24783 Advanced Engineering Computation: Problem Set 7

In Problem Set 7, you practice:

- 3D Transformation,
- Setting up vertex, color, and normal arrays, and
- Saving a Binary STL file.

(*) In the following instruction (and in all of the course materials), substitute your Andrew ID for where you see *yourAndrewId*.

START EARLY!

1 Check Out or Update Base Code and Libraries

Please make sure you have up-to-date libraries and course files before starting an assignment.

If you have not done working-directory set up as described in the first assignment (like in case you need to work from a different computer), please see Problem Set 1 and set up the working directory.

I assume you created the working directory called *24783* under you home directory and you checked out your Git repository in there.

Home directory is typically *C:\Users\username* in Windows, */Users/username* in macOS, and */home/username* in Linux, where *username* is the user name in your local computer.

First, open command-line (Developer PowerShell or Terminal), and move to your working directory by typing:

```
cd ~/24783
```

You need to check out (or clone) Git repositories once. If you have not checked out yet, do the following:

```
git clone https://yourAndrewId@ramennoodle.me.cmu.edu/Bonobo.Git.Server/course_files.git
git clone https://yourAndrewId@ramennoodle.me.cmu.edu/Bonobo.Git.Server/yourAndrewId.git
```

You need to replace "yourAndrewId" with your Andrew ID. You'll be asked to type in credentials.

Also we are going to use two additional repositories:

```
git clone https://github.com/captainys/MMLPlayer.git
git clone https://github.com/captainys/public.git
```

If you are successful, you should have the following directory structure under your home directory.

```
Your User Directory
├── (Other files and directories)
└── 24783
    ├── course_files
    └── yourAndrewID
```

If you already have checked out these repositories (most likely you did for Problem Set 1), you need to update (or git pull) in those repositories. By change directory to the location where you checked out repositries and then type:

```
git pull
```

To update all four repositories, you can type the following commands in a sequence:

```
cd ~/24783/course_files
git pull
cd ~/24783/yourAndrewID
git pull
cd ~/24783/public
git pull
cd ~/24783/MMLPlayer
git pull
```

2 Copy Base Code and Add to Git's Control

Copy ps7 subdirectory from course_files to your directory. The directory structure must look like:

```
Your User Directory
├── (Other files and directories)
└── 24783
    ├── course_files
    └── yourAndrewID
        └── ps7
```

3 Make a CMake Project

Write CMakeLists.txt for ps7. It is a graphical application, therefore make sure to use MACOSX_BUNDLE. The project name must be ps7. Case sensitive.

In this assignment, you do not have to make sub-directories. You need only one CMakeLists.txt. Make sure to include public libraries. ps7 must link ysclass and fssimplewindow libraries.

This program takes command line arguments. Once you set up CMake scripts and compile, you can run the program by typing (assuming your current working directory is the build directory):

```
.\ps7\Release\ps7.exe 6 5 5
```

if you compile with Visual C++ in Release mode. Or, if you compile on macOS,

```
./ps7.app/Contents/MacOS/ps7 6 5 5
```

4 Calculate ModelView Transformation and Render Vertex Array

Go to ApplicationMain::Draw function. See sample code from 3D rendering, and write model view transformation. The transformation must be calculated from viewRotation, viewDistance, and viewTarget member variables.

Also render geometry represented in vertex array, vtx,col, and nom.

If you do this part correctly, you will see a blue square and you can rotate it by arrow keys.

5 Calculate View Distance

The blue square is a little too big for the window. The view distance must be calculated based on the dimension of the object and field of view.

See the example we did in class, and fill ApplicationMain::ResetViewDistance function. This function must calculate viewDistance so that the object is rendered in reasonable size.

6 Make an Extrusion with an Equilateral Cross Section

Fill ApplicationMain::MakeExtrusion function. This function in the base code makes just one square taking (-5,-5,0) to (5,5,0). Your goal is to write this function so that this function creates vertex, color, and normal arrays of an extrusion of an equilateral polygon.

This function takes three parameters nDiv, radius, and height. The first parameter nDiv defines number of sides of the cross section. The second and the third parameters, radius and height, define the dimension. The main function takes values for nDiv, radius, and height from the command-line arguments, and calls this function at the beginning.

The vertex, normal, and color arrays must be for GL_TRIANGLES. Therefore, it needs to be a set of all triangles. To fill an equilateral polygon with all triangles, you can connect the center of the polygon and each edge to make a triangle as shown in Fig. 1.

The radius of the equilateral is given as the second parameter to the function.

The depth of the extrusion must take:

$$-height \leq Z \leq height$$

.

To fill the gap between front and back faces, you need to add side faces. A side face connecting a pair of edges from front and back faces is a quadrilateral. TO render with GL_TRIANGLES primitives, you need to divide a quadrilateral into two triangles by splitting by a diagonal.

