Engineering Educators Bringing the World Together
2025 ASEE Annual Conference & Exposition
Palais des congrès de Montréal, Montréal, QC • June 22–25, 2025
Paper ID #48383

BOARD # 84: WIP: Auto-gradable Hands-On Parse Tree Learning Tool in Virtual Reality

Colin Jacob Soule, Bucknell University

Colin Soule is a graduating senior awarded the presidential fellowship at Bucknell University. He is currently committed to William & Mary's Ph.D program to research algorithms and theoretical computation.

Lea Wittie, Bucknell University

Lea Wittie is an Associate Professor and Co-Chair in the department of Computer Science in the Engineering College at Bucknell University. She does computer science education research with students in first year engineering topics, concept inventories, and teaching for mastery topics.

Sing Chun Lee, Bucknell University

Sing Chun is a Computer Science Faculty at Bucknell University. During his PhD Study on Geometry Processing and Augmented Reality, he discovered his passionate interest in Computer Science Education. Currently, Sing Chun is investigating how Mixed Reality (MR) might help with Computer Science Education through the lens of multimedia learning theory and cognitive load theory. He is also interested in promoting undergraduate research and making advanced computational concepts more accessible to undergraduate students. In his spare time, he likes to immerse himself in the MR world and discuss the future of MR, particularly for education, with others.

WIP: Auto-gradable Parse Tree Learning Tool in Virtual Reality

Colin Soule
cjs056@bucknell.edu
Computer Science
Bucknell University

Lea Wittie
lwittie@bucknell.edu
Computer Science
Bucknell University

Sing Chun Lee
singchun.lee@bucknell.edu
Computer Science
Bucknell University

Abstract

Parse trees, or syntax trees, are fundamental concepts in computer science as they represent the structure of programming language expressions. Traditional teaching methods require students to manually draw syntax trees for given expressions - a process that can become tedious for students to practice and time-consuming for educators to grade. This project explores the potential of virtual reality (VR) to provide a more engaging and interactive learning experience for syntax tree education while also supporting auto-gradable exercises for scalable practice.

We have developed a web-based VR tool that enables students to construct syntax trees through drag-and-drop interactions in an immersive environment. To evaluate its effectiveness, we plan to conduct a comparative study with three groups of undergraduate computer science students: one receiving more traditional instruction using a text-based tree generator, one using a browser-based drag-and-drop tool, and one utilizing the VR tool. The evaluation will include both qualitative and quantitative measures. Qualitatively, we will assess student engagement and self-efficacy through Likert-scale surveys. Quantitatively, we will compare task completion times and scores to evaluate learning outcomes. By automating tree validation and grading, the tool not only enhances engagement but also improves teaching efficiency.

1 Introduction

Parse trees, or syntax trees, are essential in computer science education as they represent the hierarchical structure of programming language expressions. They are fundamental in understanding syntax analysis, compiler construction, and language processing algorithms. However, traditional teaching methods often involve manually constructing syntax trees through static diagrams or hand-drawn exercises. While effective for basic understanding, this process can become monotonous and disengaging for students. Additionally, the manual nature of these exercises poses challenges for educators, particularly in providing timely feedback and facilitating mass practice opportunities.

Advances in educational technology, particularly virtual reality (VR), present new opportunities to enrich learning experiences. VR can transform complex abstract topics into hands-on, interactive learning activities. Despite its broader application in education, VR's potential in teaching core concepts in programming languages, such as syntax trees, has yet to be fully explored. In this project, we developed a web-based VR tool for syntax tree construction. The tool features a drag-and-drop interface within an immersive VR environment, allowing students to construct syntax trees efficiently while receiving immediate feedback.

This work-in-progress paper presents this VR tool's design and preliminary evaluation in an undergraduate programming languages course. In the future, we plan to conduct a comparative study involving three different platforms - a traditional teaching method, a web-based drag-and-drop tool, and the VR tool, to assess its effectiveness. By combining VR's immersive potential with automated grading capabilities, this approach aims to redefine the teaching of syntax trees, making it more engaging, interactive, and scalable for students. After the tool's effectiveness study, we plan to conduct a user acceptance study to assess how the tool might affect educators' pedagogy.

