システム創成工学専攻
情報システムコース

Department of Systems Innovation
Advanced Information System Course

令和元年度 専攻科特別研究論文

脳波と視線の同時計測による
プログラム理解状態の把握

Synchronized Analysis of Eye Movement and EEG
during Program Comprehension

指導教員名　上野 秀剛 准教授
論文提出者名　石田　豊実

独立行政法人 国立高等専門学校機構
奈良工業高等専門学校 専攻科
National Institute of Technology, Nara College
Faculty of Advanced Engineering

脳波と視線の同時計測による
プログラム理解状態の把握

Synchronized Analysis of Eye Movement and EEG
during Program Comprehension

石田　豊実
Ishida Toyomi

独立行政法人 国立高等専門学校機構
奈良工業高等専門学校 専攻科 システム創成工学専攻 情報システムコース
大和郡山市矢田町 22 番地（〒 639-1080）
National Institute of Technology, Nara College, Faculty of Advanced Engineering
22 Yata-cho, Yamatokoriyama, Nara 639-1080, Japan

Abstract: Appropriate support for program comprehension contributes to efficient software development. Several previous studies used bio-information such as brain activity to classify the inner-state of programmer without interruption. In this paper, we measure programmer's brain waves and eye movement simultaneously while they comprehend the source code. In the experiment, we analyze difference of time-series brain wave features between success/failure for source code comprehension task. The result of the experiment showed the participants who success the source code comprehension significantly increased power spectrum of α wave with the time passage. Also the eye movements of the succeed participants shift their focus of fixation from specification to source code in early time. Synchronized analysis of failed programmer shows similar but slow pattern of EEG and eye movement changes compared with succeed programmer.

Keywords: Program comprehension, EEG, Eye movement

関連業績リスト

1. 石田豊実，上野秀剛: “プログラム理解時における脳波特徴の把握を目的とした時系列分析の試み”，第 200 回ソフトウェア工学研究会，Vol.200， No.2， pp.1-8，(2018).

2. Toyomi Ishida，Hidetake Uwano: “Time Series Analysis of Programmer’S Eeg for Debug State Classification”，In the 5th Edition of the Programming Experience Workshop (PX 2019).

3. Toyomi Ishida，Hidetake Uwano: “Synchronized Analysis of Eye Movement and Eeg during Program Comprehension”，In Eye Movements in Programming 2019 (EMIP 2019).

目次

図目次

表目次

1. Introduction

Appropriate support for program comprehension contributes to efficient software implementation and debugging. Classification of programmer's states during comprehension enables us to real-time support based on their situations; e.g. visualization of code snippet that uses a certain variable the programmer has an attention. Many studies proposed process model of program understanding during programming and/or debugging.

Mayrhauser and Vans compared six program comprehension models [1]. Xu [2] classify activities during program comprehension into six processes (knowledge, comprehension, application, analysis, synthesis and evaluation) based on Bloom's taxonomy, one of the intellectual activity classification [3]. These studies organize a set of subprocesses in program comprehension process from view point of logical architecture.

Our long term research goal is to empirically analyze the program comprehension process using bio information measurement. Bio information is a metric which is measured from human body and have relation with thinking and psychological state. Program comprehension is mainly performed mentally, so it is hard to understand their "state of understanding" from visual observation. In Software Engineering, many studies use bio information to analyze programmers' thinking and psychological state [4, 5].

Program comprehension requires understanding of operation in target source code (how it works) and requirement specification of the program (what to do). Understanding the operation of source code involves comprehending process flow, variables role, method functionality, the syntax of programming language, API architecture, and other miscellaneous activities.The understandings are accomplished by reading each line of the source code and the requirement specifications, then lead to the understanding of block, method, class and entire program. In this paper, we define a "micro process" as an activity to line-wise comprehension of source code and other software documents, such as variable (and method) declaration, calculation, conditional expression, method call, requirement explanation, etc. Programmers during a comprehension process perform a continuous set of micro processes to lines of source code and other documents, then understand a larger architecture such as process flow and caller-callee relationship by combining the understandings from micro processes. Therefore, line-wise analysis allows us to fine-grained understandings of programmer's comprehension process; Such knowledge contributes a real-time assist of the programmer based on a state of understanding.

Micro processes during program comprehension change frequently based on a literature in lines. These different micro processes activate different brain functionality (such as syntax recognition, calculation, memorization) in different intensities. In this paper, authors measure EEG (electroencephalogram, i.e. brain waves) during the comprehension process. Frequency bands of EEG, such as α wave and β wave reflect mental (emotional and intellectual) states and the brain activity [6]. We assume the brain activities in each micro process are reflected rapidly to the EEG, because EEG has a high time resolution. Therefore, time series changes of programmer's EEG may synchronize with their reading of lines.

Eye movements measurement is well used technique to analyze the comprehension process in programming [7, 8, 9]. Programmers read source code based on their comprehension strategies, hence the strategies are reflected in their eye movements. That is, eye movements reflect how programmers try to understand the source code through sequence of micro processes, while the EEG reflects the result of each micro process. For example, we can support appropriately for programmers if we divide EEG by eye movement as Fig.1.1.

