

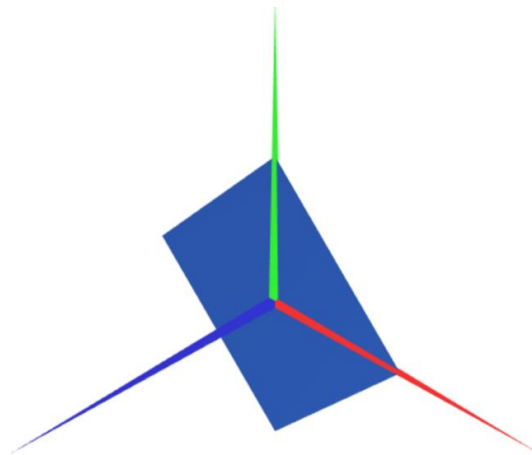
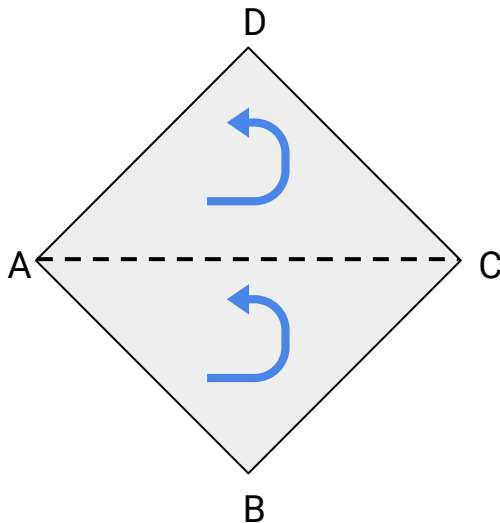
# TP4 - Textures

Concepts and Practice

# Previous lessons

We have learnt how to create geometries using **vertices** and **indices**

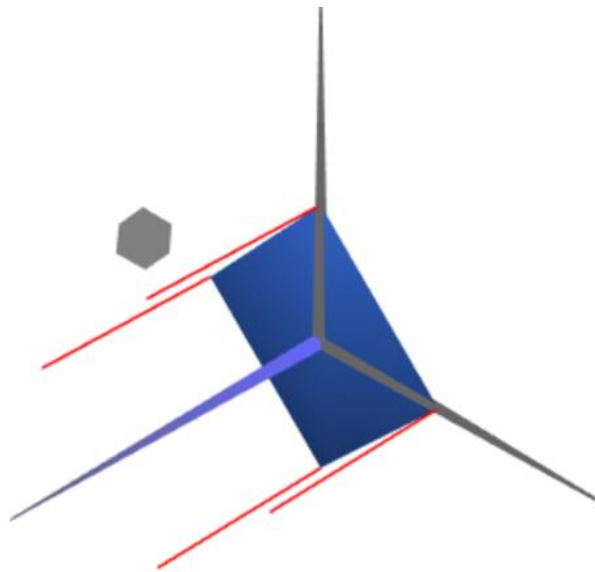
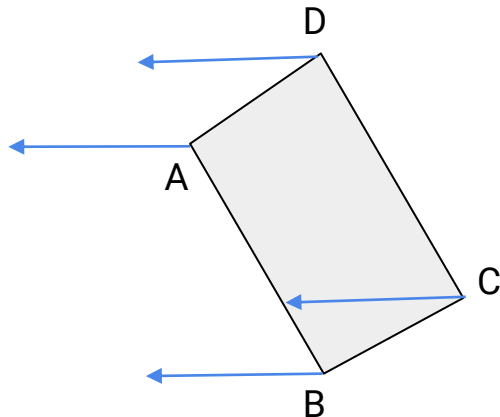
```
this.vertices = {  
  xA,yA,zA,  
  xB,yB,zB,  
  xC,yC,zC,  
  xD,yD,zD  
}  
  
this.indices = {  
  0, 1, 2,  
  0, 2, 3  
}
```



