

Announcements

- Project 1 grading delayed.
- Homework 11 will be posted later today.
- Reading for Thursday: Shirley 3rd Edition
 - Chapter 10: Surface Shading
 - Chapter 11: Texture Mapping

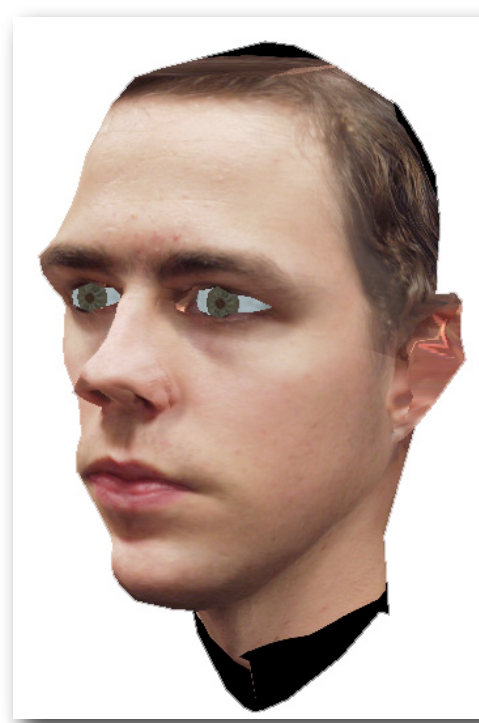
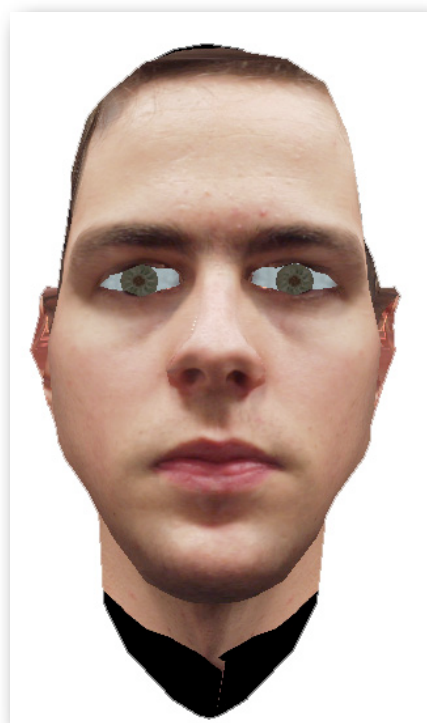
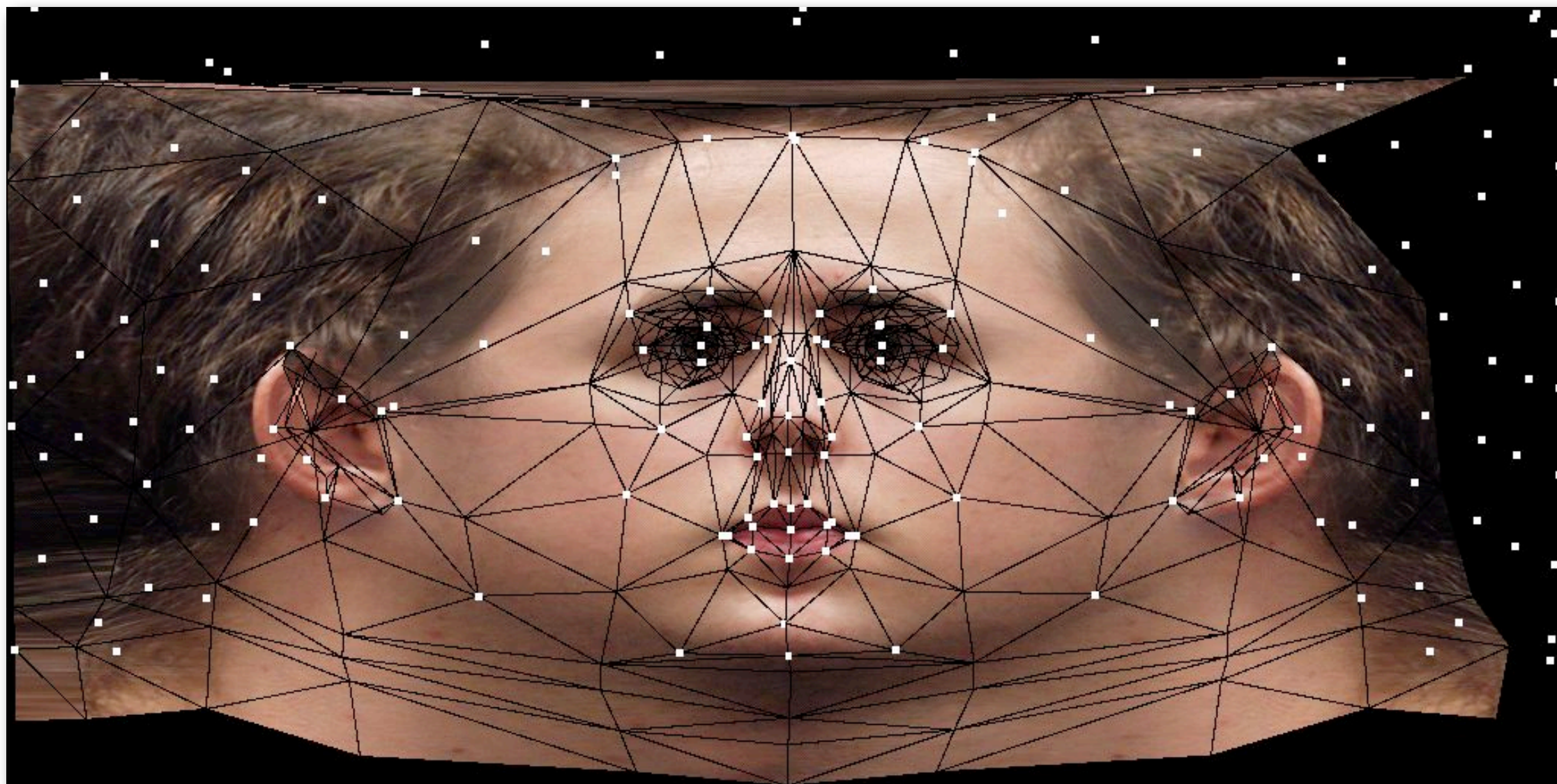
Basics of Textures

Basics of texture mapping in OpenGL

Texture Mapping

- A way of adding surface details
- Two ways can achieve the goal:
 - Model the surface with more polygons
 - » Slows down rendering speed
 - » Hard to model fine features





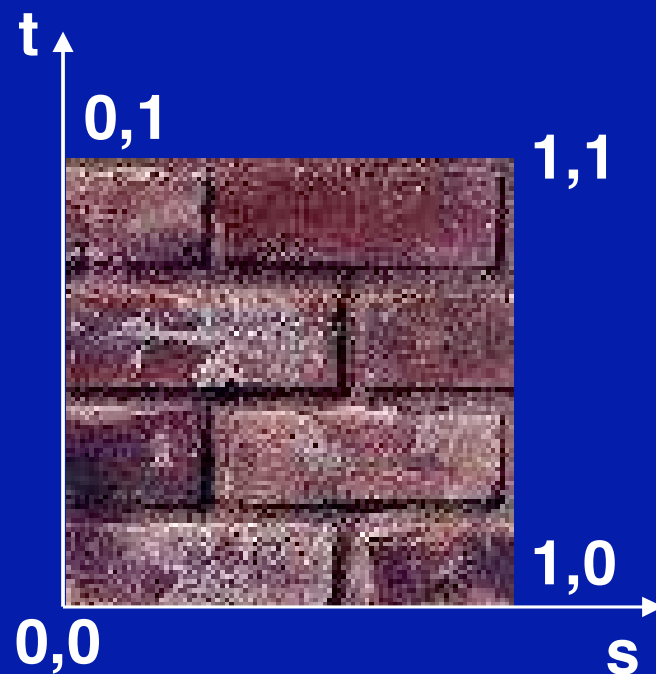
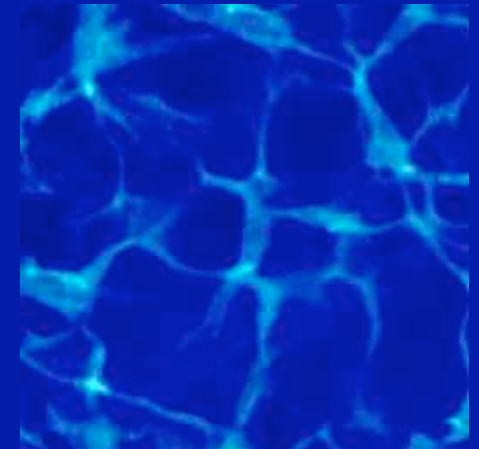
Texture Mapping

- A way of adding surface details
- Two ways can achieve the goal:
 - Model the surface with more polygons
 - » Slows down rendering speed
 - » Hard to model fine features
 - Map a texture to the surface
 - » This lecture
 - » Image complexity does not affect complexity of processing



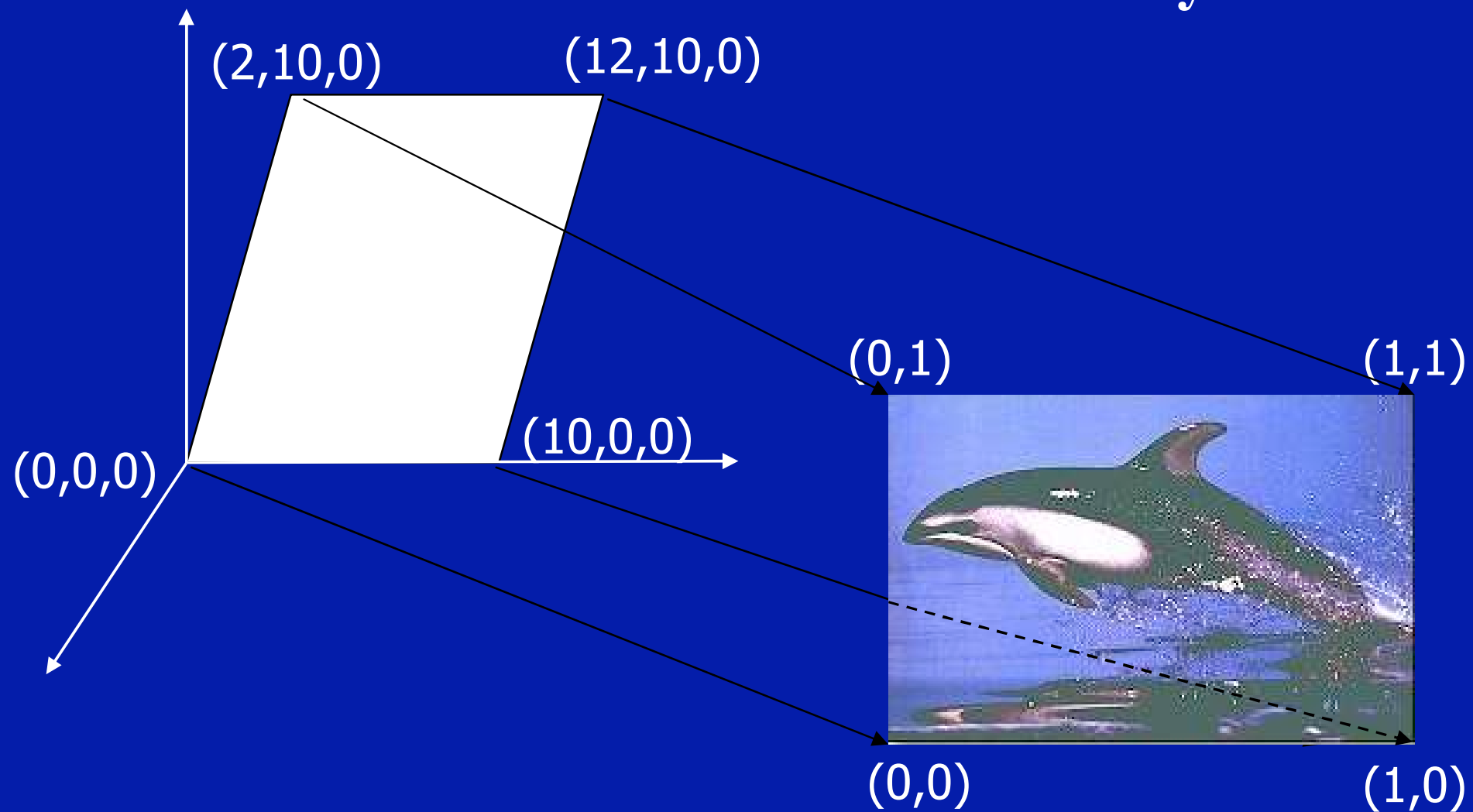
The texture

- Texture is a bitmap image
- 2D array: `texture[height][width][4]`
- Pixels of the texture called *texels*
- Texel coordinates (s,t) scaled to [0,1] range



