Universitat de Girona

Illumination with external OBJ

Gonzalo Besuievsky

IMAE - UdG

Universitat de Girona

Contents

- Including 3D OBJ with illumination
- Multiple Lights
- Materials

Universitat de Girona

JSON-based formats

- Use JSON-based formats
- Pretty simple to parse:
  JSON.parse(responseText)

– More info: http://json.org/

– Steps:

  - Convert OBJ to JSON
    – Same steps as previous Assignment
  - Parse vertex, normal, textures and indices

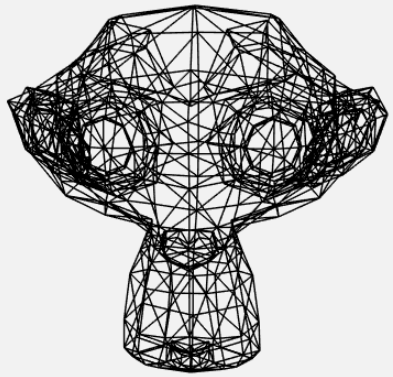
Universitat de Girona

Example: Suzzane

Convert and include file

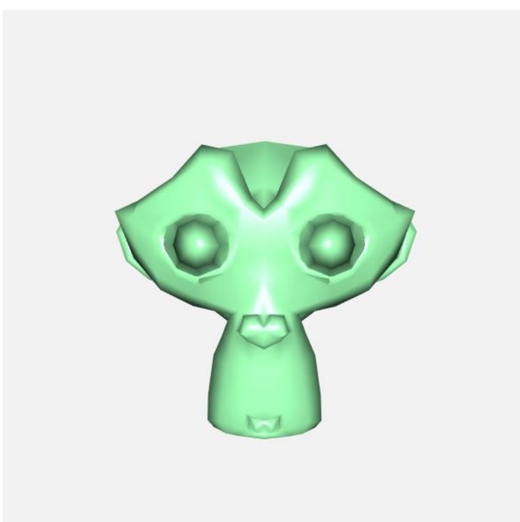
```
<script src="gl-matrix-min.js"></script>
<script src="primitivasG.js"></script>
<script src="suzzane.json"></script>
<script src="mueveLaCamara_OBJ.js"></script>
```

```
// Do the same as for other exemple objects
initBuffers(exampleOBJ);
...
```



3D OBJ with illumination

- Use JSON file
- We need to handle Normals
- Problem:
  - –Normals are in a different array as in Primitive examples
- Solution
  - –Use another buffer



{
"vertexPositions":
[-16,13,0,-7,13,0,-16,2,0,-7,13,0,1,13,0,1,6,0,-16,-9,0,-16,2,0,0,-9,0,15,-
9,0,0,-9,0,8,-1,0,15,13,0,15,2,0,8,6,0,15,2,0,15,-9,0,8,-
1,0,8,13,0,15,13,0,8,6,0,8,-1,0,8,6,0,15,2,0,1,-1,0,8,-1,0,0,-9,0,1,6,0,1,-1,0,-
16,2,0,-16,2,0,-7,13,0,1,6,0,0,-9,0,-16,2,0,1,-1,0,-16,13,14,-16,13,7,-
16,2,14,-16,13,7,-16,13,0,-16,2,0,-16,2,0,-16,-9,0,-16,2,14,-16,-9,14,-
16,2,14,-16,-9,0,-16,2,14,-16,13,7,-16,2,0,-16,13,14,-16,2,14,-7,13,14,-
7,13,14,1,6,14,1,13,14,-16,-9,14,0,-9,14,-16,2,14,15,-9,14,8,-1,14,0,-
9,14,15,13,14,8,6,14,15,2,14,15,2,14,8,-1,14,15,-
9,14,8,13,14,8,6,14,15,13,14,8,-1,14,15,2,14,8,6,14,1,-1,14,0,-9,14,8,-
1,14,1,6,14,-16,2,14,1,-1,14,-16,2,14,1,6,14,-7,13,14,0,-9,14,1,-1,14,-
16,2,14,-16,-9,0,0,-9,0,0,-9,14,0,-9,0,15,-9,0,15,-9,14,15,-9,14,0,-9,14,0,-
9,0,0,-9,14,-16,-9,14,-16,-9,0,15,-9,0,15,2,0,15,-
9,14,15,2,0,15,13,0,15,13,7,15,13,14,15,2,14,15,13,7,15,2,14,15,-
9,14,15,2,0,15,2,0,15,13,7,15,2,14,15,13,0,8,13,0,15,13,7,8,13,14,15,13,14,8
,13,7,15,13,14,15,13,7,8,13,7,8,13,0,8,13,7,15,13,7,8,13,0,8,6,0,8,13,7,8,6,0,
8,-1,0,8,-1,7,8,-1,14,8,6,14,8,-1,7,8,6,14,8,13,14,8,13,7,8,13,7,8,6,0,8,-
1,7,8,-1,7,8,6,14,8,13,7,8,-1,0,1,-1,0,8,-1,7,1,-1,14,8,-1,14,1,-1,7,8,-1,14,8,-
1,7,1,-1,7,1,-1,0,1,-1,7,8,-1,7,1,-1,0,1,6,0,1,-
1,7,1,6,0,1,13,0,1,13,7,1,13,14,1,6,14,1,13,7,1,6,14,1,-1,14,1,-1,7,1,-
1,7,1,6,0,1,13,7,1,13,7,1,6,14,1,-1,7,1,13,0,-7,13,0,1,13,7,-7,13,0,-16,13,0,-
16,13,7,-16,13,7,-16,13,14,-7,13,14,-7,13,14,1,13,14,1,13,7,1,13,7,-7,13,0,-
7,13,14,-7,13,0,-16,13,7,-7,13,14],
""vertexNormals:
[0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-
1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-
1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-
1,0,0,-1,0,0,-1,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-
1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-
1,0,0,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,
1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,
0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,-1,0,0,-1,0,0,-1,0,0,-
1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-
1,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,
1,0,0,1,0,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,
0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-
1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-1,0,0,-
1,0,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,1,0,
0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,
0,0,1,0,0,1,0,0,1,0,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,
0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0],

