HfK Bremen | Temporary Spaces Workshop | WS 2021/2022

Optimising 3D Objects for Web

Quick introduction into optimising yours 3D creations for the web and deploying them.

Small Introduction

Why bother?

Optimisation of 3D models is becoming ever so crucial to AR (augmented reality) and web-based 3D experiences.

Loading speed
Devices limitations
Ecological reasons
…

3D optimisation

Simply put, 3D optimisation is the process of **reducing the file size** of 3D models. The size of 3D models comes from two places: geometry and textures.

Geometry

Geometry refers to the actual mesh or structure of the 3D model which is made up of polygons.

Textures

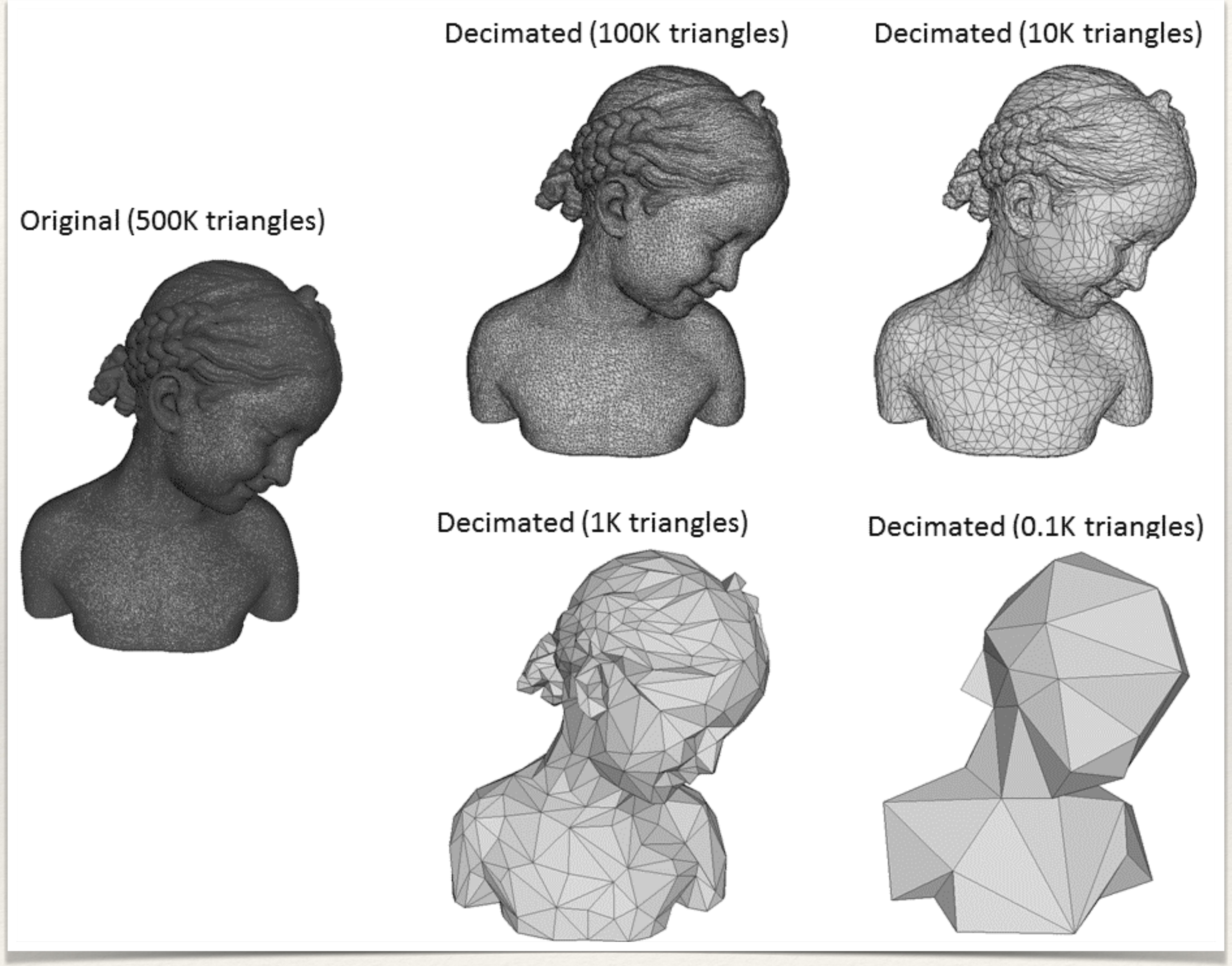
Textures are simply the images used to skin the 3D model.

Optimising a 3D model consists of two parts: mesh decimation and texture compression.

Optimising geometry

Mesh Decimation

The mesh decimation portion of 3D optimisation removes or combines polygons to reduce the overall polygon count. This reduces the overall file size and requires less GPU to render.


