Computer Graphics - Week 12

Bengt-Olaf Schneider
IBM T.J. Watson Research Center

Questions about Last Week ?

Overview of Week 12

- **Graphics Hardware**
  - Output devices (CRT and LCD)
  - Graphics architectures
  - Performance modeling
- **Color**
  - Color theory
  - Color gamuts
  - Gamut matching
  - Color spaces (next week)

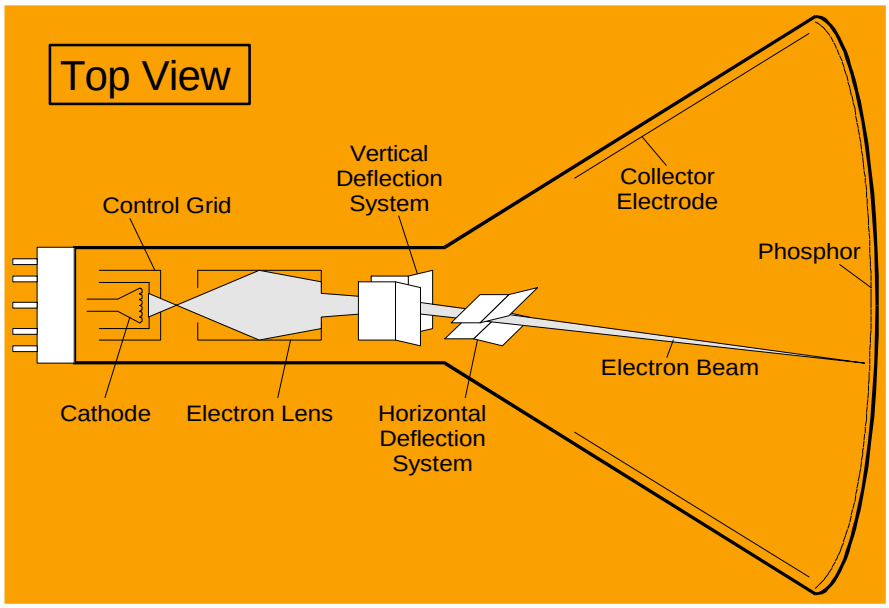
Graphics Hardware: Overview

- **Display technologies**
- **Graphics architecture fundamentals**
- **Many of the general techniques discussed earlier in the semester were developed with hardware in mind**
  - Close similarity between software and hardware architectures

Display Technologies

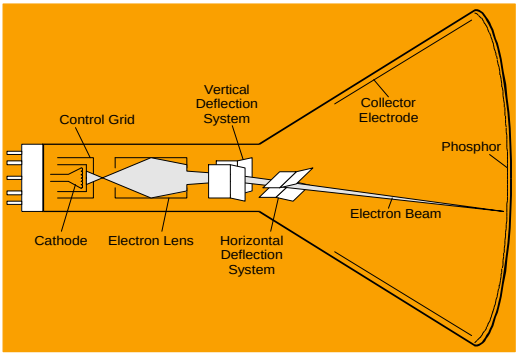
- CRT (Cathode Ray Tube)
- LCD (Liquid Crystal Display)

CRT: Basic Structure (1)



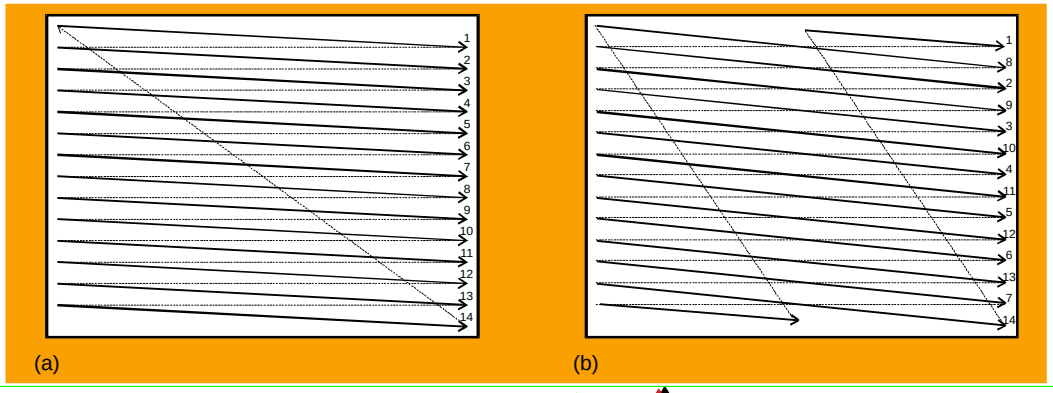
CRT: Basic Structure (2)

- **Deflection System**
  - Typically magnetic and not electrostatic
  - Reduces the length of the tube and allows wider deflection angles
- **Phospor**
  - Fluorescence: Light emitted after impact of electrons
  - Persistence: Duration of emission after beam is turned off. Determines flicker properties of the CRT.