# Previous lessons

We defined **normals** for more realistic lighting of the geometries

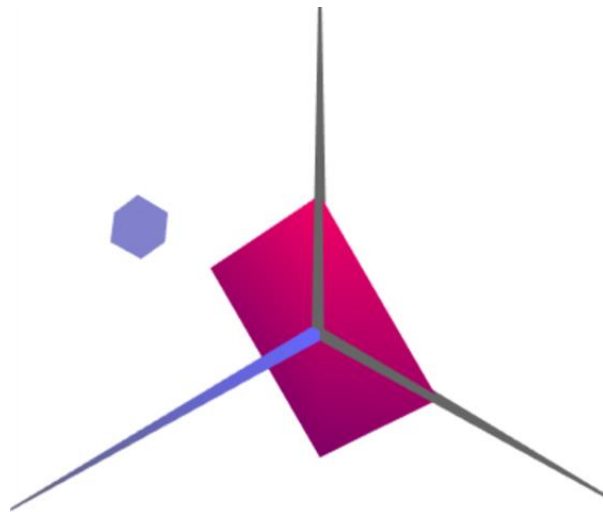
```
this.vertices = {...}  
  
this.indices = {...}  
  
this.normals = {  
  nAx,nAy,nAz,  
  nBx,nBy,nBz,  
  nCx,nCy,nCz,  
  nDx,nDy,nDz  
}
```



# Previous lessons

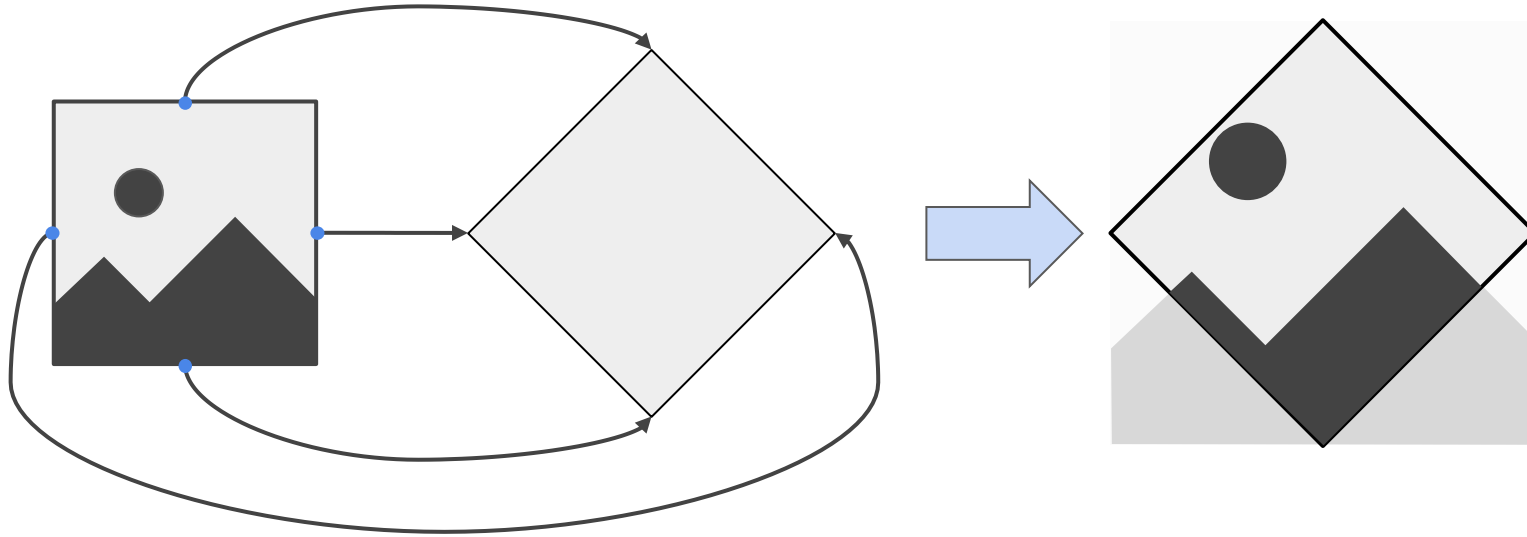
We applied **materials** to change the appearance and color of the geometries

```
init(){  
  ...  
  this.material = new CGFappearance(scene)  
}  
  
display(){  
  ...  
  this.material.apply()  
  this.object.display()  
}
```



# Application of textures

Another fundamental part of computer graphics - **textures**



# Application of textures in WebGL

In **WebGL**, images can be loaded to be used as textures for the geometries.

