Introduction to Computer Graphics with WebGL

Ed Angel

Professor Emeritus of Computer Science

Founding Director, Arts, Research, Technology and Science Laboratory

University of New Mexico

Buffers

Ed Angel
Professor Emeritus of Computer Science
University of New Mexico

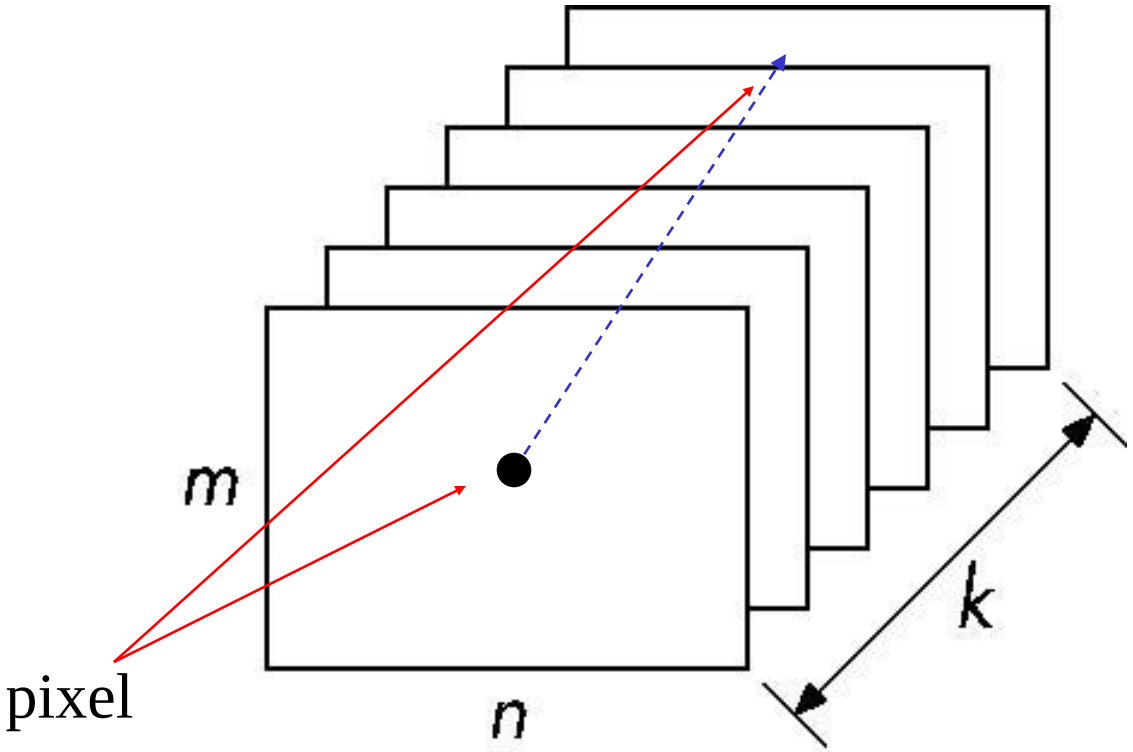
Objectives

- Introduce additional WebGL buffers
- Reading and writing buffers
- Buffers and Images

