Reading Treatment and Spatial Treatment for New Programmers: A View from Neuroimaging

Seminar: Software Engineering Research in the Neuroage

Minh-Khue Pham

Advisors: Prof. Sven Apel,
Dr. Norman Peitek

Saarland Informatics Campus,
Saarland University

1 Introduction

When we think about software engineering, we tend to think of technical aspects, such as programming languages, the development process, testing, design, etc., rather than human aspects, such as education, training, team collaboration, communication, etc. Understanding human aspects, on the other hand, is crucial for the development of the software engineering community. Considering this concept as a structure. Every stable structure requires a solid foundation. Similarly, studying novice programmers is essential for training the future generation of software engineers, providing guiding tools, and advancing software engineering pedagogy.

Recent studies of software engineers have discovered a direct correlation between spatial and programming abilities. Spatial ability refers to the capability to perceive, interpret, and retrieve the visual and spatial relationships between objects or space. Novice programmers' neural activation patterns showed no significant differences during coding or spatial reasoning. Tantalizingly, Endres *et al.* can predict students' programming performance at the end of the semester by comparing brain activation patterns during coding and spatial reasoning at the beginning of the semester [8]. Spatial skills interventions have been proven to successfully improve the final programming outcomes of not only high school students [6] but also university students in the US [3], [7].

Technical reading ability training is another prominent skill-based intervention for STEM education. As opposed to general reading ability, technical reading ability is the ability to read and comprehend technical or scientific texts. Reading technical documentation, reviewing code, or fixing bugs all involve technical reading ability. Endres *et al.* reported that new programmers who underwent a reading intervention performed significantly better on their final programming test than those who underwent a spatial intervention [3]. With the help of medical neuroimaging, several studies have observed resemblances between program comprehension and prose reading [18], [19], [9]. Intriguingly, more experienced programmers tend to treat code and reading more similarly at a neural activation level [9].

Many studies have performed Spatial Treatment and Reading Treatment to better understand the effect of spatial and reading abilities on programming achievement. While those studies were able to capture improvements in programming performance, none of them has yet exploited neuroimaging technology to pinpoint changes in brain activity before and after spatial and reading interventions. In this paper, we propose a 14-week longitudinal experiment on novice programmers enrolled in the Programming 1 course at Saarland University. The students are separated into controlled Spatial Treatment and Reading Treatment groups. After nine weeks of interventions, we measure variations in programming results as well as in brain activation patterns to explain the influence of the two Treatments on new programmers.

Vocabulary: The cerebrum of the human brain is comprised of two hemispheres, left and right, and four primary lobes: *frontal lobe* at the front of each hemisphere, *temporal lobe* on the side of each hemisphere), *parietal lobe* at the top of each hemisphere, and *occipital lobe* at the back of each hemisphere. *Bilateral* activation occurs when both hemispheres are activated and *lateral* activation occurs when one hemisphere activates disproportionately. *Brodmann Area (BA)* is an anatomical classification system for the cortex [4]. Brodmann areas divide the cortex into 52 bilateral regions based on architectural neurological features.

Notation: In this paper, the neuroimaging notation Task A > Task B was used to indicate the contrast between brain activation patterns for two experimental tasks. A positive value indicates that the specified brain area was more active during Task A than Task B, while a negative value indicates that the specified brain area was less active during Task A than Task B.

2 Related Work

In this section, we provide a summary of the state-of-the-art on the topic. Specifically, we introduce the first study that targeted novice programmers utilizing medical neuroimaging technology and time-delayed programming performance assessment.

The second and third research compared the effectiveness of the Reading Treatment, Spatial Treatment, and Control group (i.e., the group received no treatment) on novice programmers using contrast-based experiments. The second study is a follow-up research project by the same author as the first study. It was initially designed to evaluate the impact of reading interventions by comparing the programming gains of novices in the Reading Treatment group to those in the Control group. The third study analyzed the effect of spatial skills interventions on programming ability. Unfortunately, these two experiments can produce biased results between the Treatment and Control groups due to some limitations. Two research groups presented two potential solutions to this problem: Endres *et al.* compared Reading Treatment to Spatial Treatment in the same academic year, while Bockmon *et al.* performed interventions on the same course across two years, in which they collected control data in the first year and treatment data in the second year.

2.1 Coding, Reading and Spatial in Novices [8]

Many studies found strong similarities between code comprehension and prose reading [18], [19], [9], albeit one study has shown that coding is neurologically similar to spatial reasoning [11]. Even though understanding novice programmers is important to developing the software engineering community, researchers limit their focus to programmers with years of expertise. This study aims to close this gap by inspecting the cognitive processes of true *novices* in terms of coding, spatial reasoning, and reading.

