preparing posters for the competition. Warmest thanks to all members of the 2020 Steering Committee who continue to provide the much needed time and effort to the conference administration with continued grace and humility. Welcome to the University of Texas at Dallas and the 2020 South Central Region Conference we hope everyone enjoys the program.

Shyam Karrah
University of Texas at Dallas
Conference Chair and Host

Laura J. Baker
St. Edward's University
Papers and Program Chair

2020 CCSC South Central Conference Steering Committee

Shyam Karrah, Host and Chair .... The University of Texas at Dallas
Anne Marie Eubanks, Registrar .... Stephen F. Austin State University
Tina Johnson, National Board Representative .. Midwestern State University
Bilal Shebaro, Treasurer .... St. Edward's University
Laura Baker, Papers/Program Chair, ... St. Edward's University
Philip Schielke, Panels and Tutorials Chair, ... Concordia University
Michael Scherger, Posters Chair, ... Texas Christian University
Bing Yang Wei, Moderators Chair, ... Texas Christian University
Tim McGuire , At-Large Member, ... Texas AM University
Eduardo Colmenares-Diaz, Publicity Chair, ... Midwestern State University
Michael Kart, Nifty Assignments Chair, ... St. Edward's University
Abena Primo, Webmaster, ... Huston-Tillotson University
Shyam Karrah, Past Conference Chair .... The University of Texas at Dallas

Reviewers — 2020 CCSC South Central Conference

Srinivasarao Krishnaprasad . . . Jacksonville State University, Jacksonville, AL
Anne Marie Eubanks . Stephen F. Austin State University, Nacogdoches, TX
David Gurney . . . . Southeastern Louisiana University, Hammond, LA
Barbara Anthony . . . . Southwestern University, Georgetown, TX
Kenneth Rouse . . . . LeTourneau University, Longview, TX
Michael Kart . . . . St. Edward's University, Austin, TX
Jose Metrolho . . . . Polytechnic Institute of Castelo Branco, Portugal
Vipin Menon McNeese . . . . State University, Lake Charles, LA
Michael Scherger . . . . Texas Christian University, Fort Worth, TX
Brent Baas . . . . LeTourneau University, Longview, TX
Hongzhi Song . . . . . Texas AM University-Corpus Christi, Corpus Christi, TX
Philip Schielke . . . . Concordia University Texas, Austin, TX
Bilal Shebaro . . . . St. Edward's University, Austin, TX
Jeong Yang . . . . Texas AM University-San Antonio, San Antonio, TX
Abena Primo . . . . Huston-Tillotson University, Austin, TX
Tina Johnson . . . . Midwestern State University, Wichita Falls, TX
Eduardo Colmenares . . . . Midwestern State University, Wichita Falls, TX
Catherine Stringfellow . . . . . . Midwestern State University, Wichita Falls, TX
Tim McGuire . . . . Sam Houston State University, Huntsville, TX

Teaching Accessibility in a CS0 Class*

Paula Gabbert
Department of Computer Science
Furman University
Greenville, SC 29613
paula.gabbert@furman.edu

Abstract

This paper describes a CS0 course taught at Furman University for both majors and non-majors in the Spring of 2018 and Spring of 2019. The course introduces students to the discipline of Computer Science using accessible technology as an application area. This area has a rich domain of interesting examples on which to base discussions of core computing topics for new majors who need a basic understanding of accessibility issues. It is also an interesting interdisciplinary area that appeals to many different majors. It is important for both majors who will be creating new technologies as well as non-majors who will be creating digital content using technologies to understand the issues surrounding disability and accessible technology. The paper will describe key computing concepts and how they were presented in the context of accessibility. Student comments in end-of-course reflections highlight how students plan to apply what they have learned in future pursuits.

1 Introduction

There has been growing recognition of the need for including accessibility issues within the computer science curriculum. These initiatives have been coordinated since 2015 by Teach Access [1], a forum for academia and industry to sponsor programs, workshops, and forums to share ideas and curricular initiatives for integrating accessibility education into the CS curriculum. In 2016 Ladner connected the need for accessibility within the CS curriculum to the

CSForAll initiatives [11], including people with disability as another underrepresented group that needs intentional focus. In addition, Lewthwaite and Sloan advocated for pedagogical explorations of current CS classes as a means for growing the body of research on accessibility and strengthening the CS curriculum [12]. These efforts highlight the importance of addressing both the need for access to CS education by students with disabilities as well as the need for training future technology developers to ensure that new technology is accessible to all users. This paper will focus on the later issue proposing a curricular initiative for introducing accessibility concepts within the first introductory course in the major. There are many advantages to introducing accessibility issues in the first course rather than waiting until later in the curriculum. First, universal design principles dictate that technology be developed for all from the beginning and hence, introducing accessibility into the curriculum from the beginning supports the universal design principle within the curriculum. Also, accessibility issues intersect with many of the core CS concepts from data representation, architecture, operating systems, and networks so it is beneficial to introduce these intersections when introducing the concepts, especially since many of the upper level courses in these core areas may not yet address accessibility issues. Finally, introducing accessibility issues in the first course provides a broader context for specific accessibility training in later courses. Understanding the social, historical, and legal landscapes for persons with disability creates better empathy and sensitivity for developing appropriate applications in later courses.

2 Evolution of Accessibility in Furman's CS0 Class

There are many models for CS0 courses [5, 14, 10], and they all introduce the discipline of computer science through an exploration of broad topics, rather than a singular focus on programming. Furman University revamped its introductory CS course in 2013 by offering theme-based CS0 courses that teach the core computer science topics through a specific theme or application [16]. Various themes have been used in the course since then including bioinformatics, social media, multimedia, cryptography, and games. Each course meets for the traditional 150 minutes per week and also meets an additional 120 minutes per week in a lab setting.

In Spring 2018, the course was offered with a new theme of accessible technology. Most accessibility education within the CS curriculum focuses on mid to upper level courses, most notably human computer interaction [13] and web design [17]. The theme-based CS0 course provided the opportunity to explore how accessibility issues could be introduced much earlier in the curriculum. While there is an expected list of core CS topics to cover in the CS0 class, there is ample time to explore the application or theme area as well. The lab time also provides opportunity to experience accessible technology through hands-on activities. This paper will outline the goals of the course, the content used to meet these goals as well as evidence for meeting some of the goals from the reflective essays written by students at the end of the course.

3 Overview of Topics

The CS0 course at Furman University serves a dual purpose as the first course for the Computer Science or Information Technology major as well as a service course for non-majors satisfying the math and formal reasoning general education requirement. As such, the course attracts students from a variety of majors as well as students who have not yet decided on a major. This heterogeneous population of students offers opportunity for rich discussions from multiple perspectives. The course offers an exploration of analytical and algorithmic problem solving to satisfy the general education requirement as well as an overview of core computer science topics that serve as an introduction to the discipline of computer science. Many general purpose textbooks are available for covering the core computer science material. The Brookshear, Brylow textbook [4] was used in this CS0 course on accessible technology. This paper will focus only on the aspects of the course that intersect with accessible technology and that are not highlighted in typical introductory texts.

A primary objective for this offering of the CS0 course is to motivate the need for universal design and its importance for creating technology that is initially designed for use by all from the very beginning of the curriculum. Hence, it is important for computer science professionals to have a basic understanding of the legal, political, and financial issues surrounding accessible technology and to develop empathy for persons with disabilities. These social issues are integrated throughout the course while students explore the core computer science topics. The interdisciplinary nature of the material lends itself well to a heterogenous class of mixed majors, offering opportunity for lively class discussions and the ability for students from other majors to connect the topics of accessibility to their discipline.

The course explores the social issues of accessibility for technology starting with a global perspective by introducing the United Nations Convention on the Rights of Persons with Disabilities (UNCRPD) [2] and the World Health Organization (WHO) report on disability [15]. These reports and supporting materials found at these sites introduce the social model of disability into the course which states that limitations are not caused by medical issues but by society's inability to provide appropriate technology and resources. They also lay the groundwork for the concept of universal design by advocating for accessibility as a human and civil right. Other supporting materials for the UNCRPD and WHO report (i.e. [7]) introduce the broad global political and legal issues surrounding the human and civil rights of persons with disabilities.

The global perspective provided by the UNCRPD and the WHO reports is then narrowed down to a national perspective within the United States through an exploration of Section 504 of the Rehabilitation Act of 1973 and the Americans with Disability Act (ADA) of 1990. The Director of Furman University's Student Office for Accessibility Resources (SOAR) provided a guest lecture on this national perspective and also narrowed further to a local perspective by providing specific accessibility issues experienced at Furman. The local perspective was further developed by Furman University's students with disability (student with visual impairment and student with autism) willing to visit the class and describe their experiences with their disability and specifically their experiences with technology.

The global, national, and local perspectives on disability provide the students in the class the context for further exploration into the specific application of these ideas to technology and digital information. A collection of papers edited by Lazar and Stein [9] was a second text required for the course and provided material for class discussions, assignments, and student presentations which explored accessibility issues from various political and social perspectives. These readings allowed students to think critically about the essential questions computer scientists should be considering when developing technology:

- How can we provide access to digital information to all people at the same time and at the same cost?
- Who should pay for the cost of developing accessible technology/information?
- How does technology aide people with disability?
- How does technology exclude people with disability?

The core computer science topics covered in all of the CS0 courses at Furman University include: data storage, architecture, operating systems, networking and WWW, algorithms and programming, and software development. Additional topics may also be included as appropriate depending on the course theme. For the accessibility theme additional topics in databases, artificial intelligence, and graphics were explored. Each CS0 section covers the important concepts in each of these areas through the lens of their specific theme. The following sections discuss the relevant connections of each of these core areas to the theme of accessible technology.

3.1 Data Storage

All the CS0 sections cover the binary representation of data and discuss the process of creating digital files for text, numbers, sound, images and graphics. This material is ideal for discussing the importance of providing multiple types of data (captions, alternate text) and translating one type of data to another (OCR, speech to text, and text to speech) in order to create accessible digital information. Students gain an appreciation for the challenges surrounding this translation between different types of data while at the same time realizing that many sophisticated tools already exist that could be used to effectively create accessible information.

3.2 Architecture

The von Neumann architecture is discussed in each CS0 class which provides a forum for discussing hardware issues related to different disabilities. The blind student who visited the class demonstrated her braille reader. In addition, other specialized hardware devices were discussed including eye tracking devices, braille keyboards, and modified game controllers.

3.3 Operating Systems

Students in the course gained an appreciation of the importance of their operating system functionality by exploring the accessibility options on MacOS in a lab setting including dictation, VoiceOver, zoom, and flash/speech alert settings. Although they were sighted while doing these activities, they gained insight into how people with various disabilities might approach basic use of a computer and the importance of the operating system in navigating the use of all applications.

3.4 Networking and WWW

This particular topic has numerous resources available due to the direct connection between web development and accessibility. The class explores the World Wide Web Consortium Web Accessibility Initiative (W3C WAI) Web Content Accessibility Guidelines (WCAG) [3] through a hands-on lab activity to develop a website that meets the WCAG. Students also explored other sites to evaluate their accessibility through another lab activity focused on using VoiceOver. Students gained an appreciation for the challenges of creating accessible material on the web and learned how to critically evaluate web content from the perspective of disability.

3.5 Algorithms and Programming

While algorithm development is integral to all the CS0 sections, the amount of time spent on programming activities varies and each section has the flexibility to introduce any language relevant to the theme. In this CS0 on accessible technology, Python was used in a 3 week unit on programming. In addition, two libraries were used to illustrate speech to text (gTTS) and text to speech (speech_recognition) capabilities through simple hands on programming activities. Through the use of these libraries the basic fundamentals of programming are introduced while illustrating how accessibility can be managed programmatically.

3.6 Software Development

The software development life cycle unit is an ideal opportunity to discuss universal design and the importance of incorporating accessibility issues early in the development process. Agile methods that advocate for user involvement can also lead to important discussions about using representatives from different user groups, including users with disabilities. Haben Girma's presentation to the Apple Developers Conference is a wonderful resource for class discussion on this topic [6].

3.7 Databases

Databases are not covered in all CS0 courses but was included in this course focused on accessibility because of the importance of accessibility within database technology and because this subdiscipline of computer science is relatively undeveloped when it comes to accessibility. In addition to the basic concepts of database technology, the course also discusses the many challenges of database access using the University Library as an example. A paper by Jaeger, Wentz, and Bertot in [8] provides background material for this discussion.

3.8 Artificial Intelligence

Students find artificial intelligence (AI) interesting and exciting at all levels of the curriculum. Introducing AI topics in the CS0 class creates excitement about the discipline and offers another opportunity for demonstrating the ability of technology to aide in accessibility. Topics and the relevant disability explored in this unit include: image recognition and natural language processing for visual impairments, robotics for mobility impairments, and chatbots for autism.

3.9 Graphics

The CS0 class on accessibility does not spend significant time on the topic of graphics but does illustrate the capability the rich graphical environment of SecondLife to create an inclusive social environment for people with certain disabilities. Students are led through a hands-on laboratory activity in SecondLife to demonstrate the capabilities of this virtual world and then discuss the possibilities of this environment for various disabilities including hearing disabilities, mobility disabilities, and autism.

4 Results and Responses

The CS0 course incorporating accessibility topics has been offered twice at Furman University in Spring 2018 and Spring of 2019. At the end of the course, students are asked to reflect on their learning and respond to the following prompt:

> *The social definition of disability argues that a person with disability is impaired by society's inability to provide appropriate resources, not by their impairment. Reflect on your current major, potential major, or main interest area and how the topics we have discussed for accessibility and accessible technology can be incorporated by that discipline to benefit society. How can your discipline contribute positively to minimize social impacts of disability and include all persons in the activities, information access, and communications within your discipline?*

Although one purpose of this CS0 course is to provide new CS majors with insights into the issues of accessible technology as illustrated in Section 3, non-majors also benefit from this material in profound ways. This section will provide samples of responses from students from a variety of majors illustrating how students have connected topics from the course into their respective disciplines.[1]

An education major wrote, "This class has taught me to be more aware of helping to provide universal accessibility to the classes I tutor or in the places I volunteer at. I have learned that technology is everywhere and so it is important to make it as accessible as possible. As I go forward in learning more about persons with disability in my students with exceptionalities class next semester, I will bring all the knowledge I have learned this year with me."

A senior English major wrote, "As a reader and writer, I know how important it is to include all voices in the literary world – particularly those voices that have been historically unheard. By fighting for more accessible e-books,

[1] Students granted permission in writing to use quoted material in publications.

keyboards, and other ... technological aids, my field can benefit from hundreds of untold stories."

A freshmen Politics and International Affairs (PIA) major wrote, "This class has opened my eyes to many obstacles in the political system that persons with disabilities face. Persons with disabilities are the largest minority in America and helping them get involved in the political process will help make our democracy stronger." Another freshmen PIA major wrote, "We have created a system of information to keep the people informed of what is happening in the political field. That system does not include those with visual or social disabilities. Germa [sic] pointed out in her presentation at the Apple Conference that we just assume things of people with disabilities. ... We assume a blind woman won't use a camera because it hasn't been made accessible to her. We assume a man with a social disability won't be informed and active in politics because it hasn't been made accessible for him."

5 Future Work

While the qualitative results of these early offerings of CS0 with accessible technology demonstrate that students are making important connections between their potential majors and the topics in accessible technology, additional research is needed to understand the impact of this course on the future educational and research directions of majors in computer science. One of the goals of the early introduction to these concepts for majors is to ensure that future computer scientists understand the breadth of issues in this area. Future research is needed to know if this course is impacting decisions on course selection, project choices and employment opportunities.

6 Conclusion

The accessibility topics described in this paper can easily be incorporated into any introductory computing course where these core concepts are covered. This early coverage of accessibility within the curriculum further supports the important work advocated by Teach Access and CSforAll by introducing the important issues for accessible technology that can be expanded upon later in upper level courses. Student reflections indicate the impact that this material can have on students as they continue to pursue their disciplines with a new appreciation for accessibility.

References

[1] Teach access. Available: http://teachaccess.org. Accessed: 2019-05-21.

[2] United Nations - Disability: Department of Economic and Social Affairs. Available: https://www.un.org/development/desa/disabilities/. Accessed: 2019-05-28.

[3] World wide web consortium: Web accessibility initiative. Available: https://www.w3.org/WAI. Accessed: 2019-05-28.

[4] J.G. Brookshear and D. Brylow. *Computer Science an Overview*. Pearson, New York, NY, 13th edition, 2019.

[5] J.C. Ernest, A.S. Bowser, S. Ghule, S. Sudireddy, J.P. Porter, D. A. Talbert, and M.J. Kosa. Weathering MindStorms with drizzle and DIODE in CS0. In *Proceedings of the 10th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education*, ITiCSE 2005, page 353, June 2005.

[6] H. Girma. Disability and innovation: The universal benefits of accessible design. Talk presented at Worldwide Developers Conference, San Francisco: Apple, 2016. Available: https://www.youtube.com/watch?v=_bC7Mvy7Vn4. Accessed: 2019-05-28.

[7] PBS News Hour. What prevents the U.S. from signing the U.N. disabilities treaty?, March 13, 2014. Available: https://www.pbs.org/newshour/show/prevents-u-s-signing-u-n-disabilities-treaty.

[8] P.T. Jaeger, B. Wentz, and J.C. Bertot. The intersection of human rights, social justice, the Internet, and accessibility in libraries: Access, education, and inclusion. In *Disability, Human Rights, and Information Technology*. University Pennsylvania Press, Philadelphia, PA, 2017.

[9] J.Lazar and A. Stein, editors. *Disability, Human Rights, and Information Technology*. University of Pennsylvania Press, Philadelphia, PA, 2017.

[10] D. Kafura and D. Tatar. Initial experience with a computational thinking course for computer science students. In *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education*, SIGCSE 2011, pages 251–256, March 2011.

[11] R. Lander and M. Israel. Broadening participation 'For All' in 'Computer Science For All'. *Communications of the ACM*, 59(9):26, 2016.

[12] S. Lewthwaite and D. Sloan. Exploring pedagogical culture for accessibility education in computer science. In *Proceedings of the 13th Web for All Conference*, W4A 2016, April 2016. Article No. 3.

[13] S. Ludi, M. Huenerfauth, V. Hanson, N. Palan, and P. Garcia. Teaching inclusive thinking to undergraduate students in computing programs. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, SIGCSE 2018, pages 717–722, February 2018.

[14] C. Marling and D. Juedes. CS0 for computer science majors at Ohio University. In *Proceedings of the 47th ACM Technical Symposium on Computer Science Education*, SIGCSE 2016, pages 138–143, February 2016.

[15] World Health Organization. *World Report on Disability*. WHO Press, Geneva, Switzerland, 2011. Available: `http://www.who.int/disabilities/world_report/2011/`.

[16] A. Tartaro and H. Cottingham. A problem-based, survey introduction to computer science for majors and non-majors. *Journal of Computing Sciences in Colleges*, 30(2):164, 2014.

[17] Y.D. Wang. A holistic and pragmatic approach to teaching web accessibility in an undergraduate web design course. In *Proceedings of the 13th Annual Conference on Information Technology Education*, SIGITI 2012, pages 55–60, October 2012.

Learning Computer Graphics Via A Student-Led Open Source Demonstration Project*

Pushpa Kumar and Kang Zhang
Department of Computer Science
University of Texas at Dallas
Richardson, TX 75080
{pkumar,kzhang}@utdallas.edu

Abstract

Computer Graphics is a computer science subject involving heavy mathematics and many classic graphics algorithms. Providing students hands-on learning experience via programming projects is essential but insufficient. In this paper, we share our experiences in teaching Computer Graphics by letting students build their own algorithm animation and demonstrations open source software. The open source demonstration software, called CGDemo, includes animation and interactive demonstration of various classic graphics algorithms and 3D mathematic transformations. Students first learn the algorithms by developing demonstration projects in Java, following a consistent demonstration framework, and meanwhile by learning and reusing software components built by other students. The gradually built open source project CGDemo has been helping all the subsequent students to learn complex graphics algorithms via intuitive animation and interactive demonstration. We report our evaluation of students' experiences in learning computer graphics by using CGDemo. Over the years, this computer graphics course has evolved into a suite of learning resources, including a Springer textbook, a set of 37-session MOOC videos and the aforementioned CGDemo open source software.

