Middle East Technical University | Department of Computer Engineering

CENG 477

Introduction to Computer Graphics

Fall '2012-2013

Assignment 3 - Texture Mapping and Shadows in OpenGL

Due date: 13.01.2013 23:55

1 Objective

In this assignment, you are expected to implement shadows and texture mapping using OpenGL. You will construct a scene which includes models with textures and implement any preferred shadow algorithm. You will be provided a file which describes the scene properties such as the coordinates and model properties of the box, coordinates of the models, models texture mappings, position and intensity parameters of the light source etc. You are expected to correctly render the scene with texture mappings for the models and shadows. Also, you are expected to implement interactions such as the movement of the light source and rotation of the models via keyboard.

2 Specifications

1. **File Format**

   The input file describing the scene will be in the following format. Parts included in brackets will not be included in the input file, they are just for descriptional purposes:

   #Light
   f1 f2 f3 f4 [Light position in OpenGL as 4 floats]
   f1 f2 f3 f4 [Lights ambient coefficent in OpenGL as 4 floats]
   f1 f2 f3 f4 [Lights diffuse coefficent in OpenGL as 4 floats]
   f1 f2 f3 f4 [Lights specular coefficent in OpenGL as 4 floats]
   #Material
   f1 f2 f3 f3 [Material ambient coefficent in OpenGL as 4 floats]
   f1 f2 f3 f4 [Material diffuse coefficent in OpenGL as 4 floats]
   f1 f2 f3 f4 [Material specular coefficent in OpenGL as 4 floats]
   #Box [Vertex coordinates for the box as floats]
   x1 y1 z1 [Coordinates of the first vertex]
   x2 y2 z2 [Coordinates of the second vertex]
   . .
   . .
   x8 y8 z8 [Coordinates of the eighth vertex]
   a1 b1 c1 [Vertex indices for the first polygon as integer]
   a2 b2 c2 [Vertex indices for the second polygon as integer]

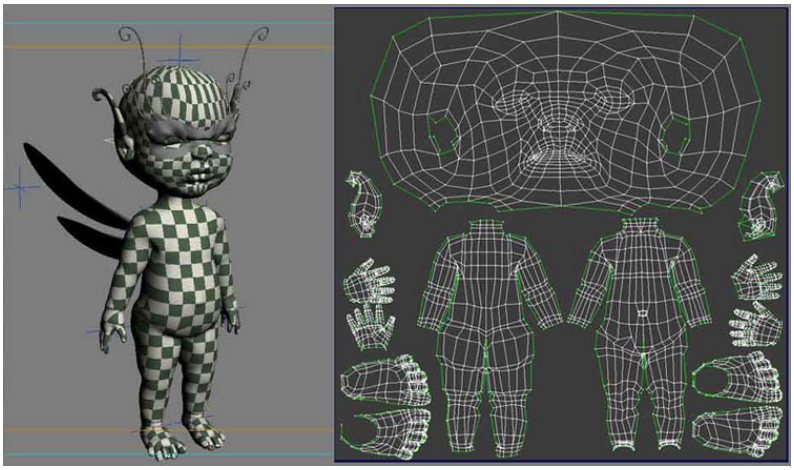
. .
. .
a12 b12 c12 [Vertex indices for the twelfth polygon as integer]
#Mesh
string [Path to the meshes texture file as string]
x1 y1 z1 [Transformation for the mesh as float]
x1 y1 z1 [Rotation for the mesh as float]
x1 y1 z1 [Scale coefficient for the mesh as float]
d1 [Number of vertices in the mesh as integer]
x1 y1 z1 [Coordinates of the first vertex as float]
x2 y2 z2 [Coordinates of the second vertex as float]
. .
. .
. .
d2 [Number of texture coordinates for vertices as float]
u1 v1 [Texture u-v coordinates for the first vertex]
u2 v2 [Texture u-v coordinates for the second vertex]
. .
. .
. .
d3 [Number of faces(polygons) in the model]
a1 b1 c1 [Vertex indices for the first polygon as integer]
a2 b2 c2 [Vertex indices for the second polygon as integer]
. .
. .
. .
#Mesh
. .
. .
. .

