Live Lecture – Week #0
September 27, 2023

Where people are from:

- Korea
- Bay Area, CA
- Chicago
- Northern California, little town called Petaluma.
- Myanmar
- Redmond, OR
- Lancaster, PA
- Ridgefield, WA
- Portland
- California
- Seattle
- Vancouver, WA
- California
- Vancouver (Canada)
- Albany, OR
- Seattle
- California
- Hillsboro
- Medford, Oregon
- Riyadh, Saudi Arabia
- Washington state
- Sherwood, OR
- India
- San Juan, Puerto Rico

15:09:26 Is OpenGL hard, because I'm new to it?

Not the way we are going to approach it.

15:38:50 Will this class be hard to do on a Mac remotely?

I am not an expert on all Macs, but lots of others have done it that way.

15:41:01 How are those axes rendered?

There is some code in the sample code to do it. It draws lines in the right places. See the Axes( ) function.

15:41:24 Will OpenGL run on rabbit / dgx?

No – there is no monitor.

15:43:32 What is teach?

It is a program the College of Engineering created to make it easier to turn in assignment files: https://teach.engr.oregonstate.edu You should all have a login since you are taking a COE course. Should be the same as your ONID.

15:44:43 Will the Ed Discussion for this course be activated soon?

Yes, but I'm going to wait a couple more days so I can get everyone at once.

15:52:02 Are the OpenGL header files bundled with visual studio?

They are there, but I think they got loaded when your graphics hardware was loaded.

15:52:14 Is there a major difference for undergrad and grad students?

The difference is the paper analysis project.

16:03:53 What happens if you set an odd number of points with GL_LINES?

The last vertex just gets ignored since there was no other vertex to pair it with.

16:03:54 Triangle strip seems the most efficient in terms of calls

It is super-efficient from the number of glVertex3f calls necessary to make a certain number of triangles.

16:04:43 Topology just determines what order the vertices are traversed then filled in?

The topology determines what happens to those vertices that are specified in that order.

16:04:44 How can we make cylindrical 3D lines?

Computer graphics systems can only draw lines and filled triangles. There is no such thing as curvy surfaces, just tiny lines or triangles drawn close together. So, you would draw a circle on the top and bottom of the cylinder (see Getting Started slide #8) and then vertical lines drawn close together on what would be the sides of the cylinder. In your zip/tar folder, you have a file, osucone,cpp, that will do that for you, but with triangles.

16:08:25 And if we are using different topologies, how could we connect all of those?

You can have as many glBegin-glEnd pairs as you want. Just keep creating new glBegin calls.

16:08:29 You can make it look even smoother by calculating the right normal per pixel later in that pipeline we looked at too

That's correct. We'll talk about that in the Lighting notes. It's something called Smooth Shading.

16:11:56 Is the process for handling non-convex shapes very difficult? I can foresee that maybe happening when making art.

Like many things, it is much harder to do it from the documentation than it is by examining sample code. If you want to do this sometime, contact me and I will set you up with some working code from previous applications I've done.

16:18:24 There is also HSV for some artistic / gradient things, but you do need to convert back to RGB to feed it to OpenGL

We'll talk about this in the Color notes. Your sample code has an HSV-to-RGB conversion function in it already.

16:21:26 No new vertices are added when scaling happens and when a circle is scaled you can see the non-smooth edges?

Correct, the "circle" is still being sent into the graphics pipeline as-is. But the scaling transformation happens between where the circle is fed into the pipeline and where the pixels come out.

16:23:51 So you cannot change color between glBegin / glEnd

You actually can. When a unique color is given to each vertex and "smooth shading" is enabled, the rasterizer will interpolate the colors throughout the polygon. More information is coming in the Color notes.

16:24:56 This is the same as the order you'd write the matrix multiplication right?

If we could see the matrix operations happening, we would see that they are being "post-multiplied" so that every new transformation encountered in executing the program looks like it takes place before the transformations previously encountered.

16:28:10 You mentioned Blender, are we allowed to use Blender to create content / assets for our projects?

Yes, the easiest way is to export your Blender object as a .obj file and then use the code in the loadobjfile.cpp file.

16:28:10 Do we have to use .obj or can use other formats like .gltf?

We are giving you a LoadObjFile function. We have not written a LoadGltfFile function, and given the complexity of a gltf file, probably never will. But, you can bring a gltf file into Blender and then re-export it as a .obj file. I call that "Laundering the Geometry".

16:28:10 What is the deal with the OpenGL versions…. Open GL 2.x / 3.x / 4.x vs Open GL ES 3.x??

OpenGL desktop is up to version 4.6. Apple, with their love/hate relationship with OpenGL, froze theirs at version 2.0. OpenGL-ES (Embedded Systems) is a cut-down version of OpenGL desktop made to run on tablets and smart phones.

16:30:19 Is there any specific 'visual studio' style of setup that you'd recommend running locally for a Linux machine?

Basically no. You should be able to un-tar that tar file and type “make”. This assumes that loading your graphics driver also unloaded the OpenGL header and library files.

16:30:52 Is there a way to mirror already created surfaces across a certain plane? (i.e. duplicate a cylinder)

This is a good use for a scale factor of -1. Draw a cylinder to the right of the origin, call glScalef(-1.,1.,1.), then draw the cylinder to the right of the origin again. The negative scale factor will cause it to appear to the left of the origin.

16:32:45 For Mac users we can edit project code in visual studio code and run in g++?

Yes.

16:34:05 If you're interested in the complexity of glTF, this is the overview from the Kronos Group, who maintains a number of standards such as OpenGL, OpenGL-ES, and GLTF: https://raw.githubusercontent.com/KhronosGroup/glTF/main/specification/2.0/figures/gltfOverview-2.0.0d.png)

This is why none of us has written an import function for it. 😊

BTW, OSU is a member of the Khronos Group. I joined a few years ago to better track the standards developments so I could let the classes know the latest information.

16:36:27 How can we tell if our laptops are good for the class?

See if the sample code runs. But, if you bought it less than 5 years ago, it should be fine.

16:37:07 I couldn’t get visual studio set up for parallel this last spring

Try loading *Visual Studio 2022 Enterprise*. I loaded it this summer. I have had no problems at all, either installing it or using it. This is not my usual experience with VS.

16:39:26 From Bailey, Mike To Everyone:
BTW, if you need support help, email: **support@engr.oregonstate.edu**

16:41:22 I do love the integrated debugger in vs proper… I can just set a breakpoint and hit go

For those who are comfortable with breakpoint debuggers, definitely use them! But the print statements are better than nothing. They tell the story of your program’s execution.

16:44:30 How long are you anticipating project 1 to take?

Once the sample code is running, you could do a minimal P1 in an hour. But, as you can see from the 2022 Collage, many people have more fun with it than that!

16:48:09 Do you mind if I use VBOs for Project #1?

Not at all. If you have previous experience with OpenGL, you can use the more advanced constructs. But, I have made the promise to all of you that no previous graphics experience is required to succeed in this class, and I meant it. That's why we start with glBegin-glEnd.