1 Pedagogy Challenges

Computer Graphics is a computer science subject involving heavy mathematics and many classic graphics algorithms. Providing students hands-on learning experience via programming projects is essential but insufficient. There are several challenges in teaching a Computer Graphics course. This subject is math-intensive and involves various mathematics concepts, particularly those in linear algebra, such as vectors and matrix transformations. Students with insufficient or weak math backgrounds, and those who are used to tool simulations without much preparation in theoretical aspects, can find this course initially daunting. There are very few readily available educational resources, either online or offline to ease this transition. Three-dimensional transformations which are central to several topics in this course can be hard to explain and conceptualize on a two-dimensional screen or medium. Even for those students with sufficient math knowledge, some graphics algorithms can be challenging to understand and visualize. There is also sometimes a disconnect between the assigned homework projects and the algorithmic concepts taught in this course. This can lead to students being unable to relate to the course material and unmotivated to learn the classic graphics algorithms.

2 Teaching Strategies and Educational Resources

Our main teaching strategy is aimed at increasing student engagement and involvement in the course. First, though demanding in mathematics and algorithms, computer graphics concepts and algorithms are best understood by practical implementations. Second, there should be a clear conceptual path and relationship from math representation to algorithm expression, and finally to code implementation, so that students can directly relate source code to math formulas (e.g. from Java variables to math symbols and vice versa). Third, students feel motivated by the opportunity of building a demonstration software for software visualization [6][7] and to contribute their own understanding of different concepts to their future counterparts.

Hands-on programming exercises are therefore assigned throughout a semester to illustrate and clarify classic algorithms and concepts from the course, using Java programming language without relying on OpenGL. They need therefore to understand the most fundamental concepts of computer graphics at the lowest level, for example, how to find points to approximate a straight line (i.e. line drawing algorithm). We consider this to be extremely important for students majoring in computer science.

We provide our students with access to several resources in our Computer Graphics courses. These include a web-based eLearning platform [3] for project announcements, course materials, discussion, evaluation, and grading. To facilitate after-class review and consolidation, students have access to an online resource called the Massive Online Open Course (MOOC) [2] with 37 well-explained and topic-based video synopses, each lasting approximately eight to eleven minutes. The required class textbook [4][5] also includes freely available Java source code implementations for all the important concepts and algorithms covered in this course. The textbook also provide course instructors with answers and sample implementation codes for exercises behind each chapter. Additionally, there is algorithm concept demonstration software provided for easy step-by-step understanding of each algorithm using algorithm and software animation techniques [1]. The remaining part of the article describes the development process and evaluation results of this open source software in details.

3 Open-Source Demonstration Software

The project on the open-source demonstration software, known as CGDemo, started more than 12 years ago, that has been gradually involving into the current state, including six different computer graphics algorithms. This web-based open source software facilitates community building and learning since it is freely available. Feedback on the demonstration is also used to improve, refactor and customize the software.

In some of the semesters in the past 12 years, the students enrolled in the graduate level computer graphics class were assigned a homework project, based on one of the classic graphics algorithms or concepts, and valued at 6-10% of the overall course grade. The requirements of the project include a consistent user interface, support of step-by-step demonstration, and display of changes in key variable values of the given algorithm using software animation and visualization techniques [6][7]. Appendix lists an example project description, that was used to develop demonstration of the Cohen-Sutherland Line-Clipping algorithm.

The best student submission for each algorithm was chosen to be included in the software. Criteria included the visual look and feel, interactiveness, intuitiveness, efficiency, speed of rendering, and algorithm conceptualization. The final user-interface was then developed by a Teaching Assistant (TA) for this course, which integrated the best submissions for various algorithms into a final platform. The entire project is still continuously evolving and changes are incorporated on an ongoing basis. Our CGDemo demonstration is available as an open source project on GitHub [1].

The following figures illustrate screenshots obtained from running our CGDemo software for classic Computer Graphics algorithms, and 3D transformation concepts. Figure 1 shows the Viewing Transformation concept that animates the process of three 3D transformation operations; Figure 2 gives Bresenham Line Drawing Algorithm that shows step-by-step selection of pixels to approximate the straight line determined by two user-clicked endpoints; The program for Cohen-Sutherland Line Clipping Algorithm in Figure 3 demonstrates how the line determined by two user-clicked endpoints is clipped step-by-step with one of the nine regions highlighted. Figure 4 depicts step-by-step how Sutherland-Hodgman Polygon Clipping Algorithm works on the user-defined polygon by clipping away polygon portions outside of the boundary. Finally, Figure 5 demonstrates how the user-selected order of Bezier Curve is generated via animation.

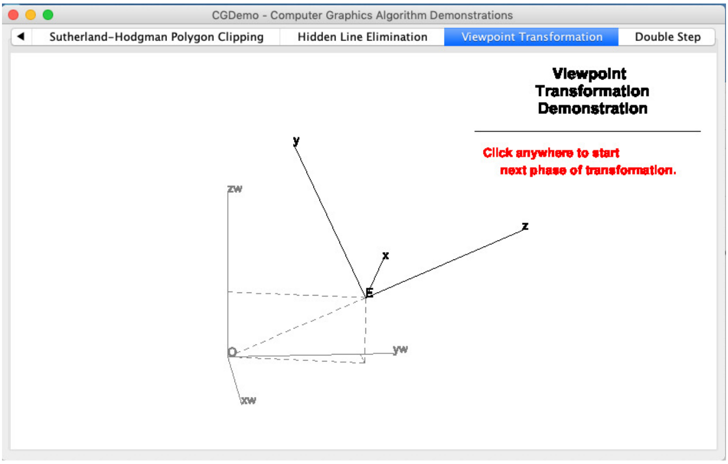


Figure 1: Viewing Transformation Animation

4 Student Evaluation

To receive objective feedback for the usability and usefulness of CGDemo, we have conducted student evaluation twice during two semesters. The two semesters are:

- 2017 Spring semester: 22 students completed the survey
- 2018 Summer semester: 16 students completed the survey

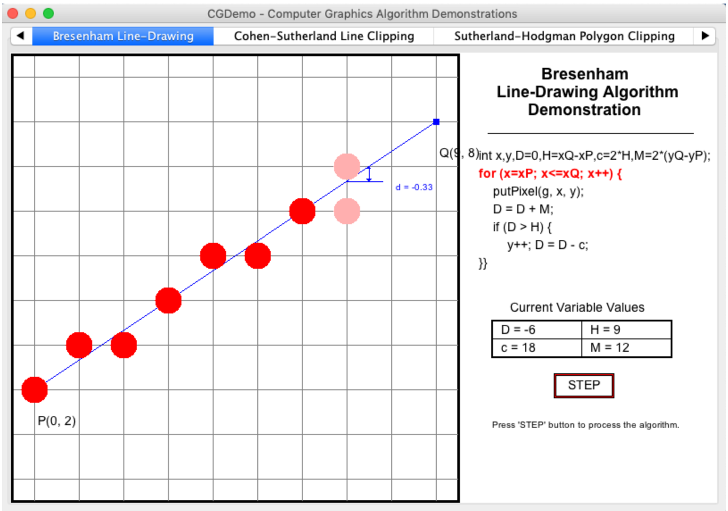


Figure 2: Bresenham Line Drawing Algorithm

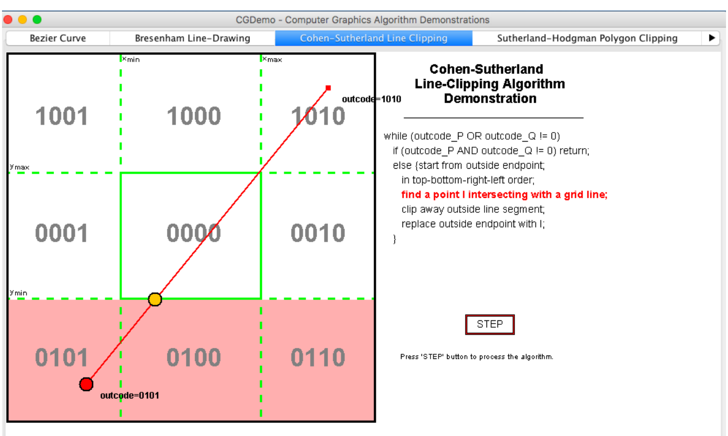


Figure 3: Cohen-Sutherland Line Clipping Algorithm

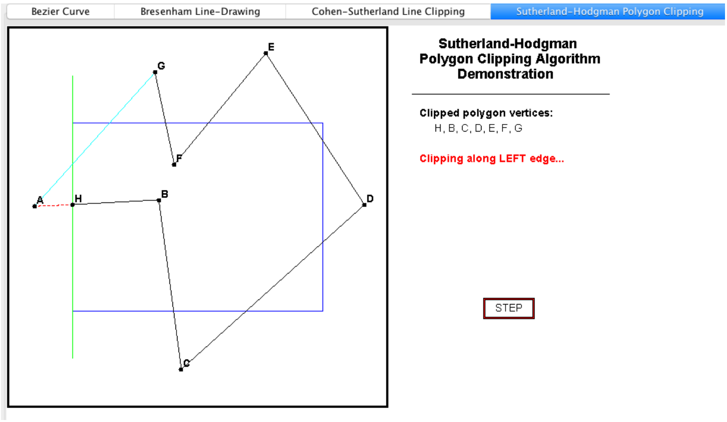


Figure 4: Cohen-Sutherland Line Clipping Algorithm

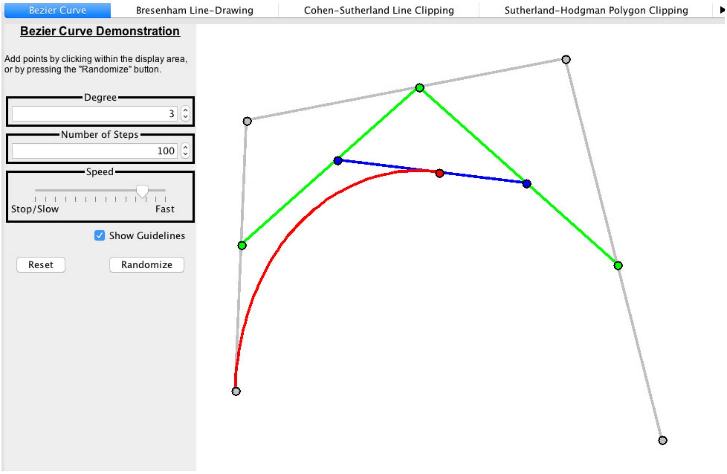


Figure 5: Bezier Curve Drawing Animation

4.1 Questionnaire

The evaluation includes a simple questionnaire that could be completed in 5 to 10 minutes. The questionnaire includes 4 questions designed to gauge the usefulness of our software CGDemo and suggestions for improving it, as shown in Table 1.

Table 1: CGDemo evaluation questionnaire

1. Have you used the open source demonstration software CGDemo throughout this class?
2. According to the demonstration by the instructor you saw in the class, and/or your own experience with CGDemo, rate in brackets how helpful each animation is (0: useless, 6: extremely helpful):
- Viewing Transformation
- Bresenham Line Drawing
- Double-Step Line Drawing
- Cohen-Sutherland Line Clipping
- Sutherland-Hodgman Polygon Clipping
- Bezier Curve"
3. Any other computer graphics algorithms you would like to be added to the software?
4. Any suggestions on improving the software?

4.2 Results

The results obtained from analyzing the student questionnaire are shown in Figures 6, 7, and 8, and Tables 2, and 3.

Figure 6 highlights CGDemo demonstration software usefulness. Out of a total of 22 students in 2017, 18 students rated between 4.0 and 6.0 for all the classic algorithms from the questionnaire. In 2018, 13 students out of 16 rated between 4.0 and 6.0 for the algorithms. It was interesting to note that the software usefulness from information obtained in the 2017 and 2018 student surveys was the same at 81%. Sutherland-Hodgman polygon clipping, Cohen-Sutherland line clipping, and Viewing Transformation animation received the highest rating (17/18 students at 94%) for usefulness in 2017 as depicted in Figure 7 while Bezier curve and Cohen-Sutherland line clipping demonstrations received the highest rating (13/13 students at 100%) in 2018 as depicted in Figure 8.

Table 2 illustrates potential additional algorithms that can be added to the CGDemo demonstration software. This is part of an ongoing software evolution process and homework on interesting topics like Fractals, Hidden

Table 2: CGDemo software evaluation results on future improvements (Question 4)

Perhaps a hidden-face removal demonstration, as well as a fractal demonstration. And the availability of the Bezier curve demo if it still exists.
Hidden face elimination.
String Grammars.
Point in polygon.
It would be cool to have a Fractal Grammar interface, and to see the algorithm explained.
Fractals – Koch curves and Mandelbrot set discussed in class was really helpful.
ViewingTransformation can allow the user to choose point E. This will be really amazing.
It will be best if we can have demo for World<->Eye<->Perspective transformation.
3D demo.
Environment Light. Texture.

Table 3: CGDemo software evaluation results on future improvements (Question 4)

Perhaps a hidden-face removal demonstration, as well as a fractal demonstration. And the availability of the Bezier curve demo if it still exists.
Hidden face elimination.
String Grammars.
Point in polygon.
It would be cool to have a Fractal Grammar interface, and to see the algorithm explained.
Fractals – Koch curves and Mandelbrot set discussed in class was really helpful.
ViewingTransformation can allow the user to choose point E. This will be really amazing.
It will be best if we can have demo for World<->Eye<->Perspective transformation.
3D demo.
Environment Light. Texture.

Table 4: CGDemo software evaluation results on future improvements (Question 4)

For the viewing Transformation animation, more comments should be added to make each step more explicit.
In Bresenham line drawing, the implementation of drawing the last point without calculation.
In Viewpoint Transformation, it would be better if each steps are explained along with their explanations on GUI while demonstrating.
For the Sutherland-Hodgman Polygon Clipping, I believe it will be better to make the start vertex bigger since sometimes it is not easy to draw the end vertex as the same position as the start one. As shown in the picture below(see A and E).
I think those are great software and help a lot with understanding the algorithms. From my point of view, it will be helpful if we have more demo for 3D-related algorithms.
Give more description about each algorithm.
If one can integrate all software in a same program and unify the GUI, it is better for the open-source software to be an advantage of this course.
The needed bug fixes for the line drawing algorithms. For the viewing transformation, perhaps instruction on the screen on how to rotate the viewpoint (the right click). For the hidden line demonstration, perhaps a larger range on how far/near you can bring the object (there may have been a limitation because of the file I used... I used the stairs.dat file from Assignment 4). Perhaps when I get through my studies I will assist in some bug fixing.

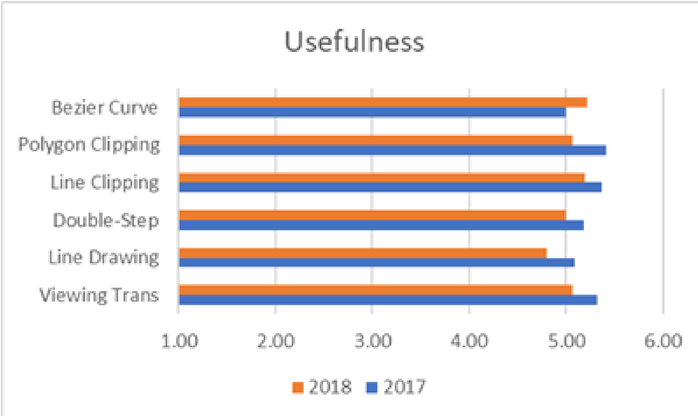


Figure 6: CGDemo software evaluation results on usefulness (Questions 1 and 2)

Face elimination, String Grammars, Transformations, and Color Theory could be designed to augment our current software.

Table 3 summarizes the feedback on potential future improvements for the CGDemo demonstration software. Efforts can be undertaken to improve the overall usability, maintainability, interactiveness, integration, and intuitiveness. Any existing bugs can also be eliminated with refactoring efforts. Specifically, Bresenham Line Drawing algorithm, Sutherland-Hodgman Polygon Clipping algorithm, and Viewpoint transformation animation can be enhanced.

5 Summary

Mathematics-intensive subjects such as Computer Graphics can be abstract and daunting to many students. To increase student understanding, engagement, and involvement in this course, we designed and assigned homework keeping this goal in mind. Students then built algorithm demonstration software using Java programming language and the best submissions were chosen for the various assignments. The open source demonstration software, called CGDemo, which integrates these submissions includes algorithm animation and interactive demonstration of various classic graphics algorithms and 3D mathematics transformations. Students first learn the algorithms by developing demonstration projects in Java, following a consistent demonstration framework, and meanwhile by learning and reusing software components built by other students. The gradually built open source project CGDemo has been

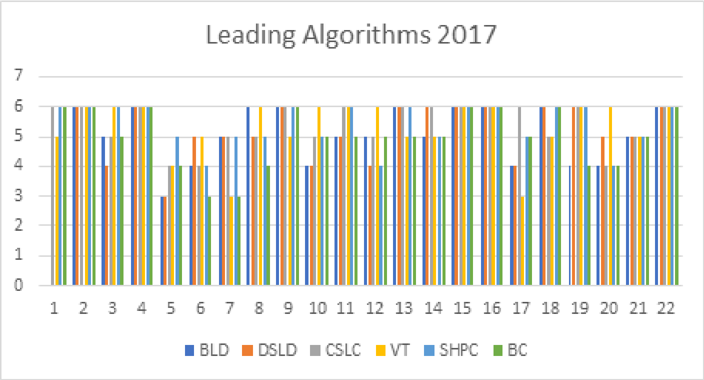


Figure 7: CGDemo leading algorithms by usefulness 2017 (Questions 1 and 2)

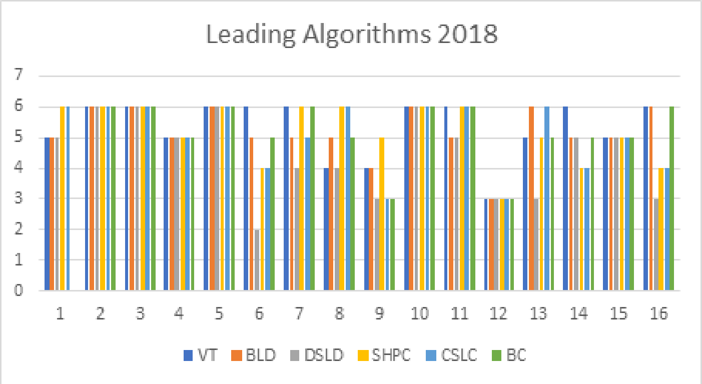


Figure 8: CGDemo leading algorithms by usefulness 2018 (Questions 1 and 2)

helping all the subsequent students to learn complex graphics algorithms via intuitive animation and interactive demonstration. The entire project is still continuously evolving and changes are incorporated on an ongoing basis.

References

[1] CGDemo Open Source Project. https://github.com/Hengxiugao/CGDemo.

[2] Massive Online Open Course. https://www.utdallas.edu/~kzhang/CG_MOOC.html.

[3] Web eLearning Platform. http://elearning.utdallas.edu.

[4] Leen Ammeraal and Kang Zhang. *Computer Graphics for Java Programmers.* John Wiley & Sons, 2nd edition, March 2007.

[5] Leen Ammeraal and Kang Zhang. *Computer Graphics for Java Programmers.* Springer, 3rd edition, September 2017.

[6] Peter Eades and Kang Zhang. *Software Visualization, Series on Software Engineering and Knowledge Engineering.* World Scientific Co., vol.7 edition, 1996.

[7] Kang Zhang. *Software Visualization - From Theory to Practice.* Kluwer Academic Publishers, April 2003.

A Appendix - A Sample Homework Project Description

Implement a demonstration system for Cohen-Sutherland line-clipping algorithm. The system visualizes how the algorithm works by showing its stepwise execution of lines being clipped on a 3x3, i.e. a 9-region grid. The grid's 9 regions are marked by their outcodes, with center region being the clipping rectangle in solid lines surrounded by 8 dotted outgoing lines, with an open outside border.

Draw the grid on the left half of the drawing space and the following pseudocode on the right half with text font size of 14:

```
while (outcode_P OR outcode_Q != 0)
    if (outcode_P AND outcode_Q != 0)
      return;
    else {
       start from outside endpoint;
       in top-bottom-right-left order;
       find a point I intersecting with a grid line;
       clip away outside line segment;
       replace outside endpoint with I;
    }
```

Add a button "Step" at bottom of the drawing space, and a small space for displaying the outcode values of the two endpoints.

To test a line to be clipped, the user clicks on a mouse button when the cursor is at the desired location of any part of the grid to represent the start point P of the line. When the user moves the cursor away to determine the end point Q, the line from P to the cursor should follow the cursor's movement (the so-called rubber-band effect) until the user clicks the mouse button. The whole line to be clipped is then drawn as a thick line on the grid. All the intersection points on the grid are drawn as red dots in a notable size.

