Enhancing Programming Skills of Engineering and Technology Students Using an Object-Oriented Multidimensional Desktop Virtual Reality (dVR) Framework

Magesh Chandramouli, Justin Heffron  
Purdue University, Calumet

Abstract

This paper proposes an interactive multidimensional desktop Virtual Reality (dVR) framework enhanced with touchscreen capabilities to stimulate interest in programming for Engineering & Technology (ET) students. This serves to strengthen ET programs by contributing to the following areas: creating ET students with superior programming skills, enhancing the quality of instruction, increasing student success by facilitating learning, and improving student employability. Programming is often misconceived as an extremely complex process and this creates a barrier in the minds of students. Students, especially those in the beginning stages, face several obstacles due to the lack of tools that can motivate them to pursue learning programming. If unattended, this can lead to student failure in programming courses and this can in turn affect the success rates, thus contributing to increase drop rates. To remedy this, a suitable approach that can persuade students needs to be developed to change students’ outlook towards learning program design and development. Programming skills are becoming increasingly critical for ET students irrespective of their major discipline. ET students with good programming skills can use programming as a key component for completing course projects and the inherent critical thinking and problem solving skills associated with programming is valued by prospective employers. However, learning to program, especially during the initial stages can be quite challenging and students may tend to get discouraged. To reduce the cognitive load and make programming a fun-learning activity, interactive virtual worlds with 3D are employed, whereby the user can dynamically interact, manipulate, modify, and re-arrange the programming elements represented in graphic format.

Introduction and Related Work

The main goals of this research are

1. To develop an inventory of programming and object oriented programming (OOP) concepts
2. To develop a multidimensional desktop VR (Virtual Reality) framework to demonstrate Programing and Object-Orientation (OO) concepts to students, and
3. To program the proposed framework to facilitate dynamic interaction to learn programming and object-orientation in an interactive manner. Open-Source software and web-friendly technologies are used to build this application to facilitate online hosting and dissemination.

Educational institutions across the nation are plagued by tuition and retention issues, and from the students’ perspective increasing educational costs and student loans are huge concerns. While there are numerous causes for such problems, a heavy dependence on proprietary commercial software may also be one of the factors that can increase the financial burden on the educational
The use of open-source programming languages (PL) ease the financial burden as there are practically no costs associated with their use.

A very practical way to ease the cognitive overload is to feel students enthused about the activity. It is difficult to make somebody feel motivated when the flow of instruction is unidirectional. For instance, in case of text-book oriented learning, typically students read a textbook and the information flow is from the textbook to the reader. The lack of interaction leads to monotony and when there the learner (student) hits learning blocks, there is not much option left but to keep reading the complex subject matter repetitively. While this kind of persistence is common among senior-level students, students at the beginner level may tend to get demotivated after few repeated attempts to understand a complex concept. The difficulty of successfully completing programming courses is one of the known factors that contribute to an increase in the dropout rates. This is a key concern to engineering and technology schools and academicians/educators have used several innovative ways to enhance the learning of foundational concepts in ET courses.

The use of non-traditional instructional tools and techniques to simplify pedagogics has been advocated by several authors (Driscoll, 2005, Chandramouli, Jin, Connolly, 2012). This study uses a portable desktop Virtual Reality (pdVR) based visualization to create virtual representations that facilitate user-interaction. This kind of framework supports active learning methods by incorporating problem-solving and project-based learning (PBL) which enable students to comprehend difficult conceptions by reducing the cognitive overload and by increasing the engagement (Mills & Treagust, 2003, Newby, Stepich, Lehman, & Russell, 2010).

Of late, a well-known and effective method is the use of games and interactive framework that is used to effectively impart programming skills to students. (Hernandez, Silva, Segura, Schimiguel, Paradela, and Bezerra, 2010, Papastergiou). The advantages of such display systems for 3d Visualizations include the following:
- The ability to generate and evaluate alternative representations
- Navigation in multiple modes (walk, fly, pan, zoom)
- Immersion
- Intuitive and real-time Interaction

3d Virtual worlds can be visualized using varied kinds of user-interfaces (UI) such as the following:
- Computer Assisted Virtual Environments (CAVE)
- Head Mounted Devices (HMD)
- Augmented VR (Virtual Reality)
- Desktop Devices

A portable system is used in this research that employs dVR extended by a touch screen and touch pad for advanced and intuitive interaction. This system serves as a compromise between the high-end systems such as CAVE that offer high fidelity and low-end systems such as desktop VR that do not offer much navigation and immersion functionalities. In an attempt to describe these different display techniques, the authors provide a brief overview to compare and contrast the pros and cons of the various methods. In a CAVE environment the system displays a virtual environment surrounding the user which places the individual into a virtual context. The use of
dVR framework for ET instruction has vast potential for the advancement of AL/PS (Active Learning/ Problem Solving) and PBL.

