UNIVERSITY OF CAPE TOWN

DEPARTMENT OF COMPUTER SCIENCE

CS/IT Honours Final Paper 2020

Title: Modelling for 3D Printing as a Tool for Increasing Student Engagement in Introductory Programming Courses

Author: Maria Nanabhai

Project Abbreviation: CODE3D

Supervisor(s): Gary Stewart

Category	*Min*	*Max*	Chosen
Requirement Analysis and Design	*0*	*20*	0
Theoretical Analysis	*0*	*25*	0
Experiment Design and Execution	*0*	*20*	20
System Development and Implementation	*0*	*20*	5
Results, Findings and Conclusions	*10*	*20*	20
Aim Formulation and Background Work	*10*	*15*	15
Quality of Paper Writing and Presentation	*10*		10
Quality of Deliverables	*10*		10
Overall General Project Evaluation (*this section allowed only with motivation letter from supervisor*)	*0*	*10*	0
Total marks		**80**	

Designing Models for 3D Printing as a Tool for Increasing Student Engagement in Introductory Programming Courses

Maria Nanabhai
Department of Computer Science
University of Cape Town
Cape Town, South Africa
nnbmar003@cs.uct.ac.za

ABSTRACT

Computer programming is an increasingly in-demand skill in modern society, but it is considered difficult to learn and students often become disinterested in pursuing it. Engagement in the subject is a counter to this, affecting performance, understanding, and retention. Many interventions to improve student motivation have been investigated. In this study, we examined one such intervention: using programmatic modelling for 3D printing as a tool for teaching. We first developed a web-application, Model3D, for programmatic graphical modelling in JavaScript. Thereafter, we evaluated this tool among educators for usability and suitability for improving student motivation in an educational context.

Evaluation was done on 5 computer science educators at the University of Cape Town using a mixed methods approach, including surveys for assessing usability and motivation, as well as semi-structured interviews. The results show that Model3D is a user-friendly tool that educators would consider useful to improve student motivation in their classes ($p<0.001$). This provides evidence for an effective intervention to the problem of student motivation in computer science education. Recommended interface improvements and applications are discussed.

CCS CONCEPTS

• **Social and professional topics** → **CS1**; • **Physical computing** → *Digital fabrication*; • **Human-centered computing** → User interface programming.

1 INTRODUCTION

Information and communications technologies (ICT) have become ubiquitous in modern society, playing increasingly important roles in the economy as well as across all disciplines. However, the growth of individuals with computing skills has not kept up to pace with the growth of technology, creating a large skills gap, especially in developing countries such as South Africa. Educational institutions and governments thus consider it a high priority to promote studying ICT, and more especially computer programming, both by making it more appealing to learners to improve enrolment rates, and by improving success rates [4].

This is a challenge given that programming is considered a difficult skill for novices to learn, which intimidates many students [23]. The computer science (CS) major has some of the highest drop-out rates across all disciplines – it is estimated that fewer than 40% of students enrolling for a computer science major persist to the completion of their degree [26]. The approaches presently used in courses do not effectively counter this – a study conducted at the University of Witwatersrand found that many students become less positive about working with computers after completing an introductory CS course, a finding in line with other research [17].

Several interventions have been proposed to address this challenge through making programming more practical, creative and engaging. These include visualisation tools and games. One such approach is to use programming, through graphical libraries, as a means for students to create a physical artefact. This can be done using digital fabrication machines such as 3D printers. Similar approaches have been found to achieve significant success in increasing student motivation towards programming – further showing dramatic improvements in student engagement, confidence, problem-solving abilities, among others– as we will discuss in section 2. However, the particular usage of 3D printing has not been studied in much detail.

In this project, we aim to investigate the prospect of creating models specifically designed for 3D printing, as a tool for teaching first-year programming. In particular, we investigate its effects on improving students' motivation and engagement with the subject.

1.1 Project Structure

This research forms part of a collaborative project researching the development of a user-friendly tool for beginner programmers to programmatically design and simulate the printing of 3D models. In this paper, we focus only on the design component; however, the system works together as one.

The system developed is targeted towards students, but we were unable to obtain student participants for this study due to the COVID-19 pandemic occurring during the study period. Thus, we opted to use expert evaluation instead.

We first developed a web-application, Model3D, for programmatic graphical modelling in JavaScript, described in section 3. Thereafter, we evaluated this tool among educators for usability and suitability for improving student motivation in an educational context.

1.2 Research Aims and Hypotheses

Based on the previous literature, the use of graphics libraries to engage and motivate students in learning introductory programming has consistently shown to be useful. Further, creating and interacting with physical objects, such as those created by 3D printing, increases experiences of autonomy, involvement and places learning into context for students. Thus, combining programming 3D models with the option to create a physical object is expected to be effective at increasing student motivation. We investigate this by evaluating the effectiveness of Model3D as a proof of concept.

The hypotheses are:

(1) Educators will consider Model3D user-friendly for novices
(2) Educators will consider Model3D useful for increasing student motivation in learning introductory programming

1.3 Project Significance

There is a shortage of usable novice-oriented programmatic computer aided design tools. With 3D printing being a rising technology with widespread applications, this system, if satisfactorily usable, will provide an accessible entry point for beginners interested in 3D printing. It also provides an engaging and unique tool for teaching and learning programming. If our hypotheses are correct, this study will also provide evidence for an effective intervention to the problem of student motivation in computer science education. This is tied to improved student retention as well as better performance in programming courses [19].

1.4 Ethics

This study used human participants and thus needs to consider associated ethical issues. Before participants were allowed to partake in any activities relating to our study, they were required to read and sign a consent form (see appendix). There were no physical risks involved in the study. Ethical clearance was obtained from the faculty as well as the university. No identifiable participant information was released with the results, and all data was kept on a password protected computer only accessible to the researchers and supervisors on this study. All software identified for use has been published under open source licenses. The software developed in this study has also been released open source, under the GNU General Public License [1].

2 BACKGROUND

2.1 Computer Science Education

2.1.1 Importance of Student Engagement. Many factors impact the level of engagement in a CS course.Biggers, Brauer & Yilmaz [12] compared graduating CS students with those who chose to leave the major, and identify five core challenge areas affecting student perception of computer science:

(1) Loss of interest due to limited understanding of the "big picture" of CS
(2) Lack of relevance of coursework
(3) Perceived asociality in the subject
(4) Lack of familiarity with CS and lack of confidence
(5) Perception of coursework as boring and pointless

These challenges speak to a lack of intrinsic motivation towards computer science. Contextualisation, personalisation and offering choice are some of the most effective strategies at encouraging intrinsic motivation for abstract tasks [14]. It is then no surprise that the recommendations of Biggers et al [12] for most of these factors involve contextualising coursework within real life and presenting computer science as being about more than just coding.

Factor 4 aligns with performance in the course. However, several studies show that motivation and performance are deeply intertwined – higher motivation leads to higher performance, and vice versa [22][11]. Thus, one of the most important aspects of CS education to address is increasing student engagement and motivation, especially by placing programming in a larger context.

2.1.2 Theoretical Background: Constructivism and Constructionism. Constructivism is a theory of learning that says that people learn by experiencing things in the world, reflecting upon them, and then integrating them into their existing ideas in order to construct their understanding of the world [10]. The most important idea for the present paper is that learning is active rather than passive.