Whenever the user clicks on "Step", one line of psuedocode in the right half is highlighted, (by coloring or boxing the statement), and the corresponding action is performed and highlighted on the left grid (such as outside endpoint flashing, intersection point flashing, outside line segment flashing and disappearing, etc). (Bonus) Add advanced features to the above system, such as showing the process of signs being generated and filled into outcodes, the order of working through the outcode bits, and any other creative but useful features that assist in understanding of Cohen-Sutherland Algorithm.

Industry Challenges for Algorithms Analysis Students*

Hala ElAarag, Mohammed H Batarfi, Isabel Ho Li,
Tram Nguyen, Jesus Argel, Brandon A Belna,
Kyle N Burda, John H. Sawyer
Math and Computer Science Department
Stetson University, DeLand, FL 32723
{helaarag, mbatarfi, jiholi, tnguy3}@stetson.edu
{jargel, bbelna, kburda, jsawyer}@stetson.edu

Abstract

One of the most important skills that students should master to succeed in the 21st century is the transfer skill. This skill enables an individual to take what was learnt in one situation and apply it in another situation. Algorithms Analysis is a core component of the Computer Science Curricula. In this paper, we share our experience in using this course as a vehicle to enable students to master their transfer skill and hence ensure a deeper learning experience. The students were asked to participate in a challenge provided by a flight booking company to solve a harder version of the Traveling Salesman Problem. In this paper, we present the challenge as well as three different student proposed solutions. We provide a comparison of the solutions based on their execution time and cost as well as the summary of a survey that demonstrates the student positive perception of such experience.

1 Introduction

Cognitive research shows that students learn more when they are engaged in their studies and see them as important. Computer Science educators strive to find novel ways to teach core concepts using exciting hands on and practical approaches[2][3]. Levitin [5]and Nohl [7] suggested using puzzles in teaching algorithm analysis. Moore [6] suggested using genetic algorithms for solving optimization problems. Xin [9] shared his experience in reforming the algorithms analysis course at an undergraduate institution. He proposed using visualization-teaching and problem-driven teaching. We totally agree with understand complex algorithms like the traveling salesman problem [4] or the smoothest path problem [8]. In this paper, we present a problem-driven approach teaching.

In 2018, a flight booking company (kiwi.com) offered a challenge based on the traveling salesman problem. The traveling Salesman problem is an NP hard problem. The task and the rules are described on their website [1] as follows:

> Your task is to determine the cheapest possible connection between specific areas, where an area is a set of cities. You will be provided with Kiwi.com flight data to help you work out the best algorithm. The solution must meet the following criteria:
>
> - The trip must start from the city we give you
> - In each area, you must visit exactly one city (but you can choose which one)
> - You have to move between areas every day
> - The trip must continue from the same city in which you arrived
> - The entire trip should end in the area where it began
>
> Several input and output files were provided to test the solutions. The evaluation input files were very extensive. The first line of the input has the number of areas that need visiting and the starting city. Then, the following lines have information of the areas and cities within each area. Finally, the routes between cities, the day in which that route is offered and their associated cost were listed. A simple example of input and the corresponding output is shown below:
>
> Input:
> 3 A
> Area1
> A
> Area 2
> B
> Area3
> C
> A C 1 20

A B 1 10
B C 2 15
B A 3 10
C A 3 10

Output:
35

The provided development environment utilized the Sphere Engine service which provides an online code compiler for many (more than 60) languages, including all of the most popular languages (C++, C#, Go, Haskell, Java, Kotlin, Node.js, PHP, Python, Ruby, Scala, Swift) and many more. Input was provided from the site through the standard input (stdin) as well as via text files downloadable from the site. Output was directed to standard output (stdout). The attempted solution was repeatedly executed with the input data for each test case, then the environment gave a few different end-statuses:

- Accepted: if the provided solution was correct (but not necessarily optimal). An accepted solution is accompanied by a score based on execution time/cost of path.
- Wrong answer: if the program ran, but didn't give a correct solution
- Time-limit exceeded: if the program ran but didn't terminate within the allowed time
- Compilation and Run-Time errors: if the program was unable to be compiled/started respectively, with standard error details provided for debugging.

Prizes were offered to the top three teams. The first prize is a trip around the world. The second and the third prizes are 1000 and 600 euros, respectively in kiwi.com travel vouchers. The top ten teams were invited to a gala in Prague.

2 Proposed Solutions

In Fall 2018, seven students were taking the algorithm analysis course. The class was divided into three groups of 2 or 3 students and every group registered as a team. In this section, we present the solution suggested by each team. The students were familiar with the Traveling salesman problem (TSP); that given a list of n cities and the distances between cities seeks to find the shortest route to start at a home city, visit each city once and return to home city. They were also familiar with a brute force solution of O (n!) and a dynamic programming solution of O (n2 2n), in addition to backtracking and branch and bound approaches to this problem. Unlike TSP, the challenge not only have cities (areas), but also multiple cities within each area. A brute force solution would be O ((n*m)!), where n is the number of areas and m is the number of cities within the area.

2.1 Approach 1 (Annealing)

This approach was based on the simulated annealing algorithm. Annealing refers to a technique that involves controlling temperature of a material. Simulated annealing, therefore, exposes a solution to heat and as it cools it will produce a more optimal solution. Simulated annealing works by moving from the starting temperature to the ending temperature for each iteration. The cycle count allows you to specify the rate at which the temperature decreases. The higher the initial temperature, the more randomness is introduced into the system.

This algorithm is implemented with the following steps:

```
Initialize beginning Temperature
Initialize ending Temperature
Initialize Cooling Rate
Generate Initial Travel
Start Cooling down process {
   Decrease temperature according to cooling rate
   Calculate current Cost
   Swap random areas
   Calculate new Cost
   If (new Cost > current cost)
     Revert to previous route
   else
     Keep updated route
    current Cost = new Cost
  }
Repeat until the temperature is below ending Temperature
```

We chose the beginning and ending temperatures to be equal to 10 and 0.0001, respectively, while the cooling rate was chosen to be equal to 0.05. In this approach, first an initial travel route was generated by starting with the specified city then the next destination is randomly selected from the list of places until all the possible areas are visited. With an initial travel itinerary, the swapping areas step is performed by picking two random areas from the travel plan and their orders are swapped to create a new route. In order to determine if the new route is cheaper than the current route, the total cost of the two routes are compared and the one with the lowest cost is saved. In the case when the new route is more expensive, a method called "Revert Swap" is performed to revert back to the route before the swap. An analysis of this algorithm is hard due to its randomness.

2.2 Approach 2 (Gready)

In this approach a Greedy technique is applied for each day, meaning the algorithm will go day by day and choose the lowest cost flight as follows:

```
Create vector for best route
best route += initial city
day = 1
currentCity = initial city
boolean areas[0..maxAreas]=F
areas[currentArea]=T
visitedAreas[0..maxAreas]=""r
visitedAreas[0]=initial city Area
destination = ""
totalPrice=0
for (i = 1 to maxAreas)
{
 Check 1: the day of the flight matches the day counter
 {
  for ( departureCity = 0 to maxCities)
  {
   Check 2: departureCity  matches the currentCity
     {
     for ( arrivalCity = 0 to maxCities)
     {
      Check 3: city has not been visited, has the cheapest cost
               and areas[arrivalCity]=F
      {
        destination = arrivalCity
        areas[arrivalCity]=T
       } // end Check 3
     } //end arrivalCity loop
   } //end Check 2
   currentCity= destination
   best route += currentCity
   total price += cost
```

```
    visitedAreas[i]=area of currentCity
    Break out of the departureCity loop
   } //end of departureCity loop
  } // end Check 1
  day ++
 } //end outermost loop
 Add initial city to the best route to close the circuit
 totalPrice += cost of  final flight
 Return the best route
```

In this algorithm, the outermost loop iterates through the number of areas. It then checks to see if area number matches the current day number. If they are equal, it executes the next for loop which iterates through the number of cities. In this loop, the algorithm checks to make sure that the city is a departure city. If true, then it goes through the innermost loop which iterates through the number of cities again and does three checks:

- If the city we are traveling to was not traveled to yet.
- If this option is the cheapest one found
- If the area was not traveled to yet.

Once that area is approved, it is added to the array of visited areas, the price is added to the total cost of the trip and the city is added to the best route vector. Then we increase the day counter. After all the flights have been found we then add the final flight to get back to the initial city based on the final city visited.

The analysis of this algorithm is simple. In the worst case, it is equal to:

$$\theta(n * m^2)$$

where n is the number of areas and m is the number of cities/area.

2.3 2.3 Approach 3 (Hybrid)

This approach is based on both the Greedy and Backtracking techniques. The algorithm works as follows:

```
Start at first city
while (there is an unvisited area) {
  find cheapest flight out to an unvisited area
  if (a flight is possible) {
    add flight to stack
    current city is now that flight's arrival city
    add area to visited areas
  }
  else {
    backtrack to conditions before previous flight
    pop flight from stack
```

```
      If (flight is a day 0 flight)
         create an entry in bad day 0 hashmap
     }
    if (all areas have been visited) {
      take cheapest flight to starting area
      if (none exist) {
         backtrack to conditions before previous flight
         pop flight from stack
         If (flight is a day 0 flight)
             create an entry in bad day 0 hashmap
      }
   }
 }
```

This approach takes the cheapest flight at each step but it backtracks if a flight is not possible. Hence, it is guaranteed to find a solution if one is available, but not necessarily the optimal one. The best case occurs when the cheapest flights in every day always connect to an unvisited area. The complexity in this case would simply be O(n), where n is the number of areas. The worst case occurs when the only possible path occurs through the most expensive flights.

Figures 1, 2 and 3 illustrate the trip selected by each of the three approaches on a simple example, its execution time and cost.

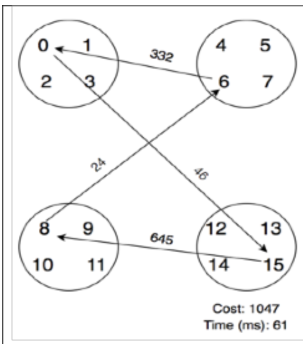


Figure 1: Annealing Algorithm

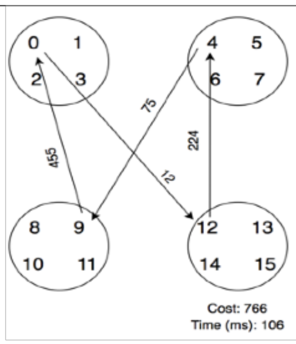


Figure 2: Greedy Algorithm

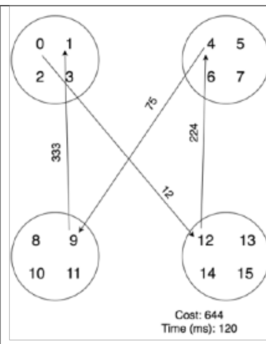


Figure 3: Hybrid Algorithm

3 Results

In this section, we present a comparison of the execution time and cost of the three proposed approaches for large numbers of areas (n) and cities (m). All approaches were run on a MacBook Pro with 2.3GHz Dual-Core 7th-generation Intel Core i5 processor, Turbo Boost up to 3.6GHz and 8GB memory. The graphs show the average of 1000 runs.

Figures 4 and 5 show the execution time in milliseconds and cost vs. the number of cities (m), when the number of areas (n) = 1000, respectively. As we can see from Figures 4 and 5, the hybrid approach is the slowest but provides the cheapest cost, while the annealing approach has the fastest run time but provides the highest cost. The Greedy approach is in the middle in terms of runtime and cost. From Figure 4, we can deduce that an average execution time of O (n*m) is an upper bound for all three approaches.

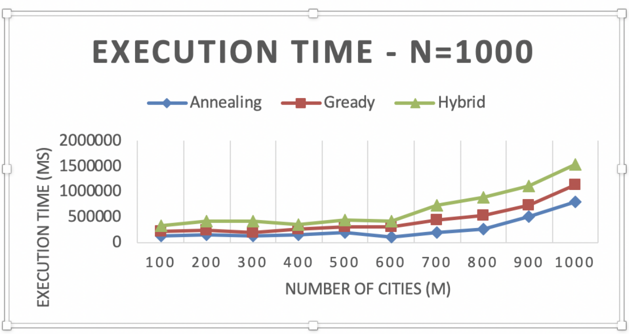


Figure 4: Execution time in ms vs. number of cities for 1000 areas

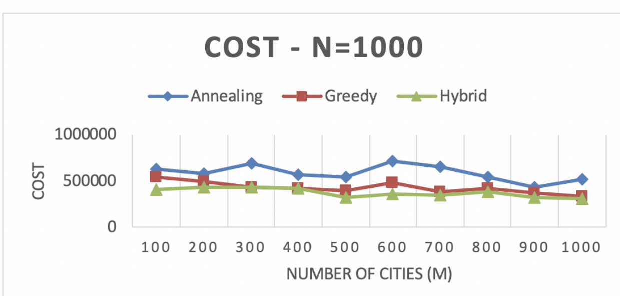


Figure 5: Cost vs. number of cities for 1000 areas

Q1. Did the TSP challenge increase your interest in the course?	Yes 100%
Q2. Did the TSP challenge increase your engagement in the course?	Yes 100%
Q3. Did the project enable you to take what you learned in the class and apply it in another situation?	Yes 100%
Q4. Did the project provide you with a deeper learning experience?	Yes 100%
Q5. Do you recommend using industry challenges as a class project in the future?	Yes 100%
Q6. On a scale from 1 to 5 (where 1 is the least and 5 is the most), Rate the difficulty of the project	3.9
Q7. Approximately how many hours you spend on the project	24.6 hrs.

4 Evaluation

To evaluate using industry challenges in an algorithms analysis course, we designed a survey that consists of 10 questions. Five out of 7 students filled the survey. Questions 1 to 5 had a yes, no, no opinion option. The following table presents the results of the survey.

The survey included three qualitative questions. We list the students' answers to these questions below.

Q8. What challenges did you face?

- Since it was our first time participating in a competition, we did not know how to store the information from the input we were given. However, after careful consideration we decided to save them in a 3D matrix with the date, departure and arrival airports as the dimensions.
- Implementing the algorithm and solving the last airport/area
- We didn't know what to do with the TSP, because there was no optimal way, so we had to brute forced everything in the beginning
- Working with 3D arrays as the main data object to analyze.
- Thinking outside of the box to create a solution that is fast and that also works.

Q9. What did you like most?

- We were able to do research and see what algorithm to implement
- Solving the test cases with different approaches
- It was something new and applicable in daily life
- The flexibility in which we could approach the challenge.
- I like the whole competition aspect that the project was with

Q10. What did you like least?

- The time it took to run with larger size of airports and cities
- Changing the test cases into inputs
- I wish we were able to make a UI for the project, but we focused on the algorithm more, so that was fine
- Not having more time to go deeper into the program and optimize it.
- The short time limit that we had I wish we had more time to explore the project.

5 Conclusion

In this paper, we have presented our experience in using challenges provided by companies in the classroom. At the beginning, the students have found it a daunting problem. However, they all raised to the challenge and each group provided a different approach to the solution of the problem. The students that semester missed the deadline to participate in the competition as the instructor became aware of it about ten days before the deadline. In spite of the small sample size, the student evaluation showed that this type of challenge provides a great tool to ensure deeper learning and arm students with the skills they need to succeed in the 21st century.

References

[1] `https://travellingsalesman.kiwi.com/`.

[2] H. ElAarag. Teaching computer organization - a practical approach. *Journal of Computing Sciences in Colleges*, 28(2):210–217, 2012.

[3] H. ElAarag. Deeper learning in computer science education using Raspberry Pi. *Journal of Computing Sciences in Colleges*, 33(2):161–170, 2017.

[4] H. ElAarag and S. Romano. Animation of the traveling salesman problem. In *Southeastcon, 2012 Proceedings of IEEE*, IEEE, pages 1–6, 2012.

[5] A. Leviting. Analyze that puzzles and analysis of algorithms. *ACM SIGCSE Bulletin*, 37(1):171–175, 2005.

[6] M. Moore. Teaching students to use genetic algorithms to solve optimization problems. *Journal of Computing Sciences in Colleges*, 16(3):19–25, 2001.

[7] D. Nohl. Solving the pharaoh's golden staircase problem through dynamic programming. *Journal of Computing Sciences in Colleges*, 20(1):54–60, 2004.

[8] J. A. Roles and H. ElAarag. A smoothest path algorithm and its visualization tool. In *Southeastcon, 2013 Proceedings of IEEE*, IEEE, pages 1–6, 2013.

[9] L. Xin. Reform and practice of introduction to algorithmic design & analysis in local undergraduate colleges. In *Proceedings of the 2nd International Conference on E-Education, E-Business and E-Technology*, ACM, pages 63–67, 2018.

SYN, SYN-ACK, and Beyond: Learning by Networking*

Bilal Shebaro
Department of Computer Sciences
St. Edward's University
Austin, TX 78704
bshebaro@stedwards.edu

Abstract

This paper aims to discuss the best practices of how to introduce students to the world of computer networks, network protocols, and network security using practical and hands-on methods. We focus our discussion on the use of the Scapy and Wireshark tools, which allow instructors to demonstrate what happens under the hood when data is sent over the network and its security. Using these tools, students will not only have a practical learning experience, but they will also be able to manipulate the network traffic and analyze network traces from both offensive and defensive perspectives.

1 Introduction

Traditionally, students are introduced to computer networks concepts through the presentation of the OSI model and the TCP/IP protocol. Students are taught about the different layers in each protocol and the information that each layer adds to the packet headers, the data encapsulation (sender), and the striping (receiver) processes. However, if these concepts are taught solely through the use of figures and theoretical models, students may fail to fully understand and apply these concepts. This may be especially problematic if these concepts are prerequisites to more advanced topics like network security or advanced networking courses. Therefore, this paper presents a practical approach to introduce computer networking protocols and network security practices through the use of tools such as Scapy and Wireshark, and through the provision of practical exercises and examples that vary from basic networking applications (such as port scanning) to common network security attacks (such as ARP cache poisoning attacks).

This paper is organized as follows: Section 2 introduces Scapy and Wireshark as the main tools used in this teaching approach. Section 3 discusses the hands-on approach of learning the network basics through the implementation of basic network applications with Scapy and its integration with Wireshark. Section 4 expands even further on Scapy's capabilities by introducing students to network security applications and examples of network attacks. Finally, we share our thoughts of more network applications and exercises for student assignments and project ideas in Section 5, followed by our concluding remarks in Section 6.

2 Scapy and Wireshark

Scapy is a powerful interactive packet manipulation tool, written in Python by Philippe Biondi, that enables users to create, forge, or decode packets and send them over the network [3]. Programmers can use Scapy to customize and implement their own network tools and perform tasks such as port scanning, tracerouting, nmap, arpscan, and other networking jobs [2, 4]. Scapy can also be used for network packet sniffing and traffic analysis, and it allows network administrators to implement various network attacks and defenses to test the security of their own network [6, 7].

Wireshark, on the other hand, is a popular network protocol analyzer tool with support for many network protocols that also allows network administrators to intercept, log, and analyze network traffic and data [1]. While Scapy provides a Python interface to manipulate and customize packets and send them over the wire, Wireshark provides a richer graphical interface for deep packet inspection and a microscopic level view of what is happening in the network.

In what follows, we demonstrate how instructors and students can use Scapy along with Wireshark to learn computer networking basics and write Python programs that can perform the jobs done by many networking tools [5, 4]. We will also provide more extensive applications of the tools in order to implement network discovery tools and attacks.

3 Networking in Practice

In this section, we highlight how Scapy and Wireshark can be used together to introduce students to how data packets are built according to the network protocol layers. We will briefly discuss how Scapy is used to manipulate and send packets over the network, and use Wireshark to record the network traffic and view our customized sent packets.