Experiment. The experiment took place over a 15-week semester and was separated into two phases: neuroimaging and a follow-up assessment. This experimental approach allows the controlled exploration of the relationships and contrasts between reading, coding, and spatial reasoning for novices.
Novice programmers were recruited in the first week. To find true novices and mitigate self-selection bias, the recruitment protocol is composed of three stages: recruitment, pre-screening, and validation using the results of a short programming test. Participants are students in the CS1 course at the University of Michigan. The final neuroimaging data from 31 participants, including 24 females and 7 males, are eligible.
During the first third of the semester, the neuroimaging were held. Neurological activation in novices' brains was examined during spatial-based, reading-based, and coding-based stimuli.
In the last week of the course (10-12 weeks after the neuroimaging), participants are invited to complete a follow-up assessment. They used the Second CS1 Assessment (SCS1) [15]. The SCS1 is a validated standard assessment for CS1 students and covers basic concepts of programming, such as Boolean logic, while loops, for loops, arrays, if statements, functions, and recursion. The questions are a combination of three types: definition questions, code tracing questions, and code replacement questions.

RQ1—What areas of the brain activate when novice software engineers program? Figure 1 depicts the brain activation patterns for Coding > Rest. The authors observed considerable bilateral neural activation in the occipital cortices, particularly in BAs 17, 18, and 19, which are the Brodmann areas linked with visual processing. Furthermore, novices also exhibit a strong bilateral activation in the posterior parietal cortex, primarily in the angular gyrus (BA 39), which is correlated with spatial cognition tasks, and a significant bilateral activation in the working memory region DLPFC (BA 46). The first conclusion was implied: the coding brain activation of novice programmers is connected with both language and spatial cognition regions; additionally, novices find programming a challenging and working memory-intensive task that demands high attention.

RQ2—How does the coding brain activation of novices compare to their brain activation during mental rotation and during reading?

Contrary to previous studies on experienced software engineers, which found more neurological similarities between coding and prose reading, novices exhibited more substantial differences in Coding > Reading than in Coding > Mental Rotation, as shown in Figures 2 and 3. For Coding > Mental Rotation, novices found coding a more spatially intensive task than simple mental rotation. Participants reported higher bilateral frontal activation while coding than while mentally rotating objects, especially in the DLPFC and the premotor cortex (BAs 6 and 46), which are involved with working memory and spatial manipulation. Moreover, high coding activation was observed in the right angular gyrus, an area related to spatial reasoning. For Coding > Reading, coding had comparatively stronger activation throughout the right hemisphere and in the premotor cortex. The right occipital, angular gyrus, and DLPFC were all more activated while coding than while reading, however, differences in other regions were due to the strong deactivation in Reading > Rest. Reading had comparatively more activation in Broca's and Wernicke's areas, which are highly correlated with language functions.

RQ3—Are there connections between coding brain activation patterns at the beginning of CS1 and their programming performance at the end of the course? At the end of the semester, the authors analyzed correlations between students' programming test results and five brain regions: the right hemisphere, left hemisphere, right hemisphere frontal, left hemisphere frontal, and occipital regions. Students who exhibited less similar neural activation patterns for coding and mental rotation in the right frontal hemisphere at the start of the semester performed better on the end-of-semester programming assessment.

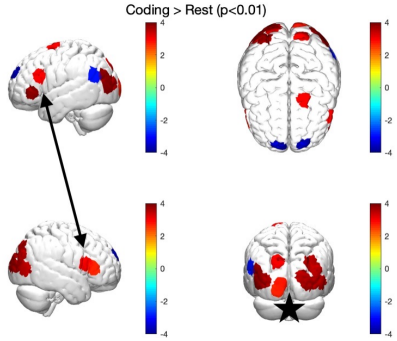

Fig. 1. Significant Coding Activation: Red indicates regions more activated during the coding task while blue indicates regions more activated during rest. The widespread bilateral activation in both the DLPFC (the arrows) and the occipital and posterior parietal cortices (the star) [8].

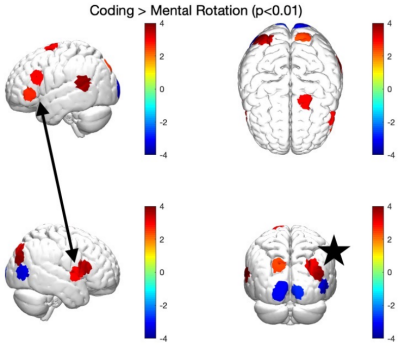

Fig. 2. Activation Contrast between Programming and Mental Rotation: Red indicates regions more activated during coding, blue indicates regions more activated during mental rotation. Frontal cortex (the arrow) and right angular gyrus (the star) [8].

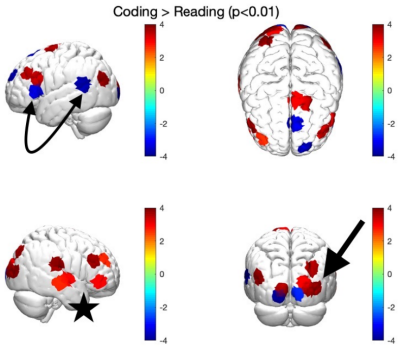

Fig. 3. Activation Contrast between Programming and Reading: Red indicates regions more activated during coding, blue indicates regions more activated during reading. Broca's and Wernicke's areas (the double arrow), right frontal cortex (the star) and right occipital/parietal cortices (the single arrow) [8].