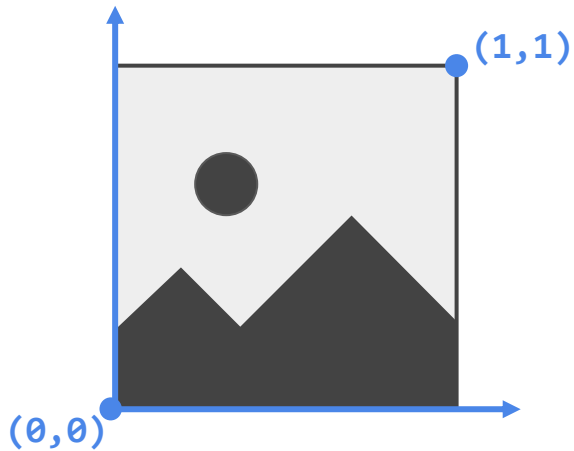
The process of **mapping a texture** to a geometry involves:

- 1 Define the **texture coordinates** for the geometry
- 2 **Load an image** into a texture
- 3 **Apply texture** before drawing geometry

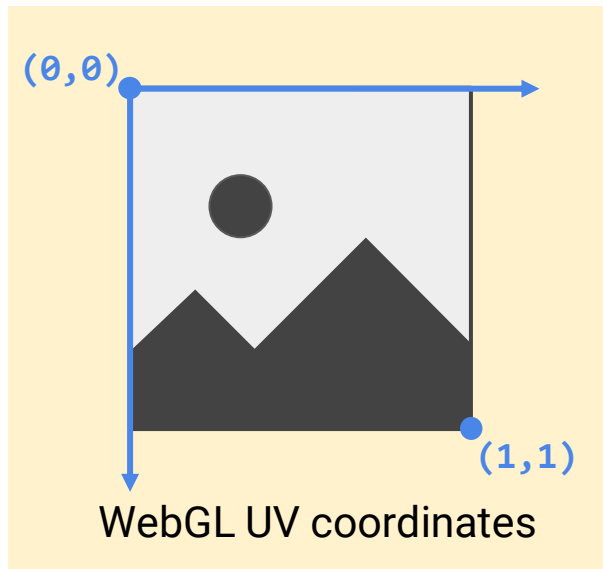
# 1 Texture coordinates

Textures use normalized  $(u,v)$  coordinates

A pair of  $(u,v)$  coordinates represents an **image pixel** (texel)



OpenGL UV coordinates




WebGL UV coordinates

# 1 Texture coordinates in WebGL/WebCGF

For each **vertex** of a geometry, a pair of texture coordinates is defined

In **object space**, texture coordinates are named (**s**, **t**)

The texture coordinates are defined in a ***texCoords*** array



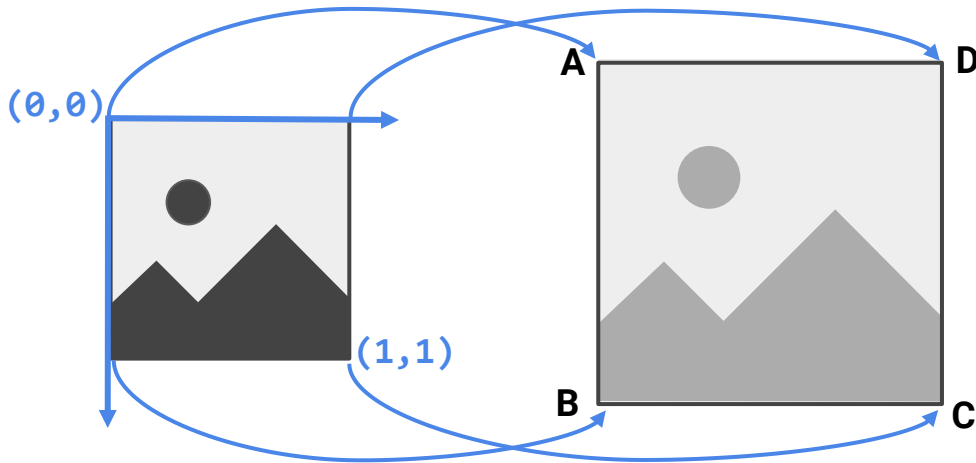
```
this.vertices = {  
  xA, yA, zA,  
  xB, yB, zB,  
  xC, yC, zC,  
  xD, yD, zD  
}  
  
this.texCoords = {  
  sA, tA,  
  sB, tB,  
  sC, tC,  
  sD, tD  
}
```