Honebain [10] says that constructivist learning environments should aim to embed learning in realistic contexts, encourage students to have a voice and ownership of the learning process, and to encourage multiple modes of representation. These goals are similar to the strategies identified in 2.1.1 Constructivist learning environments also emphasise the importance of process over product, pursuit of students' own interests and learning as an interactive process.

Seymour Papert [28] takes constructivist theory a step further to formulate a learning approach known as 'constructionism'. Papert says that the mental construction of knowledge happens most effectively when the learner is consciously involved in constructing an actual entity – learning by creating. This approach is manifested in CS education through (a) the *Creative Computing* approach, and (b) a physical computing approach. We combine these ideas in the development of our system.

2.2 Creative Computing

The most direct realisation of constructionism is the programming language Logo [5], developed by Papert himself in 1967. The environment consists of an area for entering text commands, and an area for a graphical turtle that moves around according to the entered commands. The path of the turtle traces a line onto the screen, and students can use this to draw a variety of pictures. This provides both a visual representation of the commands written, and an avenue for students' creativity. Educational programs following the Logo philosophy and using the language across over 250 classrooms were found to have dramatic improvements in student engagement, confidence, problem-solving abilities, and motivation, among others [31].

Since then, a great number of visual and creative computing tools have been developed for the teaching of introductory programming.

One such approach is the use of tools that allow a student to create a narrative, and is popular and successful. Scratch [2] is a visual, drag-and-drop programming environment derived from Logo. It allows students to create stories, games and animations, and has become one of the most popular tools for CS education [27]. It has also been found to have a high degree of success in supporting understanding CS concepts such as loops and conditionals, computational thinking, and engagement, across several contexts [24][35][29]. Alice [7] is a similar environment that extends this concept to three-dimensional objects as a means of teaching object-oriented programming.

Alternatively, conceptualising computation as a creative medium itself, many technologies extend the basic Logo turtle concept with added functionality, allowing students to create a broader range of aesthetically pleasing designs through simple code. One example

is Processing [16], a programming language designed to facilitate this. It is based in Java and provides a flexible set of code elements to create visual designs in both two and three dimensions.

Xu et al. [34] created a comprehensive creative-computing-based curriculum for first year computer science courses that, unlike most research in the field, has been applied at several institutions in the long term. It thus provides useful perspective on the sustainability and longitudinal effectiveness of creative computation approaches. They report great success in improving student interest and retention through this approach. Educators comment that it created, "by far the most engaged group of students we have worked with" in five years. The courses' pre-registrations often greatly exceed capacity, and they boast close to 100% retention rates [56].

Expanding these encouraging results into the physical realm, Jacobs and Buechley [18] created a Processing-based library for digital fabrication, Codeable Objects. This was inspired by the lack of novice-oriented computational design tools for digital fabrication. The Codeable Objects library plugs into the Processing environment and provides a means to create and export designs for 2D digital fabrication machines, such as laser cutters. After workshops, students were found to have enjoyed the process, be pleased by their creations and more comfortable with programming than before. Almost all participants indicated that they would like to program again in the future, and many said that they are specifically interested in 'making' and the creation of things that are relevant and useful in their lives.

The present project can be seen as a further conceptual expansion of the Codeable Objects approach, in that it provides a library for modelling designs for digital fabrication, but for 3D digital fabrication machines, i.e. 3D printers.

2.3 3D Printing & Modelling

The typical 3DP workflow is divided into the following four stages by Nandi et al [25]: design, compilation, printing, and iteration. This project concentrates on the design component.

A 3D model is usually designed using some Computer Aided Design (CAD) software. There is a large variety of software available for this purpose, both programmatic, such as OpenSCAD [21], and graphical. These tools function to render designs onto the screen as well as convert into formats usable for later stages.

For CS education, our emphasis is on programmatic tools, specifically tools for parametric design – design through algorithmically defining relations between structures. OpenSCAD is a free, text-based parametric solid CAD modelling tool. It renders C-style code describing the properties of the object to be modelled. Similar tools exist in block-based programming form, including BlocksCAD [8], Beetle Blocks [30] and TinkerCAD [9]. The latter two are free software.

Parametric designs usually consist of *shapes*, such as cubes and spheres; *transformations* such as rotation and translation; *Constructive Solid Geometry* operations for combining shapes in various ways, such as union, difference and intersection; as well as conventional programming aspects such as loops, modules and conditionals [13].

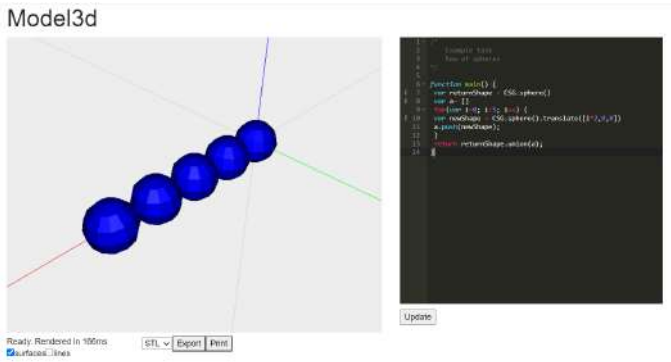


Figure 1: Basic layout of the system, showing the code for creating a row of spheres

3 SYSTEM DESIGN

3.1 Overall Structure and Architecture

We implemented the system as a web application to simplify the process for students, who can just access it through their browser without the need for any installation. The system is designed to facilitate the whole process from design to printing.

Designing a 3D model programmatically requires a development environment, graphical libraries to specify the objects and parameters forming the model, and a previewing function.

The core Model3D[1] application provides a typical IDE (Integrated Development Environment) with a text editor and the ability to run the implemented code. The viewer renders the code on-screen as a 3D visualisation of the model. This interface is shown in figure 1.

A user implements writes code that specifies a 3D artefact. The client sends this javascript file to a web server which runs the implemented code with the OpenJsCad backend. The server returns a graphical representation of the artefact to be rendered in the user interface. The user can now interact with their 3D artefact by rotating and zooming into it. An sequence diagram is shown in figure 2. NodeJs is used for the server backend.

When happy with their model, they can move to the printing simulation, or download a STL file that can be sent directly to a physical 3D printer.

3.2 Language

We chose JavaScript to be the language being taught using our system. This is because JavaScript is a highly versatile language – supporting imperative, object-oriented, and functional programming. Further, it is easy for beginners to get started and see immediate results in their web browser without any further installation required, and minimal boilerplate. It is also one of the most popular programming languages, used by over half of the developers on stack overflow [6].

[1]Code for the combined project can be found at the collaborator's github page at: https://github.com/camadams/code3Dfinal

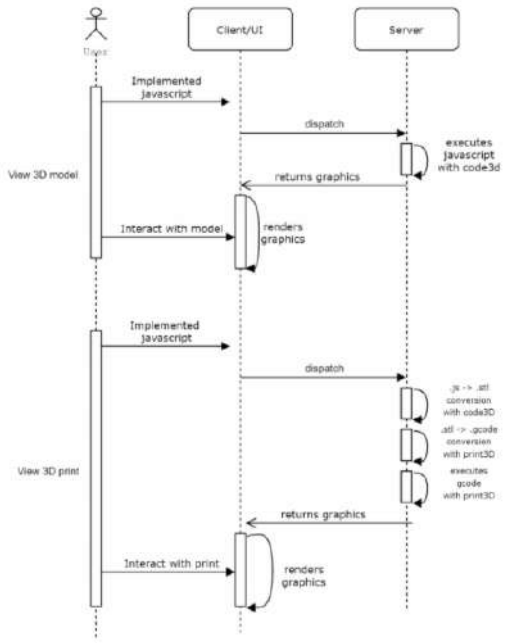


Figure 2: Sequence diagram of the visualisation aspect (the print3D parts are out of scope)

3.3 Text Editor

We used Ace [1] for the text input area in the web-app. Ace is a popular text editor written in javascript and designed to be embedded in web pages. We chose this editor because of its suitability to integrating in the web-app, as well as the extensive features available, such as syntax highlighting, automatic bracket-completion and indentation, and, importantly, live syntax-checking.