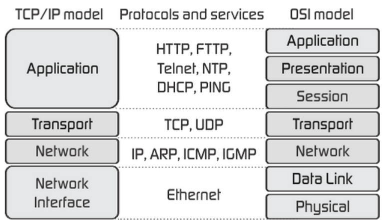


Figure 1: OSI and TCP/IP models and supported protocols

3.1 Network Protocol Layers

After students are introduced to the OSI and TCP/IP models and the protocols and services that each protocol layer supports (Figure 1), we will use the Scapy interpreter to forge data packets to appear as if it is part of the normal communication stream. Scapy uses the concept of stacking where a packet is created as layers that are stacked one upon another. The example below shows how Scapy stacking works where a packet A was created at the Network or Internet layer and has its source and destination IP addresses modified. Then, the packet B is stacked on packet A adding parameters at the Transport layer to change the values of the source and destination ports. Finally, the packet C is stacked on packet B to specify the source MAC address of the sender at the Network Interface layer. In other words, the packet C is a result of stacking the three protocol layers $Ether()/IP()/TCP()$. While Wireshark is sniffing packets on the same network interface that Scapy is using, we use Scapy to send the packet C and observe the packet details recorded on the Wireshark recorded traffic. Figure 2 shows the dissection of the last version of packet C showing the parameters that were customized using Scapy.

```
>>> A=IP()
>>> A.src="192.168.29.201"
>>> A.dst="192.168.29.202"
>>> A
<IP  ttl=32 src=192.168.29.201 dst=192.168.29.202 |>
-----------------------------------------------------
>>> B=A/TCP()
```

```
>>> B.sport=123
>>> B.dport=456
>>> B
<IP  proto=tcp src=192.168.29.201 dst=192.168.29.202 |
<TCP  sport=123 dport=456 |>>
---------------------------------------------------------
>>> C=Ether()/B
>>> C[Ether].src="aa.bb.cc.dd.ee.ff"
>>> C
<IP  proto=tcp src=192.168.29.201 dst=192.168.29.202 |
<TCP  sport=123 dport=456 |
<Ether  src=aa.bb.cc.dd.ee.ff |>>>

>>> send(C)
Sent 1 packets.
```

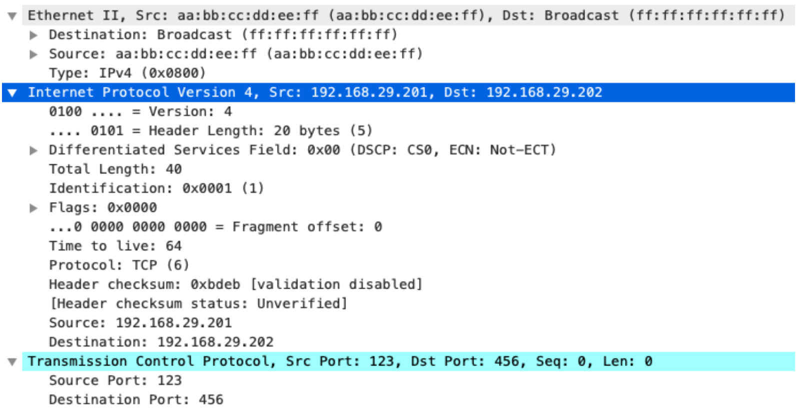
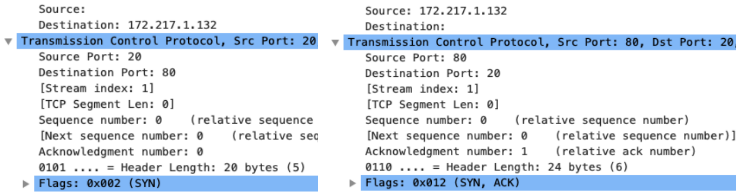


Figure 2: Wireshark output of forged Scapy packet

3.2 TCP 3-Way Handshake

One of the basic topics in computer networks is explaining the TCP 3-Way Handshake process. This is the initial process of communication that happens between devices over the internet. As such, three packets are involved in a 3-way handshake full TCP connection as both parties need to synchronize their segment sequence numbers used during communication. To demonstrate such a process, the example below shows how we use Scapy to send a TCP request to *www.google.com* on the *http* port 80 to initialize a connection between our device and Google servers. Notice the S for SYN and SA for SYN-ACK flag values that were returned in Scapy as well in the Wireshark snapshot in Figure 3.

```
>>> ans, unans = sr(IP(dst="www.google.com")/TCP(dport=80),timeout=1)
Begin emission: Finished sending 1 packets.
Received 2 packets, got 1 answers, remaining 0 packets
>>> ans.summary()
IP / TCP 192.168.29.201:ftp_data > 172.217.1.132:http S
==> IP / TCP 172.217.1.132:http > 192.168.29.201:ftp_data SA
```



(a) Packet sent with SYN request (flag=SYN)

(b) Response received with a SYN-ACK (flag=SYN,ACK)

Figure 3: Wireshark recoded our TCP 3-Way Handshake request to www.google.com.

3.3 Port Scanning

Port scanning is a known network technique that is used to identify open ports and services available on a network host. Network administrators use this simple tool to close any unnecessary open ports and audit their servers for vulnerabilities, while hackers can also use this tool to target victims. Using the Scapy example below, students can learn how this tool works by sending a SYN request packet to a range of ports on a target server, and listening for responses. Those answered responses with a SYN-ACK flag define the open ports, while the unanswered packets define the closed ports. The responses we received in the example below show that our webserver has ports *http* (port 80) and *https* (port 443) open, while ports *ftp* (port 21) and *telnet* (port 23) are closed.

```
>>> ans, unans = sr(IP(dst="192.168.29.1")/TCP(dport=[21,23,80,443]),timeout=1)
Begin emission:.....Finished sending 4 packets.
Received 14 packets, got 2 answers, remaining 2 packets

>>> ans.summary()
IP / TCP 192.168.29.201:ftp_data > 192.168.29.1:http S
==> IP / TCP 192.168.29.1:http > 192.168.29.201:ftp_data SA

IP / TCP 192.168.29.201:ftp_data > 192.168.29.1:https S
==> IP / TCP 192.168.29.1:https > 192.168.29.201:ftp_data SA

>>> unans.summary()
IP / TCP 192.168.29.201:ftp_data > 192.168.29.1:ftp S
IP / TCP 192.168.29.201:ftp_data > 192.168.29.1:telnet S
```

4 Beyond the Basics

Additionally, there are more basic network tools that can be easily implemented with Scapy, such as traceroute and Address Resolution Protocol (ARP) ping.

However, students who have mastered implementing these tools may be eager to go beyond the basics and into the scope of network attacks and security. In this section, we show Scapy examples that instructors and students may find useful or inspiring to further their own networking knowledge or their use of Scapy.

4.1 ARP Cache Poisoning Attack

ARP cache poisoning is one of the most popular ways of doing a man-in-the-middle (MITM) attack on a local area network. It is basically the act of modifying the local ARP cache table of the gateway and a target machine so that data packets sent between them are being forwarded first to the attacker machine. In simpler terms, the attacker machine inserts itself between a target machine and the gateway or router. The example below shows how such an attack can be implemented using Scapy. Basically, two ARP packets are created to falsify the records of the ARP cache tables; the first one is sent to the target machine and links the MAC address of the attacker machine to the gateway IP address, while the second one is sent to the gateway and links the MAC address of the attacker machine to the target machine IP address.

```
send(ARP(op=2, pdst=gateway_ip, hwdst=attacker_mac, psrc=target_ip))
send(ARP(op=2, pdst=target_ip, hwdst=attacker_mac, psrc=gateway_ip))
```

Using Wireshark we can view these ARP messages and also observe how the attacker machine is receiving all traffic between target machine and router. Finally, as soon as the attack is completed, it is important to remember to restore the ARP tables of the machines. This can be done through an ARP broadcast to retrieve the correct ARP cache records for the target machine and the gateway. The Scapy code below shows how to reverse the ARP poison attack.

```
send(ARP(hwdst="ff:ff:ff:ff:ff:ff",pdst=gateway_ip,hwsrc=target_mac,psrc=target_ip))
send(ARP(hwdst="ff:ff:ff:ff:ff:ff",pdst=target_ip,hwsrc=gateway_mac,psrc=gateway_ip))
```

4.2 CAM Overflow Attack

While hubs use the flooding technique to deliver data between connected machines, switches have a dynamic table called Content Addressable Memory (CAM) that maps MAC addresses of the connected devices to the ports on the Switch so that data is sent directly to the destination machine. The CAM overflow attack occurs when an attacker intentionally fills up the CAM table to capacity with random MAC addresses, forcing the switch to perform like the hub flooding technique. This attack can be demonstrated using Scapy (see example below) by creating a list of thousands of Ethernet packets with random MAC and IP addresses and forwarding them to the switch CAM table.

```
for i in range(1,10000):
 packet = Ether(src=RandomMAC(),dst=RandomMAC())/IP(src=RandomIP(),dst=RandomIP())
 sendp(packet,iface='SWI0')
```

5 Sample Exercises and Applications

Once students become familiar with working in Scapy and Wireshark and have gained exposure to core network protocols, instructors have the flexibility to design programming assignments and course projects that meet their course goals and objectives. For instance, if the objective is to teach students how to analyze network traces for protection from network attacks, instructors can prepare a network packet trace (a pcap file) that contains specific vulnerability behaviors and the student task is to use Wireshark to find and extract these behaviors. Examples of vulnerability behaviors could include the risks of using FTP or finding the data transferred in non-https website connections. Instructors could lead students with a set of questions that can stir them towards finding those vulnerabilities. If the exercise objective is to automate the process of traffic analysis for detecting security problems, students can write Python programs that use the Scapy module to implement port scanning on a local server, and then scan through the resulting pcap file to identify source addresses that may be attempting a port scan.

Other Scapy programming assignments could also be related to re-design existing network tools to customize their functionality. One example is to design a network monitoring application such as a simplified version of tcpdump that can capture traffic from a network interface and present them in a customized output.

Finally, network attacks can be another popular field to explore. Students can use Scapy to implement network attacks such as attacks on the TCP protocol like SYN flooding attacks or TCP reset attacks, and use Wireshark to find and investigate on such malicious attacks. Such exercises can lead students to learn about network traces from both offensive and defensive perspectives.

6 Conclusion

While Scapy is a Python module that is used to build several computer networking tools and applications, this paper highlights how such a tool can be used as a perfect fit for instructors teaching computer networks-related courses.

Not only does Scapy allow instructors to teach networking concepts in more practical and effective methods, but also its simplicity and flexibility provides instructors with many opportunities for in-class exercises and assignments, some of which are discussed in this paper. We encourage instructors to explore the various capabilities and applications of Scapy and other complimentary tools in order to offer students the opportunity to acquire and practice basic and advanced topics in computer networks and network security.

References

[1] Wireshark: Go deep. https://www.wireshark.org/.

[2] Phillippe Biondi. Network packet forgery with scapy. https://scapy.net/conf/scapy_pacsec05.pdf.

[3] Phillippe Biondi. Scapy: Packet crafting for python. https://scapy.net/.

[4] Phillippe Biondi. Scapy's documentation. https://scapy.readthedocs.io/en/latest/.

[5] Adam Maxwell. The very unofficial dummies guide to scapy. https://theitgeekchronicles.files.wordpress.com/2012/05/scapyguide1.pdf.

[6] Nils Weiss and Enrico Pozzobon. Automotive penetration testing with scapy. https://www.troopers.net/troopers19/agenda/znxvht/.

[7] Craig S. Wright. Searching for exploits, scapy fuzzing. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3153525.

Invoking Interest in Computer Science and Engineering: A Cyber Security Approach*

Christopher Ivancic and Dae Glendowne
Department of Computer Science
Stephen F. Austin State University
Nacogdoches, TX 75962
{ivanciccp,glendowndj}@sfasu.edu

Abstract

With the continuing rise of computing in the modern world, there is a growing need for computer scientists to meet those demands. In an attempt to increase interest in computer science as a major, focus has been given to educational programs at the high school and middle school levels. By exposing students to computer science concepts earlier in their education, students should be better prepared to obtain a computer science degree. Exposure is built around high school curriculum and summer camps. Continuing this trend, we developed a curriculum for a summer camp aimed at middle school students. In this paper we describe the four-part curriculum and how it is presented. The results from the first run show a net increase in student interest in computer science.

1 Introduction

It can be challenging to get students with the aptitude for computer science to pursue a major in computer science out of high school [3]. Different attempts have investigated ways to increase interest in computer science [3, 4, 5]. Some research has investigated building or modifying high school computer science programs [3], while others have built short run summer camps to introduce computer science concepts to high school or middle school students [4, 5, 6]. Despite differing opinions on where to start, the research seems to agree that exposure to simple concepts at an early age increases the interest in computer science. Many students get to college and do not know what to expect from the major and early exposure helps align their expectations before getting to college [3].

A big field in computer science currently is cyber security. The National Security Agency (NSA) and National Science Foundation (NSF) co-fund summer camps targeted to middle school and high school aged students called GenCyber. Beyond needing increased interest in computer science, this summer camp has the mission statement "Our vision is for the GenCyber program to be part of the solution to the Nation's shortfall of skilled cybersecurity professionals" [2]. The need for exposure of cyber security concepts to younger students is how to get them interested in a career path aligning with this mission statement.

Our college of Science and Mathematics has a summer camp program put on by our Science, Technology, Engineering, and Mathematics (STEM) center called Investigations into Mathematics and Science (IMAS). The goal of this camp is to expose middle school students to different STEM related fields and serve as a path for students to major in one of these fields who may not have thought about majoring in them before. This past summer we developed a curriculum around exposing the students to simple computer science concepts and basic encryption techniques. The goal of this curriculum was to use encryption to show why computer science is necessary and how we use computers to solve cyber security related problems.

This paper outlines the curriculum we used for the summer camp and shows the results from our initial offering. The goal is to share our experience and information with others in building computer science programs to grab the interest of middle school students.

2 Developed Curriculum

During the week we instructed four groups of students totaling 46 students. In this cohort we had one group of 6th graders, one group of 7th graders, and two groups of 8th graders. Our goal with the curriculum was to introduce these students to an aspect of computer science outside of commercial software development. This module takes the students from learning binary counting to exclusive-or (XOR) encryption. The course is taught over a four-hour period one day and is taught four times throughout the week with a different group each day.

Students were provided with the materials necessary to complete the course. This included instructional handouts, exercises, and a laptop preloaded with all of the necessary files and software we developed for the course. The students performed some tasks by hand to reinforce the concepts before using the laptop to solve more complex problems. The students worked in pairs to allow for them to experience problem solving together.

2.1 Part One - Binary Counting

The first step of implementing the curriculum is introducing the students to the concept of binary numbers. For this part we allowed the students to perform a hands-on task in learning how to count in binary. There are various approaches to facilitate this [1]. For this course, the students were given 50 pennies to be used in learning how to group numbers. The students are presented with an example of taking the 50 pennies and splitting them into groups to get 27 total pennies. 1 shows the arrangement.

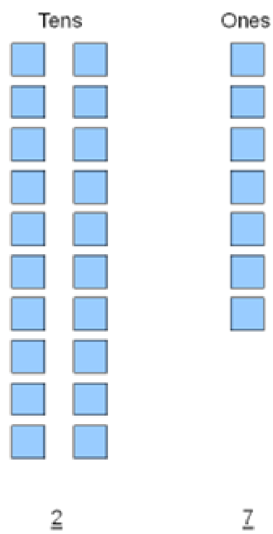


Figure 1: 27 units in base 10.

The students see that by having 2 groups of 10, and 1 group of 7 we get a total of 27 pennies. To show how binary works we had the students take the 27 pennies and group them in groups of 16, 8, 4, 2, and 1. 2 below shows the unit configuration.

The students were given a series of cards to help with counting in binary. The cards had dots on them corresponding to the powers of 2. By using these cards, we gave the students a series of number in decimal and they converted

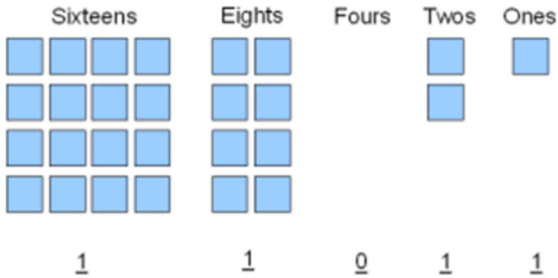


Figure 2: 27 units in base 2.

to binary. When a card is facing up (with the dots showing) it represents 1 group of that power. When a card is facing down, it represents 0 groups of that power. The students can verify that this is correct by counting the total dots showing. This activity gave the students a chance to convert numbers between binary and decimal getting use to the idea of the same information being represented in different ways. 3 below sows the cards.

Figure 3: 27 in binary using the cards.

2.2 Part Two - XOR Encryption

With an understanding of binary numbers, the students are then taken through the concept of encryption. We start with a simple Caesar Cipher to illustrate the basic concepts of symmetric encryption: a shared secret (key) and a two way process (cipher). By applying a known cipher to an unencrypted message (plaintext) with a specific key, an encrypted message (ciphertext) will be produced. By applying the same cipher and the same key to that ciphertext, the original unencrypted message will be revealed. The next step is showing the students an XOR cipher. The logical rules for an XOR operation are provided along with multiple examples applying these rules to two binary numbers to produce a new value. The students then have to reverse the process to produce the original number, further solidifying the concepts of ciphers and keys.

Students were provided with an ASCII table and shown how letters are represented using numbers. Using their new skills in converting decimal to binary, they are given an encrypted message and must use the ASCII table to get the underlying decimal values. They then convert the decimal values to binary. From there, the students apply the XOR cipher to the provided ciphertext in binary and convert it back to the original plaintext.

2.3 Part Three – Python Encryption

The next step is to give the students a brief introduction to programming. We developed Python scripts to do bitwise XOR encryption. The students used these scripts to encrypt and decrypt different files. The students can see how the same encryption process they just performed by hand can be automated by a computer through programming. The students are then allowed to create text files and encrypt them. They can see how the text in the files changes and using the same scripts and key they can decrypt the files to get the original plain text file back.

2.4 Part Four – Forensics Case

The last part of the curriculum takes all the pieces and gives the students an exercise to test their skills. The students are given a series of text files encrypted using the same scripts given to them in the previous part. They are given a write up of a fake crime of someone placing credit card skimmers around the town. The files are text files found on a thumb drive from the suspect, but the files need to be decrypted to find out what happened. The students have the knowledge and tools to decrypt the ciphertext, but they need to determine the keys. We give the students one more script that can look at a file and give a list of possible keys. Using the resulting list, the students then try out the keys to decrypt a file until they can read the plain text. The students use all the skills learned in the earlier portion to decrypt the files and put together a timeline of events.

3 Student Experience

The overall summer experience was successful. We had positive reactions from the students and were encouraged to do this camp again in the future. In this section we will describe what we noticed among the different age groups and the reactions of the students. We also have survey data to quantify the opinions of the students and will also be detailed in this section.

3.1 In Class Feedback During Instruction

In general, the students did quite well in the first section, but it was hard keeping the attention of the ones who seemed to grasp it faster than others. Binary counting was not the favorite of the students, but they soon realized how important it is when we moved on to the encryption. The students enjoyed seeing how using a key to XOR a value to a cipher value could be taken back using the same operation. Once we moved to using the Python scripts the students actively became interested. They really enjoyed being able to perform the same encryption they were previously doing by hand with a computer. This excitement grew more once we gave them time to work on the police case, we gave them. The consensus of the students was that the case portion was the most fun part of the whole curriculum. When asked, the students responded by saying it was fun because they got to decrypt files and used the information to "solve a mystery". One student even admitted to casually going through the curriculum until he hit the case at the end. This same student became passionate about solving the crime using the skills learned in the earlier part.

3.2 Quantitative Survey Results

As part of our STEM center's summer camps, they include a pre and post survey to help with improvement of the summer camps. For this we looked at the responses for the increased interest in computer science or cyber security.

The summer camp had 46 participants who made it through the whole week and completed both the pre and post surveys with to test interest in the STEM fields. While the survey dealt with all STEM fields, we were interested in the two that asked about interest in Computer Science and Technology, and Engineering. The results are in the tables below.

Table 1: Pre and Post survey data on interest in computer science

	Not at all interested	Not so interested	Interested	Very Interested
Pre Survey	8	5	18	15
Post Survey	7	3	20	16

Table 2: Pre and Post survey data collection into interest and non-interest

	Interested	Not interested
Pre Survey	33	12
Post Survey	36	10

The question in the survey asked was “How interested are you in a career in Computer Science?”. The trend was a net increase in computer science interest. The overall net was a gain in 3 students out of 46.

Table 3: Overall results for interest in technology and engineering

	Strongly Agree	Agree	Disagree	Strongly Agree
I am more interested in Technology	8	5	18	15
I am more interested in Engineering	7	3	20	16

Table 4: Interest in technology and engineering percentage

	Agree	Disagree
I am more interested in Technology	74%	26%
I am more interested in Engineering	80%	20%

While the summer camp had many modules that all contribute, we did get an increase in the increased interest in technology with 74% of all the respondents saying they agreed (and strongly agreed) with that statement. This reflected similar responses we received during the instruction of the module.

3.3 Results Analysis

As table 1 above shows there was a small net gain of students who had an increase in interest in computer science. In the results there was actually a decrease of 5 students who went from being interested to not being interested in some form. With this change there was a total of 8 new students who were not originally interested to being interested. The aggregated information in the table also treats as ‘Not at all interested’ and ‘Not so interested’ as one group while the other two are one group. Within the ‘Interested’ group there were 4 students who originally said they were interested then later responded with ‘Very interested’. The table treated that as no change but there is a slight change in that question. Conversely there were only 2 who went from ‘Not so interested’ to ‘Not at all interested’.

