# Using Virtual Reality to Build Mental Models for Beginning Programmers

O'Brien, Tommy
*Dept. of Computer Science*
*University of Alabama*
Tuscaloosa, USA
ttobrien@crimson.ua.edu

***Abstract*—Introductory computer science (CS) courses are challenging for students all around the world. Educators give their students a multitude of examples, analogies, and resources, but pass rates are still abysmal. Virtual reality (VR) is a new and quickly growing field, but no attempts to use it as a means of instruction for CS have been made. In this paper, I attempt to combine VR technology with an underutilized approach of teaching by prioritizing students' understanding of memory. I created a VR lesson emphasizing memory and designed a study to determine if this form of instruction could be beneficial in CS classrooms. While data was unable to be collected at this time, I discuss what information will be gathered and analyzed.**

***Index Terms*—computer science education, virtual reality**

## I. Introduction

Nowadays, people commonly talk about the rapidly growing number of IT positions available. The supply of quality graduates from CS or other related degrees must increase in accordance with this demand. As a CS major, I have seen a frustrating number of my peers struggle greatly, get disheartened that they could never learn to code, and then drop out early on in the program which causes me to wonder how the demand will be met in the job market. With the current teaching methods failing to educate so many students, something needs to change.

Based off the following research, I now propose a new method of assisting CS students to succeed in their introductory courses by using VR to sharpen their mental models of how programs affect a computer's memory. The VR lesson that has been developed is discussed in detail. Although no data was able to be collected in the spring of 2020, the plan for data collection is outlined. The combination of VR with a memory-first approach to instruction could powerfully impact the success of students in CS classes going forward.

## II. Related Works

A mental model refers to how one understands something operating and the set of rules under which he or she believes that thing operates. The goal of Bornat's research was to discover whether or not students with a consistent mental model are more successful in their introductory programming courses. The results of this study are inconclusive. However, there are still important facts to be picked up on. The primary research was in the UK and Australia which shows students are struggling worldwide with CS1. Students were given a test involving assignment statements executed in sequential order, one of the most basic and fundamental concepts in programming. With varied results from students even after being taught the material, it is clear that one cause of the poor performances is a lack of ability for students to generalize natural life occurrences and algebraic concepts to the arena of programming. A couple more important things to note from this research are that expert programmers build their programs bottom-up and that understanding a concept requires a valid mental model to have been developed. [1]

The focus of Hosseini's study was to evaluate the effectiveness of annotated code examples against animated code examples for students in an introductory Java course. An initial realization is that students need non-mandated, beneficial practice material that they want to use. Program visualization in the animated examples shows a graphical representation of an executing code snippet. An analysis of their findings shows that time spent with animated examples positively impacts grades. Oppositely, the more time students use annotated examples the worse the grade is. Clearly, program visualization is beneficial to students' understanding of programming concepts. [2]

Mahoney conducted a study in a local computer center for urban youth. All of the learning that occurs there is by the students' own volition. An interesting statistic is how only 9% of a youth's time is spent in school, and that time is divided among many subjects. Children need to spend time outside of an educational institution to dive deep into a subject. The local computer center has many applications available for the youth to experiment with. When Scratch, a drag and drop coding environment, was introduced it quickly became the predominant application in use at the center. It is in use by both boys and girls of color. Through Scratch, students learn most core programming concepts with minimal external assistance and enjoy their time doing so. [3]

VR was implemented in an introductory programming course by Thomas. An important thing to note is that electrical engineering majors make up a significant portion of CS1 classes, yet they leave with minimal understanding of programming concepts. Also, CS1 courses focus primarily on small, isolated, easy problems which is contrary to industry application where these same people will work with complex systems in a large framework. With the previous point in mind, CS1 students were given a project to create a 3D scene in VR.

