UNAM-Facultad de ingeniería

Semestre 2023-2

Computación Gráfica e Interacción Humano Computadora

Group: 05

Project 02 - Scenario based on the cartoon 'The Fairly OddParents'.

Delivery date: 08/June/2023

Members:

317087354

Objective

Apply all the knowledge acquired about computer graphics: modeling, lighting, animation, as well as the use of the OpenGL API in the C++ programming language to recreate a scene.

Scope

Create a scene where we observe a facade of a real or fictitious space in which there are 2 rooms with 5 objects that are consistent with the environment of the scene, in addition to 4 animations, lighting and synthetic camera.

The project must have as results an executable and a documentation among which there will be objectives, scope, schedule of activities and the tools used, as well as a technical manual containing a dictionary of functions and variables and finally the conclusions.
The above must be done within a term no later than June 8, 2023, the day on which the project will be delivered.

Work methodology

The Scrum methodology has been selected for this project. It is a highly recommended work methodology due to its ability to adapt to constantly evolving changes and needs. With Scrum, the team can remain flexible and quickly adjust priorities and objectives as new situations arise. In addition, Scrum is based on early and continuous delivery of value, allowing stakeholders to realize tangible benefits from the start and provide valuable feedback. By using iterative and incremental development cycles, Scrum encourages continuous process improvement, resulting in greater efficiency and productivity. In addition, by breaking the project into sprints and conducting periodic reviews, Scrum helps identify and mitigate risks early, which helps reduce potential roadblocks and ensure overall project success.

3

Activity schedule

Name	Number	State	Start date	End date
Know the needs of the project.	1	Done	2023-04-27	2023-04-27
Choose the theme of the project	2	Done	2023-05-08	2023-05-09
Set the bases for the code project	3	Done	2023-05-16	2023-05-16
Model the house	4	Done	2023-05-16	2023-05-21
Get model from internet (bed, chair, fishbowl, table, lamp, TV, armchair, characters) in FBX format	5	Done	2023-05-21	2023-05-23
Combine the acquired models to the main model in Blender	6	Done	2023-05-23	2023-05-24
Texture all scene with Blender	7	Done	2023-05-24	2023-05-25
Merge and accommodate the whole scene	8	Done	2023-05-25	2023-05-28
Import individually models who will be animated	9	Done	2023-05-27	2023-05-28
Modify shaders to animate models	10	Done	2023-05-28	2023-05-30
Configure third person camera and character	11	Done	2023-05-31	2023-06-02

WEEK 1						WEEK 2							WEEK 3						
L	M	M	J	V	S	L	M	M	J	V	S	D	L	M	M	J	V	S	D
24	25	26	27	28	29	1	2	3	4	5	6	7	8	9	10	11	12	13	14
			1																
													2						

Table 1. Weeks 1-3

WEEK 4							WEEK 5							WEEK 6					
L	M	M	J	V	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S
15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	1	2	3
	3																		
		4	4	4	4	4													
						5	5	5											
								6	6										
									7	7	7	7	7						
											8	8	8						
												9	9						
													10	10	10				
																11	11	11	

Table 2. Weeks 4-6

Cost analysis

Costs					
Concept	Cost per hour MX$	Taxes IVA & ISR	Total hours		Total MX$
3D Modeler	$100.00	26%	15		$1,890.00
Designer	$90.00	26%	5		$567.00
Programmer	$80.00	26%	10		$1,008.00
Labor	$50.00		30		$1,500.00
Depreciation					
	Cost per day MX$	Days worked			Total MX$
Computer use	$10.00	30			$300.00
Expenses					
Concept	Costo mensual MX$				
Food	$600.00				
Water	$150.00				
Internet	$300.00				
PC power consumption	$200.00				
Total	**$1,250.00**				
Price					
Costs	$6,515.00				
Profit of 40%	$2,606.00				
Price	$9,121.00				

Table 3. Costs analisys

Work tools

We worked in the C++ programming language.

In addition, to homogenize the work environment, it is proposed to use the following software tools:

Microsoft Visual Studio

Visual Studio Code as a code editor and compiler, for its ease of use and wide range of tools that help us configure our project.

GitHub Desktop

This will allow for greater synchronization when working on the project by enabling the ability to interact with GitHub using a GUI instead of the command line or web browser.

Blender

Blender is an open-source 3D graphics software. It is widely used to create 3D models, animations, visual effects, and interactive applications.

Technical manual

Modeling

The first stage of the product development was the modeling, for which Blender software was used. Most of the models were obtained from the internet, so they had to be scaled and the number of polygons had to be reduced to speed up the loading of models in OpenGL.