Map textures to surfaces

The polygon can have arbitrary size and shape



The drawing itself

- Use `GLTexCoord2f(s,t)` to specify texture coordinates

- Example:

```
glEnable(GL_TEXTURE_2D)
glBegin(GL_QUADS);
glTexCoord2f(0.0,0.0); glVertex3f(0.0,0.0,0.0);
glTexCoord2f(0.0,1.0); glVertex3f(2.0,10.0,0.0);
glTexCoord2f(1.0,0.0); glVertex3f(10.0,0.0,0.0);
glTexCoord2f(1.0,1.0); glVertex3f(12.0,10.0,0.0);
glEnd();
glDisable(GL_TEXTURE_2D)
```

- State machine: Texture coordinates remain valid until you change them or exit texture mode via
`glDisable (GL_TEXTURE_2D)`

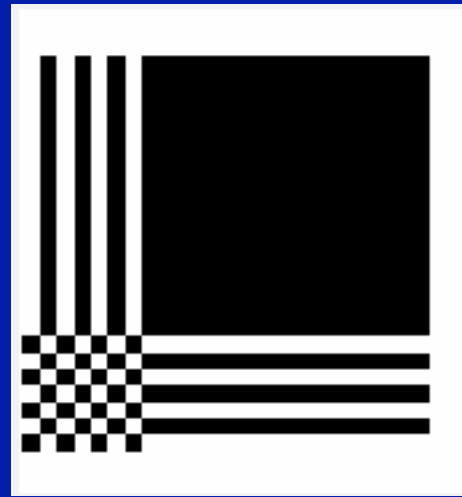
Color blending

- **Final pixel color = f (texture color, object color)**
- **How to determine the color of the final pixel?**
 - **GL_REPLACE** – use texture color to replace object color
 - **GL_BLEND** – linear combination of texture and object color
 - **GL_MODULATE** – multiply texture and object color
- **Example:**
 - `glTexEnvf(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_REPLACE);`

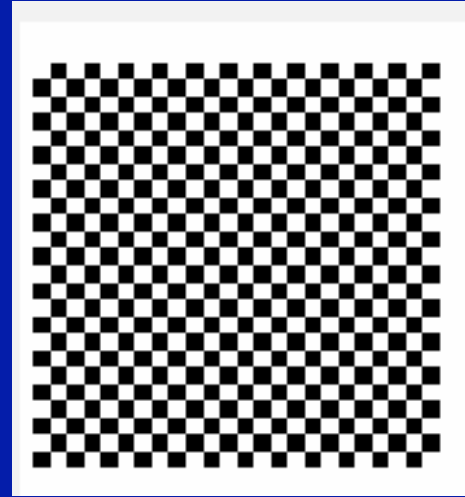
What happens if texture coordinates outside [0,1] ?

- Two choices:
 - Repeat pattern (GL_REPEAT)
 - Clamp to maximum/minimum value (GL_CLAMP)
- Example:
 - `glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_CLAMP)`
 - `glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_CLAMP)`

What happens if texture coordinates outside $[0,1]$?



clamp

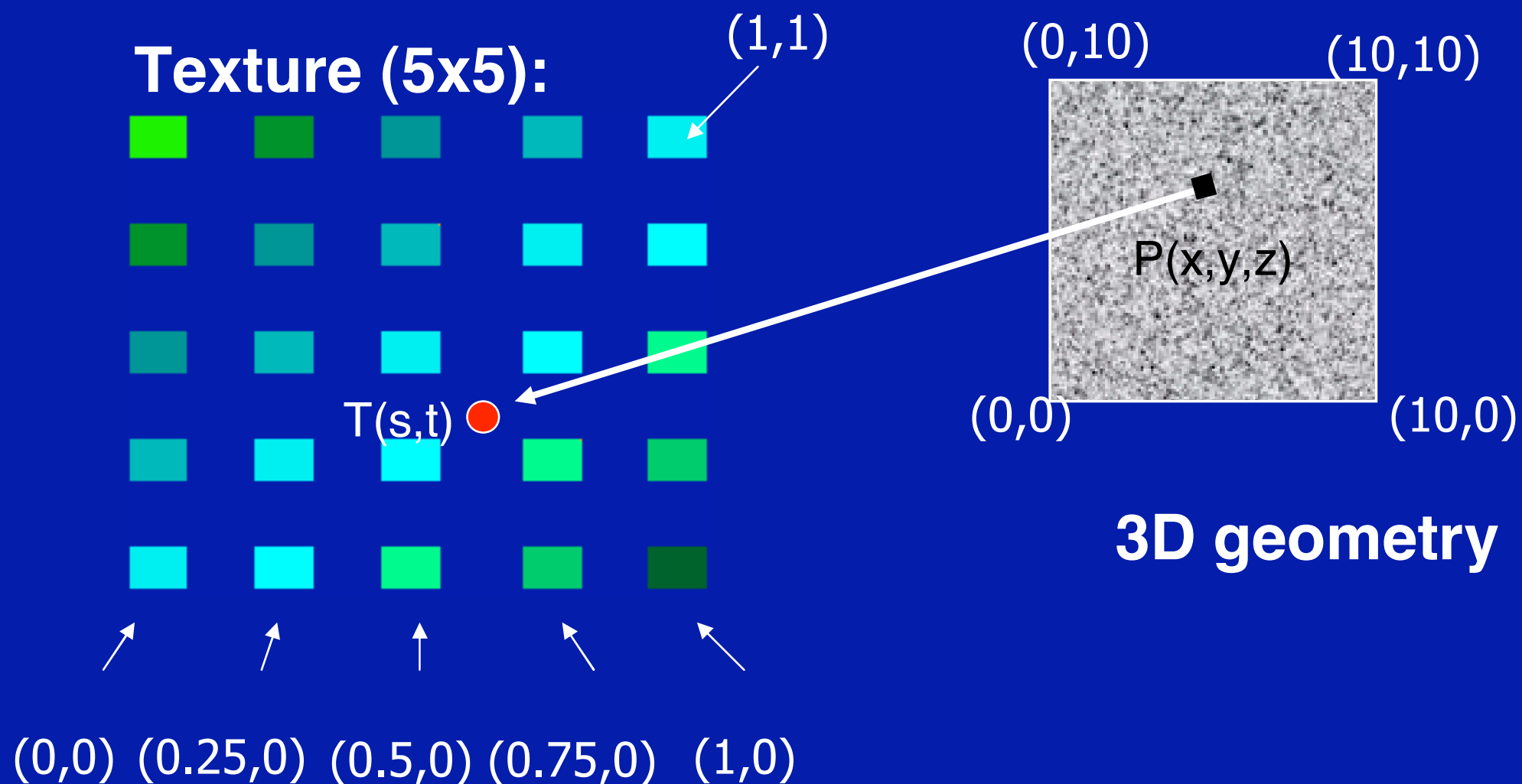


repeat

```
glTexCoord2f(0.0, 0.0); glVertex3f(0.0, 0.0, 0.0);  
glTexCoord2f(0.0, 3.0); glVertex3f(0.0, 10.0, 0.0);  
glTexCoord2f(3.0, 0.0); glVertex3f(10.0, 0.0, 0.0);  
glTexCoord2f(3.0, 3.0); glVertex3f(10.0, 10.0, 0.0);
```

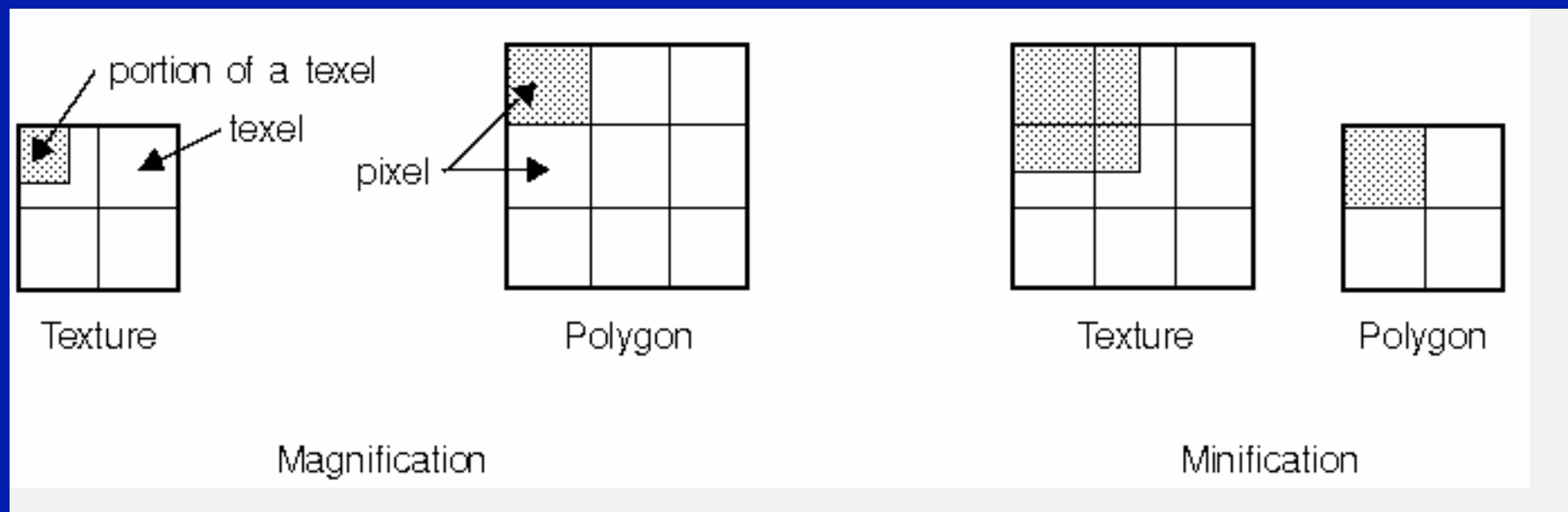
Texture Value Lookup

- For given texture coordinates (s,t) , we can find a unique image value, corresponding to the texture image at that location