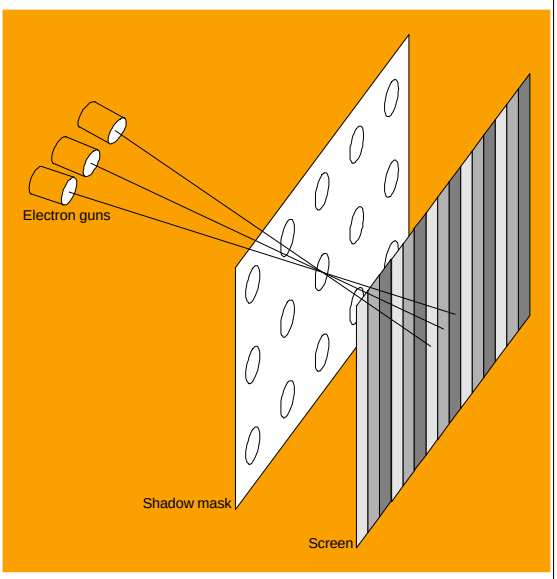
CRT: Scan Pattern

- **Non-interlaced**
  - A.k.a. progressive
  - Used in computer displays
- **Interlaced**
  - Used in standard TVs
  - Works because of irregular nature of TV images
  - Horizontal patterns flicker



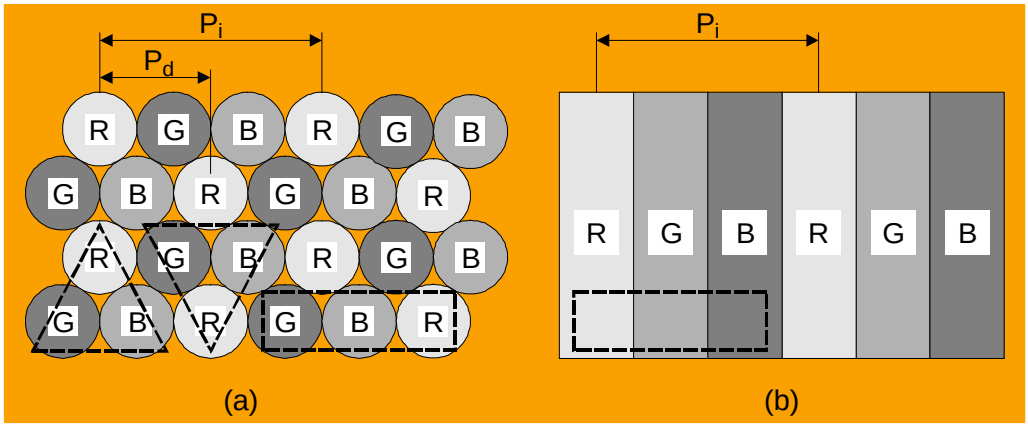
CRT: Shadow Mask

- **For color CTRs, beam must focus exactly on the phosphor to avoid cross-talk between pixels**
  - Shadow mask blocks off parts of the beam outside the pixel
  - Creates sharper image
  - Allows for wider deflection angles (and shorter tubes).
  - Requires careful registration of mask with phosphor



