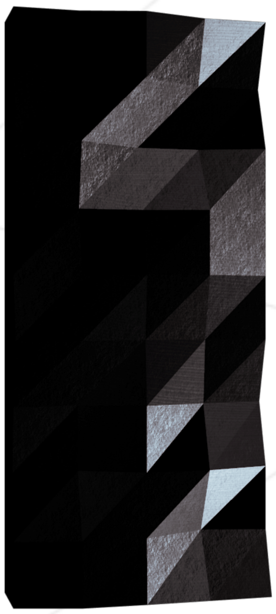

SCOP

Basic GPU rendering with OpenGL—or other GPU rendering APIs

Summary: This mini project is a first step towards using OpenGL— or other GPU rendering APIs.

Version: 4.1

Contents

Chapter I

Foreword

It's really because we've never done that one yet!

Chapter II

Subject

II.1 Feeling good doesn't hurt.

Sometimes, it just feels good to feel good. So let's make a little app that makes us feel great.

Also, this project will give you a nice workout—intellectually speaking. So yes, there will be some restrictions to make things interesting.

II.2 What You Need to Do

Your mission is to build a small program that displays a 3D object created using a modeling tool like Blender.

This 3D object will be stored in a `.obj` file. You'll be responsible for parsing it and rendering it properly.

Inside a window, the 3D object must be displayed in perspective (i.e., things that are far away look smaller), and it should rotate around its main axis of symmetry (basically, its center).

Using various colors, you should make the different faces of the object visually distinguishable. The object must be movable along all three axes, in both directions.

Finally, pressing a dedicated key should apply a texture to the object. Pressing that same key again should toggle it back to the colored view. A smooth transition between the two is expected—no hard cuts.

Here are the technical constraints:

- You're free to use any language (C / C++ / Rust preferred).
- You can choose between OpenGL, Vulkan, or Metal.
- Provide a classic Makefile (you know the drill).
- External libraries are allowed only for window and event management (e.g., MinilibX).
- No libraries are allowed for loading the 3D object, creating matrices, or loading shaders—you'll have to do that yourself.

Since this project is a bit of a self-pat on the back, it's essential that, during the defense, you can showcase the 42's logo (provided in the resources), spinning around its central axis (not around one of its corners!).

The sides should have subtle shades of gray, and the texture should be something cheerful—like ponies, kittens, or unicorns. Your choice.

`During the defense, expect to test additional 3D objects too.`

Chapter III

Bonus

Here are a few bonus ideas:

- Proper handling of tricky `.obj` files—like concave or non-coplanar ones. The teapot included in the resources comes in two versions: one is the original with some quirky edge effects; the other has been re-exported from Blender (no manual tweaking), just slightly normalized by the software. Try to render the first one correctly.
- Apply textures more subtly—no ugly stretching on the sides!
- We're sure you can think of even more awesome features to add.

`The bonus section will only be evaluated if the mandatory part is PERFECT. "Perfect" means every single requirement has been completed and works flawlessly. If you miss even one mandatory feature, the bonus won't be considered at all.`

Chapter IV

Submission and Peer-Evaluation

Submit your work in your `Git` repository as usual. Only what's in the repo will be reviewed during the defense. Double-check that your folder and file names are correct!

Chapter V

Beautifulz

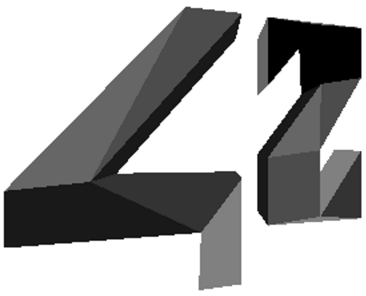

Figure V.1: The 42 logo with different colors depending on sides

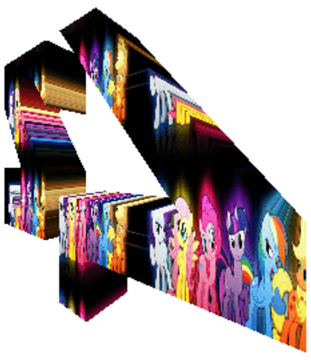

Figure V.2: From the back, with a texture

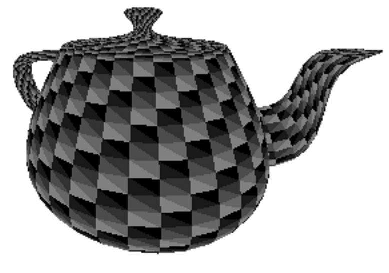

Figure V.3: The teapot