Limitation. Although the researchers managed to mitigate the majority of the threats to validity, some limitations remained. The study results cannot be generalized since all subjects are students at the same US university. Another issue is whether they measured what they claimed to measure. For instance, testing spatial ability using only mental rotation stimuli, measuring programming ability with the SCS1 assessment, etc.

Conclusion. Endres *et al.* discovered that coding, prose reading, and mental rotation are psychologically separate for beginner programmers by using medical neuroimaging to analyze the cognitive processes of 31 novice programmers. However, they found more significant differences between reading and coding than between mental rotation and coding. They also noticed a neural activation pattern that predicted programming performance 11 weeks later.

2.2 Reading Treatment versus Spatial Treatment [7]

Reading and spatial abilities have been shown to play an important role in STEM education. Many studies have demonstrated that developing these cognitive abilities can help students enhance their programming proficiency. The question here is: on which abilities do new programmers place the most emphasis? In this study, they compared the effectiveness of spatial skills interventions to their proposed technical reading skills training curriculum that is specified for computer science students.

Experiment. The experiment was conducted over 11 weeks. In the first week, novice programmers were recruited. They were arbitrarily assigned to either the Reading Treatment group or the Spatial Treatment group. Both Treatments had nine two-hour weekly training sessions. Before and after the Treatment, participants took two assessments to measure the variation in their spatial, reading, and programming performance. This experiment design allows researchers to evaluate the effectiveness of their proposed CS-focused technical reading and established spatial ability curriculum.
Akin to Endres *et al.*'s previous experiment, students had to go through three recruitment stages: recruitment, pre-screening, and a brief programming test. Participants were enrolled in the same CS1 course at the University of Michigan. Even though 97 participants passed the screening process, only 57 met the eligibility criteria and were used for analysis.
The overall training session plan consists of three phases: warm-up exercises, short lectures, and practice. For Spatial Treatment, they administered an established spatial ability course developed by Sorby and Baartmans [20]. For Reading Treatment, they emphasized technical reading strategies and vocabulary. Students practiced using CS-papers and API documents.
Endres *et al.* used the Paper Folding Test (PFT) [10] and the Revised Purdue Spatial Visualization Test (PSVT:R II) [21], which are validated standard assessments of different facets of spatial ability, as measurements for spatial ability. To assess reading ability, they used the verbal sections of the Graduate Record Examination (GRE) [1]. For programming ability, participants completed the Second CS1 Assessment (SCS1) developed by Parker *et al.* [15].

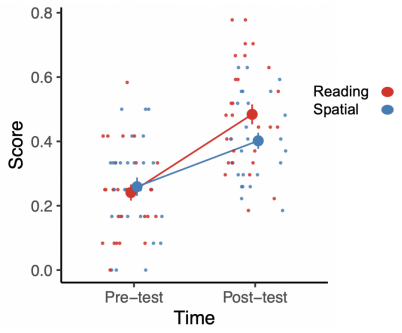

Fig. 4. Percentage Programming Gains by Treatment: Reading Treatment participants exhibit significantly larger gains than Spatial Treatment participants [7].

Validation. Spatial and reading assessments reported astonishing results: both interventions did not appear to improve students' spatial ability or GRE reading scores. The fact that the intervention showed no effect on spatial skills is

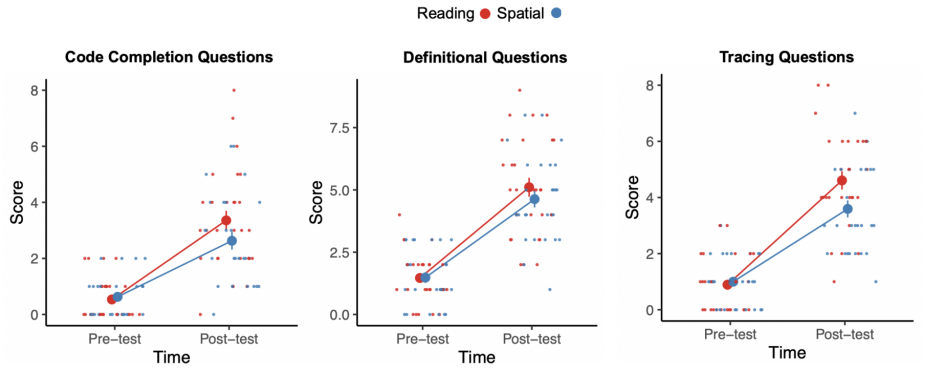

Fig. 5. Question-type specific programming gains by Treatment: Reading Treatment participants exhibit significantly larger gains than Spatial Treatment participants on the Tracing questions but not on the Code Completion or Definitional questions. For each graph, score is out of a maximum of 9 [7].

likely due to the ceiling effect, as participants already had high incoming spatial ability. The unaffected reading performance is due to the fact that the GRE was chosen as a proxy for CS-focused technical reading ability since their reading intervention focused on different skills that could not be measured by the GRE.