Interpolating colors

- Some (s,t) coordinates not directly at pixel in the texture, but in between

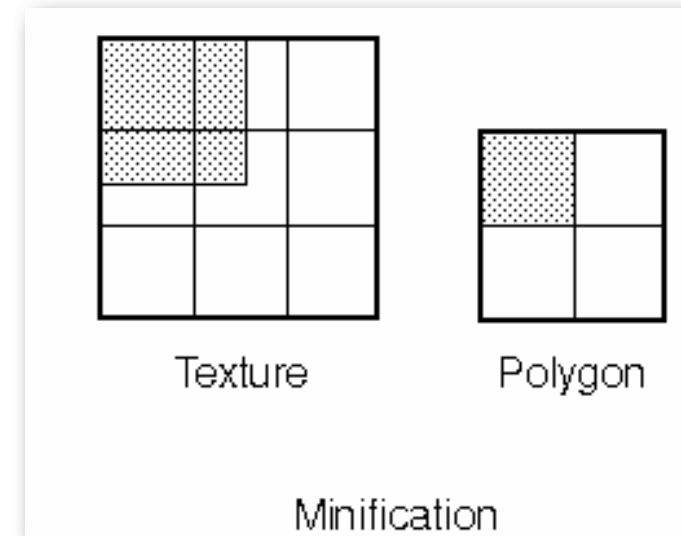


Interpolating colors

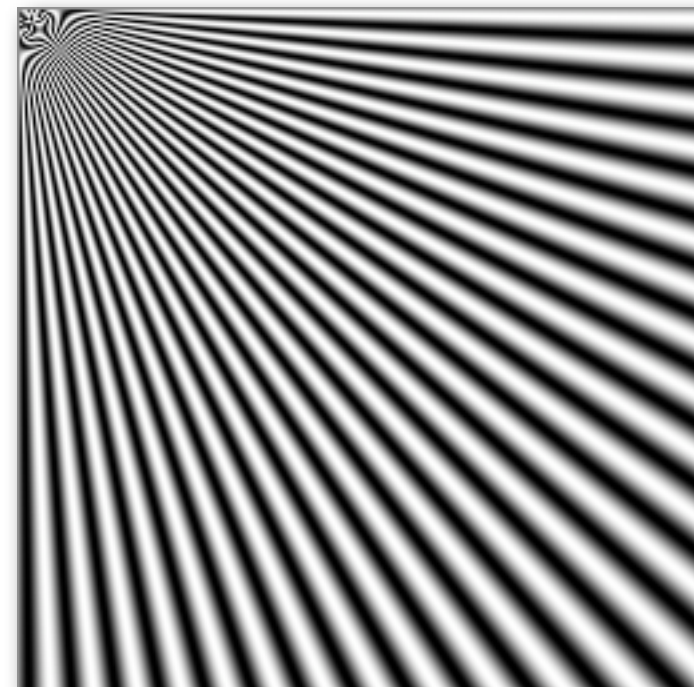
- **Solutions:**
 - **Nearest neighbor**
 - » Use the nearest neighbor to determine color
 - » Faster, but worse quality
 - » `glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST);`
 - **Linear interpolation**
 - » Incorporate colors of several neighbors to determine color
 - » Slower, better quality
 - » `glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR)`

Other solutions

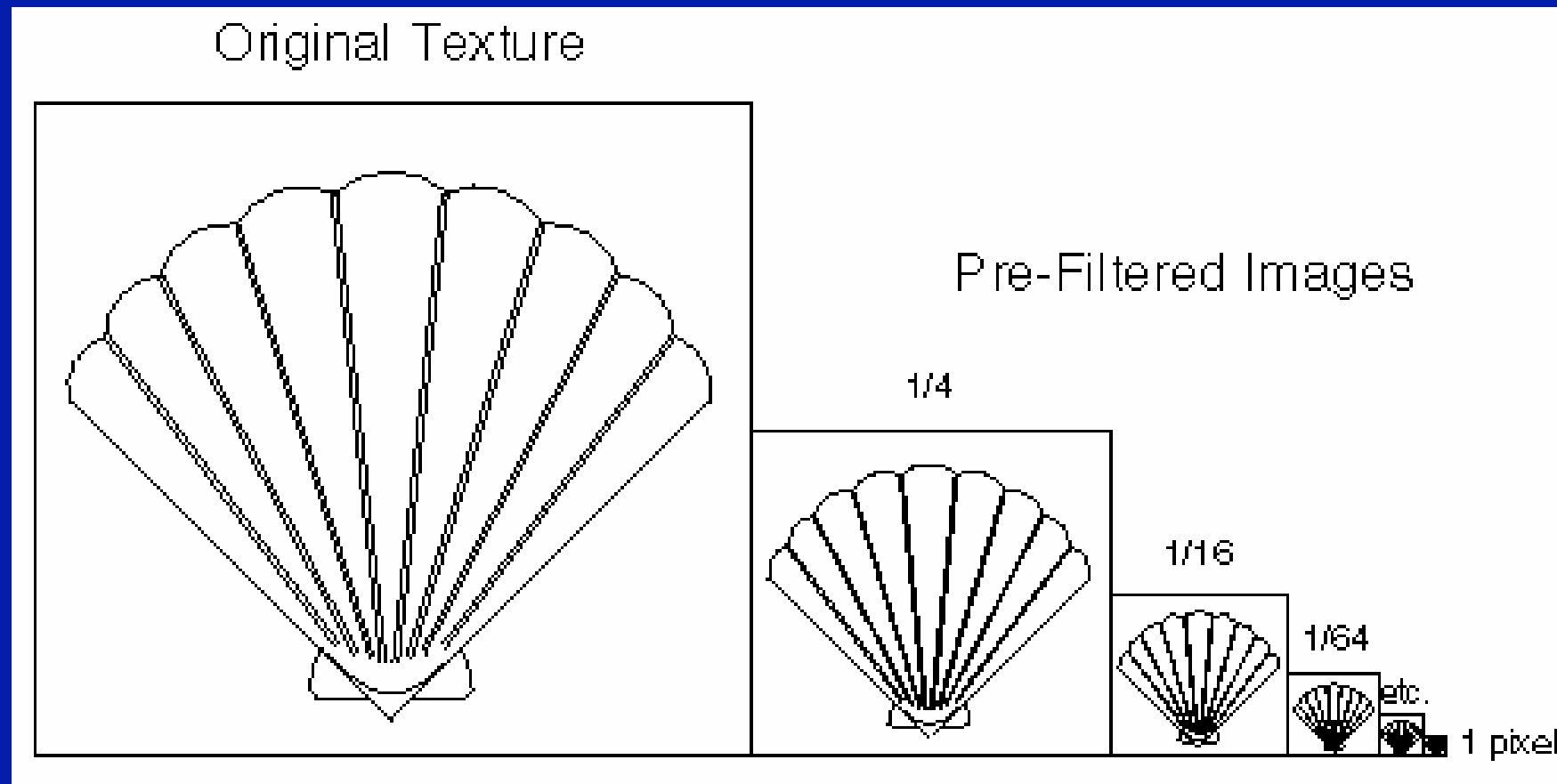
- Signal processing.
- What is wrong with linear interpolation...

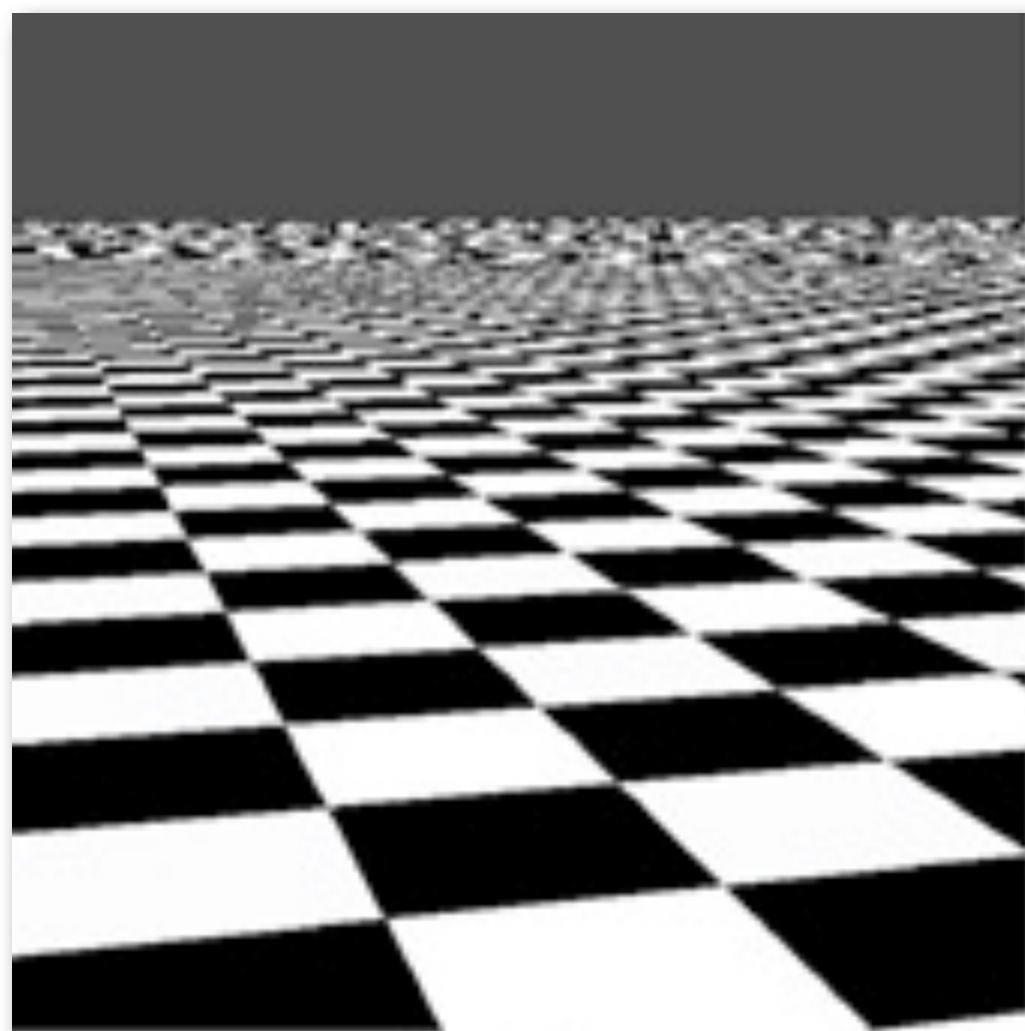


Antialiasing...