This research effort uses a combination of Open-Source programming languages/platforms to each Object-Oriented (OO) programming concepts to ET students in a relatively shorter time period with less complexity. In this research effort, geometry and graphics are employed to design and implement a multidimensional framework for learning programming concepts and applying them to solving sample ET problems. On the whole, this research effort will serve to strengthen Engineering Technology Programs by enhancing the programming skills of ET students, thus enabling them to succeed in their major area of study and also increase their employability.

Methodology

There are two major components in this interactive framework for programming instruction. One if the hardware component and the other one is the software component (Figure.1).

![Figure 1: Software and Hardware Components](image)

The hardware used in this research consists of the following major components (Figure.2)
- Alien ware 18 Base
Alienware is primarily a gaming laptop that employs Dual NVIDIA® graphics and has an 18.4 display. To enhance the interactive experience the research uses a Dell touchscreen monitor in combination with a wireless touch pad. One of the primary advantages offered by touchscreen monitors is that they greatly enhance interacting with digital information. This is especially true in the case of 3D visualization, simulation, and interactive applications as it eliminates or significantly minimizes the need for tedious interaction via mouse and keyboard.

Touch screen interfaces have been explored and studies by notable studies in the past in facilitating a natural interaction and usability studies point out the usefulness of such touchscreen interfaces in collaboration as well. Also, touchscreens are also regarded as an effective means of assistive technology. Some of the notable advantages of employing touch screen interfacing are as follows:

- User-friendliness
- Intuitiveness
- Speediness of task completion
- Economy of space

Also, it is very important to note that a touchscreen takes I/O to a significant level as it performs both as an input and output device. Typically, the input is provided with mouse and/or keyboard and the output is seen through the monitor. In case of the touch interface, the screen performs the dual role of input and output, as the screen is ‘touch-enabled’ thereby allowing user-input via touch. Different kind of hand gesticulations that are facilitated by Touchscreen include

- Tap
Basic Building Blocks: Concept Inventory

The foremost step involved in the software component development is the organization of the concept inventory. The basic programming concepts identified for this study include:
- Data Types
- Variables/Constants
- Scope
- Functions/Methods
- Conditionals
- Loops
- Recursion

After organizing a list of concepts that students need to learn to understand programming, the relations among the various programming elements, concepts, classes, methods, and libraries are described. Conceptual examples to elucidate the programming notions to students and design a plan to generate the 3D VR models corresponding to these conceptual plans. Next, a detailed description of the nodes that will be used to generate the aforementioned 3D VR models.

Development of Virtual Worlds to Demonstrate Programming Concepts

Virtual world is a 3D scene with objects created using geometric shapes. These objects are programmed using shapes with geometry and appearance. More complex objects are created by grouping and organizing shapes. Commonly, a hierarchical structure known as a scene tree is used to plan how complex objects can be built using simpler or primitive shapes. The virtual worlds in this study are created completely using the following Open Source and web-friendly programming technologies:
- VRML (Virtual Reality Modeling Language)
- X3D (Extensible 3d)
- Java
- JavaScript

Figure 3 and Figure 4 respectively show the 3D virtual representations that are used to form the scenes corresponding to the concept inventory and the actual coordinate systems in which they are built.
The purpose for this choice of tools and platforms (Open Source) is two-pronged:

- To facilitate online dissemination
- To keep the application development free of costly commercial software.

Programming languages (including graphics programming language) provide objects known as pre-built objects in the form of libraries. Many times for some specific applications these objects are not sufficient. In such a case, custom-built or user-defined shapes called prototypes or PROTOS in VRML/X3D are created. These objects represent ‘reusable software objects’ which are based on the notion of ‘build once and use as often’. These can be built once and accessed later using the EXTERNPROTOS (external prototypes).

A coordinate or coordinates is/are the numerical representation(s) of the location or position or something (Figure.3). Figure.3 shows spherical object created within this coordinate system. The spherical object is shown in two modes: the smooth one is known as a shaded version, wherein the lines are not seen. The other one is known as wire-frame where the lines making the shape and the surface can be seen. The part of the figure on the right (Figure.3) shows a rectangle in a Cartesian coordinate system with the following vertices: (3,2), (-3,2), (-3,-2), (-3,2). In this case, the center is at the origin (0,0). The area of the rectangle is 24 square
units. The origin is typically at the center of the reference coordinate system. A virtual scene can be viewed as being composed of elements or objects, each of which has its own properties or attributes. A parent object can include any number of children, which can be grouped or assembled to function as one single entity. This sort of hierarchical arrangement helps in the step-by-step design of the object and also understanding the framework at any later stage. A scene-tree construction is used in Virtual Scene Renderings. The root or the parent object consists of whole scene grouped together and all the other components are grouped under this parent object using ‘parent-child’ relationships.

