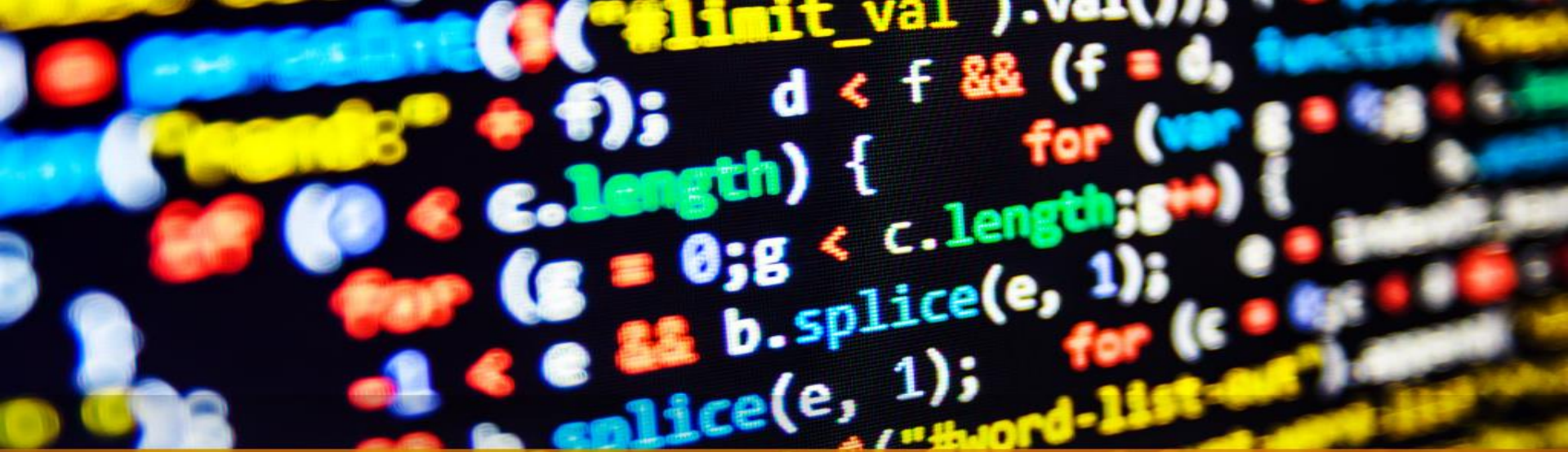
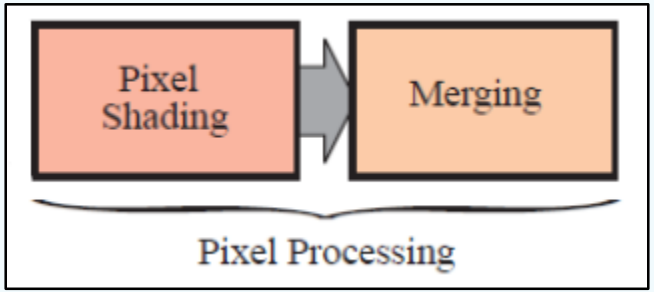


Lecture 10

The Merging Stage

Prepared by Ban Kar Weng (William)

The Pixel Processing Stage Revisited

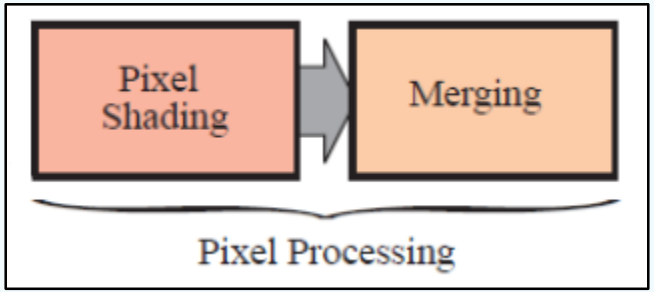


Pixel Shading

- Performs any per-pixel shading computation.
- In OpenGL, pixel shading is executed by fragment shader.
- **Example**: texturing

Image taken from Figure 6.9 of Real-time Rendering (4th edition)

The Pixel Processing Stage Revisited



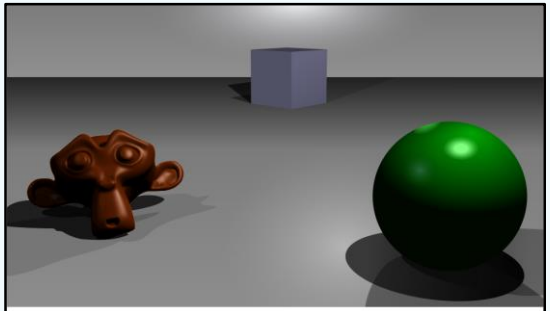
Merging

- Combine fragment colour with the colour currently in the colour buffer.
- Not fully programmable, but highly configurable.
- Some common operations:
  - z-buffering
  - blending

