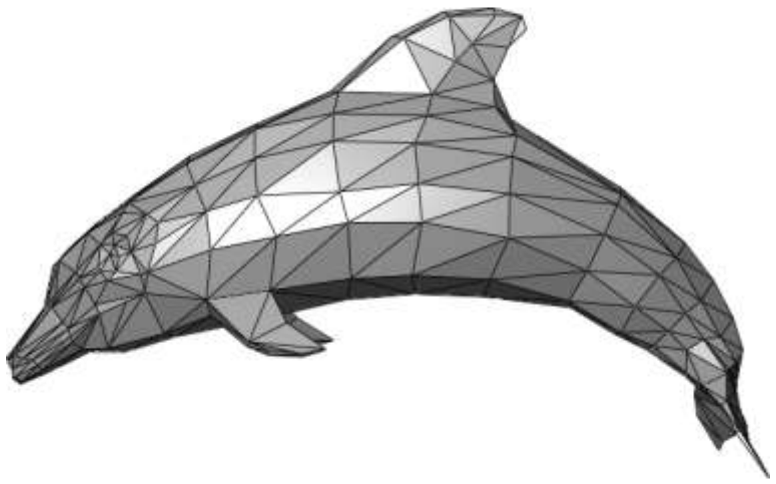
15-462 Programming Assignment 1: Mesh Rendering

Due Thursday September 11th 11:59pm

Triangle Meshes

A triangle mesh is a construct commonly used in computer graphics. It is comprised of a set of triangles that are connected by common edges. Objects in three dimensions are often modeled by approximating their surfaces using triangles.



In this assignment, you will be asked to render a 3D object based on its raw mesh model, and then allow the user to manipulate the object in three dimensions by rotating, translating, or scaling it. You will then use this program to create an animation.

Why?

This assignment is intended as a hands-on introduction to OpenGL and programming in three dimensions. The started code provided is minimal, giving only the functionality to read an IFS mesh file, write a JPEG image, and handle mouse and keyboard input. You will write the code to create a window, handle all camera transformations, perform any and all rendering, and handle any other functionality you may desire. We highly recommend the use of GLUT (The OpenGL Utility Toolkit). You may use an alternate library (FLTK, GTK, etc) if you desire, but your submission must run in the graphics cluster and we will likely be unable to provide support if you run into problems.

For information on GLUT, please refer to the OpenGL Programming Guide. If you choose not to purchase this book, information can be found online at:

- OpenGL.org
- A page of OpenGL tutors
- The on-line 'Red Book' – Recommended!

Information

In this assignment, we will only look at mesh models with an easy-to-use indexed face set format (IFS). The basic object in this mesh model format is a vertex, a point in 3D space with the origin at (0,0,0). An object in IFS format is formed by a set of triangle faces, which is defined by 3 non-identical vertices.

Implementation

For your implementation, you are given a starter code that parses an input IFS file and stores triangle meshes in a data structure for you. You will need to read the vertices of the triangles from this data structure and render the object in 3D perspective view using OpenGL.

Your program must be able to render the object in 3 different modes:

- Points
- Wireframe
- Solid Triangles

While rendering in solid triangle mode, your program should use OpenGL's depth buffer for hidden surface removal.

Light Source: An object modeled using a simple IFS model does not contain animation or texture data (i.e. color, shininess, etc). Displaying the object with a single color would make the image look dull. You are required to add one or more light sources into your scene so the image can display the variation on its surface.

You are also required to add transformation into your object rendering using OpenGL library. A user should be able to move, rotate and resize your object using their mouse.

Starter Code: The starter code in C, a Makefile, and examples are located here. This should work on the graphics cluster (64-bit version). Please let us know if you find any problems with the starter code. An older version of the starter code which works on 32-bit machines can be found here. You're final code however, should work on the graphics cluster.

Note: The recommended language for this assignment is C. You may use another language if you wish, but your program must compile and run in the graphics cluster OR on the TA's computer although that may be difficult to ensure. Implementing your project with the OpenGL library is strongly recommended as it will prepare you for later assignments.

We have provided a sample executable file to aid your understanding of what your project could resemble.