2 Related Work

Teaching parse trees and syntax analysis is fundamental to computer science education, particularly in compiler design and programming language courses. Traditional approaches typically involve students manually constructing/drawing parse trees for given expressions. This method, while effective for introducing the concept, can be labor-intensive and repetitive, often leading to student disengagement. Furthermore, educators face challenges in providing timely feedback because of the time-consuming nature of grading these manual drawings.

To mitigate these challenges, various interactive tools have been developed to enhance the learning experience. For example, Arboratrix is a graphical parse-tree editor that enables users to build trees by defining tree nodes and arranging them via a drag-and-drop interface[1]. It supports saving parsed sentences as XML files, thereby aiding both learning and auto-grading. Similarly, platforms like Syntax Tree Generator allow users to generate and customize tree diagrams from text input[2]. Another example, the Parsing Algorithm Visualization Tool (PAVT), goes further by animating parsing algorithms step-by-step, enhancing understanding of complex syntax analysis processes[3]. However, while these tools provide interactivity, they are primarily two-dimensional and lack the immersive and engaging environment that VR can offer.

On the other hand, the application of VR in education has shown promise in creating engaging and interactive learning environments. VR has been applied in various fields, such as teaching geometry and medical training, to make abstract concepts more tangible. For instance, NeoTrie VR provides a fully immersive virtual space for students to interact with geometric shapes, thereby improving spatial understanding[4]. Their studies have demonstrated that the use of Neotrie VR in real classrooms stimulates the acquisition of 3D visual and spatial thinking among students. Malik *et al.* conducted a systematic literature review on applications of VR in computer science education[5]. They suggested future research focus on refining interaction techniques to deepen the level of immersion within virtual reality environments. To the best of our knowledge, the application of VR specifically for teaching syntax trees remains unexplored.

Our work addresses the gap by introducing a web-based VR tool specifically designed for syntax tree construction and analysis. Unlike existing two-dimensional tools, our VR application immerses students in a virtual environment where they can intuitively construct trees through drag-and-drop interactions. This immersive embodiment approach fosters engagement and a deeper understanding of hierarchical structures. Moreover, the tool incorporates auto-grading capabilities, providing immediate feedback while reducing the grading workload for educators.

3 Methods

This section describes the VR tool and its functionality, followed by the proposed evaluation study.

