24783 Advanced Engineering Computation: Problem Set 6

In Problem Set 6, 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 ps6 subdirectory from course_files to your directory. The directory structure must look like:

```
Your User Directory
├── (Other files and directories)
└── 24783
    ├── course_files
    └── yourAndrewID
        └── ps6
```

3 Make a CMake Project

Write CMakeLists.txt for ps6. It is a graphical application, therefore make sure to use MACOSX_BUNDLE. The project name must be ps6. 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. ps6 must link ysclass and fssimplewindow libraries.

4 Write ModelView Transformation

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.

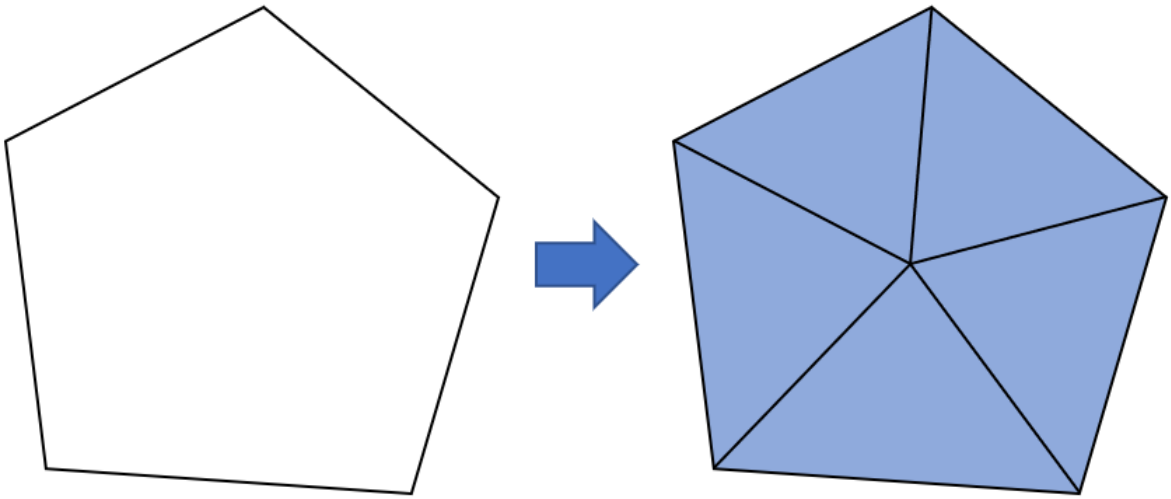

Fig. 1: Divide a Polygon into Triangles by Connecting the Center and Each Edge

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

5 Calculation of View Target and Distance

The blue square is a little too big for the window. Also it is off center. The view target must be the center of the object, and 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::ResetViewDistanceAndTarget function. This function must calculate viewTarget and viewDistance so that the object is at the center of the window and reasonable size.

6 Make an Extruded Shape from Clicked Points

The base code includes clickdraw example we did in class. You can click points on the window to draw a line strip. The coordinates of the line strip is stored in vtx2d member variable.

Modify ApplicationMain::RecreateExtrusion function so that it creates an extruded shape of the polygon defined by the points of the line strip. Add thickness $-100.0 < Z < 100.0$.

The vertex and color buffers must be for GL_TRIANGLES. Therefore, it needs to be a set of all triangles. To make it simpler, you can assume the polygon is always convex (or convex enough) so that it can be decomposed into all triangles by connecting the center and each edge of the polygon as shown in Fig. 1.

Also, for calculating normal vector, assume that the user always draw a polygon clockwise on the window. Flip the sign of Y in vtx3d so that the shape is right-side up in the 3D coordinate.

Make front and back faces all blue.