**Figure 5 illustrates a 2-dimensional grid with a box on the left-hand side containing several text objects. These objects, or variables, can be dragged by the user into one of the four quadrants labeled “String,” “Int,” “Float,” and “Char.” If the user drags the variable into the quadrant representing its corresponding data type, the color of the text will change. This program is designed to teach the basics of data types using four fundamental types. As the user drags each individual element and watches them change color, they would begin to realize which kind of variable belongs in each quadrant and can associate the data-type label with that kind of information. The logic behind this program is rather simple. The interactive element of the user moving the text was accomplished by using what is known as a plane sensor. This allows the user to move any designated object around on a 2d plane, which is by default the x-y plane. A script then processes the x and y coordinates of each text element. If the coordinates are within the bounds of the quadrant of their respective data-type, the script will trigger the change of color which is the output back into the text-object. If the x and y coordinates are outside of the quadrants bound, the color outputted will return to white.**
In the program illustrated by Figure 6, the user is able to enter in text with their keyboard and display it on the screen. Based on the text entered, a label specifying the data-type of the input will appear, the color of the text will change, and an animation will appear in the background. This program expands upon the first task in teaching data-types but allows for more dynamic input on the part of the user. While the first task is limited to the variables provided, this file allows the user to enter in any text input available on their keyboard. To check which input is associated with which data, user can experiment with different input. Here key board input is the most essential part. This element was created using a VRML keyboard sensor. The input received from this sensor is then input into a text element as a string which displays on-screen whatever the user types. Whenever the string changes due to input, a script will run which scans the character code of each character in the string. If the characters all match up with a particular data-types requirements, the function will change the label to the name of that data type as well as change the background object and animation accordingly.

![Figure 6: Virtual Worlds for Illustration of Data Entry & Display](image)

Figure 7 illustrates the notion of conditionals using text objects, which abstractly symbolize the concept. At the top are two strings labeled “true” and “false” can be selected. By selecting one of these objects, the text “yes” or “no” will change color depending on the selection. A second program functions much like the previous function, only with different input. Along the top is displayed an actual programming conditional, or “if statement,” with a blank space to represent a missing piece of data. Underneath this are several numbers which the user can select to “fill” the blank space above. If the user selects the answer which fulfills the functions conditional statement (3), then the text “yes” will be highlighted in green. This program gives a basic idea of how conditional functions work. A data entry is input to the function and is then processed. If the statement entered appears to be true, then the following code will run. If the statement is false, then the program will move on and ignore the code inside of the conditional. In the second
program, the user is able to get a glimpse into the actual programming syntax and how it works rather than before which only showed a simple concept.

As the user clicks on one of the strings, a Boolean (“true” or “false”) variable (or and integer value in the second example) will change to the respective value. Then a conditional will run to determine the value of the variable and will change the text “yes” to green if true, or change the text “no” to red if false. An initialization script is run before all else to set the default color of the “yes” and “no” text to black as no value has yet been selected. Figure 7 shows the setup including the components such as gaming laptop, interactive touchscreen, touchpad, and multifunctional mouse.

Results and Discussion

Figure 7: Setup showing the Gaming laptop extended by the Touchscreen, Touchpad, and Multifunctional Mouse
Figure 8 summarizes the concepts as a canvas that offers an entry point to various programming notions discussed earlier. The top part shows the various concepts such as conditionals, loops, data types, etc.

Figure 8: Interface for Interacting with Virtual Worlds

Web forms are a concept which all computer users should be quite comfortable with and using these elements can greatly simplify how a user interacts with these programs (Figure 9). Users can now input a coordinate into an HTML form. Once the button is pressed, the 2D Processing window will display a colored text string of the data type. Similar to the original work, the 3D window (this time created using X3D), displays a 3D object of different shapes.

Figure 9: Web Forms for Coordinate Entry & VR Display

Proceedings of the 2015 Conference for Industry and Education Collaboration Copyright ©2015, American Society for Engineering Education
Besides the above fundamental programming concepts, other advanced notions such as event-driven programming were also covered. Below is an example of the interactive interface being used to teach basic programming steps that can be used to generate graphics concepts (Figure 10).

![Interactive Interface Example](image)

**Figure 10: Programming Interface to Demonstrate Bezier Functionality**

**Conclusion**

In this study, we implemented a dVR system that is capable of explaining fundamental programming concepts to students using dynamic and real-time interaction. This is a flexible VR system that is an optimal trade-off since the educational institutions that cannot afford expensive systems like CAVE can still obtain a satisfactory level of interaction. Having a low-cost portable system can be more convenient for developing and demonstrating small-scale applications involving programming appealing to the prospective clients than using the conventional displays or resorting to the high-end systems like CAVE. The framework demonstrated basic concepts of programming such as data types, variables, and constant using a dynamic VR framework and greatly assists PBL and active learning in Engineering and Technology curriculum, especially those involving programing components.

**Acknowledgements:** The authors would like to acknowledge the support from the Engineering Technology Division (ETD), ASEE (American Society Engineering Education) for their funding support that facilitated this study.

**References:**


