Programming Driven 3D Modeling on the Web

Andy Yeh
Queensland University of Technology
Australia
a.yeh@qut.edu.au

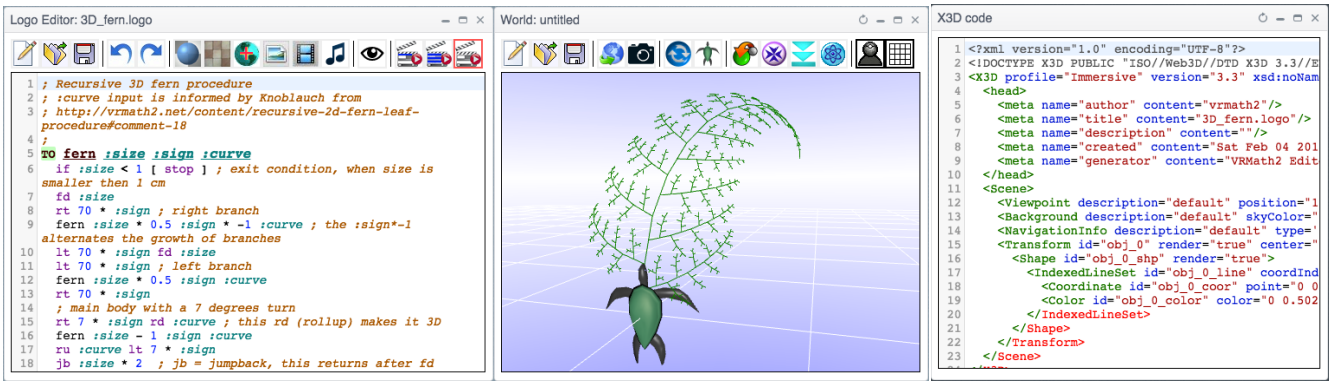

Figure 1: 3D fern leaf and X3D code generated online from Logo program.

ABSTRACT

This paper introduces an online 3D modeling environment named VRMath2 and discusses its applications. VRMath2 utilises a Logo programming language with a set of extended 3D primitives, to create 3D contents (in HTML5 format) in most modern web browsers. The 3D contents are rendered by X3DOM and VRMath2 can then export and publish the 3D contents in X3D format onto web pages. VRMath2 is an educational application and was originally designed for learning about 3D geometry. After the development in the last four years, its applications have now included science, technology, engineering and mathematics (STEM) education, largely due to its nature of programming driven 3D modeling.

CCS CONCEPTS

• **Social and professional topics** → Computational thinking; **Human-centered computing** → Interaction paradigms; • **Applied computing** → Education;

KEYWORDS

Logo programming language, X3D, HTML5, STEM education

ACM Reference format:

Andy Yeh. 2017. Programming Driven 3D Modeling on the Web. In *Proceedings of Web3D '17, Brisbane, QLD, Australia, June 2017*, 9 pages.

Web3D '17, June 05-07, 2017, Brisbane, QLD, Australia

DOI: http://dx.doi.org/10.1145/3055624.3075953

DOI: http://dx.doi.org/10.1145/3055624.3075953

1 INTRODUCTION

Beginning in 2001, the author was interested in human spatial abilities, and started designing a desktop VR application named VRMath[1] to investigate how different types of spatial ability can be developed as a result of using the designed VR application. Spatial ability is often linked with higher academic achievement and future success in mathematics, science and engineering related careers (Ivie and Embretson 2010; Pittalis and Christou 2010), and therefore it is a key area in educational research. Many researchers have tried to classify spatial abilities, and in general, three types of spatial abilities have been recognized: **spatial visualization**, **spatial orientation** and **spatial relations** (Lohman 1988). The computer 3D graphics (i.e. desktop VR in VRMath2) has the potential to develop all three types of spatial ability. Further from educational point of view, language is always at the core of learning. In searching for a language that is natural in computer 3D environment and friendly to learners, the Logo turtle graphics was chosen. The Logo turtle graphics has a set of egocentric movements such as *forward*, *back*, *left* and *right*, as well as a set of universal movements using coordinate system, such as *setx* and *sety*. These language or commands are suitable for developing spatial ability, in accordance with its developmental sequence (Darken 1996). However, traditional Logo turtle graphics operates on two-dimensional graphics. In order to better investigate and develop 3D spatial ability, a set of extended 3D primitives is added into the VRMath2's Logo programming language. These will be further explained in later sections.

In addition to the above educational points of view, affordability, accessibility and sustainability are among the main

[1] VRMath was renamed as VRMath2 in 2009. See https://vrmath2.net/about.

considerations for designing this 3D modeling environment. As a result, the Web has been chosen as the platform, and subsequently, advantages and limitations applied to this online modeling environment. VRMath2 is free and easy to access online. It incorporates current HTML5 and X3D standards, which are expected to sustain into the future years. Both the Logo interpreter (i.e., jslogo[2]) and the X3D rendered by X3DOM[3] are implemented in JavaScript. This has the advantage of easy access online in web browsers. However, the current state of JavaScript has limitations such as multi-threading and communication with computer hardware. Nevertheless, the current implementation of the Logo interpreter is powerful enough to offer a unique alternative for 3D content creation online in modern web browsers.

From educational stance, coding or programming has great value in the process of learning and problem solving. However, using programming to create 3D contents is not necessarily the best way for all kind of 3D contents and purposes. The 3D contents could include static 3D solids to dynamic 3D simulations and large-scale virtual worlds. This paper will highlight both the strength and weakness about the programming driven 3D modeling in the VRMath2 application.