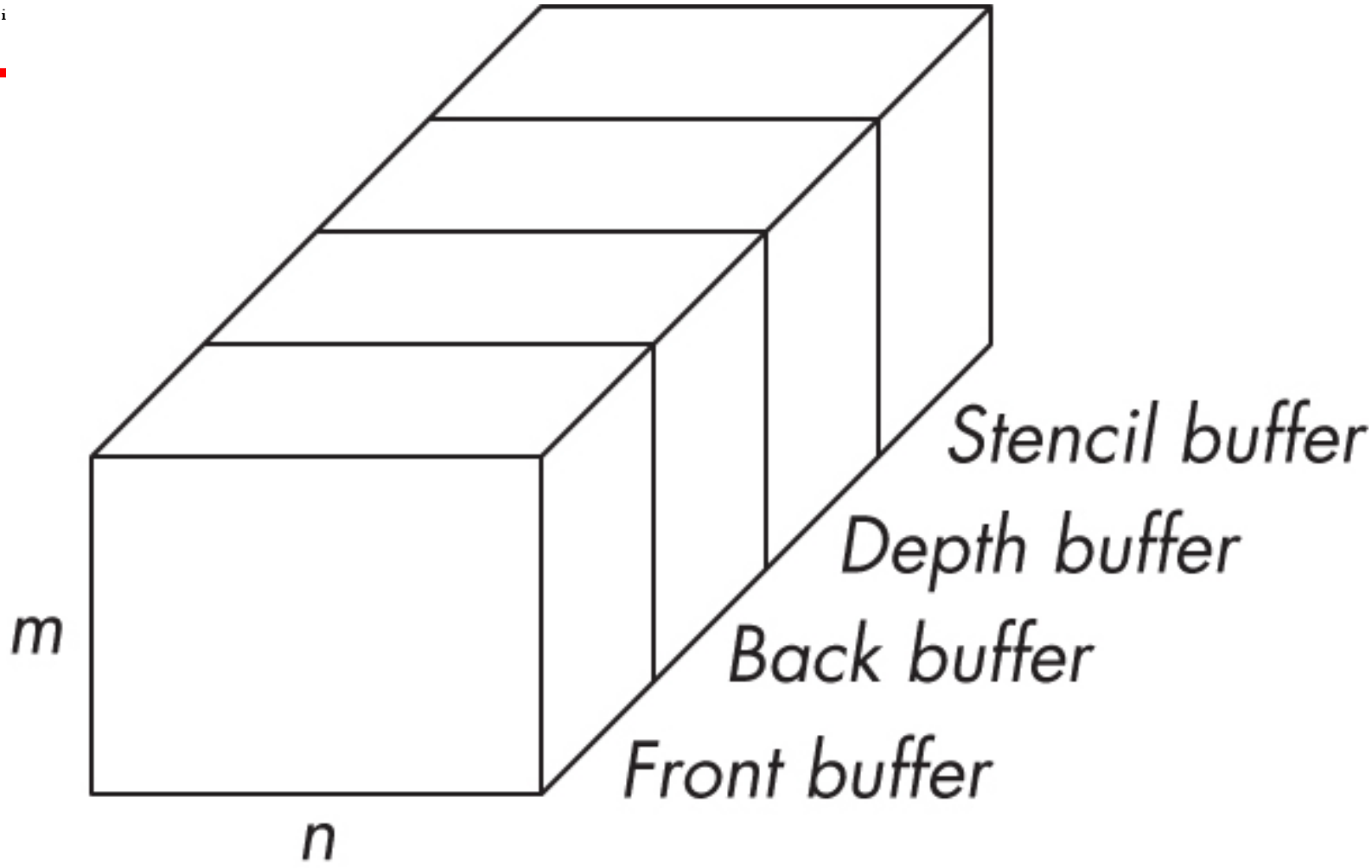
The University of New Mexico

Buffer

Define a buffer by its spatial resolution (n x m) and its depth (or precision) k, the number of bits/pixel



WebGL Frame Buffer



Where are the Buffers?

- HTML5 Canvas
  - Default front and back color buffers
  - Under control of local window system
  - Physically on graphics card
- Depth buffer also on graphics card
- Stencil buffer
  - Holds masks
- Most RGBA buffers 8 bits per component
- Latest are floating point (IEEE)

Other Buffers

- desktop OpenGL supported other buffers
  - auxiliary color buffers
  - accumulation buffer
  - these were on application side
  - now deprecated
- GPUs have their own or attached memory
  - texture buffers
  - off-screen buffers
    - not under control of window system
    - may be floating point

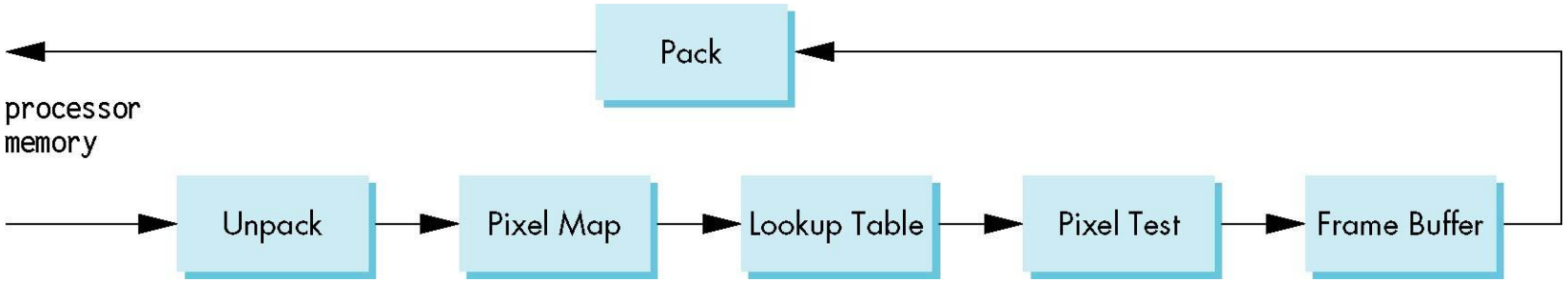
The University of New Mexico

Images

- Framebuffer contents are unformatted
  - usually RGB or RGBA
  - one byte per component
  - no compression
- Standard Web Image Formats
  - jpeg, gif, png
- WebGL has no conversion functions
  - Understands standard Web formats for texture images