3.4 Graphics Library

Graphical modelling was done using the CSG.js [32] library by Ewan Wallace. It makes use of the Constructive Solid Geometry approach to CAD, which consists of having solid object primitives whose volumes are combined using set Boolean operations. It processes these objects into renderable meshes that are used by the viewer. The basic primitive objects available are sphere, cube, and cylinder, with some variations, see figure 3. However, it is also possible to define custom polyhedra by specifying points and faces.

The set Boolean operations are union, intersection, and difference, see figure 4. Any level of complexity can be achieved through these basic objects and operations.

3.5 Processing, Rendering, and Conversion

The OpenJsCad [20] library was used for processing model data. It handles the text input, employing the CSG library in order to process it into mesh designs. It also sets up the viewer and processes the mesh designs into a format that can be displayed. The designs are rendered on-screen with a viewer based largely on lightgl.js [33] and three.js [3], which are tools for WebGL (Web Graphics Library) – a Javascript API for rendering interactive designs within web browsers.

Finally, the OpenJsCad library also converts the design into the STL format that can be downloaded for printing or sent to the printing simulation.

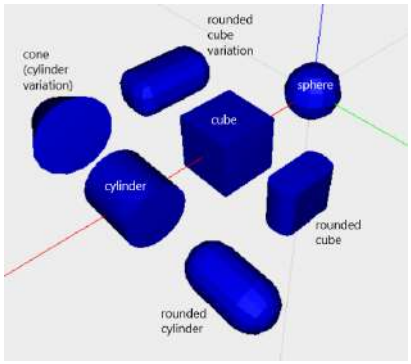


Figure 3: Basic set of CSG primitives

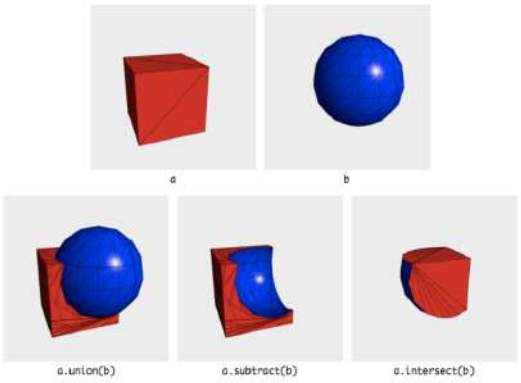


Figure 4: Illustration of the CSG operations: union, intersection, and difference

4 EVALUATION

4.1 Research Design

Due to the constraints imposed by the COVID-19 pandemic, we chose to use expert evaluation as our testing strategy for assessing the system. On account of the small available sample size, we employ a mixed-methods approach, combining both quantitative and qualitative methods. This consisted of:

(1) Qualitative observations about system usage made during user testing sessions
(2) Quantitative surveys regarding system usability and motivation
(3) An open-ended "general feedback" question added to the surveys (qualitative)
(4) Semi-structured interviews to discuss in greater depth participants' experiences and background

The usage of multiple measurement approaches means that results are more likely to be due to underlying phenomena than the particular method used, reducing bias and error and improving validity [19].

4.2 Participants

All participants were staff members from the Computer Science department at the University of Cape Town, recruited using the purposive sampling method through our supervisor, Gary Stewart. This means that participants are selected on the basis of the experience and knowledge they possess, which is an efficient method for collecting expert data [15]. The inclusion criteria for participants were that they should have some experience with teaching introductory programming in a university context.

We contacted 7 qualifying staff members via email with an invitation to participate – 2 first-year computer science lecturers, 4 teaching assistants, and a scientific officer who assists in running the courses. The 2 lecturers were unavailable; thus, the study consisted of the remaining 5 participants. All participants were male, and the TAs were in the 21 – 35 age range and concomitantly pursuing either Masters or PhD studies in Computer Science at UCT. The age of the scientific officer is unknown. Two of the participants were black, two white, and one Indian.

4.3 Materials/Measures

The instruments discussed below have been widely applied and found to be reliable and valid, with reliability scores between 0.8 and 0.97.

Post-Study System Usability Questionnaire (PSSUQ). [13] Assessment of system usability will be done using the PSSUQ, a 19-item questionnaire with 7-point Likert items. This questionnaire measures overall satisfaction with the system, along with three subscores: system usefulness, information quality, and interface quality. The survey can be found at this Google form.

Modified Survey About Educational Robotics for Instruction (M-SERI). [7] Instructor attitudes towards the system as a tool for increasing motivation in teaching will be assessed using the M-SERI, an 18-item questionnaire with 5-point Likert items. Three subscores can be calculated: ability to use the tool, perceived utility, and intent to use it in the future. We modify the original survey by interchanging "educational robots" with "3D printing" and altering questions to concentrate on utility towards motivation – this modified survey can be found at this google form

4.4 Procedure

Evaluation was done in two stages: (i) an initial pilot study, which provided feedback and suggestions for the system, and (ii) a final assessment to evaluate the improved system once suggestions had been implemented. The overall structure followed for both stages was similar. Participants booked a timeslot online for an individual video call testing session. They were then sent an agenda for the session, and instructions for setting up the web-app on their computers in preparation. Once the session had concluded, they were sent links to the two surveys to complete. The procedures followed during a session for each stage are detailed below.

4.4.1 Pilot Study. The main objective of the pilot study was to garner feedback regarding obvious issues with the user interface. The session began with an introduction to the project and the system overall, including a tour of the interface and a brief overview of how to use the OpenJsCad modelling library. We also showed them a demonstration of creating a simple model – a row of spheres (see figure 1). Participants were then asked to recreate two simple models using the tool, based on pictures provided of the desired objects.

These two models were selected for their simplicity, while still employing several of the basic concepts of the library – shape objects, transformations and CSG operations. The first task, cube grid, is a straightforward extension of the row of spheres demonstration into two dimensions. The second task, spiral pyramid, continues the usage of cube objects, CSG unions, and translation transforms, but also incorporates rotation and is therefore slightly more advanced.

The researchers supervised participants during these tasks, providing guidance and assistance where needed, as well as taking notes on challenges encountered or criticisms mentioned by the participants. Upon the completion of these tasks, an opportunity was provided for informal discussion and general feedback.

4.4.2 Final Assessment. As users were already familiar with the project and basic interface at this stage, we only provided an overview of the changes from the previous version as an introduction. Similarly to the pilot study, participants were then asked to recreate two models.

The first model, cube tower, came with skeleton code provided as a starting point. This allowed a demonstration of how external code can be used in the web-app and user-defined functions incorporated, as well as assign a more complex model without overwhelming the users. The second model, pyramid, was selected for being fairly simple to create, but with a strategy not immediately obvious from the image. Through this, participants were encouraged to make creative attempts to approximate the shape. As in the pilot study, the researchers provided supervision, and noted criticism as well as observations.