This required students to use more creativity and engage in a larger framework. Surveys were given to students before and after the project, and stark differences are observed between the two results. After completing the VR assignment, students become much more interested in programming. [4]

Several key points can be gathered from this prior research. Primarily, that learning how to program is an extremely challenging task no matter what someone's background is. Moreover, additional materials, especially those using graphical representation, prove beneficial in learning CS topics for students of various backgrounds. Utilizing VR caused students to become more interested in, engaged with, and passionate about their programming class. The current means of communication of CS material in a college course fails to bridge the gap between students' current knowledge and the application of that knowledge to their course. What is lacking from prior research also needs to be noted. Currently, there is a minimal prioritization on teaching students how memory works in order for them to develop a correct mental model. While VR is engaging to students, no efforts have been made to create study materials for CS1 topics with VR.

## III. Approach

Clearly, there is a worldwide dilemma with the current ways of teaching CS1 being inadequate for sufficient student progression in programming abilities. The University of Alabama (UA) is certainly included in this problem. The current way of teaching CS100 at UA has 3 50-minute lectures a week, online homework and readings, weekly labs, and several projects. Zybooks, an online animated textbook and assessment platform, is heavily utilized inside and outside the classroom at UA. Despite the numerous means of engagement with the material, students are failing and withdrawing at stunningly high rates and are leaving the CS department in droves. Too many students are not getting the one thing that they should receive for their time and money put towards this course, an understanding of the fundamentals of programming.

At UA, CS 100 is taught in the C language. C is more challenging to start with than Java or Python but has many long-term benefits. C is more closely connected to the hardware, so students get a better understanding of how memory works. After experimenting and researching with which language is best for their students to start with, UA has settled upon C for good reason. The challenge now is to figure out a better way for students to fully grasp C. Anecdotally, it has been observed that one of the trickiest parts of the class for many students is working with arrays of characters (c-strings). With c-strings being such a stumbling block for beginner programmers, it is an important topic to be able to teach well.

This research takes two unexplored angles for teaching CS1 and combines them to create a new approach to help students of all backgrounds succeed in an introductory programming class. The first facet is using VR as the means of representation. It has been successful for other subjects, but has not yet been used to explicitly teach computer science lessons. VR is growing in popularity and certainly is a technology that will progress in the near and far future. The educational realm must keep up to date with what students are using outside of school. All students learn in different ways, and VR could be the way students effectively receive information when projects and lectures do not suffice. Secondly, this approach prioritizes learning memory reliant programming concepts. Most often, students are taught just how to program in one language without understanding what exactly the words on the screen are doing to the computer's memory. By putting memory first, a student is better equipped to form a correct mental model of what their code is actually doing. CS1 students are more capable of detecting and fixing errors with their C code when they know how their code is manipulating memory.

```
1: char appt[] = "Wednesday 4pm";
2: char newDay[] = "Monday";
3: char newTime[] = "8pm";
4: int i = 0;
5: for (int j = 0; j < strlen(newDay); j++)
6:     appt[i++] = newDay[j];
7: for (int k = 0; k < strlen(newTime); k++)
8:     appt[i++] = newTime[k];
9: printf("%s\n", appt);
```

The above code snippet is the code that is implemented in the VR environment for students to engage with. The code focuses on a few tricky yet key programming concepts. It is important to understand the observation that c-strings are difficult for students in CS1. This example in VR visually displays a linear stack that grows backwards. This improves a student's mental model of what is happening on a lower level with their programs. One of the key features of c-strings is the end of string character, '\0'. Students often forget about this character when manipulating c-strings which causes unexpected program behavior. The code demonstrates that every character in the c-string up to '\0' is printed. The intention of the code is clear; the day and time are being replaced with new values. However, the program fails to carry out that task despite the fact that it looks correct, especially to a novice. Misunderstandings about c-strings are further attacked in the above code by traversing parallel arrays. This type of problem is mentally strenuous and requires a better understanding of how c-strings are stored in memory. When showing the swapping of characters in memory in VR, the abstraction of letters is used opposed to the true ASCII representation of the letters in the c-strings. This abstraction makes it simpler for the students to understand what is happening with the data, and the concept of ASCII encoding is simple enough to be bridged later. By forcing students to embrace the challenge of manipulating multiple c-strings concurrently and providing a more than adequate way for them to learn this concept, students can be more successful working with c-strings going forward.