RQ1—Did Reading Treatment participants perform better than Spatial participants on the final programming test? This is the main question of the study. By juxtaposing the Final Programming Outcomes of two Treatment groups, they noticed that novices in the Reading Treatment group performed better than those in the Spatial Treatment group (see Figure 4).

RQ2—Were the effects the Treatments more pronounced for some programming question types than others? Students in two Treatment groups showed variations in outcomes depending on programming question types. Reading Treatment participants outstood Spatial Treatment participants on the tracing questions but not on the code-completion and definitional questions (see Figure 5).

Limitation. Besides the limitations of interventions as mentioned in Validation 2.2, one more threat to validity is the inhomogeneity in incoming spatial ability between the two Treatment groups. However, the authors attempted to mitigate this by using models that account for pre-existing individual differences.

Conclusion. In this paper, Endres *et al.* conducted a controlled, contrast-based experiment to evaluate the effectiveness of their proposed CS-focused technical

reading intervention and an established spatial intervention on novices' programming performance. Although there exists some threat to validity, Reading Treatment was shown to improve students' programming outcomes more than Spatial Treatment, especially problems that require code tracing.

2.3 Spatial Treatment versus No Treatment [3]

In a 2015 study by Cooper *et al.* [6] on high school students, they observed a direct correlation between students' spatial ability and their success in learning to program. This paper replicated Cooper *et al.*'s study on introductory computer science students at three universities in the US to validate the impact of spatial skills intervention on novices' programming abilities.

Experiment. The experiment lasted two years. In year one, the control data were acquired by gathering 247 participants' programming and spatial assessment results at the start and end of the course. In year two, the treatment data were acquired by administering a spatial skills intervention to 71 participants during the semester between two assessments.
The hybrid spatial intervention consists of weekly online lectures and worksheets that cover eight modules: surfaces and solids of revolution; isometric drawings; coded plans; orthographic drawings; rotations about a single axis; rotations about two or more axes; reflections and symmetry.

RQ—Can spatial skills successfully be taught to students taking introductory computing classes, and do students who learn spatial skills also demonstrate improved performance in computing? Students in the second-year Treatment group exhibited significantly larger gains in the post-test programming assessment SCS1 than students in the first-year Control group, as shown in Figure 6). The dissimilarity in programming outcomes indicated that the spatial skills intervention successfully improved students' programming skills.

	Control $\mu_c(rank_c)$	Treatment $\mu_t(rank_t)$	p-value
Pre-SCS1R	1.7(169)	1.9(187)	0.15
Post-SCS1R	2.8(163)	3.5(209)	< 0.01
Δ SCS1R	1.1(166)	1.6(199)	0.01

Fig. 6. The raw score averages (μ) and rank score averages (rank) of the Control and Treatment group scores for the SCS1R [3].

Limitation. Even though the results are aligned with Cooper *et al.*'s work, many threats to validity exist. First, all participants were voluntary, which causes selection biases. Secondly, the complexity of programming assessment is not appropriate for students in an introductory course. The last threat to validity is that all exams were administered online and not controlled.

Conclusion. With the use of spatial intervention, students in the Treatment group performed much better than those in the Control group. The authors hope that their study can convince instructors to make spatial skills intervention a mandatory part of software engineering education.

3 Experimental Idea

Our experimental design was motivated by two publications from Endres in 2021, which have been summarized in sections 2.1 and 2.2: one used neuroimaging to investigate the cognitive processes behind novices' coding brain [7], and the other performed interventions on novices [8]. Based on all the insights from these existing studies, we propose an experimental design that combines neuroimaging with reading and spatial skills interventions to explain how such interventions affect novices' neurological reasoning about coding. Below, we address two research questions and our motivation for choosing them:

RQ1—Does Reading Treatment group have better programming results than Spatial Treatment group? According to posttreatment programming outcomes in studies performed by Endres *et al.* [7] and Bockmon *et al.* [3], Reading Treatment is likely to be the most efficacious intervention, followed by Spatial Treatment, and finally No Treatment. However, this conclusion cannot be generalized since all participants are students at universities in the US. We want to validate this finding by running a 9-week Reading Treatment and Spatial Treatment with novice programmers at a German university and then comparing the posttreatment programming results between the two Treatment groups. Furthermore, we propose directions to avoid the remaining limitations from previous studies.