2 RELATED WORK

The idea of using programming to generate 3D graphical contents is not new. In fact, many mathematical (e.g., MATLAB) and engineering (e.g., various CAD/CAM) software offer their own scripting and/or specific programming language to construct 3D objects. Generally, programming interface complements interactive interface (i.e. direct manipulation of 3D objects), and can offer precise calculation for mathematical visualization and scientific simulation. These works are all inspiring to the development of VRMath2. However, since the purpose of VRMath2 is to facilitate learning (including developing spatial abilities) and enable accessible 3D contents on the Web, the focus here is on reviewing 3D environments that are either using turtle graphics and/or using X3D on the Web.

2.1 Turtle Graphics

One of the main reasons for choosing turtle graphics or turtle geometry is the aforementioned egocentric movement such as *forward*, *back*, *left* and *right*, which are integrated as part of the Logo programming language. Egocentric movement is an intuitive way of moving around in space as it constantly refers to the mover's own direction and/or location. The 'turtle' in turtle graphics is the mover, which serves as a reference point for geometric location and direction. Therefore, a 100 by 100 pixel square can be created with the following program.

```
repeat 4 [ forward 100 right 90 ]
```

As a functional programming language, 2D regular polygons such as a heptagon can be easily created with the Logo program below.

```
TO polygon :side :length
    repeat :side [ forward :length right 360/:side ]
END
polygon 7 100
```

Using the egocentric movement, the creation of geometry does not need to concern about the coordinates. The coordinates are calculated by the movements and can be collected in Logo program if desired. It is considered that the turtle geometry is highly relevant to the development of spatial abilities.

Logo programming language was created in 1967 and over the last five decades, there have been many implementations and dialects of this programming language. However, the majority of these implementations are using 2D graphics. Of the few 3D implementations, Elica Logo[4] has a more natural 3D turtle graphics. However, unlike VR 3D, its 3D graphics is non-interactive. For example, In Elica Logo, users cannot change viewpoint by direct mouse dragging but can program a camera turtle to change and/or animate the 3D viewpoint. Most noticeably, it has extended the traditional Logo to include a set of 3D movements and rotations (see Table 1 for example).

Table 1: 3D Movement Commands in Elica Logo

Command	Explanation of movement
forward (fd)	Moves turtle forward
backward (bk)	Moves turtle backward
rise	Moves turtle up
lower	Moves turtle down
stepleft	Moves turtle to the left
stepright	Moves turtle to the right
left (lt)	Turns turtle to the left
right (rt)	Turns turtle to the right
uppitch (up)	Turns turtle up
downpitch (dn)	Turns turtle down
leftroll (lr)	Rolls turtle to the left
rightroll (rr)	Rolls turtle to the right

These 3D movements integrated in Logo programming cover a wide range of spatial awareness and thus have great potential for developing spatial abilities. This Elica Logo 3D environment, however, is limited in terms of 3D interactivity (e.g., no free navigation in 3D virtual space) and system dependency (i.e., a standalone application for Microsoft Windows system).

2.2 X3D on the Web

X3D and HTML5 are both current international standards of Web technology, and they are looking to sustain into future years of online applications and services. There are a few online

[2] The Logo programming language adopted and extended in VRMath2 is jslogo (Logo in JavaScript), which is available at https://github.com/inexorabletash/jslogo.
[3] X3DOM is a JavaScript library, which integrates X3D into HTML5. It is available at https://www.x3dom.org/.
[4] Elica Logo: http://elica.net

3D applications, in which 3D models can be built with direct interaction (e.g., direct dragging to manipulate 3D models) and/or via programming or scripting. Direct interaction is a friendly way for 3D modeling. However, in order to include the educational value of programming, the author is particularly interested in the later, and OpenJSCAD[5] is such an online application that uses the programmer's approach for 3D model development.

OpenJSCAD is an object-oriented programming language that has JavaScript-like syntax and is able to use JavaScript programming concepts and libraries. This is powerful because programmers can then design some HTML form (e.g., menu, button and checkbox etc.) for some interactive event control or dynamic generation of 3D models (see Fig 2.).

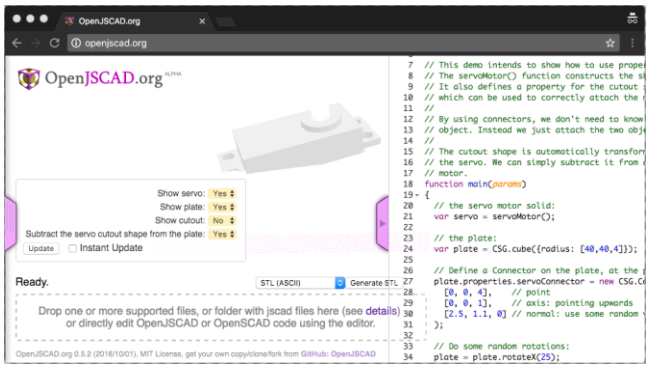

Figure 2: OpenJSCAD with example program, 3D model and interactive web widgets.

OpenJSCAD is tuned to develop 3D models for 3D printing. It supports a range of file format such as STL for the purpose of 3D printing. It also exports its 3D models to X3D format using the IndexedFaceSet element or node of X3D. This is enough to record the 3D solids but the majority of the X3D features (e.g., animation, interaction) are not utilised.

This programmer's approach in OpenJSCAD is inspiring and is powerful to achieve its aim of developing precise 3D models for 3D printing. However, like most professional applications, the 3D geometry is mainly operating on formal 3D coordinate system. It would suit for users with the same purpose of creating CAD solids for 3D printing, but not necessary for developing spatial ability as users would need to have developed some higher level of spatial sense and programming skills already to use this programming environment for 3D modeling.

3 VRMATH2 ENVIRONMENT

VRMath2 started its development in 2001, and over the years, it has evolved to include social aspects of learning. The "2" in VRMath2 stands for the Web 2.0 or the rewrite Web, so learners are able to design, create and share (discuss) online their creations of 3D virtual world. The main purpose of this paper is to introduce the programmatic approach in VRMath2 (specifically to its 3D modeling environment called VRMath2 Editor at https://vrmath2.net/VRM2, see Fig. 3.), and discuss its applications. In this section, the focus is on two of its key components: Logo programming and VR 3D space.