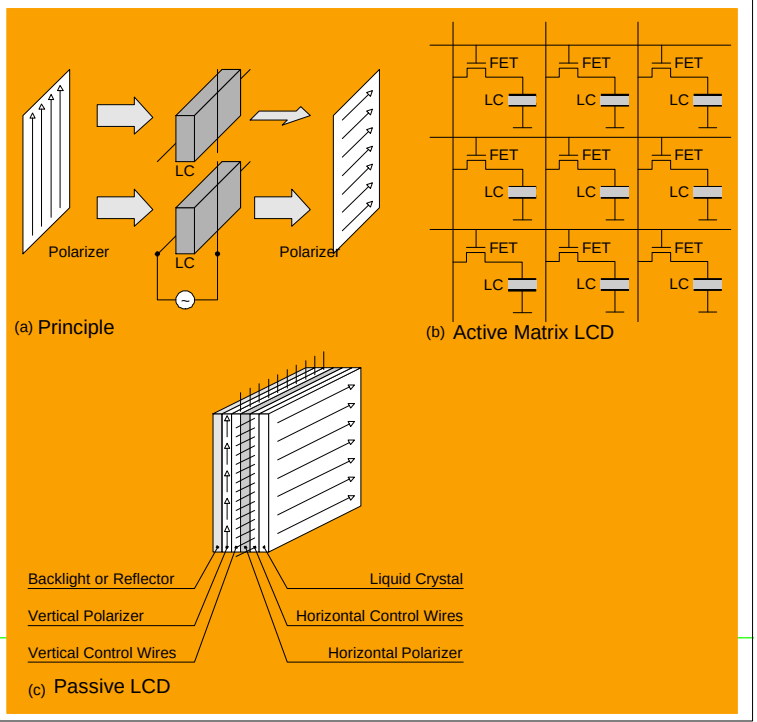
CRT: Phosphor Arrangments

- **Delta arrangement**
  - High-quality monitors
  - P_d approx. 0.2 mm
- **Inline arrangement**
  - TV monitors
  - P_i approx. 0.6 mm



LCD: Basic Structure

- **Types**
  - Backlit
  - Reflective
  - Transflective
- **Passive LCD**
  - Needs refresh
  - Low contrast
- **Active LCD**
  - No refresh
  - Higher contrast
- **Color LCD**
  - Color filters aligned with the liquid crystals.

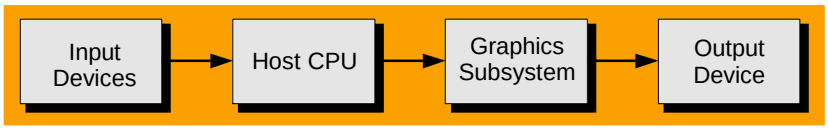


Graphics Architecture Fundamentals

- **Review of the basic structure of graphics computers**
- **Frame Buffer Access Problem**

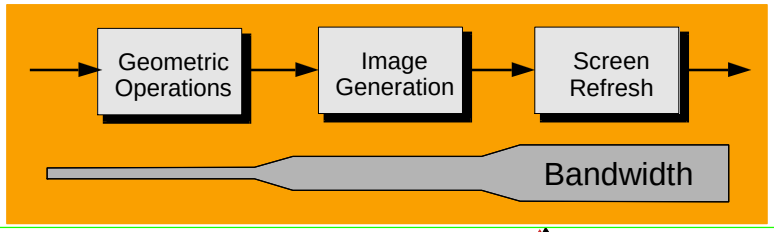
Graphics Computers

- **Host**
  - Operating system
  - Application software
  - Input device polling/interrupts
  - Generation of graphics data
- **Graphics Subsystem**
  - Receive graphics data and commands from host
  - Generate and refresh image on the output device



Graphics Subsystem

- **Geometric Operations**
  - Operates on graphics primitives, e.g. triangles
  - Typically in floating point
  - Basic steps:
    - Modeling + viewing transformations
    - Lighting
    - Clipping
    - Perspective transformations
    - Viewport mapping
- **Image Generation**
  - Different for raster and vector
  - Typically in fixed point
  - Conversion of screen-space primitives into pixels / strokes
- **Screen refresh**
  - For volatile displays (CRT)
  - Illusion of standing image

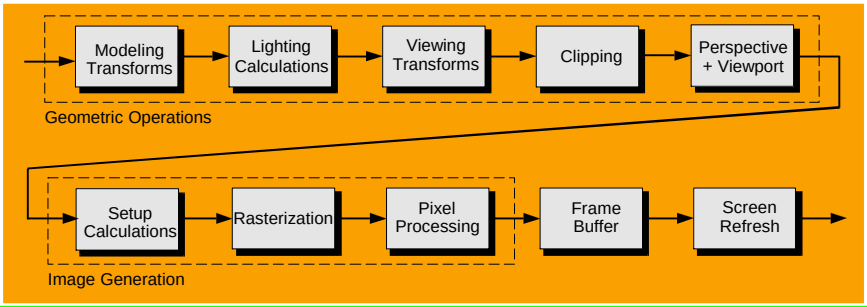


Raster Graphics Systems

- **Raster graphics systems have replaced vector graphics system everywhere but in niche applications, e.g. radar**
- **Principal advantages over vector displays are**
  - ability to display images of arbitrary complexity without flicker
  - display of shaded images instead of wireframes
- **Frame Buffer is the key architectural component of every raster system.**
  - Provides storage for every pixel value
  - Updated by rasterizer or CPU
  - Read by video controller for screen refresh
- **Frame buffer access problem as a result of contention between rasterizer and refresh.**

