



UNIVERSIDAD NACIONAL AUTÓNOMA DE  
MÉXICO

FACULTAD DE INGENIERÍA  
DIVISIÓN DE INGENIERÍA ELÉCTRICA  
INGENIERÍA EN COMPUTACIÓN



LABORATORIO DE COMPUTACIÓN GRÁFICA  
e INTERACCIÓN HUMANO COMPUTADORA

## **EJERCICIOS DE CLASE Nº 06**

**NOMBRE COMPLETO:** CARBAJAL REYES IRVIN JAIR

**Nº de Cuenta:** 422042084

**GRUPO DE LABORATORIO:** 11

**GRUPO DE TEORÍA:** 04

**SEMESTRE 2025-2**

**FECHA DE ENTREGA LÍMITE: 24 DE MARZO DE 2025**

**CALIFICACIÓN:** \_\_\_\_\_

## Actividades

1. Texturizar su cubo con la imagen dado\_animales ya optimizada por ustedes

Para este ejercicio se utilizó la imagen optimizada dado\_mod.jpg la cual se implementó en la textura dadoTexture.

```
272 |     dadoTexture = Texture("Textures/dado_mod.jpg");
273 |     dadoTexture.LoadTextureA();
```

Conforme a los vértices de cada cara según la imagen siguiendo la escala  $s, t$  se colocaron dentro de la función CrearDado con sus respectivas componentes dentro de *cubo\_vertices*.

Caras.

```
203 |     GLfloat cubo_vertices[] = {
204 |         // front
205 |         //x   y   z   S   T   NX   NY   NZ
206 |         -0.5f, -0.5f,  0.5f,  0.26f,  0.34f,  0.0f,  0.0f, -1.0f, //0
207 |         0.5f, -0.5f,  0.5f,  0.49f,  0.34f,  0.0f,  0.0f, -1.0f, //1
208 |         0.5f,  0.5f,  0.5f,  0.49f,  0.66f,  0.0f,  0.0f, -1.0f, //2
209 |         -0.5f,  0.5f,  0.5f,  0.26f,  0.66f,  0.0f,  0.0f, -1.0f, //3
210 |         // right
211 |         //x   y   z   S   T
212 |         0.5f, -0.5f,  0.5f,  0.51f,  0.01f, -1.0f,  0.0f,  0.0f,
213 |         0.5f, -0.5f, -0.5f,  0.74f,  0.01f, -1.0f,  0.0f,  0.0f,
214 |         0.5f,  0.5f, -0.5f,  0.74f,  0.32f, -1.0f,  0.0f,  0.0f,
215 |         0.5f,  0.5f,  0.5f,  0.51f,  0.32f, -1.0f,  0.0f,  0.0f,
216 |         // back
217 |         -0.5f, -0.5f, -0.5f,  0.51f,  0.34f,  0.0f,  0.0f,  1.0f,
218 |         0.5f, -0.5f, -0.5f,  0.74f,  0.34f,  0.0f,  0.0f,  1.0f,
219 |         0.5f,  0.5f, -0.5f,  0.74f,  0.66f,  0.0f,  0.0f,  1.0f,
220 |         -0.5f,  0.5f, -0.5f,  0.51f,  0.66f,  0.0f,  0.0f,  1.0f,
```

```

222     // left
223     //x      y      z      S      T
224     -0.5f, -0.5f, -0.5f,  0.51f,  0.67f,    1.0f,   0.0f,   0.0f,
225     -0.5f, -0.5f,  0.5f,   0.74f,  0.67f,    1.0f,   0.0f,   0.0f,
226     -0.5f,  0.5f,  0.5f,   0.74f,  0.99f,    1.0f,   0.0f,   0.0f,
227     -0.5f,  0.5f, -0.5f,  0.51f,  0.99f,    1.0f,   0.0f,   0.0f,
228
229     // bottom
230     //x      y      z      S      T
231     -0.5f, -0.5f,  0.5f,   0.76f,  0.66f,    0.0f,   1.0f,   0.0f,
232     0.5f,  -0.5f,  0.5f,   0.99f,  0.66f,    0.0f,   1.0f,   0.0f,
233     0.5f,  -0.5f, -0.5f,  0.99f,  0.34f,    0.0f,   1.0f,   0.0f,
234     -0.5f, -0.5f, -0.5f,  0.76f,  0.34f,    0.0f,   1.0f,   0.0f,
235
236     //UP
237     //x      y      z      S      T
238     -0.5f,  0.5f,  0.5f,   0.01f,  0.34f,    0.0f,  -1.0f,  0.0f,
239     0.5f,  0.5f,  0.5f,   0.24f,  0.34f,    0.0f,  -1.0f,  0.0f,
240     0.5f,  0.5f, -0.5f,  0.24f,  0.65f,    0.0f,  -1.0f,  0.0f,
241     -0.5f,  0.5f, -0.5f,  0.01f,  0.65f,    0.0f,  -1.0f,  0.0f,
242

```

Dentro del *main* se llama la textura.