The (Old) Pixel Pipeline

- OpenGL has a separate pipeline for pixels
  - Writing pixels involves
    - Moving pixels from processor memory to the frame buffer
    - Format conversions
    - Mapping, Lookups, Tests
  - Reading pixels
    - Format conversion



Packing and Unpacking

- Compressed or uncompressed
- Indexed or RGB
- Bit Format
  - little or big endian
- WebGL (and shader-based OpenGL) lacks most functions for packing and unpacking
  - use texture functions instead
  - can implement desired functionality in fragment shaders

Deprecated Functionality

- glDrawPixels
- glCopyPixels
- glBitMap

Buffer Reading

- WebGL can read pixels from the framebuffer with gl.readPixels
- Returns only 8 bit RGBA values
- In general, the format of pixels in the frame buffer is different from that of processor memory and these two types of memory reside in different places
  - Need packing and unpacking
  - Reading can be slow
- Drawing through texture functions and off-screen memory (frame buffer objects)

WebGL Pixel Function

```
gl.readPixels(x,y,width,height,format,type,myimage)
```

start pixel in frame buffer

size

type of image

type of pixels

pointer to processor memory

```
var myimage[512*512*4];

gl.readPixels(0,0, 512, 512, gl.RGBA,
      gl.UNSIGNED_BYTE, myimage);
```

Render to Texture

- GPUs now include a large amount of texture memory that we can write into
- Advantage: fast (not under control of window system)
- Using frame buffer objects (FBOs) we can render into texture memory instead of the frame buffer and then read from this memory
  - Image processing
  - GPGPU

Introduction to Computer Graphics with WebGL

Ed Angel

Professor Emeritus of Computer Science

Founding Director, Arts, Research, Technology and Science Laboratory

University of New Mexico

BitBlt

Ed Angel
Professor Emeritus of Computer Science
University of New Mexico

Objectives

- Introduce reading and writing of blocks of bits or bytes
- Prepare for later discussion compositing and blending

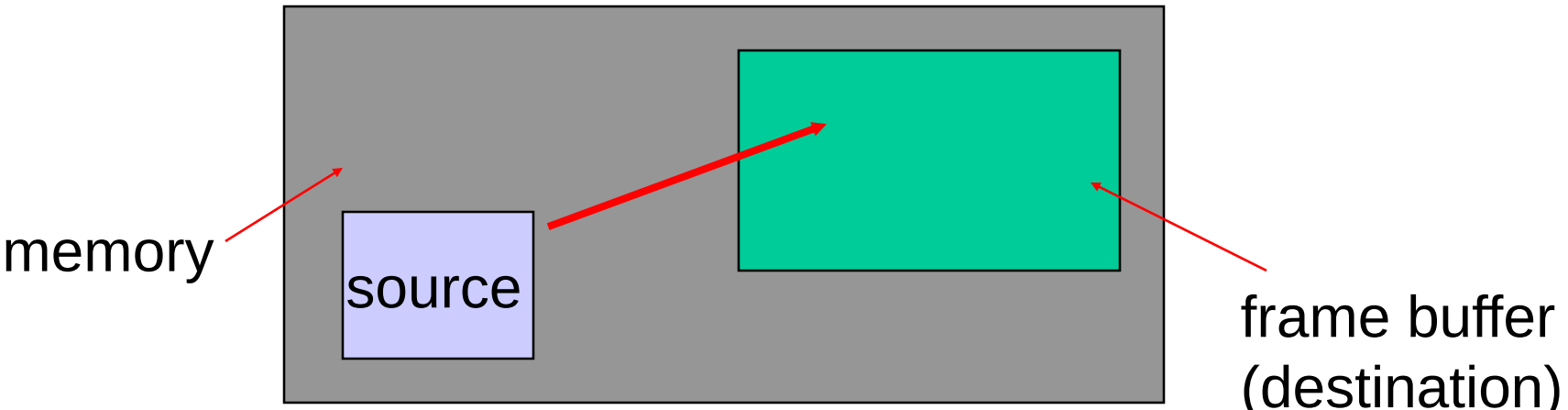
Writing into Buffers

- WebGL does not contain a function for writing bits into frame buffer
  - Use texture functions instead
- We can use the fragment shader to do bit level operations on graphics memory
- Bit Block Transfer (BitBlt) operations act on blocks of bits with a single instruction