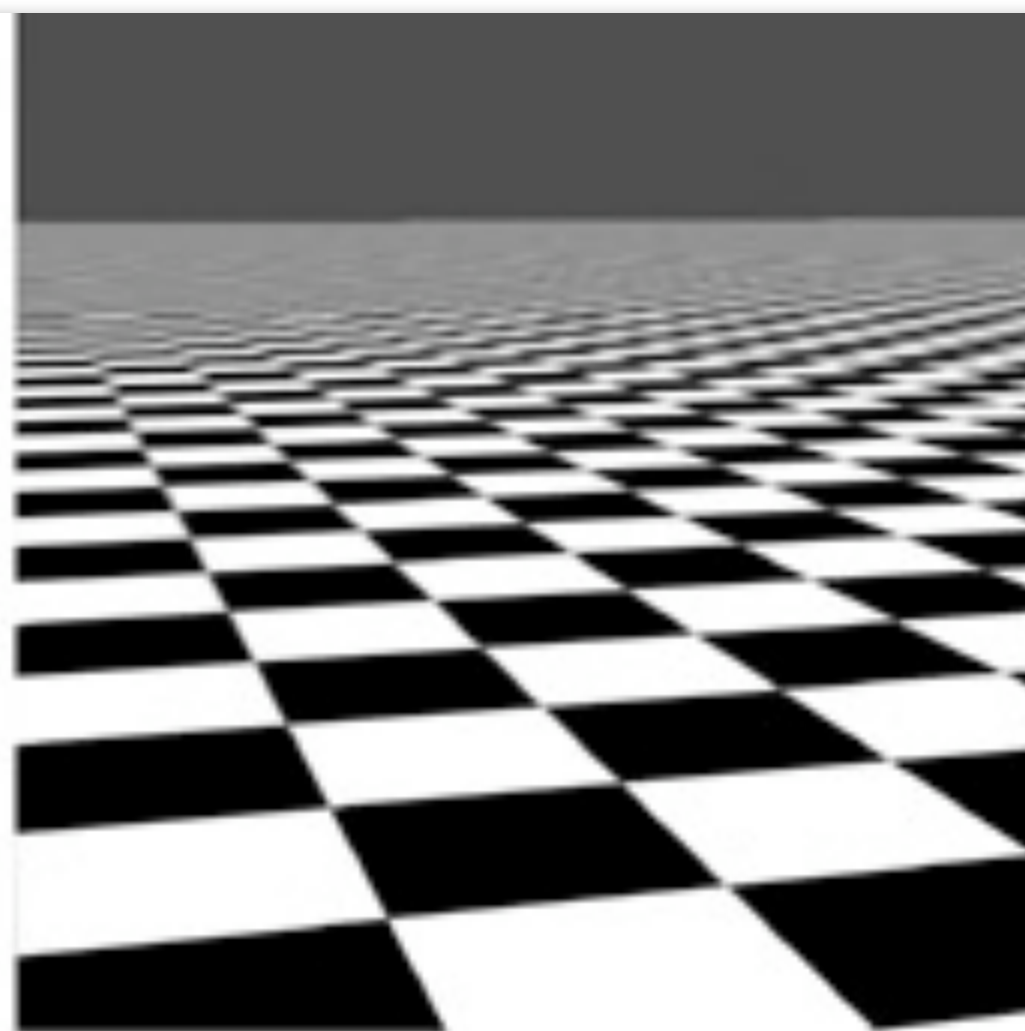


Texture Levels





(a)



(b)

Texture mapping in OpenGL

- **In init():**
 - Specify texture
 - » Read image from file into an array in memory or generate the image using the program
 - Specify texture mapping parameters
 - » Wrapping, filtering, etc.
 - Define (activate) the texture
- **In display():**
 - Enable GL texture mapping
 - Draw objects: Assign texture coordinates to vertices
 - Disable GL texture mapping

Specifying texture mapping parameters

- Use `glTexParameter`
- Example:

// texture wrapping on

```
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S,  
GL_REPEAT); // repeat pattern in s texture coordinate
```

```
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T,  
GL_REPEAT); // repeat pattern in t texture coordinate
```

// use nearest neighbor for both minification and magnification

```
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER,  
GL_NEAREST);
```

```
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER,  
GL_NEAREST);
```

Defining (activating) texture

- Do once in `init()` to set up initial pattern
- To use another texture, make further calls in `display()` to `glTexImage2D`, specifying another image
 - But this is slow: use Texture Objects itself
- The dimensions of texture images **must be powers of 2**
 - if not, rescale image or pad with zeros
- `glTexImage2D(Glenum target, Glint level, Glint internalFormat, int width, int height, Glint border, Glenum format, Glenum type, Glvoid* img)`
- Example:
 - `glTexImage2D(GL_TEXTURE_2D, 0, GL_RGBA, 256, 256, 0, GL_RGBA, GL_UNSIGNED_BYTE, pointerToImage)`

Enable/disable texture mode

- Can do in `init()` or successively in `display()`
- `glEnable(GL_TEXTURE_2D)`
- `glDisable(GL_TEXTURE_2D)`
- Successively enable/disable texture mode to switch between drawing textured/non-textured polygons
- Changing textures:
 - Only one texture active at any given time
 - make another call to `glTexImage2D` to make another pattern active

The drawing itself

- Use `GLTexCoord2f(s,t)` to specify texture coordinates
- State machine: Texture coordinates remain valid until you change them or exit texture mode via `glDisable (GL_TEXTURE_2D)`

- Example:

```
glEnable(GL_TEXTURE_2D)
glBegin(GL_QUADS);
glTexCoord2f(0.0,0.0); glVertex3f(-2.0,-1.0,0.0);
glTexCoord2f(0.0,1.0); glVertex3f(-2.0,1.0,0.0);
glTexCoord2f(1.0,0.0); glVertex3f(0.0,1.0,0.0);
glTexCoord2f(1.0,1.0); glVertex3f(0.0,-1.0,0.0);
...
glEnd();
glDisable(GL_TEXTURE_2D)
```

Everything together

```
void init(void):
{
...
put image into 2D memory array; // can use libpicio library

// specify texture parameters
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT); // repeat pattern in s
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT); // repeat pattern in t

// use nearest neighbor for both minification and magnification
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_NEAREST);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST);

// make the pattern at location pointerToImage the active pattern
glTexImage2D(GL_TEXTURE_2D, 0, GL_RGBA, 256, 256, 0,
             GL_RGBA, GL_UNSIGNED_BYTE, pointerToImage)

...
}
```


Everything together (contd.)

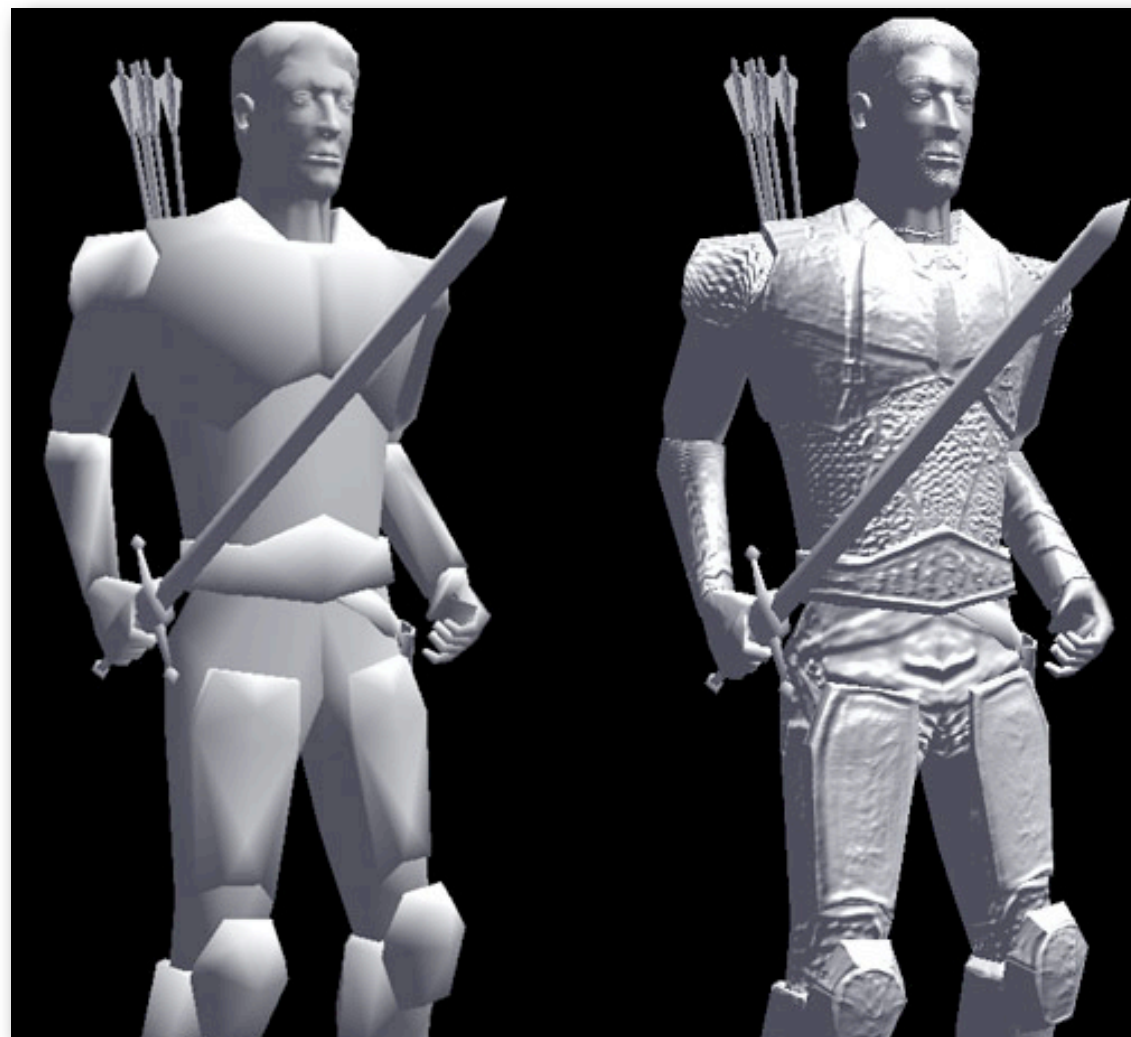
```
void display(void):
{
    ...
    // no blending, use texture color directly
    glTexEnvf(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_REPLACE);
    // turn on texture mode
    glEnable(GL_TEXTURE_2D);

    glBegin(GL_QUADS); // draw a quad
    glTexCoord2f(0.0,0.0); glVertex3f(-2.0,-1.0,0.0);
    glTexCoord2f(0.0,1.0); glVertex3f(-2.0,1.0,0.0);
    glTexCoord2f(1.0,0.0); glVertex3f(0.0,1.0,0.0);
    glTexCoord2f(1.0,1.0); glVertex3f(0.0,-1.0,0.0);
    ...
    glEnd();

    // turn off texture mode
    glDisable(GL_TEXTURE_2D);

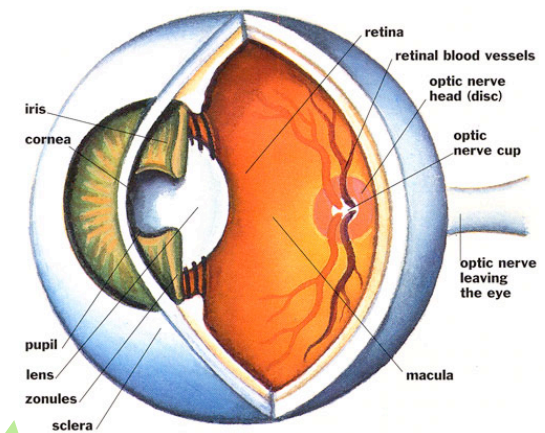
    // draw some non-texture mapped objects
    ...
    // switch back to texture mode, etc.
    ...
}
```

Generalizations of Texture Mapping?