Raster Graphics Subsystem: Review

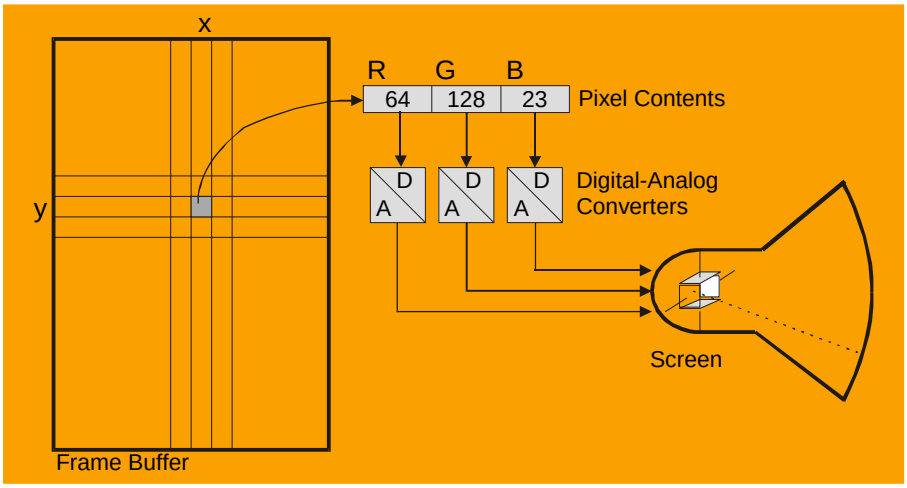
- **Setup**
  - Per-primitive calculations to initialize the rasterization
  - E.g. slopes of edges
- **Rasterization**
  - Break object into pixels
  - Assign pixel values
- **Pixel-processing**
  - Modifies generated pixel values with already stored pixel values, e.g. texture mapping or blending
  - Storing of pixel value into the frame buffer



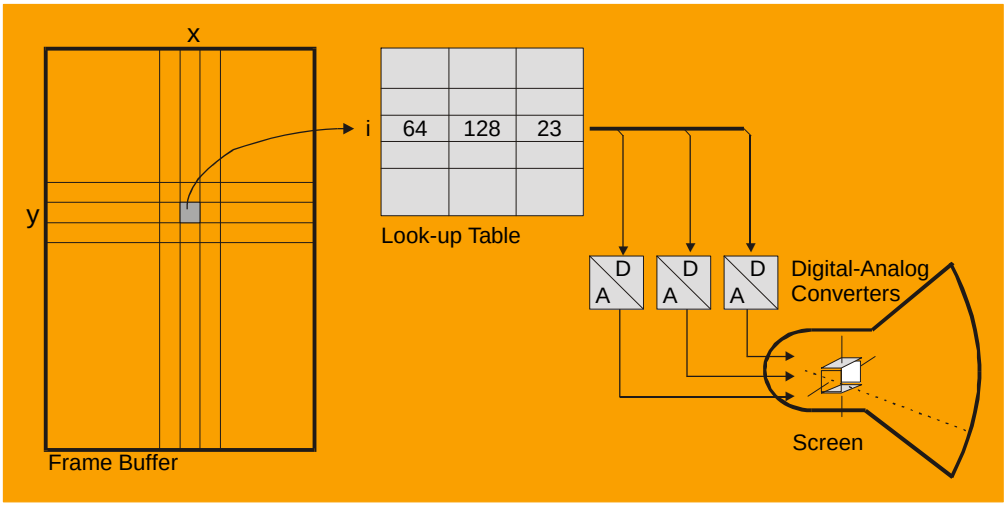
Screen Refresh

- Required for non-persistent display devices
- Creates illusion of standing image
- Refresh rate
  - Minimum approx 30 Hz
  - Ergonomic standards require 80+ Hz
- Refresh logic
  - Continuously reads the frame buffer in scanline order
  - Converts the pixel values into analog signals for monitor

