



# Final Project

## Technical Manual

**Student:** 316332046

**Group:** 4

**Semester:** 2022-2

**Delivery date:** 11/05/2022

Contents

# Target

The student must apply and demonstrate the knowledge acquired throughout the course of Computer Graphics and Human-Computer Interaction.

# Scope and limitations

To meet the objective of this project, it was proposed to recreate a space of our preference with a room, different objects, animations for some objects, etc. were alsoadded. All this using the OpenGL tool and the use of different knowledge as well as materials acquired during the course.

The recreated room must have 7 3D objects that match the artistic design of the scenario. Each of these objects is textured to look as realistic as possible according to the reference image.

On the other hand, to give more realism to the represented scenario five animations were added where three are simple and two are complex, these animations have a context and behave in harmony with the surrounding environment.

The project will not have the complete recreation of the interior of the facade will only havethe room that has been decided to work, another aspect to consider is that the interior of the room has the basics to represent the structure, but will only have the seven objects that were selected to recreate so the room will not have extra elements except doors or windows that are part of the structure, the same applies to the facade.

# Schedule of activities

To distribute the work to meet the objectives of the project, as well as to avoid an overload of work in the last days, a chronogram of the activities to be conducted was created, including the deadline.

As my work style is a bit chaotic as I tend to do large chunks of project work on weekends, so I do not tend to subdivide tasks too much, instead I tend to divide activities into medium sized activities that are made up of related topics which helps me to establish a relationship between those tasks allowing me to save time.

Note: The week cell is subdivided into seven columns, each corresponding to the day of theweek starting with Monday and ending on Sunday.

| Activities | March | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Week 7-13 | | | | | | | Week 14-20 | | | | | | | Week 21-27 | | | | | | | Week 28-3 | | | | | | | | |
| Election of reference image | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Repository creation | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Modeling of the first object | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Modeling of the second object | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Modeling of the third object | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Modeling of the fourth object | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Modeling of the fifth object | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Modeling of the sixth object | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | April | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | Week 4-10 | | | | | | | Week 11-17 | | | | | | | Week 18-24 | | | | | | | Week 25-1 | | | | | | | | |
| Modeling of the seventh object | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Room modeling | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Facade modeling | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Importing and placing models in space | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Object animations | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Comment on the code | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | May | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | Week 2-8 | | | | | | | Week 9-15 | | | | | | | Week 16-22 | | | | | | | Week 23-29 | | | | | | | | |
| Creating the excutable | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Creating documentation | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

# Development of the activities

As the space to be designed was based on a video game,  we had the advantage of being able to see in more detail the objects that were going to be recreated, so we had a better result when modeling the objects, so we obtained equivalent results.

The problem of this is that I am limited to the mobility offered by the game in terms of the camera because in this game is not designed for exploration preventing me to observe in detail the facade of the bedroom, in particular the back because it was not  possible for me to move around the building so I had to imagine the design of the back, however for the restof objects could be analyzed without major problem except my problem to differentiate the colors that had the objects.

For the texturing we tried to get the closest color to the one observed through the screen, weused programs like GIMP to create the textures and edit them to add the Alpha channel that will serve for the effect of transparency, through Maya we edited the UV maps to be able totexture the object correctly.

In the case of animations as we needed three simple and two complexes where they were related to the object avoiding creating situations out of context.

- First Animation: Door opening and closing
  In this first animation there is a door that opens and closes constantly.
- Second animation: Laptop lid opening and closing.
  Here there is an animation that constantly opens and closes the lid of the laptop.
- Third animation: Room windows closing and opening.
  This time the action of opening and closing the windows is performed vertically in the room.
- Fourth animation: Bird flying following a straight and diagonal trajectory.
  The bird during the animation follows a straight- l i n e flight path that varies atvarious points moving to a diagonal displacement, all this while moving the wingsto simulate the behavior of a bird while flying.
- Fifth animation: Rabbit moving by means of jumps.
  The rabbit during his animation moves by means of jumps following a straight and diagonal path, the action being a jump has a parabolic trajectory in which the rabbit moves his limbs to simulate a more realistic behavior either when jumping or when he is about to fall.