The VR lesson was created using React 360, and is set up with 6 panels and 28 states. Each state shows the low-level

steps of the program's execution. An annotated snapshot of the VR environment can be seen in Figure 1. The top panel has two buttons ('Next' and 'Back') that allow the user to incrementally move forward and backward through the states. The center panel shows the code snippet, and the highlights the piece of code that is currently executing. The bottom panel displays the stack. The stack grows backwards to best emulate a stack's true behavior. The stack starts with blank cells that each represent one byte. As variables are declared and go out of scope, the stack grows and shrinks respectively. The bytes that are being affected by the currently executing code are highlighted in yellow. The top right panel shows comments to give annotation and further explanation for each step. The bottom right panel display the output, and only appears if there is data being printed. Lastly, the panel on the left is a stack of color-coded variables. The variables only appear as they exist in memory. They serve as buttons that can turn on and off a highlight of the same color for the corresponding memory cells of that variable. There are several components to this environment, but each exists to help students further engage with and understand the lesson.

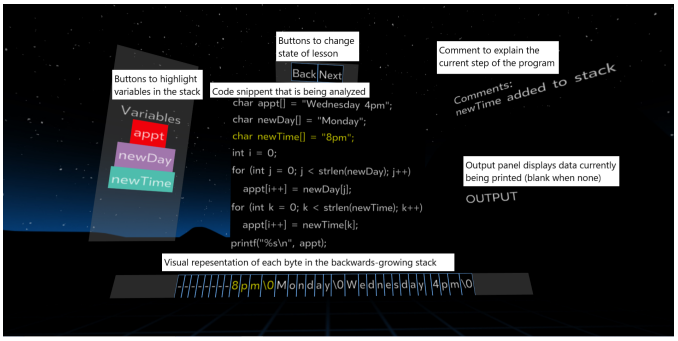


Fig. 1. Annotated snapshot of the VR lesson's environment when the newTime c-string variable is being declared and initialized.

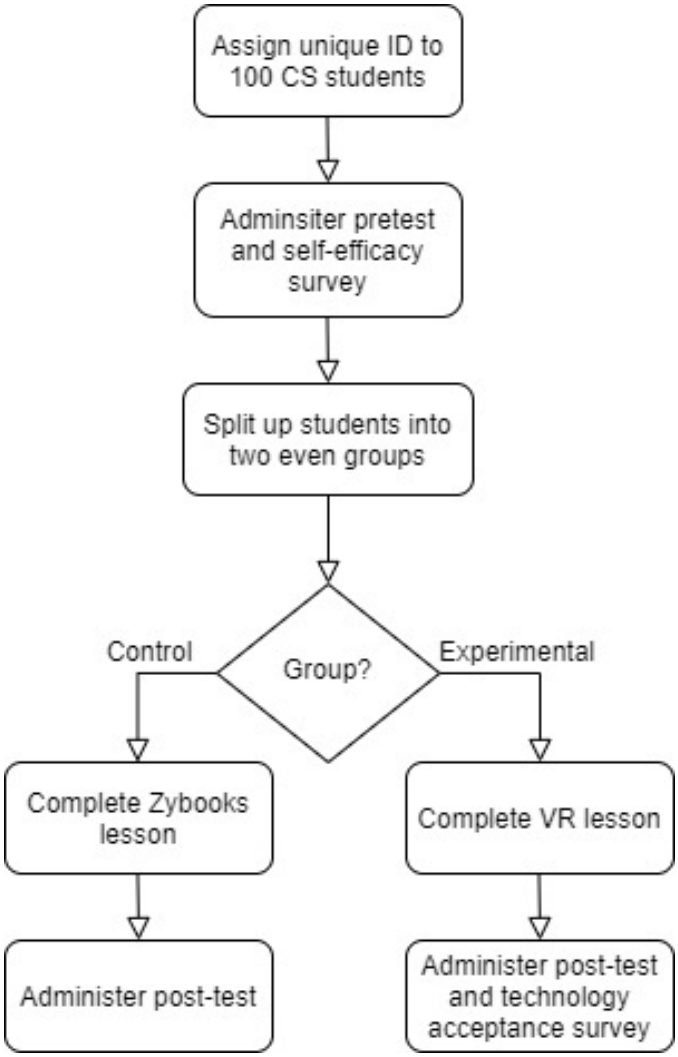


Fig. 2. Flowchart outlining the proposed experiment to determine the effectiveness of the VR lesson.

