Creative Coding for All Students

M. R. Zargham and **C. K. Danduri**
Department of Computer Science
University of Dayton
Dayton, Oh USA

Abstract - *This paper represents progress of our ongoing effort on new way of teaching an introductory programming course. Students use a simple environment, called Innovative Coding (IC), to write a code for designing a game or a virtual world. As they code, they see the development of their world in a 3D environment step by step. Once they have completed their design, they can wear a VR headset to interact on their design in a virtual environment. Currently, IC supports five different types of landscapes (desert, fantasy island, mountain, farm, and city), 8 different princesses, 8 boys with different ethnicities, 10 different animals, 10 different monsters, one helicopter, and one car. Excluding landscapes, all other objects are animated being able to perform many tasks, such as walk, run, jump, eat, sleep, dance, etc. IC also supports many static objects that are not animated, such as: tracks to construct different roller-coasters, plants, flowers, cabins, decorating objects, and walls.*

Keywords: VR Headset; Roller-Coaster; Virtual Reality; Computer Science Education; Programming.

1. Introduction

We have created an innovative coding (IC) environment in which students can write simple codes consisting of several statements to create a game or a virtual world. This paper will explain the present functionality of IC, that emerged from a project funded by AAC&U that was intended to broaden participation in computer science at all levels.

At present there are several graphical coding platforms that many K-12 schools use; the most popular ones are: Alice 3D by Carnegie Mellon University, Scratch by MIT Media Lab, and Snap by Berkeley [1, 2, and 3]. Although these platforms provide visual programming environment and are popular among some students, they still have some drawbacks such as visualizing the entire programming process in a drag-and-drop manner, teacher training, and difficulty of data and control flow observation [4, 5, 6]. The students working with such systems may struggle to move to a traditional programming environment due to the lack of experience to linking between the programming concepts acquired from those visual tools and the implementation details in an actual programming language. IC overcomes these drawbacks by integrating both the traditional text-based programming platform and immersive virtual environment. IC creates an environment in which students can write simple codes consisting of several statements to create a world of their choice in a 3D and Virtual Reality environment. VR goes beyond a simply visual stimulus, and allows students to become directly involved, experiencing the code they write in a very tangible, interactive, and expeditious scheme(strategy). This allows them to experience both the successes and failures of writing (creating, developing) correct code versus incorrect code. Students can appreciate both the scale and the scope of the results of their code immediately, which prompts them to become invested in the quality of their coding skills, and strive to continually improve their logic, programming and code writing abilities. Through this Immersive and Responsive Visual Stimulus Learning, students become more engaged in coding. In general, the idea is to teach the introductory programming class in such a way that follows project-driven learning process and encourages students to develop problem solving and teamwork skills while fostering creativity and logic. The goal is to not only provide students with some "programming maturity," but to also engage them with existing projects related to their interest.

2. Current Functionality of IC

Currently, students can design their projects in different landscapes (backgrounds) using a variety of objects. For example, they can use the environment to design a 3D modeling of a roller-coaster based on their choice of options. Presently, there are several statements users may select from and implement, including statements that allow students to declare variables, assign values to variables, arithmetic and logical expressions, array statements, object statements (such as designing a track/wall segment by defining its length, direction, and angles), assignment statements, conditional statements, repeat statements, animation statements, function statements, and I/O statements. Figure 1 represents a sample code (on the right side) for a roller-coaster and its design. In the editor mode, the small window on the right-top corner is utilized by the user to enter his/her code for building the roller-coaster. During this process, the user can see the progress of his/her code in 3D. When the user is satisfied with the design, he/she will click the 3D or VR button to start the animation. If a VR headset (HTC Vive) is connected to the computer, the users can actually be given the sense that they are riding on their own roller-coaster design; see Figure 2. Note that it is not required to have VR headset to utilize IC; the 3D mode also allows the user to be engaged and interact with his/her created environment.

The code editor allows students to add or delete a statement in their partial code very easily. The effect of such modification on their partially built roller-coaster can be seen instantly by clicking on "wooden" or "steel." The user can then use "*for-loop*" to build a large track by repeating a section of the track many times. The "*while-loop*" can be used to design a track that reaches a certain height. The "*if*" statement can be used to make a track dynamic by assigning it different types of objects.

Figures3 and 4 represent different scenes and objects of the current version of IC. Currently, IC supports five different types of landscapes (desert, fantasy island, mountain, farm, and city), 8 different princesses, 8 boys with different ethnicities, 10 different animals, 10 different monsters, one helicopter, and one car. Excluding landscapes, all other objects are animated being able to perform many tasks, such as walk, run, jump, eat, sleep, dance, etc. IC also supports many static objects that are not animated, such as: tracks to construct different roller-coasters. plants, flowers, cabins, decorating objects, and walls. Different types of interaction are possible with the designed world, such as riding a roller-coaster, flying a helicopter, and driving a car (see figures 5 and 6, below).

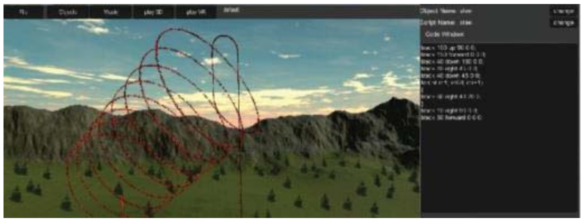

Figure 1. The editor window for designing a roller-coaster.