- There will be only one light source in the scene and its properties such as position and intensities are given below the #Light text. You are also expected to visualize the light source as a small sphere with respect to its position. The color and size of the sphere is up to you.
- Material properties will be used for the box. There will be only one type of material and its properties are given below #Material text.
- You will use triangles to visualize the box. Under the #Box text, coordinates for the 8 vertices are given first. Then vertex indices (a,b,c) which form a polygon are given. Since there will be 12 polygons in the box, there will 12 line entries in this section. You should not worry about the correct ordering, indices will be given counter-clockwise.
- There could be more than one object in the scene and their properties are given under #Mesh sections. So expect to see more than one #Mesh entry in the input.
- There are different sections for the #Mesh entries. The first line specifies the path of the texture file for the mesh. This path is not absolute, so you could always assume that the necessary file will be present in the same folder with your executable. Following three lines specifies the transformation, rotation and scaling for the respected mesh. In the fifth section, the vertex coordinates(x,y,z) for the mesh is given. The sixth section gives the texture u-v coordinates for the vertices. The sorting here is same as the fifth section. So the first entry in

the fifth section and the first entry in the sixth section corresponds to the same point. In the last section, the vertex indices for the polygons (a,b,c) are given. An entry such as [1,4,3] in this section means that vertices 1,4 and 3 form a polygon.

- You should not transform the camera or change where it points. The scene will be given such that without any camera transformations, models and part of the box will be observeable. Don't forget to do perspective calculations though!

2. **Texture Mapping**
   Texture mapping is the process of mapping a texture(an image) to the surface of a polygon. Every vertex in the polygon is assigned a texture coordinate. These texture coordinates are also known as u-v coordinates. To compute these coordinates, a method called unwrapping is used. You could think of this as converting a 3D model into a 2D image. Then a 2D image specifying the texture is matched with the 2D representation of the model and for each vertex, its u-v coordinates are computed. These u-v coordinates are between 0.0 and 1.0, where the lower left corner is specified as (0.0,0.0) and upper left corner as (1.0,1.0). You will not handle u-v coordinate computation in this homework. You will be provided with uv coordinates for each vertex. You will read the specified texture file using OpenGL and when rendering the object you will use the uv coordinates to correctly visualize your model. A simple example of unwrapping is depicted in the image below:



3. **Shadows**

   The other specification of this homework is to implement shadows. You will render your scene and illuminate it correctly using the OpenGL functions. To make the scene more realistic, you should also implement shadows. There is no restriction on the method of creating and visualizing shadows but you are strongly advised to use shadow mapping or shadow volume algorithms. Your implementation should create correct shadows regarding the position of the light source and objects. So, we would like you to correctly cast shadows of the loaded models onto the box(and onto other models).

4. **Keyboard Interaction**

   The created scene will be interactive. Using keyboard input, the position of the light source and the rotation of the models could be changed. Predefined inputs are as follows:

   - 'l L' : Move the light source in the +x direction.

- 'k K' : Move the light source in the -x direction.
- 'i I' : Move the light source in the +y direction.
- 'j J' : Move the light source in the -y direction.
- 'o O' : Move the light source in the +z direction.
- 'u U' : Move the light source in the -z direction.
- 'Up arrow' : Rotate the models in the +x direction.
- 'Down arrow' : Rotate the models in the -x direction.
- 'Left arrow' : Rotate the models in the +y direction.
- 'Right arrow' : Rotate the models in the -y direction.

The movement and rotation speed is up to you but reasonable speeds are preferrable. You should rotate all the models present in the scene when the corresponding keyboard input is given.

3 Notes

- You should use C++ and OpenGL library.
- We will test your codes on departmental machines using "g++". Please make sure to run tests on ineks.
- A sample run of your program will be "$ ./hw3 scene_file.txt".

4 Submission

Submission will be done via COW. You should upload a single zipped file called "hw3.zip", which includes your source code and a makefile. The executable output of your program is called hw3.

Warning: Late submissions are allowed for this homework, based on the policy on the course web site.

5 Grading

Grading will be made using the scala on the course's website.

6 Cheating Policy

We have zero tolerance policy for cheating. People involved in cheating will be punished according to the university regulations. See the course website for more information.