Color array must be made so that front and back faces (two cross sections) appear green, and the side faces appear blue.

If you implement successfully, your program should look like 3. This screenshot is taken with command line arguments 6 5 5.

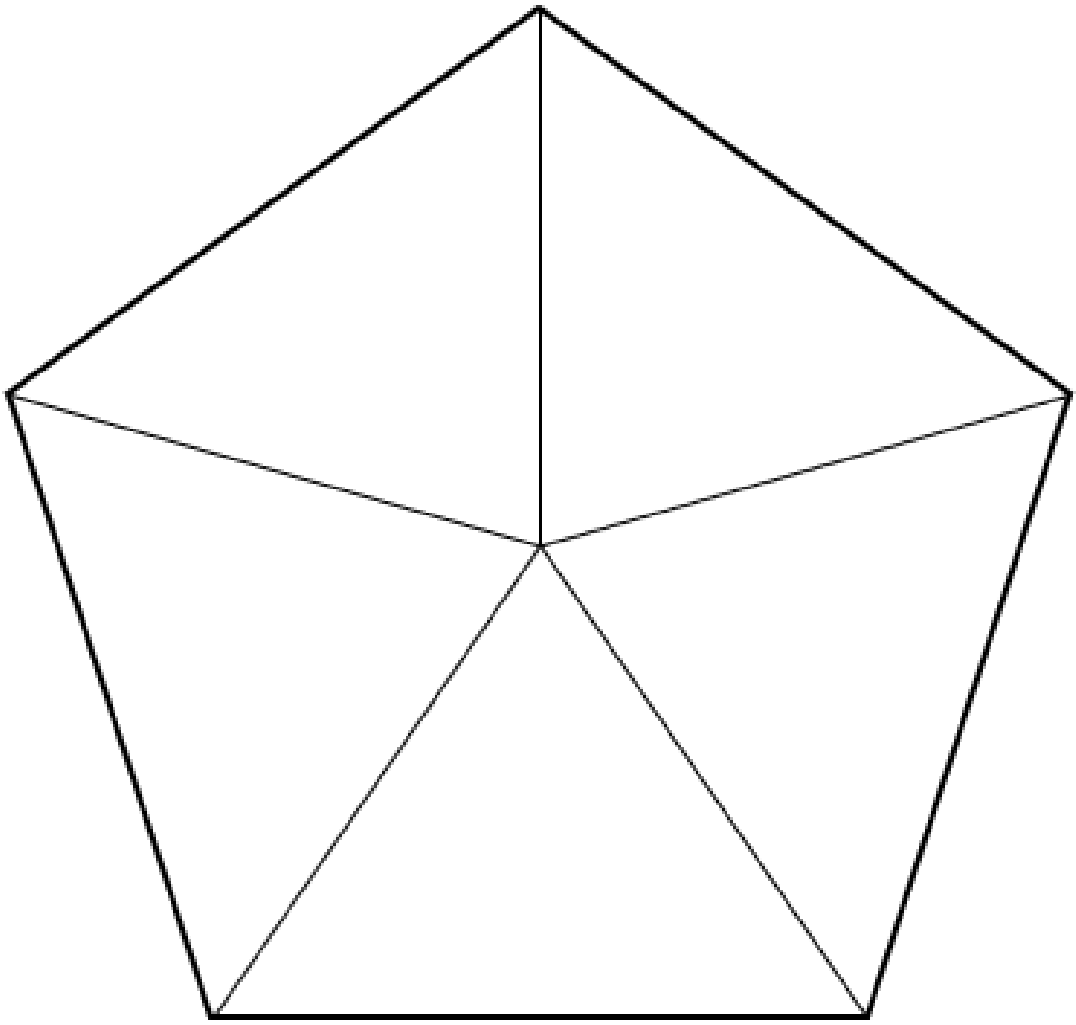

Fig. 1: Divide a Polygon into Triangles by Connecting the Center and Each Edge

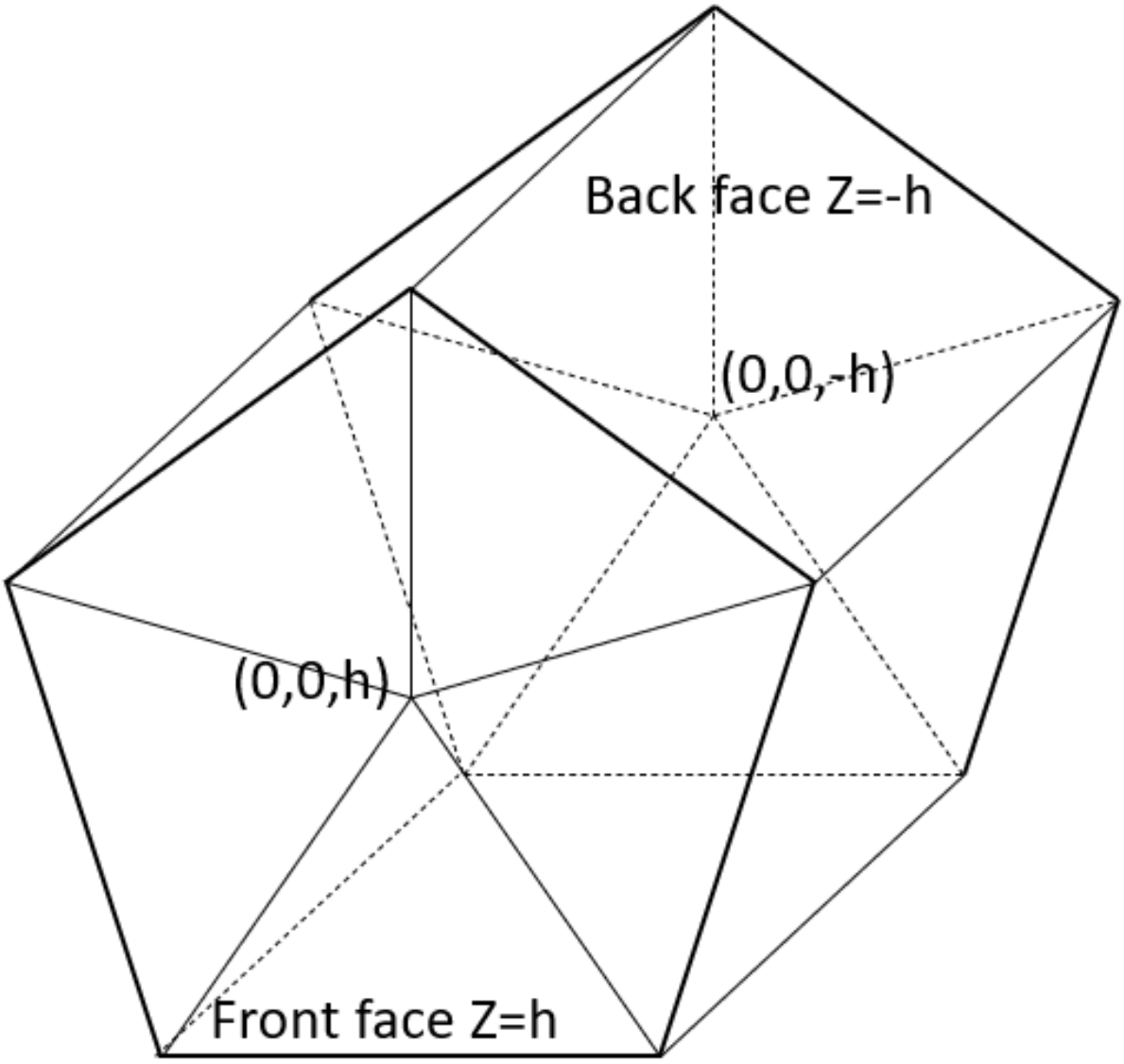

Fig. 2: Make Side Faces by Connecting Front and Back Edges

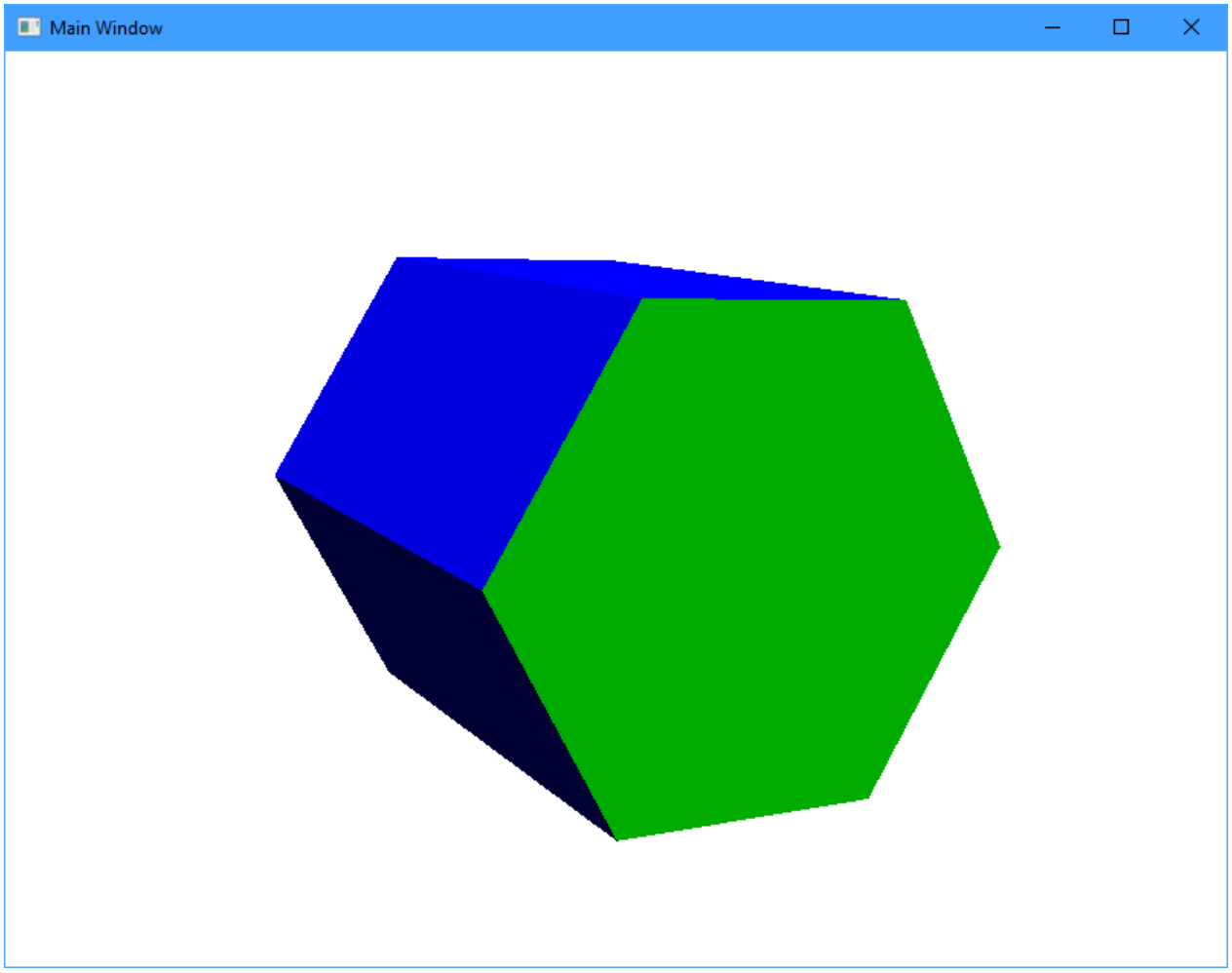

Fig. 3: Screenshot

7 Export it to the Binary STL

Fill ApplicationMain::SaveSTL function. This function takes a file name as input, and needs to save the shape represented by vertex and normal arrays in binary STL format.

This function is called from the main function of the base code. Therefore, if you implement this function correctly, "output.stl" will be saved in the current working directory.

Make STL models with different command-line arguments and verity output with a few STL viewers. You can use one we created in class. Windows 10 has its own built-in viewer. There are other viewers available online.

8 Test Your Code on the Compiler Server