# Code documentation

- Editing the application:
  Now it is time to explain the code that was used, as well as point out that it can be modified to change the behavior of the program whether you want to expand the scope of the project or remove elements that are not needed.
  In line 147 we start with the loading of the models to be used to recreate the facade, the room, and the objects with the respective path of each model.

```
// Load models
	// Fachada
	Model barandillasMadFachada((char*)"Models/Fachada/Barandillas_Madera_Fachada.obj");
	Model barandillasMetFachada((char*)"Models/Fachada/Barandillas_Metal_Fachada.obj");
	Model baseFachada((char*)"Models/Fachada/Base_Fachada.obj");
	Model chimeneasFachada((char*)"Models/Fachada/Chimeneas_Fachada.obj");
	Model columnasFachada((char*)"Models/Fachada/Columnas_Fachada.obj");
	Model detallesFachada((char*)"Models/Fachada/Detalles_Fachada.obj");
	Model escalerasFachada((char*)"Models/Fachada/Escaleras_Fachada.obj");
	Model marcosFachada((char*)"Models/Fachada/Marcos_Fachada.obj");
	Model murosFachada((char*)"Models/Fachada/Muros_Fachada.obj");
	Model puertaAFachada((char*)"Models/Fachada/PuertaA_Fachada.obj");
	Model puertaB1Fachada((char*)"Models/Fachada/PuertaB1_Fachada.obj");
	Model puertaB2Fachada((char*)"Models/Fachada/PuertaB2_Fachada.obj");
	Model puertaSFachada((char*)"Models/Fachada/PuertaS_Fachada.obj");
	Model techoFachada((char*)"Models/Fachada/Techo_Fachada.obj");
	Model vidriosFachada((char*)"Models/Fachada/Vidrios_Fachada.obj");

	// Habitacion_1
	Model marcosH1((char*)"Models/Habitacion_1/Marcos_H1.obj");
	Model muroAtrH1((char*)"Models/Habitacion_1/Muro_Atr_H1.obj");
	Model muroDerH1((char*)"Models/Habitacion_1/Muro_Der_H1.obj");
	Model muroFreH1((char*)"Models/Habitacion_1/Muro_Fre_H1.obj");
	Model muroIzqH1((char*)"Models/Habitacion_1/Muro_Izq_H1.obj");
	Model pisoH1((char*)"Models/Habitacion_1/Piso_H1.obj");
	Model puertaH1((char*)"Models/Habitacion_1/Puerta_H1.obj");
	Model roperoH1((char*)"Models/Habitacion_1/Ropero_H1.obj");
	Model techoH1((char*)"Models/Habitacion_1/Techo_H1.obj");
	Model ventana1AH1((char*)"Models/Habitacion_1/VentanaA_H1.obj");
	Model ventana1BH1((char*)"Models/Habitacion_1/VentanaB_H1.obj");
	Model vidrioVentana1AH1((char*)"Models/Habitacion_1/Vidrio_VentanaA_H1.obj");
	Model vidrioVentana1BH1((char*)"Models/Habitacion_1/Vidrio_VentanaB_H1.obj");
	Model ventana2AH1((char*)"Models/Habitacion_1/VentanaA_H1.obj");
	Model ventana2BH1((char*)"Models/Habitacion_1/VentanaB_H1.obj");
	Model vidrioVentana2AH1((char*)"Models/Habitacion_1/Vidrio_VentanaA_H1.obj");
	Model vidrioVentana2BH1((char*)"Models/Habitacion_1/Vidrio_VentanaB_H1.obj");

	//Objetos Habitacion 1
	Model camaH1((char*)"Models/Objetos_H1/Cama/Cama_H1.obj");
	Model laptop1H1((char*)"Models/Objetos_H1/Laptop/Laptop1_H1.obj");
	Model laptop2H1((char*)"Models/Objetos_H1/Laptop/Laptop2_H1.obj");
	Model estanteH1((char*)"Models/Objetos_H1/Estante/Estante_H1.obj");
	Model escritorioH1((char*)"Models/Objetos_H1/Escritorio/Escritorio_H1.obj");
	Model libros1H1((char*)"Models/Objetos_H1/Libros/Libros1_H1.obj");
	Model libros2H1((char*)"Models/Objetos_H1/Libros/Libros2_H1.obj");
	Model sillaH1((char*)"Models/Objetos_H1/Silla/Silla_H1.obj");
	Model sofaH1((char*)"Models/Objetos_H1/Sofa/Sofa_H1.obj");
```

```
        //Objeto Pajaro
        Model alaDerPajaro((char*)"Models/Objetos_H1/Pajaro/Ala_Derecha_Pajaro.obj");
        Model alaIzqPajaro((char*)"Models/Objetos_H1/Pajaro/Ala_Izquierda_Pajaro.obj");
        Model cuerpoPajaro((char*)"Models/Objetos_H1/Pajaro/Cuerpo_Pajaro.obj");

        //Objeto Conejo
        Model cuerpoConejo((char*)"Models/Objetos_H1/Conejo/Cuerpo_Conejo.obj");
        Model pata1DelDer((char*)"Models/Objetos_H1/Conejo/Pata1_Delantera_Der.obj");
        Model pata1DelIzq((char*)"Models/Objetos_H1/Conejo/Pata1_Delantera_Izq.obj");
        Model pata1TraDer((char*)"Models/Objetos_H1/Conejo/Pata1_Trasera_Der.obj");
        Model pata1TraIzq((char*)"Models/Objetos_H1/Conejo/Pata1_Trasera_Izq.obj");
        Model pata2DelDer((char*)"Models/Objetos_H1/Conejo/Pata2_Delantera_Der.obj");
        Model pata2DelIzq((char*)"Models/Objetos_H1/Conejo/Pata2_Delantera_Izq.obj");
        Model pata2TraDer((char*)"Models/Objetos_H1/Conejo/Pata2_Trasera_Der.obj");
        Model pata2TraIzq((char*)"Models/Objetos_H1/Conejo/Pata2_Trasera_Izq.obj");
```