- Linux executable
- We will try having a windows executable up soon

Models: The Brown Mesh Set is a library of 3D models which will be useful for testing your program. You can also create your own IFS format using a 3D modeling engine such as G3D.

Requirements

- Handle any IFS mesh data in BMS set for your mesh object at interactive frame rates. Make sure you test plenty of models as we will test your program on more complicated models as well.
- Be able to render the mesh object as points, wireframe, or solid triangles.
- Users must be able to switch between the three modes.
- Render as a perspective view, utilizing GL's depth buffer for hidden surface removal.
- Use input from mouse to spin the mesh object using glRotate.
- Use input from mouse to move the mesh object using glTranslate.
- Use input from mouse to change the dimensions of the mesh using glScale.
- Add one or more light source using OpenGL such that the depth of object surface is clearly seen.
- Add some material and/or lighting properties to your model. Allowing the user to switch them on/off.
- The code must be reasonably commented and written in an easily understandable manner
- Submit a writeup.txt file which summarizes the features your implemented. The format for this file is included in the starter code package.
- Fulfill the animation requirement.

Animation Requirement

After finishing the program, you are required to submit an animation, represented by a series of TIFF images which are screenshots from your program. Functionality to output a screenshot is included in the starter code and assumes that you are using a window size of 640x480. Please name your TIFF frames 00.tiff, 01.tiff and so on, where 00.tiff is the first frame of your animation. Please do not exceed 100 frames. Expect a frame rate of 15 frames per second. It's probably a good idea to test your animation using animate ("man animate" for info) in the graphics cluster. This program takes a sequence of images and displays them as an animation at your desired framerate.

The method of generating your frames is left up to you – there is a large amount of room for creativity. You may use any software you wish to generate the source image for your mesh objects. The GIMP is available on graphics cluster machines (and for download for most other systems). You may also use your animation to show any extra features you choose to implement.

Your animation will receive credit based on its artistic content, whether pretty, funny, or just interesting in some manner.

Submission

Please submit your code along with your Makefile to your hand-in directory. Your handin folder is at '/afs/cs.cmu.edu/academics/class/15462-s08-users/<your_andrew_id>/assn1'. The folder has been already created for you. When we run "make" in this directory it should compile your program successfully. It is extremely important that you ensure that running "make" in your directory in the graphics cluster compiles the program without error. Within this directory, make a subdirectory called movie, and place the frames for your animation within, numbered as described above.

Right after the assignment deadline, the hand-in directory will be locked. Once the submission has been backed up, it will be reopened for late submission.

If you experience any difficulty with the hand-in directory, please email a tar.gz or tar.bz2 of your submission to dhirenb@andrew.cmu.edu. The time of submission will be determined by the time the email arrives at the final mail server.

For scp/sftp users: since the handin directory resides on SCS AFS network, your username on the directory is actually "<username>@ANDREW.CMU.EDU", not just "<username>". To transfer your submission to the handin directory please scp/sftp as "<username>@ANDREW.CMU.EDU" via weh5336-?.intro.cs.cmu.edu.

Grading

The grading of this assignment will roughly be divided as follows:

- Camera and Transformation: 35%
- Object rendering and coloring: 40%
- Animation and programming style: 25%

Tips

- Familiarize yourself with GL's viewing transformations before attempting to render the object itself.
- Try to render a simple object first.
- Finish your program completely before worrying about the animation.
- You can change the structure of the program. In fact, it is recommended that you divide your program into manageable files instead of maintaining all the code in starter.c.
- If you are having trouble with Makefiles, Google is a great resource. You are permitted to search for and use sample Makefiles.
- Don't try to do this at the last minute. This assignment is relatively easy, but it could take a while to get used to this style of programming.
- **Make sure you can log in right away**. Notify us immediately if you have any problems.

Extras

You may choose to implement any combination of the following for extra credit (maximum grade is 110%). You can also come up with your own extra features (Highly recommended!)

- Experiment with advanced material/lighting properties.
- Texturemap the surface with an arbitrary image.
- Allow the user to interactively change the object appearance by moving vertices.
- Level of detail (mip mapping/trilinear) to allow mesh object of significantly higher resolution.