COMP3320 Introduction to OpenGL

Alex Biddulph

The University of Newcastle, Australia

Based on the work provided at `www.learnopengl.com`

Semester 2, 2019

Vertex Attributes

- Allows us to specify auxilliary data for each vertex
- Colour, texture coordinates, etc.
- An example specifying vertex colour information

```
float vertices[] = {
    // positions          // colors
     0.5f, -0.5f, 0.0f,  1.0f, 0.0f, 0.0f,
    -0.5f, -0.5f, 0.0f,  0.0f, 1.0f, 0.0f,
     0.0f,  0.5f, 0.0f,  0.0f, 0.0f, 1.0f };
```

- Must specify offset and stride for `glVertexAttribPointer`

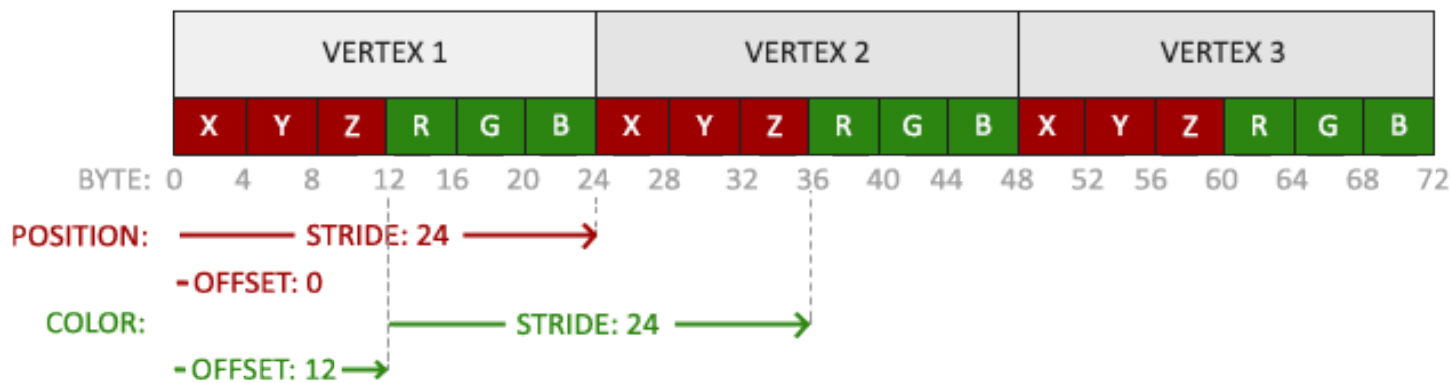
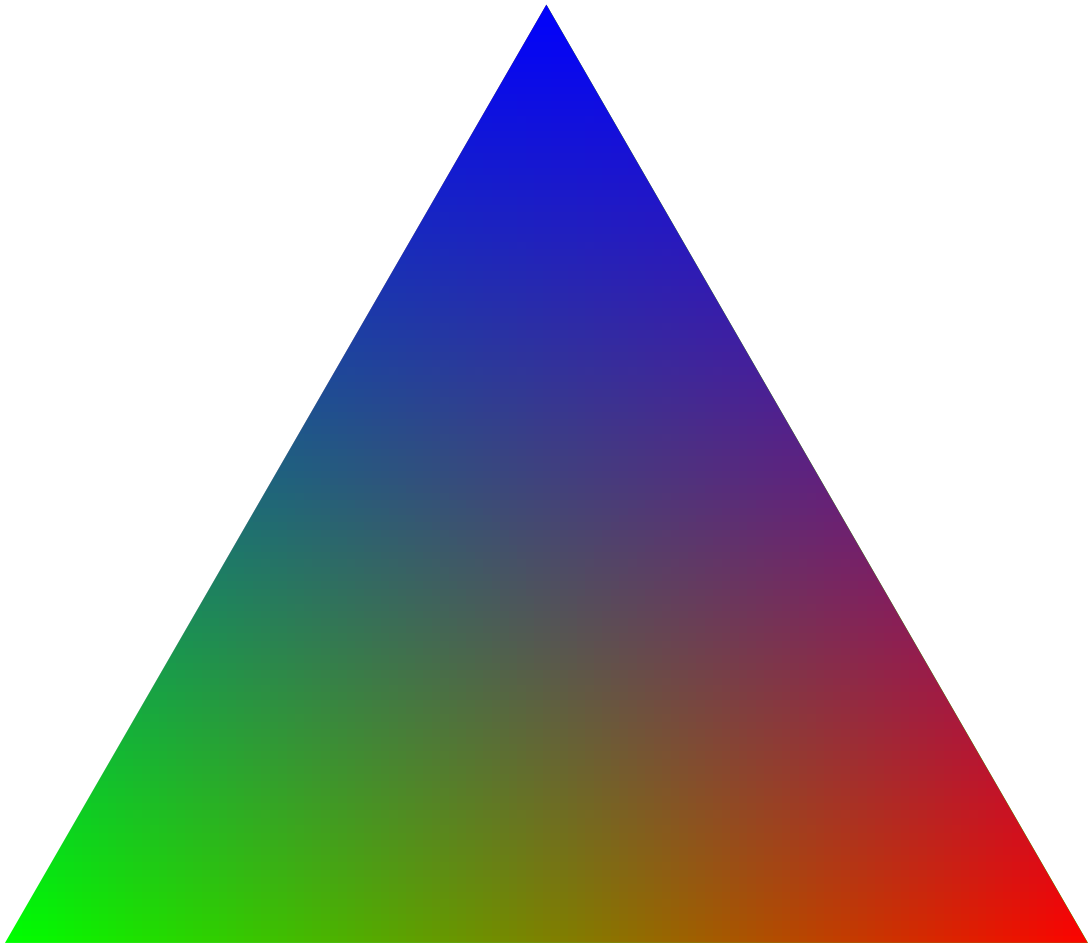