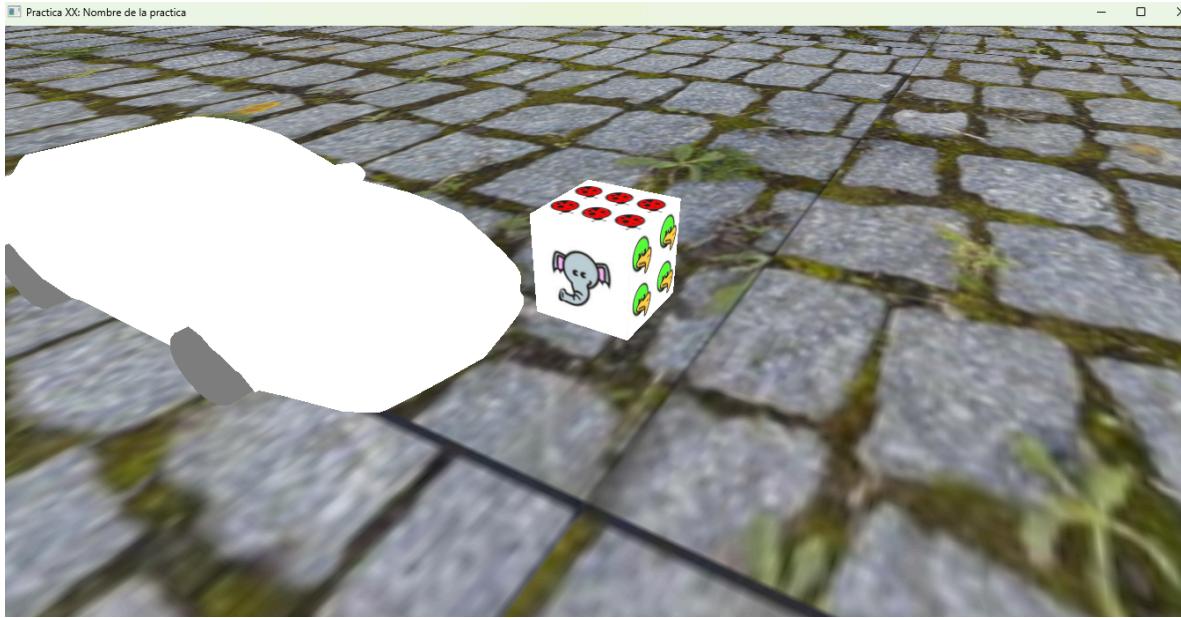
```

340     //Dado de OpenGl
341     //Ejercicio 1: Texturar su cubo con la imagen dado_animales ya optimizada por ustedes
342     model = glm::mat4(1.0);
343     model = glm::translate(model, glm::vec3(-1.5f, 4.5f, -2.0f));
344     glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
345     dadoTexture.UseTexture();
346     meshList[4]->RenderMesh();

```

## Ejecución



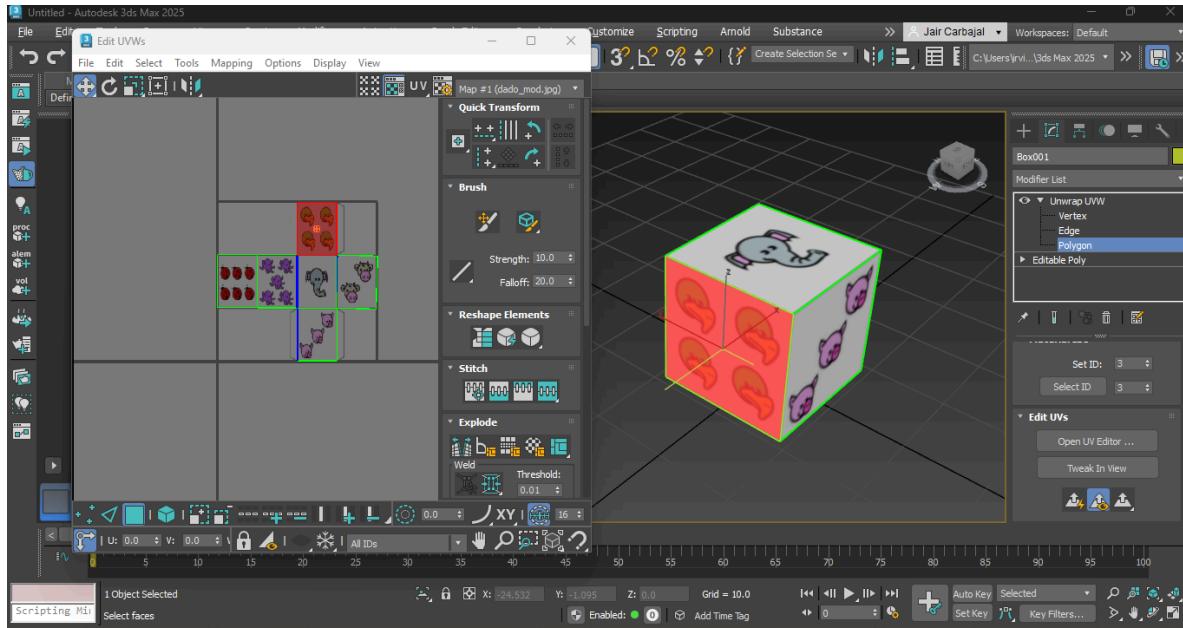


2. Importar el cubo texturizado en el programa de modelado con la imagen dado\_animales ya optimizada por ustedes

Para este ejercicio, se utilizó *LegacyMaterials* dentro del Material Editor para agregar nuestra textura optimizada.