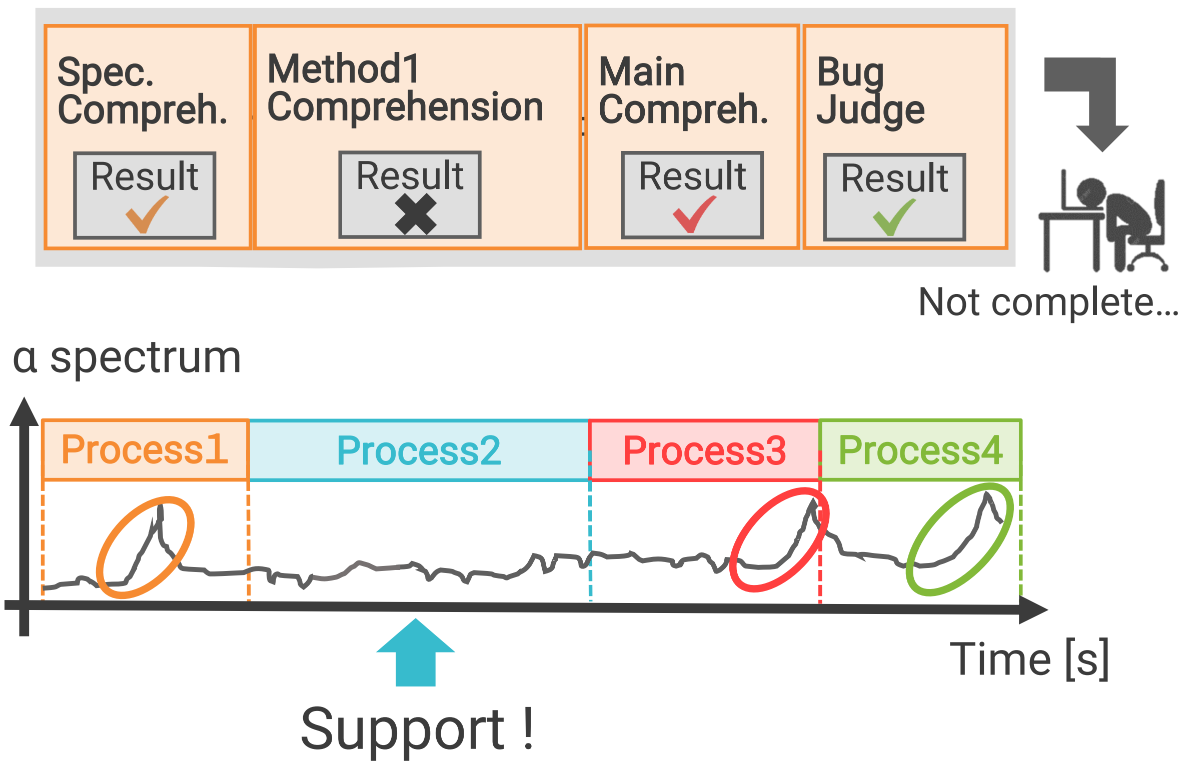

Fig.1.1: Classification by combining EEG and eye movement

In this paper, we measure programmer's EEG and eye movements during comprehension process simultaneously. We evaluate whether the EEG and the eye movements reflect the process of comprehension or not from pilot experiment. As mentioned above, time series changes of programmer's EEG may synchronize with their reading during comprehension. Our pilot experiment examines change of EEG and eye movements occur simultaneously. Also a power spectrum of EEG and eye movement are compared between who succeeds program understanding task and who fails. We examine two research questions from an experiment:

- RQ1: Are change of EEG and eye movement during program comprehension synchronize with any comprehension process?

- RQ2: Are time series changes of EEG and eye movement different between success and failure of understanding task?

2. Related Work

Some bio-information is used for quantitative analysis of program comprehension processes in prior works. Siegmund et al. investigated the activation at each brain portion during program comprehension by using fMRI [10, 5]. The fMRI is a device that measures brain activity from blood flow changes in a human head. In the experiment, participants read short source code snippets for two types of error detection, 1) syntax error and 2) logic error. As the results show participants who understand the source code had activated brain regions that related to problem solving, memorization, and sentence comprehension.

The fMRI has a high spatial resolution compares with other brain measurement devices such as EEG, and therefore is suitable for understanding which brain region is activated in program comprehension. On the other hand, the fMRI is not suitable for measurement in practical environment because the device requires participants to lie on the device.

The combination of EEG and eye movements is well used in research of program comprehension. Fritz et al. predicts the programmer's states and task characteristics using machine learning with combination of several bio-information [11]. Their method predicts the subjective difficulties (high and low) from EEG (α, β, and other Frequency bands), eye movements (pupil size, saccades, and fixations), and EDA (skin conductance). Zuger et al. predicts the programmer's condition which good for interruption [8]. Muller et al. builds a psychological model for estimate the programmer's emotion (concentration and happiness) [9].

These researches show that EEG and eye movements are useful metrics for analysis and prediction for programmer's states. In this paper, authors analyze the difference of brainwave features during program comprehension between who succeed to comprehend and who fail. Especially we focus on time series analysis of EEG; change of α wave in comprehension task is analyzed. Both EEG and eye movements have a high time resolution, hence the combined analysis is suitable method to understand micro processes in program comprehension. Also, devices that record the EEG and the eye movements are easy to apply for programmers during their working or students in class compared with fMRI and other brain activity measurement devices.

3. EEG and Eye Movements

3.1 EEG

An EEG is an electrical activity that arises from brain and recorded through electrodes placed on the scalp. Brain electric potential is measured as difference of two electrodes [12]. Each electrode is placed on a point specified by the International 10-20 system shown in Figure 3.1 [13]. The International 10-20 system designates 19 electrode placements except the electrodes defining ground potential.

The purpose of measurement determines place and number of electrodes. There are two electroencephalographic derivation methods; standard and bipolar method. The standard method is used when two electrodes are placed near each other. The bipolar method is used to measure the difference of two specific positions and remove the irrelevant background components. EEG waveforms are generally classified according to their frequency [12]. The frequency is calculated by FFT (Fast Fourier Transform), or STFT (Short Time Fourier Transform). The familiar classification of waveforms is following bandwidths.