Shading

Part I: What is Light?



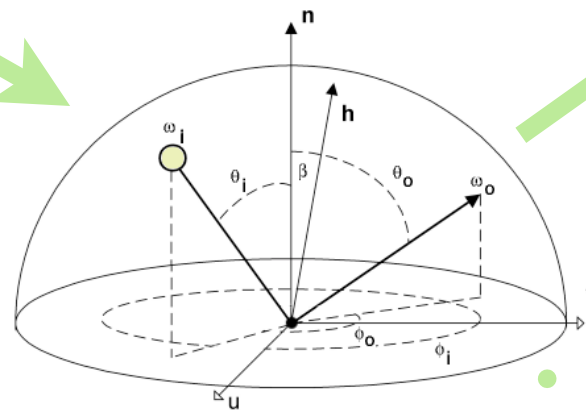
Part II: The Eye

Part V: Shadows

No
Light

Part III: Reflectance

Part IV: Lighting



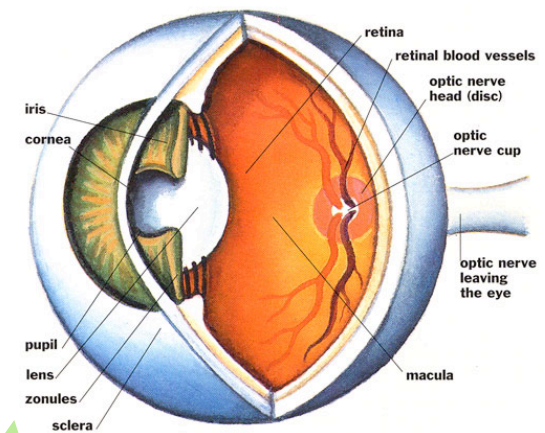
Light

Light

Light

Shading

Part I: What is Light?



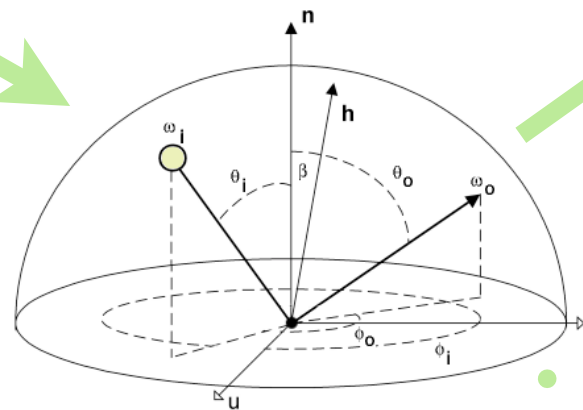
Part II: The Eye

Part V: Shadows

No
Light

Part III: Reflectance

Part IV: Lighting



Light

Light

Light

What are the patterns of light in this room?

Projector as light source

Light transmitted through windows

Blue light reflecting from screen

Blackboard is matte surface

Edge of screen is shiny surface

Shadows underneath the desks

Physics of Light and Color

Electromagnetic (EM) radiation

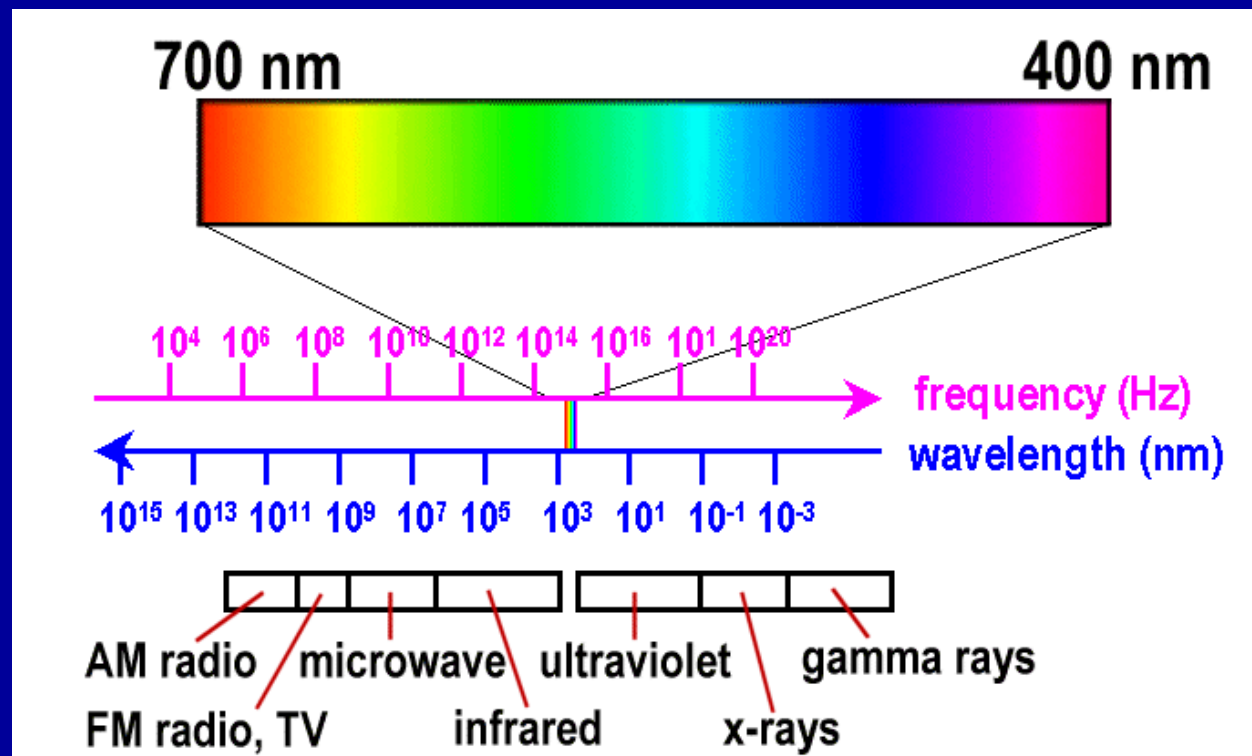
Different colors correspond to radiation of different wavelengths λ

Intensity of each wavelength specified by amplitude

Frequency $\nu = 2 \pi / \lambda$

long wavelength is low frequency

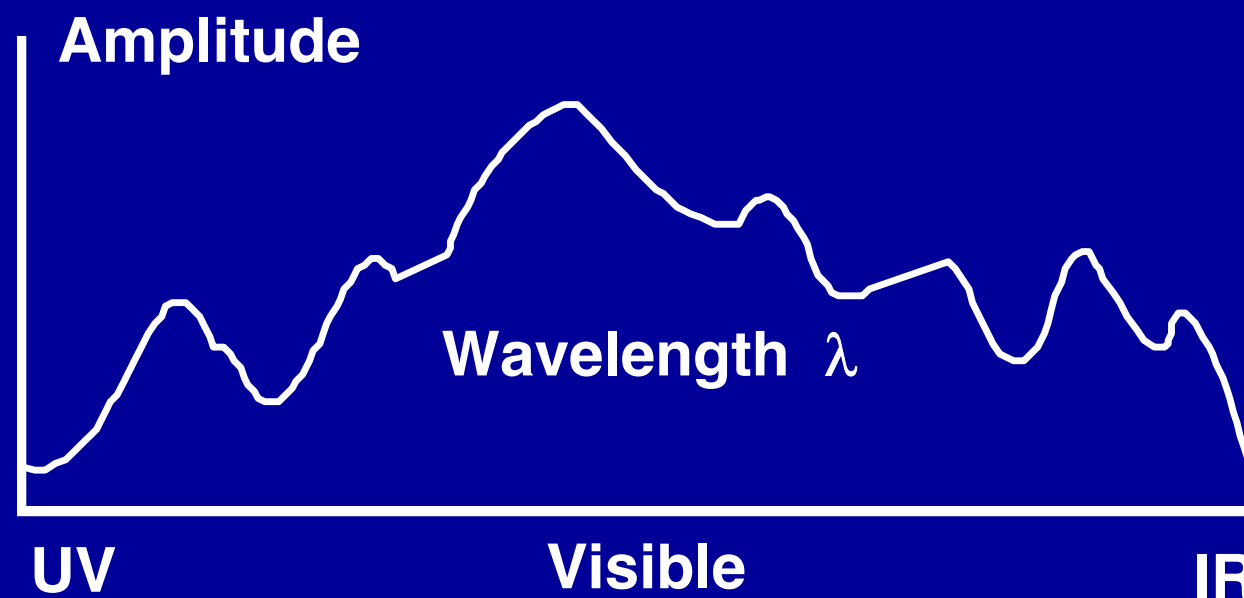
short wavelength is high frequency



We perceive EM radiation with λ in the 400-700 nm range

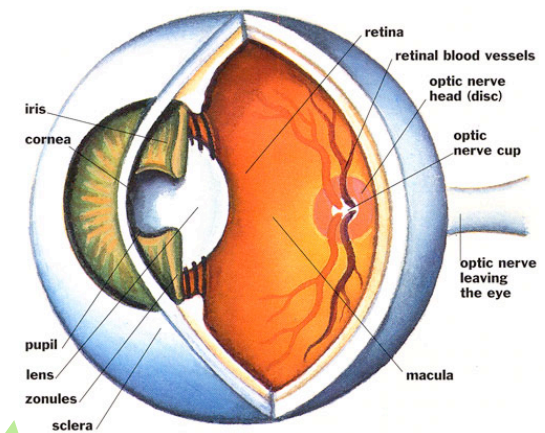
Color: What's There vs. What We See

- Human eyes respond to “visible light”
 - tiny piece of spectrum between infra-red and ultraviolet
- Color defined by the emission spectrum of the light source
 - amplitude vs wavelength (or frequency) plot



Shading

Part I: What is Light?



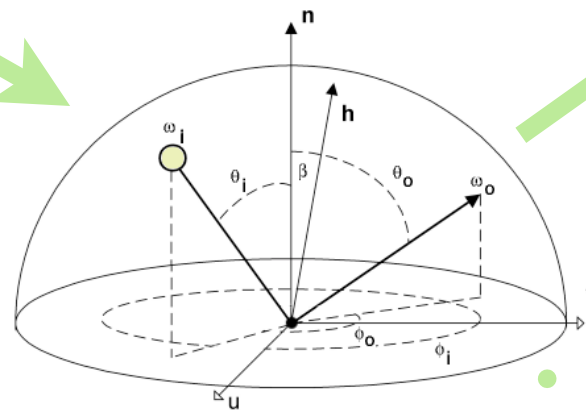
Part II: The Eye

Part V: Shadows

No
Light

Part III: Reflectance

Part IV: Lighting



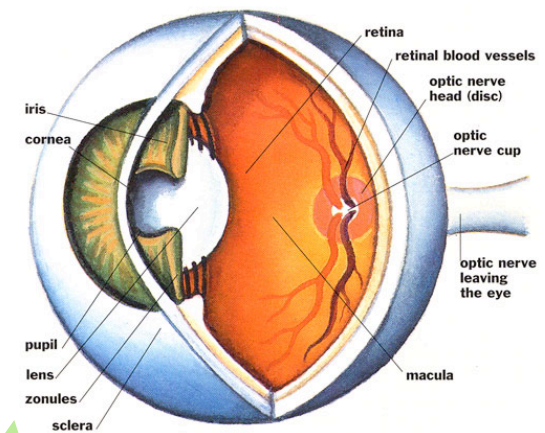
Light

Light

Light

Shading

Part I: What is Light?