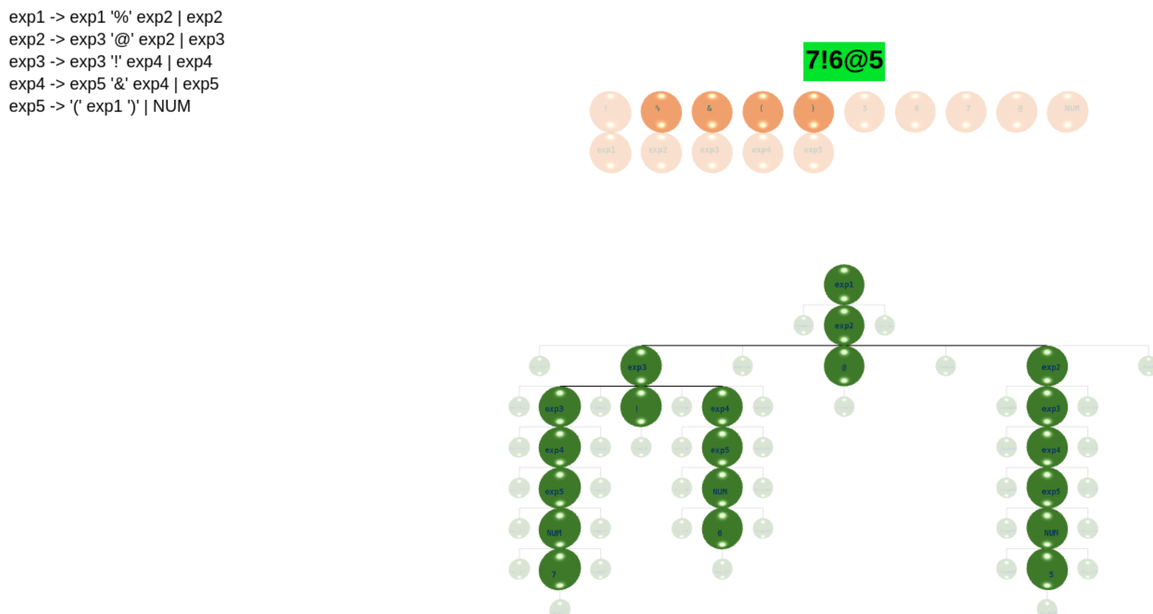

Figure 1: An example of a question shown in the VR tool. At the top, it shows the asking programming expression. Below the question expression is followed by draggable nodes (orange) representing the components of the expression and the BNF grammar. Students can drag the orange nodes to the working space below to construct the syntax tree.

3.1 The VR Tool

We used WebXR, an open standard that enables immersive experiences directly in web browsers, to develop the VR tool. We chose WebXR for its cross-device compatibility, allowing students to access the tool through any device with a supported browser (*e.g.* Google Chrome) and a VR headset (*e.g.* Meta Quest 3). However, we recognize that performance may vary across different devices, particularly on older PCs or lower-end mobile devices. To address this, future development will provide various levels of rendering quality and interaction responsiveness. We plan to make the VR tool publicly available on GitHub pages, allowing educators and students to integrate it into their learning environments and further contribute to its development and evaluation.

WebXR also conveniently supports "in-line" browser visualization, allowing direct comparative study between browser-based and VR-based tools while maintaining an identical learning task. The difference between these two interfaces is the user interactions. In VR, students interact with their hands or via game controllers, while in the "in-line", they use mouse-and-keyboard.

The tool utilizes WebGL for rendering interactive 3D environments and Babylon.js to handle the graphical interface. It offers an intuitive drag-and-drop interface within the immersive VR environment. When the page is loaded, the tool reads the question configuration and initializes the setting. The programming language grammar in Backus-Naur Form (BNF) and an expression are then displayed to students (see Figure 1 as an example). To the left of the area, a panel shows students the BNF grammar; on the top, students can see the asking expression. Below the question is the syntax tree construction area, in which "ghost" nodes (semi-transparent green in Figure 1) are displayed to annotate where students can insert new nodes to the tree. Students can drag and drop any nodes from the inventory above (orange in Figure 1) to construct the corresponding syntax tree to the expression. The students can also remove any node by dragging the node back to the inventory. Figure 1 depicts an example of a constructed syntax tree. Practice examples can be found in the footnote[1,2].

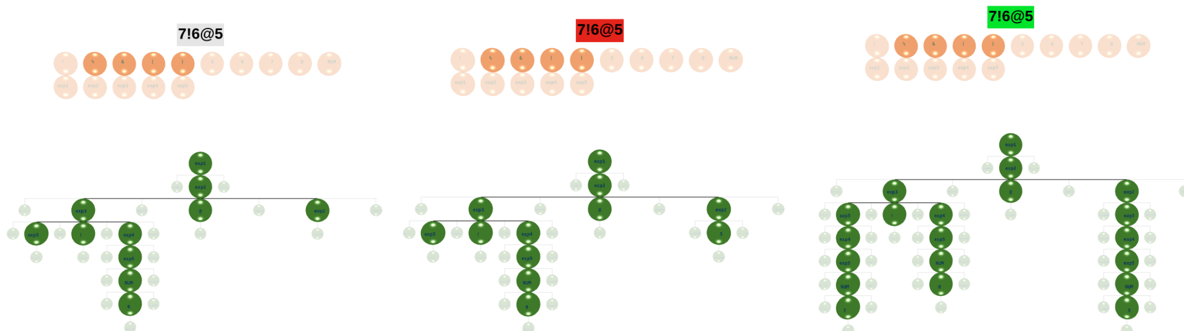

Figure 2: In the immediate feedback mode, the question expression changes its color immediately after the student drags a node to the tree. If the constructing tree partially matches the answer (*i.e.* the constructing tree is a sub-set of the answer tree), the question expression will remain gray (left). Otherwise, it turns red, as shown in the middle. If it is the correct answer, it turns green (right). In the quiz mode, the question expression always remains gray.