Universitat de Girona

3D OBJ with illumination

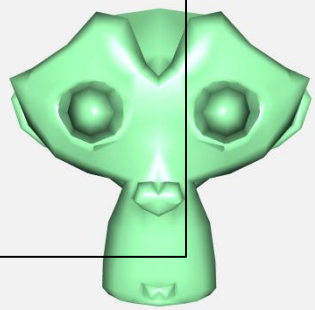
```
function initBuffers(model) {

  model.idBufferVertices = gl.createBuffer ();
  gl.bindBuffer (gl.ARRAY_BUFFER, model.idBufferVertices);
  gl.bufferData (gl.ARRAY_BUFFER, new Float32Array(model.vertices), gl.STATIC_DRAW);

  // here we should pass the buffer for normals
  model.idBufferNormals = gl.createBuffer ();
  gl.bindBuffer (gl.ARRAY_BUFFER, model.idBufferNormals);
  gl.bufferData (gl.ARRAY_BUFFER, new Float32Array(model.vertexNormals),
gl.STATIC_DRAW);

  model.idBufferIndices = gl.createBuffer ();
  gl.bindBuffer (gl.ELEMENT_ARRAY_BUFFER, model.idBufferIndices);
  gl.bufferData (gl.ELEMENT_ARRAY_BUFFER, new Uint16Array(model.indices),
gl.STATIC_DRAW);

}
```



Universitat de Girona

3D OBJ with illumination

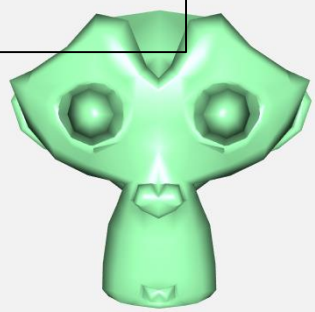
```
function drawSolidOBJ(model) {
  // here we should change the way to decode the vertex and normals vèrtex

  gl.bindBuffer (gl.ARRAY_BUFFER, model.idBufferVertices);
  gl.vertexAttribPointer (program.vertexPositionAttribute, 3, gl.FLOAT, false, 0,   0);

  // normals
  gl.bindBuffer (gl.ARRAY_BUFFER, model.idBufferNormals);
  gl.vertexAttribPointer (program.vertexNormalAttribute,   3, gl.FLOAT, false, 0, 0);

  gl.bindBuffer   (gl.ELEMENT_ARRAY_BUFFER, model.idBufferIndices);
  gl.drawElements (gl.TRIANGLES, model.indices.length, gl.UNSIGNED_SHORT, 0);

}
```



Universitat de Girona

3D OBJ with illumination

- But what if we have both kind of objects:
  - –3D OBJ
  - –Basic primitives
- Possible solution:
  - –Initialize the buffers in the same way
  - –Call objects with diferent funtions:
    - •function drawSolidOBJ(modelOBJ) { ...}
    - •function drawSolid(modelPRIM) { ...}