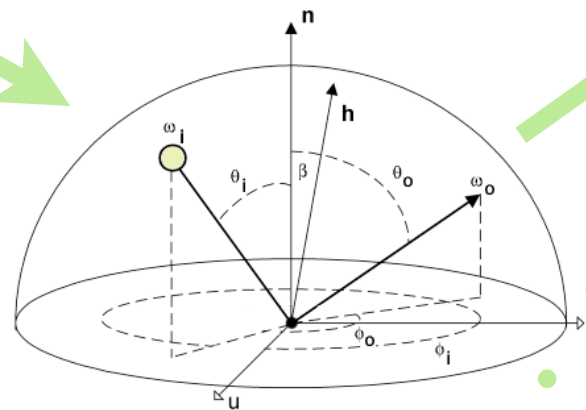
Part II: The Eye

Part V: Shadows

No
Light

Part III: Reflectance

Part IV: Lighting

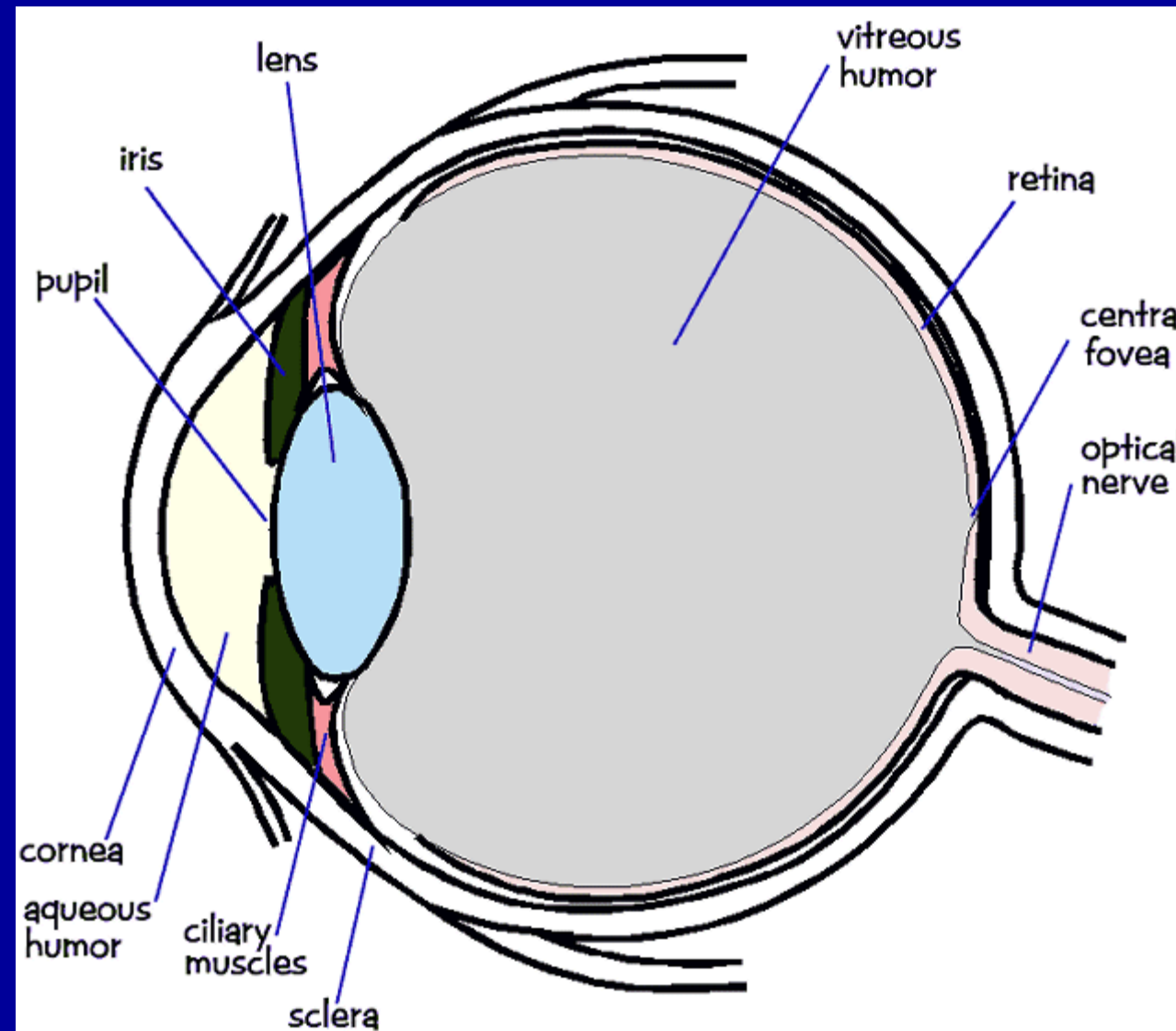


Light

Light

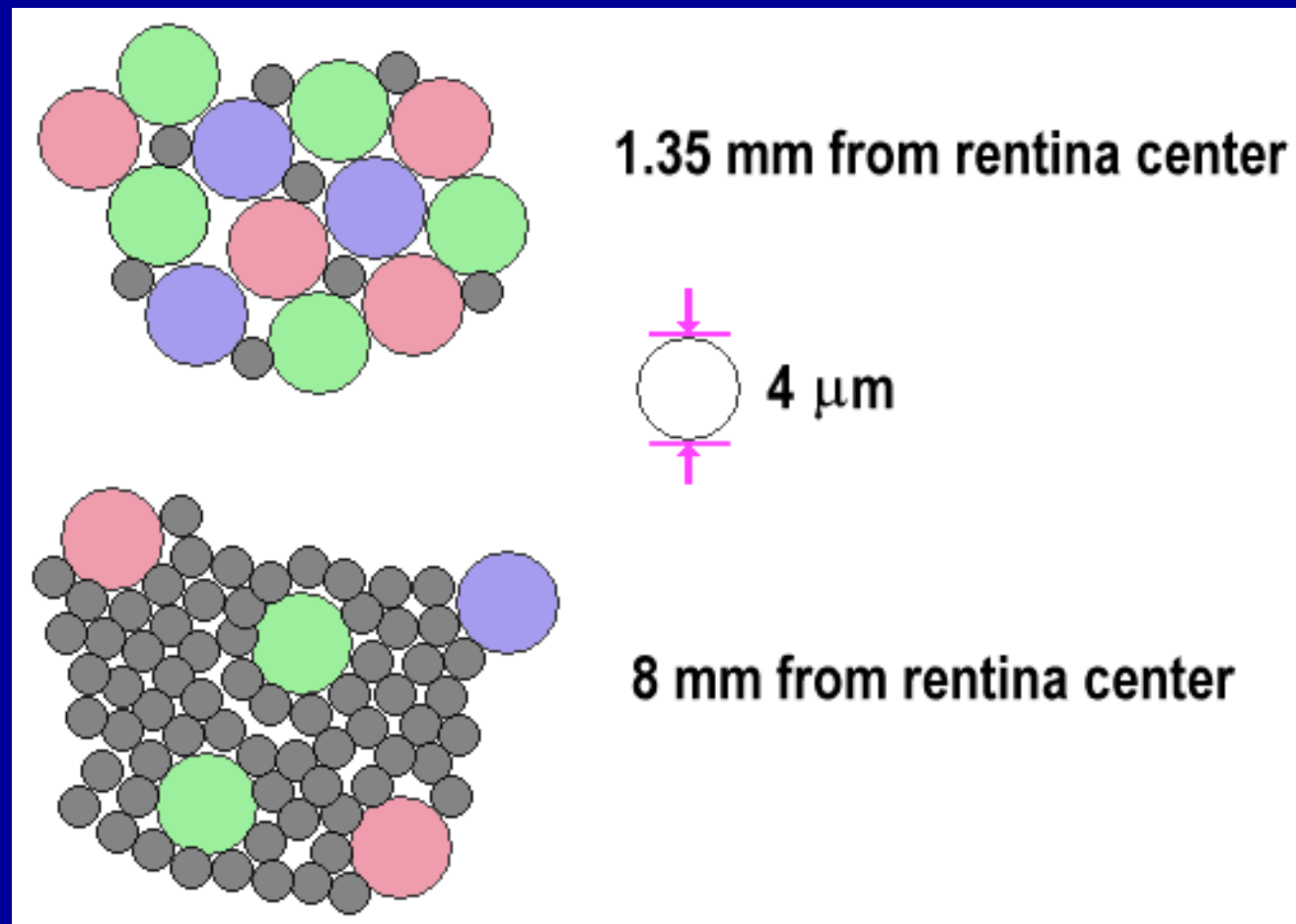
Light

The Eye

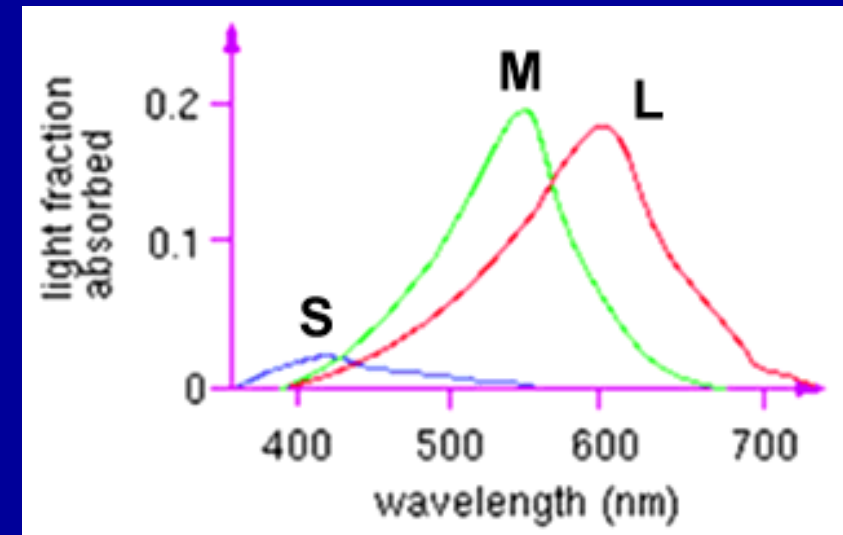


- The image is formed on the *retina*
- Retina contains two types of cells: *rods* and *cones*
- Cones measure color (red, green, blue)
- Rods responsible for monochrome night-vision

The Fovea



Cones are most densely packed within a region of the retina called the *fovea*



Three types of cones: S,M,L
Corresponds to 3 visual pigments

Roughly speaking:

S responds to blue

M responds to green

L responds to red

Not uniform sensitivity

Colorblindness

deficiency of one cone/pigment type

Color Filters

Rods and cones can be thought of as filters

Cones detect red, green or blue parts of spectrum

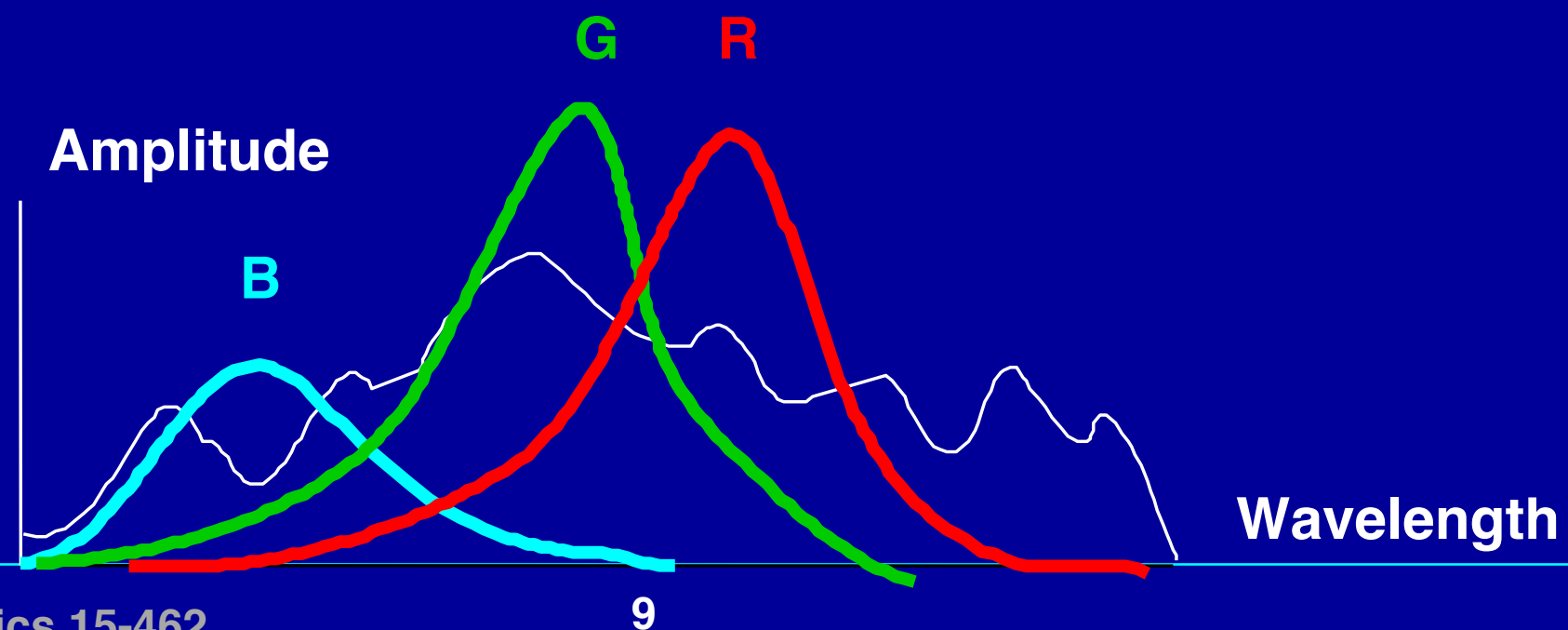
Rods detect average intensity across spectrum

A physical spectrum is a complex function of wavelength

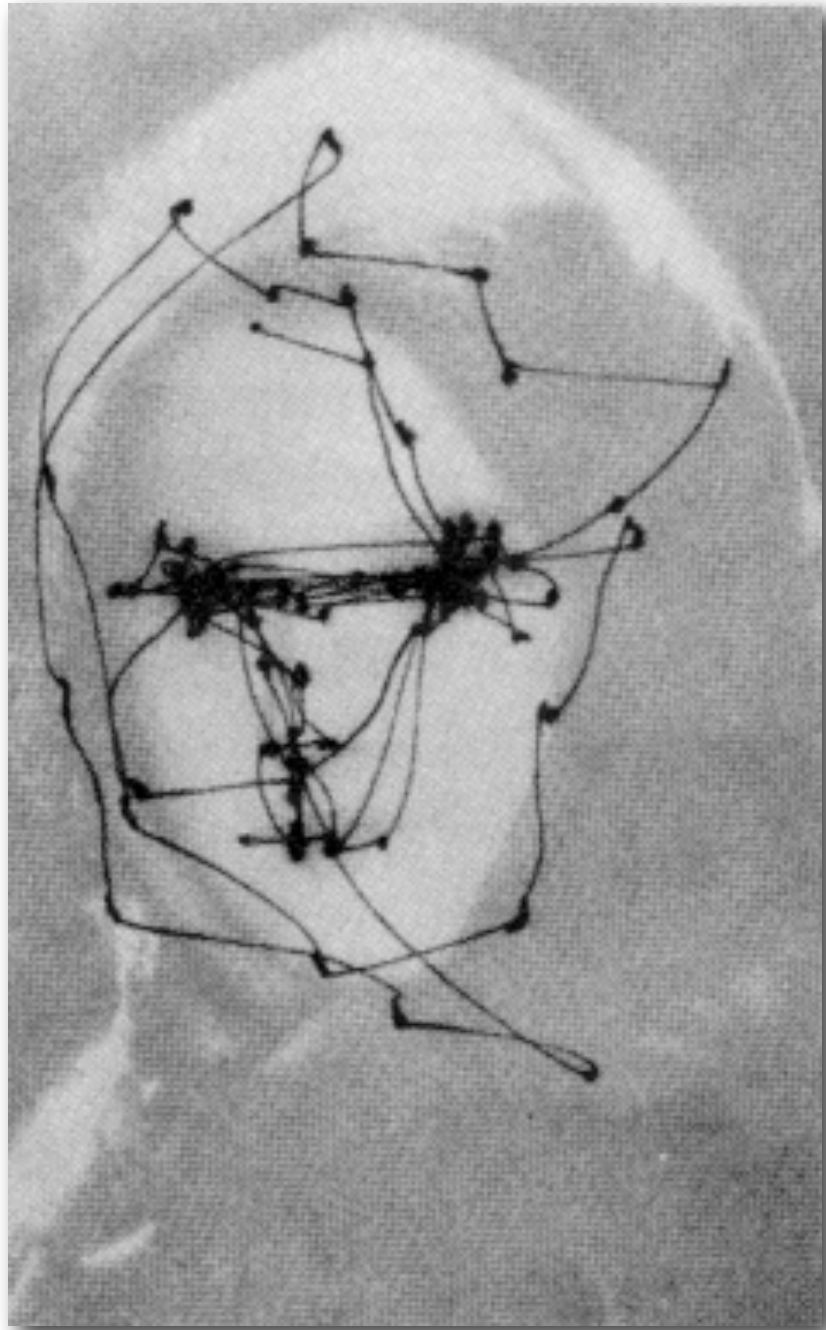
But what we see can be described by just three numbers—the color filter outputs

How can we encode a whole function with just three numbers?

We can't—we can't distinguish certain colors--*metamers*



Saccades



Vision and the brain

The retina is part of the central nervous system

2 million fibers from retina to lateral geniculate nucleus (*LGN*),
10 million from there to brain.

Primary connection is *Primary Visual Cortex* or *V1*

2 cm² on back of brain

Hypothesis: V1 gets used as a sort of image buffer for higher processing in the rest of the brain

Steps:

Saccade ends

Retina accumulates image

LGN opens connections, image gets written to V1

Rest of brain accesses that info

Meanwhile, a point of interest is being generated for next saccade

Next saccade happens perhaps 250ms later; go back to step 1

All automatic; eye tracking systems can discern attention but
pointing with eyes doesn't work very well for user interfaces.

Color Models

Okay, so our visual system is quite limited, but maybe this is good news. . .

We can avoid computing and reproducing the full color spectrum since people only have three color channels
everything would be much more complex if we perceived the full spectrum

- transmission would require much higher bandwidths

- display would require much more complex methods

- real-time color 3D graphics is feasible

- any scheme for describing color requires only three values

- lots of different color spaces--related by matrix transformations

Color Spaces

Spectrum

allows any radiation (visible or invisible) to be described
usually unnecessary and impractical

RGB

convenient for display (CRT uses red, green, and blue phosphors)
not very intuitive

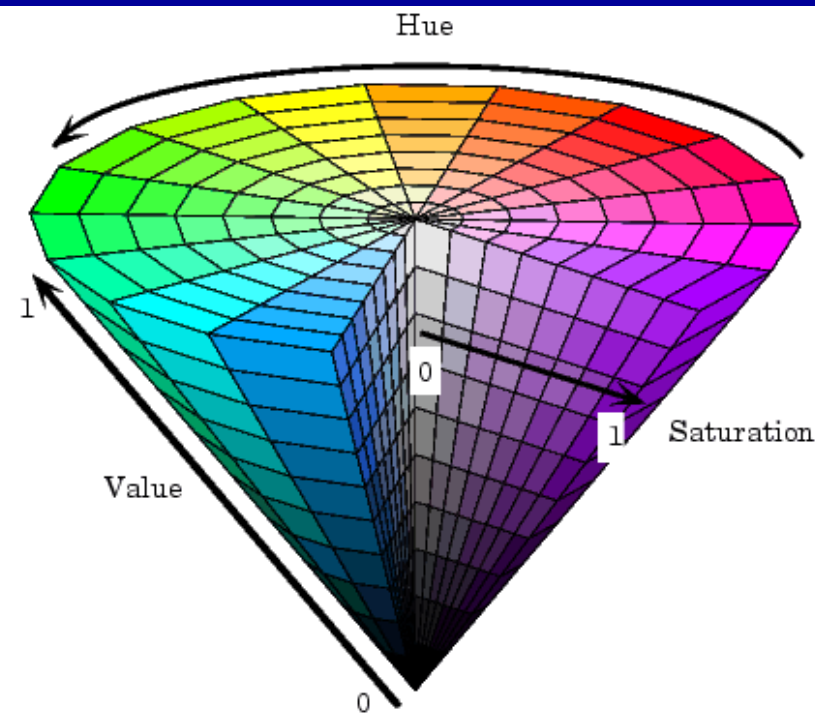
HSV

an intuitive color space
H is hue - what color is it? S is saturation or purity - how non-gray is it? V is value - how bright is it?
H is cyclic therefore it is a non-linear transformation of RGB

CIE XYZ

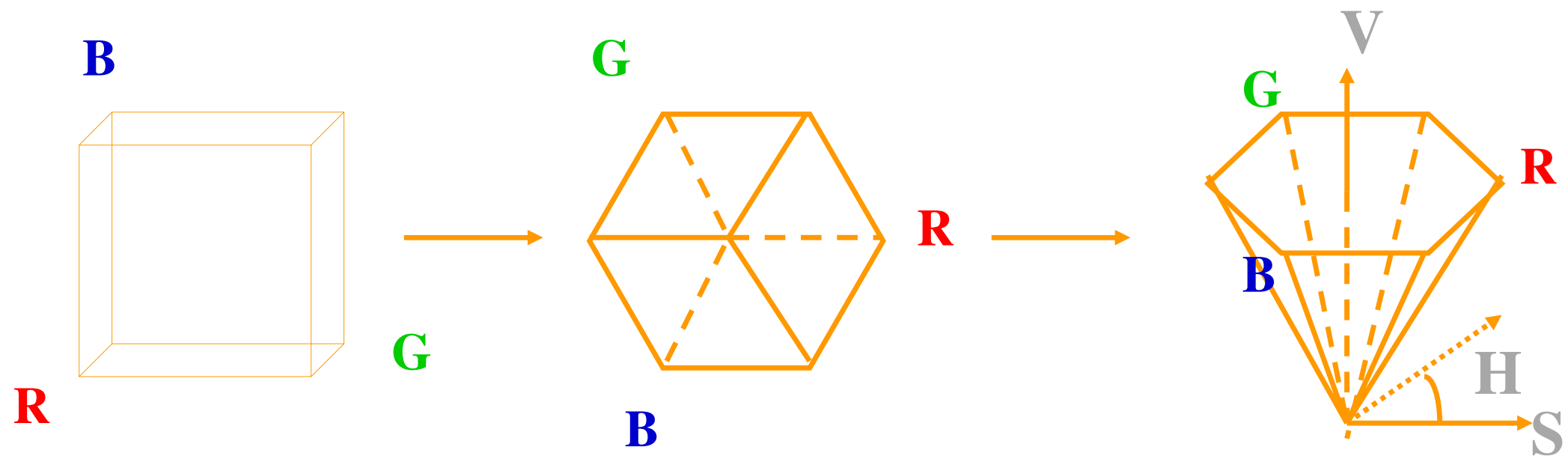
a linear transform of RGB used by color scientists