Original (500K triangles)
Decimated (100K triangles)
Decimated (10K triangles)
Decimated (1K triangles)
Decimated (0.1K triangles)

Optimising textures

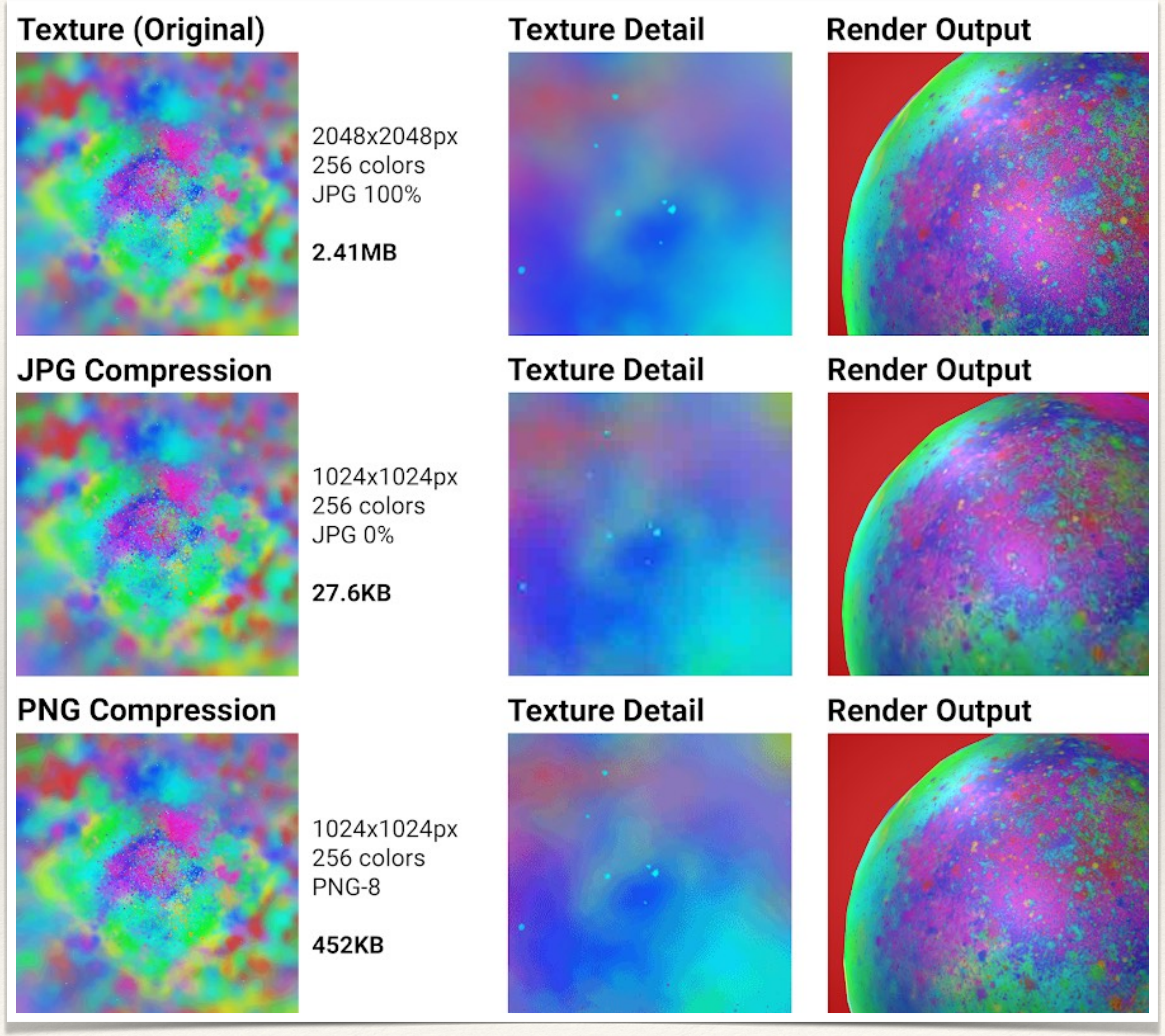
Texture Optimisation

Resizing

First, textures are resized to adjust the image's dimensions based on the amount of space they occupy. Often times, textures are much large than they need to be.

Compression

One example of how texture compression reduces files sizes is by grouping similar colours together to use fewer colours overall.



Texture (Original)

2048x2048px
256 colors
JPG 100%

2.41MB

Texture Detail

Render Output

JPG Compression

1024x1024px
256 colors
JPG 0%

27.6KB

Texture Detail

Render Output

PNG Compression

1024x1024px
256 colors
PNG-8

452KB

Texture Detail

Render Output

glTF 2.0 to the rescue!

glTF minimizes the size of 3D assets, and the runtime processing needed to unpack and use them.

.gltf

produces a JSON text-based file describing the overall structure, along with a **.bin** file containing mesh and vector data, and optionally a number of **.png or .jpg** files containing image textures

.glb

is a binary form of glTF that includes textures instead of referencing them as external image

Pick the strategy that you like most and feel comfortable with!

Let’s try this hands-on!