3.1 Logo programming in VRMath2

The Logo programming language in VRMath2 is derived from jslogo, which is a JavaScript implementation of Berkeley Logo[6]. The core components of this Logo programming such as data type, syntax and control structure etc. are preserved, but its 2D turtle graphics has been extended to 3D graphics in VRMath2. As a result, many Logo primitives (commands) are modified and/or added/removed in VRMath2's Logo language.

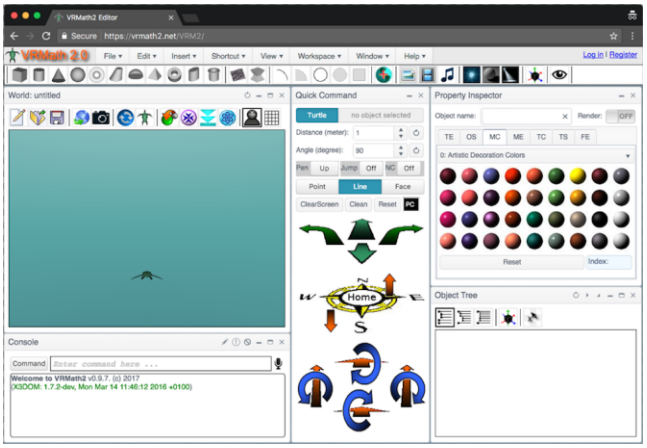

Figure 3: VRMath2 Editor.

Moving from 2D finite space (i.e., a rectangular area on screen) to 3D space (i.e., virtually infinite VR 3D space), there are some key conceptual differences to reflect on the Logo language. Firstly, there is no need for the window management commands such as *wrap* (the turtle will continue on the other side if it moves off the edge of the screen), and *fence* (the turtle will stop at the edge if it tries to move past the edge of the screen). Secondly, the distance to *forward* or *back* will change from the number of pixels to meters in 3D virtual space. Lastly and the most important change is the inclusion of potential 3D movements and their naming.

3D movements can be classified into three frames of reference (FoR): **egocentric**, **fixed** and **universal** (Darken 1996). Egocentric FoR based movements are moving according to the mover's own direction and location. Fixed FoR based movements are moving towards landmark or cardinal/compass points. Universal FoR based movements are moving according to a globally defined coordinate system. Table 2 shows some key Logo commands associated with these three FoR in VRMath2.

[5] OpenJSCAD: http://openjscad.org

[6] Berkeley Logo or UCB Logo was developed by Brian Harvey. The latest version 6.0 is available at https://people.eecs.berkeley.edu/~bh/.

Table 2: 3D Movement Commands in VRMath2 Logo

Command	Explanation	Change
Egocentric FoR		
forward (fd)	Moves turtle forward	Location
back (bk)	Moves turtle backward	Location
left (lt)	Turns turtle to its left	Direction
right (rt)	Turns turtle to its right	Direction
rollup (ru)	Pitch turtle head up	Direction
rolldown (rd)	Pitch turtle head down	Direction
tiltleft (tl)	Tilt left on forward axis	Direction
tiltright (tr)	Tilt right on forward axis	Direction
Fixed FoR		
up	Slides turtle up	Location
down (dn)	Slides turtle down	Location
east	Slides turtle east	Location
west	Slides turtle west	Location
north	Slides turtle north	Location
south	Slides turtle south	Location
Universal FoR		
setxyz	Moves turtle to (x,y,z)	Location

Using these 3D movement commands, the programmer can move the turtle to create points (using PointSet node in X3D), lines (IndexedLineSet), and faces (IndexedFaceSet) in 3D space. The traditional Logo's color and fill commands have become a series of color (Color), material (Material) and texture (ImageTexture) commands.

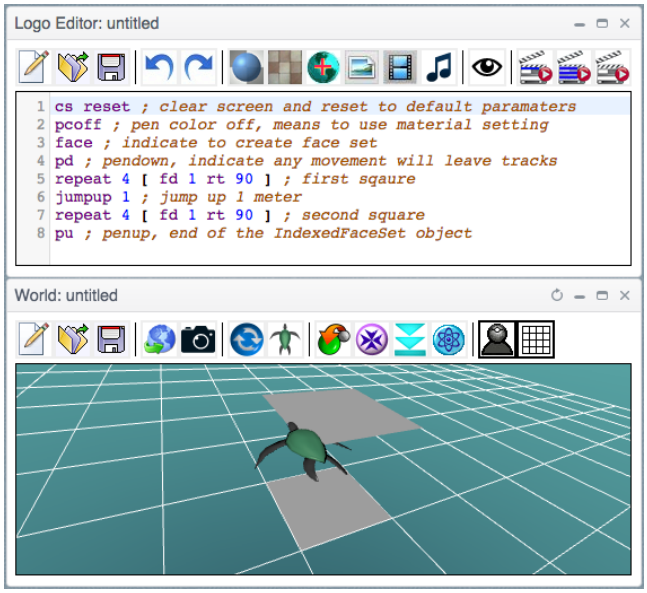

Figure 4: Two separate faces in one IndexedFaceSet object.

In addition to the movement commands in Table 2, a series of *jump* commands are implemented. Most of the movement commands have their associated jump commands. For example, there is a *jumpforward* (*jf*) command for *forward* command, and *jumpeast* (*je*) for east command and so forth. The purpose of jump series of commands is to temporarily disable turtle tracks during the creation of PointSet, IndexedLineSet or IndexedFaceSet object. For example, Fig. 4 shows two faces in one IndexedFaceSet node created by a Logo program in VRMath2.