It is recommended that if the path of the models is modified and the code path should also be updated because, although it will not prevent the execution of the program, it will prevent the model from being loaded. On the other hand, if you want to prevent a specific object from being shown, it is necessary to comment with // all the *Model* lines that make up that object.

Finally, if you want to add a created object, just move the object to the project folder and create the instance in the code with the path where the object in question is; it is recommended that the texture of the object is in the same folder to avoid failures.

Line 250 sets the parameters to establish the lighting that will have the environment so if you want to modify it is recommended to modify the values of *dirLight.ambient* and *dirLight.diffuse* considering that 1.0 is the maximum and 0.1 the minimum. The rest of the values it is recommended to avoid changing their value unless you think it is necessary and you know what you are doing.

```
// Directional light
                glUniform3f(glGetUniformLocation(lightingShader.Program,
"dirLight.direction"), -0.2f, -1.0f, -0.3f);
                glUniform3f(glGetUniformLocation(lightingShader.Program, "dirLight.ambient"),
0.4f, 0.4f, 0.4f);
                glUniform3f(glGetUniformLocation(lightingShader.Program, "dirLight.diffuse"),
0.4f, 0.4f, 0.4f);
                glUniform3f(glGetUniformLocation(lightingShader.Program, "dirLight.specular"),
1.0f, 1.0f, 1.0f);
```

From line 273 to line 561 we start to load the characteristics of the Model instances defining in some cases its position, rotation, or scale. In this project the scale and position of the object with respect to the space was established by means of the Maya program to avoid a too high cost of resources that if it was done directly in OpenGL, so it is recommended that if you want to place the new object without animation you do it by means of Maya. If the object in question is going to be animated it is recommended that you, do it by means of basic transformations in the code and only Maya is in charge of establishing the scale and the pivot of the object, all this to avoid having failures by a bad handling of the pivot which would affect the animation.