- δ wave: 0.1 - 4 Hz
- θ wave: 4 - 8 Hz
- α wave: 8 - 13 Hz
- β wave: 13 - 30 Hz
- γ wave: 30 - 100 Hz

EEG frequency differs depending on different behavior and mental conditions of the brain [6]. For example, α wave appears strongly when a person is relaxed or concentrating. When the person is thinking or stressed, β wave becomes stronger

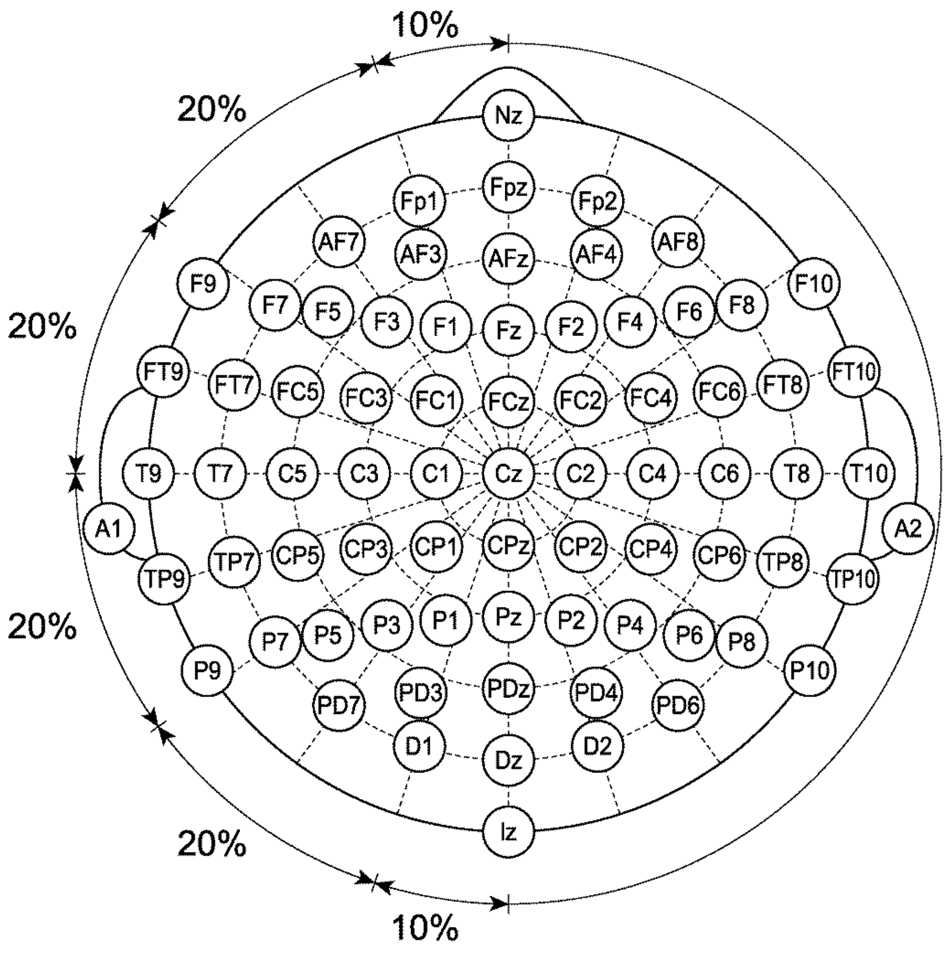

Fig.3.1: International 10-20 system of electrode placement

and α wave is weakened relatively. The EEG is also used to evaluate mental (not emotional but intellectual) activities [11, 8]. In this paper, we use the EEG as metric to classify programmers who understand a program behavior and/or succeed to judge the program contains a bug. For example, we can classify success/failure of comprehension by analyzing programmers' EEG during comprehension as Fig. 3.2.

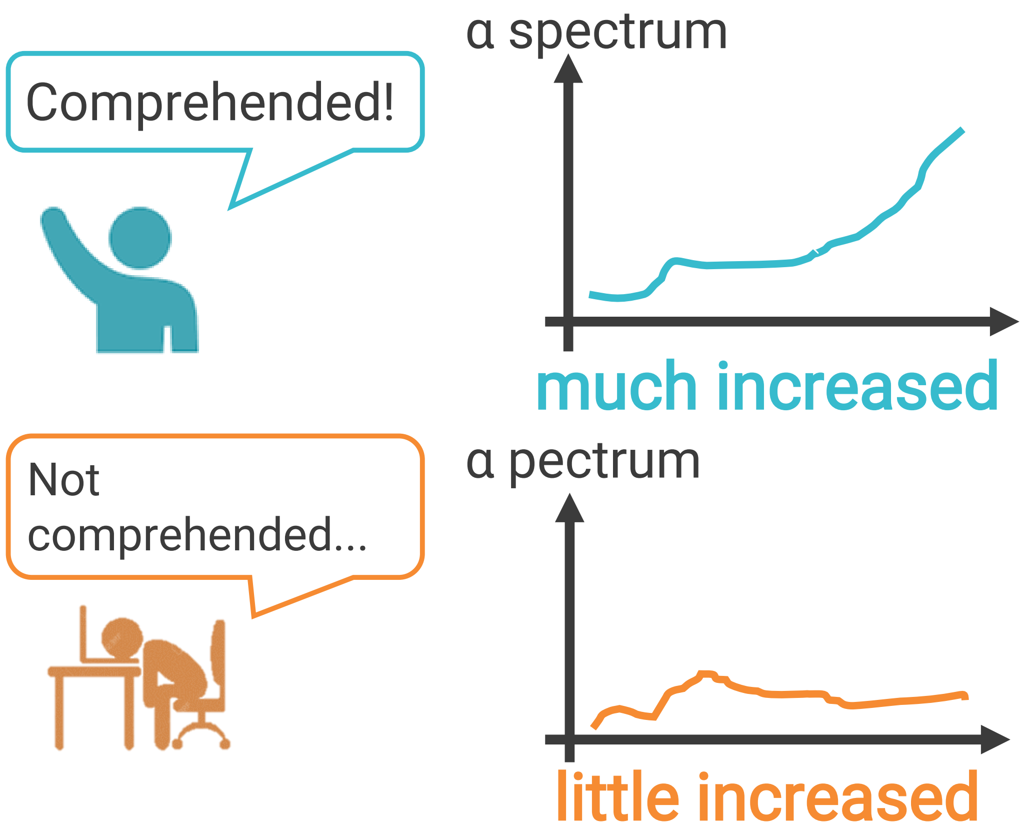

Fig.3.2: Classifying of comprehension result using EEG

3.2 Eye Movement

Eye movement is recorded as a series of the coordination (x and y) on display that calculated from the eyeball movement [14]. Analyzing the eye movements shows us where the participant is looking at and how long each gaze takes place. Eye movement is used to analyze the difference between novices and experts, especially in cognitive engineering. The eye movement analysis is also used to program comprehension and debugging research. Behroozi et al. distinguish who understands source code via eye movements analysis [15]. Busjahn et al. found differences in eye movement while reading source code between experts and novices [16]. Software documents such as