Because X3D is used to represent the 3D objects/worlds, the Logo language in VRMath2 thus includes a set of 2D and 3D primitive objects as defined in X3D. These include geometric objects (e.g., *arc*, *pie*, *circle*, *box*, *cylinder*, *sphere*, *cone*, *torus*, *dish*, *elevationgrid*, *extrusion* etc.), lighting objects (e.g., *spotlight*, *directionlight*, *pointlight*), and objects for animation (e.g., *timesensor*, interpolators and *route*). Every geometry object when created is enclosed within a Transform node. Another set of scaling commands (e.g., *setscale* x y z) can provide scale information for the Transform node/object, while its translation and rotation fields can be derived from the turtle's location and direction. With X3D, the 3D turtle graphics in VRMath2 records individual object and are able to manipulate objects programmatically through document object model (DOM) in HTML5. Examples can be seen in later section about applications of VRMath2.

3.2 VR 3D Space

The virtual reality 3D space in VRMath2 is achieved by the X3DOM library, which renders the X3D (in HTML5 format) as generated from the Logo programs. In comparison with other 2D and 3D turtle graphics, this VR 3D space in VRMath2 has better interactivity to facilitate the development of the three types of spatial ability (Lohman 1988). These three types of spatial ability are defined as below:

Spatial visualization: the ability to mentally rotate, manipulate, and twist two- and three-dimensional stimulus objects.

Spatial orientation: the comprehension of the arrangement of elements within a visual stimulus pattern; the aptitude for remaining unconfused by the changing orientations in which a figure may be presented; the ability to determine spatial relation with respect to one's body.

Spatial relations: the ability to mentally transform (e.g., translate or rotate) objects with respect to an environmental frame of reference (e.g., a landmark or cardinal points) while one's egocentric reference frame does not change.

In VRMath2, **spatial visualization** can be developed or put in action when commanding the turtle to move or turn, or when experiencing the animation of objects. In either case, the programmer/learner is mentally manipulating objects and can receive real-time visual feedback. **Spatial orientation** can be developed or put in action when navigating in the 3D virtual space and examining the virtual objects/worlds. In this situation, the programmer/learner is aware that his/her own orientation is changing in the virtual space, and thus the view of virtual objects. Lastly, the **spatial relations** can be developed or put in action when the programmer/learner is designing a world with arrangement of many geometric and environmental objects. The

use of fixed FoR movements in VRMath2's Logo language is also related to the development of spatial relations.

The VR 3D space (i.e., 3D turtle graphics of VRMath2) is interactive in terms of its free navigation. However, with respect to creation of 3D objects/worlds, it is semi-interactive. By this it means that using programming to construct 3D contents is non-interactive because there is no direct dragging to scale or move objects in the 3D virtual space. However, the VRMath2 Editor has a few graphical user interfaces such as the Property Inspector and Object Tree (see Fig. 3) that enable some interactions to assist the creation of 3D contents.

Both the Logo programs and virtual worlds created in VRMath2 Editor can be saved online and re-opened for editing (or remixing if shared) or presenting in the VRMath2 community site. The 3D world window also has a publish function to export the 3D contents as X3D files (see Fig. 1). The X3D files can be inserted into other virtual worlds with *world* or *inline* command at the turtle's position and direction. This could reduce the size of X3D and Logo files, and assist with the replication or reuse of objects/worlds in the online 3D contents.

4 APPLICATIONS

As stated earlier, this VRMath2 environment was originally designed as an educational tool for developing spatial ability. Spatial ability is also mostly associated with geometry and mathematics. However, over the past four years, it has been identified that this VRMath2 environment can have applications in areas other than mathematics education. More specifically, the programming driven 3D modeling in VRMath2 can facilitate learning in STEM education (Yeh and Chandra 2015).

4.1 In Mathematics Education

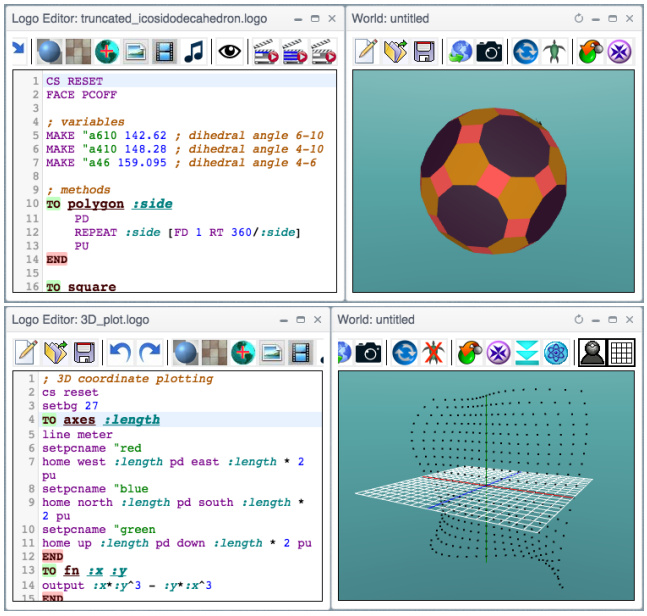

Figure 5: Applications in mathematics education.

Logo's turtle geometry is an intuitive language for geometry and measurement. The 3D extension of Logo in VRMath2 further enabled a wide range of mathematical activities. These activities range from constructing basic 2D and 3D shapes, plotting and visualizing 3D functions to fractals with recursive procedures (see Fig. 5 and Fig. 1).

The 3D movements in Table 2 allow young children multiple ways to construct 3D shapes such as a frame of a cube (Yeh 2013). The non-commutative nature of 3D rotations can also be explored in this VRMath2 environment (Yeh 2004). Through programming in the 3D turtle geometry, learners are engaged in mathematical thinking and reasoning about location, direction and movement, and thus develop further their spatial abilities.

4.2 In Science Education

The Logo language in VRMath2 includes environmental objects (e.g., background and lighting) and animation feature of X3D. This enables applications in scientific visualization and simulation. Fig. 6 shows a solar system simulation for observing day and night, and seasons, and a sundial simulation for observing the variation of shadows in a day.