HSV



Hue: color
Saturation: how non-grey
Value: brightness

From mathworks



Better Color Models?


























| Scanned Paint | 101 Samples Riemann Sum | 8 Samples IMPasto | 3 Samples RGB w/ K-M | 3 Samples RGB Linear |
|---|---|---|--|---|
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

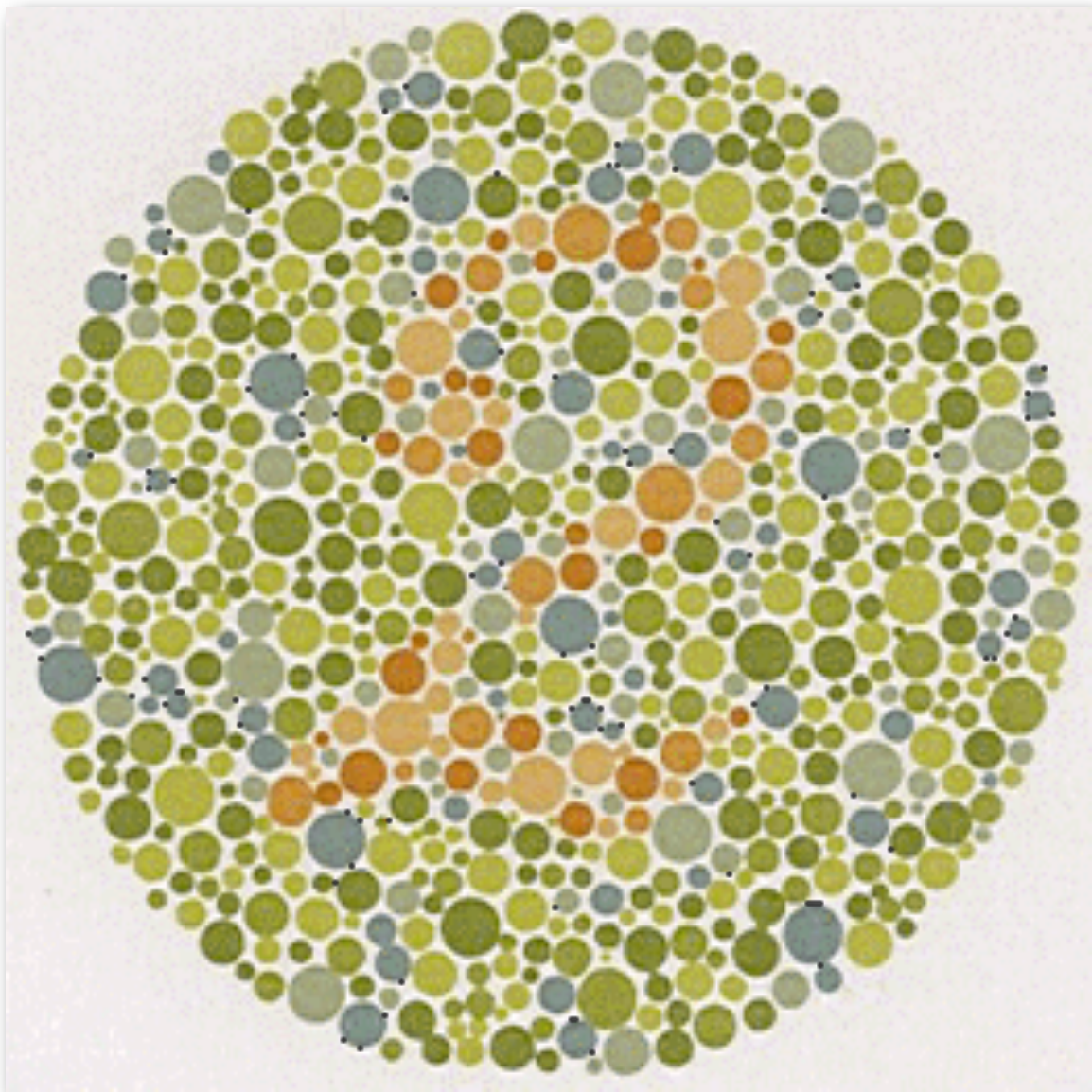


Figure 11: A painting created with IMPasto, after a painting by

source:

IMPasto: a realistic, interactive model for paint
William Baxter, Jeremy Wendt, Ming C. Lin
NPAR 2004, June 2004, pp. 45-56.

Tetrachromacy



Additive vs. Subtractive Color

- Working with light: additive primaries
 - Red, green and blue components are added by the superposition property of electromagnetism
 - Conceptually: start with black, primaries add light
- Working with pigments: subtractive primaries
 - Typical inks (CMYK): cyan, magenta, yellow, black
 - Conceptually: start with white, pigments filter out light
 - The pigments remove parts of the spectrum

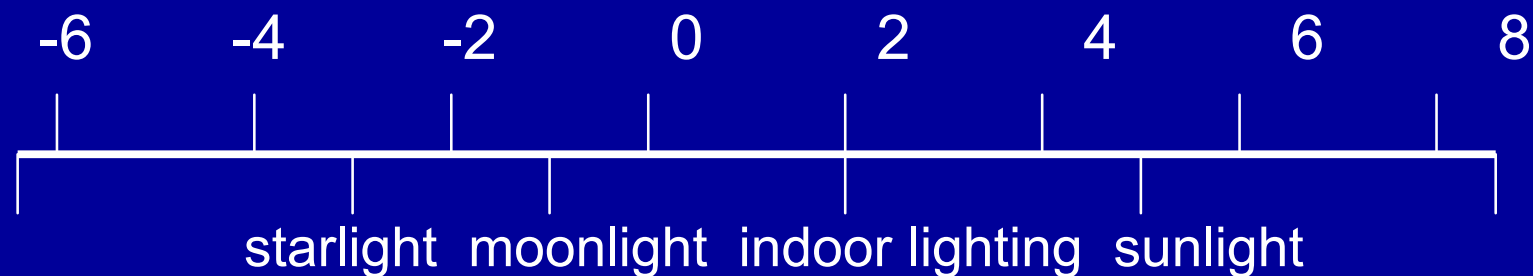
| dye color | absorbs | reflects |
|-----------|---------|----------------|
| cyan | red | blue and green |
| magenta | green | blue and red |
| yellow | blue | red and green |
| black | all | none |

- Inks interact in nonlinear ways--makes converting from monitor color to printer color a challenging problem
- Black ink (K) used to ensure a high quality black can be printed

What about displays?

Humans can't see most of the spectrum but displays can't display most of what we can see

Human Overall Luminance Vision Range
(14 orders of magnitude, scale in log cd/m²)



Human Simultaneous
Luminance Vision Range

5 orders
of magnitude

Today's Devices

2-3
orders

BrightSide Technologies

5 orders
of magnitude

What about displays?

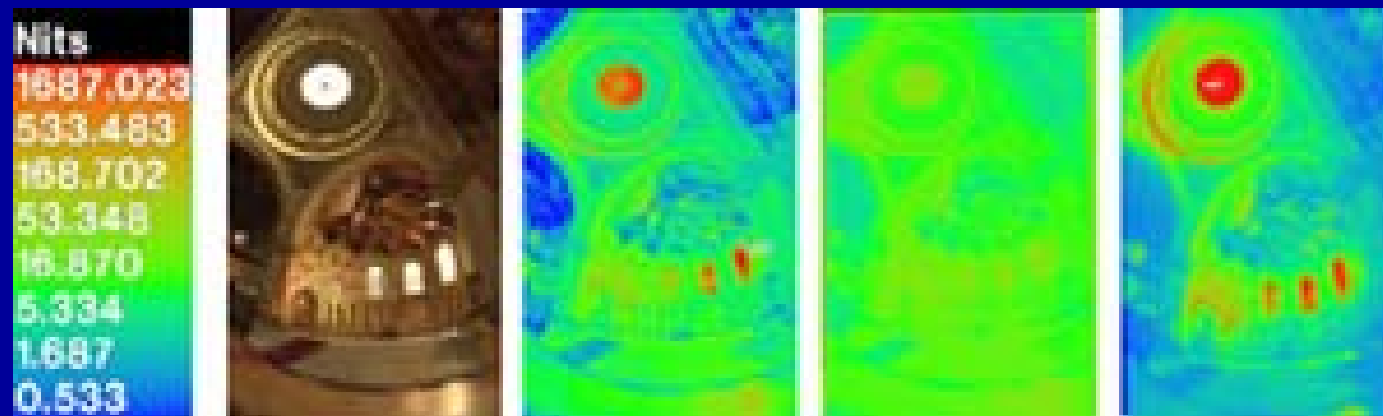
Conventional CRTs have 600:1 dynamic range

Flat-panel LCDs are 500:1.

BrightSide's HDR displays achieve 200,000:1

10 times higher brightness than any commercially available display while at the same time delivering a black that is over 10 times darker than that of conventional displays.

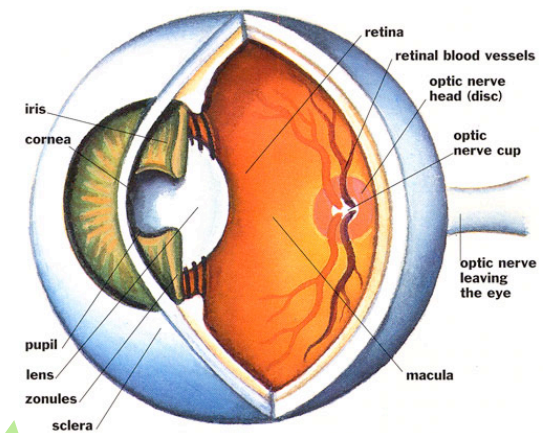
<http://www.brightsidetech.com>



HDR image, range, conventional display, HDR display

Shading

Part I: What is Light?



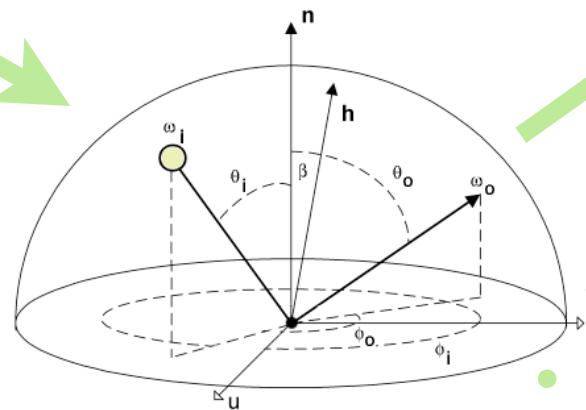
Part II: The Eye

Part V: Shadows

No
Light

Part III: Reflectance

Part IV: Lighting



Light

Light

Light