3D Model Viewer – Test Document

CSC 478 - Spring 2014 - Group 1 Choppers

Andrew Baker

Adam Le

Todd Previte

# Testing Documentation

## Black-box Testing

### Test Case # 101 – Load and Render 3D model

**Requirements Tested**

3.1.1

**Rationale**

After Selecting a .OBJ file a 3D model of the object should be loaded onto the screen

**Steps**

1. Press the “Open File” button
2. A file browser window opens, select the desired .OBJ file
3. After selecting the file the object is rendered in the content window.

**Expected Output**

Pressing the “Open File” button will bring up the file browser, after selecting a .OBJ file the model is rendered on the screen

**Outcome** **PASS or FAIL**

**Notes**

### Test Case #102 – Rotate 3D Model

**Requirements Tested**

3.1.6

**Rationale**

Click the object and moving the mouse rotates the 3D model accordingly

**Steps**

1. Click the 3D Model and move the mouse to the left
2. Click the 3D Model and move the mouse to the Right

**Expected Output**

The 3D model should rotate right and rotate left in accordance the direction the mouse is moved.

**Outcome** **PASS or FAIL**

**Notes**

### Test Case #103 – Zoom

**Requirements Tested**

3.1.7

**Rationale**

The scroll wheel provided zoom functionality for the 3D model

**Steps**

1. Click on the 3D model and scroll the wheel forward.
2. Click on the 3D model and scroll the wheel backward.

**Expected Output**

Scrolling the wheel forward should zoom in on the 3D model. Scrolling the wheel backward should zoom out on the 3D model.

**Outcome** **PASS or FAIL**

**Notes**

### Test Case #104 – 3D Model Popup Button

**Requirements Tested**

3.1.8

**Rationale**

Sample 3D models is provided to show application functionality

**Steps**

1. Click on the “3D Model” popup button.
2. Click on one of the provided 3D models.

**Expected Output**

After clicking on a 3D model name, the model should be rendered in the content view.

**Outcome** **PASS or FAIL**

**Notes**

### Test Case #105 – Lighting

**Requirements Tested**

3.1.2

**Rationale**

Lighting should be enabled if there is lighting information

**Steps**

1. Click on the “Lighting Enable” popup button.
2. Click on “Disable GL\_LIGHT0”
3. Click on “Enable GL\_LIGHT0”

**Expected Output**

When “Disable GL\_LIGHT0” is selected the object will be rendered with rainbow colored lines because there is no lighting. When “Enable GL\_LIGHT0” is selected the object will be rendered in black lines. White lines will be used on where the lighting is set on the object.

**Outcome** **PASS or FAIL**

**Notes**

### Test Case #106 – Shading

**Requirements Tested**

3.1.3

**Rationale**

Two shading formats provided are flat and smooth.

**Steps**

1. Click on the “Shading Model” popup button.
2. Click on “Flat Shading”
3. Click on “Smooth Shading”

**Expected Output**

When “Flat Shading” is selected the object’s polygons will be shaded. When “Smooth Shading” is selected the color changes from pixel to pixel

**Outcome** **PASS or FAIL**

**Notes**

### Test Case #107 – Rendering mode

**Requirements Tested**

3.1.4

**Rationale**

Two rendering modes are provided wireframe and solid.

**Steps**

1. Click on the “Rendering Mode” popup button.
2. Click on “Wireframe Mode”
3. Click on “Solid Mode”

**Expected Output**

When “Wireframe Mode” is selected the object’s surface polygons will not be filled in allowing transparency. When “Solid Mode” is selected the object’s surface polygons will be filled in **not** allowing transparency.

**Outcome** **PASS or FAIL**

**Notes**

### Test Case #108 – Surface Normal

**Requirements Tested**

3.1.5

**Rationale**

Surface Normals are provided

**Steps**

1. Check the “Show Surface Normals” check box.
2. Un-check the “Show Surface Normals” check box.

**Expected Output**

When “Show Surface Normals” is checked the normals on the surface of the 3D model will be drawn. When “Show Surface Normals” is unchecked no normals will be displayed.

**Outcome** **PASS or FAIL**

**Notes**

## White-box Testing

### Test Case #109 Check Parser/Loader algorithm

**Requirements Tested**

3.2.1 and 3.2.2

**Rationale**

Check that the data loaded and parsed matches the original file

**Steps**

1. Under the Xcode developer environment add a break point.
2. Run the application inside of Xcode
3. Compare the data stored inside the data structure to the data stored in the 3D model file.

**Expected Output**

The data should match the data stored in the 3D model file. There should not be missing entries. Every line that is required to render the 3D model should be inside the data structure.

**Outcome** **PASS or FAIL**

**Notes**