Image sourced from `learnopengl.com/Getting-started/Shaders`

Vertex Attributes

Result should look like this



Textures

- Rather than using colours to add detail to an object, use an image
- Easier to add a lot of detail to an object
- To apply a texture we just need to assign texture coordinates to each vertex

```
float vertices[] = {
    // positions          // colors           // textures
     0.5f, -0.5f, 0.0f,  1.0f, 0.0f, 0.0f, 1.0f, 0.0f,
    -0.5f, -0.5f, 0.0f,  0.0f, 1.0f, 0.0f, 0.0f, 0.0f,
     0.0f,  0.5f, 0.0f,  0.0f, 0.0f, 1.0f, 0.5f, 1.0f };
```

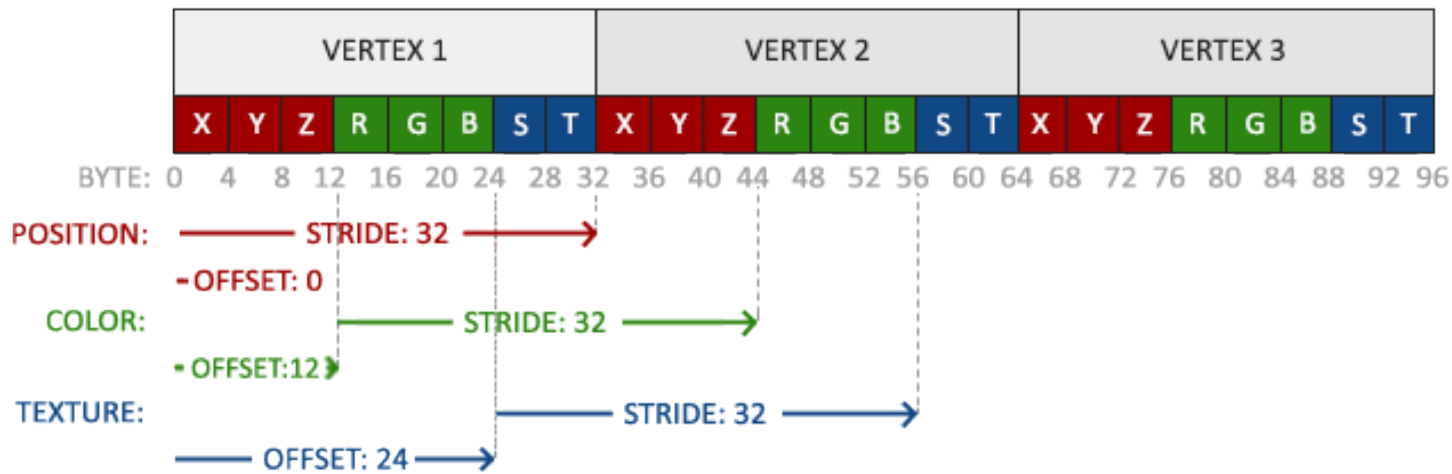


Image sourced from `learnopengl.com/Getting-started/Textures`

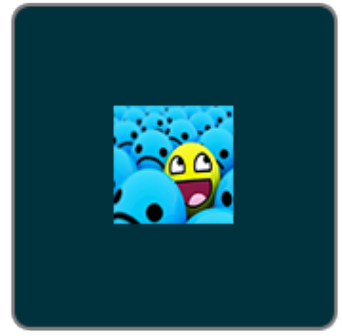
Texture Wrapping

- Texture coordinates range from $(0, 0) \rightarrow (1, 1)$
- What should happen if coordinates outside this range are specified?