To fill the gap between front and back faces, you need to add side faces. You can easily imagine that you

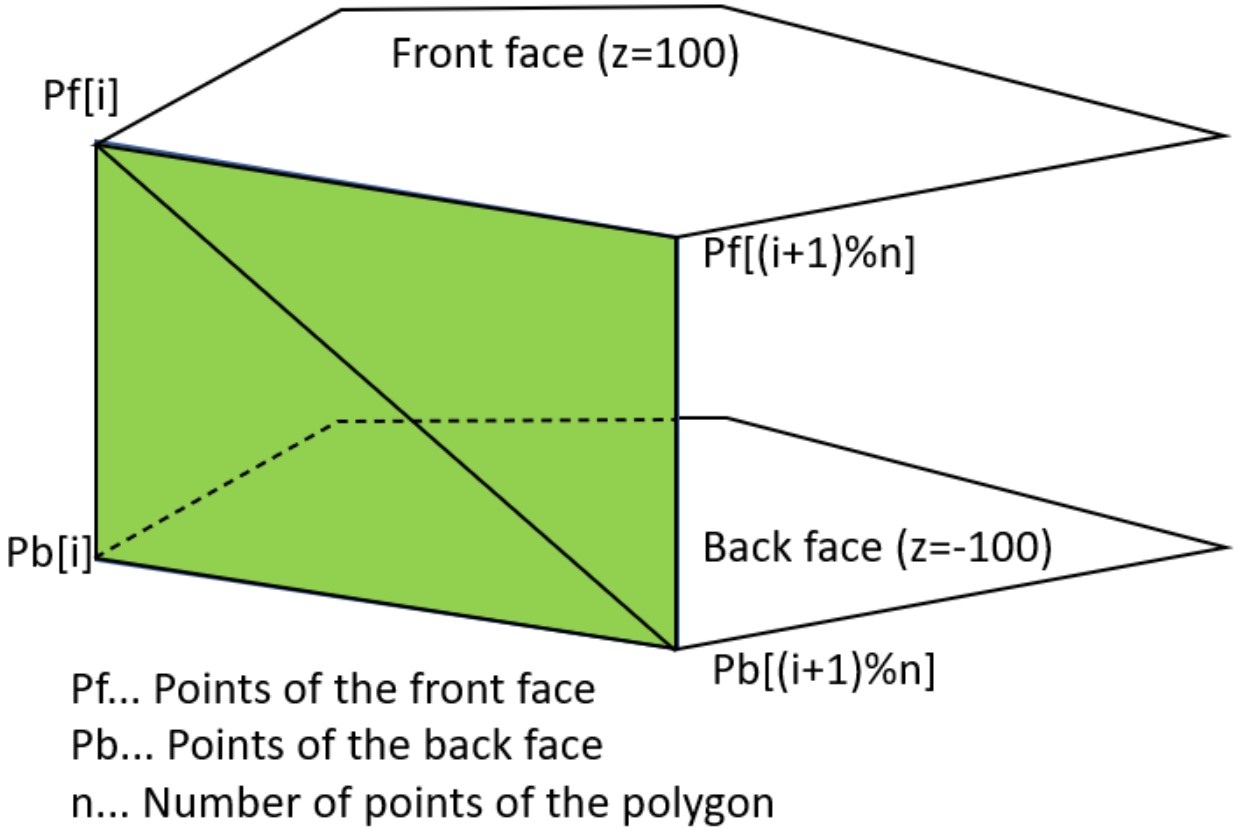

Fig. 2: Make Side Faces by Connecting Front and Back Edges

get one quadrilateral by connecting an edge of the front face and the corresponding edge of the back face. To give side faces to OpenGL as GL_TRIANGLES, you need to split a quadrilateral into two triangles by a diagonal as shown in Fig. 2.

Make all side faces green.

If you implement this function correctly, you will see a solid shape as user clicks points on the window. You need to click at least three points to see a solid. Two points will show you only a side face.

If you implement successfully, your program should look like 3. As long as the polygon can be divided into triangles by connecting center point and each edge (such as a star shape as shown in Fig. 3), it is rendered correctly.

7 Enable Lighting

Enable lighting so that you assigned a correct normal vectors. Use a headlight (light source moves with the viewer). The direction of the light should be $(0, 1/\sqrt{(2)}, 1/\sqrt{(2)}, 0)$.