Video Controller: True Color



Video Controller: Color Map



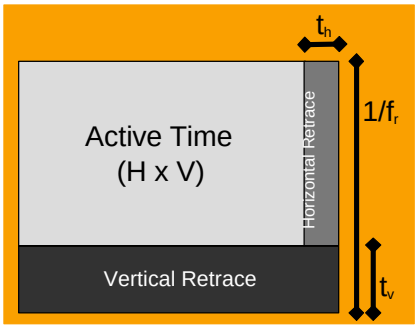
Screen Refresh: Pixel Timing

- **Pixel time:**

$$t_{pixel} = \frac{\left(\frac{1}{f_r} - t_v\right) / V - t_h}{H}$$

$$B_r = \frac{bpp}{t_{pixel}}$$

- **Bandwidth:**



H x V x bpp	f_r *[Hz]*	t_h *[usec]*	t_v *[usec]*	t_{pixel} *[nsec]*	B_r *[MB/sec]*
512 x 512 x 1	60	6.62	1250	45.88	21.8
1024 x 768 x 2	60	4.0	593	16.53	121.0
1280 x 1024 x 3	60	4.0	596	9.14	328.2

Frame Buffer Access Problem

- **Pixel access time during refresh(10-50 nsec) is much shorter than typical DRAM cycle times (100-200 nsec).**
- **Rasterizer and screen refresh compete for access to the frame buffer ... and the rasterizer always loses.**

- **Therefore:**
  - The frame-buffer memory bandwidth must be increased
  - Rasterizer accesses and screen refresh must be decoupled

- **Solutions:**
  - Seminconductor technology
    - Access modes and different memory types, e.g. VRAM, FBRAM, TEXRAM
  - Frame-buffer architecture
    - Multi-bank memory, Double-buffering

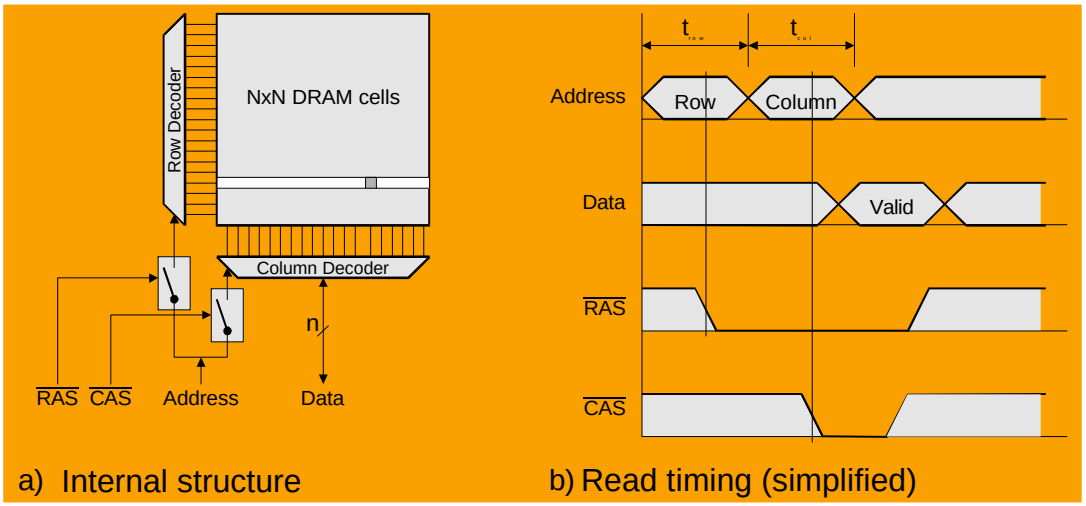
Frame Buffer: Bandwidth Budget

- **Example:**
  - 64-bit memory interface
  - 50 nsec/Byte access time (20 MHz)
  - Bandwidth = 160 MBytes/sec

  - Screen: 1280 x 1024 x 8bpp
  - Refresh rate: 60 Hz
  - Refresh bandwidth = 110 MBytes/sec

