http://www.diva-portal.org

This is the published version of a paper presented at *Eye Movements in Programming*.

Citation for the original published paper:

Kontogiorgos, D., Manikas, K. (2016)
Towards identifying programming expertise with the use of physiological measures.
In: University of Eastern Finland

N.B. When citing this work, cite the original published paper.

Permanent link to this version:
http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-194351

Towards identifying programming expertise with the use of physiological measures

Dimosthenis Kontogiorgos
Department of Computer Science
University of Copenhagen, Denmark
dimo@di.ku.dk

Konstantinos Manikas
Department of Computer Science
University of Copenhagen, Denmark
kmanikas@di.ku.dk

ABSTRACT

In this position paper we propose means of measuring programming expertise on novice and expert programmers. Our approach is to measure the cognitive load of programmers while they assess Java/Python code in accordance with their experience in programming. Our hypothesis is that expert programmers encounter smaller pupillary dilation within programming problem solving tasks. We aim to evaluate our hypothesis using the EMIP Distributed Data Collection in order to confirm or reject our approach.

Keywords

code comprehension, programming expertise, pupillometry

1. INTRODUCTION

Recently, neuroscientists used functional magnetic resonance imaging (fMRI) to identify and measure intelligence, but also to predict individuals' cognitive behaviour during tasks [4]. Considering the ethical implications, researchers have tried numerous times to extract information from subjects' cognitive activity in order to classify them according to their intelligence or expertise.

Programmers today work in several programming languages and often develop expertise in a number of them. In the industry, however, seniority is usually dependent on knowledge in a specific domain (i.e. data structures, algorithms), rather than a specific programming language. Therefore, programming experience is not necessarily related to expertise. A novice programmer might experience different challenges than an expert, while comprehending code [6].

In software engineering experience is an important parameter within programming comprehension. It is defined as "the amount of acquired knowledge regarding the development of programs, so that the ability to analyse and create programs is improved" [11]. There has not been, however, an agreed way of measuring programming experience.

Psychologists in the 60s found a correlation between cognitive activity and pupillary dilation. There can be various reasons on why our pupils can encounter dilation, and one of them is our cognitive workload. Using pupillary response we can therefore measure our cognitive activity.

In this position paper, we examine how programming expertise influences the programmer's cognitive activity. We are interested in identifying connections between expertise in a specific technology (i.e. Java, Python) and physiological measures. Several studies have measured cognitive activity using physiological means such as pupillometry. We propose experimenting with novice and experts in Java/Python programming to investigate how experience in a technology differentiates within physiological measures.

2. RELATED WORK

2.1 Programming expertise

Independently of acquired knowledge, intelligence is a parameter used to measure one's capacity in reasoning and problem solving [4]. However, in domain-oriented tasks, the amount of acquired knowledge is an important ability to analyse and solve problems. Knowledge is a combination of our long-term memory and our working memory. Within cognitive load theory, working memory is described as the number of elements we can hold while processing a task. A large number of material elements may be a single element for one with expertise in a particular task [12].

Sweller et al. derived three fundamental types of measuring cognitive load: (1) subjective (individuals' self-reporting), (2) physiological (heart rate, brain activity, pupillary dilation) and (3) task performance [12]. Within program comprehension, different means of measuring cognitive load or programming experience have been utilised, mainly including self reporting or subjective techniques, but also physiological measures (fNIRS) [11, 13].

To the extent of our knowledge, there has not been a correlation between cognitive effort and programming experience while using pupillary response, and several studies have shown its precision as means of measurement. Whether we should use such measures to identify individuals' expertise independently of their self reporting is posed as an open question.

2.2 Pupillary dilation

Psychology. Beatty found that task-evolved pupillary response is a good indicator of cognitive load and cognitive activity [2]. It has been used in several disciplines, as pupillary dilation is a very precise and non-invasive manner of measuring cognitive activity and can be easily measured [9]. In terms of expertise, Ahern and Beatty showed a correlation between intelligence and pupillary response; more intelligent subjects showed smaller task-evolved pupillary dilation in arithmetic problems [1].

Computer science. Klinger used pupillometry to assess visual interfaces by the amount of cognitive load they require from a user [8]. Maier et al. investigated how particular UML diagram layouts affect cognitive load [10]. Within programming tasks, Iqbal investigated how well pupil size

correlates with mental workload [7]. Fritz et al. assessed and predicted task difficulty on expert programmers [5].