The tool supports different feedback modes. First, the immediate feedback mode lets students see if the newly inserted node is incorrect by making the question expression red, as depicted in Fig. 2 left. We assume the grammar is not ambiguous; therefore, there is always a unique solution, allowing us to automatically grade the student's solutions. The immediate feedback mode is suitable for in-class and at-home practices. Second, the quiz mode delays the feedback until students finish the tree construction. In this mode, summative feedback is provided at the end to allow students to review their mistakes (see an example in Fig. 2 right). Last is the exam mode for in-class exams, where no feedback is provided and time is constrained.

3.2 The Research Method

To evaluate the tool's effectiveness, we will conduct a mixed-method study involving both qualitative and quantitative methods. We plan to recruit undergraduate computer science students in a programming language course, where participation is voluntary. The participants will be randomly but evenly divided into three groups - one receiving instructions using a text-based tree construction tool[2] (our baseline reference), one using a browser-based drag-and-drop tool (non-VR platform), and one utilizing the our tool (VR platform). To ensure statistical power to detect "large" effect size[6], we need about 30 students in each group, computed using G*Power[7].

[1]Browser: `https://eg.bucknell.edu/~scl019/tool/start_practice.php?q=q1`
[2]VR: `https://eg.bucknell.edu/~scl019/tool/start_practicevr.php?q=q1`

3.3 Study Design

The programming languages course at Bucknell University is a teaching-to-mastery course in which students must complete four quizzes with increasing difficulty for the syntax tree topic. Since the number of enrollments is 34, the experimental design will be parallel-group triple-anonymous, ensuring balanced allocation among the three groups. The study will be performed during the last three syntax tree quizzes. After successfully finishing a level, the student will switch to a different group. Before students attempt the quizzes, they have free access to the same tool they will use for practice. Students will be given clear instructions about the study procedure before participating. Students can also switch groups if they strongly prefer a particular platform, allowing us to assess whether students strongly prefer a specific intervention. We will also collect student demographic information to examine if there is any correlation between their preferences and their demographic background.

3.3.1 Qualitative Measures

When comparing the three experimental platforms, we aim to understand students' engagement and self-efficacy. We collect self-reported survey data from students after each successful quiz attempt. The survey consists of ten questions, as shown in Table 1. The qualitative data will be analyzed thematically to identify trends and patterns in student experiences. The analysis will provide a holistic view of the tool's effectiveness and insight into VR tool development.

Q1	How engaging is the *intervention*?
Q2	How easy can you follow the instructions when using the *intervention*?
Q3	How likely will you continue using the *intervention* for your study?
Q4	How much do you prefer using the *intervention*?
Q5	How much effort do you need to put in when using the *intervention* to learn the concept?
Q6	How much do you find the content engaging?
Q7	How confident are you in describing the learned concept?
Q8	How confident are you in solving problems related to the learned concept?
Q9	How much more have you learned about the concept?
Q10	How much did the *intervention* help you understand?

Table 1: The proposed ten questions of the engagement and self-efficacy survey. Each question has a Likert scale answer scaled from 1 to 5, where 1 is the least (negative) and 5 (positive) is the most agreement with the question.

3.3.2 Quantitative Measures

Quantitatively, we will measure the learning outcome and the tool's efficiency by capturing the quiz completion time, number of attempts, accuracy, and score improvement using a post-quiz test (four tests in total for each student - one after each level). The collected data will be analyzed statistically to identify significant differences in performance metrics. The VR group's results will be compared against the other two platforms.

4 Preliminary Results

The research team received the IRB approval for the proposed study. In Fall 2024, we conducted a pilot study on the VR tool using the survey in Table 1. In the pilot study, we recruited four students who enrolled in the programming languages course at Bucknell University. All students were new to the syntax tree content and had limited to no experience with VR.

The pilot study shows overwhelmingly positive student perceptions of the VR tool. Across all ten survey questions, the responses are consistently positive, with most mean scores above 4.25, suggesting strong student engagement and high self-efficacy. Table 2 lists the individual survey results. Notably, the highest-rated questions are Q1 (*How engaging is the intervention?*) and Q4 (*How much do you prefer using the intervention?*), related to engagement and enjoyment. We note that one student rated 1 for Q5 (*How much effort do you need to put in when using the intervention to learn the concept?*). We suspect that the wording of the survey might confuse the students to interpret 1 as requiring the least effort. In the coming study, we will revise the questions to more concise descriptive sentences to address this issue. Furthermore, we will revise the survey to assess further interaction experience, student preference, and auto-grading efficiency.

