Chang, M. et al. (Eds.) (2019). Proceedings of the 27th International Conference on Computers in Education. Taiwan: Asia-Pacific Society for Computers in Education

Exploring Common Code Reading Strategies in Debugging

Christine Lourrine TABLATIN[a,b*]
[a]*Ateneo de Manila University, Philippines*
[b]*Pangasinan State University, Urdaneta City, Philippines*
*tablatinchristine@gmail.com

Abstract: Code reading is a prerequisite of program comprehension which is a fundamental task in software development. Strategies employed on code reading affect the programmer's success rate of understanding tasks such as debugging. However, there is still limited knowledge about the code reading strategies used by students while performing bug finding task. In this paper, the author describes a summary of her research on novice programmer debugging skills using eye tracking data as a methodology. Eye tracking data were extracted and analyzed using visual effort metrics and sequential analysis of scanpaths using a clustering algorithm to determine common code reading patterns. The author's research findings revealed differences on the code reading patterns and code reading strategies of high and low performing students. Empirical evaluation on the effectiveness of the strategies used by high performing students was also conducted which suggests that by teaching these strategies to students, improved debugging performance can be observed.

Keywords: Eye tracking, debugging code reading patterns, sequential analysis, code reading strategies.

1. Introduction

Debugging is a necessary skill for students learning programming but can be a major stumbling block, particularly for novices. Hence, it is imperative that research be conducted to discover new methods or strategies which could help students improve their debugging and code comprehension skills. One of the interesting approaches to study how programmers read code, is the use of eye tracking data. Eye tracking is a technique that provides a direct measure of the visual attention of a programmer, which is far less intrusive compared to the more established think-aloud protocol. Recent studies have used eye tracking to uncover programmer's cognitive processes while performing tasks such as program comprehension (Chandrika and Amudha, 2017, Jbara and Feitelson, 2016, Lin, et al., 2016). While efforts have been made to understand how programmers read code, we still have limited knowledge on code reading patterns. Further studies are needed to come up with general code reading patterns and visual strategies of different comprehension tasks. This study aimed to analyze and interpret the code reading patterns and infer strategies of students while performing code comprehension to detect bugs.

2. Theoretical Framework

Eye tracker is used to capture visual attention by collecting eye-movement data of a participant while performing a task and have recently become a common tool used to perform empirical studies in programming (Sharafi, et al., 2015). Increased use of eye tracking data is based on the assumptions relating to eye-movements and comprehension. Eye movements are closely linked to the allocation of attention and the frequency of fixation provides information on the importance of the element in the visual stimulus. Duration of fixation on the other hand, provides information on the complexity of the visual stimulus (Nivala, et al., 2016). Using these visual effort metrics, we can identify the visual attention pattern of a subject while performing comprehension task. Further, analyzing scanpath as a whole entity could draw explicit conclusions on the nature and interpretation of the cognitive processes which can be used to infer code reading strategies.

3. Methods

Eye tracking data of students performing bug finding task from four universities in the Philippines were recorded and stored in a CSV file format. The data were then segmented to extract the timestamp, fixation location and fixation duration of the eye gazes which were used for the analysis. The OGAMA (Open Gaze and Mouse Analyzer) software was used to get the coordinates of the Areas of Interest (AOIs) of the stimulus. The fixation points were then mapped to these AOIs to construct the individual scanpaths. To determine the visual attention patterns, fixation counts and fixation durations were calculated for each AOIs. Sequential analysis of the generated scanpath was performed using Scanpath Trend Analysis with a tolerance to reveal common code reading patterns and code reading strategies.

4. Major Findings and Achievements, to date:

Initial work on code reading patterns using Scanpath Trend Analysis (Tablatin and Rodrigo, 2018) revealed that high performing students use varied code reading patterns while low performing students follow similar patterns. Further, STA has weak performance in generating the common scanpath of high performing students. This may be due to the fact that high performing students used varied code reading patterns. To account for the individual differences in code reading exhibited by high performing students, STA with a tolerance was used (Tablatin and Rodrigo, 2018). This algorithm generated common scanpaths that are more similar to the individual scanpaths which means it performs better than the STA without a tolerance. STA with tolerance was used in the analysis of the dataset of my dissertation to identify the common code reading strategies employed by the students.

5. Challenges and Questions

One of the challenges of conducting this study was making inferences on the cognitive processes of the group of students based on their code reading patterns. Further, the type of injected errors and the complexity of programs are different from each stimulus, which made it difficult to identify code reading strategies specific to the type of programs. The errors and the complexity of programs may have an effect on the code reading patterns. Thus, more eye tracking experiments using programs of the same type of errors and code complexity may be conducted to validate the results of this study.

References

Chandrika. K. R., and Amudha, J. (2017). An Eye Tracking Study to Understand the Visual Perception Behavior while Source Code Comprehension. International Journal of Control Theory and Applications. International Science Press vol. 10(19)

Jbara, A., and Feitelson, D. G. (2016). "How Programmers Read Regular Code: A Controlled Experiment Using Eye Tracking," 2015 *IEEE 23rd International Conference on Program Comprehension*, Florence, pp. 244-254. doi: 10.1109/ICPC.2015.35

Lin, Y. T., Wu, C. C., Hou, T. Y., Lin, Y. C., Yang, F. Y.,& Chang, C. H. (2016). Tracking students' cognitive processes during program debugging—An eye-movement approach. *IEEE transactions on education, 59(3)*, 175-186. Retrieved from https://ieeexplore.ieee.org/document/7312518/

Nivala, M., Hauser, F., Mottok, J., and Gruber, H. (2016). Developing visual expertise in software engineering: An eye tracking study. 2016 IEEE Global Engineering Education Conference (EDUCON), Abu Dhabi, 2016, pp. 613-620. doi: 10.1109/EDUCON.2016.7474614

Sharafi, Z, Soh, Z., and Guéhéneuc, Y-G. (2015). A systematic literature review on the usage of eye- tracking in software engineering. Information and Software Technology 67 (2015), 79–107.

Tablatin, C. L. S. (2018). Analysis of Static Code Reading Patterns. In *Proceedings of the 18th Philippine Computing Science Congress*. (pp. 32-43). Retrieved from https://drive.google.com/file/d/1PGrCmNIv4_mEot5SnBEhCngsHGUMQrkl/view

Tablatin, C. L. S. (2018). Identifying Common Code Reading Patterns using Scanpath Trend Analysis with a Tolerance. In *Proceedings of the 26th International Conference on Computers in Education*. Retrieved from http://icce2018.ateneo.edu/wp-content/uploads/2018/12/C3-08.pdf