# 1 Texture coordinates - Example

**Basic example:** Mapping a square texture to a square object

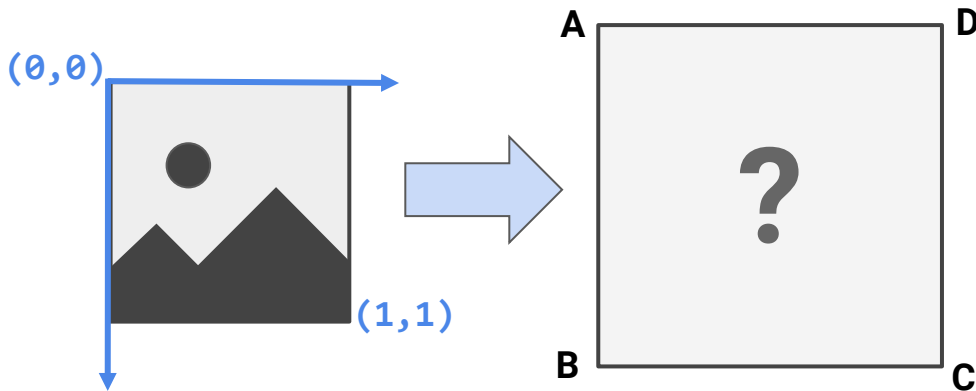
```
this.vertices = {...}  
  
this.texCoords = {  
    0, 0, //A  
    0, 1, //B  
    1, 1, //C  
    1, 0 //D  
}
```



# 1 Texture coordinates - Example

**Question:** What happens when texture coordinates are out of [0-1] bounds?

```
this.vertices = {...}  
  
this.texCoords = {  
    0, 0, //A  
    0, 1, //B  
    2, 1, //C  
    2, 0, //D  
}
```

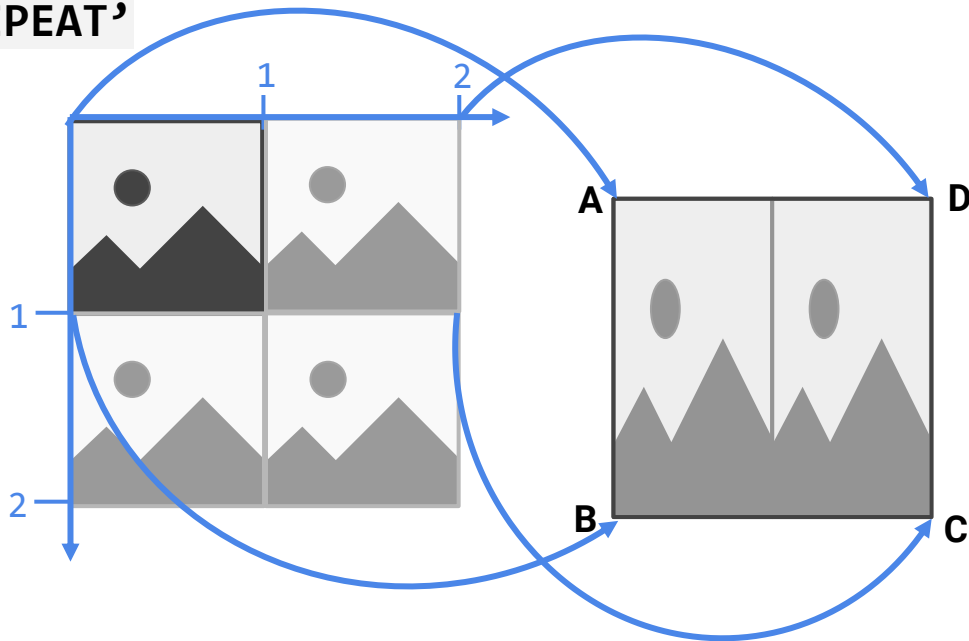


# 1 Texture coordinates - Wrap Mode

Defines how texture is sampled for coordinates out of [0-1] range

By default, wrap mode is **'REPEAT'**

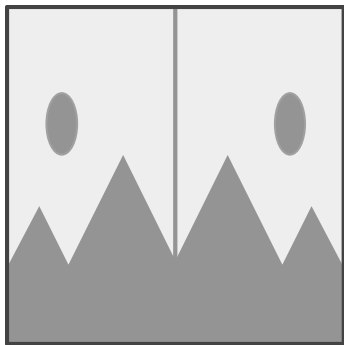
```
this.vertices = {...}  
  
this.texCoords = {  
    0, 0, //A  
    0, 1, //B  
    2, 1, //C  
    2, 0  //D  
}
```