The camp is targeted at students who have an interest in STEM activity and with those students we had 8 students go from ‘not interested’ to ‘interested’ with 4 more having an increase in interest level who previously were interested. The overall trend was a net increase in student interest in computer science after the camp.

4 Challenges

The initial curriculum we developed contained more content than could reasonably be covered in the allotted time. As the week progressed, the curriculum was adjusted to better fit the camp duration as well as the prior knowledge of the students. Part of the reason some students struggled was simply a lack of familiarity with basic computer usage (opening directories, creating files, file system navigation, etc...). To save some time, we would log into the provided laptops during a break and setup the environment for the upcoming tasks.

Another challenge we experienced was the disparity between the age groups. As mentioned previously, we had two groups of 8th graders. These groups were the hardest ones to make stay focused on the tasks. This could be that the module was not challenging enough. The 7th and 6th grade groups enjoyed the module the most. These two groups stayed on task and seemed genuinely excited when they broke the encryption on the files and got to build the timeline of case events for the crime they were investigating.

In the future, we expect to add additional challenges for each module to help the more advanced students continue progressing.

5 Conclusions

Our first attempt at this summer camp was a positive experience. We set out to expose young students to computer science through simple cyber security concepts. We were able to get positive feedback from all the students expressing interests in computer science and they left with a better understanding of what is possible in computer science. The results of our survey show a net increase in the number of students interested in computer science after taking the course. We will continue to improve this module for future IMAS summer camps and use this as the basis for a week long summer camp for computer science only as a future GenCyber summer camp.

Acknowledgements

We thank Jana Redfield, Adrienne Aul, and Nola Schmidt from the STEM Research and Learning Center for their help in preparing and refining this summer camp and for helping with the data collection process.

References

[1] Computer science without a computer. https://csunplugged.org/en/.

[2] Inspiring the next generation of cyber stars. https://www.gen-cyber.com/about/.

[3] L. Carter. Why students with an apparent aptitude for computer science don't choose to major in computer science. *ACM SIGCSE Bulletin*, 38(1):27–31, 2006.

[4] D. Dampier, K. Kelly, and K. Carr. Increasing participation of women in cyber security. *ASEE-SE Regional Conference, Starkville, MS*, 2012.

[5] G. Jin, M. Tu, T. H. Kim, J. Heffron, and J. White. PNW GenCyber summer camp: Game based cybersecurity education for high school students. *Mid-Atlantic ASEE Conference, Penn State*, 2017.

[6] B. R. Payne, T. Abegaz, and K. Antonia. Planning and implementing a successful NSA-NSF gencyber summer cyber academy. *Journal of Cybersecurity Education, Research and Practice*, 3, 2016.

Light-Weight Student-Driven Workshops for Positive Attitude Change towards Programming in Early College*

Wei Jin, Xin Xu, Sonal Dekhane
Information Technology
Georgia Gwinnett College
Lawrenceville, GA 30043
{wjin,xxu,sdekhane}@ggc.edu

Abstract

In Fall 2018, we observed that ~33% of early-college participants of game-development outreach workshops became interested in taking a programming course. In Spring 2019, a new group of near-peer tutors developed new games and conducted new workshops in a similar fashion. Effectiveness of the spring workshops was almost the same as that for fall. These workshops were developed and conducted by upper-level students in a service-learning class for participants in lower-level classes. They were light-weight, ~75 minutes long. Early college presents a unique opportunity for outreach, but this has not been explored adequately in previous work. In this paper, we present analysis of the outreach workshops' effectiveness and characteristics of the workshops that may have contributed to participants' positive attitude change. We found that the attitude change is strongly associated with participants' gender and declared majors, while not with their race and ethnicity. We also found that the technologies/platforms chosen by the peer tutors and their ease of use had an impact on the effectiveness of the workshops.

1 Introduction

In an effort to recruit and retain students in computing, especially those who are under-represented, Technology Ambassadors Program (TAP) was created on our campus. This dual-purpose program includes Information Technology (IT) majors in a service-learning elective course. As part of the course, students conduct outreach workshops to middle and high school girls and college students, and in the process of achieving this, improve their own technical, communication and leadership skills.

In an earlier paper [5], we discussed the program's impact on the TAP students' confidence and persistence in computing. In this paper, we attempt to assess the impact of the outreach workshops on workshop participants, who are usually early-college students enrolled in our general education information literacy class. Research showed that students often have stereotypical views of the IT discipline as being boring, solitary and lacking real-world context [20]. These perceptions could be changed by different outreach programs [11, 13, 18, 19]. We hope that our game-based workshops conducted by peer tutors can help students overcome these negative perceptions. Our goal is, however, more focused due to the nature and duration of the workshops. It is to help participants to have positive changes of attitudes towards programming through fun and engaging activities.

Most of our workshop participants share similar characteristics. Many are traditional incoming freshmen, recently graduated from high school. Some of them are still undecided about their majors; some are not certain about or do not really understand their chosen fields. Even those who have made up their mind about their majors could still benefit from knowledge of programming. Programming skills could improve a person's career competitiveness. For example, coding can automate repetitive tasks in business, data analysis, arts and design, science and engineering.

2 Related Work

We will address the related work from two perspectives: (1) the workshop audience and the reason why we believe early college provides a good opportunity for outreach, and (2) the program structure and sustainability.

2.1 Target Audience

The majority of the previously reported outreach activities targeted middle/high school students. A systematic literature review of ACM and IEEE journals and conference proceedings from 2009-2014 shows that there are 101 relevant articles, and 82% of such efforts aimed at middle/high school students [17]. The assessment of the outreach efforts usually focuses on how much students learned in programming or developed computational thinking [1, 10, 6] and how such efforts changed their attitude/perception towards computing (e.g. computing is fun) [9, 13, 16, 18, 19].

The uniqueness of our work is in providing such outreach activity to early college students. As described below, the majority of early college students do not have experience with programming and for those who had experience earlier on. The impact might have waned when they reach college [15]. Researchers of [17] also surveyed more than 700 students from three college-level institutions. The results showed that 39.1% students reported that they had participated in computing activities such as programming, games, hardware, robotics in their middle/high school years, which means more than 60% of the college students they surveyed did not have experience with such activities. This means that there is still ample room for outreach in early college for students.

Another advantage of outreach activities targeting early-college students on campus is that the participants will have opportunity to take immediate action in actually taking a programming class. Since our workshops are usually offered during the last month of a semester, students have the option to enroll in a general-education programming course or a more rigid programming course designed for IT majors for the following semester.

2.2 Structure and Sustainability

In this subsection, we compare the structure of our program with other works. There is a rich collection of outreach programs [1, 2, 3, 4, 6, 7, 8, 9, 10, 12, 13, 14, 16, 18, 19, 20], the majority of which targeted middle/high school students. These outreach programs were usually planned by faculty members, and the activities were also created and led by faculty. In our case, faculty only did the planning, but workshops and demos were developed and led by college students. In addition, the outreach activities are a part of the curriculum of the service-learning course. There is a set of learning objectives and outcomes tied to outreach (e.g. communication and leadership skills). Students know what to expect and faculty can introduce incremental improvements.

The program in [14] is most similar to ours. They have a well-organized service-learning course, where students developed workshops and used the workshops for outreach. The education school faculty taught students pedagogy and then students designed and taught workshops that introduce CS concepts to high schoolers, who in the post-survey expressed increased knowledge of computing or computational thinking and more confidence about computing. The challenge they had was the recruitment of service-learning students and eventually the service-learning course was stopped.

In comparison, our service-learning course is project-based and completely student-driven, from choosing a game development environment, to project ideas, to coding, to workshop content. Students are highly motivated and the interest in the course is high. In addition, we do outreach where we are, on campus in a general-education IT course for early-college students. Most of the participants recently graduated from high school, so they were similar to the participants in [14]. The relatively low overhead in scheduling on-campus workshops makes our efforts manageable and sustainable and we can reach a large number of participants, 150-200 students per semester.

3 Description

Technology Ambassadors Program (TAP) originally began in 2012 with funding from NSF sponsored STARS Computing Corps. Since its inception, the format of the program has changed from an extracurricular program to a service-learning course, making it more sustainable. However, the goals remain the same. TAP is an effort to broaden participation in computing, with emphasis on both recruitment and retention of underrepresented minorities.

Since Spring 2016 the program has been offered as an upper-level service learning elective course. This format gives students course credit for participating in the program and the faculty receive teaching credit. Students apply to the program each semester and are interviewed by two faculty teaching the course. Up to 12 students are selected to participate in the program each semester. The purpose of the 3-credit hour elective is to enhance the students' technical, communication and leadership skills. The format of the class is highly non-traditional, putting students in the driver's seat, while the faculty play the role of a facilitator and a mentor. Students work in teams of 3-4 and are responsible for creating an outreach game development project for a non-computing audience. They are encouraged to use technologies and choose ideas that would be appealing to their audience. We strive to recruit female students into the TAP program and have at least one female student in each team.

Each team investigates at least three technologies at the beginning of the semester. The instructors introduce the teams to technologies used successfully in the past. The teams can also investigate or use other technologies. Teams present their technology choices and project ideas to the rest of the class. They make their final decision based on their research and feedback from peers and instructors.

Along with the game development project, each team also creates a workshop that can be presented at outreach events. The workshop allows the audience to modify the existing project and/or create a similar project. The goal is to make the audience aware of the wide array of possibilities using technology, with the hope that the audience develops an increased awareness/interest in programming. Before the teams conduct their workshops, they first demonstrate their projects to middle/high school girls at our on-campus STEM outreach event, Super Saturday Series, each semester. The demonstrations are short and allow the girls to interact with and make minor changes to the code and see its impact on the result. This turns out to be a popular activity among attendees.

The workshops, on the other hand, are longer and are geared towards our early-college population. These workshops are conducted in our Introduction to Computing course. This is a general education course required for all students at the college and thus has a large number of sections where the workshops can be conducted. Our college being a majority-minority institution, this class also offers opportunities to reach out to underrepresented minorities. All sections have the same curriculum, including the same textbooks and very similar schedules. This makes it easy to schedule the workshops towards the end of the semester. Each team conducts their workshop in three sections of this class. The workshop lasts ~75 minutes. At the end of the workshop, the participants complete a survey reporting on the impact of the workshop on their interest in programming.

4 Evaluation

The goal of the workshops is to generate positive impact on students' attitude towards programming. Therefore, besides the questions to evaluate the workshops themselves, the survey specifically asks the participants to reflect whether the workshops have caused them to reconsider taking programming classes.

We observed in Fall 2018 that ~33% of the 204 participants said that they either made up their mind to take a programming class or started to consider taking a programming class due to the workshop experience. Excluding the 54 students who already planned to take a programming class before the workshops, the percentage of students who had positive attitude change was ~45%. We were not sure whether it was due to the group of great service learning students in the fall and were interested in whether the new group of service learning students in Spring 2019 could achieve something similar and if so what might be the reasons for the effectiveness.

Table 1 contains the number of participants who responded to the surveys and the number of students in each of the following groups for both semesters:

- *Already group*: students who already planned to take a programming class before the workshop.
- *Positive Attitude Change group*: students who had either one of the following changes
  - from undecided to decided to take a programming class
  - from not considering to considering to take a programming class.

- *Still Not-Going-to group*: students who remained convinced they would not take a programming class after the workshop.

Table 1: Number of Participants and Their Attitude for Taking a Programming Class

Group Change Group to Group	Already Positive Attitude Still Not Going		
Fall 2018 (N=204)	54	67	83
Spring 2019 (N=158)	20	59	79

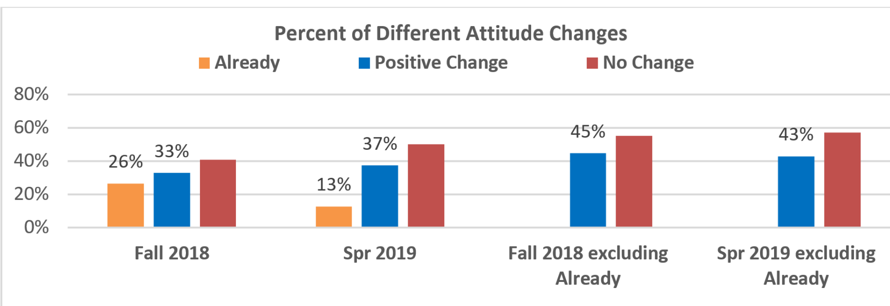


Figure 1: Attitude Changes of Fall'18 and Spring'19 Workshop Participants

Figure 1 shows the comparison of Fall 2018 and Spring 2019. The two left bar clusters show the different percentages for each group for the fall and spring. The difference between the fall and spring regarding students' original plan for taking a programming class is outside our control. With the *Already group* excluded, the two right clusters look very similar: ~45% students had a change of attitude in the fall and ~43% in the spring. The structure of our program may have some innate characteristics that help achieve this effectiveness.

Below we will first examine the effectiveness for different demographic groups and then study the possible contributing factors to the workshop effectiveness. Unless noted, the *Already group* is excluded from the data set.

4.1 Effectiveness for Demographic Groups

(1) Gender: It is a well-known fact that women are under-represented in the IT field, so it is not surprising that only ~13% of the women had already planned to take a programming class before the workshops, while over 28% of the men had already planned so. The ratio of the female rate to the male rate is 13/28, or ~46%. With the *Already group* excluded from the data set, women have positive attitude change ratio 38% vs. 54% for men. The difference between two genders is statistically significant ($p = 0.007$) according to Fisher's exact test. Even though the effectiveness for women is less than satisfactory, we see some hope. The ratio of women's attitude change rate to that of men's is 38/54, ~72%, 1.54 times that of the *Already group*'s ratio 46%.

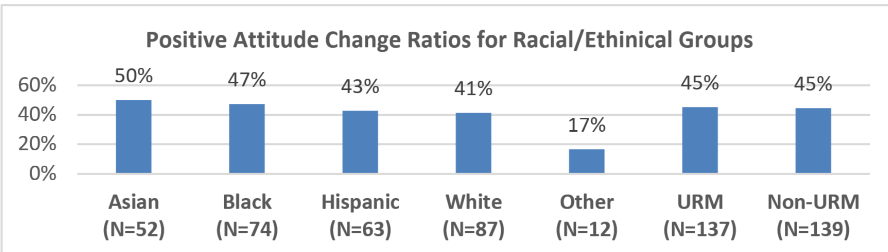


Figure 2: Workshop Effectiveness for Different Racial/Ethnic Groups

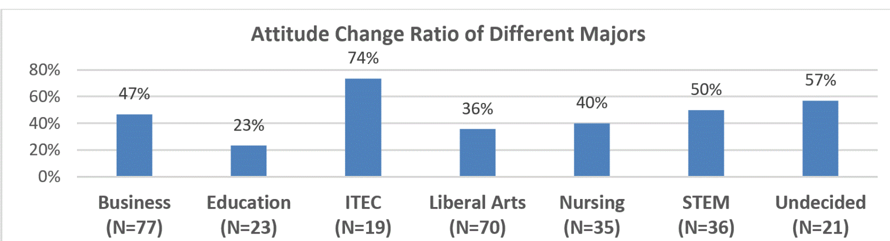


Figure 3: Workshop Effectiveness for Different Majors

(2) Race/Ethnicity: Figure 2 shows the attitude change ratios for major racial/ethnic groups. As shown by the last two columns, there is almost no difference between the ratios for the Under-Represented Minority group (URM) (Black and Hispanic) and the non-URM group (Asian and White).

(3) Declared Major Figure 3 shows that students of different majors became interested in taking a programming class. It is interesting to see that not all IT majors planned to take a programming class before the workshop and there were still IT students not planning to take a programming class after the workshop. This validates what we stated earlier that not all students understand their majors very well. There is also a fear of programming among students, and some of our students delay taking the class because they do not understand the significance of taking this fundamental class (pre-requisite to other classes) in a timely manner. The difference among majors is statistically significant ($p = 0.011$). The attitude change rate for IT students is very high, around 70% and it is 57%, 50%, and 47% for undecided, STEM, and business majors respectively. Students in these majors are those who could really benefit from the knowledge of programming.

4.2 Factors that May Impact the Effectiveness

In this subsection, we present the analysis on what factors might be the most impactful. Each semester, there were four different workshops designed and delivered by four different teams. Each workshop was normally taught three times in three different sections of the same general-education class. Table 2 shows the number of participants in each team's workshops. The analysis in this subsection is based on the data set with the *Already group* excluded.

Table 2: # Participants in Each Team's Workshops

	Fall 2018				Spring 2019			
Team No.	1	2	3	4	1	2	3	4
Total # Participants	51	35	64	54	29	58	38	33
# Participants with *Already Group* Excluded	34	31	48	37	22	60	36	30

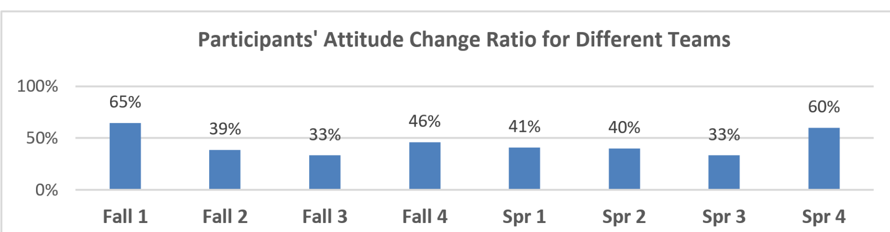


Figure 4: Team Effectiveness

Figure 4 shows each team's effectiveness. Two teams stood out: the attitude change ratio for them is ~60%, while it is 30-40% for the other teams. A team's effectiveness can be determined by many factors and we will study each of the factors below.

Table 3 lists seven characteristics that we thought might be relevant. According to data, "whether a team has URM members" has no significant impact

Table 3: Team/Workshop Characteristics

Characteristics	**Measurement**
Whether a team had female	1 for yes; 0 for no
Whether a team had URM	1 for yes; 0 for no
Observed team synergy	1 for yes; 0 for no
Game development platform	2 for easy; 1 for medium; 0 for hard
Used robots or not	1 for yes; 0 for no
Related mini-game developed	1 for yes; 0 for no/unrelated mini-game
Overall workshop experience	1-5 for various measures

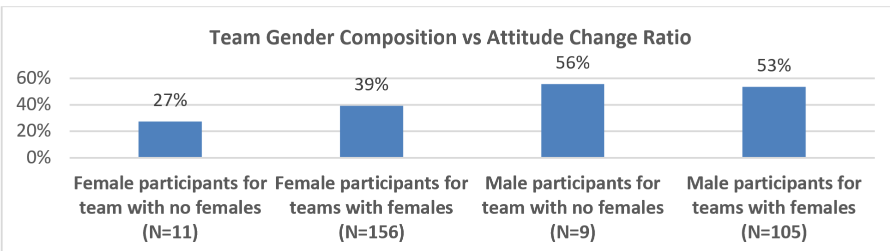


Figure 5: Team Gender Composition vs Attitude Change for Different Genders

on the attitude change ratios for URM participants or non-URM participants and "the observed team synergy by the instructors during the development phase" does not make much difference in the workshop effectiveness. Other characteristics do seem to have some relevance and we study them below.

(1) Whether a Team Had Female Members: The participants' attitude change ratio for teams with female tutors and that for the one team with no female tutors is roughly the same 43%-44%. Interestingly, however, a team's gender composition does seem to affect attitude change ratios for female and male students separately. The two left columns in Figure 5 show that teams with female tutors are more effective for female students than the one team with no female tutors. The two right columns show the opposite for males: the one team with no female tutors are more effective for male students than teams with female tutors. More data are needed to better observe the gender factor of the peer tutors in the future.

(2) Game Development Platform: Figure 6 shows the effectiveness for teams whose chosen game development platforms were either hard, medium, or easy for participants to start coding in. This measures whether the platform requires tedious setup procedure and/or programming language is complex. We can see that the easier the environment, the higher the attitude change ratio. The difference is statistically significant ($p = 0.025$).

