Digital Food FoodCAD

Luna Ruiz (ler2167)

Table of Contents

Section 1: Motivation/Goals

Our team, Digital Food, has created a 3D printer capable of printing edible materials in hopes of automating the cooking process. Unfortunately, in order to print these edible materials, a 3D model must be created using a computer-aided design (CAD) program then converted into an STL file for the 3D printer to read and execute based upon.

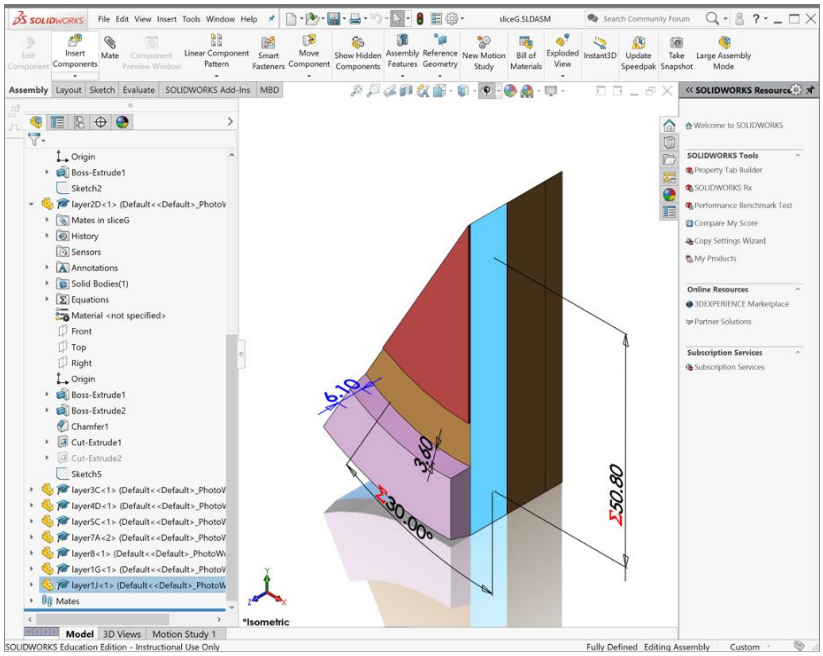

Fig. 1. Example Screenshot of SolidWorks Application

Our concerns were that our average consumer, being anyone at any cooking level, would have trouble navigating a CAD application due to its complex infrastructure. The reality is, CAD software applications require technical expertise and knowledge of their innumerable features. As you can see in Fig. 1., a CAD software application such as SolidWords is busy and not user-friendly for a first-time user like our expected audience. This poses a concern since we want our customers to be able to easily create dishes without any struggle. Aside from its overwhelming interface, existing CAD software applications do not give a true representation of what is to be expected after a dish is printed using our food printer. Fig. 1. does not give the user any idea of what the materials being used are or what the texture of the materials look like. Overall, CAD programs are either not sufficient or provide an unnecessary amount of features to our users.

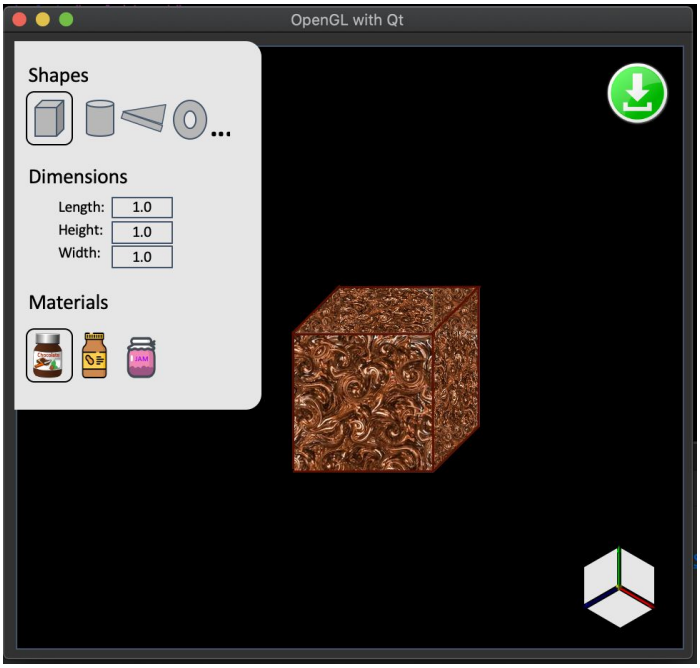

Fig. 2. FoodCAD Interface Mockup

The goal of FoodCAD is to provide users with a minimal, beginner-friendly interface that is easy to use and gives them a true representation of their final product like that of Fig. 2. For example, if one wanted to print nutella cubes shown in Fig. 2., one would choose a cube from the shapes menu, type the dimensions of the shape and choose nutella from the material menu. When a material is chosen, the user would expect to see what the texture of the material would look like on the respective shapes chosen. Finally, they would click the download 3d object in STL form using the green download button and send this file to our food printer, where the users' nutella squares would be printed for consumption.

