Student ID: 27033004
Name: Jonathan Edwards

COSC422 Assignment 2 Report

The Scene

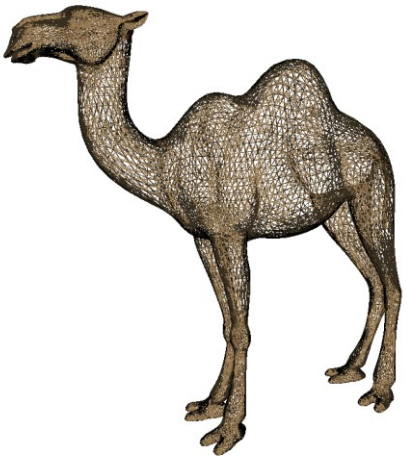


Figure 1 - Wire mode + Zoom

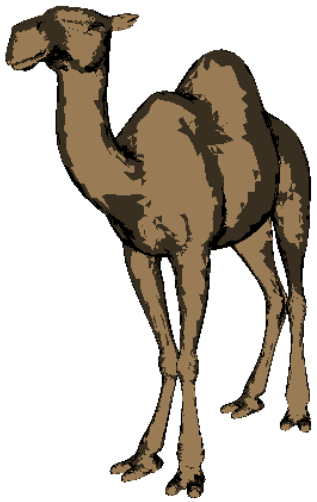


Figure 2 - Two-tone shading - view 1

Figure 3 - Two-tone shading - view 2

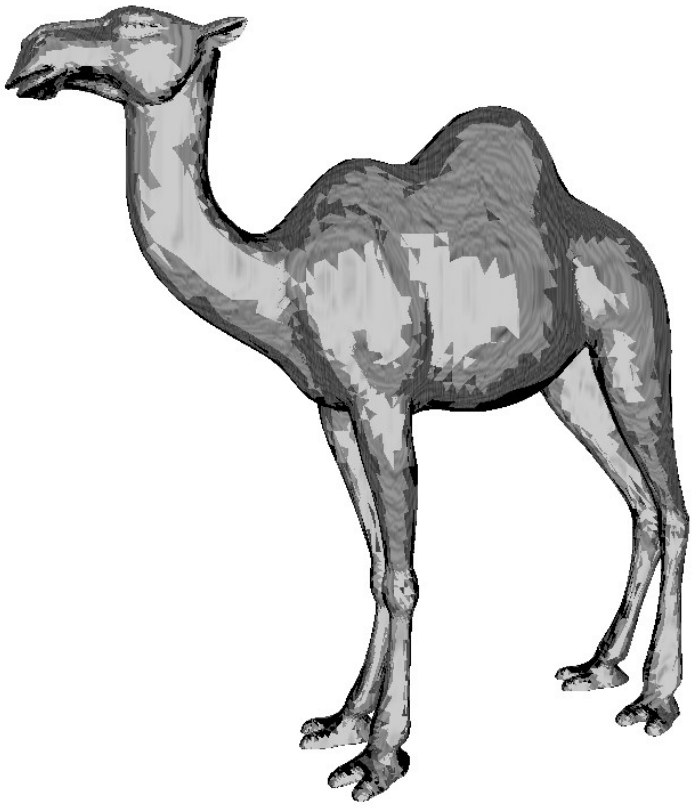


Figure 4 - Pencil shading - 3 textures

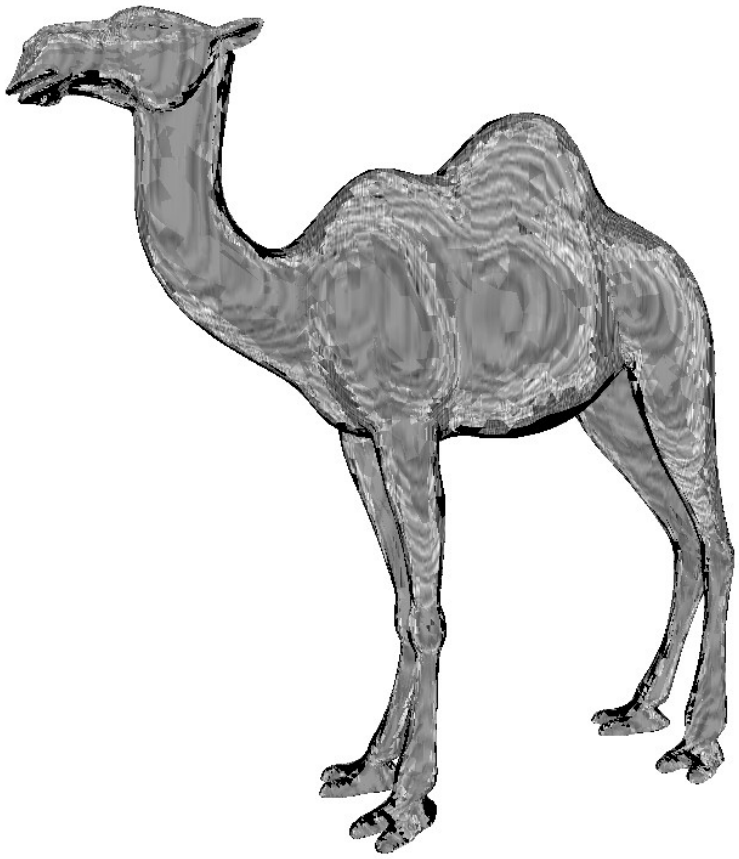


Figure 5 - Pencil shading - 5 textures & blending

Student ID: 27033004
Name: Jonathan Edwards

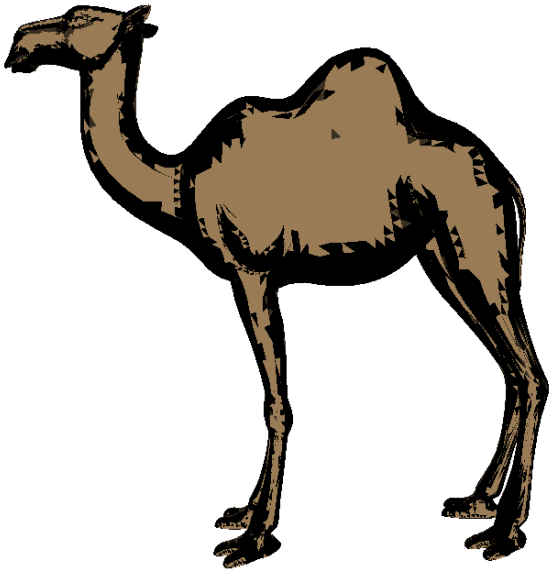


Figure 6 - Edges - thick

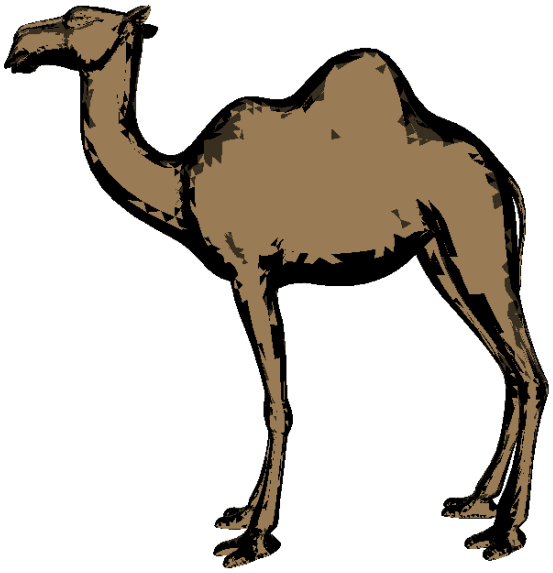


Figure 7 - Edges - thin

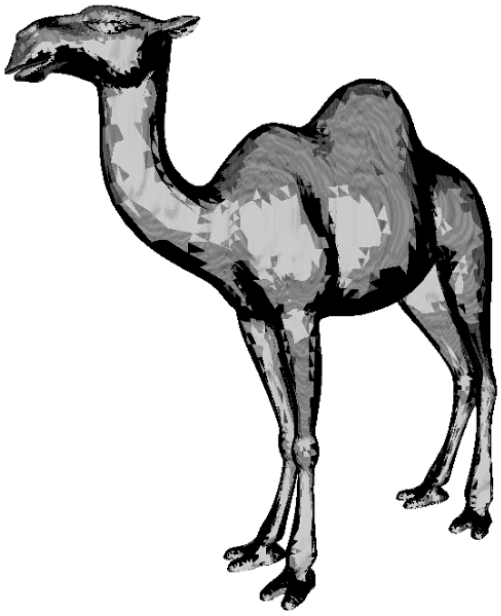


Figure 8 - Edges - Pencil shading

Basic NPR

1. **Two-tone shading:**

Figure 1 shows the model being in a wireframe mode.

Figure 2 & 3 shows Two-tone shading. i.e there are only two colours outputted.

2. **Pencil shading:**

Figure 4 shows the model shaded with 3 different textures.

Student ID: 27033004
Name: Jonathan Edwards

Figure 5 shows the model shaded with 5 different textures and blended using textured weights.

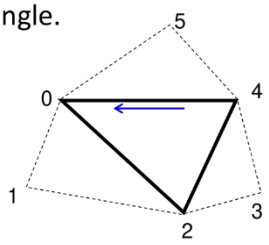
To load the textures, the devil library is used.

3. Geometry shader:

Modified Algorithm to be able to use triangle adjacency in the Geometry shader.

Using **triangles**_adjacency primitive type instead of **triangles.**

iangle.