The University of New Mexico

BitBlt

- Conceptually, we can consider all of memory as a large two-dimensional array of pixels
- We read and write rectangular block of pixels
- The frame buffer is part of this memory

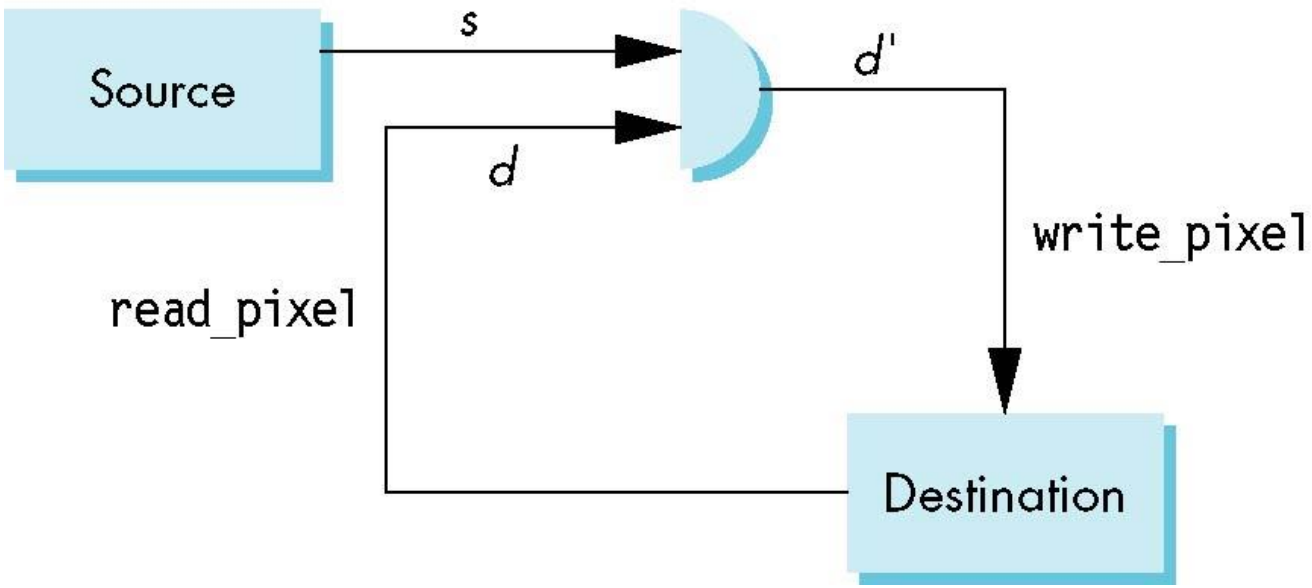


writing into the frame buffer

The University of New Mexico

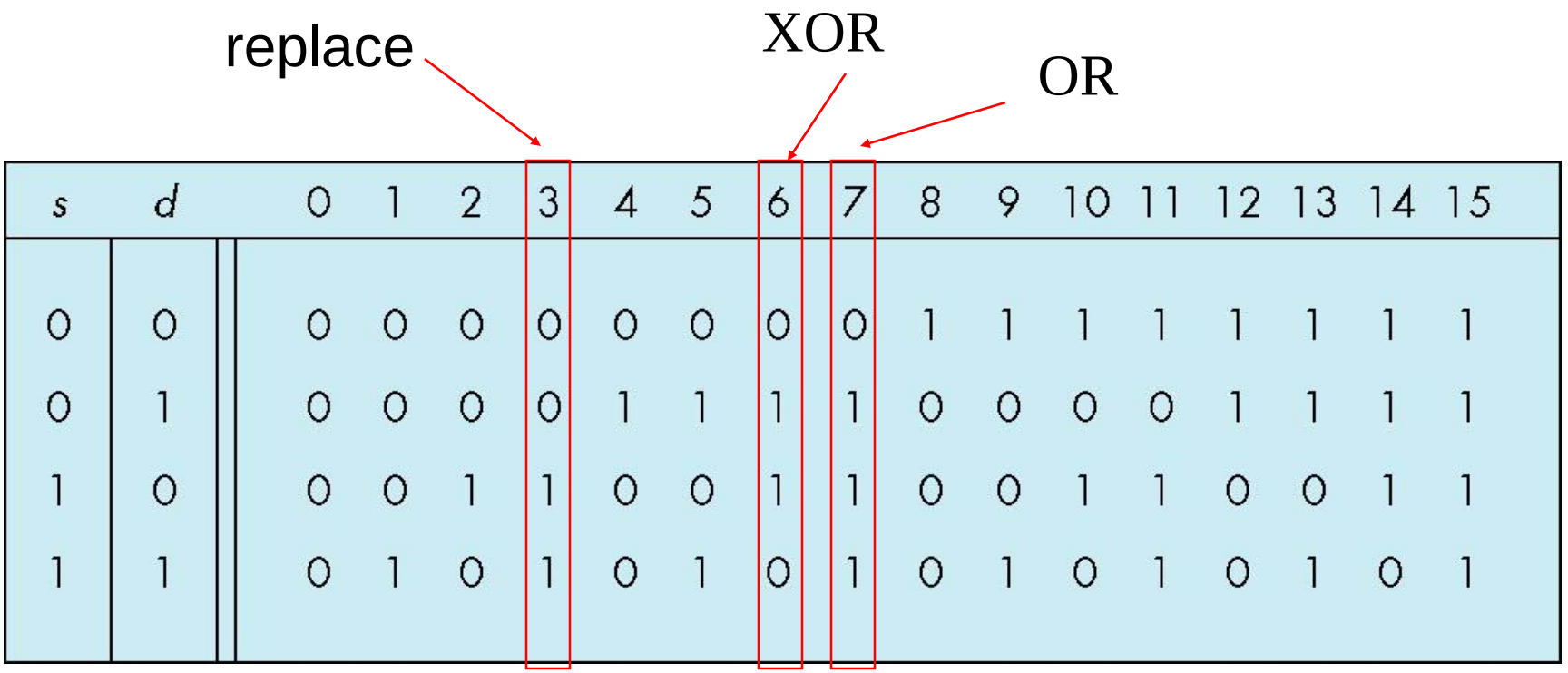
Writing Model

Read destination pixel before writing source



Bit Writing Modes

- Source and destination bits are combined bitwise
- 16 possible functions (one per column in table)



s	d	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

XOR mode