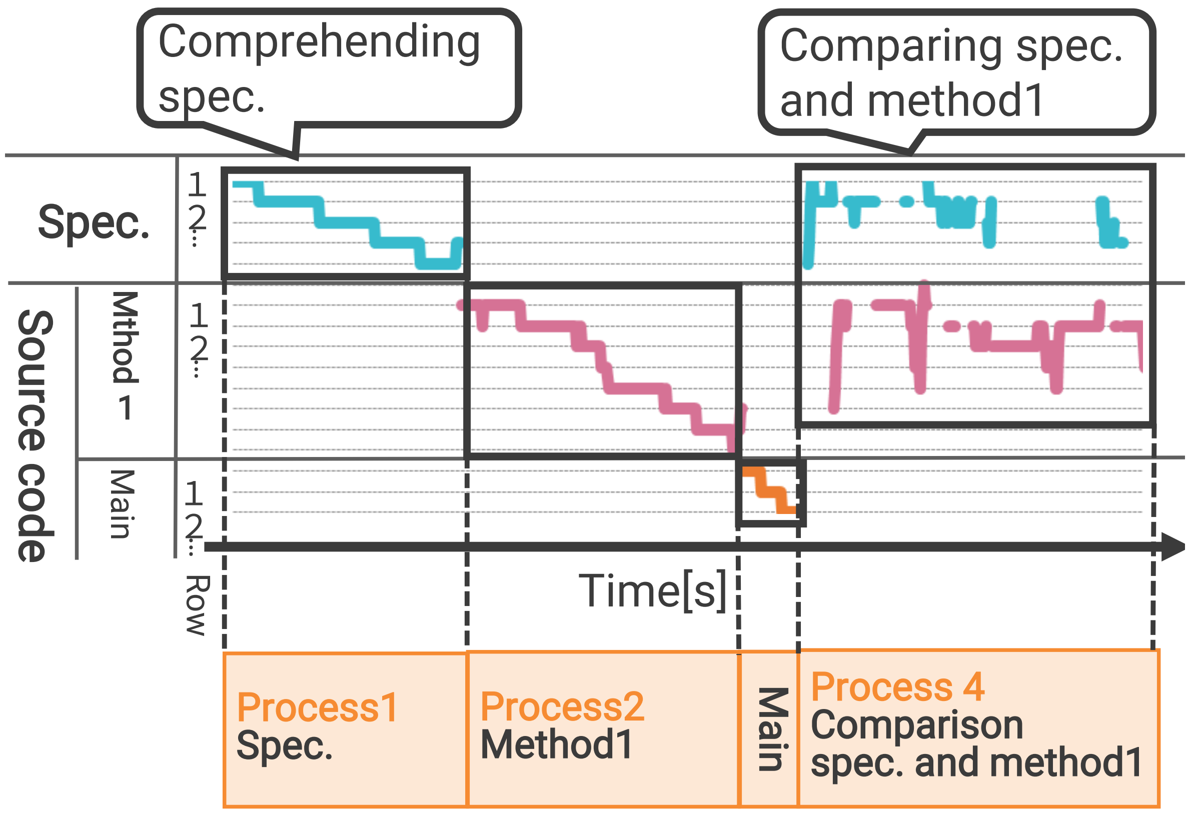

Fig.3.3: Division of comprehension process using eye movement

requirement specifications or source code consist of lines. Developers read each line to comprehend entire document and correspondence between documents. The analysis of eye movements in software engineering research enabled us to understand the line-wise analysis of software reading/comprehension process. For example, we can divide comprehension into each process by analyzing programmers' eye movement during comprehension as Fig. 3.3.

On the other hand, eye movement analysis does not allowed to know the states of understandings. In general, the longer gaze time to one line means that the line is important for entire understanding or difficult to understand meaning of the line. Eye movement analysis can not distinguish them because the eye movement hardly contains “result” of the reading; the programmer understood mean of the line or not. In this research, we combine the eye movement and time series EEG analysis for fine-grained understanding of program comprehension. We believe the combination tells us which part of source code is read by programmer and level of the programmer's understanding to the part, e.g. “She understands variable x stores result of calculation, but not sure how the calculation is work.”

4. Experiment

Five undergraduate students in the information engineering department of our college (All male, aged from 19 to 20) participated in this experiment. All students finished basic course of Java programming at least. Each participant reads Java source code as the debugging tasks. During the task, we record EEG to classify whether participants succeeded or failed tasks.

4.1 Environment

This experiment is performed in a quiet room with only one participants and two experimenters. In order to suppress artifacts due to body movements, the participants sit on a chair with armrests and footrests, and are instructed to reduce their body movement as much as possible.

We use the NeXus-10 MARK II manufactured by Nanotech Image Ltd. as our EEG measurement device. The device measures EEG with 256Hz sampling frequency, and a measured data is transferred to a PC via Bluetooth. Electrodes are located using the standard electrode derivation method. The ground electrode is located at right ear (A2), standard electrode is located at left ear (A1) and measurement electrode is located at back of the head (Pz), which same allocation of our previous study that distinguish whether programmer found an implementation strategy from requirement specification [17]. Figure 4.1 shows appearance of the subject during this experiment.