Luego, se le colocó la textura a nuestro cubo para después modificarla con *Unwrap UVW*.

Después abrimos *Open UV Editor* donde vamos a seleccionar nuestra textura y acomodaremos cada cara con su figura correspondiente.



Aquí podemos ir viendo qué cara del cubo se modifica para agregarle su textura. Se colocaron figuras en ubicaciones distintas en comparación a las del cubo texturizado por código. Además se rotó la cara de la base para que corresponda la textura original.

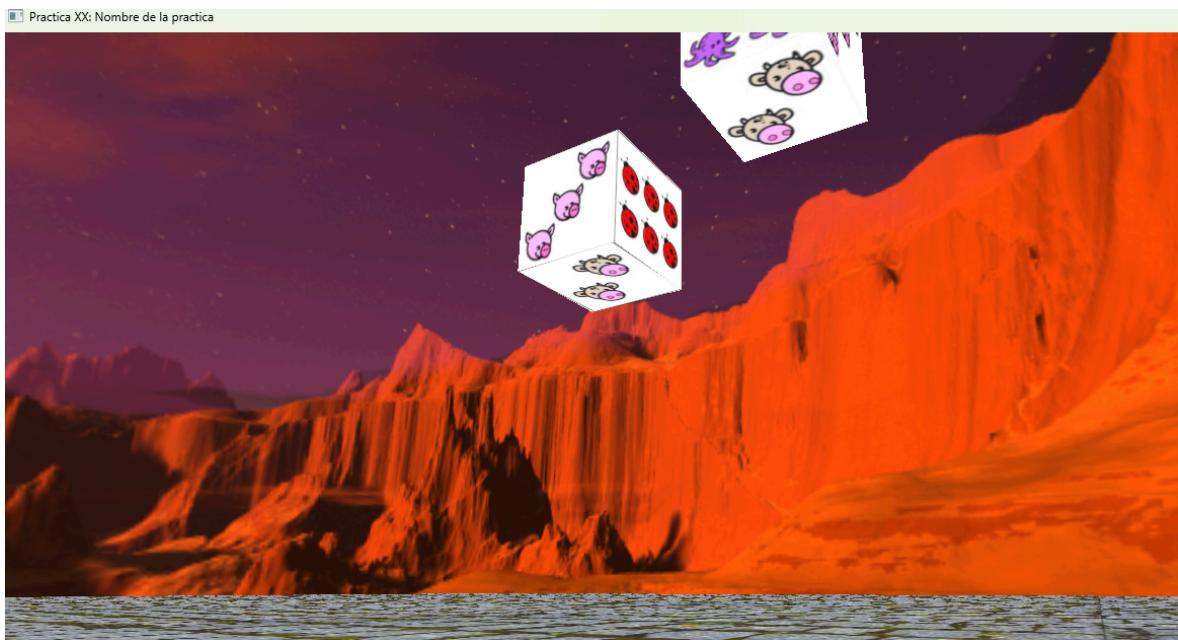
Una vez exportado el cubo como *cubo\_text.obj* se declaró en el código de Visual Studio.

```
283 | Dado_M = Model();
284 | Dado_M.LoadModel("Models/cubo_text.obj");
```

Para después llamarlo en el *main*.

```
354 | //Dado importado
355 | model = glm::mat4(1.0);
356 | model = glm::translate(model, glm::vec3(-3.0f, 3.0f, -2.0f));
357 | model = glm::scale(model, glm::vec3(0.05f, 0.05f, 0.05f));
358 | glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
359 | Dado_M.RenderModel();
```

Ejecución



### Problemas presentados

Durante la elaboración del ejercicio el único problema presentado es que no se tenía presente como implementar *Legacy materials* para agregar nuestra textura optimizada y tampoco del uso de Unwrap UVW para modificar la textura de las caras del cubo. La solución fue investigar su correcto funcionamiento dentro de 3d max y de ese modo se realizó la actividad de forma satisfactoria.

## Conclusiones

Durante la elaboración de este ejercicio fue posible entender los fundamentos de texturizado tanto por código como por modelo, entendiendo como aplicar de manera correcta conceptos básicos del tema, además de herramientas que nos permiten un buen manejo de texturas para nuestras figuras. La complejidad fue adecuada para actividades introductorias de texturizado y la explicación dentro del laboratorio fue esencial para su correcta implementación de cada ejercicio.