	Stu. 1	Stu. 2	Stu. 3	Stu. 4	**Mean**		Stu. 1	Stu. 2	Stu. 3	Stu. 4	**Mean**
Q1	5	5	4	5	**4.75**	Q6	4	5	4	5	**4.5**
Q2	5	5	3	3	**4**	Q7	4	3	4	5	**4**
Q3	5	4	4	5	**4.5**	Q8	5	3	4	5	**4.25**
Q4	5	4	5	5	**4.75**	Q9	4	4	4	5	**4.25**
Q5	1	4	4	5	**3.5**	Q10	4	4	4	5	**4.25**

Table 2: The pilot study results demonstrate that the VR tool is a promising intervention. Students found the VR tool easy to follow and engaging. They also felt learning more using the VR tool.

In addition to the survey, we asked an open-ended question (*any other comments/suggestions?*) to collect student feedback to improve the VR tool further. A student raised a concern that VR tools are inconvenient for glass-wearing users, which is interesting to consider. To further investigate it, we will also collect basic student information, including wearing glasses/contact lenses, to discover if there is a correlation between wearing glasses and engagement in using VR tools.

5 Conclusion

This work-in-progress paper introduced a web-based VR tool specifically designed to improve the teaching and learning of syntax trees in programming languages. A pilot study was conducted to show that the proposed VR tool is a promising and engaging intervention. By leveraging the intuitive drag-and-drop interface, students can construct syntax trees in a dynamic and interactive VR environment. Meanwhile, the constructed syntax trees in software allow educators to auto-grade the exercises and provide immediate feedback. The preliminary result highlights the potential of using VR to transform how foundational concepts such as syntax trees are taught. In the future, we plan to conduct a mixed-method study to further investigate the impact of using the proposed VR tool (with a headset) by comparing it with the traditional/baseline method (drawing or text-based tree generation) and the browser-based visualization.

References

[1] A. M. Corbí-Bellot and M. L. Forcada, “Arboratrix: A graphical parse-tree editor,” https://arboratrix.sourceforge.net/, accessed: 2024-11-30.

[2] M. Shang, “Syntax tree generator,” https://mshang.ca/syntree/, accessed: 2024-11-30.

[3] S. Sangal, S. Kataria, T. Tyagi, N. Gupta, Y. Kirtani, S. Agrawal, and P. Chakraborty, “Pavt: a tool to visualize and teach parsing algorithms,” *Education and Information Technologies*, vol. 23, no. 6, pp. 2737–2764, Nov 2018. [Online]. Available: https://doi.org/10.1007/s10639-018-9739-x

[4] J. L. Rodríguez, I. Romero, and A. Codina, “The influence of neotrie vr’s immersive virtual reality on the teaching and learning of geometry,” *Mathematics*, vol. 9, no. 19, 2021. [Online]. Available: https://www.mdpi.com/2227-7390/9/19/2411

[5] H. A. Malik, R. Ferdinand, B. Aurelius, M. Fajar, and P. A. Suri, “Applications of virtual reality in computer sciences education: A systematic literature review,” in *2023 International Conference on Informatics, Multimedia, Cyber and Informations System (ICIMCIS)*, 2023, pp. 659–664. [Online]. Available: https://ieeexplore.ieee.org/document/10348975

[6] J. Cohen, “A power primer.” in *Methodological issues and strategies in clinical research (4th ed.).*, A. E. Kazdin, Ed. Washington, USA: American Psychological Association, 2016, pp. 279–284. [Online]. Available: http://content.apa.org/books/14805-018

[7] F. Faul, E. Erdfelder, A.-G. Lang, and A. Buchner, “G*power 3: A flexible statistical power analysis program for the social, behavioral, and biomedical sciences,” *Behavior Research Methods*, vol. 39, no. 2, pp. 175–191, 2007. [Online]. Available: http://link.springer.com/10.3758/BF03193146