We use Eye Tracker 4C manufactured by Tobii Ltd. as eye movement measurement device. The device is low cost (less than 200 USD), non-invasive and screen-based eye tracker with 90Hz data acquisition frequency. Several research used the device for eye movement analysis [18], developers' emotion estimation [19], and virtual keyboard

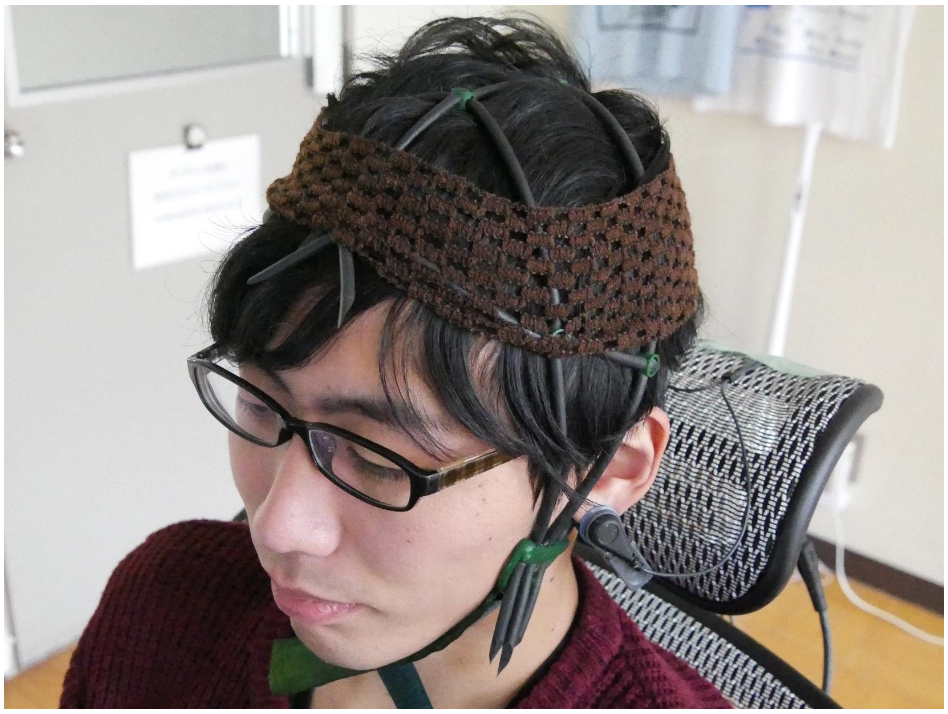

Fig.4.1: Appearance during measuring brainwave

application [20]. The device is installed to task presentation PC, the recorded eye movements are stored to CSV file. At the beginning of the experiment, participant's eye movement is calibrated using setting tool distributed by Tobii Ltd.

We develop the task presentation tool for the experiment. The tool is C# GUI application with three tabbed-windows to display the task materials explained in section 4.2. The display size and resolution for the task presentation are 21.3-inch, 1920 $\times$ 1080. Each material displayed in the tab has a 50 pixel height at each line. Participants click the tab to change the display materials in any time. The tool also records synchronized eye movement and operation (tab switching) for analysis. We use one PC for the EEG device control and recording. Eye tracker control and the experiment tool are ran by another PC. We synchronize EEG and eye movement data using signals which manually input at the start and end of each task.

4.2 Task

Participants read three task materials during the task, 1) specification, 2) source code and 3) question. Table 4.1 shows the list of tasks in the experiment. Specification is a short sentence that explain the purpose of the program written in Japanese. Source code is a written in Java with single class in one file.

The participants answer a question that confirms s/he comprehends the program behavior correctly; i.e. question: "What is the value of variable x at the sixth line in the second loop?" and answer: "x is 7." The question is displayed with the specification and the source code at display, the participant answer the question verbally. The definition of $success$ and $failure$ in understanding step is as follow:

- $success$: A participant answers question within a time limit (2 minutes 30 seconds) and the answer match with prepared one.
- $failure$: An answer of a participant is not match with prepared one, or no answer within the time limit.

The question, specification, and source code are displayed separately via experiment tool explained in section 4.1. Participants can change the displayed materials by click the tabs.

In this paper, we classify the task results into two groups, $success$ and $failure$, hence it is preferred the number of each group is similar. To achieve this, eight difficult tasks and eight easy tasks are prepared. The source code of the easy task consists of only the main method, single loop block, and single conditional branch; participants may success the task within time limit. The source code of the difficult task uses a complex algorithm which has multiple methods, recursive structure; participants may fail the task or exceed the time limit. The order of each task is counterbalanced to minimize the effect of task order.

Table 4.1: Tasks used in the experiment

	difficulty	specification
1	easy	Calculate factorial
2		Search maximum number
3		Judge prime number
4		Search median number
5		Calculate power of number
6		Swap two numbers
7		Judge string match
8		Output reversed string
9	difficult	Tower of Hanoi
10		Count route combination
11		List-up permutation
12		List-up combinations by recursion
13		Count the combination of coins
14		List-up string combination
15		Predict clouds movement
16		Calculate G.C.D. and L.C.M.

4.3 Analysis

EEG in each task is extracted to calculate power spectrum of α wave. Each EEG is divided to 0.2 seconds length, then calculated power spectrum by STFT. In this paper, we simplify the analysis by calculate the average of power spectrum in each five seconds time range from start of the task. The extracted powers include a large individual difference. Therefore, each power spectrum is normalized by the median value of each task. The normalized value means the difference of powers at each time zone during a task.

In this paper, the fixation ratio of each material is calculated by using tab switching

```
        Time,   Task Name,       Tab Name
 18:04:07.01,      task03,  specification
           ⋮
 18:04:12.22,      task03,  specification
 18:04:12.23,      task03,    source code
           ⋮
 18:04:22.99,      task03,    source code
 18:04:23.03,      task03,       question
           ⋮
 18:04:30.80,      task03,       question
           ⋮
```

Fig.4.2: Example of tab switching history

history which is outputted by the task presentation tool. We can identify the material which the participant was reading because the task presentation tool displays each matrial to tabs overlaped placed. Figure 4.2 shows an example of tab switching history. We divide lines of the history to every five seconds in each task, then calculate the percentage of fixation time to each material.

After the above process, EEG and fixation ratio to each task material are combined. Both data have a five second intervals for each task, so the data is displayed in parallel. First, we examine whether power spectrum of α wave and fixation ratio for task materials change simultaneously. To answer the task question correctly, participants must understand the specification and behavior of the source code, then compute the value of variable that designated by the question based on the understanding. Participants read the each material during the understanding, hence the fixation ratio to the target material becomes higher. Participants who succeed the understanding of the material, the fixation ratio will shift to other materials. Here, we suppose that the fixation ratio and EEG change simultaneously with the state of understanding to task materials (RQ1).

Second, we compare *success* and *failure* group to examine that the change of fixation ratio and EEG depend on task succeed/fail. Each group has different state of understanding for task materials. Therefore, we suppose that the fixation ratio and EEG which reflect participant's comprehension depend on task succeed/fail (RQ2).