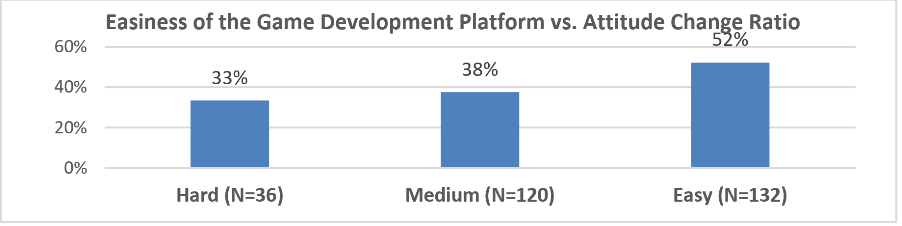


Figure 6: Easiness of Game Development Platform vs Attitude Change Ratios

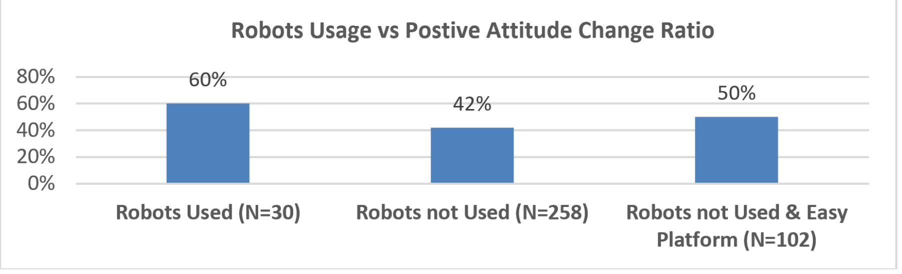


Figure 7: Robots Usage in Relation to the Attitude Change Ratio

(3) Robot Usage: The two left columns in Figure 7 show that workshops that used robots had higher attitude change ratio. The difference is marginally significant ($p = 0.079$). The workshops that used robots were considered an easy platform in Figure 6. The leftmost and rightmost columns in Figure 7 show that the workshops using robots are still more effective than the rest of the easy-platform workshops (not statistically significant). The “cool” factor of using robots may have helped make the workshops more engaging and thus more effective. Since only one team used robots, the number of participants in the robot usage workshops is relatively low. We will collect more data for robot use in the future.

(4) Relevant Mini-game Development: In addition to playing and modifying the game that the peer tutors developed and used for demo, if participants also developed a mini-version of the game, their attitude change ratio was higher (47% of 181 participants) than that for workshops where either no mini-game was developed or the mini-game developed was not related to the demo game (38% of 107). Even though the difference is not statistically significant ($p = 0.177$), this interesting observation gives us significant insights for future efforts.

(5) Overall Workshop Experience: Not surprisingly, how well a workshop was conducted is statistically significantly related to the attitude change:

- Overall workshop experience (with ratings 1-5) ($p=0.002$)
- Presentation engagement (with ratings 1-5) ($p=0.002$)
- Whether learned something new (yes or no) ($p = 0.037$)

5 Conclusion and Future Work

We are very encouraged by what the data say about our outreach program for early college students. The workshops are equally effective for under-represented racial/ethnic minorities and the non-under-represented groups. Even though the effectiveness for female students is lower than that for male, the ratio of female's attitude change rate (with the *Already group* excluded) vs male's is much higher than the female to male ratio among students who have already planned to take a programming class.

We also identified factors that are most likely to have an impact on how effective a workshop is. The most prominent factor is the game development platform. If it is simple to setup and code in, the workshops have higher effectiveness. Using robots and developing mini-versions of the games that are used in the demo are also likely to have a positive impact. Even though more data is needed, this is nonetheless very valuable for our future research.

We want to understand more about the gender factor for both the peer tutors and the participants. In addition, we believe that near-peer mentoring is a big motivating factor and we plan to study this in more depth. Last, we will track how many of the participants who had positive attitude change will actually enroll in and complete a programming course.

References

[1] Karen Brennan and Mitchel Resnick. New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 Annual Meeting of the American Educational Research Association*, Vol. 1, 2012.

[2] Orpheus Crutchfield, Christopher Harrison, Guy Haas, Daniel García, Sheila Humphreys, Colleen Lewis, and Peter Khooshabeh. Berkeley foun-

dation for opportunities in information technology: A decade of broadening participation. *ACM Transactions on Computing Education*, 11:15, 10 2011.

[3] T. Dahlberg, T. Barnes, K. Buch, and A. Rorrer. The stars alliance: Viable strategies for broadening participation in computing. *ACM Transactions on Computing Education*, 11(3):18, 2011.

[4] A. Decker and M. McGill. Pre-college computing outreach research: Towards improving the practice. In *Proceedings of the 48th SIGCSE Technical Symposium on Computer Science Education*, pages 153–158. ACM, 2017.

[5] S. Dekhane, X. Xu, N. Napier, R. Barakat, C. Gunay, and K. Nagel. Technology focused service-learning course to increase confidence and persistence in computing. *Journal of Computing Sciences in Colleges*, 34(2):147–153, 2018.

[6] P. Doerschuk, J. Liu, and J. Mann. Inspired high school computing academies. *ACM Transactions on Computing Education*, 11(2), 2011.

[7] Peggy Doerschuk, Jiangjiang Lius, and Judith Mann. Inspired broadening participation in computing: Most successful strategies and lessons learned. In *IEEE Frontiers in Education Conference (FIE)*. IEEE, 2010.

[8] M.A.L. Egan and T. Lederman. The impact of impact: assessing students' perceptions after a day of computer exploration. In *Proceedings of the 16th Annual Joint Conference on Innovation and Technology in Computer Science Education*, pages 318–322. ACM, 2011.

[9] Brendan Tangney et al. Pedagogy and processes for a computer programming outreach workshop—the bridge to college model. *IEEE Transactions on Education*, 53(1):53–60, 2009.

[10] Diana Franklin et al. Assessment of computer science learning in a scratch-based outreach program. In *Proceedings of the 44th Technical Symposium on Computer Science Education*. ACM, 2013.

[11] S. Grover, R. Pea, and S. Cooper. Remedying misperceptions of computer science among middle schoolstudents. In *Proceedings of the 45th SIGCSE Technical Symposium on Computer Science Education*, 2014.

[12] M. Guzdial, B. Ericson, T. Mcklin, and S. Engelman. Georgia computes! an intervention in a us state, with formal and informal education in a policy context. *ACM Transactions on Computing Education*, 14(2):13, 2014.

[13] A. Harriger, A.J. Magana, and R. Lovan. Identifying the impact of the spirit program in student knowledge, attitudes, and perceptions toward computing careers. In *Frontiers in Education Conference*, 2012.

[14] Yasmin Kafai, Jean Griffin, and et. al. A cascading mentoring pedagogy in a cs service learning course to broaden participation and perceptions. In *Proceedings of the 44th SIGCSE Technical Symposium on Computer Science Education*, pages 101–106. ACM, 2013.

[15] Chad Mano, Vicki Allan, and Donald Cooley. Effective in-class activities for middle school outreach programs. In *IEEE Frontiers in Education Conference (FIE)*. IEEE, 2010.

[16] G. Marcu, S.J. Kaufman, J.K. Lee, and et al. Design and evaluation of a computer science and engineering course for middle school girls. In *Proceedings of the 41st SIGCSE Technical Symposium on Computer Science Education*, pages 234–238, 2010.

[17] Monica McGill, Adrienne Decker, and Amber Settle. Undergraduate students' perceptions of the impact of pre-college computing activities on choices of major. *ACM Transactions on Computing Education*, 16(4), 2016.

[18] J. Robertson. The influence of a game-making project on male and female learners' attitudes to computing. *Computer Science Education*, 23(1):58–83, 2013.

[19] R. Taub, M. Armoni, and M. Ben-Ari. CS unplugged and middle-school students' views, attitudes, and intentions regarding CS. *ACM Transactions on Computing Education*, 12(2), 2012.

[20] S. Yardi and A. Bruckman. What is computing?: bridging the gap between teenagers' perceptions and graduate students' experiences. In *Proceedings of the 3rd International Workshop on Computing Education Research*, pages 39–50. ACM, 2007.

Positive Cycle of Integrating Teaching and Research: Machine Learning Self-Driving Car*

Heejun Choi, Travis Teague, Tanner Luce
Computer Science, Engineering and Physics Department
University of Mary Hardin-Baylor
Belton, TX 76513
hchoi@umhb.edu, {tateague,tdluce}@mail.umhb.edu

Abstract

When the computer science courses are designed, how to make the students understand the core concepts and algorithms is usually considered significantly. However, applying them to the real fields, which is considered more important, is often skipped. This research project demonstrates the example on how the positive cycle of integrating teaching inside the class and research outside of class performs successfully. In addition, it shows the detailed process on the steps that students take to build the self-driving car based on the Machine learning technology. The research project is conducted in 3 stages. First, an RC car is assembled with mechanical and electronic components (Raspberry Pi, sensors, etc.) and is tested by the signal measurement. Then, the self-driving software modules using machine learning are developed and tested. Finally, the RC car with self-driving software module is tested in terms of the data loss and real driving on the lane.

1 Introduction

This research project is a practical but good one because it includes various theoretical knowledge that is learned in the class. The purpose of this project is to build the self-driving RC car controlled by Machine learning and Big data technologies that have been taught in the big data class. The project requires numerous technologies, such as Machine learning, Big data, Robotics, Computer Vision, Electronic optical sensors and so on. Through this project, the students were able to acquire a variety of cutting-edge technologies as well as apply the knowledge acquired from the previous coursework to the project.

2 Literature review

2.1 Research and development in the car and IT industries

Self-driving car research requires various technologies such as electronics, sensors, machinery, and Machine learning (or Artificial intelligence), with its ultimate goal being the development of a car that can drive itself autonomously without a human driver. As such, it is car makers in industry that are leading the development of self-driving cars, which show a great potential for future commercialization. Toyota is currently developing a self-driving car, dubbed "the most powerful and intelligent supercomputer on wheels," that will be on major highways by 2020[5]. For this project, the company cites the importance of slashing the cost of technology and hiring proficient software engineers. Furthermore, Toyota plans to invest 2.8 billion in a new company that develops software for self-driving cars. Although Ford lagged behind the competition with respect to developing self-driving cars, its adoption of Argo AI (Artificial Intelligence) successfully demonstrated the potential for self-driving cars to replace human delivery drivers working for companies such as Walmart and Postmates. However, though the area of self-driving cars promises both technological possibility and a future market, there remain numerous technical problems that must be overcome before self-driving cars can operate without a human driver in difficult road conditions. For instance, it is not an easy task to develop software that can both utilize a sensing system to detect rapidly appearing objects and execute the swift decision for the vehicle to avoid it. This is because self-driving cars, unlike regular computers, are real-time computers with which one error can take the life of a human passenger. Cruise, a company under General Motors, planned to release a self-driving car to the market in 2019, but technical issues necessitated a delay in its commercialization.
Completely autonomous cars cannot be made with machinery, electronics, sensors, and other technologies found in conventional cars. Instead, they require a multitude of additional technologies such as AI and Big data. IT companies such as Google and Amazon are actively investing in core technologies that are seen as the future, with Artificial intelligence and Big data being the two representative ones. When AI and Big data are combined, a synergy effect can lead to an output several times greater than when only one of these technologies

is used. Indeed, both technologies are integral to the area of self-driving cars. This may be the main reason why companies such as Google and Amazon are targeting the self-driving car market at such an early stage.
Google began its self-driving car development project in 2009 and has continued to experiment with self-driving cars that can navigate complex city streets with the use of sensors and software rather than a human driver. Google [9] finally began a self-driving car project called WAYMO (Way Forward in Mobile) in 2016, and since 2017, the company has worked with FCA (Fiat Chrysler Automobiles) to develop a fully self-driving Chrysler Pacifica Hybrid minivan. Another large IT company, Amazon [10], decided to invest at least 530 million in a Silicon Valley startup called Aurora. Amazon's shipping costs in 2017 were 21.7 billion, almost twice the cost in 2015. The company believes that utilizing autonomous vehicles is the most effective way to reduce shipping costs. Microsoft [4] does not independently develop self-driving cars but has worked for a long time with auto manufacturers to integrate Microsoft software with self-driving cars. This clearly demonstrates that technologies like AI and Big data are the core technologies necessary for successful commercialization of self-driving cars. LG [14] is currently developing a self-driving car that utilizes Microsoft's AI solution, Azure.

2.2 Academic researches

So far, this has been a review of the current status of self-driving car development by car makers and IT companies. Core technological trends followed by such corporations have greatly impacted the academic curricula taken by students who end up working for these same companies in the future. In particular, fields of study such as computer science and IT attempt to more effectively teach the core technologies of AI (Artificial intelligence) and Big data by linking coursework with outside projects. Efforts such as these have directly led to the continuous publication of many outstanding research papers.
In the past, self-driving car development was conducted by installing cumbersome equipment on conventional cars and testing them on the road, but modern self-driving car research at universities is now mostly done using RC cars. For instance, when developing AI (or Machine learning) software, the brain of self-driving cars, it is entirely possible to use an RC car instead of an actual one [11]. RC cars are 1/8 to 1/16 the size of conventional cars and utilize an electronically operated motor and steering mechanism. By equipping a miniature computer called Raspberry Pi to an RC car, it is possible to both develop and experiment with AI (or Machine learning) and Big data software for self-driving cars in a small campus environment, which is typically limited in funding [12][13]. With respect to the development of AI software as the brain of a RC car, Machine learning is the most widely used method [7].

Simply put, Machine learning, which will be described in greater detail later, involves the use of inputs and outputs to make control signals that can maneuver a vehicle. In this case, the input is a real-time video of road conditions captured by cameras or sensors [2], while the output is the car's change in direction (left/right) or movement (run/stop). Input and output data can be entered into a Machine learning algorithm to create an optimal car control database [6]. Of these machine learning algorithms, the most widely used one is both adept at learning and displays excellent performance. This algorithm consists of a neural network of intricately intertwined neurons found in three layers: the Input, Hidden (or Perceptron), and Output layers [15]. Also, more advanced algorithms, which enhance the structure of a basic neuron network, have been used in the past to improve the accuracy of self-driving cars. A well-known algorithm is convolutional neural network, which is also referred to as deep learning [1]. The CNN (Convolutional Neural Network) algorithm employs a 3x3 sized window (equivalent to a mask in Digital signal processing field) that is applied to an input video in order to enhance accuracy by creating numerous Hidden layers. Currently, CNN is the foundation of the majority of AI software used in self-driving cars, and it has been the target of increased interest in the research community [3].

The most practical core technologies used in software for self-driving cars are the Machine learning algorithm and Big data, which is based on Cloud computing framework. Self-driving cars require a method for collecting and processing large amounts of data in real-time, but a simple standalone computer equipped to a car is severely limited in its capability to do so. One way to solve this issue is by using a Cloud computing platform like Amazon's AWS as a data processing device, which can be used by a self-driving car through a wireless network [8]. Cloud computing systems like AWS can process vast amounts of big data by using thousands or even tens of thousands of server computers. Big data software used in cloud computing systems are composed of ML (Machine learning Library) functions that collects and processes real-time data. Ultimately, AI and Big data technologies will become the core ones necessary for the successful commercialization of self-driving cars in the future. This is why much research on and funding for these technologies are crucial in the academic field.

2.3 Need of Machine learning in Self-driving car

By comparing the input data from video camera and the data (called model) made in the training stage, the control signal for moving self-driving car is made. In fact, the model is used to predict future values based on the past input and output data. There are many ways to create these models, especially, the one used most widely is called Regression analysis, mathematical method.

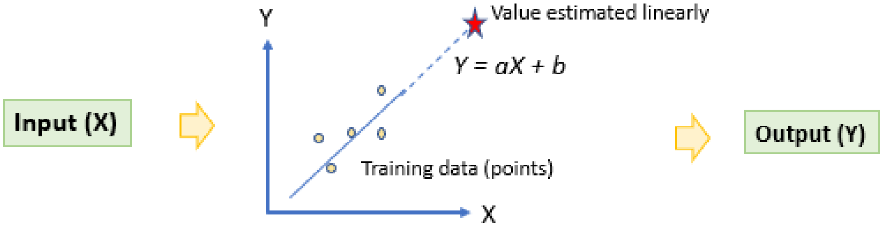


Figure 1: Estimation of values using regression analysis

As seen in Figure 1 above, a straight line is drawn based on the past input (X) and output (Y), and according to this line, an equation can be obtained. This equation is called the mathematical model.

$$Y = aX + b \tag{1}$$

The equation 1 is fit for linear model with multiple input variables and an output variable but is inappropriate for finding nonlinear model, not represented by this straight line. The video image, the input data of self-driving car, is paired with the angle (+/-) and forward/stop signals, the output variables. This data in the self-driving car shows nonlinear patterns that require a new method, Neural network to make a nonlinear model. The Neural network concept is similar to the regression analysis, mathematical method mentioned above, however, it can create a non-linear model with multiple input and output variables. In Figure 2, Neural network consists of 3 layers, such as Input, Perceptron (or Hidden), and Output. Like the Regression analysis, it changes neurons and biases in the Perceptron layer until the difference between the original output (Y´) and estimated output (Y) is minimal. This process is called training.

The equation 2 and 3 represent the making of the Perceptron layer (or Hidden layer) with multiple neurons, which is between Input and Output layers.

$$N_1 = \sigma\left(X_1 W_{11} + X_2 W_{12} + b_1\right) \tag{2}$$

$$N_2 = \sigma\left(X_2 W_{21} + X_2 W_{22} + b_2\right) \tag{3}$$

As seen in the equation 2 and 3, the multiplications of input data (X_1,X_2) and weights (W_{11}, W_{12}, W_{21}, W_{22}) are added with the biases (b_1, b_2) and then this value is passed to the activation functions (σ) to be stored as each intermediate node (N_1, N_2).

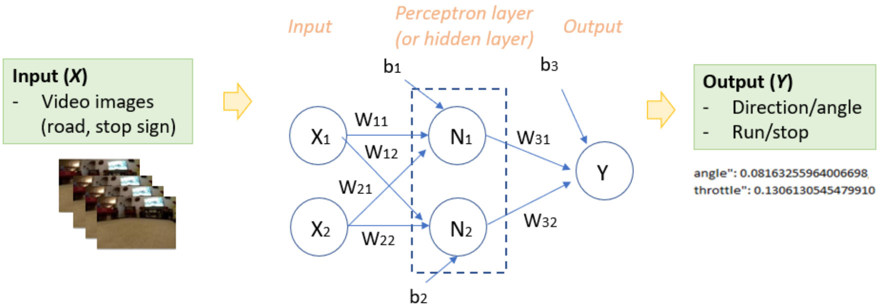


Figure 2: Estimation of values using regression analysis

Afterwards, the multiplications of new input data (N_1, N_2) and weights (W_{31}, W_{32}) are added with the biases (b_3) and then this value is transferred to the activation functions (σ) to make the final result Y as shown in the equation 4.

$$Y = \sigma(N_1 W_{31} + N_2 W_{32} + b_3) \tag{4}$$

3 Overview of system

The components of final products consist of an RC car and remote computer training system that controls and trains the RC car by wireless communication. The RC car can be purchased from Amazon and be moved by the remote controller users handle. However, it is still not the autonomous car that moves by its decision-making system. For its self-driving, a raspberry pi 3 (small computer) with sensors (such as camera and Laser) is put on top of the RC car and is connected to ECU (Electronic Control Unit) that controls car thrust and wheel steering. In addition, it will detect other obstacles, such as passengers, by the camera or laser sensor attached to the RC car.
The remote training system is software developed in the Python language and is installed in the laptop. It controls the car movement through the software modules installed in the minicomputer (Raspberry Pi) in the RC car. At this time, the left/right direction angle information and the video coming from the camera are inputted into the training system. This process is called the machine running training, and once it is trained, the RC car will automatically move, without a driver, through the training data.

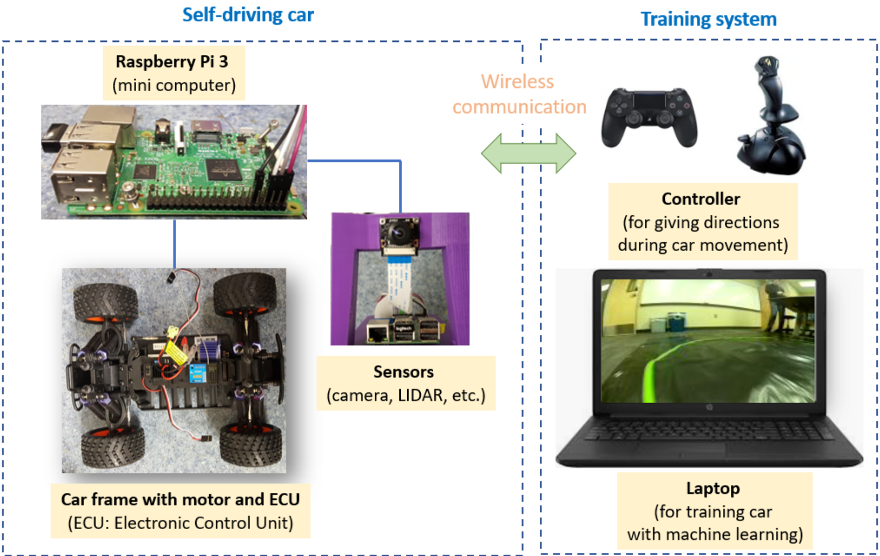


Figure 3: Estimation of values using regression analysis

4 Implementation of System

The project is conducted in the 3 stages. First, an RC car is assembled with mechanical and electronic components (Raspberry, sensors, etc.) and is tested by the signal measurement. Then, the self-driving software modules using Machine learning technology is developed and tested. Finally, the RC car with self-driving software module is tested on the road.

