6-2018

Motivating Students to Learn a Programming Language: Applying a Second Language Acquisition Approach in a Blended Learning Environment

Lulu Sun
Embry-Riddle Aeronautical University, Lulu.Sun@erau.edu

Christina Frederick
Embry-Riddle Aeronautical University, christina.frederick@erau.edu

Caroline Liron
liron9e5@erau.edu

Li Ding
Embry-Riddle Aeronautical University, Li.Ding@erau.edu

Lei Gu
Georgia State University

See next page for additional authors

Scholarly Commons Citation

Sun, L., Frederick, C., Liron, C., Ding, L., Gu, L., Griggs, A. C., & Espejo, P. S. (2018). Motivating Students to Learn a Programming Language: Applying a Second Language Acquisition Approach in a Blended Learning Environment. *Proceedings of the ASEE Annual Conference and Exposition,* (). Retrieved from https://commons.erau.edu/publication/1635

Authors
Lulu Sun, Christina Frederick, Caroline Liron, Li Ding, Lei Gu, Andrew Calvin Griggs II, and Paula Sanjuan Espejo

2018 ASEE Annual Conference & Exposition
June 24–27, 2018, Salt Palace Convention Center, Salt Lake City, UT
Paper ID #22740

Motivating Students to Learn a Programming Language: Applying a Second Language Acquisition Approach in a Blended Learning Environment

Dr. Lulu Sun, Embry-Riddle Aeronautical University

LDr. Lulu Sun is an associate professor of Engineering Fundamentals Department at Embry-Riddle Aeronautical University, where she has taught since 2006. She received her B.S. degree in Mechanical Engineering from Harbin Engineering University (China) in 1999, and her Ph.D. degree in Mechanical Engineering from University of California, Riverside in 2006. Before joining Embry-Riddle, she worked for Arup at Los Angeles office as a fire engineer. Her research interests include second language acquisition in programming languages, flipped classroom, and best practices of virtual training delivery. She is a professional member of the Society of Fire Protection Engineers, and a member of the American Society for Engineering Education. She has published over 40 journal and conference articles nationally and internationally.

Dr. Christina Frederick, Embry-Riddle Aeronautical University

Dr. Frederick is currently a Professor and MS Graduate Program Coordinator in the Human Factors and Systems Department at Embry-Riddle Aeronautical University in Daytona Beach, Florida. Dr. Frederick received her Ph.D. in 1991 from the University of Rochester with a major in Psychological Development. She previously taught at the University of Rochester, Southern Utah University and the University of Central Florida. In 2000, Dr. Frederick joined the Human Factors and Systems Department at Embry-Riddle, where her work focused on applied motivation and human factors issues in aviation/aerospace. Dr. Frederick also served in various roles in University administration between 2004-2012, including Vice President for Academics and Research. Dr. Frederick's current research interests examine how individual differences interact with technology to enhance educational engagement and performance. Dr. Frederick is the author of more than 50 research publications, 4 book chapters and over 60 regional, national and international conference presentations on a wide range of topics in human factors and psychology. She is active in a number of professional associations, and is a Consultant for Psi Chi, the National Honor Society in Psychology.

Prof. Caroline Liron, Embry-Riddle Aeronautical University

Caroline Liron is an Assistant Professor in the Engineering Fundamentals Department, at Embry-Riddle Aeronautical University (ERAU), where she has been teaching since 2005. She obtained her bachelor's in Aeronautics and Space from EPF, Ecole d'Ingénieur (France), and her M.S. in Aerospace Engineering from ERAU. She currently teaches Introduction to Programming for Engineers. She is involved in developing and maintaining the hybrid version of that class, and researching improvements methods to teach programming to incoming freshmen using new technologies. She also researches means to incorporate more engineering mathematics and physics into the programming course.

Dr. Li Ding

Li Ding is a visiting professor of the Department of Engineering Fundamentals at Embry-Riddle Aeronautical University, where she has been since 2012. She received her Ph.D in Environmental Engineering from the University of Illinois at Urbana-Champaign in 2010. She taught several undergraduate courses in engineering and in science, and she currently teach Introductory to Programming for Engineers. From a background of an engineer, she is transitioning into an educator, and has been working with other principle researchers on education studies since 2015.