3. METHOD

In this proposed exploratory study, we intend to measure programmers' cognitive load and mental effort during programming tasks, in order to identify and measure programming expertise. We argue that measures such as self reporting [11] might pose a potential threat to the validity of measuring experience and attempt to correlate physiological measures to programming expertise.

Previous studies in this workshop have experimented with patterns of reading behaviour within novice and expert programmers. To investigate individual behaviour and detect experts from novices, we apply pupillometry measures in order to identify cognitive effort during programming tasks.

3.1 Research design

Hypothesis. Our hypothesis is that expert programmers experience significantly less cognitive load within programming tasks than novice programmers. In order to measure cognitive activity we use pupillary response to various types of task difficulty. We therefore form two research questions:

RQ1. Can we use pupilometry to identify/measure programming expertise?

RQ2. Does programming expertise influence pupillary dilation within programming tasks?

3.2 Distributed data

Using the Distributed Data Collection from the workshop, we attempt to validate or reject our hypothesis within program comprehension. The data includes gaze interactions from novice and expert programmers while assessing code in Java/Python. Our approach is to combine raw sensor data such as fixation, pupil size and gaze interaction (dependent variables) with demographics including age, gender, years of experience and education (independent variables).

4. EXPECTED RESULTS

In programming (similarly to reading) processing of visual information only occurs during eye fixations [3], where we also expect to encounter pupillary dilation in terms of cognitive processing. We attempt to assess the ability to read and understand code, as it differentiates from the ability to immediately recognise its flaws or various imperfections.

Researchers have tried to identify criteria that can be used to measure experience on novice or expert programmers, but to our knowledge no attempts were made to do so using pupillary response. To address our research questions, we experiment with pupil dilation in programming tasks and propose the measurement of programming expertise by measuring cognitive activity on novice and expert programmers.

5. REFERENCES

[1] S. Ahern and J. Beatty. Pupillary responses during information processing vary with scholastic aptitude test scores. *Science*, 205(4412):1289–1292, 1979.

[2] J. Beatty. Task-evoked pupillary responses, processing load, and the structure of processing resources. *Psychological bulletin*, 91(2):276, 1982.

[3] T. Busjahn, C. Schulte, B. Sharif, Simon, A. Begel, M. Hansen, R. Bednarik, P. Orlov, P. Ihantola, G. Shchekotova, and M. Antropova. Eye tracking in computing education. In *Proceedings of the tenth annual conference on International computing education research*, pages 3–10. ACM, 2014.

[4] E. S. Finn, X. Shen, D. Scheinost, M. D. Rosenberg, J. Huang, M. M. Chun, X. Papademetris, and R. T. Constable. Functional connectome fingerprinting: identifying individuals using patterns of brain connectivity. *Nature Neuroscience*, (October):1–11, 2015.

[5] T. Fritz, A. Begel, S. C. Müller, S. Yigit-Elliott, and M. Züger. Using psycho-physiological measures to assess task difficulty in software development. In *Proceedings of the 36th International Conference on Software Engineering*, pages 402–413. ACM, 2014.

[6] M. Hansen, A. Lumsdaine, and R. L. Goldstone. An experiment on the cognitive complexity of code. *In Proceedings of the Thirty-Fifth Annual Conference of the Cognitive Science Society, Berlin, Germany*, 2013.

[7] S. T. Iqbal, X. S. Zheng, and B. P. Bailey. Task-evoked pupillary response to mental workload in human-computer interaction. In *CHI'04 extended abstracts on Human factors in computing systems*, pages 1477–1480. ACM, 2004.

[8] J. M. Klingner. *Measuring cognitive load during visual tasks by combining pupillometry and eye tracking.* PhD thesis, Stanford University, 2010.

[9] B. Laeng, S. Sirois, and G. Gredebäck. Pupillometry a window to the preconscious? *Perspectives on psychological science*, 7(1):18–27, 2012.

[10] A. Maier, N. Baltsen, H. Christoffersen, and H. Störrle. Towards diagram understanding: A pilot study measuring cognitive workload through eye-tracking. In *International Conference on Human Behaviour in Design 2014*.

[11] J. Siegmund, C. Kästner, J. Liebig, S. Apel, and S. Hanenberg. Measuring and modeling programming experience. *Empirical Software Engineering*, 19(5):1299–1334, 2014.

[12] J. Sweller, J. J. Van Merrienboer, and F. G. Paas. Cognitive architecture and instructional design. *Educational psychology review*, 10(3):251–296, 1998.

[13] M. P. Uysal. Evaluation of learning environments for object-oriented programming: measuring cognitive load with a novel measurement technique. *Interactive Learning Environments*, pages 1–20, 2015.