Test your source files (.cpp and .h files) on the compiler server. Some assignment may not require .h files. You do not have to test files that you don't make modifications. The files you need to test are the ones you write or modify.

We have four compiler servers:

- http://freefood1.andrew.cmu.edu
- http://freefood2.andrew.cmu.edu
- http://freefood3.andrew.cmu.edu
- http://freefood4.andrew.cmu.edu

Make sure you don't see red lines when you select your files and hit "Compile Test" button on the server.

We have multiple servers to make it less likely that all of them need to shut down for maintenance. If do not have to test on all of the servers. You need to make sure that your code passes on one of the servers.

9 Submit

Lastly, you need to submit using git. What you need to do are two things: (1) add files to git's control, and then (2) send to the git server.

9.1 Add Files to git's control

In this case, you want to add all the files under ps6 subdirectory. To do so, type:

```
git add ~/24783/yourAndrewID/ps7
```

This command will add ps6 directory and all files under the subdirectories.

9.2 Send to the Git Server

In Git, sending files to the server is a two-step process. The first step is local commit. You can do it by:

```
git commit -m "Problem Set 7 solution"
```

The message can be anything, but it is recommended to type something meaningful, at least you can see what changes you made to your repository.

Local commit is just local. Git server does not know about any local commit unless the commit is sent (or pushed) to the server. To do so, type:

```
git push
```

Make sure to do it in the CMU network. If you are working from home (probably most likely), use VPN to connect to the CMU network.

You can re-submit (commit and push) your solution as many times as you want with no penalty before the submission due.

10 Verification

It is recommended to clone your repository to a different location and make sure that all of your files have been sent to the Git server.

You can do the following:

```
cd ~
mkdir 24783Verify
cd 24783Verify
git clone https://yourAndrewID@ramennoodle.me.cmu.edu/Bonobo.Git.Server/yourAndrewId.git
```

Once you made sure all the files have been submitted, you can delete files and directories under 24783Verify directory.