In the following code fragment the bed object is not animated and the laptop object is

animated causing the latter to have basic transformations.

```
//Carga de modelos de objetos de la Habitacion_1
              //Cama
              model = glm::mat4(1);
              glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
              glUniform1i(glGetUniformLocation(lightingShader.Program,
"activaTransparencia"), 0);
              camaH1.Draw(lightingShader);

              //Laptop
              model = glm::mat4(1);
              model = glm::translate(model, glm::vec3(12.008f, 4.827f, -1.868f));
              model = glm::rotate(model, glm::radians(47.256f), glm::vec3(0.0f, 1.0f,
0.0f));
              model = glm::rotate(model, glm::radians(laptop1H1Rot), glm::vec3(1.0f, 0.0f,
0.0f));
              glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
              glUniform1i(glGetUniformLocation(lightingShader.Program,
"activaTransparencia"), 0);
              laptop1H1.Draw(lightingShader);

              model = glm::mat4(1);
              model = glm::translate(model, glm::vec3(12.008f, 4.827f, -1.868f));
              model = glm::rotate(model, glm::radians(47.256f), glm::vec3(0.0f, 1.0f,
0.0f));
              glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
              glUniform1i(glGetUniformLocation(lightingShader.Program,
"activaTransparencia"), 0);
              laptop2H1.Draw(lightingShader);
```

From line 563 to 605 the characteristics of the *Model* instances are loaded in the same way with the difference that these objects have characteristics of degrees of transparency as is a glass for a window allowing you to see what is on the other side. If you want to change the level of transparency just change the fourth value of the *colorAlpha* parameter considering that 1.0 is the maximum and 0.1 is the minimum.

```
//Codigo para objetos semitrasnparentes
              glEnable(GL_BLEND);//Activa la funcionalidad para trabajar el canal alfa
              glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);

              //Carga de modelo de Habitacion_1
              model = glm::mat4(1);
              model = glm::translate(model, glm::vec3(12.209f, 5.525f, -1.548f));
              glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
              glUniform1i(glGetUniformLocation(lightingShader.Program,
"activaTransparencia"), 1);
              glUniform4f(glGetUniformLocation(lightingShader.Program, "colorAlfa"), 1.0f,
1.0f, 1.0f, 0.5f);
              vidrioVentana1AH1.Draw(lightingShader);
```

```
            model = glm::mat4(1);
            model = glm::translate(model, glm::vec3(12.209f, 5.525f+ventanaBH1Tra, -
1.548f));
            glUniformMatrix4fv(modelLoc, 1, GL_FALSE, glm::value_ptr(model));
            glUniform1i(glGetUniformLocation(lightingShader.Program,
"activaTransparencia"), 1);
            glUniform4f(glGetUniformLocation(lightingShader.Program, "colorAlfa"), 1.0f,
1.0f, 1.0f, 0.5f);
            vidrioVentana1BH1.Draw(lightingShader);
```

In the *DoMovement()* function the input is handled by means of the keys that affect the movement of the camera with w, s, a, d, etc. As already explained in the use of the application. If you want to change the key that performs these movements, you only need to modify the condition by setting the key to your preference.
On the other hand, the function when called constantly serves to define the animations that you require either of new objects that you decide to add or those that are already in the project. If you decide to delete a particular animation, you can choose to comment it out or deactivate the key to that animation, how to do this will be explained later.

```
void DoMovement()
{
	// Camera controls
	if (keys[GLFW_KEY_W] || keys[GLFW_KEY_UP])
	{
		camera.ProcessKeyboard(FORWARD, deltaTime);
	}

	if (keys[GLFW_KEY_S] || keys[GLFW_KEY_DOWN])
	{
		camera.ProcessKeyboard(BACKWARD, deltaTime);
	}

	if (keys[GLFW_KEY_A] || keys[GLFW_KEY_LEFT])
	{
		camera.ProcessKeyboard(LEFT, deltaTime);
	}

	if (keys[GLFW_KEY_D] || keys[GLFW_KEY_RIGHT])
	{
		camera.ProcessKeyboard(RIGHT, deltaTime);
	}
```