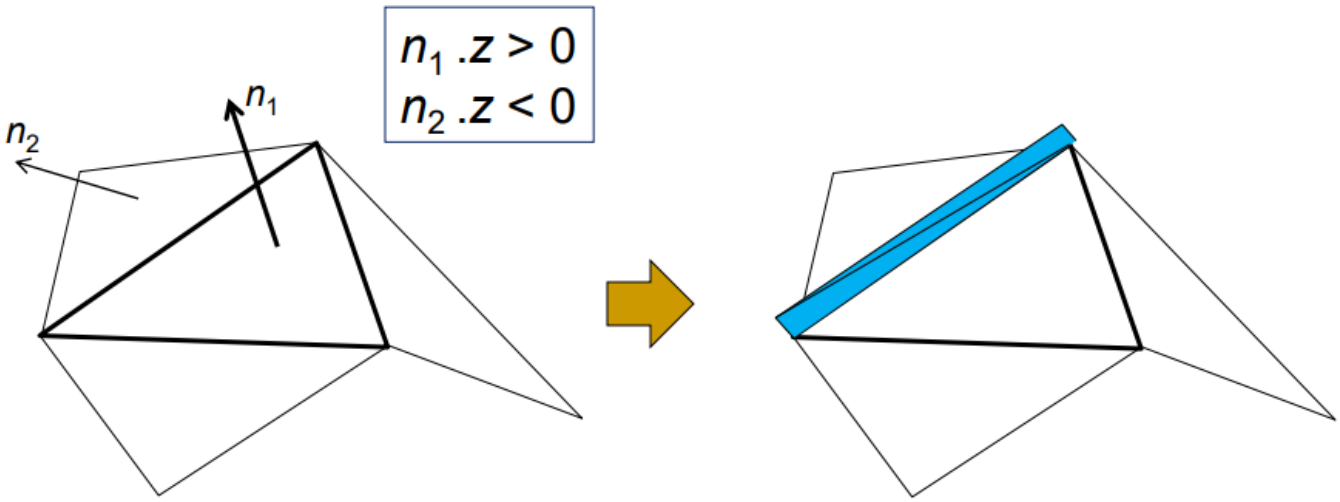


triangles_adjacency primitive now has 6 elements.

```
indx = 0;
for (fit = mesh.faces_begin(); fit != mesh.faces_end(); fit++)
{
    facH = *fit;

    OpenMesh::HalfedgeHandle heH = mesh.halfedge_handle(facH); // one of the edges of the current tri.

    verH2 = mesh.from_vertex_handle(heH);  // Vertex A
    elems[indx] = verH2.idx();
    indx++;

    verH2 = mesh.opposite_he_opposite_vh(heH);    // Vertex D1
    elems[indx] = verH2.idx();
    indx++;

    heH = mesh.next_halfedge_handle(heH);

    verH2 = mesh.from_vertex_handle(heH);  // Vertex B
    elems[indx] = verH2.idx();
    indx++;

    verH2 = mesh.opposite_he_opposite_vh(heH);    // Vertex D2
    elems[indx] = verH2.idx();
    indx++;

    heH = mesh.next_halfedge_handle(heH);

    verH2 = mesh.from_vertex_handle(heH);  // Vertex C
    elems[indx] = verH2.idx();
    indx++;

    verH2 = mesh.opposite_he_opposite_vh(heH);    // Vertex D3
    elems[indx] = verH2.idx();
    indx++;
}  // GL_TRIANGLES_ADJACENCY - V3
```

4. Silhouette and crease edges

Comparing the centre face normal with the others. This is illustrated in the diagram below and the code snippet.




```
// -- Silhouette Edge Identification Levelling ------------------------------
if (normals[1] < thickness_silhouette && normals[0] > 0)
    outColor = vec4(0.0);
else if (normals[2] < thickness_silhouette && normals[0] > 0)
    outColor = vec4(0.0);
else if (normals[3] < thickness_silhouette && normals[0] > 0)
    outColor = vec4(0.0);
// ------------------------------------------------------------------------
```

Student ID: 27033004
Name: Jonathan Edwards

If there is a gap, fill it with the color black. Varying levels using keys described below.

Extra features

1. Texture Weighting & Blending

Blending the two textures using an arbitrary weight “t” that corresponds to the diffTerm.

```
else if (diffTerm >= 0.85 && diffTerm < 0.95)
{
    t = (diffTerm-0.7) / 0.1;
    col1 = texture(pencil_stroke[2], tex_coords);
    col2 = texture(pencil_stroke[1], tex_coords);
    outColor = vec4(0.50) * ((1-t) * col1 + t * col2);
}
```

Control Functions

↑ Rotate Under	'q' Increases thicknesses of silhouette & crease edges
↓ Rotate Over	'a' Decreases thicknesses of silhouette & crease edges
← Rotate Right	't' Toggle texture weighting & blending
→ Rotate Left	'f' Wireframe mode
Spacebar Toggles between Two-tone & Pencil shading	'ESC' Exit application
Page up Zoom in	
Page Down Zoom out	

Compiling & Running Application

On a Linux Machine, on the Makefile level, type: ‘make && make program’

References to Sources

- All Models and Textures used were from the lab and lectures.