We are going to use door textures from Joao Paulo

3dtextures.me

**Color (or albedo)**

The albedo texture is the most simple one. It'll only take the pixels of the texture and apply them to the geometry.

A picture containing cabinet, wooden, wood

Description automatically generated

**albedo)**

**Alpha**

The alpha texture is a grayscale image where white will be visible, and black won't.

Shape, square

Description automatically generated

**Height**

The height texture is a grayscale image that will move the vertices to create some relief. You'll need to add subdivision if you want to see it.

Shape

Description automatically generated

**Normal**

The normal texture will add small details. It won't move the vertices, but it will lure the light into thinking that the face is oriented differently. Normal textures are very useful to add details with good performance because you don't need to subdivide the geometry.

A picture containing chart

Description automatically generated

**Ambient occlusion**

The ambient occlusion texture is a grayscale image that will fake shadow in the surface's crevices. While it's not physically accurate, it certainly helps to create contrast.

A picture containing text, building, door

Description automatically generated

**Metalness**

The metalness texture is a grayscale image that will specify which part is metallic (white) and non-metallic (black). This information will help to create reflection.

Graphical user interface, application

Description automatically generated with medium confidence

**Roughness**

The roughness is a grayscale image that comes with metalness, and that will specify which part is rough (white) and which part is smooth (black). This information will help to dissipate the light. A carpet is very rugged, and you won't see the light reflection on it, while the water's surface is very smooth, and you can see the light reflecting on it. Here, the wood is uniform because there is a clear coat on it.

A picture containing shape

Description automatically generated

### PBR

Those textures (especially the metalness and the roughness) follow what we call PBR principles. PBR stands for Physically Based Rendering. It regroups many techniques that tend to follow real-life directions to get realistic results.

While there are many other techniques, PBR is becoming the standard for realistic renders, and many software, engines, and libraries are using it.

For now, we will simply focus on how to load textures, how to use them, what transformations we can apply, and how to optimize them. We will see more about PBR in later lessons, but if you're curious, you can learn more about it here:

* <https://marmoset.co/posts/basic-theory-of-physically-based-rendering/>
* <https://marmoset.co/posts/physically-based-rendering-and-you-can-too/>

### Getting the URL of the image

To load the texture, we need the URL of the image file.

Because we are using Webpack, there are two ways of getting it.

You can put the image texture in the **/src/** folder and import it like you would import a JavaScript dependency:

Or you can put that image in the **/static/** folder and access it just by adding the path of the image (without **/static**) to the URL:

const imageSource = '/image.png'

console.log(imageSource)

**JavaScript**

Copy

Be careful, this **/static/** folder only works because of the Webpack template's configuration. If you are using other types of bundler, you might need to adapt your project.

We will use the **/static/** folder technique for the rest of the course.

### Loading the image

You can find the door textures we just saw in the **/static/** folder, and there are multiple ways of loading them.

With native JavaScript, first, you must create an **Image** instance, listen to the **load** event, and then change its **src** property to start loading the image:

const image = new Image()

image.onload = () =>

{

console.log('image loaded')

}

image.src = '/textures/door/color.jpg'

**JavaScript**

Copy

You should see **'image loaded'** appears in the console. As you can see, we set the source to **'/textures/door/color.jpg'** without the **/static** folder in the path.

We cannot use that image directly. We need to create a [Texture](https://threejs.org/docs/index.html#api/en/textures/Texture) from that image first.

This is because WebGL needs a very specific format that can be access by the GPU and also because some changes will be applied to the textures like the mipmapping but we will see more about that a little later.

Create the texture with the [Texture](https://threejs.org/docs/#api/en/textures/Texture) class:

Textures

UV unwrapping

colorTexture.magFilter = THREE.NearestFilter

use this to make images that are too small appear sharp.

And better performance and better frame rate

* [poliigon.com](http://poliigon.com/)
* [3dtextures.me](http://3dtextures.me/)
* [arroway-textures.ch](http://arroway-textures.ch/)

find textures here or can create them yourself on

substance3d.com