Once the tasks were complete, we commenced a semi-structured interview. The questions sought to gather information around the following main categories:

(1) Participant background and context
(2) Experience of and feedback regarding the system
(3) Perceptions of relevant concepts before and after engaging with the system
(4) Expectations of student experiences using the system

5 RESULTS

5.1 Pilot Study

One of the participants was unavailable during the pilot study and was only present for the final evaluation. Further, one of the participants failed to complete the motivation survey, thus leaving 4 and 3 participants for the usability and motivation components respectively. Consequently, many of the results were not found to be statistically significant.

Table 1: Initial usability survey score and subscores

M-SERI score	*M*	*SD*	Std Err	t
OVERALL	4.26*	0.42	0.24	5.23
ABILITY	4.22*	0.35	0.20	6.10
UTIL	4.47*	0.31	0.18	8.32
INTENT	4.14*	0.57	0.33	3.46

Table 2: Initial motivation survey score and subscores

M-SERI score	*M*	*SD*	Std Err	t
OVERALL	4.26*	0.42	0.24	5.23
ABILITY	4.22*	0.35	0.20	6.10
UTIL	4.47*	0.31	0.18	8.32
INTENT	4.14*	0.57	0.33	3.46

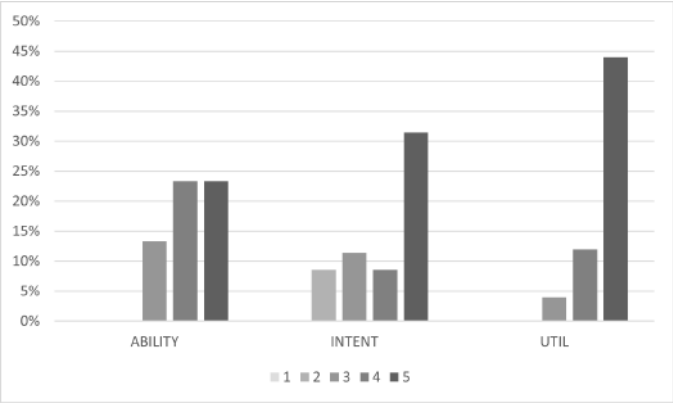


Figure 5: Distribution of ratings for each of the PSSUQ subscales in the initial assessment

The results showed that the participants found the system usable and effective as a tool for increasing student motivation in introductory programming courses, however, many of the scores were very varied between participants. The overall mean usability score on the PSSUQ was 5.68 out of 7 (N = 4, SD = 0.63), and the mean motivation score on the M-SERI was 4.24 out of 5 (N=3, SD=0.42). A one-sample, single-tailed t-test was conducted on these scores and found that there was a statistically significant difference between the achieved total scores and the expected mean of 4.0 for the PSSUQ, and 3.0 for the M-SERI ($p < 0.05$). This was not, however, the case for all of the subscores. A summary of these scores is shown in tables 1 and 2; the statistically significant scores are marked with an asterisk ('*'). The distributions can be seen in figures 5 and 6.

Most of the individual items did not show any statistically significant values, and have not been analysed.

5.2 Final Assessment

The results showed that the participants found the system usable, and a viable tool for increasing student motivation towards programming, with a mean score on the PSSUQ of 5.98 out of 7 (N = 5, SD = 0.43), and 4.4 out of 5 on the M-SERI (N = 5, SD = 0.34). A one-sample, single-tailed t-test was conducted on these scores and found that there is a statistically significant difference between the achieved total scores and the expected mean of 4.0 for the PSSUQ, and 3.0 for the M-SERI ($p < 0.001$). This was also found to be the case for each of the subscores. A summary of these scores is shown in tables 3 and 4. The same test was also conducted for each of the questions individually; these scores can be found in the appendix at 5.

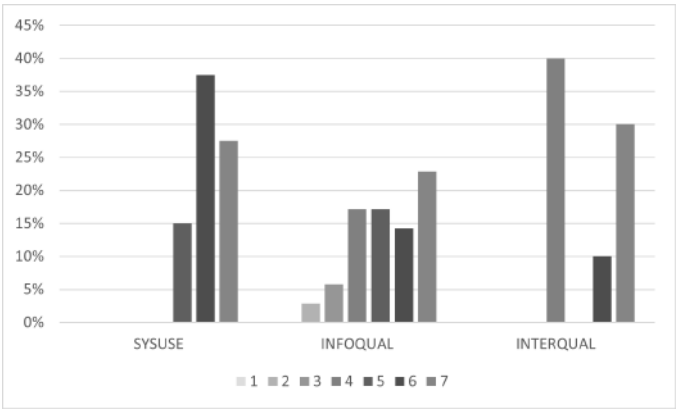


Figure 6: Distribution of ratings for each of the M-SERI subscales in the initial assessment

5.2.1 Usability Survey. All of the participants agreed that the system was useful (SYSUSE), and the quality of the interface (INTERQUAL) was good, with unanimously strong agreement on the latter. The distribution of responses can be seen in figure 7. The responses regarding the quality of information provided (INFOQUAL) were less definitive, but over half the participants (54%) still agreed that it was good.

All but two of the individual questions on the survey were also found to have statistically significant differences between the achieved and expected average ratings on a one-sample t-test, $p < 0.05$. These two questions were from the INFOQUAL subscale – item 11 ("The information [..] provided with this system was clear"), and item 12 ("It was easy to find the information I needed"). These were also the lowest-scoring items, along with item 9 ("The system gave error messages that clearly told me how to fix problems").

The highest-scoring items on the survey were item 8 ("I believe I could become productive quickly using this system") and 17 ("I liked using the interface of this system"), with a mean score of 6.60 (SD = 0.55).

5.2.2 Motivation Survey. Participants were overwhelmingly in favour of the system's viability as a tool to increase student motivation (see figure 8. Over 90% agree or strongly agree that they feel capable of using the system in the classroom (ABILITY), and that it will be useful in that context (UTIL). Most (77%) participants also agree that they would want to use the system in the classroom (INTENT), but some are more uncertain about this.

The highest-scoring item was item 6 ("I am confident that I could learn to use this software for teaching"), with all participants rating

Table 3: Final usability assessment score and subscores

PSSUQ Score	*N*	*M*	*SD*	Std. Err	*t*	*d*
OVERALL	5	5.97	0.43	0.19	10.20	1.88
SYSUSE	5	6.25	0.44	0.20	11.38	2.10
INFOQUAL	5	5.51	0.55	0.25	6.16	1.14
INTERQUAL	5	6.50	0.50	0.22	11.18	2.07

Table 4: Final motivation assessment score and subscores

M-SERI Score	*N*	*M*	*SD*	Std. Err	*t*	*d*
OVERALL	5	4.41	0.34	0.15	9.30	1.72
ABILITY	5	4.60	0.30	0.14	11.82	2.18
UTIL	5	4.52	0.46	0.21	7.38	1.36
INTENT	5	4.17	0.50	0.22	5.25	0.97

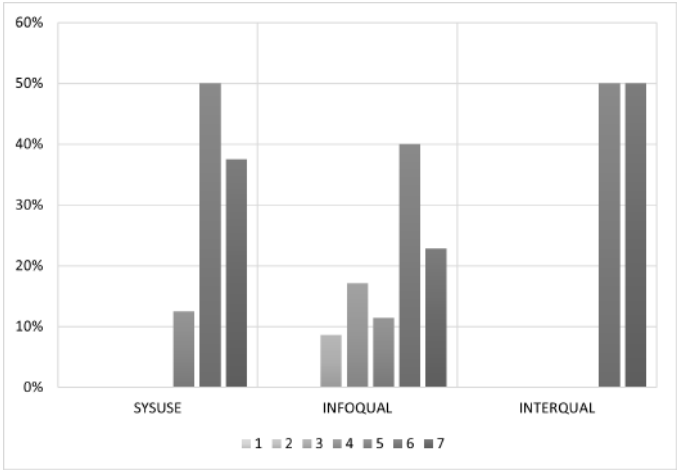


Figure 7: Distribution of ratings for each of the PSSUQ subscales in the final assessment

strong agreement with this statement. The second-highest scoring items (M = 4.80, SD = 0.45) were:

- 1. This software sounds like problematic instructional technology to me. (reverse-scored)
- 8. I think this software is not interesting. (reverse-scored)
- 13. I believe this software will help motivate students.
- 14. I would enjoy using this software in my classes.