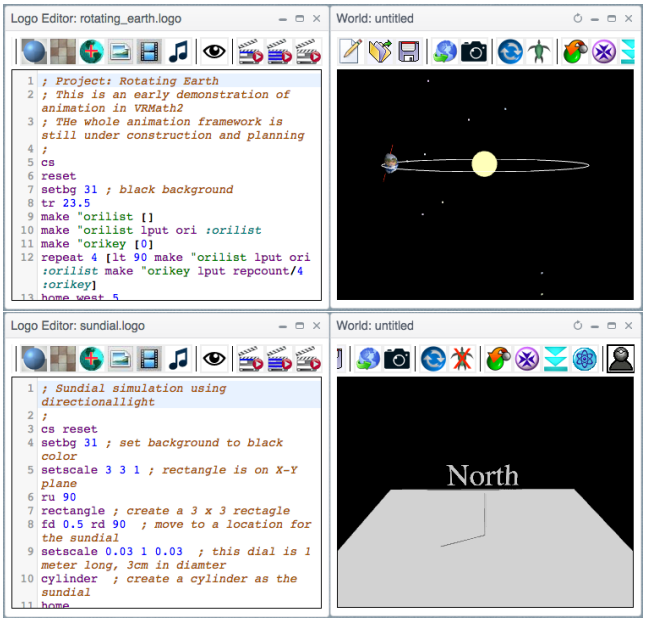

Figure 6: Solar system and sundial simulations.

In a recent project, a group of 84 Year 9 students have also utilized VRMath2 to model atoms and molecules. Through 3D modeling activity, they investigated the structures, characteristics and theories about atoms and molecules. This atomic modeling project was designed using a STEM integrated approach. Not only they learnt about atomic science, students also commented on their learning in mathematics, particularly in geometry about location and angle to place atoms. For example, some students are able to discover the tetrahedral angle when modeling a methane molecule. Other students also commented a

steep learning curve, as they have not had any programming experience before. There is a mixture of frustration and joy about programming in using VRMath2's Logo language, but most students are able to collaborate and program animations of their atoms as shown in Fig. 7.

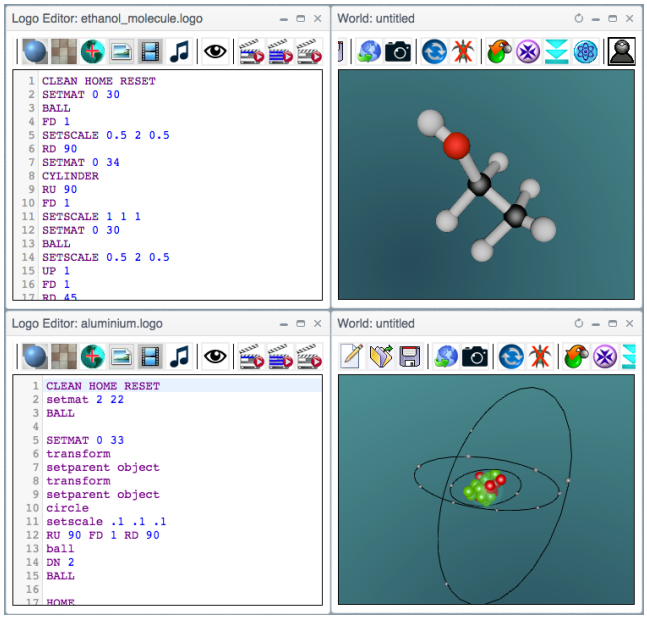

Figure 7: Ethanol molecule and aluminum atom models.

4.3 In Technology Education

The current Australian National Curriculum: Technologies promotes systems thinking, design thinking and computational thinking (ACARA 2017). Projects such as furniture design using VRMath2 could encourage students in these types of thinking. Fig. 8 depicts the use of Logo programming to produce a coffee table, a study desk and a customizable clock.

The design process involved analysis of the product both as a whole system and individual components. The Logo programs show the parts of furniture are systematically dealt with individually then put together as a whole. The design thinking involved using the 3D movements to navigate the way for the turtle to construct the virtual products. The Logo programming in VRMath2 also enables the design of interactive 3D objects. As can be seen in the analog clock example, users are able to click on different border sizes and colors to customize their clock. This reflects a need analysis in the design thinking (ACARA 2017). The programming and virtual products reflect significant computational thinking (ACARA 2017), in which data are quantified and digital solutions (i.e., the programming codes and the virtual products) are created. Kafai and Burke (2013) argued that computational thinking, while often associated with computer science, is the extension of computer science principles applied across disciplines including mathematics and science, and also to the humanities in fields such as journalism and literature. This supports the idea that the VRMath2 environment can be applied in STEM education because of the computational thinking promoted in the use programming.