RQ2—What are the differences between novices' brains after Reading and Spatial Treatment? Although many studies have investigated the impact of Reading and Spatial Treatment on software engineers, to the best of our knowledge, none have looked into alterations in brain development caused by Treatments. To answer this study issue, two types of variation between the two Treatment groups are examined and compared. The first variation is posttreatment coding brain activation, i.e., comparing PostReadingTreatment > Rest to PostSpatialTreatment > Rest. The second variation is cognitive development processes, i.e., comparing PostReadingTreatment > PreReadingTreatment to PostSpatialTreatment > PreSpatialTreatment. Analyzing contrasts enables

us to scrutinize the neurological factors that may be decisive for the development of software engineers, as well as the influence of reading and spatial skills interventions on coding brains.

3.1 Design Overview

We schedule a controlled, contrast-based experiment that lasts 14 weeks during one study semester. In the first week, novice programmers are recruited. Participants are then divided into either Reading Treatment or Spatial Treatment groups. Both Treatments last for 9 weeks, akin to Endres *et al.*'s experimental design [7]. Two weeks before and two weeks after the Treatment, variations in programming outcomes and coding brain activation are measured. In particular, we describe our recruitment protocol (3.2), propose our training procedure (3.3) and assessment instruments (3.4), and outline our neuroimaging data collection process (3.5). This experimental design assists in validating the effect of two skills interventions in Endres *et al.*'s study. Moreover, with the help of medical neuroimaging techniques, we have a better understanding of the neurological development process after interventions.

3.2 Participant Recruitment

Participants are students in an introductory Programming 1 course at Saarland University in Germany. They are recruited through forum posts, flyers on bulletin boards, and in-class presentations. For each week they attend Treatment sessions, for two scanning sessions, and for two assessments, participants are compensated €15 (€135 for participants attending a total of 9 weeks of Treatment, €165 for participants attending all Treatment, scanning, and assessment sessions).
The recruitment protocol consists of three parts: recruitment, prescreening, and a brief programming test. In the recruitment stage, we deliberately select students who have no prior programming experience among all applicants. Participants should have adequate English proficiency, i.e., enough to be able to understand technical reading materials and questions in assessments. We mitigate self-selection biases through the prescreening stage by asking students whether they "had any prior programming experience". Only students selecting "No" are retained; others selecting "Yes" or "Unsure" are eliminated from the study. A brief programming test at the end validates the population's programming experience. The average scores are expected to be low since novices only have limited programming knowledge. At the end of the semester, only the data of students who have completed six out of nine training sessions are included in our analysis.

3.3 Treatment Materials

We have two design goals for the Treatment. On one hand, to directly compare Reading Treatment to Spatial Treatment, both participants in the two groups

attend the same 2-hour weekly training session that is a combination of a 20-minute warm-up, a 10-minute lecture, and a practice session that accounts for half of the training time with the same instructors. On the other hand, to encourage active learning, we require that both Treatments consist of physical props in their training materials (e.g., vocabulary flashcards for Reading Treatment in comparison to snap-blocks for Spatial Treatment), make use of the collaborative learning strategy think-pair-share, group work, and in-class practice problems. The active learning method has been proven to harness the benefits of interventions, allow students to deepen their understanding of a given topic, and enhance course completion rates.

Spatial Treatment. For Spatial Treatment, we use an established spatial ability curriculum developed by Sorby and Baartmans [20] that covers nine topics: surfaces and solids of revolution, combining solid objects, isomorphic sketching, orthographic sketching, orthographic projections, flat pattern folding, 3D shape rotation around single and multiple axes, object reflection or symmetry, and mental cutting. Participants study through provided lecture videos and practice using physical props, software, and workbooks.

Reading Treatment. The warm-up portion contains some icebreaker exercises to keep students motivated for the rest of the training, such as vocabulary practice emphasizing words that are prevalent in scientific writing and workbook problems from Reading Comprehension Skills and Strategies Level 8 [2]. During weekly lectures, students study various reading strategies. The majority of these strategies centered on enhancing text skimming skills, as experienced programmers tend to read code non-linearly and solely focus on high-level features [5], [16]. At the end of each training session, students practice learning transfer by working with computer science research papers and API documents. The research papers are chosen based on several criteria. First, to get students excited, we select influential papers that are interesting to first-year students and published in top-tier computer science journals and conferences. Secondly, to ensure that the materials are understandable for novice programmers, they are required to only read a curated subset of the paper, such as the introduction, related work, and conclusion. Participants are asked to summarize, write, and share their responses after reading papers since studies have shown that paraphrasing helps learners strengthen reading comprehension, critical thinking, and broader learning for transfer.

3.4 Assessment Instruments

During the initial assessment phases, reading, spatial, and programming abilities of all participants are measured. The pretreatment performances are used to divide participants into either Reading Treatment or Spatial Treatment groups, such that both groups are homogeneous in all abilities. The second assessment

phase at the end of the semester controls students reading, spatial, and programming abilities one more time to estimate the performance variations as well as the effectiveness of each treatment. Three types of assessments are partitioned into halves, one for the pre-assessment phase and one for the post-assessment phase, to avoid re-testing effects.