4.1 Hardware configuration

In Figure 4, the motor, ECU (Electronic Control Unit), and wheels are attached to the car frame, and the wheels are steered according to the direction control signal from the ECU. Additionally, the ECU makes the control signal for turning the car left or right and sends them to motor and steering system of wheels. The Raspberry Pi, a mini computer that is placed on the car frame, deploys the software that makes the control signal and then sends them to the ECU in real time. The Raspberry Pi is a kind of computer with a processor and memory and has various input ports, such as camera, GPIO (General Purpose Input Output), and USB. As shown in Figure 4, the GPIO port of the Raspberry Pi is used to connect with the ECU using 4 pins. Besides, the

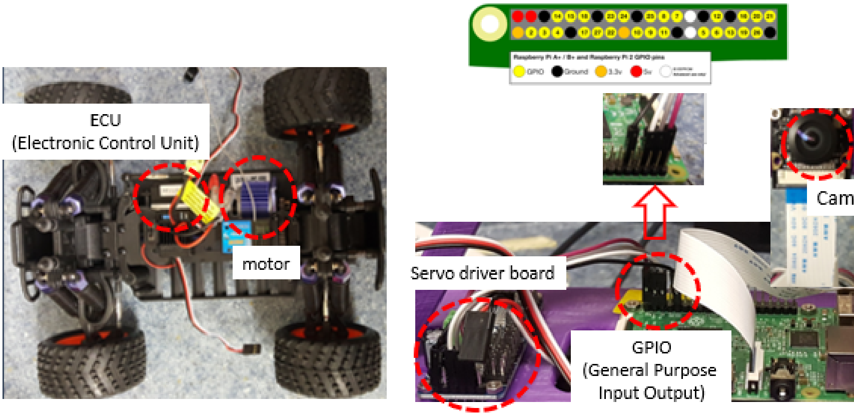


Figure 4: Hardware configuration

camera port is used to connect with the video camera in order to receive input video image. In addition, the Raspberry Pi is powered by 5V external battery.

4.2 4.2 Application of the CNN algorithms to the self-driving car

As mentioned in section 2.3, the prior data for moving self-driving car is a non-linear data with multiple inputs and outputs. To process these types of data, Neural network, one of Machine-learning algorithms, can be useful. However, basic Neural network is not sufficient in making the model for moving self-driving car in the real time and more advanced algorithms from basic Neural network is very needed. Recently, cutting-edge algorithms in the Neural network field have emerged and the representative algorithm is Deep learning.

In fact, the core technology of AI (Artificial Intelligence) recognized as future technology is Deep learning that is the base of CNN (Convolutional Neural Network) algorithm used in the self-driving car project of this article. Deep learning is advanced Neural network algorithm with multiple Perceptron layers but has its drawbacks such as vanishing gradient and calculation overload problems. Vanishing gradient problem causes the gradient to be 0 as the number of layers increases. The gradient descent method produces a huge amount of calculations as it differentiates the whole data during update process. To resolve these problems, SGD (Stochastic Gradient Descent) method and Adam method came out, therefore, Deep learning has revolutionarily improved. Especially, the Adam algorithm has contributed in the enhancement of the accuracy in Deep learning technology. The CNN algorithm is based on the Deep learn-

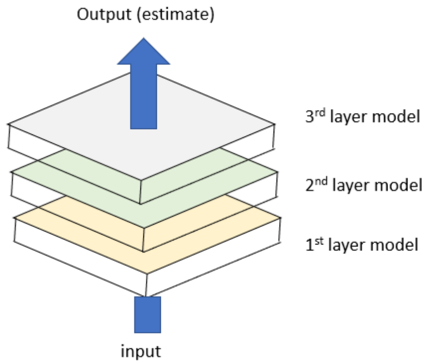


Figure 5: Multi-layers in the Deep learning

ing and has great performance in recognizing the properties of image. In fact, the convolution in the CNN is the concept used in the DSP (Digital Signal Processing) field and uses the mask with specific values. The mask values are multiplied by the pixel values in the original image (like the signal in the DSP) and these products are added together like Figure 6. As a result, this process is called convolution and causes the reduction of image properties, like DSP.

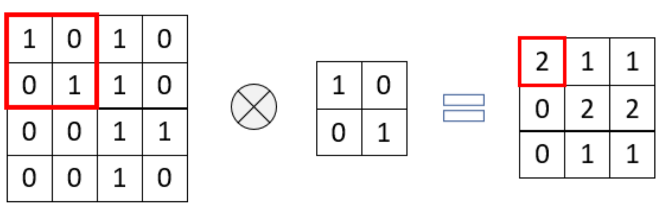


Figure 6: Convolution

The Max pooling, the core algorithm in the CNN, reduces processing time significantly by decreasing the image properties. Figure 7 shows the effect of Max pooling which reduces the image property from a 4x4 to a 2x2. Also, to avoid overfit problem, Dropout technique that deactivates neurons in the Perceptron layer is used in the CNN. Besides this, Softmax normalizes the total of the input values to 1 and each normalized value is binarized as 1 or 0 by using one-hot-encoding technique. Those techniques mentioned earlier makes CNN algorithms more efficient and have been used in the self-driving car project.

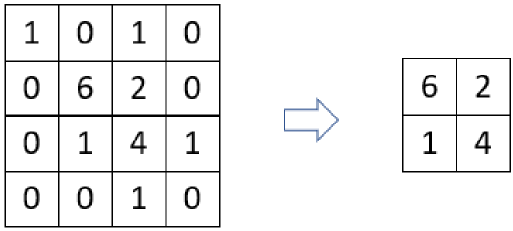


Figure 7: Max pooling

Figure 8 shows the steps of the CNN Deep learning algorithm and includes all techniques mentioned earlier. In Figure 8, the number of convolutions and sub-algorithms may defer according to the system designed by developers.

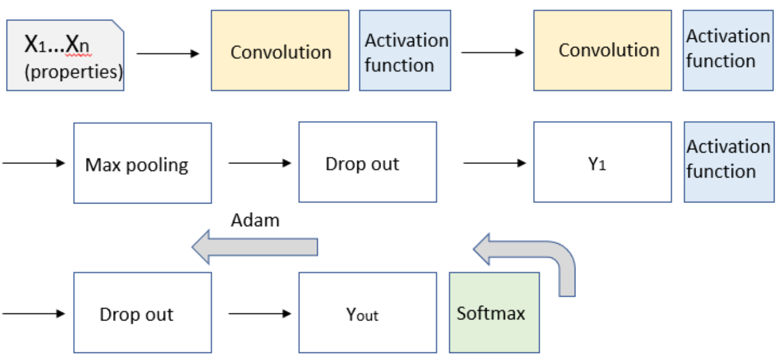


Figure 8: Steps in the CNN algorithm

5 Experiments and result

Figure 9 shows the self-driving car moving on the road on top of the matt. As seen in the figure the computer receives the input data (video images) that comes from the video camera in real time and then compares them with the premade model, in order for the self-driving car to move autonomously.

There are many ways to measure how close the real data is to the data created by the model. Here, the RMSE (Root Mean Squared Error) is used and it is called data loss in this article. The p_i is the real data and the y_i is the data predicted by the model in the equation 5. The n is the number of data

Figure 9: Moving the self-driving car on the road

used in the training.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=0}^{n}(p_i - y_i)^2} \tag{5}$$

In the figure 10 and 11, the 'train' is the real data and the 'validate' is the data predicted by the model. As seen in Figure 10 and 11, the data loss is decreased from 0.81309 to 0.045128 by 94% and the model accuracy improves significantly as the amount of training data is increased from 50% to 80%. In other words, the data loss increases as the amount of training data decreases.

For reference, the self-driving car shows different driving patterns according to the models made from different environments. Especially, the car showed different driving patterns based on the different light brightness and the various surrounding objects. Conclusively, after several trial-and-errors, the model that makes the car move successfully on the road was made.

6 Conclusion and further study

This project was conducted along with the students under the Faculty development grant. In the self-driving car project mentioned in this article, the CNN

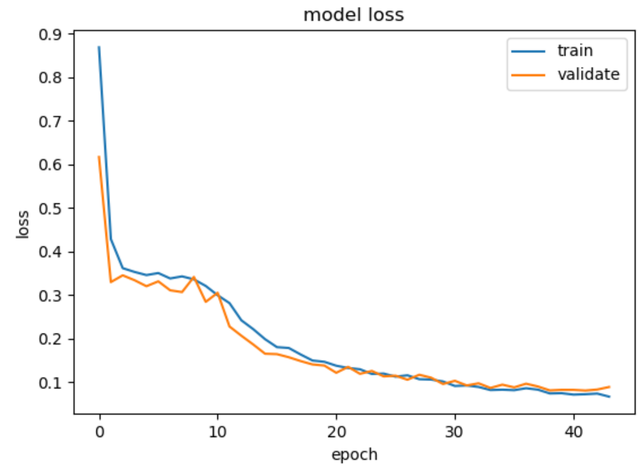


Figure 10: Data loss with 50% of images used during training

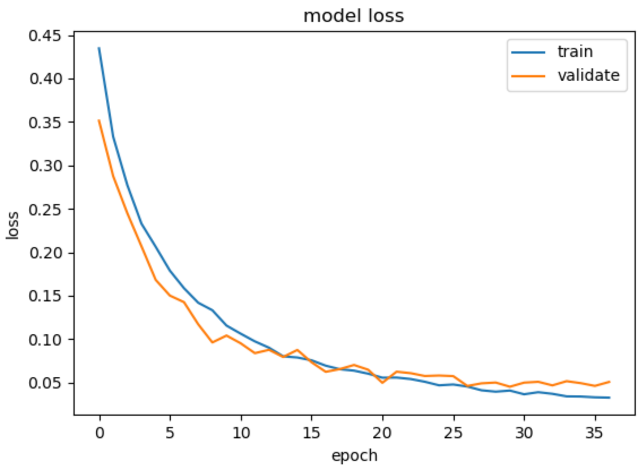


Figure 11: Data loss with 80% of images used during training

Machine learning technique was used and proved the effectiveness to move the self-driving car through road experiments. Above all, the most important meaning of this project was to implement the system learned during lecture. In fact, there have been issues where knowledge from the classroom was occasionally not applicable to the industry, but this project is expected to show the possibility of subsiding this issue.
There is more room to study in order to make the system more efficient, such as how much convolution is needed, what mask size is optimal, what window size is optimal for the max pooling, what activation functions are the best, etc. In addition, new method such as the wavelet, mathematical theory, can be applied to the CNN algorithm used in the self-driving car.

References

[1] Michael G. Bechtel, Elise McEllhiney, Minje Kim, and Heechul Yun. *Deep-Picar: A Low-cost Neural Network- based Autonomous Car*. CS Technical Report (arXiv: 1712.08644), Cornell University, USA, 2017.

[2] Mayur Bhangale, Gaurav Dabhade, Akshay Khairnar, and Mamta Bhagat. Self-driving car to demonstrate real time obstacles & object detection. *International Research Journal of Engineering and Technology*, 3(11):913–915, 2016.

[3] Cameron Franke. *Autonomous Driving with a Simulation Trained Convolutional Neural Network*. B.S. Thesis, University of the Pacific, USA, 2017.

[4] Chris Giarratana. Microsoft's unique approach to the autonomous car industry. `https://www.trafficsafetystore.com/blog/microsofts-unique-approach-to-the-autonomous-car-industry`.

[5] Hans Greimel. Report: Toyota will have a self-driving car for sale in a year. `https://autoweek.com/article/autonomous-cars/report-toyota-will-have-self-driving-car-sale-year`.

[6] Lei Kang, Wei Zhao, Bozhao Qi, and Suman Banerjee. Augmenting self-driving with remote control: challenge and directions. *19th International Workshop on Mobile Computing Systems & Applications*, 2018.

[7] Dmitrii Krasheninnikov. *Autonomous Control of a RC Car with a Convolutional Neural Network*. B.S. Thesis, University of Applied Sciences, Finland, 2017.

[8] Shaoshan Liu, Jie Tang, Chao Wang, Quan Wang, and Jean-Luc Gaudiot. *Implementing a cloud platform for autonomous driving.* CS Technical Report (arXiv:1704.02696), Cornell University, USA, 2017.

[9] WAYMO LLC. We're building the world's most experienced driver. `https://waymo.com/`.

[10] Matt McFarland. Amazon invests in self-driving vehicle startup aurora. `https://www.cnn.com/2019/02/07/tech/amazon-autonomous-vehicles/index.html`.

[11] Gurjashan Singh Pannu, Mohammad Dawud, and Pritha Gupta. Design and implementation of autonomous car using raspberry pi. *International Journal of Computer Applications*, 113(9):22–29, 2015.

[12] Brian Pfretzschner. *Autonomous Car Driving using a Low-Cost On-Board Computer.* B.S. Thesis, Technische Universitat Darmstadt, Germany, 2013.

[13] K.N.V. Satyanarayana, B. Tapasvi, P. KanakaRaju, and G. RameshBabu. Based on machine learning autonomous car using raspberry-pi. *International Journal of Engineering Research and Application*, 7(12):76–82, 2017.

[14] Saqib Shah. Microsoft will help train lg's self-driving car soft@miscware. `https://www.engadget.com/2019/01/09/microsoft-azure-lg-self-driving-cars`.

[15] Himanshu Sharma, Mridul Chaudhary, Nitesh Manjwani, Ruchi Kulkarni, and Amita Jajoo. Autonomous rc car using neural network and deep learning. *International Journal of Innovative Research in Computer and communication Engineering*, 6(5):242–248, 2018.

Facelock — A Labware for Teaching Photo Privacy in Online Social Networks through Face Recognition*

Sadia Shormin, Lin Li, Na Li
Department of Computer Science
Prairie View A&M University
Prairie View, Texas 77446
sshormin@student.pvamu.edu, {lilin,nali}@pvamu.edu

Abstract

Online photo sharing has become a popular activity for Internet users. Semantically rich photos often contain not only the information that the uploaders want to share but also the information that is sensitive to others. However, most of the current online social networks do not have well-defined mechanisms for user privacy protection. This paper discusses the design and implementation of a labware—Facelock which was developed for teaching photo privacy. The goal is to increase students' awareness of privacy protection while sharing photos in online social networks. Through the labware, students can gain a thorough understanding of the photo privacy and essential concepts of face recognition. This labware can be used in both cybersecurity and data science courses. It was pilot-tested through a workshop in 2019 among thirteen students majoring in Computer Science. Surveys results showed the effectiveness of the labware and the attainment of the learning objectives.

1 Introduction

Today, people frequently interact with their families, friends, and colleagues through online social networks (OSN), especially with the advent of smart phones with high-quality camera embedded which enables users to record their life anytime and anywhere. People enjoy posting and sharing their photos in online communities, blogs, and content sharing sites. For example, there are over 350 million photos uploaded daily on Facebook [11], and users share over 700 million photos daily on Snapchat [5].

Many times, the semantically rich pictures contain sensitive contents which may disclose users' private information such as location and personal habits. More importantly, sharing such pictures online may unnecessarily expose users' friends and acquaintances involved in the photos, thereby leading to their privacy breach [3]. Most photo sharing websites allow users to configure their privacy preferences to some extent. For example, Facebook allows users to decide whether they want to share a photo with their friend group or family group or the public. Unfortunately, recent studies have shown that users had struggles to set up and maintain such privacy settings [13]. One of the primary reasons addressed is that, given the amount of shared information, such configuration process can be tedious and error-prone as too many factors need to be evaluated before the decision is made. Therefore, many researchers have acknowledged the need of access control and privacy policy recommendation systems which can assist users to properly configure privacy settings.

The education on image privacy, however, has not been well integrated into the undergraduate security classes. The main obstacle is the lack of effective hands-on learning labs related to this topic. To address this problem, the authors were motivated to develop a labware with the following objectives: (1) increase students' awareness of image privacy protection; (2) make students understand different users' privacy preferences; (3) teach students basic protection mechanisms for image privacy; and (4) understand the trade-off between privacy protection and its cost.

The rest of the paper is organized as follows: Section 2 briefly discusses the literature related to image privacy protection and describes the current development of privacy education. Section 3 introduces the design and implementation of the Facelock system. Section 4 presents the evaluation results after surveying thirteen students majoring in Computer Science at Prairie View A&M University through a workshop conducted in 2019. Finally, a conclusion is made in Section 5.

2 Related Work

2.1 Image Privacy Protection

To enhance image privacy protection, some existing work [7, 15] focused on developing access control based approaches. They leverage photo tags which are either labelled by people or generated by the tagging service provided in most photo sharing sites for purposes including organization, search, communication, and description of photos. The tags can help users intuitively create and maintain fine-grained access-control policies more. Another group of technologies focus on developing privacy policies which help photo uploaders automatically decide whether a photo should be labeled as public or private. Different photo related information, including tags, captions, comments, meta data and image content, is used to classify private and public photos [12, 16, 8]. More recently, big data technologies are also used to extract better features which can be used for the photo classification [17, 14]. In addition, some photo privacy preservation technologies [7, 15] manage privacy protection level to objects involved in photos instead of the entire photo [14]. Then some image operations, such as blurring, local encryption, warping, pixelation, and masking, are applied on the sensitive objects identified from the photos [6, 9].

2.2 Privacy Education

The ACM's Computer Science Curriculum 2013 has acknowledged the importance of privacy education [1]. Currently, the development of curriculum for teaching privacy to younger students is still insufficient. A team of cross-disciplinary members, including computer scientists, educators, and social scientists, from the International Computer Science Institute (ICSI) and the University of California at Berkeley, have developed an online privacy curriculum [2] including ten principles with the purpose of spreading the awareness of protecting online privacy among younger students. However, their curriculum focuses on the general education of online privacy instead of photo privacy in OSN. To the best of the authors' knowledge, the work described in this paper is the first one on a learning labware development particularly for educating younger students to protect privacy while sharing photos in OSN.

3 Design of the Labware

The in-house developed labware—Facelock is a web based system. It consists of three components: (1) a SQLite database, (2) a web server, and (3) a face recognition library. The three components work together with an emphasis on the image privacy protection principles, photo recognition, and the access control procedure. A framework of the labware is illustrated in Figure 1.

3.1 SQLite Database

The Facelock system uses SQLite database to store the information of the registered users, including their post records, profiles, friend lists, etc. For face recognition purpose, each registered user must upload a "standard" picture as

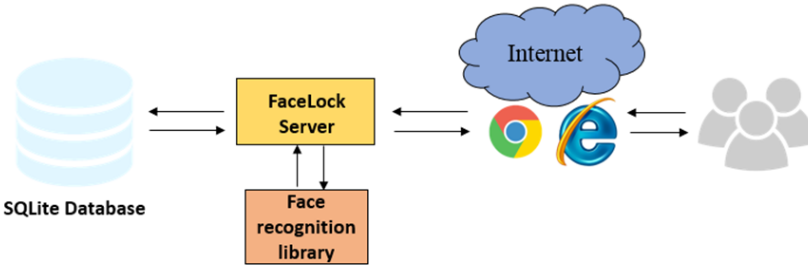


Figure 1: The system architecture of the Facelock labware

part of his/her profile, as shown in Figure 2. The profile pictures of the users will be used for face recognition by the face recognition library. Compared with other relational database management systems, SQLite is a lightweight disk-based database system contained in a C programming library. It does not require a separate server process and allows database access using a nonstandard variant of the SQL query language.