- XOR is especially useful for swapping blocks of memory such as menus that are stored off screen

  If S represents screen and M represents a menu the sequence

  $$S \leftarrow S \oplus M$$
  $$M \leftarrow S \oplus M$$
  $$S \leftarrow S \oplus M$$

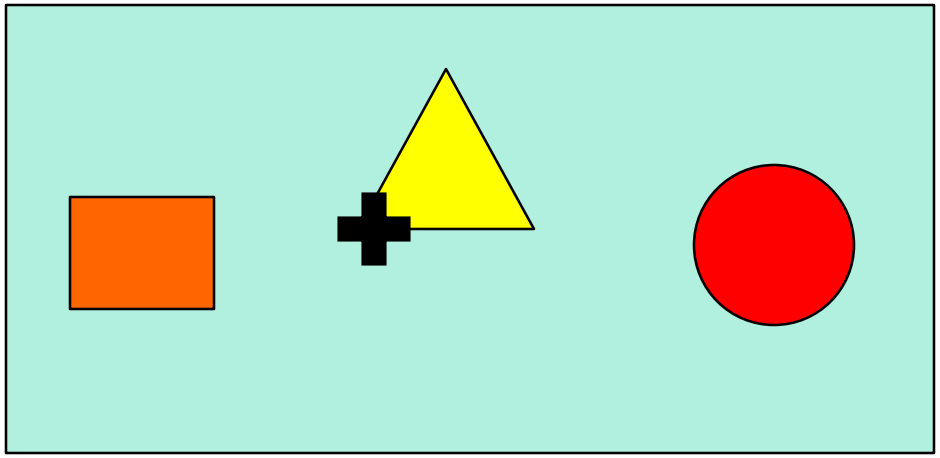
  swaps S and M

- Same strategy used for rubber band lines and cursors

The University of New Mexico

Cursor Movement

- Consider what happens as we move a cursor across the display
- We cover parts of objects
- Must return to original colors when cursor moves away



Rubber Band Line

- Fix one point
- Draw line to location of cursor
- Must return state of crossed objects when line moves