The lowest-scoring items did not achieve statistical significance at $\alpha = 0.05$ – item 7 ("The challenge of using this software for instruction does not appeal to me") and item 10 ("I will do as little work with this software as possible").

5.2.3 ***Qualitative Assessment.*** The analysis of qualitative data was done using thematic analysis. Interview transcripts, along with responses to the open-ended question on the surveys, were coded by topic by the researcher. Thereafter, the labelled data was analysed and the following themes were identified:

Theme 1: *Perceptions of the Programming Process.* It is known that students often find learning programming challenging, a statement

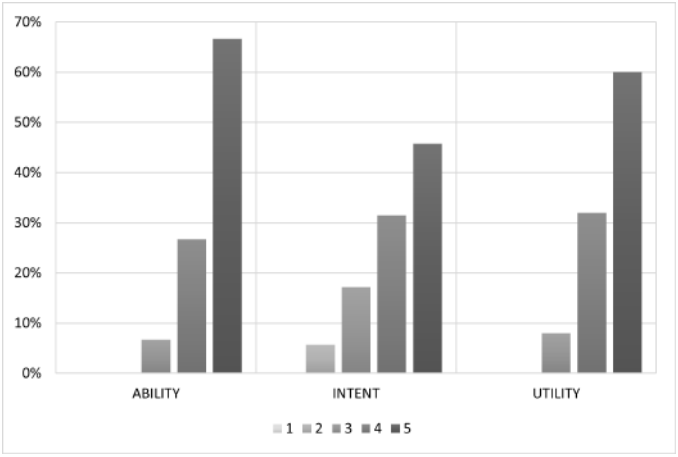


Figure 8: Distribution of ratings for each of the M-SERI subscales in the final assessment

that all the participants agreed upon. However, for most of them, they did not personally find it difficult.

Luke says, "I don't really see programming as a problem for me, because I believe any problem can be broken into [...] simple steps. I think it can be easy if done step-by-step." Benjamin similarly says that he does not perceive programming as difficult, although it can be for some people, because he has a mathematical background and is therefore experienced in problem-solving.

This form of iterative problem-solving was also evident in the participants' style when completing the tasks, where they began by creating a single shape and testing it out, and then attempting to create two, and then a row, and so on. The system design was mostly effective at facilitating this process, as described under Theme 2 below. However, most of the participants did not feel that interacting with the system influenced their perceptions of programming.

Several participants mentioned that using documentation was an important part of their process. Luke, in particular, said, "I don't really memorise things" and emphasised that documentation should be embedded in the system. The absence of this feature was also mentioned by Itumeleng and Rethabile.

Additionally, programming consists of a lot of testing and debugging, and many participants felt that the debugging aspects provided (Javascript stack traces) were unsatisfactory in this regard.

> The error feedback is not very clear. For Model3D this comes in the form of inserting the stack trace above the code, which just clutters the UI.

Benjamin mentions that the lack of error catching might be discouraging for beginners if they face repeated failures, and this is his least favourite aspect of the system.

Theme 2: *Feedback & Interactivity.* Programming can present a challenge to students when neither the final outcome nor the steps in the process of creating a piece of code are immediately obvious. For example, Itumeleng said that he has always disliked programming and continues to, because the process seems opaque to him and he struggles to see how code relates to outputs.

The rapid-feedback and interactivity in the system allowed participants to instantly see the results from their code on-screen and interact with the model. Luke expressed that this is "really great for motivation" – moreover, students can keep checking and modifying the code accordingly until they get the results they want:

> It gives a very clear visualisation of the code, and with the code and result on the same screen, it is easy to see the result of small changes to the code and figure out how things work.

Itumeleng mentioned that in this case, he "really enjoyed" writing code. Benjamin also mentioned that it was really significant for him to be able to immediately see the results of his code – this experience was expressed almost as an 'aha' moment, where he can say, "Oh, it does this!" He said that the immediacy of the updating provides a sense of achievement that is "invaluable for beginners". Several participants also mentioned that physical 3D printing is a slow process; in contrast, this form of short feedback-loop would be beneficial for motivation.

This was by far the most commonly-expressed theme, with mentions from all 5 participants, and named by 3 participants as their favourite aspect of the system.

Theme 3: *Concrete Creations.* All of the participants considered the creation of an actual artefact beyond pure code a major source of value for the project. At least 3 of them used the exact words that a student might express when succeeding at it – "I made this!" – something that provides a great sense of satisfaction and pride in their work.

Several participants expressed that this would only be heightened with the ability to create an actual physical artefact if the project is carried forward into physical rather than virtual printing. Michael mentions that students could be able to create real, usable items that they could show their friends and boost that sense of pride. Itumeleng also mentions that it would be "really nice" and "even cooler" if a physical object could be created.

Theme 4: *Contextualisation.* Many participants' motivations for entering computer science revolved around the potential of technology. Itumeleng says that he studied computer science because of being "excited about new technology" despite not really knowing anything about programming. Michael taught himself programming as a teenager because he enjoyed building things. Luke was interested in game design and engineering in addition to computer science. All of these motivations revolve around real-life applications of computer science.

However, introductory computer science courses often do not connect material to this context. Itumeleng says that the courses were "not what [he] expected". Rethabile discusses the difficulties first-year students often face in learning programming, saying that it can be difficult to move quickly from introduction to many new concepts and then building upon them, lacking any connections made between concepts or real-life context.

3D printing is an emerging technology, and both Rethabile and Itumeleng mention that tools like this can helps students make those connections – both between programming concepts (as in theme 2), and with their applications in real life.

Theme 5: *Practicalities of Teaching.* Several participants spoke about how the system could be directly used for education. Rethabile mentions that the first-year computer science curriculum at UCT already includes an assignment involving the creation of ASCII art, such as triangles. Changing this to three dimensions and having students instead create 3D models such as pyramids, is an obvious point for integration. However, he says that the utility of the system depends to a great extent on how it is used and the quality of teaching involved. He expresses concerns that students may assume that the system is complicated and difficult or intimidating ("scary") due to the printing component, unless it is introduced and explained properly.

Michael discusses the scalability of the system, and upon us explaining that it is based on a fully-featured graphics library used for professional CAD, says that this is very promising:

> A curriculum fit to this tool would be needed, and consideration would have to be taken with regard to the 3D printing costs. However, I think this tool would be a superb addition to the classroom. It is eminently scalable and could work on pretty much every concept in the average first year Computing courses. It wouldn't cover everything, but enough to become a large part of the curriculum.