Dr. Lei Gu, Georgia State University

Dr. Gu is an Engineering Assistant Professor at Georgia State University. She is a Georgia Tech Regents' Engineering Pathway Program (REPP) coordinator. Dr. Gu received her Ph.D. in Material Science and Engineering from Norfolk State University. Her research interests include scientific visualization and engineering education. Dr. Gu has taught five freshman and sophomore level engineering course.

Mr. Andrew Calvin Griggs II, Embry-Riddle Aeronautical University

Andrew Griggs is a human factors graduate student at Embry-Riddle Aeronautical University.

Paula Sanjuan Espejo, Embry-Riddle Aeronautical University

Undergraduate Aerospace Engineering student at Embry-Riddle Aeronautical University. I worked as an undergraduate research assistant for the SLA-aBLe project from Spring 2014 until the end of 2017.

Paula Sanjuan Espejo, Embry-Riddle Aeronautical University

Motivating Students to Learn a Programming Language: Applying a Second Language Acquisition Approach in a Blended Learning Environment

Introduction

Learning a programming language typically involves acquisition of new vocabulary, punctuation, and grammatical structures to communicate with a computer. In other words, learning a programming language is like learning a human language. A recent study showed that programmers use language regions of the brain when understanding source code and found little activation in other regions of the brain devoted to mathematical thinking. Even though programming code involved mathematical operations, conditionals, and loop iterations, researchers found that programming had less in common with mathematics and more in common with human language [1].

In our study, we applied the well-developed cognitive framework used in second language acquisition (SLA), into a Blended Learning (aBLe) programming language course. SLA is also called "sequential language learning." Linguist Stephen Krashen identified five distinct stages of SLA, which includes preproduction, early production, speech emergence, intermediate fluency, advanced fluency [2–6]. We believe this approach can accommodate a variety of learning needs and abilities, while potentially increasing student engagement in online components, reducing the intimidation and anxiety associated with learning programming languages, and providing better preparation for face-to-face classes. SLA-aBLe will encourage the development of problem solving skills needed to succeed in higher education [6–10].

The online module consists of a series of short videos (~10 minutes), online quizzes with tiered questions including program writing problems, and a topic-specific discussion board led by student researchers. Lab practice time is used to augment the online content through collaborative learning exercises, such as Think, Pair, and Share. The SLA-aBLe program utilized strategies in five stages defined in the SLA five stage model, such as self-testing, tiered questions and visually-aided explanations in the screencasts, more online programming writing assessments, greater collaboration, and 'speak aloud' in the lab. In the past two years, we have conducted a series of assessments to measure program outcomes, including student demographics, perceptions, attitudes, and satisfaction level comparing SLA-aBLe, and control groups. Students' academic performance between SLA-aBLe course sections and Non-SLA sections was compared as well. The online modules were implemented by a two-year public institution in the southeast in fall 2017 as supplemental learning materials of a MATLAB (Mathworks, Natick, MA) programming language course [12].

The research questions that will be addressed in this paper include:

- Will SLA-aBLe help motivate students to learn in a simplified and easy to understand environment?
- Will SLA-aBLe improve student performance in programming language study?
- How did students perceive the effectiveness of their learning experience in the SLA-aBLe course?

Course Redesign

In this project, different cognitive skills are focused on at each of five stages of SLA with the implementation of associated instructional strategies in an Introduction to Computing for Engineers course at a private institution in the southeast initially. In fall 2017, a two-year public institution in the southeast adapted SLA-aBLe on one campus. The course teaches engineering students how to learn a programming language, using MATLAB in a blended learning mode [12–16] . Table 1 shows a comparison of current blended learning and SLA-aBLe development. There are five topics (introduction to MATLAB, data type, input and output, conditional statements, and loops), which were designed and implemented following the SLA approach in the past two years.