GL_REPEAT


GL_MIRRORED_REPEAT


GL_CLAMP_TO_EDGE

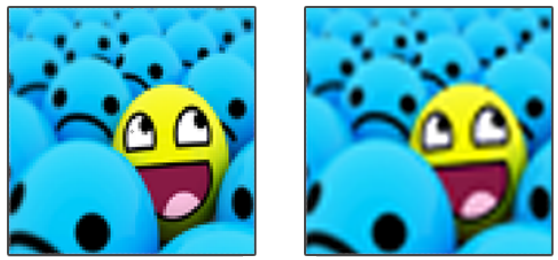

GL_CLAMP_TO_BORDER

- Specify behaviour using `glTexParameteri`
- Specify border colour using `glTexParameterfv`

Image sourced from `learnopengl.com/Getting-started/Textures`

Texture Filtering

- Floating-point texture coordinates are mapped to integer pixel coordinates
- What should happen if texture coordinates have a fractional component?
  - For example, texture coordinates $(0.75, 0.0)$ maps to pixel coordinates $(480.3, 300)$



- Specify behaviour using `glTexParameteri`
- Behaviour can be specified for both minifying and magnifying operations

Image sourced from `learnopengl.com/Getting-started/Textures`

MipMaps

- No need to use a high resolution image to texture an object a long distance away
- Can also result in undesirable artifacts on small objects
- The solution?
  - Create multiple scaled down versions of the high resolution image
  - Select a different scaled down texture based on the distance from the camera

- Generating mipmaps is cumbersome, use `glGenerateMipmaps`
- Can change mipmap filtering behaviour for minifying and magnifying operations as well

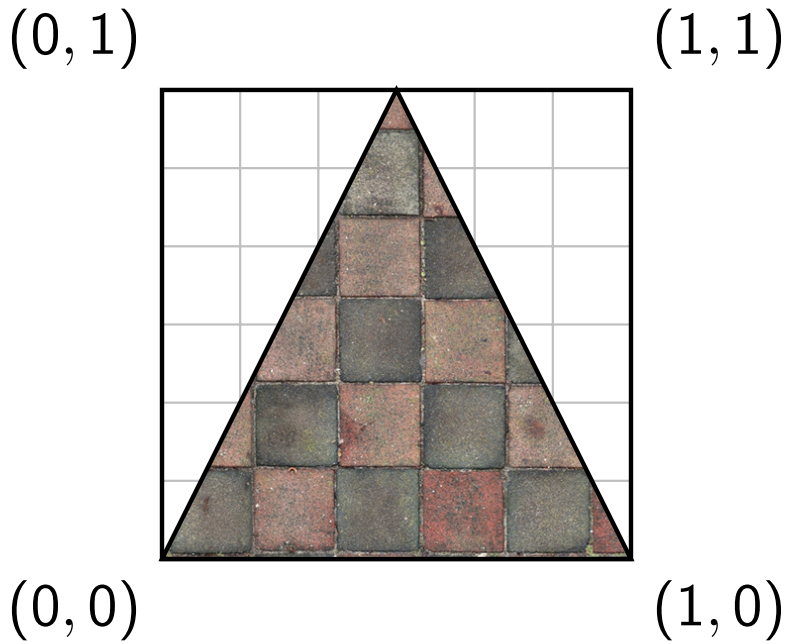
Image sourced from `learnopengl.com/Getting-started/Textures`

Loading Textures

- A number of C/C++ libraries available for loading images
- SOIL is a common library specifically targetting OpenGL
- Use `glGenTexture` to generate a texture object
- Use `glBindTexture` to bind a texture object and make it active, be sure to do this before rendering objects
- Use `glTexImage2D` to attach the raw texture data to the texture object
- Generate mipmaps after attaching the raw texture data
- It is now safe to delete any pointers to the raw texture data
- It is possible to use multiple textures in a single program, use `glActiveTexture` to select which texture unit to use before binding
  - Should be a minimum of 16 texture units available

Textures

Result should look like this



Brick wall image sourced from

`learnopengl.com/Getting-started/Textures`