5. Result and Discussion

We obtained 80 data (16 tasks $\times$ 5 participants) as result of experiment, and every data is used in the analysis. Large body movements and immediate answer (without thinking) is not observed during the experiment. Table 5.1 shows the number of $success$ and $failure$ at each step.

Figure 5.1 and Figure 5.2 show combination of EEG and fixation ratio for three task materials in $success$ and $failure$. At first, focus on fixation of $success$ and $failure$. These figure show the fixation ratio to specification is large at beginning of task, then fixation ratio to the source code increases gradually in both group. After that, fixation to the specification and question keep in some ratio, however allocation rate is different between $success$ and $failure$. $Success$ group has a stable tendency that fixation ratio for question is larger than specification except first two time range (1-5 and 6-10). On the other hand, fixation ratio for specification and question is almost same from 31-35 to 106-110 at $failure$ group.

Secondly, we analyze EEG and Eye movement. In Figure 5.1, fixation ratio to the source code reaches dominant (74.2%) at 16-20 seconds when the α wave in $success$

Table 5.1: $Success$ and $failure$ in the experiment

Participant ID	$success$	$failure$
1	10	6
2	6	10
3	9	7
4	11	5
5	11	5

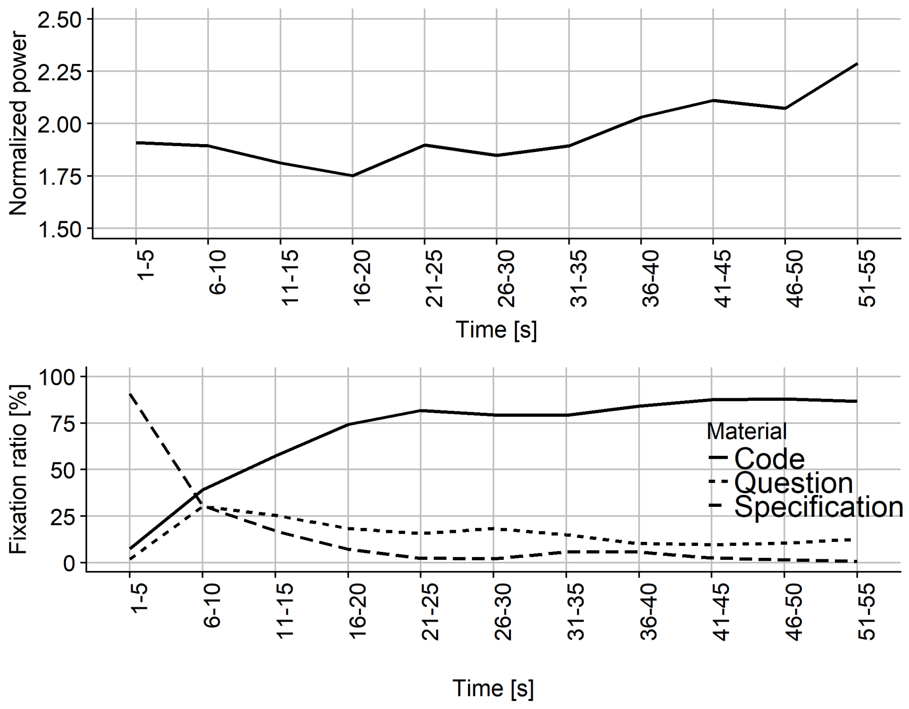

Fig.5.1: Eye movement and EEG in *success* group

starts to increase. Then the fixation ratio for specification and question stay low, at 41-45 time range that EEG increased significantly from task start, 87% of fixation is concentrated to source code; in contrast, only 3% for specification.

It is necessary for participants to comprehend the process of source code and a content of question to answer a question (such as “what is the value if variable x at sixth line in second loop?”) correctly. The comprehension of specification helps participants to understand source code. Also the specification is displayed at the start of each task; therefore, the participants read the specification at first, then shift to the source code for clear understandings. The result of the synchronized analysis suggests that the *success* group understand the specification in early time during task, then proceed their source code comprehension without any difficulties; the successful task progress appears to rapid increase of EEG (α wave).

The result of the *failure* group (figure 5.2) shows similar tendency with *success*; fixation ratio to source code reaches dominant (76.6%) at 41-45 seconds when the α