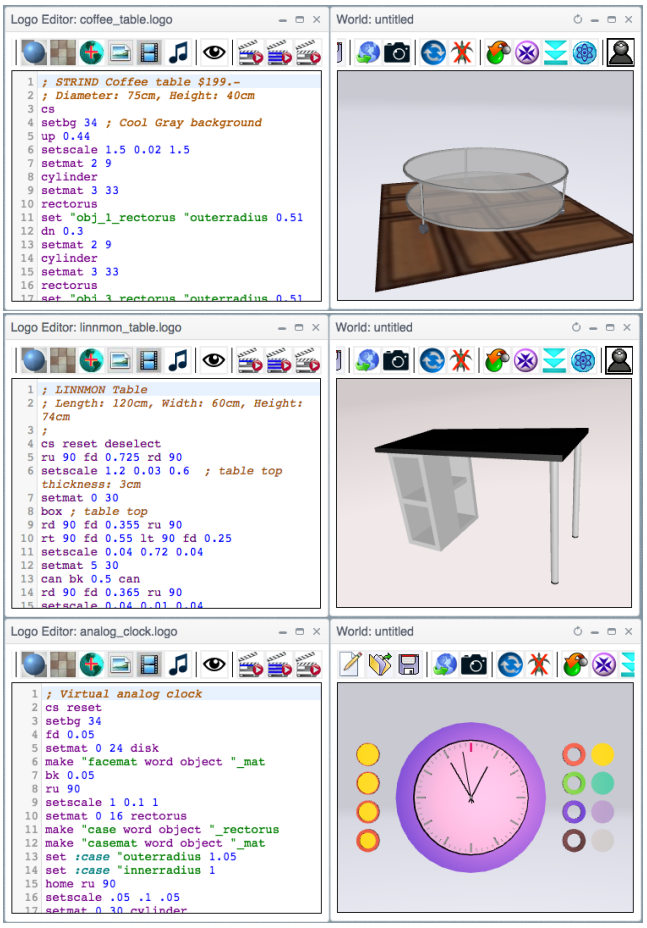

Figure 8: Design and visualization of products.

4.4 In Engineering Education

Engineering education has become more prominent in school education in recent years. It is seen as applications of mathematics and science, involving iterative design/engineering processes to define problems (e.g., criteria and constraints), generate and evaluate solutions, and optimize solutions through systematic testing and refining (English 2016). Fig. 9 demonstrates the engineering processes when programming and designing gears.

The engineering processes of gears involve the study of mathematics and science about involute gears. For example, the pressure angle and diametral pitch etc. are required in order to program precise meshing gears. There are cyclic testing, debugging, and evaluating during the programming, and eventually the spur gears are completed with IndexedFaceSet and the helical gears are completed using Extrusion (a X3D node) for faster creation and performance in this gear project. The use of Extrusion seems to be a more optimized solution but

both uses of IndexedFaceSet and Extrusion are valuable experiences in terms of learning and education.

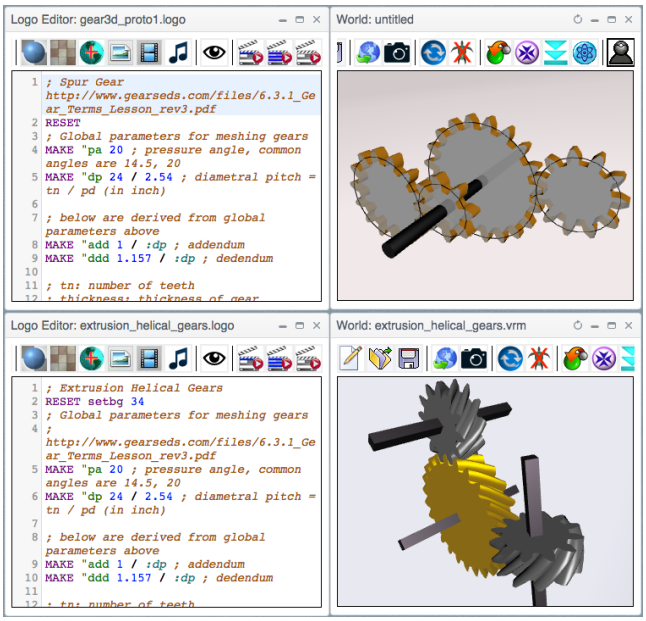

Figure 9: Engineering spur and helical gears.

To sum up for the applications, it should be noted that although they are reported in individual STEM area, each application is in fact problem-based or project-based learning that involves more than just one discipline in STEM. This is attributed to the nature of the Logo programming language that is used to create the 3D contents in the VRMath2 environment.

5 DISCUSSIONS

The use of computer programming to generate 3D models or virtual worlds is neither a new idea nor the best way to create 3D contents. However, in the case of VRMath2, the use of Logo programming language to create 3D virtual worlds on the Web is the first of its kind. Nevertheless, it has its strengths and weaknesses in different contexts. The Logo programming language adapted in the VRMath2 environment is still developing, in the directions of aiming to better facilitate learning and generating more sophisticated online 3D virtual worlds.

5.1 Strength

X3D is an international standard for 3D on the Web. It is powerful to represent and describe 3D contents with its current specification, and more importantly it is still evolving to further 3D representations. However, it is unusual to code directly in X3D for complex objects such as a mesh or multi-faces objects that have large sets of 3D coordinates for triangles (e.g., IndexedTriangleSet) and faces (e.g., IndexedFaceSet). Using programming such as in Logo language can generate these coordinates randomly and/or through certain mathematical patterns and functions. This is particularly useful for mathematical and scientific visualizations and simulations. For example, using recursive function calls, which include the control structure of programming (e.g., *if else)*, can generate coordinates for fractals such as the 3D fern leaf in Fig. 1.

Another strength of this programming driven 3D modeling is, perhaps debatable, attributed to the egocentric and fixed 3D movements in the Logo language. In the sundial simulation example (see Fig. 6), a directional light is tilted 23.5 degrees to simulate the earth's tilt on its spinning axis, and the sun passes the northern part of the sky as viewed on southern hemisphere. There are ways to create such virtual world in different modeling software, but in the Logo programming, a simple *tiltright 23.5* can achieve this modeling with precision.

As a high-level programming language, Logo programs are easily readable and recorded in plain texts. This makes it very easy to repeat (i.e., loop structure) for patterns and reuse the programming codes. Using loop structure, this Logo language in VRMath2 can generate large-scale virtual worlds or data sets. 3D contents can be recreated by running the Logo programs and/or remix and share with others. Data such as coordinates generated can be stored in variables and used later for animation or objects such as IndexedFaceSet and Extrusion. Fig. 10 shows a Logo program uses loop structure (i.e., *for [:var start end step]*) to generate coordinate data through a mathematical function, then applies the coordinate data to animate an ElevationGrid. This is done in 34 lines of codes, and the Logo program can be easily modified with different grid sizes, heights and of course the mathematical functions for different animations.