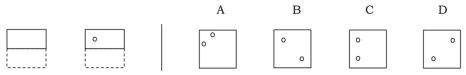

Fig. 7. PFT: Participants select the choice which corresponds to the paper on the left unfolded. Answer is "C" [7]

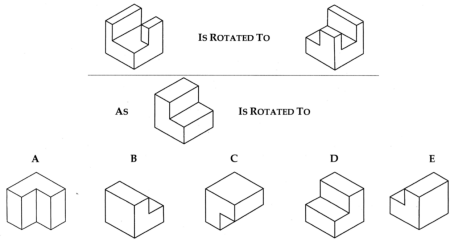

Fig. 8. PSVT:R II: Participants select the shape that is rotated in the same way as the top shape. Answer is "D" [7].

Spatial Assessment. Spatial ability assessments are similar to those in Endres *et al.*'s and Bockmon *et al.*'s studies. The Paper Folding Test (PFT) [10] and the Revised Purdue Spatial Visualization Test (PSVT:R II) [21] are valid, standardized tests of several aspects of spatial aptitude. While the PFT measures mental folding ability and contains a set of 20 questions, the PSVT:R II measures mental rotation ability and contains 30 questions. Students complete two controlled assessments with a 30-minute time constraint. An example of PFT questions and PSVT:R II questions are shown in Figures 7 and 8.

Reading Assessment. Since the technical CS-focused Reading Treatment curriculum has only recently been published, there is currently no validated reading evaluation available to measure the impact of this curriculum on novice programmers. Endres *et al.* used the verbal sections of the Graduate Record Examination (GRE) [1], however, the GRE cannot measure students' improvement in technical reading ability. Therefore, we suggest developing a technical CS-focused reading assessment that should satisfy certain criteria. The primary criterion is

that the designed assessment must be capable of measuring the enhancement in students' skimming proficiency, namely tempo and accuracy. The capacity of students to understand technical terminologies in scientific papers and API documents is the next measurement criterion. The technical reading assessment is designed to last 30 minutes and is in the form of multiple-choice questions, akin to the spatial assessment and the GRE.

Programming Assessment. The Second CS1 Assessment [15] is a validated measurement of CS1 programming ability that has been employed in many research papers, including studies performed by Endres *et al.* and Bockmon *et al.* We noted that our subjects are students in the Programming 1 course instead of the CS1 course; the SCS1 is not appropriate to evaluate novices' programming skills in our study design. As for the disparity in course content, we propose a language-agnostic programming assessment tailored to Programming 1 students. Three types of questions are covered in the assessment: definition questions, trace-based questions, and code-completion questions. All questions are ensured to be compatible with novices' limited initial programming abilities.

3.5 Scanning Session

Neuroimaging Technique. We make use of the neuroimaging technique fNIRS (functional near-infrared spectroscopy) for two main reasons. First, this technique provides better spatial resolution than EEG and better temporal resolution than fMRI, which is important since we only need to focus on analyzing specific brain regions related to spatial and reading tasks. fNIRS devices are not only economical but also lightweight and portable, allowing participants to freely interact with computers throughout the scanning session.
As the near-infrared light propagates through the head, it is alternately scattered or absorbed by the tissue through which it travels. fNIRS estimates oxygen saturation from changes in the absorption of near-infrared light. Participants wear a cap with transmitters and receivers strategically placed on brain areas that need to be scanned. Since fluctuations in neural blood flow only peak slightly after stimuli are presented and gradually return to baseline level, each of our stimuli is designed to last no more than 30 seconds.

Procedure. Two groups' neuroimaging data are collected during a single 1.5-hour session. Participants provide informed consent and watch a training video preparing them for the scan, e.g., what must be paid attention to during the scanning session and how the stimuli should be responded to. The entire process takes about 30-45 minutes.
Each participant completes 30 coding-based stimuli and has up to 30 seconds to respond to each stimulus. Before moving on to the next stimulus, a fixation cross was shown for 2-10 seconds to recalibrate neural responses. The stimulus set is arbitrarily partitioned into three blocks, each of which contains 10 coding-based stimuli. Participants can take longer breaks between blocks.

Coding Stimuli. We use similar coding stimuli to those in Endres *et al.*'s study to mitigate variation in scanning results. Even though the programming languages in the Programming 1 course and the CS1 course are different (OCaml with C++, Python), coding stimuli are nevertheless understandable for first-year students since they only contain boolean logic, while loops, for loops, and arrays. The stimuli require either choosing the correct output or returning the value of a short code snippet. Participants respond to the stimuli by selecting one of two options. One example of a stimulus is shown in Figure 9.