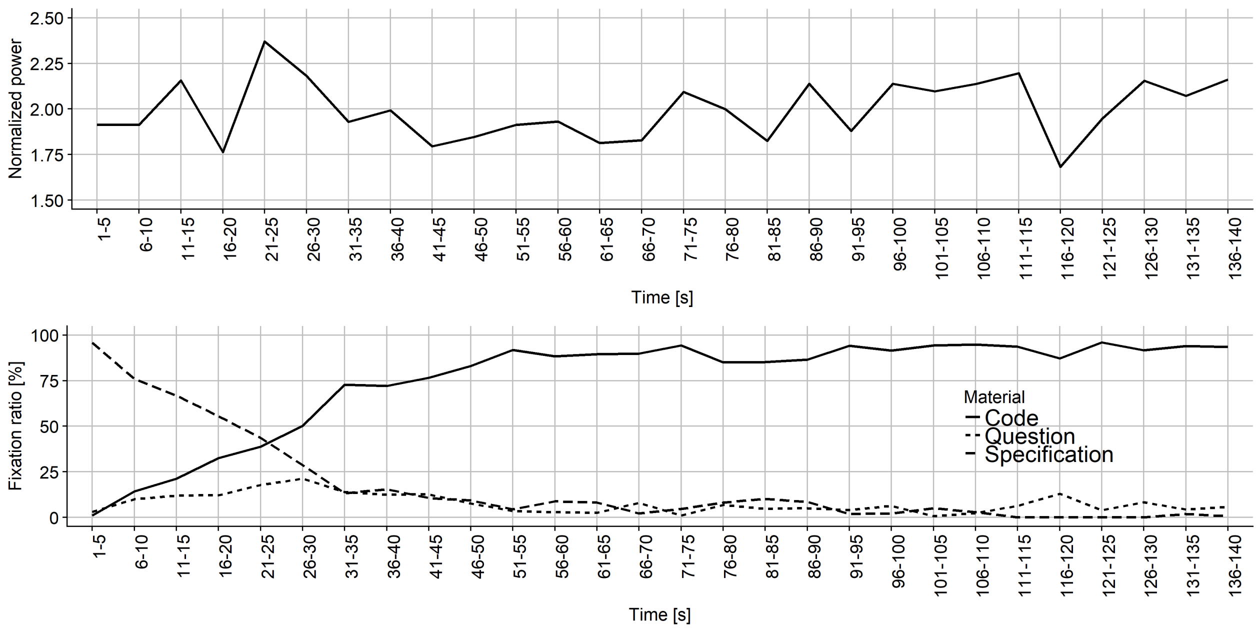

Fig.5.2: Eye movement and EEG in *failure* group

wave starts to increase. At 106-110 seconds that EEG increased significantly from task start, none of fixation for specification is observed.

Synchronized analysis of *failure* shows similar but more slowly change of EEG and eye movement compared with *success*. Until the fixation ratio to source code reaches 75% (EEG start to increase in both group), it takes 15 seconds in *success* and 40 seconds in *failure* from the start of each task (2.67 times); time from the start of EEG increment to significant increase is 26 seconds in *success* and 65 seconds in *failure* (2.6 times). In this experiment, 52% of *failure* task is not completed within the time limit. The result of the synchronized analysis suggests that the *failure* group cannot understand the specification in early stage during task, then proceed their source code comprehension without any clue from specification; the failure task progress appears to slow increase of EEG (α wave).

Figure 5.2 shows that fixation ratio to specification and question changes at 106-110; this is the same time range what EEG shows significant difference in *failure*. After the time range, the fixation ratio to specification is lower than question. In *success*, the fixation ratio to question always exceeds the ratio to the specification except at the beginning of task. In this experiment, question ask the specific value of a variable at certain state. It is difficult to understand what the question ask without comprehension of specification and source code. Therefore, the higher fixation ratio to question than specification means the participant thinks they understand the specification. Same characteristic is observed at 106-110 seconds in $failure$, this may describe some participants understand source code incorrectly, therefore fail to answer the question.

- RQ1: EEG and fixation ratio to source code during program comprehension increased with understand the specification.
- RQ2: Participants who success their understanding have an early increase of α wave compared with participants who fail the understanding.

Therefore, we suppose that the fixation ratio and EEG which reflect participant's comprehension depend on task succeed/fail (RQ2).

6. Conclusion and Future Work

In this paper, we analyzed time series changes of eye movements and EEG during program comprehension. Result of the experiment showed the fixation ratio and EEG changed simultaneously with the comprehension of materials. The α wave started to increase when the fixation to source code is dominant. The fixation ratio and EEG which reflect participant's comprehension depended on task succeed/fail. Synchronized analysis of $failure$ shows similar but slow pattern of EEG and eye movement changes compared with $success$. Also, the result of the synchronized analysis suggests that the $failure$ group cannot understand the specification in early time during task, then proceed their source code comprehension without any clue from specification; the failure task progress appears to slow increase of EEG (α wave). These results surpport our research question RQ1 and RQ2. Synchronized analysis of EEG and eye movements has a possibility as an metric to detect a programmer who cannot understand the program in seconds-scale.

One of our future work is analysis of line-wise eye movement. Time series analysis of fixation ratio to each line in materials suit to combine with fine-grained EEG for detailed micro process understandings. Labeling to each line based on characteristics is another interest research theme. For example, each line of source code is labeled from process type within the line such as variable declaration, calculation, store of process result, method call/return, and condition evaluation. These labels are useful characteristics for pattern mining or machine learning when combine with brain activity measurement such as EEG because the difference characteristics in line (or process-chain of code snippet) will lead the different brain activity. Additional measurement of EEG from different place is also interest.

7. Additional Analysis in Process

7.1 Background

We have demonstrated that the changes in EEG and eye movement data can be synchronized during program comprehension tasks. Technically, subjects' α spectrum in EEG and attention rates on source code were increased in early phases if they succeeded to comprehend a given code snippet while the changes were occurred lately if they failed. Here we use the synchronized changes in EEG and eye moment data to classify developers' success/failure on program comprehension. More dynamic and appropriate support might be enabled if the support system can be aware of developers' mental states from biometric data.

Several recent studies in software engineering domain have combined more light-weight biometric sensors with machine learning techniques to classify mental states of software developers[21]. Frits et al. trained a Naive Bayes classifier on three biometric sensors involving EEG, electrodemal activity (EDA), and eye movements to predict the difficulty of program comprehension tasks[11]. Müller et al. trained a decision tree classifier on EEG, EDA, eye movements, and heart-related data to classify developers' mental states (i.e. stuck or in flow, happy or frustrated) and resulted in around 70 percent classification accuracies[9]. Both of their results suggested that combinations of two or more different biometric data could be useful to estimate developers' mental states or subjective difficulties.

Here we aim to estimate whether a subject succeeded to comprehend a code snippet or not from EEG and eye-movement data that measured simultaneously. In particular, we focus on their time courses and synchronized changes. Our combined analysis would be suitable for understand micro processes of program comprehension since the EEG and eye movements can be measured in high temporal resolutions. Further, the devices are easy to apply real environments such as company working spaces and classrooms for students.

7.2 Analysis using Machine Learning

The programmer concentrates his/her fixation to source code when the programmer try to comprehend source code. Then, the programmer's α in EEG increases continually after s/he tried to comprehend source code. After that, the programmer's α increases greatly when programmer have comprehended. Therefore, α and fixation to source code should change as follows when programmers have comprehended successfully. At first, α and fixation to source code should increase due to start of source code comprehension. After that, α should increase greatly due to source code comprehension. The programmer's α without comprehension also increases continually after he/she tried to comprehend source code. After that, however, the programmer's α increases greatly greatly late or increase as same due to failure of comprehension. Therefore, α and fixation to source code should change as follows when programmers have not comprehended. At first, α and fixation to source code should increase due to start of source code comprehension. After that, α should increase greatly late or increase as same due to failure of comprehension.