Table 1. A comparison of current blended learning and SLA-aBLe development

	Preproduction (minimal comprehension)	**Early Production (limited comprehension)**	**Speech Emergence (increased comprehension)**	**Intermediate Fluency (very good comprehension)**	**Advanced Fluency**
Current Blended Learning	No introduction of the new keywords, or syntax. No connection between the new keywords, syntax, and the new sentences.	There are multiple choice questions but no simple programs. Questions are all at the same level.	Students begin reading and writing in their programming language by solving different engineering problems.	Give students more challenging problems to synthetize what they have learned.	Open-ended engineering project to challenge their understanding and expand their knowledge.
Teaching Strategies in SLA-aBLe	Use pictures and visuals; speak slowly and use simple and shorter words to draw connection between SLA and programming languages; Reinforce learning by giving more self testing questions without adding in pressure.	Reinforce learning by asking students to produce simple programs in addition to the tiered multiple choice questions; use discussion board to encourage group discussion.	Emphasize tiered questions and ask students to do a “think, pair, share” to process the new concepts.	Emphasize compare and contrast different concepts. Allow students to explain their problem solving process.	Project presentation opportunity will be offered to students to enhance their understanding.

At each of the five stages of SLA, different proficiencies were focused on and different cognitive skills related to language learning were developed. Past research shows that at the preproduction stage of SLA, students have minimal comprehension [6–9]. PowerPoint slides were designed to include pictures, animation, interactive tiered questions, and MATLAB programming. The font of the learning materials was changed from an easy to read font, Calibri, to a hard-to-read font, Comic Sans MS so that the material can improve memory performance and educational outcomes. PowerPoint slides were recorded at a slower speed of narration recommended by SLA techniques and divided into a series of 10 minute long interactive screencasts using Camtasia [18]. Closed captions were created to increase learning engagement, comprehension of the topic, and help with accessibility. Music was added at the beginning and the end of each 10 minute long video to engage learning experience as well. Screencasts were uploaded to Edpuzzle website to track the usage statistics. Figure 1 shows a snapshot of the PowerPoint slides and screencasts for the preproduction stage following SLA-aBLe development.

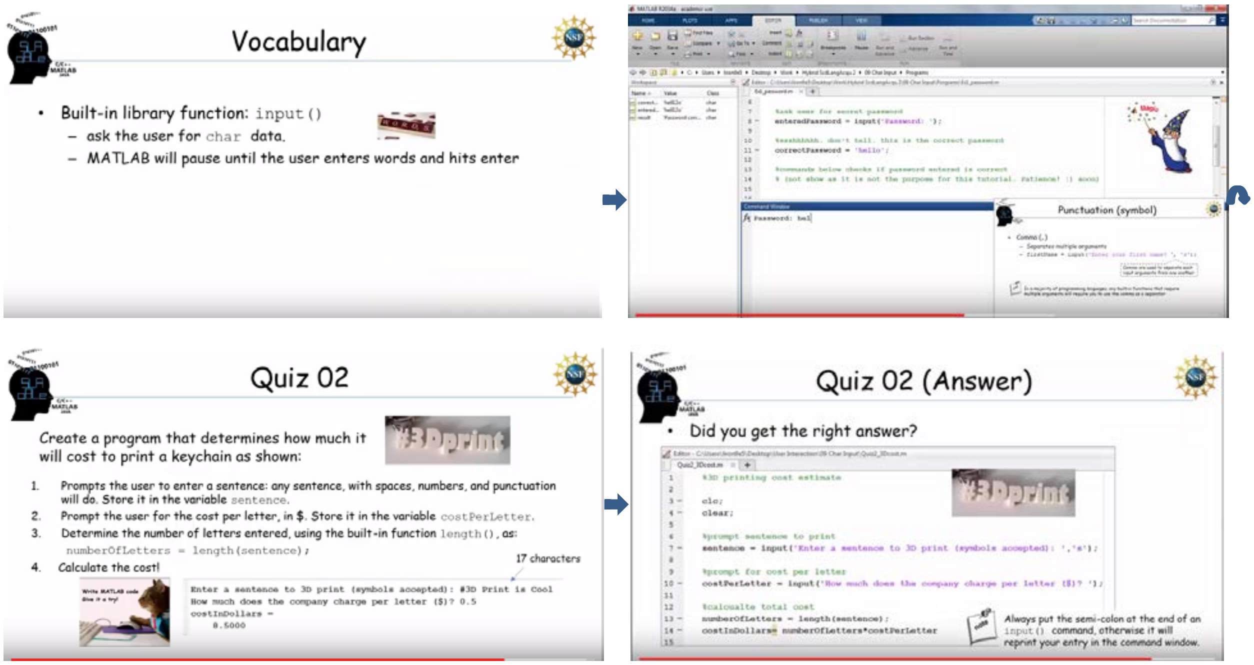
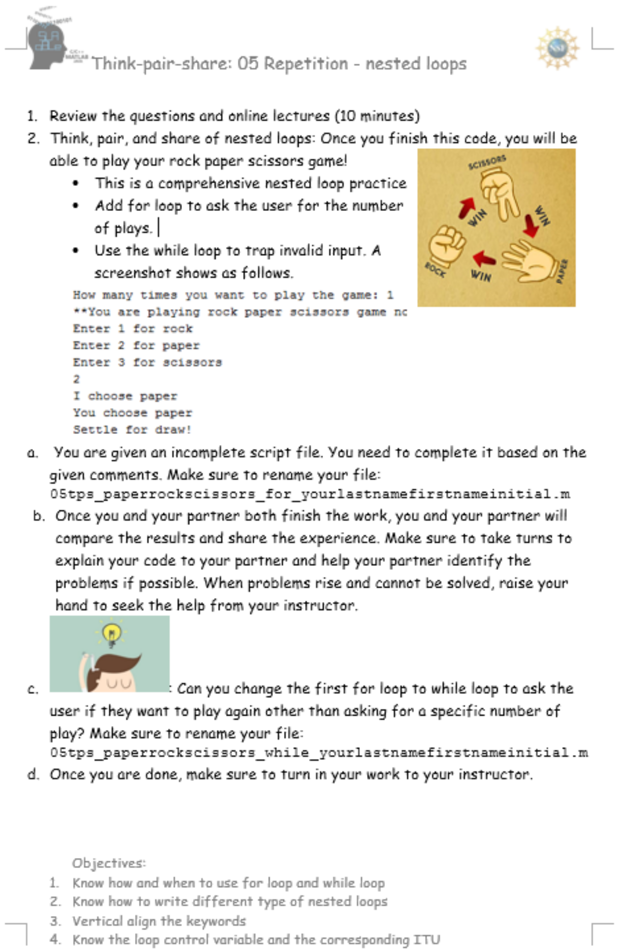