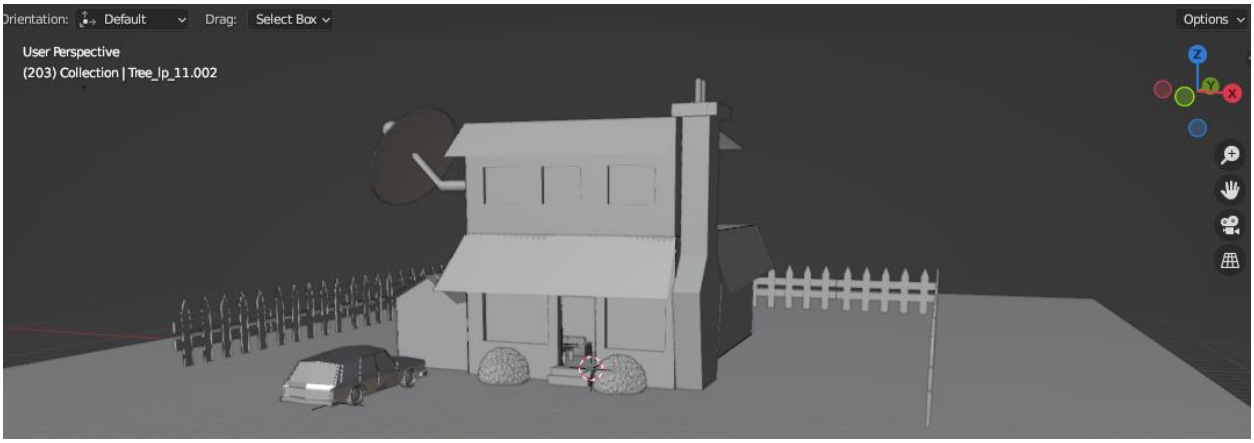

Figure 1 Facade of the house. Solid view

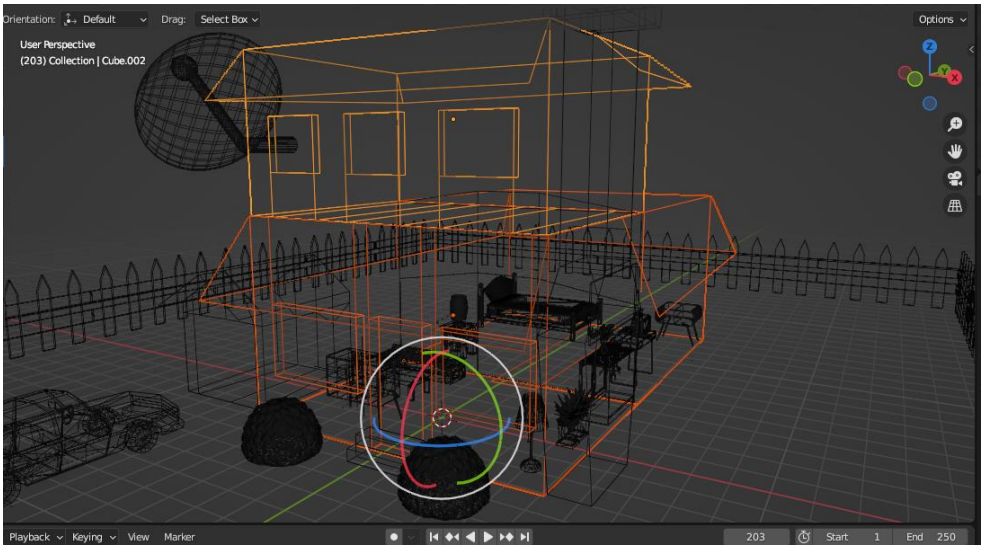

Figure 2 Facade of the house. Mesh view

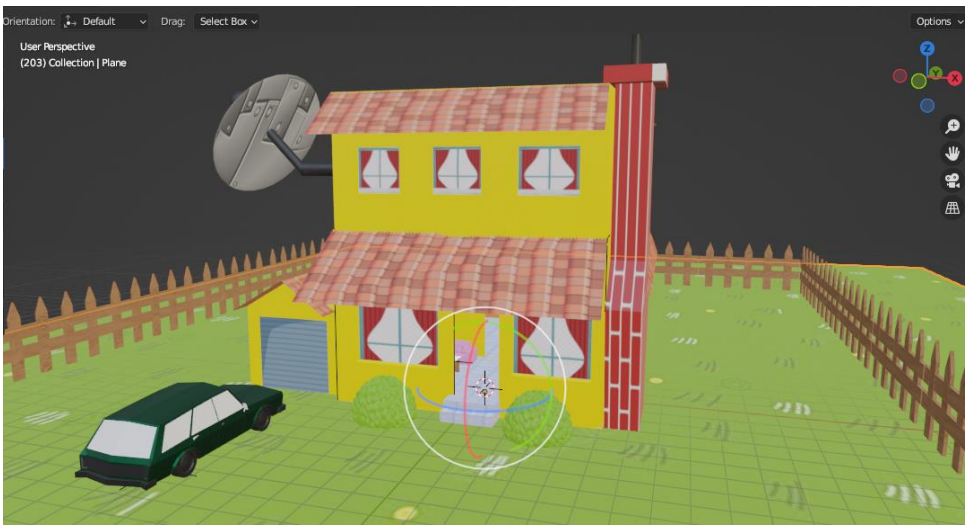

Figure 3 Facade of the house. View with materials