Universitat de Girona

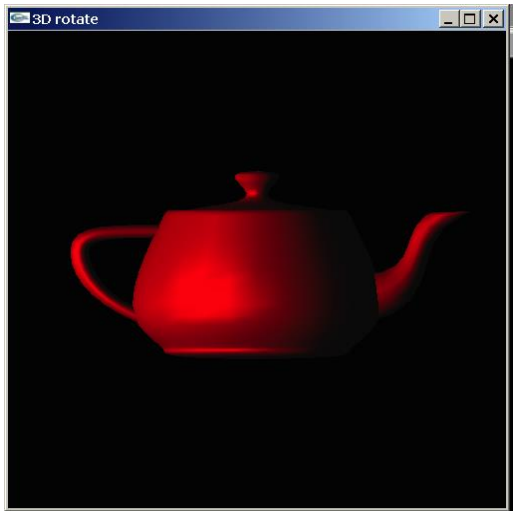
Multiple Light Sources

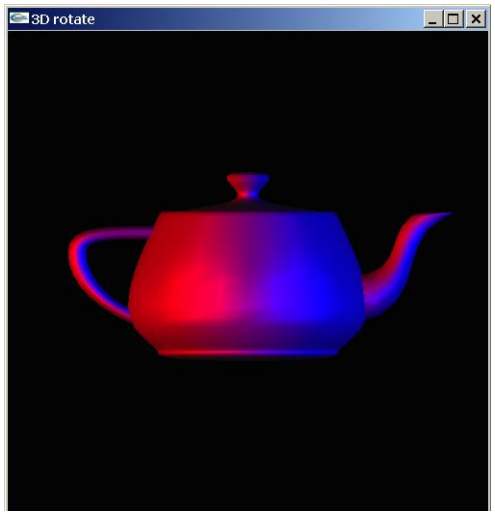
$$I = I_a k_a + \sum_i (I_{i,d} k_d \cdot lambert_i + I_{i,s} k_s \cdot (phong_i)^n)$$

Universitat de Girona

Multiple light sources

- The total reflection at p is the
  sum of all contributed intensities from all sources
- WebGL allows us to define several light sources (as uniforms)





Universitat de Girona

Multiple lights

- Implementation:
  - –Use an array of Lights

```
// in Fragment Shader
struct LightData {
        vec3 Position;
        vec3 La;       // Ambiente
        vec3 Ld;       // Difusa
        vec3 Ls;       // Especular
      };
      uniform LightData Light[100];
      uniform int nLights;
...
for (int i=0; i<nLights; i++) { …
// call Phong for Light i
```

Universitat de Girona

Materials

```
var Gold = {
  "mat_ambient" : [ 0.24725, 0.1995, 0.0745 ],
  "mat_diffuse" : [ 0.75164, 0.60648, 0.22648 ],
  "mat_specular": [ 0.628281, 0.555802, 0.366065 ],
  "alpha"       : [ 51.2 ]
};
```

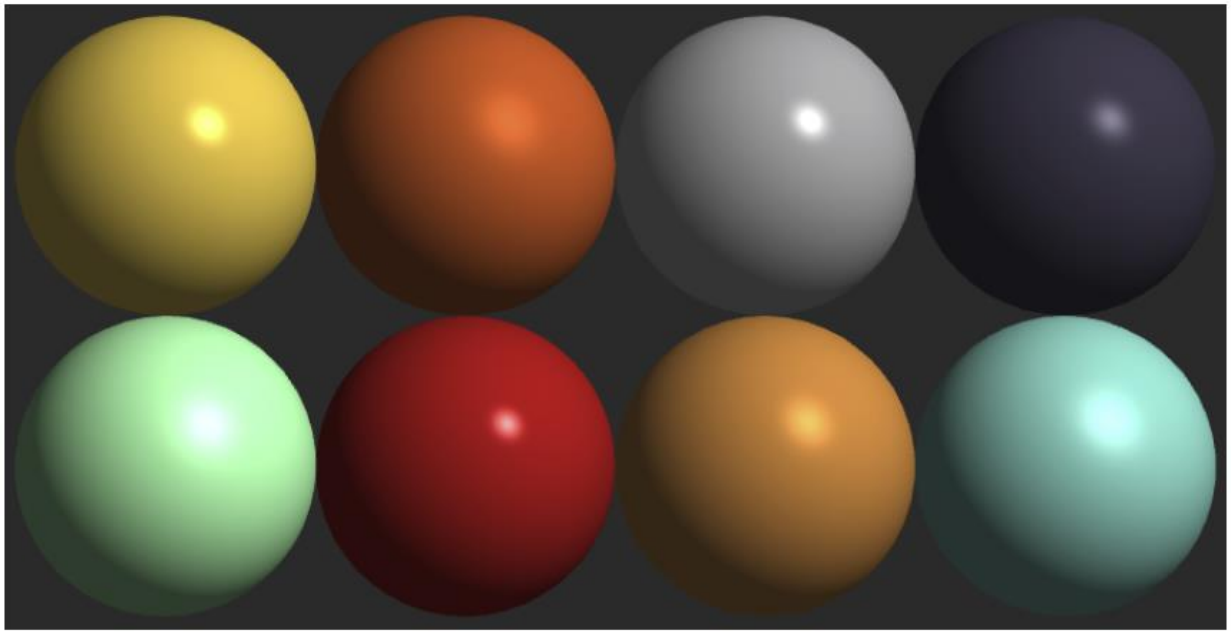


Figura: Ejemplos de materiales, de izquierda a derecha y de arriba a abajo: Oro, Cobre, Plata, Obsidiana, Jade, Rubí, Bronce, Turquesa.