Figure 1. PowerPoint slides design following SLA-aBLe development

Early production skills were obtained by asking students to take an online quiz after each screencast study. There were usually five tiered questions in each online quiz. Students could take the quiz up to three times and the highest score was included into their gradebook. For each topic studied, there was at least one program-writing problem included in the quiz, which needed to be manually graded by the research assistant and project researcher. A discussion board on Canvas was used to facilitate group discussion and provide instructional assistance online. On the following day in the lab, each instructor spent the first 5-10 minutes going over the common mistakes found in the online quizzes. Then students were required to conduct “think, pair, share” exercises in the following 25 minutes so that they could think about what they learned online, explain their learning to their partners, and share their experience facilitating cognitive skills development in the speech emergence stage. Figure 2 shows a snapshot of the “think, pair, share” exercise following SLA-aBLe development.

After the “think, pair, share” exercise, students started their more complicated individual assignment. It was expected that after the completion of the individual assignment, students would demonstrate excellent comprehension and enter the intermediate fluency stage. Finally, at the advanced fluency stage, students developed and refined their knowledge of more sophisticated aspects of grammar and syntax when they started the open-ended final project. It was expected the final project would enhance student’s understanding of the comprehensive materials learned in the whole semester.



Think-pair-share: 05 Repetition - nested loops

1. Review the questions and online lectures (10 minutes)
2. Think, pair, and share of nested loops: Once you finish this code, you will be able to play your rock paper scissors game!
   - This is a comprehensive nested loop practice
   - Add for loop to ask the user for the number of plays.
   - Use the while loop to trap invalid input. A screenshot shows as follows.

```
How many times you want to play the game: 1
**You are playing rock paper scissors game nc
Enter 1 for rock
Enter 2 for paper
Enter 3 for scissors
2
I choose paper
You choose paper
Settle for draw!
```

a. You are given an incomplete script file. You need to complete it based on the given comments. Make sure to rename your file: `05tps_paperrockscissors_for_yourlastnamefirstnameinitial.m`

b. Once you and your partner both finish the work, you and your partner will compare the results and share the experience. Make sure to take turns to explain your code to your partner and help your partner identify the problems if possible. When problems rise and cannot be solved, raise your hand to seek the help from your instructor.

c. : Can you change the first for loop to while loop to ask the user if they want to play again other than asking for a specific number of play? Make sure to rename your file: `05tps_paperrockscissors_while_yourlastnamefirstnameinitial.m`

d. Once you are done, make sure to turn in your work to your instructor.