## IV. Data Collection

Unfortunately due to COVID-19, testing was unable to proceed as planned. However, this section will discuss the testing methodology that will be implemented. Figure 2 shows a flowchart outlining the study that is described in this section. The sampling population will be from UA's CS100 and CS101 classes. 50 students from each class will be selected and given an unique testing id. Both of these classes use an online textbook from Zybooks for homework, labs, tests, and supplementing classroom learning. The students will be administered the pretest that can be seen in Figure 3. This pretest will establish an adequate baseline to determine each student's starting competency and confidence with c-strings. In addition to the pretest, students will be given a self-efficacy survey. Karsten and Roth's research on computer self-efficacy was used in developing the demographic and self-efficacy survey in Figure 4 that will be collect relevant and important data for this study. [5]

After the pretest and survey, students will be split up into 2 equally sized groups, a control group and an experimental group. The control group will have 50 students (25 from CS100 and 25 from CS101). The experimental group (the 50 remaining students) will use the VR lesson. The control group will go through the Zybooks lesson on c-strings as UA studnet normally would. Students in the experimental group will don an Oculus Go VR headset. These students will be taken into the experiential learning environment of VR where they will walk through a lesson demonstrating the ongoing effects to the stack as they walk through the code of the first pretest problem. Figures 5-7 show snapshots of the environment at different states during the lesson.

After completing the VR lesson, both groups will be administered a post-test consisting of problem 2 of the pretest and another challenging c-string tracing problem, and they will be asked to rate how confident they are in their answers. The post-test serves to evaluate the effectiveness of a VR experiential learning lesson after one use for a specific topic. In addition to the post-test, a technology acceptance survey will be administered to the experimental group. This survey

Testing id:

Provide the output to both C programs below.

1) Code

```
char appt[] = "Wednesday 4pm";
char newDay[] = "Monday";
char newTime[] = "8pm";
int i = 0;
for(int j = 0; j < strlen(newDay); j++)
    appt[i++] = newDay[j];
for(int k = 0; k < strlen(newTime); k++)
    appt[i++] = newTime[k];
printf("%s \ n", appt);
```

1) Output:

1) On a scale of 1 (very unconfident) to 5 (very confident), rate how confident you are that your answer is correct.

2) Code

```
char str1 = "abcdef";
char str2[6];
for(int i = 0; i < 6; i++)
   str2[i] = str1[6-i-1];
printf("%s\n", str1);
printf("%s\n", str2);
```

2) Output:

2) On a scale of 1 (very unconfident) to 5 (very confident), rate how confident you are that your answer is correct.

Fig. 3. This is the pretest that students took before using the virtual realty lesson.

| Testing id: |
|---|
| Demographic Information |
| Gender: |
| Age: |
| School Year: |
| Major: |
| Race: |
| Nationality: |
| Number of previous CS and programming classes: |
| Circle the following instruction styles that are effective for you: Oral Visual Reading/Writing Hands-on/Experimental |
| Self-Efficacy Survey (Scale: 1 = strongly disagree ~ 5 = strongly agree) |
| I understand words/terms relating to computer software: |
| I understand words/terms relating to computer hardware: |
| I feel confident in my ability to write a simple program: |
| I feel confident in my ability to write intermediate level program: |
| I feel confident writing a program in the C language: |
| I feel confident using arrays of character: |
| I understand how variables are stored in memory: |
| I understand how assignment statements affect memory: |
| I feel confident in my ability to solve a tracing problem: |
| I enjoy my current CS class: |
| I feel that the instruction in my CS class is effective for me: |

Fig. 4. This is the survey that collects each student's demographic information and evaluates their self-efficacy.

will collect important information to help further evaluate the effectiveness of the VR lesson and what can be improved relating to the content and environment. This survey can be seen in Figure 8, and was partially derived from Park's technology acceptance model. [6] The open response portion will provide more detailed information on how the lesson and environment can be enhanced.