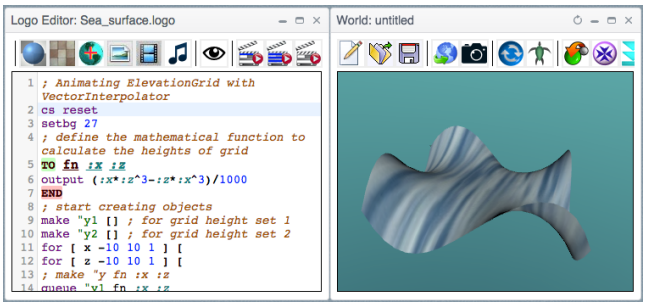

Figure 10: Sea surface animation simulation.

The current implementation of VRMath2's Logo language also includes a set of animation primitives that utilizes the interpolators of X3D. Simplified commands such as *spin*, *travel*, and *tour* can interpolate rotation, translation and viewpoints respectively. There is also a developing prototype of behavior model in this Logo that can capture input events (e.g., click, mouseover and mouseout) to manipulate 3D objects. Fig. 11 shows a spinning Box, which also acts as a switch (when clicked) to toggle a DirectionalLight. This strength of programming animation and behavior is attributed to the close meshing between the Logo language, HTML5 (including the DOM and JavaScript) and the features of X3D in VRMath2.

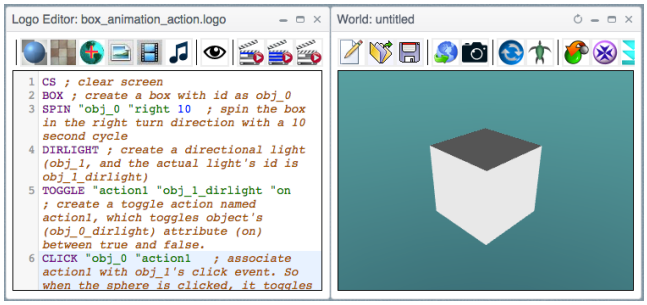

Figure 11: A spinning box that controls a directional light.

5.2 Weakness and limitation

Despite the programming language can be powerful to create 3D virtual worlds, the immediate weakness is on this dialect of Logo language. Unlike direct interactive modeling software with user-friendly graphical interface, this text-based Logo programming requires considerable time to master. Users who have prior programming experience may find Logo programming language easy to code. However, users who are new to programming may find it difficult except for the turtle geometry component of the Logo. As an educational tool, this could be the biggest hurdle for students to enjoy learning in the VRMath2 environment.

The text-based programming paradigm of Logo (as well as Basic and Pascal) in schools has suffered disinclination since mid 1990s. One of the key reasons for the diminishing of text-based programming is the interactive multimedia and visual programming paradigm such as Scratch[7] emerged (Kafai and Burke 2013). Fig. 12 shows an example program, which creates a square in Scratch. The visual programming can be more appealing to beginning programmers. However, the types and effectiveness of learning from both text-based and visual programming require more empirical studies to validate.

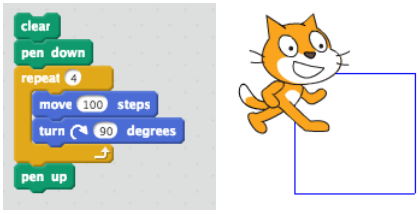

Figure 12: Visual programming of a square in Scratch.

Other weaknesses and limitations can be derived from the Web platform that this Logo programming of VRMath2 adopts. On the surface, it is cross-platform and easy to access with just Internet connection and a web browser. However, this also means that offline editing is not possible. In some cases, without offline editing capability presents some equity issues in education. VRMath2 is also dependent on X3DOM for the 3D interface. And as a web application, its performance and functionalities in programming 3D contents are limited by the JavaScript and the implementation of X3DOM. Examples of these limitations are provided in the next section.

5.3 Future direction

In order to encourage more participation to this VRMath2 application, the refining of the programming interface for 3D content creation is one the main tasks that this author is constantly aiming to improve. At this stage however, the programming paradigm in VRMath2 will still be kept as text-based, but it will be assisted with more graphical widgets and new technologies. For example, there are three developing prototypes of component in VRMtah2.

Speech recognition interface. At this stage, the speech recognition can recognize most of the 3D movement commands, and execute to create simple geometry (see Fig. 13). This is only a simple direct mapping of spoken words to the programming language. In the future, it will aim at analyzing more natural speech and translate to not just a command, but a set of codes to achieve the 3D modeling required by the user. Currently, this speech recognition prototype only works in Google Chrome browser because the Web Speech API[8] is still a W3C draft and only the Google Chrome browser has implemented this API.

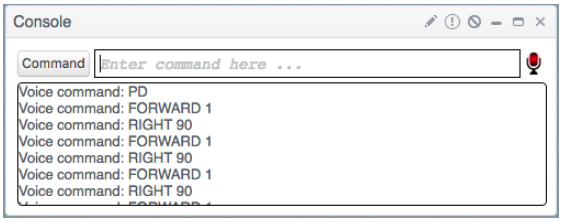

Figure 13: Speech recognition interface.

Mobile sensor interface. This working prototype[9] utilizes the gyro sensor in mobile devices such as an iPhone to transmit 3D rotation data for the turtle. In doing so, the user can hold and rotate a mobile device to control the direction or rotation of the turtle during construction of 3D objects or collection of data such as location and direction. Collected data can be stored in variables for creating animation and/or 3D objects later in Logo programs. This prototype utilizes Node.js [10] to setup a communication server, which essentially is the concept of a chat room that broadcast a transmitter's (e.g., an mobile phone with gyro sensor) 3D rotation data to all receivers (e.g., computers running VRMath2 Editor). The transmitter device needs to have a gyro sensor and a browser that supports the DeviceOrientation

[7] Scratch is a visual programming language developed by MIT. See https://scratch.mit.edu/

[8] The Web Speech API is a draft Web technology at https://dvcs.w3.org/hg/speech-api/raw-file/tip/webspeechapi.html

[9] For example, see https://vrmath2.net/content/how-control-movement-turtle-using-mobile-phone

[10] Node.js is a server side JavaScript runtime. See https://nodejs.org/en/

Event API[11]. Unfortunately, the support and implementation of this API are inconsistent and vary in different mobile browsers.

