Computer Graphics

Theme 3 – Milestone 2

Due Sunday, October 23, 2011 at 11:59:59pm

Triangle Meshes and OBJ files

1. Introduction

Recall from the light milestone that you learned how to implement different shading and lighting models. This led to the first truly 3D graphics in the course, when you could sense depth in a 2D image and learned how to specify different lighting component in the geometry you defined. In this milestone you will use all that knowledge, in conjunction with what you learned of Triangle Meshes, to load a file in OBJ format[1]. After this milestone you will be able to see complex geometric figures in your computer screen.

2. Academic Honesty Policy

You are permitted and encouraged to discuss your work with other students. You may work out equations in writing on paper or a whiteboard. You are encouraged to use the Wiki bulletin board to converse with other students, the TA, and the instructor.
HOWEVER, you may NOT share source code or hardcopies of source code. Refrain from activities or the sharing materials that could cause your source code to APPEAR TO BE similar to another student's enrolled in this or previous years. We will be monitoring source code for individuality. Cheating will be dealt with severely. Cheaters will be punished. Source code should be yours and yours only. Do not cheat.

3. Grading and Lateness

Assignments will be divided into different themes. Each theme will be divided into milestones. Each milestone will count equally towards your final grade.

Late submissions lose 1% per six minutes of lateness. For example: a submission that is two hours late is penalized 20%, and a submission that is ten hours late receives no credit.
Plan ahead. The only exception to this policy is a documented medical emergency. In order to ensure fair grading, exceptions are not possible for holidays, sport meets, theater appearances, indigestion, etc. Plan ahead. If you believe that you have a just cause for submitting a late assignment in a non-medical-emergency circumstance, please obtain written permission from the instructor or TA at least one week prior to the assignment deadline. Plan ahead.

[1] See the Wikipedia entry http://en.wikipedia.org/wiki/Wavefront_.obj_file for and introduction on the OBJ file format.

4. Starter Code

No starter code will be provided for this milestone. Instead, another student's reviewed solution to the previous milestone will be posted. You can use this as a starting point, but make sure you acknowledge this fact in your README file. Make sure you follow the submission instructions carefully. You will implement this milestone in C++ using the OpenGL API. Information about OpenGL is widely available online. Some recommended resources include http://www.openglbook.com and http://www.opengl.org, OpenGL Superbible (available online through CLIO), and the "Red" book: http://glprogramming.com/red/.

Additional information regarding OBJ files can be found in Wikipedia or any other place on the internet. The official OBJ file format specification can be found at http://netghost.narod.ru/gff/vendspec/waveobj/obj_spec.txt . OBJ files can be downloaded from a number of different websites. One sample OBJ file will be posted in the Wiki, but you might want to load other models. A good survey of websites that offer 3D models is available at http://www.hongkiat.com/blog/60-excellent-free-3d-model-websites/.

5. Due Dates and Program Submission

This assignment is due on Sunday, October 23, 2011, at 11:59:59pm eastern time. Please submit your solution to the TA as a zip file using the online submission system available at http://www.cs.columbia.edu/~au2158/ . The zip file should be named "<your_uni>_t03m02.zip" (For example: *au2158_t03m02.zip, IMPORTANT: make sure you submit a zip file*). Include a README file in your submission listing all the files included, and extra features you wish to explain. You should also include a Makefile to compile your code and make sure your code compiles in the CLIC machines. Your Makefile should compile your code by typing:

```
make
```

into an executable called main_t03m01.

If you have any questions about the submission process, please speak with the TA during office hours BEFORE the milestone is due. Further, please allow 24 hours for responses from the TA via email or the course Wiki.

6. Required Features

a. (40 Points) OBJ file reader

Implement an OBJ file reader. Your program should read the filename specified by the first program argument as follows

```
main_t03m02 <OBJ_filename>
```

It will then use the specified file to read the vertex and face information. You have to support the following types of data: vertex (v), vertex normal s(vn), and faces (f). You should read the file and create a HE mesh object (See 6.b), then query the mesh to have a list of all triangles and use this list to render it in OpenGL. Since OBJ files can have thousands of triangles, and since the model will have to be rendered from all four viewports, you should use display lists. To learn more about display lists, read chapter 7 of the "Red" book: http://glprogramming.com/red/chapter07.html.