# 1 Texture coordinates - Wrap Mode

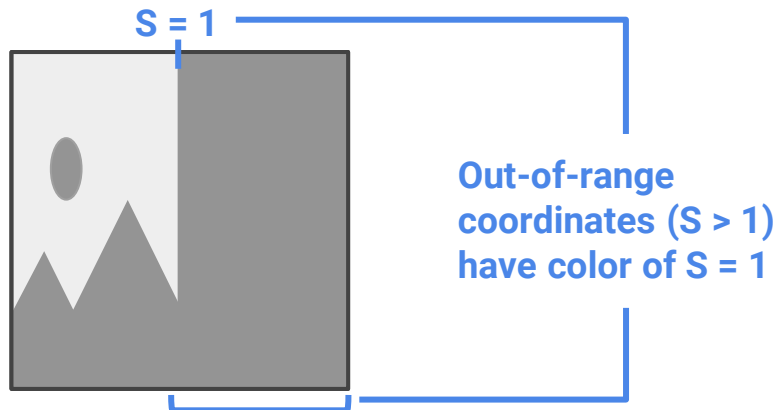
Wrap mode is applied to **each axis separately**

The other wrap modes are **'MIRRORED\_REPEAT'** and **'CLAMP\_TO\_EDGE'**



**'MIRRORED\_REPEAT'**

Texture is mirrored in S axis



**'CLAMP\_TO\_EDGE'**

Out-of-range coordinates  
copy margin values

## 2 Load image to texture with WebCGF

The **WebCGF** library has a class for textures - **CGFtexture**

```
new CGFtexture(scene, url)
```

Textures may be **associated to materials**

```
material.setTexture(CGFtexture)
```

Images may also be **loaded directly** to materials

```
CGFappearance.loadTexture(url)
```

### 3 Apply texture before drawing object

Textures may be applied in two different ways:

**Indirectly**, associating it to a material

```
material.setTexture(texture)
...
mat.apply()
object.display()
```

The **CGFappearance** class also provides a function to define **wrap mode**

```
setTextureWrap(wrapS, wrapT)
```

### 3 Apply texture before drawing object

Textures may be applied in two different ways:

**Directly**, by binding it to the WebGL context

```
texture.bind()  
object.display()
```

This requires the use of WebGL functions to define **wrap mode**

```
gl.texParameter[fi](target, pname, param)
```

Documentation for texParameter function [\[link\]](#)

# Additional texture parameters

Other texture parameters may be defined with *texParameter()* function

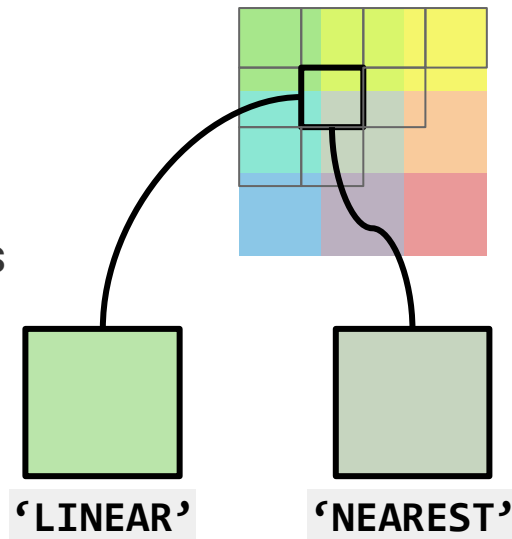
**Texture filtering** mode can be changed to adapt to current requirements

**Example:** Small texture and a large object

1+ pixel covers 1 texel (magnification)

**Linear filtering** blends colors of 4 closest texels

**Nearest filtering** uses color of closest texel





# Documentation and guides

CGFtexture documentation

<https://paginas.fe.up.pt/~ruirodrig/pub/sw/webcgf/docs/#!/api/CGFtexture>

CGFappearance documentation

<https://paginas.fe.up.pt/~ruirodrig/pub/sw/webcgf/docs/#!/api/CGFappearance>

WebGL texture parameter function

<https://developer.mozilla.org/en-US/docs/Web/API/WebGLRenderingContext/textParameter>

WebGL textures tutorial

<https://webglfundamentals.org/webgl/lessons/webgl-3d-textures.html>

Texture wrap mode demonstrator

<https://webglfundamentals.org/webgl/webgl-3d-textures-repeat-clamp.html>