The *KeyCallback()* function is called every time a key is pressed or released, which is very useful when indicating whether an animation is activated or deactivated. To define a new animation, it is enough to create a condition in which the key that will activate the animation is indicated, as well as to create a bool variable that allows to verify that the condition is

fulfilled.
As mentioned before to disable an animation just remove the condition that activates it or change the content of the condition for the new animation you want to add.

```
void KeyCallback(GLFWwindow* window, int key, int scancode, int action, int mode)
{
	if (GLFW_KEY_ESCAPE == key && GLFW_PRESS == action)
	{
		glfwSetWindowShouldClose(window, GL_TRUE);
	}

	if (key >= 0 && key < 1024)
	{
		if (action == GLFW_PRESS)
		{
			keys[key] = true;
		}
		else if (action == GLFW_RELEASE)
		{
			keys[key] = false;
		}
	}

	if (keys[GLFW_KEY_1])
	{
		puertaH1Anim = !puertaH1Anim;
	}

	if (keys[GLFW_KEY_2])
	{
		laptop1H1Anim = !laptop1H1Anim;
	}

	if (keys[GLFW_KEY_3])
	{
		ventanaBH1Anim = !ventanaBH1Anim;
	}

	if (keys[GLFW_KEY_4])
	{
		pajaroAnim = !pajaroAnim;
	}

	if (keys[GLFW_KEY_5])
	{
		conejoAnim = !conejoAnim;
	}
}
```

The MouseCallback() function is called every time the mouse is moved, it is used to define the movement of the synthetic camera so it is not recommended to modify your code unless you need to and you know what you are doing.

```
void MouseCallback(GLFWwindow* window, double xPos, double yPos)
{
	if (firstMouse)
	{
		lastX = xPos;
		lastY = yPos;
		firstMouse = false;
	}
	GLfloat xOffset = xPos - lastX;
	GLfloat yOffset = lastY - yPos;  // Reversed since y-coordinates go from bottom to
left

	lastX = xPos;
	lastY = yPos;

	camera.ProcessMouseMovement(xOffset, yOffset);
}
```

# Conclusions

With the development of this project I was able to meet the objective and scope established at the beginning of the document since the knowledge acquired through the practices were applied from the modeling of an object through code or software as well as texturing and lighting to add more realism and the use of various concepts such as basic transformations that allow changing the appearance of the object and can be used to create simple or complex animations, hierarchical modeling was used for these animations that allows greater flexibility to work, etc.. On the other hand, the scope of the project was fulfilled since the façade, the room and the seven objects chosen at the beginning of the project were recreated without forgetting that the five required animations were added where three were simple and two were complex.

To conclude, in my opinion it was a good course that allowed me to approach a little to the development of applications related to computer graphics that helped me to better understand the process behind several programs that we use in everyday life but we do not usually give importance, however, now I can appreciate and know how is the process that is done in each game, graphic application or program so I feel quite satisfied.

# References and programs used.

Programs:

- Maya 2023 for modeling and texturing objects, also to define their scale and position in space.
- GIMP for creating and editing textures.
- Visual Studio for the development of the program code to be able to recreate the space to be modeled.
- Gitkraken for github repository management

References:


- Textures for 3D, graphic design and Photoshop!. Retrieved May 9, 2022, from https://www.textures.com/.
- 3DModelsforProfessionals.Retrieved 9 May 2022, from https://www.turbosquid.com/.
- Freepik| Graphic resources for everyone. Retrieved May 9, 2022, from https://www.freepik.es/.
- PILIIGON | Search 4000+ Assets for 3D Artists. Retrieved May 9, 2022, from https://www.poliigon.com/.