He also says that, as an educator, he was "very interested in motivation" and is interested in tools that can help facilitate it. Both he and Luke discuss the potential of enabling complex designs and assignments through the use of scaffolding and providing code skeletons.

Benjamin notes, however, that although the system would be helpful to beginners, he might want to use it rather as a secondary step rather than as introduction to those unexperienced with programming. He cites the lack of error catching and potential complexity as reasons for this. Rethabile mentions that in spite of the limitations discussed, he considers it a viable educational tool:

> While I have given a fair amount of feedback I do feel like the project is a great success. I felt really motivated to continue experimenting after trying the tool. The suggestions I have given are aimed at making it more easier to use in an educational setting where users should spend as little time as possible figuring out the tools they use (as that is not the focus of their work).

6 DISCUSSION OF RESULTS

The results collected from our study were found to support our hypotheses, which says that educators will find Model3D to be user-friendly for novices, and useful for increasing student motivation in learning introductory programming. This was supported by the quantitative surveys, which found significantly above average positive attitudes both in terms of usability and utility towards motivation. The qualitative data also supports the hypotheses, where participants express concerns about particular aspects, but on the whole find the system user-friendly, interesting, and useful for teaching.

In terms of user-friendliness, we found that participants felt that the interface was pleasant and useful, rating those scales more than 6 out of 7 on average.

The "information quality" scale was rated less well. This includes item 9 – "The system gave error messages that clearly told me how to fix problems" – which corresponds to participants' criticisms that the system lacked appropriate debugging and error-catching features. It also includes item 11 – "The information (such as on-line help, on-screen messages and other documentation) provided with this system was clear." – which connects directly to participants' criticisms about a lack of documentation provided. We thus conclude that debugging and documentation features are a significant aspect of a system such as ours.

However, participants clearly enjoyed using the system, rating item 17 – "I liked using the interface of this system" – 6.6 out of 7 on average. The largest part of their enjoyment of the system seems to come from the interactivity, and close relationship between the viewer and editor being adjacent on the screen. In the first round of testing, when the editor was located below the viewer and sometimes required scrolling up to see the results, participants were less enthusiastic about the interface. Indeed, the "interface quality" scale's score increased by over 1 point based on this simple change, going from "slightly agree" to between "agree" and "strongly agree".

In terms of motivation, our results suggest that educators are highly enthusiastic about the applicability of Model3D to student motivation. They overwhelmingly agree about their capabilities of using the system in the classroom (rated on average 4.6 out of 5), as well as that it would be useful in that context (rated on average 4.5 out of 5), which is what was predicted by our hypothis. The statements they most agree with include those about the software being interesting and expecting to enjoy using the software in their classes. Most importantly, they rate the core item being investigated, item 13 – "I believe this software will help motivate students" – very highly, 4.8 out of 5.

Their reasons for enthusiasm align well with the theory and literature discussed – the participants ubiquitously place emphasis on the value of rapid feedback and a high degree of interactivity. From a constructivist position, people learn by a feedback loop of experiencing things in the world, reflecting upon them, and then integrating them into their understanding. Learning is an interactive process, and emphasises process over product. The rapid feedback and interactivity aspects of Model3D facilitate this approach, as well as emphasising focusing on the process by encouraging iterative problem-solving.

Participants also expressed that a lot of their interest and enjoyment of the system comes from the ability to create a concrete artefact rather than an abstract programme – "I made this!" – and place programming into a real-life context. This is unsurprising from a constructionist perspective, as it speaks to the embedding of learning in real-life context, and constructing an actual entity.

7 LIMITATIONS AND FUTURE WORK

It is important to keep in mind that this was a proof-of-concept study conducted on a small and limited sample (all males and predominantly teaching assistants) and thus cannot be generalised very greatly. There are limitations on the quantitative side owing to the small sample size, that may have inflated results, as well as on the qualitative side, where data was analysed by only one researcher and is subject to bias. However, the triangulation from the combination of these sources provides some reliability.

Overall, our results are promising and worthy of further investigation. A good next step would be a larger study on a cohort of first-year students as we had initially planned. The feedback obtained about interface issues can also contribute to an improved system – notably, including documentation and debugging features.

8 CONCLUSIONS

Computer science and programming are increasingly important in modern society, but it is considered difficult to learn and students often become disinterested in pursuing it. This has resulted in high drop-out rates and poor performance. We identified motivation as an important factor that affects engagement with the material, retention, and performance. Creative computing, such as graphic design, is one effective intervention for improving student motivation. Physical computing, such as using 3D printers, has also been used to engage students in material.

In this project, we integrated design and making by creating Model3D – a simple web-based tool for students to programmatically design 3D models that can be printed or viewed using the adjacent printing simulation project. This tool was found to be highly engaging by educators, who consider it to be a promising tool for use in introductory programming courses.

REFERENCES

[1] [n.d.]. Ace - The High-Performance Code Editor for the Web. https://ace.c9.io/
[2] [n.d.]. Scratch - imagine, program, share. https://scratch.mit.edu/
[3] [n.d.]. three.js. https://threejs.org/
[4] 2007. Tertiary Level ICT Skills Development. http://www.cs.ru.ac.za/ICTSkills/Declaration%20-ICT-Skills%202007.pdf
[5] 2015. Logo history. https://el.media.mit.edu/logo-foundation/what_is_logo/history.html
[6] 2015. StackOverflow Developer Survey Results 2016. https://insights.stackoverflow.com/survey/2016
[7] 2017. Alice: tell stories. build games. learn to program. https://www.alice.org/
[8] 2019. BlocksCAD editors. https://www.blockscad3d.com/editor/
[9] 2019. Tinkercad | from mind to design in minutes. https://www.tinkercad.com/
[10] Steve Olusegun Bada and Steve Olusegun. 2015. Constructivism learning theory: A paradigm for teaching and learning. *Journal of Research & Method in Education* 5, 6 (2015), 66–70.
[11] Susan Bergin and Ronan Reilly. 2005. The influence of motivation and comfort-level on learning to program. (2005).
[12] Maureen Biggers, Anne Brauer, and Tuba Yilmaz. 2008. Student perceptions of computer science: a retention study comparing graduating seniors with cs leavers. *ACM SIGCSE Bulletin* 40, 1 (2008), 402–406.
[13] Christos Chytas, Ira Diethelm, and Alexandros Tsilingiris. 2018. Learning programming through design: An analysis of parametric design projects in digital fabrication labs and an online makerspace. In *2018 IEEE Global Engineering Education Conference (EDUCON)*. IEEE, 1978–1987.
[14] Diana I Cordova and Mark R Lepper. 1996. Intrinsic motivation and the process of learning: Beneficial effects of contextualization, personalization, and choice. *Journal of educational psychology* 88, 4 (1996), 715.
[15] Ilker Etikan, Sulaiman Abubakar Musa, and Rukayya Sunusi Alkassim. 2016. Comparison of convenience sampling and purposive sampling. *American journal of theoretical and applied statistics* 5, 1 (2016), 1–4.
[16] Ben Fry and Casey Reas. 2020. Processing.org. https://processing.org/
[17] Vashti C Galpin and Ian D Sanders. 2007. Perceptions of computer science at a South African university. *Computers & Education* 49, 4 (2007), 1330–1356.
[18] Jennifer Jacobs and Leah Buechley. 2013. Codeable objects: computational design and digital fabrication for novice programmers. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 1589–1598.
[19] R Burke Johnson, Anthony J Onwuegbuzie, and Lisa A Turner. 2007. Toward a definition of mixed methods research. *Journal of mixed methods research* 1, 2 (2007), 112–133.