Cardboard VR application. This is not directly related to the programming of 3D contents, but a viewing application for the 3D contents. Currently, this prototype [12] allows all conforming X3D files in the VRMath2 website to be viewed on a mobile phone as stereoscopic 3D for cardboard VR experience (see Fig. 14). This application may encourage users to create and rethink their design of 3D virtual world, before they export and publish their 3D contents in VRMath2 website. For example, the designer could include multiple viewpoints and behaviors (e.g., mouse click) so the viewers could interact with some virtual objects through this Cardboard VR application.

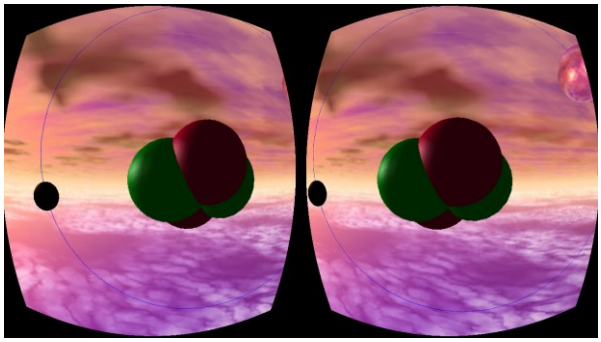

Figure 14: Stereoscopic VR of helium atom from VRMath2.

Currently, this Cardboard VR application in VRMath2 utilised the RenderedTexture and ComposedShader nodes as implemented in X3DOM to achieve the stereoscopic VR effect. The X3DOM Runtime API pictRect[13] is used to help implement a mouse click or touch event. The utilization of mobile magnetometer sensor through JavaScript has very limited support so the use of magnets to simulate click or touch event cannot be achieved in this current prototype. In the future, the WebVR API[14] may bring more possibilities for this Cardboard VR application in VRMath2.

6 CONCLUSION

This paper introduces the use of Logo programming to create 3D virtual worlds online in the VRMath2 application. Using a programming language to generate 3D contents can have high precision for mathematical and scientific visualization and simulation. The programming codes can be easily modified, shared and reused to generate and handle large sets of data, which can be used for purposes such as complex objects and animations. The Logo programming in VRMath2 has a range of 3D movements (i.e., egocentric, fixed and universal) that are designed for developing spatial abilities, and offering a unique and convenient way of constructing 3D virtual worlds in X3D format on the Web. The process of programming for 3D contents can foster computational thinking and skills for problem solving. With this programming driven 3D modeling approach, VRMath2 has the potential to facilitate learning in STEM education. There are limitations due to Web technologies, but perhaps the biggest limitation is one's imagination about how the VRMath2 environment and programming approach can be utilized for education and sophisticated 3D content creation.

[11] The DeviceOrientation Event API is a draft Web technology at https://w3c.github.io/deviceorientation/spec-source-orientation.html

[12] For example, see https://vrmath2.net/content/vrbox-world

[13] See https://doc.x3dom.org/author/runtime.html#pickRect

[14] See https://w3c.github.io/webvr/

REFERENCES

Australian Curriculum, Assessment and Reporting Authority (ACARA). 2017. F-10 Curricuum: Technologies – Key ideas, v8.3, (February, 2017), http://www.australiancurriculum.edu.au/technologies/key-ideas.

R. P. Darken. 1996. Wayfinding in large-scale virtual worlds. Unpublished Doctor of Science thesis, George Washington University, Washington, DC.

Lyn D. English. 2016. STEM education K-12: perspectives on integration. *International Journal of STEM Education*, 3:3 DOI 10.1186/s40594-016-0036-1

Jennifer L. Ivie and Susan E. Embretson. 2010. Cognitive process modeling of spatial ability: The assembling objects task. *Intelligence*, 38(3), 324-335.

Yasmin B. Kafai and Quinn Burke. 2013. Computer programming goes back to school. *The Phi Delta Kappan*, 95(1), 61-65.

D. Lohman. 1988. Spatial abilities as traits, processes and knowledge. In R. J. Sternberg (Ed.), *Advances in the psychology of human intelligence*, Vol. 40. Hillsdale, LEA, 181-248.

Marios Pittalis and Constantinos Christou. 2010. Types of reasoning in 3D geometry thinking and their relation with spatial ability. *Educational Studies in Mathematics*, 75(2), 191-212.

Andy Yeh. 2013. Constructing a frame of cube: connecting 3D shapes with direction, location and movement. In Steinle, V., Ball, L., & Bardini, C. (Eds.) *Mathematics Education : Yesterday, Today and Tomorrow* (Proceedings of the 36th Annual Conference of the Mathematics Education Research Group of Australasia), pp. 690-697. Melbourne, VIC: Mathematics Education Research Group of Australasia Inc.

Andy Yeh. 2004. Two Turns Must Take Turns: Primary School Students' Cognition about 3D Rotation in a Virtual Reality Learning Environment. In Yang, Wei-Chi, Chu, Sung-Chi, De Alwis, Tilak, & Ang, Keng-Cheng (Eds.) Asian Technology Conference in Mathematics (ATCM), 13-17 December, 2004, Singapore.

Andy Yeh and Vinesh Chandra. 2015. Mathematics, Programming and STEM. In M. Marshman, V. Geiger, & A. Bennison (Eds.). *Mathematics education in the margins* (Proceedings of the 38th annual conference of the Mathematics Education Research Group of Australasia), pp. 659–666. Sunshine Coast: Mathematics Education Research Group of Australasia Inc.