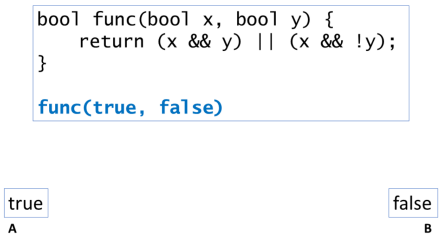

Fig. 9. Example fNIRS Stimuli for coding task: correct answer is "A" [8].

4 Validation

We first validate the consistency between our pretreatment fNIRS data and those reported by Endres *et al.*'s work in Figure 1. Do our subjects also exhibit brain activation patterns related to language, spatial cognition, and the DLPFC region of working memory while coding? Are the signal intensities of those brain regions similar to the results found by Endres *et al.*? Numerous factors can affect our scanning results. One factor is the differences in subjects' study backgrounds. Variations in high school mathematical study content can significantly alter the reliance of novices on spatial ability during coding. Another factor is instrumental error, which is the inaccuracy or imprecision of the instruments used in our experiment. Moreover, participants may perform the tasks incorrectly during the scanning session, resulting in human error.

We next analyze the effectiveness of both interventions. After Spatial Treatment, students should show improvement in post-test scores evaluating their spatial ability. Intriguingly, the post-test scores in Endres *et al.*'s study have shown no improvement in novices' spatial ability, as it is likely due to the "ceiling effect", i.e., the majority of the subjects had reached their upper limit in spatial ability by the start of Spatial Treatment. The spatial ability pre-test scores are used to not only split participants into two Treatment groups but also to determine whether the "ceiling effect" exists in our study. As the GRE could not measure CS-focused technical reading gains in novice programmers [7], we evaluate the effectiveness of Reading Treatment using our designed reading assessment. If

participants show no differences after the intervention, it is either because our designed assessment is an inadequate proxy for CS-focused technical reading ability or because the intervention has no effect on novices.

5 Limitations

Besides some threats to validity mentioned in Validation 4, this section addresses the remaining limitations and threats to the validity of our experimental design. One threat to validity is the fact that English is not the mother tongue of our subjects. Although we deliberately control the complexity level of practice papers and assessments to ensure that the materials are understandable for first-year students, participants may still encounter a few challenges. The language barrier creates uncertainties in RQ1 and RQ2 results.

There are major differences in study content between the Programming 1 course (OCaml) and the CS1 course (C++, Python). Because of the study content and the topics of programming assessments, one skill is likely to be prejudiced over the other, resulting in one Treatment group benefiting more than the other. For example, while tree-based data structures trigger spatial brain regions, array-based data structures do not [12]. One neuroimaging study found that code reading was less associated with spatial ability than code writing [13].

Finally, we also face generalizability limitations. The majority of German high schools made computer science a compulsory curriculum. Our sample size may be insufficient to draw any conclusions. Although there are still novice programmers with no prior experience at the university, most of them are likely to be foreign students and female students, which introduces population bias. The majority of the population (77%) in one of Endres *et al.*'s studies were female [8]. This is due to the fact that female students often sign up for college studies at a higher rate than male students [14], and another explanation is that men tend to already have some prior programming experience [17]. As our study contains intense training for 14 weeks, attrition is unavoidable, which may result in a relatively small sample size.

6 Conclusion

In a 14-week longitudinal experiment, we not only replicate Endres *et al.*'s Spatial Treatment and Reading Treatment to validate the impact of two interventions on novice programmers but also extend the original study by applying medical neuroimaging techniques to explain the cognitive processes behind the development of novice programmers. In particular, we use fNIRS to analyze the effects of spatial and reading skill interventions on coding. While some limitations in our experimental design remain, we expect that this interdisciplinary work can detect critical brain regions for software engineers, explain the mechanism behind cognitive development processes, and moreover, support software engineering pedagogy and establish better guidelines to educate the next generation of software engineers. We hope that our findings can inspire computer

science educators to incorporate spatial and reading skills interventions into their curricula.