Objectives:

1. Know how and when to use for loop and while loop
2. Know how to write different type of nested loops
3. Vertical align the keywords
4. Know the loop control variable and the corresponding ITU

Figure 2. “Think, pair, share” implementation in the class time

Assessment Results

There were six surveys conducted each semester in the past two years. The NASA TLX was administered six times across the study to answer the first research question. NASA TLX is a well-established measure of self-assessed workload which measures six dimensions: mental demand, physical demand, temporal demand, performance, effort and frustration [18–20]. Student’s final grades were collected to examine the second research question. Face-to-face interviews were conducted and used to answer the third research question. Statistical analysis was performed by using SPSS software (SPSS, Chicago, IL). The perceived workloads items were analyzed using t-tests and the results are shown in Table 2. No significant differences were found for physical demand and performance demand. However, the mean scores of mental demand, temporal demand, effort, and frustration in SLA-aBLe sections were significantly lower than the mean scores in the non-SLA-aBLe sections. The findings demonstrate lower workload in SLA-aBLe sections than in the non-SLA-aBLe sections.

Table 2. NASA TLX statistics

Workload variable	Class type	n	mean	SD	SE	*Sig.(2-tailed)*
Mental Demand	SLA-aBLe	75	11.95	5.549	.641	<.001
	non-SLA-aBLe	32	16.59	3.387	.599	<.001
Physical Demand	SLA-aBLe	71	5.49	5.783	.686	.220
	non-SLA-aBLe	32	7.09	6.836	1.208	.253
Temporal Demand	SLA-aBLe	74	11.31	5.620	.653	<.001
	non-SLA-aBLe	32	15.94	4.103	.725	<.001
Performance Demand	SLA-aBLe	74	7.16	4.458	.518	.079
	non-SLA-aBLe	32	8.84	4.552	.805	.083
Effort	SLA-aBLe	75	12.13	5.861	.677	<.001
	non-SLA-aBLe	32	16.72	3.531	.624	<.001
Frustration	SLA-aBLe	75	9.94	5.968	.689	<.001
	non-SLA-aBLe	32	14.41	5.471	.967	<.001

The second research question was answered by running a t-test of independence on students' final grades in SLA-aBLe sections and non-SLA-aBLe sections for all four semesters. In this analysis, 336 students were included; 133 in non-SLA-aBLe sections of the class and 203 in SLA-aBLe sections. To minimize selection bias, students were not aware of this project study until the project introduction in the first week of each semester in the randomly selected SLA-aBLe sections given by the researcher. A chi-square analysis showed no significant differences in grades across SLA-aBLe vs. non-SLA-aBLe sections of the class. When grades were transformed into numerical equivalencies (A=4, B=3, etc.) and compared using a t-test, results failed to reach significance (t-.224, p>.05). Although significant differences did not exist across section type, there were fewer failing grades (C, D, and F), and a larger percentage of B grades in the SLA-aBLe sections than the non-SLA-aBLe sections as shown in Figure 3. The data from the public institution shows same trends [12].