The goal is to carry out this study in the fall of 2020. There are several different correlations that will be analyzed once the data is collected. Some of these are performance on the pretest vs performance on the post-test, confidence on the

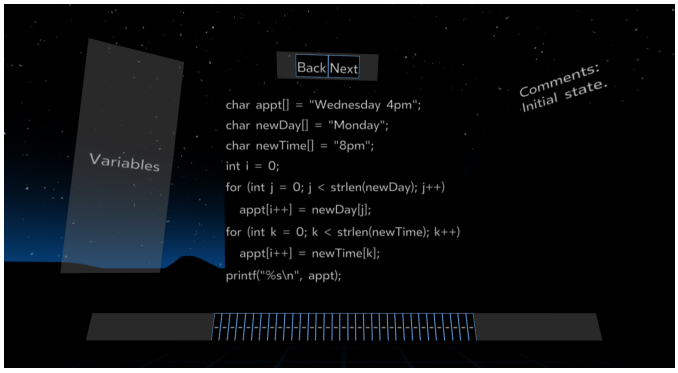


Fig. 5. Initial state of the VR lesson.

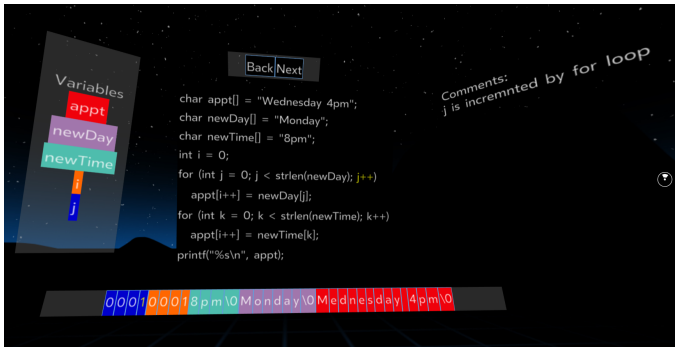


Fig. 6. This is a snapshot of the state of the VR when j is being incremented during the first pass of the first for loop. Each variable in the stack on the bottom of the screen has its color coding of the memory allocation toggled on.

pretest vs confidence on the post-test, confidence vs performance, demographics and their corresponding performances, and post-test performance of the Zybooks group vs post-test performance of the VR group. It is not just CS majors that take the introductory CS courses at UA. Electrical engineering, computer engineering, and some math majors need to pass at least CS100 in order to graduate from their programs. Most students in the class will likely be freshman with little to no prior programming experience. In order to better serve these students who will most likely struggle more than others, the standards must not be lowered but alternate methods like VR must be explored, and the correlation should be positive for their performance after using the VR lesson. We will also study the correlation between performance and race in order to better teach students of all races. The expectation is that the collection of students will perform significantly better and have greater confidence in their abilities after using the VR lesson.

## V. Conclusion

The route that will be pursued after this study will depend on the results that will be collected. Using VR to teach computer science is on the cutting edge of technology in education. There are a few different paths that will be explored to further refine the technology. First, a focus group will be assembled

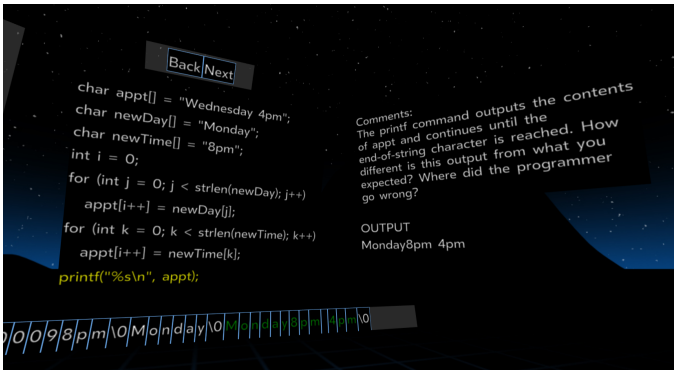


Fig. 7. The last state is shows the output of the program.