Figure 2. A class of 22 5th grade students designing different roller coasters

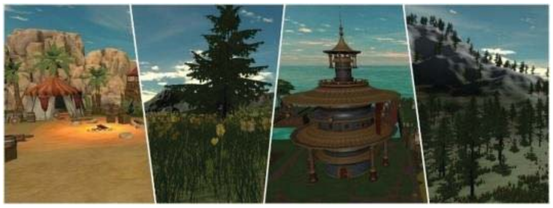

Figure 3. Landscapes supported with current version of IC.

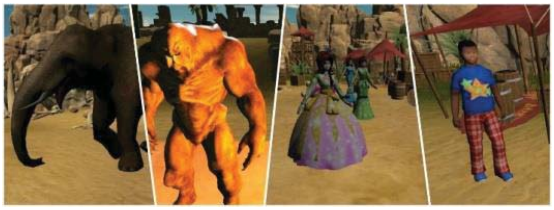

Figure 4. A sample of four different objects, animals, monsters, princess, and boys.

Early in 2017, the current version of IC was used in two different situations: a field trip of a fifth grade class from Harman Elementary School, and the Dayton Techfest 2017. In both occasions the outcomes were outstanding. In a short period of time students were able to design complex roller-coater using *for* and *while*-loops. They were also able to explore IC to experience interacting with helicopter, car, humans, and animals. In case of fifth grade, students were so engaged in coding different scene on IC that they did not want to go back to school. They all expressed that their experience was AMAZING! Students were glad to receive a copy of software on a memory stick to do coding at home.

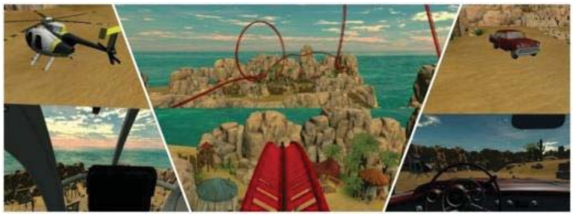

Figure 5. Different types of interaction with designed world: riding a roller-coaster, flying a helicopter, and driving a car.

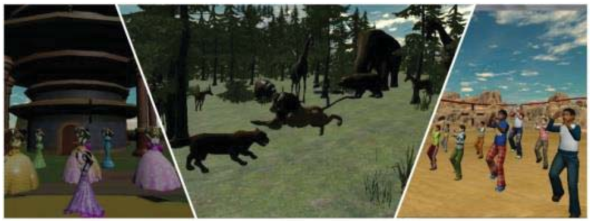

Figure 6. Different environments with programmed objects.

At present we are emphasizing on one project (roller-coaster). In future we will be adding more projects such as: car racing, 3-D store, and games.

Once students have learned the basics of programming through various design projects, they will be encouraged to design games in an environment such as Unity by learning more programming statements and techniques.

To teach a programming course in this way, will attract students with diverse educational backgrounds to form teams to tackle various problems. Beside students in K-12, computer science, electrical/computer engineering and business, students from other disciplines such as mathematics, physics, chemistry, biology, communication, and arts are encouraged to enroll and participate.

To demonstrate the simplicity of the IC, a possible code for a roller-coaster can be developed as below:

```
track 40 up 60 0 0;
track 40 down 120 0 0;
track 30 up 60 0 0;
track 30 forward  0 0 0;
var i;
generate j 1 400;
i = j%4;
if(i=0)   { pumpkin 20; }
else if(i=2)  {  goblin 20;}
else  { bear 20;  }
i = maximumheight;
j=currentheight;
display i j;
track 10 up 60 0 0;
while (j<i) {
track 1 forward 0 0 0;
j=currentheight;
}
track 40 down 120 0 0;
for(i=0,i<3,i++){
track 60 up 360 0 30;
}
brake 50;
track 10 forward 0 0 0;
```

3. Conclusions

In this paper, we have described a 3D environment that can be assigned in an introductory computer science course to retain students. The current version of IC has been used in two different situations: a field trip of a fifth grade class from Harman Elementary School, and the Dayton Techfest 2017. In both occasions the outcomes have been outstanding. Students all expressed that their experience was amazing! We are planning to expand our work to include variety of objects and coding tutorials. once it is completed, we will make it available through internet to all K-16 students.

4. References

[1] Rodger, S. H., Hayes, J., Lezin, G., Qin, H., Nelson, D., Tucker, R., Lopez, M., Cooper, S., Dann, W., and Slater, D, “Engaging Middle School Teachers and Students with Alice in a Diverse Set of Subjects,” In ACM SIGCSE Bulletin, vol. 41, pp. 271–275. ACM, 2009.

[2] Maloney, J., Resnick, M., Rusk, N., Silverman, B., and Eastmong, E, “The Scratch programming language and environment”, ACM Transactions on Computing Education, vol. 10, issue. 4, no. 16, pp. 1- 15, 2010.

[3] Harvey, B. and Monig, J., “Snap! reference manual,” p. 64. [Online]. Available: http://snap.berkeley.edu/SnapManual.pdf.

[4] Meerbaum-Salant, O., Armoni, M., and Ben-Ari, M., “Learning computer science concepts with scratch,” Computer Science Education, vol. 23, no. 3, pp. 239-264, 2013

[5] Robertson, J., Macvean, A., and Howland, K., “Robust evaluation for a maturing field: The train the teacher method,” International Journal of Child-Computer Interaction, vol. 1, no. 2, pp. 50-60, 2013.

[6] Wolz, U., Leitner, H. H., Malan, D. J., & Maloney, J., “Starting with scratch in CS 1,” ACM SIGCSE Bulletin, vol. 41, no. 1, pp. 2-3, 2009.