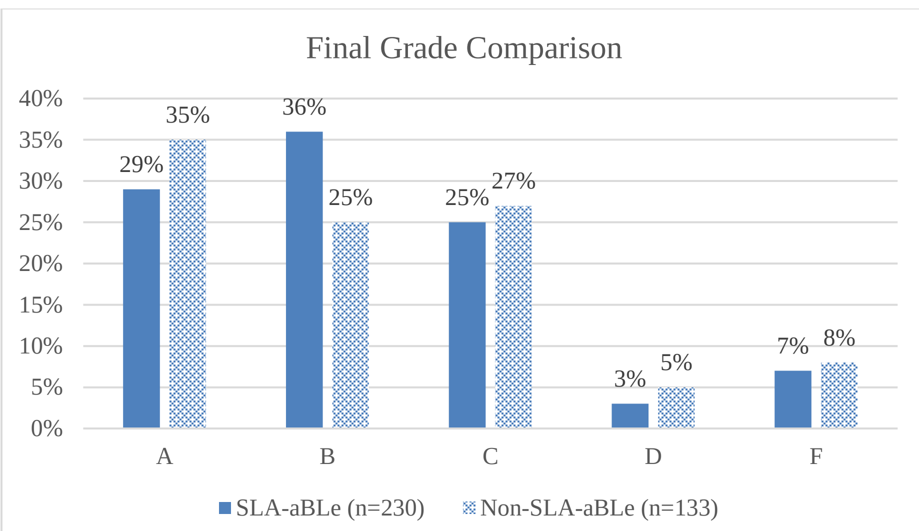


Figure 3. Grades distribution in SLA-aBLe and non-SLA-aBLe sections in four semester study

The third research question was answered by analyzing face-to-face interview results. Six students each semester were interviewed regarding their perception of the course design and their experiences. The questions asked during the interview are listed in Table 2.

Table 2. Face-to-face interview questions in three semesters

Number	Questions
1	Please indicate your previous second language, and programming language experience.
2	Are you in the non-SLA-aBLe section? What is your biggest concern of the class?
3	If you are in the SLA-aBLe section, please answer the following questions: • Do you like the new videos? If yes, what do you like most? If no, explain. • Do you like the online quizzes? If yes, what do you like most? If no, explain. • Do you like the discussion board? If yes, what do you like most? If no, explain. • Do you like the think-pair-share in the lab? Please explain. • Does SLA-aBLe helped engage the study of programming language in a simplified and easy to understand environment? Please explain

From these interviews, it was suggested that students' biggest concern was the feeling of intimidation in learning a programming language. Students in the SLA-aBLe course sections believed that teaching programming using SLA techniques was helpful to their learning. Students who already have second language learning experience especially confirmed this during the interview. Students indicated more engagement with the online interactive video, compared to the topics that were presented in a traditional non-interactive format. The captions in the videos help students understand the specific terms. Music does not play an important role in the video design. They pointed out that the tiered examples in the videos and tiered quiz questions eased their anxiousness and helped their comprehension of the materials. Students expressed a desire to flip all topics to the SLA-aBLe format. Students also commented on the laboratory sessions, indicating that the "think, pair, share" activities encouraged collaboration which was helpful to learning and comprehension. Students would rather take the discussion board as an open source information system than use it as an online discussion area.

Conclusion and Future Work

This paper presented a two-year study of the SLA-aBLe project that was implemented in two institutions. The study tested the hypothesis that the use of cognitive frameworks in second language acquisition in the development of a blended learning experience for a programming language can improve engagement and the learning experience of engineering students. Quantitative and qualitative data were collected and analyzed to support the evidence. The first research question was answered by conducting perceived workloads of students in SLA-aBLe and non-SLA-aBLe sections. For the workload study, students reported significantly lower mental demands, lower temporal demands, lower effort, and lower frustration in SLA-aBLe section. The second research question was answered by running a t-test of independence on students' final grade in SLA-aBLe sections and non-SLA-aBLe sections. There was no significant difference found between the SLA-aBLe and non-SLA-aBLe sections, however there were more A and B grades and less failing grades in SLA-aBLe sections than those in non-SLA-aBLe section. The third research question was answered by analyzing face-to-face interviews in three semesters. From 24 interviews conducted, all indicated effectiveness of SLA-aBLe design, which includes interactive videos with captions, tiered examples, and questions online, and collaborative learning in the lab. Positive results in the pilot study let researchers believe that SLA-aBLe is a promising approach, which can help students learn programming language in an easy-to-understand environment.

Acknowledgements
The authors are grateful for the support provided by the National Science Foundation, Division of Engineering Education and Centers, grant number EEC 1441825. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. The authors also would like to acknowledge the effort from Dr. Matthew Verleger, who helped conduct the project in his classes, Dr. James Pembridge who offered suggestions on the project design and implementation, Dr. Joseph R. Keebler who helped analyzed the survey results, and the support from the Institution Research at Embry-Riddle Aeronautical University who conducted and collected the survey data for this project.

