# Math 155A - Introduction to Computer Graphics – Winter 2017

**Instructor: Sam Buss, Univ. of California, San Diego**

## Project #3 – Build a 3D wireframe scene.

**Due date: Friday, February 3, 9:00pm.**

**Goals:** Create a parametric surface (namely, a portion of a surface of rotation) using quad strips and triangle fans or strips. Learn how to model simple scenes with the supplied GLUT shapes such as spheres, cones, etc. Discover that wireframe objects, especially when combined with animation can look very three-dimensional. Discover, however, that flat, solidly colored objects look much too flat and non-three-dimensional.

We will use this project as a basis for the next project, which will add materials and lighting to the scene.

**What to hand in:** When you are done, place your C++ files, executable, and Visual Studio solution **together in a separate folder**, named "Project3" as a suggestion, in your PC computer account in the APM basement labs. (This should be under your networked storage so it is not subject to erasure.) The source files' creation/modification dates should before the turn in deadline. The program must compile and run on these computers. Grading will be personalized and one-on-one with a TA or with Sam Buss. Your program must run on the PC lab, you must come into the PC lab and meet one of us. As usual, you will have to show your source code, run the program, make changes on the spot to your program and recompile as requested by the grader, and be able to explain how your program works and why it renders what it does. The grading should be completed no later than Monday, February 13.

## INSTRUCTIONS:

1. **Download** the file **WireFrameProject3.zip** from the web. This zip file contains four files: (1&2) C++ source files: WireFrameScene.cpp and WireFrameScene.h, (3) an executable WireFrameSceneDemo.exe. and (4) a document about floating point roundoff. (Full URL for the zip file is: http://www.math.ucsd.edu/~sbuss/CourseWeb/Math155A_2017Winter/Project3/WireFrameProject3.zip.)
2. **Run the executable** on a PC. You will see a scene with a surface of rotation and a letter "S". (That's my initial.). Note the following commands act on the scene:
    a. Pressing the arrow keys changes the view position. The rate of movement is controlled by pressing the "**R**" or "**r**" key to increase or decrease the step size for movements.
    b. Pressing the "**w**" key toggles wire frame mode. Note how flat and non-three-dimensional the objects look when filled in. (We will fix this in the next assignment when we add material properties and lights to the scene).
    c. Pressing the "**c**" key toggles culling of back faces. Try this in both wireframe mode and in filled-in mode. Try viewing the surface of rotation from the top and the bottom while culling back faces.
    d. Pressing the "**M**" or "**m**" increases or decreases the mesh resolution on the surface of rotation.
3. Your job is to re-create this program -- sort of!!. You will re-create the surface of revolution exactly, and form something creative based on your own first initial instead of the "S". You will also supply some animation.

4. **Form a new project and solution.** Include the two source files in it. The supplied source code handles all the keystroke commands, and draws the flat base plane, and a sphere that marks the position of the surface of rotation. It has routines which you will need to rewrite to form the surface of rotation and the geometry based on your initials.
5. **Rewrite the code that generates the surface of rotation,** and render this in place of the sphere which is rendered in the downloaded source code. The surface is a surface of rotation formed from the $\cos(x)/(1+x^2)$ curve for $0 \le x \le 4\pi$, that is, x in $[0,4\pi]$. You should scale and position the surface of rotation so that it looks approximately like the one in the supplied executable. It should be about the same size, and it should be above the base plane. We will discuss in class more details on how to form this surface. Also, please see the discussion at the end of the assignment below.
   You should use quad strips and/or triangle strips and/or triangle fans to build the surface of rotation.
6. **Design some geometric shapes that are based (*loosely* based is OK) on the first initial of your first name (or last name, or even some other letter, if you prefer)**. Do something creative so that no two of you have the same geometry. Your scene should be at least as complex as the scene supplied in the sample code rendering an "S". Try to be a bit artistic and creative.
   Suggestions: use the glutSolid objects, such as spheres, cones, dodecahedra, tori, etc. to build your objects. The course web page has pointers to GLUT documentation that describes how to call these routines. The glutSolid... objects will be automatically rendered in wireframe mode when wireframe mode is activated. You may also use quad strips, triangle fans, etc. if you wish. The geometric shapes should be solid (i.e., they should not consist of points and lines).
   If you wish to use cylinders: GLUT does not supply a cylinder drawing routine, but you can find a substitute on the textbook's web page under OpenGL software.
7. **Animate some portion of your geometry**. You do **not** need use the same kind of animation as in the supplied code. In the supplied code, note that the top piece of the "S" is animated half the time and stationary half the time. You will need to call glutPostRedisplay() at the right time in order to make the animation run. (You can see where this is done in the supplied code and in the RollDisk program. You should understand what this call is achieving.)
8. **Your $(\cos r)/(1+r^2)$ surface** should be well-formed: It should not have missing quads or triangles. It should not have quads or triangles that are drawn twice. It should not have pixel-sized misalignments due to floating point roundoff error. (If you leave pixel sized holes, they will be visible when create Project #4!) For tips on avoiding floating point roundoff problems see the PDF file: Floating Point Perils: important advice on controlling your loops for the surface of revolution.
9. **The keyboard controls that toggle back face culling and wireframe mode, and also the mesh controls,** should also apply to your new geometries whenever possible. Note that the mesh count parameter can be supplied to the glutSolid… functions. **The "r" and "R" commands** should still work too.
10. Leave the rectangular base plane the same as it is in the supplied code. Your finished program should look the same as the supplied demo executable, except the "S" is replaced by your own creation and with animation of your own design.
11. **Turn in the project as described above.**

**Program grading: Scale of 0 to 10. Personal grading session with a TA or Professor Sam Buss.**

Completely optional for now: The next project (Projet #4) will add lighting and materials to the scene. If you want to get a head start on this, you may wish to add normals to your geometric objects in the scene.

**Suggestions on the (cos r)/(1+r$^2$) surface of rotation.** The best way to form this surface is to express it as a parametric function f(θ,r), where θ and r are scalar parameters. The parameter r will control the radial distance from the center, the parameter θ will determine the rotation around the center of the sombrero. The function (cos r)/(1+r$^2$) gives the height of the parametric surface, but this should be appropriately scaled to more-or-less match the shape in the demo program.

The mesh detail for your surface must be controlled by the "m" and "M" commands as in the sample program supplied with the assignment.

There are several ways to form the surface of rotation. One possibility is as quad strips plus a triangle fan. Another possibility is as radial triangle fans.

As an important reminder: do not forget about the "floating point perils" pitfalls, as discussed in the project assignment.