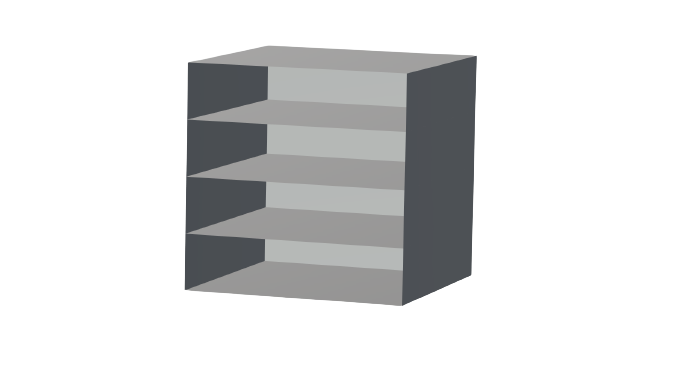
CABLE BOX



David Landete Expósito

Simón Brotons Cabrera

DM3D

11/04/2022

INDEX

[INTRODUCTION 3](#_Toc100429537)

[DEVELOPMENT 4](#_Toc100429538)

[RESULTS 7](#_Toc100429539)

[PERFORMANCE 8](#_Toc100429540)

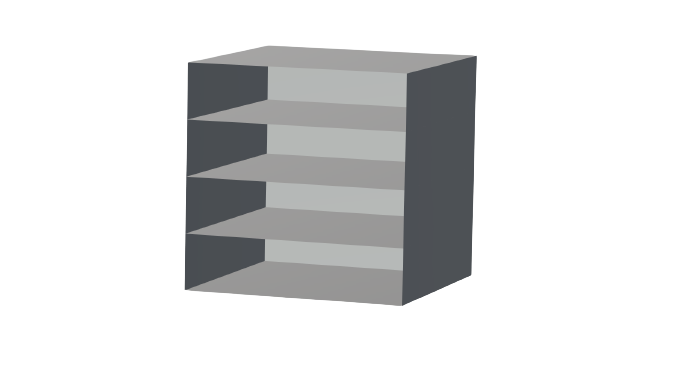
[REFERENCES 8](#_Toc100429541)

References: Sources and references that you have consulted during the development of the project.

# INTRODUCTION

We have developed a program to create a cable box. In the program you can define a few parameters to create a final model in STL format. The parameters you need to introduce are the width, height, depth and number of levels to manage the cables.

The basic model of a cable box with 4 levels, including floor. Using this box, you can pass the cable through the box so you can have an enter and exit hole.



The theorical concepts treated in this program is the generation of an STL file to create the model of the object. This STL file can be rendered and printed.

# DEVELOPMENT

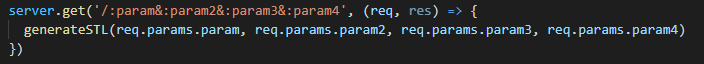
To develope the program we are using a basic HTML + JS + CSS combination.

The view is created with HTML and using js we can have a frontend function to send the parameters to the backend.

To create the backend, we have a basic server that receives a message with the parameters and generates the exit file in STL format.

The first thing to talk about is the creation of the **REST** **API** that we use to communicate the frontend and backend of our application.

Our main method, **generateSTL,** makes use of four parameters: width, height, depth and levels, to generate the final STL file.



In the program, the first step is to create the outside box that will contain the levels to arrange the cables.

Texto

Descripción generada automáticamente

The next step is to calculate the distance between the levels of the box. We are using JavaScript so we need to use the function Number because if we don’t use it, JavaScript will treat the addition as a String. If we don’t use the function, the result of the addition would be ‘31’ if we use ‘3’ as the number of levels. In the other hand, when we use the function, the result is the required, ‘4’.

Texto

Descripción generada automáticamente

In order to create a face, we need to calculate the triangles of the facet. We use the methods **generateVerticalFacet**, **generateOppositeVerticalFacet**, **generateHorizontalFacet** and **generateOppositeHorizontalFacet** to calculate the different facets. We need to create opposite facets because the object can be seen from different angles. If they’re not calculated, we will see void where there should be a facet.

Each generate method, calls the function **facetTriangleString** four times, to generate the four triangles that represent the facet.

Diagrama

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Each facet is composed by two points and a normal vector, all of them belonging to the facet.

Texto

Descripción generada automáticamente

During the creation of the facet, we call the method **calculateNVector**, where we multiply two vectors of the same facet to obtain the normal.

Texto

Descripción generada automáticamente

As you can see, the parameters of **facetTriangleString** do not include two vectors. In order to calculate them, we use the following function:

Texto

Descripción generada automáticamente

# 

# RESULTS

As a result of the development, we have a web divided in two zones. One that asks for input parameters (width, height, depth and levels) and a second one where a model is shown depending on the parameters that the user submitted.

Given the scenario where the user wants to modify the object, the user would have to press the refresh button in order to render it once again.

Interfaz de usuario gráfica, Texto, Sitio web

Descripción generada automáticamente

Captura de pantalla de computadora

Descripción generada automáticamente

This is an example of a cable box with one level.

Captura de pantalla de computadora

Descripción generada automáticamente

In this picture, we can see the generated STL file.

# PERFORMANCE

We divided the tasks of the project so that we would end up doing 50% of the work.

# REFERENCES

Fetch method front to back communication.

<https://www.youtube.com/watch?v=2Xm9P_tXtK8&ab_channel=CarlosAzaustre-AprendeJavaScript>

STL format and normal vector calculation.

Polygon Mesh Modelling pdf

<https://poliformat.upv.es/access/content/group/GRA_11640_2021/Modules/2.%20Geometric%20Modelling/Polygon%20Mesh%20Modeling.pdf>

Render a stl object from an html file.

<https://stackoverflow.com/questions/12880980/need-js-and-html-example-for-displaying-stl-3d-objects-in-a-web-page>