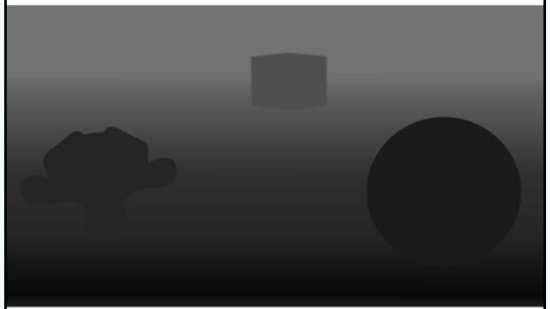
Image taken from Figure 6.9 of Real-time Rendering (4th edition)

Z-buffering

Z-buffering



A simple three-dimensional scene

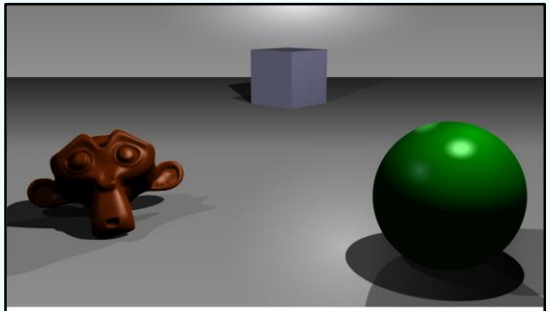


Z-buffer representation

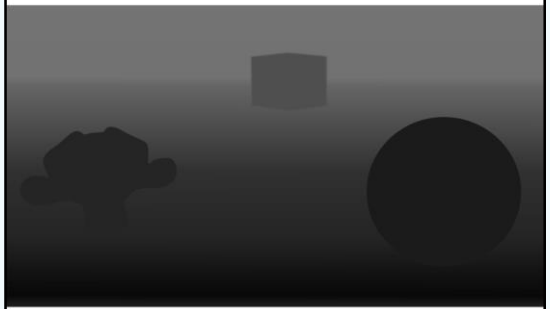
Introduction

- Purpose: resolve visibility
- When a scene is rendered, the colour buffer should contain the colours of visible primitives from the camera's view point.
- This is done with a z-buffer (a.k.a *depth buffer*).

Z-buffering



A simple three-dimensional scene

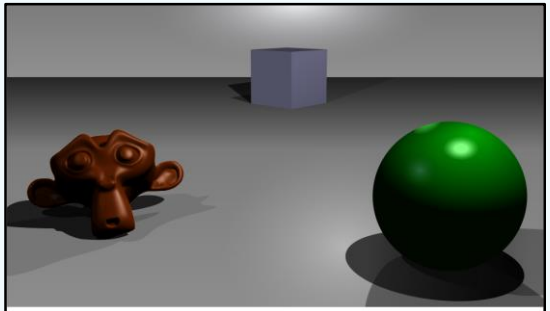


Z-buffer representation

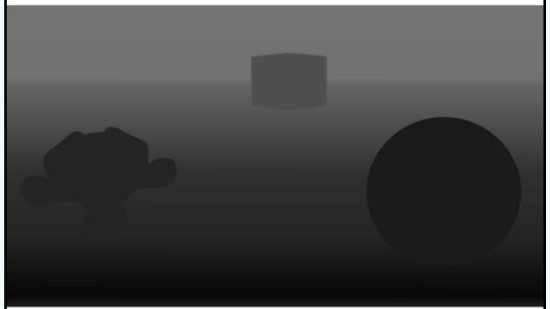
Z-buffer

- Has the same and shape as the colour buffer.
- Each pixel stores the z-value to the currently closest primitive.
- Z-value is just a single value.

Z-buffering



A simple three-dimensional scene

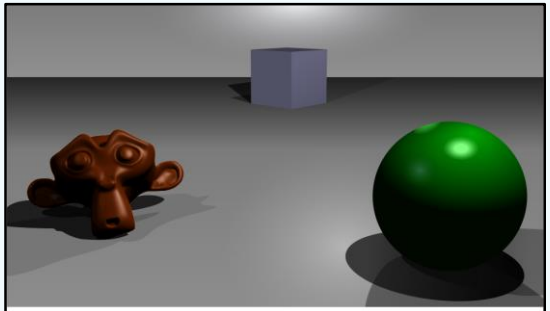


Z-buffer representation

Z-buffering

- The z-value on a rendered primitive at a pixel is computed and compared to the contents of the z-buffer at the same pixel.
- If **new z-value < z-value in z-buffer** at that pixel, update the pixel's z-buffer and colour.
- Otherwise, colour buffer and z-buffer is left untouched.