Keep in mind that an OBJ face can have more than 3 vertices associated with it, describing a polygonal face instead of a triangle face. If this happens, you have to split the polygon into triangles. For example, if an OBJ line reads (see Figure 1):

```
f 4 6 2 10 11
```

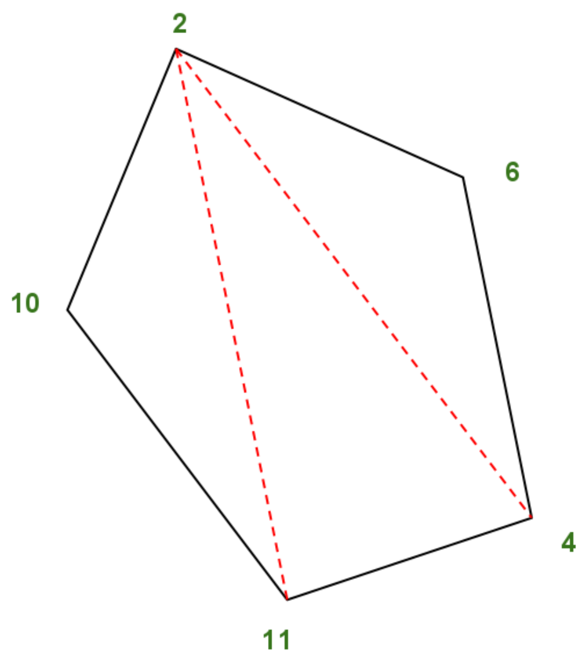

Figure 1 – Splitting a polygon into smaller triangles

You will have to create three triangles for this face. One possibility would be triangles (2, 10, 11), (2, 11, 4), and (2, 4, 6).

b. (60 Points) Half-edge Triangle Mesh

Implement a half-edge (HE) triangle mesh data structure that supports the following query operations. You should parse the program arguments, and if you detect that a query is being requested, then do NOT call OpenGl, only output the results of the query:

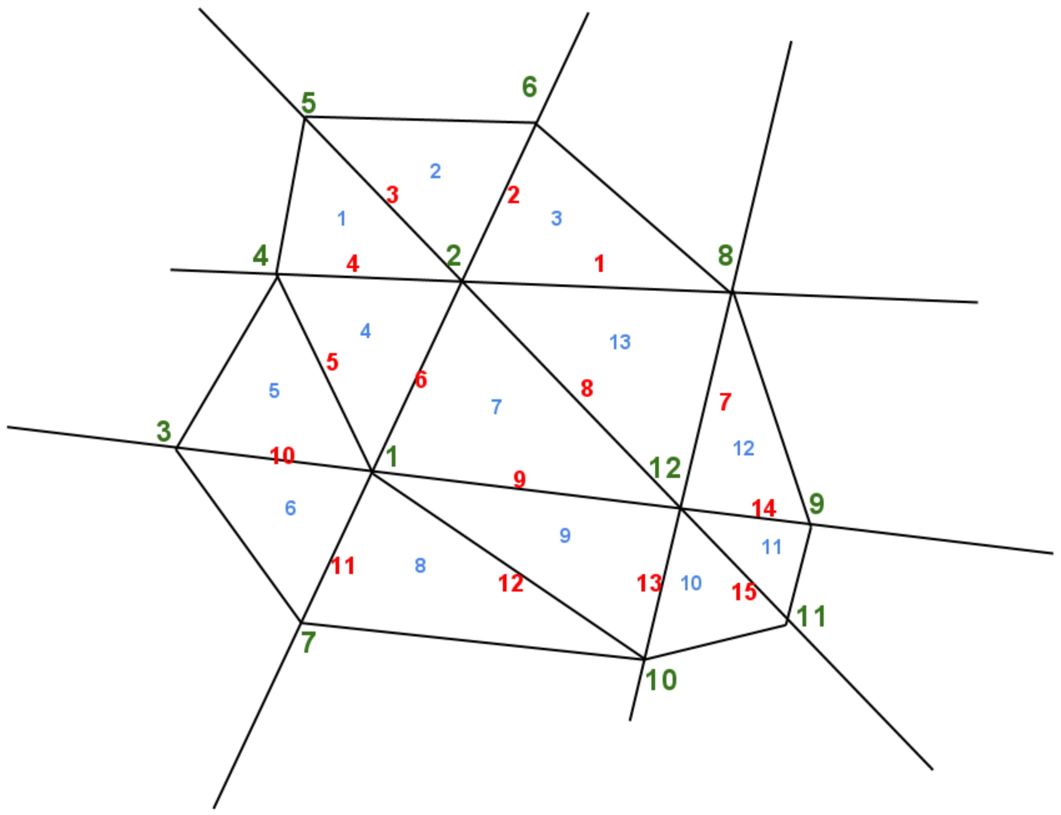

Figure 2 – Triangle mesh with triangle, edge and vertex indices in blue, red and green respectively.

EdgesOfVertex

```
main_t03m02 <OBJ_filename> -ev <vertex_index>
```

Should be executed by your program when the –ev flag is given. Following the flag, one vertex index is specified. Your program output should be:

```
<vertex_index>

N

<edge1_endpoint> <edge2_endpoint> … <edgeN_endpoint>
```

Where N is the number of edges, and <edgeX_endpoint> are the vertex indices of the edge endpoints different than the specified vertex index. The latter can be in any order. For example, referring to Figure 2, the output of

```
main_t03m02 <OBJ_filename> -ev 2
```

Should be:

```
2

6
```

```
1 4 5 6 8 12
```

<u>EdgesOfTriangle</u>

```
main_t03m02 <OBJ_filename> -et <triangle_index>
```

Should be executed by your program when the –et flag is given. Following the flag, one triangle index is specified. Your program output should be:

```
<triangle_index>
<edge1_endpoint> <edge2_endpoint> <edge3_endpoint>
```

Where <edgeX_endpoint> are the vertex indices of the edge endpoints that make up the triangle in counterclockwise order. These can be in any cyclic permutation. For example, referring to Figure 2, the output of

```
main_t03m02 <OBJ_filename> -et 9
```

Should be:

```
9
12 1 10
```

<u>TrianglesOfTriangle</u>

```
main_t03m02 <OBJ_filename> -tt <triangle_index>
```

Should be executed by your program when the –tt flag is given. Following the flag, one triangle index is specified. Your program output should be:

```
<triangle_index>
<triangle1_index> <triangle2_index> <triangle3_index>
```

Where <triangleX_index> are the triangle indices of the adjacent triangles to the specified triangle, in counterclockwise order. You can output any cyclic permutation. For example, referring to Figure 2, the output of

```
main_t03m02 <OBJ_filename> -tt 9
```

Should be:

```
9
7 8 10
```

Make sure that your code does exactly what is asked, otherwise you will be given 0 points for this feature.

7. Optional Features

This milestone will have optional features that are meant to be food for thought. If you are interested in driving your skills to the limit, you might consider implementing the following features. Naturally, if you decide to go for it, this means you will have more 'ingredients' at your disposal to create an extremely beautiful creative scene.

a. (Optional) MTL file integration

If you feel you want to do even more, you can opt for including material definitions using the MTL file format. This is a separate file that specifies which materials to use in the OBJ file read. You should make this file callable by your program through the console in the following way:

```
main_t03m02 <OBJ_filename> <MTL_filename>
```

If you decide to do this keep in mind that you will be required to extend your OBJ reader to support at least group names (g) and material names (usemtl).

8. Bonus (15 points) Creative Submission

As part of this assignment you will have the option of submitting an OpenGL code snippet that creates an image/animation/interactive app where you can show what your program can do. This submission will count towards the final grade of your assignment. A committee will judge your image on the following criteria.

1. How well the scene shows off this milestone's 'ingredients' (a la *Iron Chef)*
2. Aesthetic considerations. The more beautiful, the better.
3. Originality.

Top examples will be posted to the course wiki, as well as demoed during class by their creators.