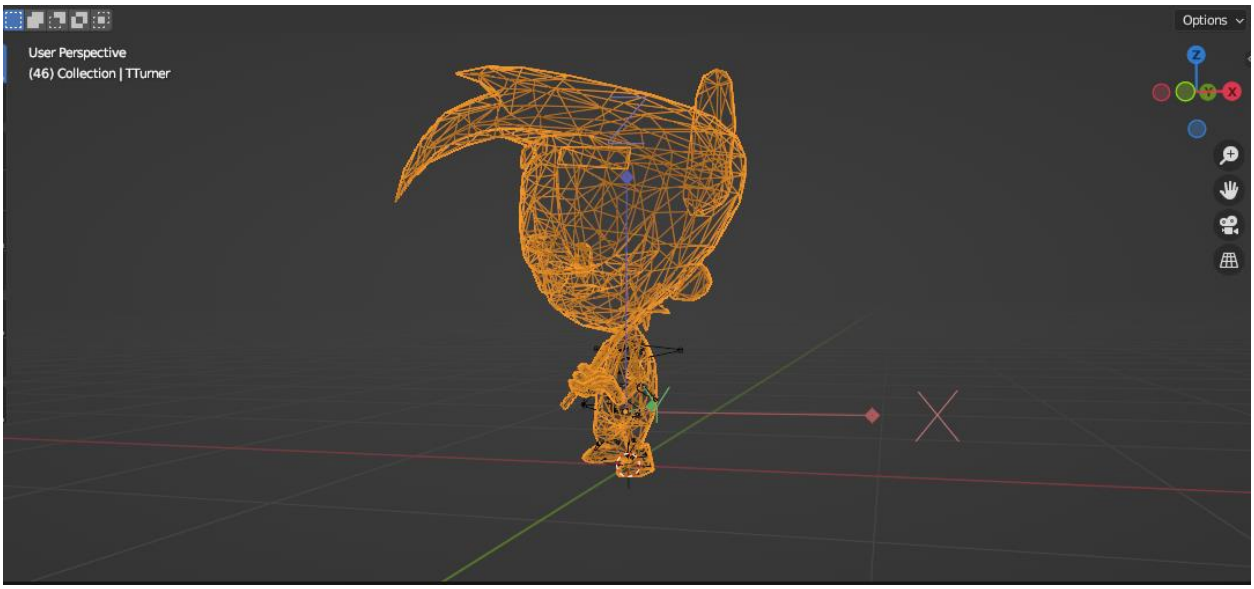

Figure 4 Character in mesh view

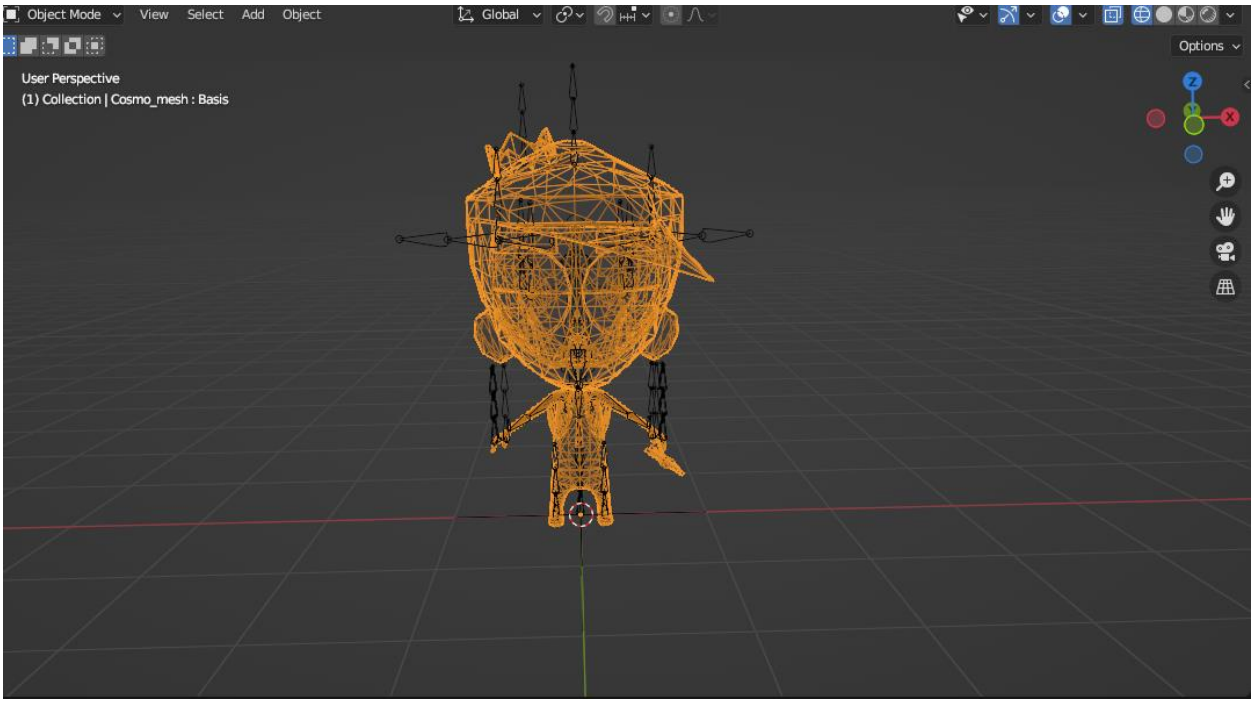

Figure 5 Character in mesh view

Flowcharts

For the development of the C++ code, we use 2 main functions. They follow the following path.