Section 2: Methods/Process Taken to Accomplish Goals

In order to reach our goal, we conducted some research for current user-interface and 3D rendering software applications, preferably open-source. After much deliberation, we decided to use Qt in conjunction with OpenGL API. Qt is a framework designed to create graphical user interface applications using a widget toolkit. OpenGL API is an API used to render 3D models using vectors. Once we downloaded Qt on our computers along with OpenGL, we split our goals into smaller goals. These goals were:

1. Draw a 3D shape
2. Rotate shape about the x,y and z-axis
3. Draw several shapes and stack them on top of each other
4. Stack shapes of different colors on top of each other
5. Stack shapes of different colors and different dimensions on top of each other
6. Clear all or undo one shape at a time
7. Implement materials to fill in for colors
8. Write STL file for 3D model
9. Implement materials to fill in for colors
10. Write STL file for 3D models and materials

Section 3: Challenges/Solutions

3.1: What went wrong?

Throughout this project, we ran across several obstacles. Initially, we tried to work with another user-interface framework capable of 3D rendering called FreeCAD. FreeCAD is an open source CAD software application meant to work similarly to CAD programs such as SolidWorks. However, it is open source so we thought we could dissect and modify the code in order to implement our desired functionality and our functionality only. We spent a few weeks working on FreeCAD before we realized it was not going to be the best option for us. We struggled to get the program to run correctly. Then, we proceeded to struggle with understanding what the code was doing, some of which we would not need in our final product. After this, we decided to go back to the drawing board and work with Qt. It is one actually used by FreeCAD. We knew this would not be enough to implement the 3D rendering functionality, but we found a way to import the OpenGL API, which gives us access to the 3D rendering functionality we would need.

While we did have experience using the OpenGL API, we were unfamiliar with how to navigate the Qt space. Aside from spending a few hours downloading the needed software to start working on Qt, we had trouble using the widget toolkit and connecting widgets to our code using the signals and slots system. This was essential for connecting and controlling the widgets in our user-interface with our implemented code. When we were able to figure this out, we struggled with reaching some of our goals, specifically goal 8. in Section 2: Write STL file for 3D model. We were not able to successfully create an STL file. It was difficult for us to correctly identify the vector points and normals of the triangles for the 3D model and, therefore, unable to successfully create the STL file for the 3D model. Due to not being able to successfully complete this goal and time constraints, we were unable to implement the other goals.

Finally, in addition to these challenges, we underwent a serious pandemic, caused by COVID-19, that prevented or delayed some of us from working on this project. Most students at Columbia University were told to leave campus and return home to quarantine, while others could not and remained quarantined on campus. This had a huge effect on everyone, some more than others, and had a direct effect on our final results in Section 4.

3.2: What worked?

Despite all the challenges and hardships we faced, we were able to pull through and come up with a product that mostly worked. We were able to create a program that could meet our needs and was possible to work with. Qt had all we needed to implement our graphical user-interface needs, while OpenGL had all we needed to implement our 3D rendering needs. It was a matter of putting them together using signals and slots, which we were able to figure out how to do. More on the specifics of what worked is outlined in Section 4.

3.3: What would you change?

We did what we believed were the right actions but could have spent less time working on things that were not working. For example, we could have spent less time working with FreeCAD. Also, we should have focused on the most important features of FoodCAD first. Rather than adding several shapes and colors, we could have figured out how to make one shape work with one material and convert that to an STL file for printing. This would have put us in a place to test our program against our food printer and determine whether our application was on the right track.

Section 4. Final results

Our final results included being able to successfully implement goals 1-7 from Section 2. This includes the ability to stack different shapes of different dimensions and colors on top of each other, rotate the shape about the x,y, and z-axis, and clear all or undo one shape. All of these functionalities are shown in Fig. 3, where a green cube of size 0.2 is stacked on top of a blue rectangular prism of dimensions 0.1 x 0.3 x 0.5 stack on top of a red cube of size 0.3. There are three buttons in the shapes menu for the user to choose from, a color button for the the user to choose color for next shape to be drawn (to be replaced with materials), spin boxes for the user to specify the dimensions of the shape chosen, three sliders to rotate shape about the xyz axes, clear and undo buttons, and a download to download the 3D model as an STL file (in progress). A screen recording of an example user using the final results of FoodCAD is shown in Fig. 4.