Here, we classify programmers who comprehend successfully or not using machine learning. As attribute, we use the time lag when programmer's fixation to source code increase and when programmer's α increase.

Acknowledgment

これまで研究を進める当たって，様々なご指導とご助言を頂きました上野秀剛准教授に深くお礼申し上げます．おかげさまで多くの研究会に参加でき，多くの経験を積むことができました．本科 5 年生から専攻科を修了するまでの 3 年間が充実できたのは先生のご協力があってのことだと思っております．また，研究力向上セミナーにて松村教授，山口賢准教授，市川嘉裕助教にご意見を頂いたことで，自身の研究に対して違った角度から見つめなおすことができました．実験に参加して頂いた多くの方々には，負担の大きい実験に 1 時間強付き合っていただきました．この場を借りてお礼申し上げます．

なお，本研究は JSPS 科研費 JP16K00114 の助成を受けたものです．

References

[1] Mayrhauser, A. V. and Vans, A. M., ”Program comprehension during software maintenance and evolution,” in Computer, Vol. 28, No. 8, pp. 44-55 (1995).

[2] Xu, S., ”A cognitive model for program comprehension,” In Proceedings of the Third ACIS International Conference on Software Engineering Research, Management and Applications (SERA’05), pp. 392-398 (2005).

[3] Bloom, B. S., Engelhart, M. B., Furst, E. J., Hill, W. H. and Krathwohl, D. R., ”Taxonomy of educational objectives. The classification of educational goals. Handbook 1: Cognitive domain,” New York: Longmans Green (1956).

[4] Torii, K., Matsumoto, K., Nakakoji, K., Takada, Y., Takada, S., and Shima, K., ”Ginger2: an environment for computer-aided empirical software engineering,” in IEEE Transactions on Software Engineering, Vol. 25, No. 4, pp. 474-492 (1999).

[5] Siegmund, J., Peitek, N., Parnin, C., Apel, S., Hofmeister, J., Kastner, C., Begel, A., Bethmann, A., and Brechmann, A., ”Measuring neural efficiency of program comprehension,” In Proceedings of the 11th Joint Meeting on Foundations of Software Engineering(ESEC/FSE), pp. 140-150 (2017).

[6] Birbaumer, N., Elbert, T., G M Canavan, A., Rockstroh, B., ”Slow potentials of the cerebral cortex and behavior,” Physiological Reviews, Vol.70, pp. 1-41 (1990).

[7] Rodeghero, P., Liu, C., McBurney P. W., and McMillan, C., ”An eye-tracking study of java programmers and application to source code summarization,” in IEEE Transactions on Software Engineering, Vol.41, No.11, pp. 1038-1054 (2015).

[8] Zuger, M., and Fritz, T., ”Interruptibility of software developers and its prediction using psycho-physiological sensors,” In Proceedings of 33rd Annual ACM Conference on Human Factors in Computing Systems, pp. 2981-2990 (2015).

[9] Muller, S. C., and Fritz.T., "Stuck and frustrated or in flow and happy: Sensing developers' emotions and progress," In Proceedings of 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering, Vol.1, pp. 688-699 (2015).

[10] Siegmund, J., A. Brechmann, Apel, S., Kastner, C., Liebig, J., Leich, T., and Saake, G., "Toward measuring program comprehension with functional magnetic resonance imaging," In Proceedings of the ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering (FSE'12), No.24 (2012).

[11] Fritz, T., Begel, A., Müller, S. C., Yigit-Elliott, S., Züger, M., "Using psycho-physiological measures to assess task difficulty in software development," In Proceedings of the International Conference on Software Engineering (ICSE), pp. 402-413 (2014).

[12] Satheesh, Kumar, J., Bhuvaneswari, P., "Analysis of electroencephalography (EEG) signals and its categorization - A study," In Proceedings of the International Conference on Modeling, Optimization and Computing (ICMOC), Vol.38, pp. 2525-2536 (2012).

[13] Klem, GH, Luders, H, Jasper, HH, Elger, C., "The ten-twenty electrode system of the international federation," The International Federation of Clinical Neurophysiology. Electroencephalography and clinical neurophysiology, Vol.52, pp. 3-6 (1999).

[14] Duchowski, A. T., "Eye tracking methodology," Springer (2006).

[15] Behroozi, M., Lui, A., Moore, I., Ford, D., and Parnin, C., "Dazed: measuring the cognitive load of solving technical interview problems at the whiteboard," In Proceedings of the International Conference on Software Engineering (ICSE), pp. 93-96 (2018).

[16] Busjahn, T., Bednarik, R., Begel, A., Crosby, M., Paterson, J. H., Schulte, C., Sharif, B., Tamm, S., "Eye movements in code reading: Relaxing the linear order," In Proceedings of the 23rd International Conference on Program Comprehension (ICPC), pp. 255-265 (2015).

[17] Yamamoto, A., Uwano, H., and Ikutani, Y., "Programmer's electroencephalogram who found implementation strategy," In Proceedings of 4th International

Conference on Applied Computing & Information Technology (ACIT), pp. 164-168 (2016).

[18] Sun, WT., Sheu, FR., and Tsai, MJ., ”Understanding inquiry-based searching behaviors using scan path analysis: a pilot study,”. Innovative Technologies and Learning. ICITL 2018. Lecture Notes in Computer Science, Vol. 11003 (2018).

[19] Girardi, D., Lanubile, F., Novielli, N., and Fucci, D., ”Sensing developers' emotions: The design of a replicated experiment,” In Proceedings of the 3rd International Workshop on Emotion Awareness in Software Engineering (SEmotion), pp. 51-54 (2018).

[20] Soundarajan, S., and Cecotti, H., ”A gaze-based virtual keyboard using a mouth switch for command selection,” In Proceedings of the 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), pp. 3334-3337 (2018).

[21] Lee, Seolhwa., Hooshyar, Danial., Ji, Hyesung., Nam, Kichun., Lim, Heuiseok., ”Mining biometric data to predict programmer expertise and task difficulty,” Cluster Computing, pp.1097-1107, (2018).