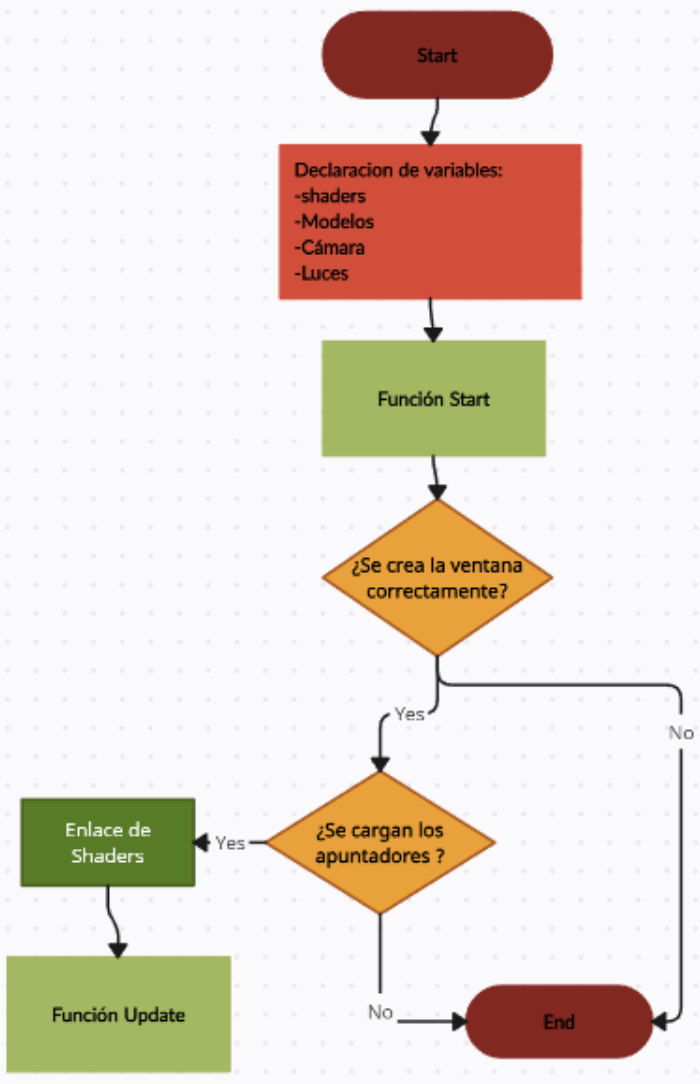

Figure 6 Flowchart of the Start function

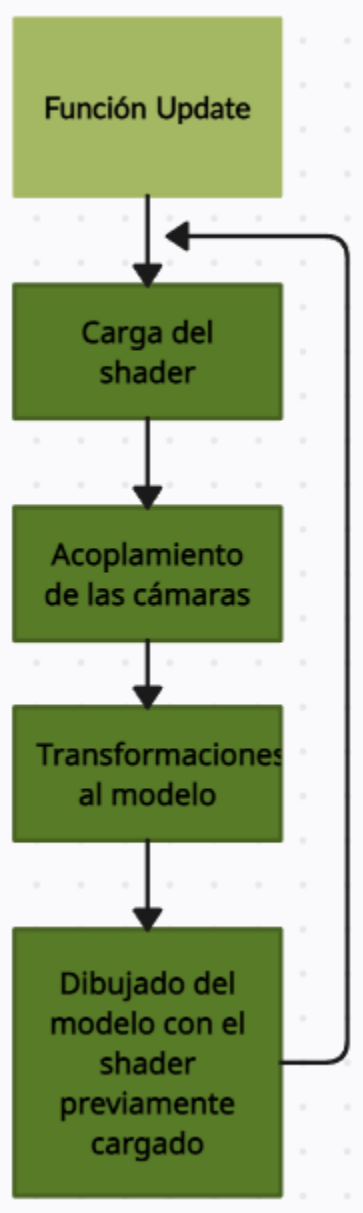

Figure 7 Flowchart of the Update function

11

Variables table

Variable name	Type	Description
camera	Camera	Object that represents the camera in first person, it has the properties of view, position, and vector up.
Camera3rd	Camera	Object that represents the third person camera that follows the character's view, it has the properties of view, position, and vector up.
character	Model	Model type pointer, which points to the FBX file of our character that we control.
house	Model	Apuntador de tipo Model, que apunta al archivo FBX de la fachada de la casa.
door	Model	Model type pointer, which points to the FBX file of the house facade.
pecera	Model	Model type pointer, which points to the FBX file of a fishbowl model.
hada	Model	Model type pointer, which points to the FBX file of a model of our character 'Cosmo'.
cobija	Model	Model type pointer, which points to the FBX file of a blanket model.
ourShader	Shader	Shader type pointer, which points to the shader files used to draw the door.

keyFrames	Shader	Shader type pointer, which points to the shader files used to draw and animate the ‘character’ model.
cubemapShader	Shader	Shader type pointer, which points to the shader files used to draw the skybox of the scenery.
mLightsShader	Shader	Shader pointer, which points to the shader files used to calculate the lighting components.
proceduralShader	Shader	Shader pointer, which points to the shader files used to draw and animate the 'hada' model.
wavesShader	Shader	Shader pointer, which points to the shader files used to draw and animate the ‘cobija’ model.
peceraShader	Shader	Shader type pointer, which points to the shader files used to draw and animate the 'pecera' model.

Table 4. Variables dictionary

The variables of the objects that were not imported together with the main scenario of the facade, is because they are objects that were animated and therefore need to be imported individually.

Table of functions