| Testing id: |
|---|
| Technology Acceptance Survey (Scale: 1 = strongly disagree ~ 5 = strongly agree) |
| I find the lesson easy to use: |
| I find virtual reality more engaging than current instruction: |
| This lesson increased my understanding of the C language: |
| This lesson increased my understanding of arrays of characters: |
| This lesson improved my understanding of how variables are stored in memory: |
| This lesson improved my understanding of how assignment statements memory: |
| The virtual reality environment was easy to use: |
| The virtual reality environment worked well for this lesson: |
| This lesson was more confusing than helpful: |
| I want more, similar lessons to assist in learning more programming concepts: |
| Learning through virtual reality is a wise idea: |
| I am positive towards using virtual reality in education: |
| Virtual reality would improve my classroom performance: |
| Virtual reality would increase my academic productivity: |
| Virtual reality would make it easier to study course content: |
| Open Response Questions |
| What would you fix about the environment? |
| What was your biggest takeaway from this lesson? |
| What are you still confused about relating to c-strings and memory? |
| Other comments: |

Fig. 8. This is the technology acceptance survey.

of CS teachers, students, and participants of this study. The focus group will assist in refining the VR's visual design and methods of explanation. Next, the technology will be converted to a full lesson for a CS100 topic for all students to experience. This will aid in determining the tool's effectiveness in the classroom setting and feasibility of implementation. One possible way that this could be implemented is with mixed reality, ie one student or instructor wears the headset and there is an additional external display for others to view what the user is seeing. Finally, experiments will be conducted to trial this tool with higher level CS topics in classes such as Data Structures & Algorithms, Operating Systems, and Programming Languages. Having a correct mental model of the interaction between a program and memory is essential and fundamental for anyone to pass a computer science course or pursue a future in computer science.

In summary, the current methods of teaching CS are deeply flawed since students are failing and dropping in high numbers. VR is a new technology that has limitless applications, yet there has been no previous attempts to disseminate knowledge in a CS class through VR. Understanding how memory works is fundamental to being able to code correct programs. React 360 was used to create a VR environment that allows a user to go step-by-step through a code snippet. The designed lesson primarily focuses on c-strings. A study with a pretest, post-test, self-efficacy survey, technology acceptance survey, control group, and experimental group was designed in order to produce data that will determine the current and potential effectiveness of the VR lesson. Ultimately, I am confident that using VR to build mental models for beginning programmers can provide a revolutionary change in CS education for students of many backgrounds.

## REFERENCES

[1] R. Bornat et al., "Mental Models, Consistency and Programming Aptitude Proceedings of the Tenth Conference on Australasian Computing Education - Volume 78," pp. 53-61, 2008

[2] R. Hosseini et al., "Animated Examples as Practice Content in a Java Programming Course," Proceedings of the 47th ACM Technical Symposium on Computing Science Education - SIGCSE '16, 2016

[3] J. H. Maloney et al., "Programming by choice: urban youth learning programming with scratch," in 40(1), ACM, 2008

[4] S. J. Thomas et al., "Recruiting Virtual Reality Troopers: Bringing Introductory Programming Projects to Life in VR," 2018

[5] R. Karsten and R. M. Roth, "Computer Self-Efficacy: A Practical Indicator of Student Computer Competency in Introductory IS Courses," Informing Science, vol. 1, no. 3, pp. 61-68, 1998

[6] S. Y. Park, "An Analysis of the Technology Acceptance Model in Understanding University Students' Behavioral Intention to Use e-Learning," Educational Technology & Society, vol. 12, no. 3, pp. 150-162, 2009