Scalable Rendering for Graphics and Game Engines

Antonio Chica Calaf
achica@cs.upc.edu

Marc Comino Trinidad
mcomino@cs.upc.edu

LEGACY OPENGL

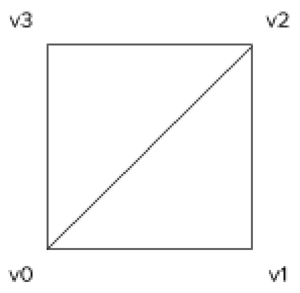
CPU

```
// Render Time
glBegin(GL_TRIANGLES);
   glVertex3fv(v0);
   glVertex3fv(v1);
   glVertex3fv(v2);

   glVertex3fv(v0);
   glVertex3fv(v2);
   glVertex3fv(v3);
glEnd();
```

GPU

// Render Time
Receives 9 x F floats.



VERTEX ARRAYS

CPU

```
// Render Time
GLfloat vertices[] = {v0, v1, v2, v3};
GLubyte indices[] = {0,1,2,0,2,3}.

glEnableClientState(GL_VERTEX_ARRAY);
glVertexPointer(3, GL_FLOAT, 0, vertices);

glDrawElements(GL_TRIANGLES, 6,
GL_UNSIGNED_BYTE, indices);

glDisableClientState(GL_VERTEX_ARRAY);
```

GPU

// Render Time
Receives 3 x V floats and 3 x F integers.

VERTEX BUFFER OBJECTS	
CPU	**GPU**
// Initialization Time	// Initialization Time
GLuint vbo_v_id, vbo_n_id, faces_id;	Receives and **stores** 3 x V floats and 3 x F integers.. Optionally receives and **stores** 3 x V for each additional attribute.
glGenBuffers(1, &vbo_v_id); glBindBuffer(GL_ARRAY_BUFFER, vbo_v_id); glBufferData(GL_ARRAY_BUFFER, dataSize, vertices, GL_STATIC_DRAW);	
glGenBuffers(1, &vbo_n_id); glBindBuffer(GL_ARRAY_BUFFER, vbo_n_id); glBufferData(GL_ARRAY_BUFFER, dataSize, normals, GL_STATIC_DRAW);	
glGenBuffers(1, &faces_id); glBindBuffer(GL_ELEMENT_ARRAY_BUFFER, faces_id); glBufferData(GL_ELEMENT_ARRAY_BUFFER, dataSize, indices, GL_STATIC_DRAW);	
// Render Time	
glBindBuffer(GL_ARRAY_BUFFER, vbo_v_id); glVertexPointer(3, GL_FLOAT, 0, 0); glEnableClientState(GL_VERTEX_ARRAY);	
glBindBuffer(GL_ARRAY_BUFFER, vbo_n_id); glNormalPointer(3, GL_FLOAT, 0, 0); glEnableClientState(GL_NORMAL_ARRAY);	
glBindBuffer(GL_ELEMENT_ARRAY_BUFFER, faces_id)	
glDrawElements(GL_TRIANGLES, 6, GL_UNSIGNED_BYTE, 0);	
glDisableClientState(GL_VERTEX_ARRAY); glDisableClientState(GL_NORMAL_ARRAY);	
glBindBuffer(GL_ARRAY_BUFFER, 0); glBindBuffer(GL_ELEMENT_ARRAY_BUFFER, 0);	

VERTEX ARRAY OBJECTS	
CPU	**GPU**
// Initialization Time GLfloat data[] = {v0, n0, v1, n1, v2, n2, v3, n3}; GLuint vbo_id; glGenBuffers(1, &vbo_id); glBindBuffer(GL_ARRAY_BUFFER, vbo_v_id); glBufferData(GL_ARRAY_BUFFER, dataSize, data, GL_STATIC_DRAW); GLuint vao_id; glGenVertexArrays(1, &vao_id_); glBindVertexArray(vao_id_); glVertexAttribPointer(0, 3, GL_FLOAT, GL_FALSE, dataSize, 0, 0); glEnableVertexAttribArray(0); glBindVertexArray(vao_id_); glVertexAttribPointer(1, 3, GL_FLOAT, GL_FALSE, dataSize, stride, offset); glEnableVertexAttribArray(1); glBindVertexArray(0); glBindBuffer(GL_ARRAY_BUFFER, vbo_v_id); // Render Time glBindVertexArray(vao_id_); glBindBuffer(GL_ELEMENT_ARRAY_BUFFER, faces_id) glDrawElements(GL_TRIANGLES, 6, GL_UNSIGNED_BYTE, 0); glBindBuffer(GL_ELEMENT_ARRAY_BUFFER, 0); glBindVertexArray(GL_ARRAY_BUFFER, 0);	// Initialization Time Receives and **stores** 3 x V floats and 3 x F integers.. Optionally receives and **stores** 3 x V for each additional attribute.

MORE RESOURCES:

https://learnopengl.com/Model-Loading/Mesh

http://www.songho.ca/opengl/gl_vertexarray.html

http://www.songho.ca/opengl/gl_vbo.htmlç

https://www.khronos.org/opengl/wiki/Vertex_Specification#Vertex_Array_Object