[20] joostn. 2016. OpenJsCad. http://joostn.github.io/OpenJsCad
[21] Marius Kintel. [n.d.]. OpenSCAD. http://www.openscad.org/
[22] Külli Kori, Margus Pedaste, Eno Tõnisson, Tauno Palts, Heilo Altin, Ramon Rantsus, Raivo Sell, Kristina Murtazin, and Tiia Rüütmann. 2015. First-year dropout in ICT studies. In *2015 IEEE Global Engineering Education Conference (EDUCON)*. IEEE, 437–445.
[23] Louis Major, Theocharis Kyriacou, and O Pearl Brereton. 2012. Systematic literature review: teaching novices programming using robots. *IET software* 6, 6 (2012), 502–513.
[24] Jesús Moreno-León and Gregorio Robles. 2016. Code to learn with Scratch? A systematic literature review. In *2016 IEEE Global Engineering Education Conference (EDUCON)*. IEEE, 150–156.
[25] Chandrakana Nandi, Anat Caspi, Dan Grossman, and Zachary Tatlock. 2017. Programming language tools and techniques for 3D printing. In *2nd Summit on Advances in Programming Languages (SNAPL 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.
[26] Matthew W Ohland, Sheri D Sheppard, Gary Lichtenstein, Ozgur Eris, Debbie Chachra, and Richard A Layton. 2008. Persistence, engagement, and migration in engineering programs. *Journal of Engineering Education* 97, 3 (2008), 259–278.
[27] Sofia Papavlasopoulou, Michail N Giannakos, and Letizia Jaccheri. 2017. Empirical studies on the Maker Movement, a promising approach to learning: A literature review. *Entertainment Computing* 18 (2017), 57–78.
[28] Seymour Papert and Idit Harel. 1991. Situating constructionism. *Constructionism* 36, 2 (1991), 1–11.
[29] Mona Rizvi and Thorna Humphries. 2012. A Scratch-based CS0 course for at-risk computer science majors. In *2012 Frontiers in Education Conference Proceedings*. IEEE, 1–5.
[30] Eric Rosenbaum, Duks Koschitz, Bernat Romagosa, and Jens Monig. [n.d.]. Beetle Blocks - Visual code for 3D design. http://beetleblocks.com/
[31] Saint Paul Public Schools. 1985. *Logo: Learning in a Computer Culture.* https://el.media.mit.edu/logo-foundation/resources/archive_docs/st_paul.pdf
[32] Evan Wallace. 2016. csg.js. https://github.com/evanw/csg.js
[33] Evan Wallace. 2016. lightgl.js. https://github.com/evanw/csg.js
[34] Dianna Xu, Ursula Wolz, Deepak Kumar, and Ira Greenburg. 2018. Updating Introductory Computer Science with Creative Computation. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*. 167–172.
[35] LeChen Zhang and Jalal Nouri. 2019. A systematic review of learning computational thinking through Scratch in K-9. *Computers & Education* 141 (2019), 103607.

Appendices

Table 5: Final Usability Assessment Survey

	N	*M*	*SD*	*Std Err*	*t*	*d*
1. Overall, I am satisfied with how easy it is to use this system.	5	6.20	0.45	0.20	11.00	2.03
2. It was simple to use this system.	5	6.40	0.55	0.24	9.80	1.81
3. I could effectively complete the tasks and scenarios using this system.	5	6.00	1.00	0.45	4.47	0.83
4. I was able to complete the tasks and scenarios quickly using this system.	5	6.20	0.45	0.20	11.00	2.03
5. I was able to efficiently complete the tasks and scenarios using this system.	5	6.00	0.71	0.32	6.32	1.17
6. I felt comfortable using this system.	5	6.40	0.55	0.24	9.80	1.81
7. It was easy to learn to use this system.	5	6.20	1.10	0.49	4.49	0.83
8. I believe I could become productive quickly usingthis system.	5	6.60	0.55	0.24	10.61	1.96
9. The system gave error messages that clearly told me how to fix problems.	5	5.40	1.14	0.51	2.75	0.51
10. Whenever I made a mistake using the system, I could recover easily and quickly.	5	5.80	0.84	0.37	4.81	0.89
11 .The information (such as on-line help, on-screen messages and other documentation) provided with this system was clear.	5	5.20	1.64	0.73	1.63	0.30
12. It was easy to find the information I needed.	5	4.60	1.82	0.81	0.74	0.14
13. The information provided for the system was easy to understand.	5	5.80	1.10	0.49	3.67	0.68
14. The information was effective in helping me complete the tasks and scenarios.	5	5.80	1.10	0.49	3.67	0.68
15. The organization of information on the system screens was clear.	5	6.00	1.22	0.55	3.65	0.67
16. The interface of this system was pleasant.	5	6.40	0.55	0.24	9.80	1.81
17. I liked using the interface of this system.	5	6.60	0.55	0.24	10.61	1.96
OVERALL	5	5.98	0.43	0.19	10.20	1.88
SYSUSE	5	6.25	0.44	0.20	11.38	2.10
INFOQUAL	5	5.51	0.55	0.25	6.16	1.14
INTERQUAL	5	6.50	0.50	0.22	11.18	2.07

Table 6: Final Motivation Assessment Survey

	N	*M*	*SD*	*Std Err*	*t*	*d*
1. This software sounds like problematic instructional technology to me.	5	4.80	0.45	0.20	9.00	1.66
2. I don't know how to use this software (i.e. this software is unfamiliar to me)	5	4.40	0.55	0.24	5.72	1.06
3. Some day, I will use this software in my classroom.	5	4.00	1.00	0.45	2.24	0.41
4. If given the opportunity, I would like to learn to use this software for instructional activities.	5	4.60	0.55	0.24	6.53	1.21
5. Children should be introduced to this software in school.	5	4.00	1.00	0.45	2.24	0.41
6. I am confident that I could learn to use this software for teaching.	5	5.00	.00000a	0.00		
7. The challenge of using this software for instruction does not appeal to me.	5	3.80	1.30	0.58	1.37	0.25
8. I think this software is not interesting.	5	4.80	0.45	0.20	9.00	1.66
9. I am unsure of my ability to integrate this software into my classes.	5	4.20	1.10	0.49	2.45	0.45
10. I will do as little work with this software as possible.	5	3.60	1.14	0.51	1.18	0.22
11. I think that integrating this software with teaching would take too much time.	5	4.40	0.89	0.40	3.50	0.65
12. I am willing to spend time setting up this software for instruction.	5	4.00	0.71	0.32	3.16	0.58
13. I believe this software will help motivate students.	5	4.80	0.45	0.20	9.00	1.66
14. I would enjoy using this software in my classes.	5	4.80	0.45	0.20	9.00	1.66
15. I think working with this software in class would be enjoyable and stimulating.	5	4.60	0.55	0.24	6.53	1.21
16. I think using this software in class would not work for me.	5	4.40	0.55	0.24	5.72	1.06
17. This software can create more learning opportunities for students.	5	4.60	0.55	0.24	6.53	1.21
18. This software is a valuable educational tool.	5	4.60	0.55	0.24	6.53	1.21
OVERALL	5	4.41	0.34	0.15	9.30	1.72
ABILITY	5	4.60	0.30	0.14	11.82	2.18
UTIL	5	4.52	0.46	0.21	7.38	1.36
INTENT		4.17	0.50	0.22	5.25	0.97