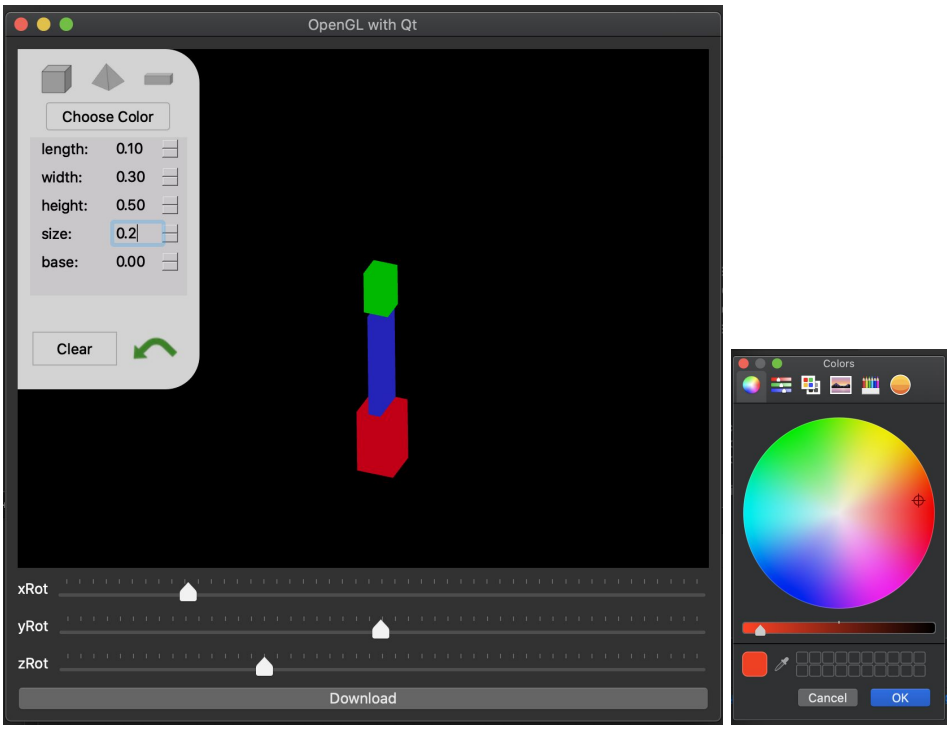

Fig. 3. Screenshot of Final Product with color wheel

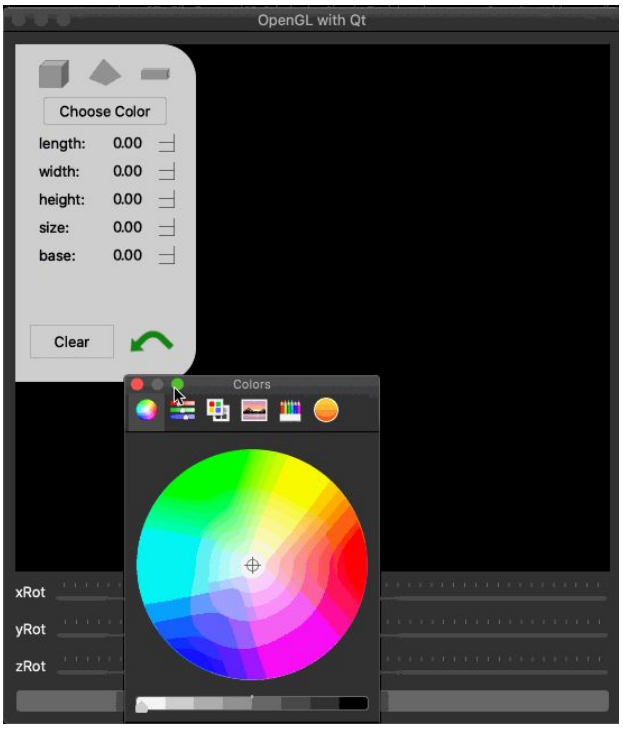

Fig. 4. Screen Recording of FoodCAD program

Section 5: Future Goals

Given that we did not accomplish all of our goals, we would like to focus on doing so in addition to other features. For the features that we did not implement (i.e. materials, 3D model to STL file conversion), we would like to focus on them first in the near future. In addition to this, we thought it would be a great idea if the consumer could edit a shape in between other shapes rather than undoing a shape only on the top of the stack (possibly using edge detection). We also thought it would be great if the user could rearrange the shapes in any order they wanted. Of course, we would also like to add more shapes to the menu. All in all, this project has endless possibilities due to its endless applications.

Section 6: Sources Used During this Project

- https://all3dp.com/what-is-stl-file-format-extension-3d-printing/
  This source was used to give knowledge on the format of STL files.
- https://stackoverflow.com/questions/53897949/opengl-object-outline
  This source was meant to give knowledge on how to highlight different shapes on mouseover or, in our case, how to select a shape.
- https://www.bogotobogo.com/Qt/Qt5_OpenGL_QGLWidget.php
  This source played the biggest role in helping us get started in Qt and OpenGL. This source served as the starting point for our project, teaching us how to create a program that creates a custom widget for rendering 3D objects, how to draw a pyramid using OpenGL, rotate this pyramid about the x, y and z axes and how to use signals and slots to connect our implemented code to our widgets (i.e. connect our sliders to the widget that contains the window where the pyramid is located).
- https://open.gl/drawing
  This source was meant to act as a tool to learn how to implement shaders in OpenGL, which will help in implementing materials for the 3D shapes.