Function name	Description
main()	Calls our 2 most important functions: 'Start' and 'Update'.
Start()	In this function, variables provided by the included libraries are initialized, allowing the creation of the window where the image will be displayed. Additionally, this function links the Shader type pointers with the files where the program that will execute each shader is hosted. In addition to linking the Model type pointers with the modeling file that we are going to import to the program in FBX format.
Update()	The Update function is the one that gives us the dynamic effect to our whole scenario since in it the models and their respective transformations are being updated while the program is running.
ProcessInput()	It is a function controlled by the GLFW library, which allows us to add keyboard interactions.
Draw(Shader *var)	Method of the Model class that performs the process of drawing the model on the screen.
translate(model, glm::vec3(x,y,z))	Function that applies a translation in any of the 3 axes.
rotate(model, radians, glm::vec3(x,y,z))	Function that applies a rotation on any of the 3 axes. It is recommended to apply this transformation to only one axis at a time.
scale(model, glm::vec3(x,y,z))	Function that applies a scaling in any of the 3 axes.

Table 5. Dictionary of functions

Repository link

GitHub - MIKEcsl5/Proyecto_CompuGrafica at Entregable-4(Final)

Conclusion

The realization of this project was a challenge not only technically, but mostly in the organization, since I had to investigate how to document a project, work methodologies and thus organize my time for the completion of this project and meet deadlines. Additionally, the cost analysis was a little difficult, since I didn't know everything there is behind the quotation of a job. On the technical side I was very pleased to face the challenge of modeling for the first time and realize that the tools are intuitive. I also learned how the graphics generation works at a lower level and how is the process for the computer to generate an image.