User Testing Agenda

1. Intro (2 min)
2. OpenJSCAD basics (5min)
    a. Anatomy
    b. Functions and objects
    c. System Overview
    d. Example task
3. User task (15mins)
    a. Grid of cubes
    b. Stack of cubes rotation
4. Feedback and discussion (3mins)

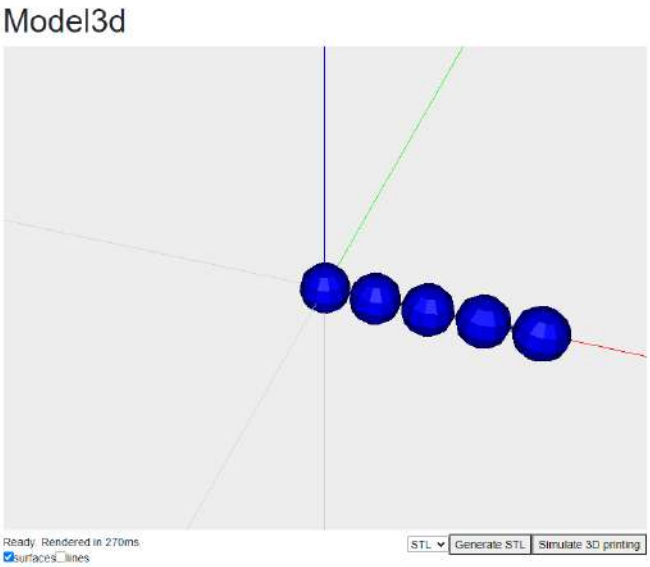
Intro

- **What**: Using graphics libraries and 3D printing to increase motivation in beginner programmers
- **Why**: Graphics libraries such as Processing, p5.js and Turtle have been shown to do so. We try to see if a 3D graphics library, openjscad, and 3D printing simulation can do so to
- **How**: Creating a web app that allows users to create a 3D artefact and watch the simulation of its 3D print

OpenJSCAD basics

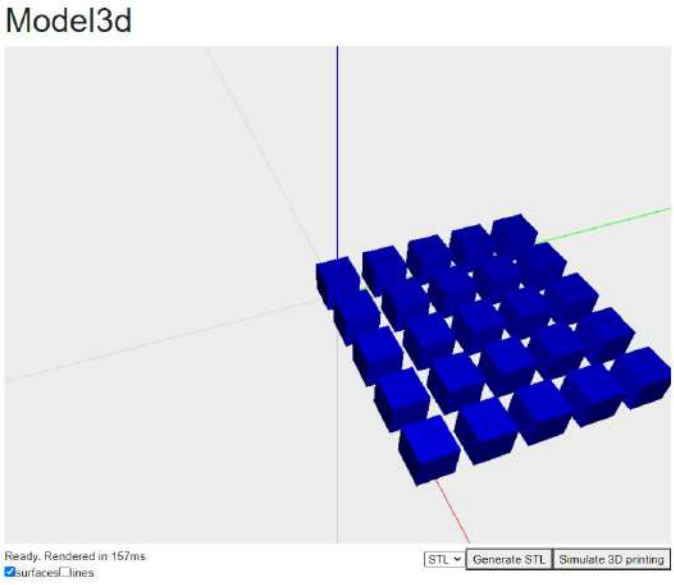
- **Anatomy**: must have a main function returning a ‘shape’ object
- **Functions and objects:** Shape objects - cube, sphere, cylinder. Can union two shapes like cube1.union(cube2)
- **System Overview:** *Model3D:* Text editor with model view. *Print3D:* Print controls with simulation view
- **Example task:** A row of spheres

```
function main() {
    var returnShape = CSG.sphere()
    var a= []
    for(var i=0; i<5; i++) {
        var newShape =  CSG.sphere().translate([i*2,0,0])
        a.push(newShape);
    }
    return returnShape.union(a);
}
```

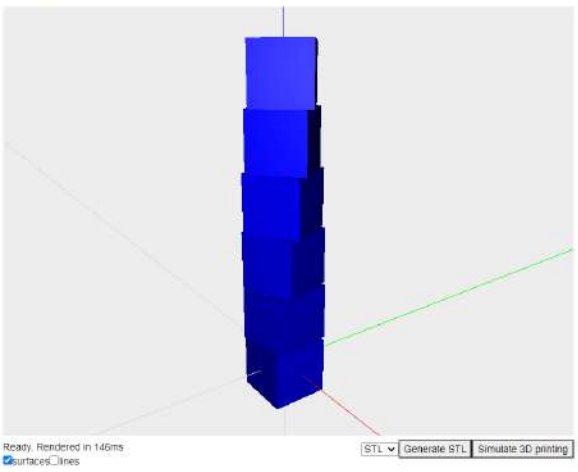


User tasks

1. Cube grid



2. Spiral pyramid



DEPARTMENT OF COMPUTER SCIENCE

UNIVERSITY OF CAPE TOWN
PRIVATE BAG X3
RONDEBOSCH 7701
SOUTH AFRICA

RESEARCHER/S: Cameron Adams
Maria Nanabhai
TELEPHONE: +27653033075
E-MAIL: ADMCAM003@myuct.ac.za
NNBMAR003@myuct.ac.za
URL: https://www.cs.uct.ac.za/

Informed Voluntary Consent to Participate in Research Study

Graphics Libraries and 3D Printing as a Tool for Increasing Student Motivation in Introductory Programming Courses

Invitation to participate, and benefits: You are invited to participate in a research study conducted with staff in the computer science department who have experience teaching introductory programming courses. The study aim is to investigate the applications of 3D printing to improving student motivation in these courses. We believe that your experience would be a valuable source of information, and hope that by participating you may gain useful knowledge.

Procedures: During this study, you will be asked to:

1. Complete some sample programming exercises to create and print a model
2. Participate in a brief discussion regarding your experience working with the 3D printing interface
3. Fill out two surveys.

The entire process is expected to take around 30 minutes.

Risks: There are no potentially harmful risks related to your participation in this study.

Feedback: You will receive feedback about the results of this research in an email once the study has been concluded.

Disclaimer/Withdrawal: Your participation is completely voluntary; you may refuse to participate, and you may withdraw at any time without having to state a reason and without any prejudice or penalty against you. Should you choose to withdraw, the researcher commits not to use any of the information you have provided without your signed consent. Note that the researcher may also withdraw you from the study at any time.

Confidentiality: All information collected in this study will be kept private in that you will not be identified by name, but only as educators at UCT.

Confidentiality and anonymity will be maintained as pseudonyms will be used.

What signing this form means: By signing this consent form, you agree to participate in this research study. The aim, procedures to be used, as well as the potential risks and benefits of your participation have been explained verbally to you in detail, using this form. Refusal to participate in or withdrawal from this study at any time will have no effect on you in any way. You are free to contact me, to ask questions or request further information, at any time during this research.

I agree to participate in this research (tick one box) ☐ Yes ☐No Initials.

Full Name.	Signature (use initials)	Date
_____Cameron Adams_____	_____C.A._____	**13/09/2020**
Name of Researcher	Signature of Researcher	Date
_____Maria Nanabhai_____	_____M.N._____	**13/09/2020**
Name of Researcher	Signature of Researcher	Date