8 Export it to the Binary STL

When the user presses the S key, save the shape in the binary STL. Make sure your STL can be read and rendered by a conventional STL viewer, such as ParaView, Paint3D, and also the STL viewer we made in

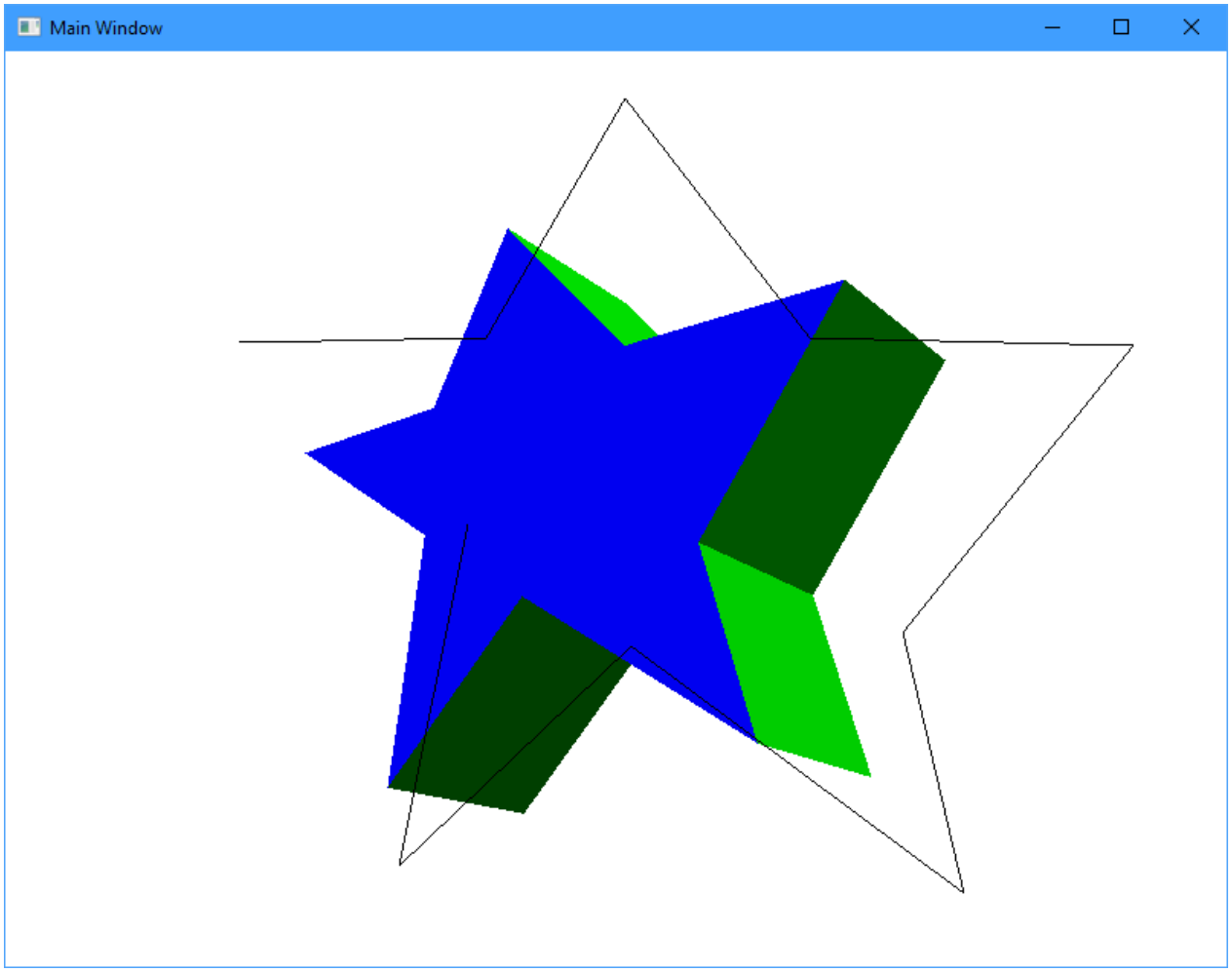

Fig. 3: Screenshot

class. (Since .STL cannot store color information, it will be rendered without color.)

9 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:24780
- http://freefood2.andrew.cmu.edu:24780
- http://freefood3.andrew.cmu.edu:24780
- http://freefood4.andrew.cmu.edu:24780

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.

10 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.

10.1 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/ps6
```

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

10.2 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 6 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.

11 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.