Z-buffering



A simple three-dimensional scene

Z-buffer representation

Pros:

- Simple
- Works for any primitives.
- Primitives can be drawn in any order.

Cons:

- Not straight-forward for partially transparent primitives. (either render them after all opaque primitives, or use a more complex order-independent algorithm)

All images are taken online. To get the link, right click the image and select “Edit Hyperlink” in the context menu.

Z-buffering

OpenGL Functions	Descriptions
`glEnable(GL_DEPTH_TEST)`	Enable z-buffering
`glClear(GL_DEPTH_BUFFER_BIT)`	Clear the z-buffer
`glDepthMask()`	Call this function with GL_FALSE if you want to perform depth test, but not update the z-buffer.
`glDepthFunc()`	Change the comparison operator used in z-buffering

Blending

Blending

Full transparent window

Partially transparent window

- Colour from the fragment shader can be blended with the colour in the colour buffer at the same pixel.
- Example: implement transparency within objects.

Blending | The Equation

Full transparent window

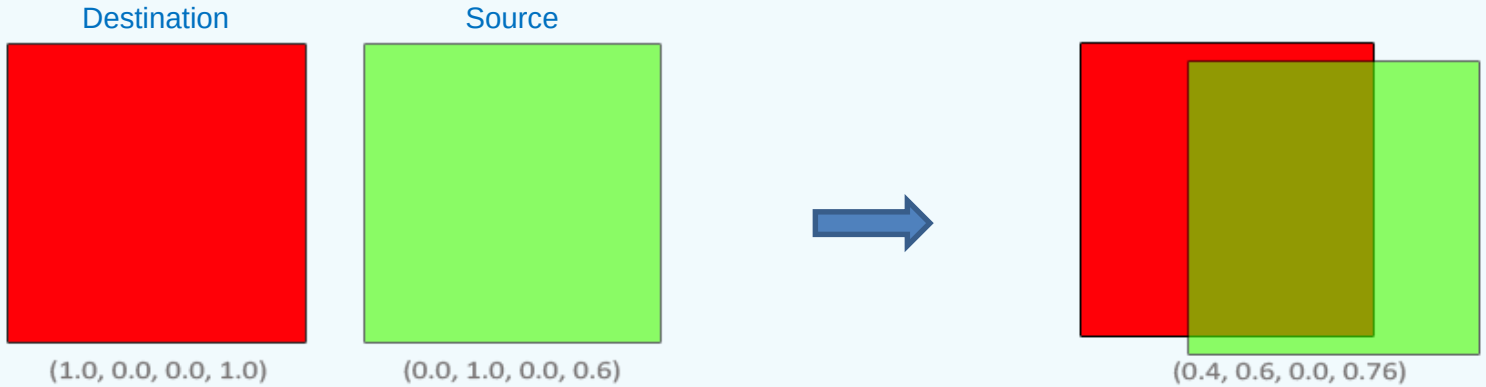
Partially transparent window

$$C = C_{src}F_{src} + C_{dest}F_{dest}$$

- C_{src} → the source colour vector (output of fragment shader).
- C_{dest} → the destination colour vector (colour currently in colour buffer).
- F_{src} → the source factor value.
- F_{dest} → the destination factor value.

Blending

Example: Alpha blending



$$C = C_{src}F_{src} + C_{dest}F_{dest}$$

$$C = \begin{pmatrix} 0.0 \\ 1.0 \\ 0.0 \\ 0.6 \end{pmatrix} 0.6 + \begin{pmatrix} 1.0 \\ 0.0 \\ 0.0 \\ 1.0 \end{pmatrix} (1 - 0.6)$$

Blending | OpenGL functions

OpenGL Functions	Descriptions
`glEnable(GL_BLEND)`	Enable blending
`glBlendFunc(GLenum sfactor, GLenum dfactor)`	Specify F_{src} and F_{dest} via `sfactor` and `dfactor` respectively.
`glBlendEquation(GLenum srcRGB, GLenum dstRGB, GLenum srcAlpha, GLenum dstAlpha)`	Similar to **`glBlendFunc()`** in purpose, but allows separate factors for RGB and alpha component.
`glBlendEquation(GLenum mode)`	Change the operator between source and destination part of the equation.

Double Buffering

Double Buffering

- The screen displays the contents of the colour buffer.
- With single buffering, human viewer could see the primitives as they are rendered and sent to the screen.
- With double buffering, rendering takes place off screen, in a back buffer.
- Once rendering on back buffer completes, the back buffer is swapped with front buffer.
- The swapping occurs during **vertical retrace**, a time when it's safe to do so.

Q & A

Acknowledgement

- This presentation has been designed using resources from PoweredTemplate.com