References

1. Ets.org. 2023. gre home. ets, https://www.ets.org/gre, accessed on August 31, 2023
2. Reading Comprehension Skills and Strategies Level 8. High-Interest Reading Comprehension Skills and Strategies Series, Saddleback Pub (2002), https://books.google.de/books?id=cMJ4ypi2_HIC
3. Bockmon, R., Cooper, S., Koperski, W., Gratch, J., Sorby, S., Dorodchi, M.: A cs1 spatial skills intervention and the impact on introductory programming abilities. In: Proceedings of the 51st ACM Technical Symposium on Computer Science Education. p. 766–772. SIGCSE '20, Association for Computing Machinery, New York, NY, USA (2020), https://doi.org/10.1145/3328778.3366829
4. Brodmann, K., Garey, L.: Brodmann's: Localisation in the Cerebral Cortex. Springer US (2007), https://books.google.de/books?id=l1QsF0UdufgC
5. 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: 2015 IEEE 23rd International Conference on Program Comprehension. pp. 255–265 (2015)
6. Cooper, S., Wang, K., Israni, M., Sorby, S.: Spatial skills training in introductory computing. In: Proceedings of the Eleventh Annual International Conference on International Computing Education Research. p. 13–20. ICER '15, Association for Computing Machinery, New York, NY, USA (2015), https://doi.org/10.1145/2787622.2787728
7. Endres, M., Fansher, M., Shah, P., Weimer, W.: To read or to rotate? comparing the effects of technical reading training and spatial skills training on novice programming ability. In: Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering. p. 754–766. ESEC/FSE 2021, Association for Computing Machinery, New York, NY, USA (2021), https://doi.org/10.1145/3468264.3468583
8. Endres, M., Karas, Z., Hu, X., Kovelman, I., Weimer, W.: Relating reading, visualization, and coding for new programmers: A neuroimaging study. In: Proceedings of the 43rd International Conference on Software Engineering. p. 600–612. ICSE '21, IEEE Press (2021), https://doi.org/10.1109/ICSE43902.2021.00062
9. Floyd, B., Santander, T., Weimer, W.: Decoding the representation of code in the brain: An fmri study of code review and expertise. In: Proceedings of the 39th International Conference on Software Engineering. p. 175–186. ICSE '17, IEEE Press (2017), https://doi.org/10.1109/ICSE.2017.24
10. French, J., Service, E.T., Ekstrom, R., Price, L., of Naval Research, U.S.O.: Kit of Reference Tests for Cognitive Factors. Educational Testing Service (1963), https://books.google.de/books?id=iSFpyAEACAAJ
11. Huang, Y., Liu, X., Krueger, R., Santander, T., Hu, X., Leach, K., Weimer, W.: Distilling neural representations of data structure manipulation using fmri and fnirs. In: Proceedings of the 41st International Conference on Software Engineering. p. 396–407. ICSE '19, IEEE Press (2019), https://doi.org/10.1109/ICSE.2019.00053

12. Huang, Y., Liu, X., Krueger, R., Santander, T., Hu, X., Leach, K., Weimer, W.: Distilling neural representations of data structure manipulation using fmri and fnirs. In: 2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE). pp. 396–407 (2019)
13. Krueger, R., Huang, Y., Liu, X., Santander, T., Weimer, W., Leach, K.: Neurological divide: An fmri study of prose and code writing. In: Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering. p. 678–690. ICSE '20, Association for Computing Machinery, New York, NY, USA (2020), https://doi.org/10.1145/3377811.3380348
14. Lobato, L., Bethony, J.M., Pereira, F.B., Grahek, S.L., Diemert, D.J., Gazzinelli, M.F.: Impact of gender on the decision to participate in a clinical trial: a cross-sectional study. BMC Public Health 14 (2014), https://doi.org/10.1186/1471-2458-14-1156
15. Parker, M.C., Guzdial, M., Engleman, S.: Replication, validation, and use of a language independent cs1 knowledge assessment. In: Proceedings of the 2016 ACM Conference on International Computing Education Research. p. 93–101. ICER '16, Association for Computing Machinery, New York, NY, USA (2016), https://doi.org/10.1145/2960310.2960316
16. Rodeghero, P., McMillan, C.: An empirical study on the patterns of eye movement during summarization tasks. In: 2015 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM). pp. 1–10 (2015)
17. Sackrowitz, M.G., Parelius, A.P.: An unlevel playing field: Women in the introductory computer science courses. In: Proceedings of the Twenty-Seventh SIGCSE Technical Symposium on Computer Science Education. p. 37–41. SIGCSE '96, Association for Computing Machinery, New York, NY, USA (1996), https://doi.org/10.1145/236452.236488
18. Siegmund, J., Kästner, C., Apel, S., Parnin, C., Bethmann, A., Leich, T., Saake, G., Brechmann, A.: Understanding understanding source code with functional magnetic resonance imaging. In: Proceedings of the 36th International Conference on Software Engineering. p. 378–389. ICSE 2014, Association for Computing Machinery, New York, NY, USA (2014), https://doi.org/10.1145/2568225.2568252
19. Siegmund, J., Peitek, N., Parnin, C., Apel, S., Hofmeister, J., Kästner, C., Begel, A., Bethmann, A., Brechmann, A.: Measuring neural efficiency of program comprehension. In: Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering. p. 140–150. ESEC/FSE 2017, Association for Computing Machinery, New York, NY, USA (2017), https://doi.org/10.1145/3106237.3106268
20. Sorby, S.A., Baartmans, B.J.: The development and assessment of a course for enhancing the 3-d spatial visualization skills of first year engineering students. Journal of Engineering Education 89(3), 301–307 (2000), https://onlinelibrary.wiley.com/doi/abs/10.1002/j.2168-9830.2000.tb00529.x
21. Yoon, S.: Psychometric properties of the revised purdue spatial visualization tests: Visualization of rotations (the revised psvt:r) (01 2012)