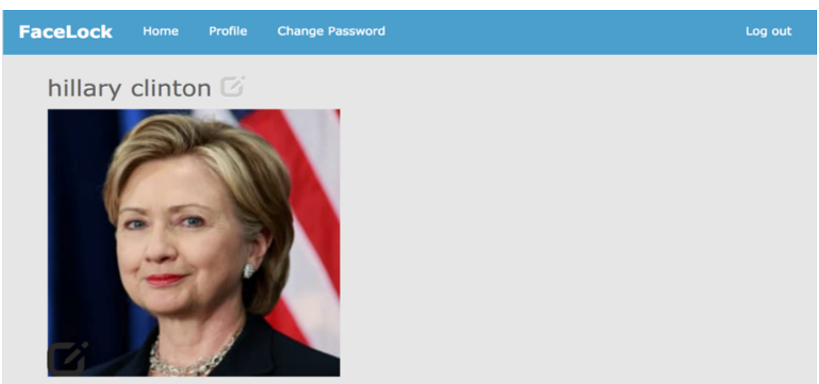
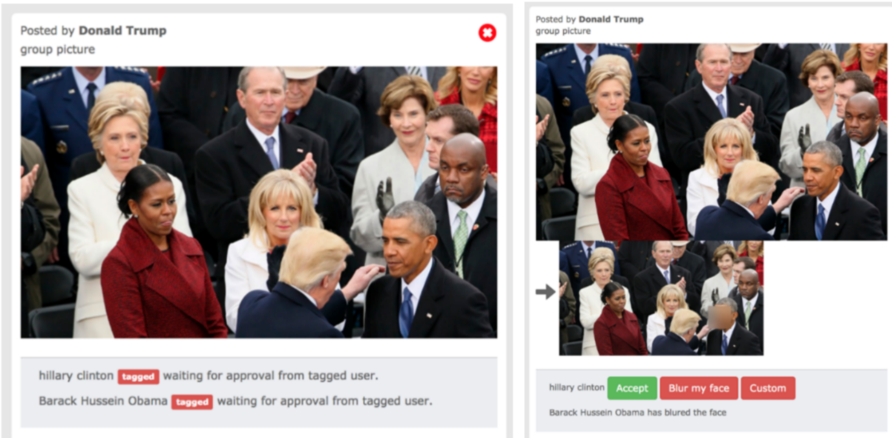


Figure 2: The profile page of a Facelock user

3.2 Facelock Web Server

The Facelock web server hosts a Facebook like social network environment where users can post messages and pictures, search people, and join friend circles, etc. Apart from these basic functions, Facelock reinforces photo privacy protection. Specifically, when a user attempts to post a photo, the web server will utilize the face recognition library to tag users on Facelock who are included



(a) Tagged users through face recognition (b) After clicking "Blur my face"

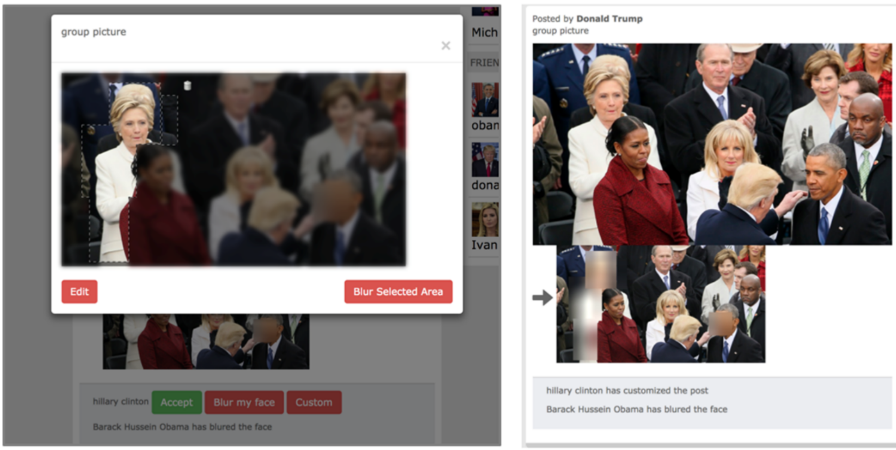
Figure 3: Individually tagged users through face recognition

in the photo. Each tagged user will receive an alert and then take action to edit the photo to meet his privacy preference. The post will not be made publicly available until all tagged users respond with their privacy protection choices. Figure 3a depicts a scenario where the user "Donald Trump" tries to post a picture, in which recognized Facelock users such as "Hillary Clinton" and "Barack Obama" are tagged. Therefore, "Donald Trump" has to wait for them to grant the releasing of the picture before sharing it with the public.

At the current stage, the Facelock system implemented three privacy protection options for tagged users, including *no anonymization*, *face anonymization*, and *selected area anonymization*.

No anonymization does not provide any protection to the user. The tagged user who selects this option (by clicking the "Accept" button shown in Figure 3b) is careless about releasing his picture to public by the uploader. With this option, the tagged user grants the system and the uploader to post the picture.

Face anonymization allows the tagged user to protect his face image. Once the "Blur my face" button is clicked, the tagged person's face will be blurred automatically and the blurred picture will be displayed together with the original picture for preview. After that, the photo uploader and the other tagged users will be notified that this tagged person has rejected the post and chosen to hide his identity in the image. Figure 3b shows a scenario that when the tagged user "Hillary Clinton" logged into the system, she noticed that her



(a) Custmize blurring area

(b) After area blurring

Figure 4: Photo preview after privacy protection using "Custom" option

face was shown in a new post created by "Donald Trump" and that another tagged user "Barack Obama" already chose to blur his face.

Selected area anonymization enables the tagged user to customize the image anonymization if he is not satisfied with simply blurring his face. In this case, the user can click the "Custom" button, and then he will see a modal window with the "Edit" option. By clicking the "Edit" button, the user can select multiple areas to blur and then confirm it by clicking "Blur Selected Areas". Figure 4 shows a scenario where the tagged user "Hillary Clinton" further customized the picture by blurring her image.

Once all tagged users made their privacy protection choices, the original photo uploader can review the altered photo. The blurred areas will be the "cost" of posting the picture online. This is because a blurred picture may lose its original value for visual aesthetics. Therefore, the uploader can make a decision to post the photo edited by the tagged users or to give up the post. In Facelock, the image manipulation function was implemented using OpenCV which is a library of functions commonly used in the research and applications related to computer vision [10]. The library contains different options for blurring images such as Gaussian Blurring, Median Blurring, and Bilateral Filtering. In Facelock, Gaussian Blurring was used.

3.3 Face Recognition Library

A key component of Facelock is the Face Recognition library which is used to recognize faces in the photos for user tagging. There have been different machine learning models developed for image classification. Considering the model complexity and the speed required for fast web processing, the one adopted by Facelock is a deep leaning based open source application available on GitHub [4]. As the model script was written in Python and Facelock was also developed using the same language with the Django framework, this model was seamlessly integrated into Facelock. The advantage of this library is its light weight and high accuracy. Particularly, it only requires the registered user to upload one "standard" profile picture to reach high accuracy and it can achieve the recognition accuracy of 99.38% on the labeled faces in the Wild benchmark (http://vis-www.cs.umass.edu/lfw/). All photos posted in Facelock will be examined with the existing profile pictures to recognize users. A code snippet of using the library is as follows:

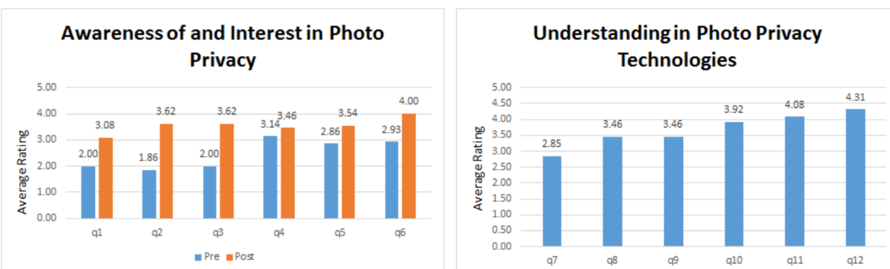
```
# Import the face recognition library
import face_recognition
# Detecting "Trump" by comparing the profile picture with the unknown picture
known_image=face_recognition.load_image_file("trump.jpg")
unknown_image=face_recognition.load_image_file("unknown.jpg")
trump_encoding=face_recognition.face_encodings(known_image)[0]
unknown_encoding=face_recognition.face_encodings(unknown_image)[0]
results=face_recognition.compare_faces([trump_encoding],
                                        unknown_encoding)
```

4 Evaluation

In order to expose students to privacy preservation and inspire their learning interests, the authors pilot-tested this educational tool in an educational workshop conducted in fall 2019 for student training. A total of thirteen undergraduate students from Prairie View A&M University majoring in Computer Science participated in the evaluation of the labware. The learning and evaluation activities fell into two categories: (1) classroom presentation to introduce photo privacy in OSN and anonymization basics, and (2) hands-on labs using Facelock to examine the three anonymization options discussed in Section 3 and understand the tradeoff between privacy preservation and its cost. Pre and post surveys were conducted at the beginning and the end of the testing session to evaluate and analyze the outcomes. The student survey questions are listed in Table 1. All questions use a rating scale of 1 to 5 with 5 being the greatest deal or the most positive.

Table 1: Pre and post survey questions

#	Survey Questions	Type
1	Rate your awareness of access control mechanism in OSN?	Pre & Post
2	Rate your awareness about privacy disclosure from photo sharing?	Pre & Post
3	Rate your awareness about photo anonymization mechanism?	Pre & Post
4	Rate your interest in access control mechanism in OSN?	Pre & Post
5	Rate your interest in privacy disclosure from photo sharing?	Pre & Post
6	Rate your interest in photo anonymization mechanism?	Pre & Post
7	Rate your learning about access control mechanism in OSN?	Post
8	Rate your learning about privacy disclosure from photo sharing?	Post
9	Rate your learning about photo anonymization mechanismlab?	Post
10	This lab helped me understand the access control mechanism in OSN	Post
11	This lab helped me know how a privacy-aware tagging system works	Post
12	This lab and privacy preservation should be taught in security courses	Post



(a) Awareness and interest change

(b) Understanding of photo privacy

Figure 5: Experimental results

Questions 1 to 6 were surveyed in both pre and post questionnaires. Figure 5a plots the average rating with regard to the discrepancy of pre and post survey results. A significant increase is observed on students' awareness of and interest in access control mechanism in OSN, privacy disclosure of photo sharing, photo anonymization after the lab. Questions 7-12 measured how much students gained in understanding the related concepts. The survey results were positive and encouraging with the average rating being greater than 3.4 from five of the questions, as shown in Figure 5b. One lesson the authors learned from the lab is to extend the lecturing time on the access control mechanism which would bring better feedback for questions 4 and 7.

5 Conclusion

This paper presents the design and implementation of a tool intended for teaching photo sharing privacy in online social networks. The goal was to provide students with a solid understanding of the topic by using an in-house developed labware—Facelock, which not only enables students to be aware of the possible privacy disclosure of photo sharing in online social networks, but also equips them with necessary defense technologies. After pilot-testing the labware among thirteen students in fall of 2019, the authors got very positive feedback. Students confirmed that they thoroughly understood the photo privacy in OSN through the lab activities, and the tool helped them learn the concepts effectively. This shows that the teaching tool can be widely used in classrooms and integrated into security curriculum.

References

[1] Computer science curricula 2013. `https://www.acm.org/binaries/content/assets/education/cs2013_web_final.pdf`.

[2] Teaching privacy. `http://teachingprivacy.org`.

[3] Andrew Besmer and Heather Richter Lipford. Tagged photos: Concerns, perceptions, and protections. In *Proceedings of the 27th International Conference Extended Abstracts Human Factors Computer System*, pages 4585–4590, 2009.

[4] Adam Geitgey. Face recognition. `https://github.com/ageitgey/face_recognition`.

[5] Ellis Hamburger. Real talk: the new snapchat brilliantly mixes video and texting. `http://www.theverge.com/2014/5/1/5670260/real-talk-the-new-snapchatmakes-texting-fun-again-video-calls`.

[6] Rakibul Hasan, Eman Hassan, Yifang Li, Kelly Caine, David James Crandall, Roberto Hoyle, and Apu Kapadia. Viewer experience of obscuring scene elements in photos to enhance privacy. In *Proceedings of the ACM CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2018.

[7] Panagiotis Ilia, Iasonas Polakis, Elias Athanasopoulos, Federico Maggi, and Sotiris Ioannidis. Face/off: Preventing privacy leakage from photos in social networks. In *Proceedings of 22nd ACM SIGSAC on Computer and Communication Security*, pages 781–792, 2015.

[8] Fenghua Li, Zhe Sun, Ang Li, Ben Niu, Hui Li, and Guohong Cao. Hideme: Privacy-preserving photo sharing on social networks. In *Proceedings of IEEE Conference on Computer Communications*, 2019.

[9] Yifang Li, Nishant Vishwamitra, Bart P. Knijnenburg, Hongxin Hu, and Kelly Caine. Effectiveness and users' experience of obfuscation as a privacy-enhancing technology for sharing photos. In *Proceedings of the ACM: Human Computer Interaction*, 2017.

[10] Kari Pulli, Anatoly Baksheev, Kirill Kornyakov, and Victor Eruhimov. Real-time computer vision with opencv. *Communications of ACM*, 55(6):61–69, June 2012.

[11] Cooper Smith. Facebook users are uploading 350 million new photos each day. `http://www.businessinsider.com/facebook-350-million-photos-each-day-2013-9`.

[12] Anna Cinzia Squicciarini, Dan Lin, Smitha Sundareswaran, and Joshua Wede. Privacy policy inference of user-uploaded images on content sharing sites. *IEEE Transaction on Knowledge and Data Engineering*, 27(1):193–206, Jan. 2015.

[13] Katherine Strater and Heather Richter Lipford. Strategies and struggles with privacy in an online social networking community. In *Proceedings of the 22nd British HCI Group Annual Conference on People and Computers: Culture, Creativity, Interaction*, pages 111–119, 2008.

[14] Ashwini Tonge and Cornelia Caragea. Image privacy prediction using deep features. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 4266–4267, 2016.

[15] Nishant Vishwamitra, Yifang Li, Kevin Wang, Hongxin Hu, Kelly Caine, and Gail joon Ahn. Towards pii-based multiparty access control for photo sharing in online social networks. In *Proceedings of the 22nd ACM on Symposium on Access Control Models and Technologies*, pages 155–166, 2017.

[16] Jun Yu, Zhenzhong Kuang, Baopeng Zhang, Wei Zhang, Dan Lin, and Jianping Fan. Leveraging content sensitiveness and user trustworthiness to recommend fine-grained privacy settings for social image sharing. *IEEE Trans. Information Forensics and Security*, 13(5):1317–1332, May 2018.

[17] Jun Yu, Baopeng Zhang, Zhengzhong Kuang, Dan Lin, and Jianping Fan. iprivacy: Image privacy protection by identifying sensitive objects via deep multi-task learning. *IEEE Transaction on Information Forensics and Security*, 12(5):1005–1016, May 2017.

Teaching High Level Abstraction Using Directed Active Learning*

Conference Tutorial

David Klappholz
Stevens Institute of Technology
Hoboken, NJ
davidk6@gmail.com

Abstraction has repeatedly been recognized as a critical component of software development, and, therefore, as a critical component of Computer Science Education. Most published work on the subject, however, deals only with abstraction at a low level, for example, at the level of classes and data structures (CS1 and CS2) [2],[4], [5], [6] but abstraction at higher levels is equally critical to software development. This was made clear to Prof. Klappholz when a faculty colleague proposed a potential software development project for his capstone course [1]. The design of the software turned out to require the construction of an abstraction that class members were unable to construct using undirected active learning, and that had to be given to them by the author in order for them to be able to further develop the software. The author has subsequently used the design of the same software as an assignment in both junior- and senior-level undergraduate and entry-level MS courses, and has found that virtually all class members have been unable to construct the required abstraction. After learning that active learning often fails unless it is carefully directed[3], the author has used the design of the software in question in a junior-level course, but, this time, has carefully directed the student teams' active learning, and has achieved far greater success. In this tutorial we will cover the topics of the teaching/learning of high level abstraction and of directed active learning, with detailed examples from the software development project described above, and how others might use directed active learning to teach high level abstraction.

References

[1] Capstone Course: `https://sites.google.com/view/sitseniordesign/home`.

[2] J. Bennedsen and M.E. Caspersen. Abstraction ability as an indicator of success for learning object-oriented programming? *SIGCSE Bulletin*, 38(2):39–43, 2006.

[3] Paul A. Kirschner, John Sweller, and Richard E. Clark. Why minimal guidance during instruction does not work: An analysis of the failure of constructivist, discovery, problem-based, experiential, and inquiry-based teaching. *Educational Psychologist*, 41(2):75–86, 2006.

[4] D.Z. Nguyen and S.B. Wong. OOP in introductory CS: Better students through abstraction. In *Procedings of the Fifth Workshop on and Tools for Assimilating Object-Oriented Concepts*, OOPSLA '01, 2001.

[5] R. Orbach and I. Lavy. Cognitive activities of abstraction in object orientation: an empirical study. *SIGCSE Bulletin*, 36(2):82–86, 2004.

[6] P. Sprague and C. Schahczenski. Abstraction the key to CS1. *Journal of Computing Sciences in Colleges*, 17(3):211–218, 2002.

A Gentle Introduction to Deep Learning*

Conference Tutorial

Eduardo Colmenares[1], Yujin Yoshimura[1], Heng Wu[2]
[1] Computer Science Department
Midwestern State University
Wichita Falls, Texas
eduardo.colmenares@msutexas.edu, yujinyoshimura@gmail.com
[2] Computer Science Department
West Virginia State University
Institute, West Virginia
heng.wu@wvstateu.edu

Thanks to emerging technologies such as many-core devices and the vast amount of data created during the last decade, the field of Artificial Intelligence has experienced a considerable growth, so to have the fields of Machine Learning, Deep Learning and Data Science. This workshop will begin by familiarizing members of the audience with important concepts such as what is a Neural Network, the components of a Neural Network, Forward Propagation, Back Propagation, Activation Functions, Gradient Descent and Error Estimation among several other concepts. In order to walk the audience through several of these concepts a step by step numerical example that can be easily adopted as class material by other interested faculty members will be presented. Last but not least, the audience will be introduced to at least one of the most popular Deep Learning Frameworks (either Tensorflow or Keras). The use of the framework(s) will serve to reinforce several of the concepts previously introduced.

Communicating Sequential Processes in the Go Programming Language*

Conference Tutorial

Perugini Saverio[1] *and David J. Wright*[2]
[1]*Department of Computer Science*
[2]*Learning Teaching Center*
University of Dayton
Dayton, Ohio
{sperugini1,dwright1}@udayton.edu

Project Site: http://sites.udayton.edu/operatingsystems

1 Tutorial Summary

This 60-minute tutorial will cover the Communicating Sequential Processes (CSP) model of concurrency—a (re-)emerging concurrent programming model that is increasing in popularity, especially in the Go programming language[1]. Intended participants are instructors of undergraduate Operating System (OS) courses. This tutorial will also be of interest to instructors of concurrent programming and programming languages courses. Participants will be exposed to concurrent programming using CSP in Go, through the use of typed channels for synchronous or asynchronous communication. In particular, we will lead participants through a variety of active-learning laboratories for teaching a 'concurrent programming and synchronization' module of an undergraduate OS course using CSP in Go. The in-class laboratory plans that attendees can adopt involve programming classical problems of synchronization (e.g., producer-consumer and dining philosophers) through the lens of the CSP model. Participants should have working knowledge of concurrency/synchronization in a language such as C or Java and are encouraged to bring laptop

[1]http://golang.org

computers, especially to follow the labs demonstrated. Unlike semaphores and condition variables, the CSP model naturally focus the programmer's locus of attention on the interaction between concurrent entities to collaboratively solve a problem—the primary learning outcome—rather than the level-low details of language syntax and the minutia of which individual data to protect/lock and how to protect/lock it. Therefore, more broadly, we also anticipate fostering and facilitating a discussion of innovative approaches toward teaching the topic of concurrency and synchronization in an undergraduate OS course[1].

2 Schedule of Activities

- (10 minutes) Introduce attendees to the fundamentals of CSP and Go.
- (40 minutes) Work through two active-learning laboratory plans of the CSP model of concurrency in Go.
  - The Producer-Consumer problem
  - The Dining Philosophers problem
- (10 minutes) Discussion
  - Share our experience in teaching the OS course using this model.
  - How to use the active-learning labs in our Laboratory Manual.
  - Invite participation in our community of OS educators.
  - Questions and Answers.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant Numbers 1712406 and 1712404. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] Saverio Perugini and David J Wright. Developing a contemporary and innovative operating systems course. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, pages 1248–1248, 2019. Conference Birds-of-a-Feather; DOI:`http://doi.acm.org/10.1145/3287324.3293734`.

Hands-On Intro to Google Cloud for Your Courses*

Conference Tutorial

Wesley Chun
Developer Advocate
Google Cloud
`wesc+api@google.com`

This session begins with a brief high-level intro to Google Cloud (as a review of the full talk last year), providing an overview of relevant products for your students or research, covering core product APIs along w/code samples and information on our education grants. A hands-on follows (attendees are invited to follow along on their laptops) which demonstrates how to use several of our serverless tools for student code. Several demos of using Google APIs are also covered here. Sample code shown will be in Python & JavaScript, so having this knowledge would be useful but not required.