References

[1] Janet Siegmund *et al.*, “Understanding Understanding Source Code with Functional Magnetic Resonance Imaging,” in *Proceedings of the 36th International Conference on Software Engineering*, Hyderabad, India, 2014, pp. 378–389.

[2] R. Ellis, *The Study of Second Language Acquisition*. Oxford University Press, 1994.

[3] S. D. Krashen, *Second language acquisition and second language learning*. Oxford: Pergamon Press, 1981.

[4] K. J. Krahnke, “Principles and Practice in Second Language Acquisition,” *TESOL Q.*, vol. 17, no. 2, pp. 300–305, Jun. 1983.

[5] S. D. Krashen and T. D. Terrell, *The Natural Approach: Language Acquisition in the Classroom*. The Alemany Press, 1983.

[6] J. N. Williams, “Memory, Attention, and Inductive Learning,” *Stud. Second Lang. Acquis.*, vol. 21, no. 1, pp. 1–48, 1999.

[7] C. Frederick and L. Sun, “Work in Progress: Using Second Language Acquisition Techniques to Teach Programming-Results from a Two-Year Project,” in *2017 ASEE Annual Conference & Exposition*, Columbus, OH, 2017.

[8] Sun, Lulu, Frederick, Christina, Ding, Li, and Rohmeyer, Rebecca, “The Application of Second Language Acquisition to Programming Language Study: American Society for Engineering Education,” in *2017 ASEE Annual Conference & Exposition*, Columbus, OH, 2017.

[9] L. Sun, C. Frederick, P. Sanjuan Espejo, and R. Cunningham, “Can We Teach a Programming Language as a Second Language?,” *Comput. Educ. J.*, vol. 7.7, no. 3, p. 105, 2016.

[10] C. Frederick, L. Sun, C. Liron, M. Verleger, R. Cunningham, and P. Espejo, “Implementation and Evaluation of a Second Language Acquisition-Based Programming Course,” in *2016 ASEE Annual Conference & Exposition*, New Orleans, LA, 2016.

[11] L. Sun and C. Frederick, “Applying Second Language Acquisition to Facilitate a Blended Learning of Programming Languages,” in *2015 ASEE Annual Conference & Exposition*, Seattle, WA, 2015.

[12] L. Gu and L. Sun, “Motivating Student’s Programming Language Study by Applying Second Language Acquisition,” in *ASEE Southeastern Section Conference*, 2018.

[13] D. Marsh, *Blended learning creating learning opportunities for language learners*. Cambridge Univeristy Press, 2012.

[14] A. Azemi and L. Pauley, “Teaching the introductory computer programming course for engineers using Matlab and some exposure to C,” in *2006 ASEE Annual Conference & Exposition*, Chicago, IL, 2006.

[15] G. Bjedov and P. K. Andersen, “Should freshman engineering students be taught a programming language?,” in *Proceedings of Frontiers in Education Conference, 1996. FIE ’96. 26th Annual Conference*, 1996, vol. 1, pp. 90–92 vol.1.

[16] M. E. Herniter, D. R. Scott, and R. Pangasa, “Teaching programming skills with MATLAB,” *Comput. Educ. J.*, vol. 12, no. 1, 2002.

[17] L. Sun, M. Kindy, and C. Liron, “Hybrid Course Design: Leading a New Direction in Learning Programming Languages Paper presented at the 2013 American Society of Engineering Education Annual Conference & Exposition,” in *2012 ASEE Annual Conference & Exposition*, San Antonio, TX, 2012.

[18] L. A. Smith and E. T. Smith, “Using Camtasia to Develop and Enhance Online Learning: Tutorial Presentation,” *J Comput Sci Coll*, vol. 22, no. 5, pp. 121–122, May 2007.

[19] S. G. Hart, “Nasa-Task Load Index (NASA-TLX); 20 Years Later,” *Proc. Hum. Factors Ergon. Soc. Annu. Meet.*, vol. 50, no. 9, pp. 904–908, Oct. 2006.

[20] A. Cao, K. K. Chintamani, A. K. Pandya, and R. D. Ellis, “NASA TLX: Software for assessing subjective mental workload,” *Behav. Res. Methods*, vol. 41, no. 1, pp. 113–117, Feb. 2009.

[21] S. G. Hart and L. E. Stavenland, “Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research,” in *Adv. Psychol*, vol. 52, 1988, p. 139.