- **Bandwidth budget:**
  - 31.25% rasterizer
  - 68.75% screen refresh

Dynamic RAM: Principle

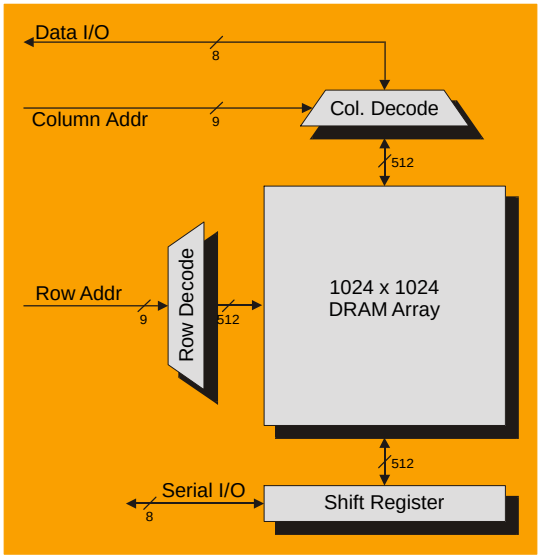


Dynamic RAM: Faster Access

- **Page mode**
  - Accesses into the same page is faster
  - Only repeat through CAS cycle
- **Multi-bank access**
  - Parallel organization of several DRAM chips
  - Each access cycle retrieves multiple bits
  - Applicable to both on-chip organization or multi-chip implementation
  - Problem: Increasing memory density
- **Wider memories organizations**
  - For example: x4, x8, x16 or x32
  - In the limit: access entire row, e.g. embedded memories or VRAM

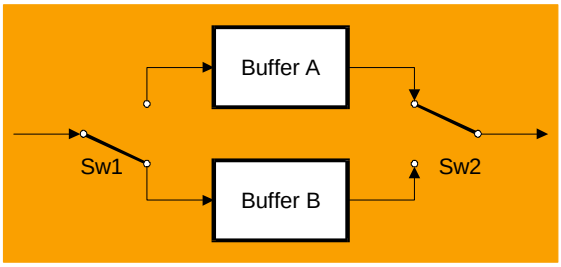
Video RAM

- Dual-ported memory
- Load entire row in one cycle into the shift register
- Shift register I/O is independent from main port I/O
- Bandwidth budget:



Double Buffering

- Frame buffer bandwidth
  - Screen refresh
  - Image generation
- Double-buffering decouples image generation from screen refresh
- Front buffer always contains a complete image while back buffer is rebuilt



Summary of Key Concepts

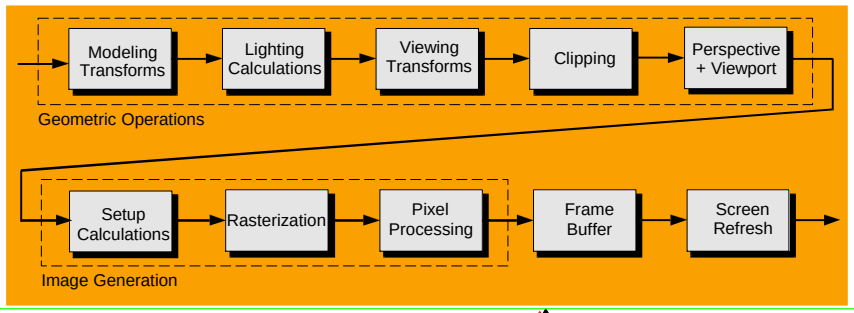
- **Classification criteria for displays**
- **Principles of operation for CRTs and LCDs**
- **Basic structure of graphics systems, both vector and raster displays**
- **Data flow through and function of each block in raster displays**
  - Host CPU - Rasterizer - Frame buffer - Screen refresh
- **Frame-buffer design objectives and solutions**

Computational and Bandwidth Demands

- **Computational Complexity**
  - Floating-point vs. Fixed-point (integer) operations
  - Types of operation (add/subtract, multiply, divide)
  - Memory accesses (read / write)
- **Bandwidth = Data / Time [MB/sec, Mb/sec]**
  - Applies to speed with which datapath components can process, transmit or access data
  - Memory bandwidth is often critical
  - Other datapath elements: processors, busses, D/A converters

Raster Graphics Subsystem

- **Setup**
  - Per-primitive calculations to initialize the rasterization
  - E.g. slopes of edges
- **Rasterization**
  - Break object into pixels
  - Assign pixel values
- **Pixel-processing**
  - Modifies generated pixel values with already stored pixel values, e.g. texture mapping or blending
  - Storing of pixel value into the frame buffer



Per-triangle Operations

Step	Addition	Multiplication	Division	Conversion
Setup				
Rasterization	9		3	3
Attributes	5 + 5a	2 + 3a	1	1
Total	14 + 5a	2 + 3a	4	4

23 + 10a instructions/pixel
(a is the number of attributes to be interpolated across the triangle)

Per-vertex Operations

Step	Addition	Multiplication	Division	Exponentiation
Model transform	16	25		
Phong lighting	6L	3 + 9L		L
View transform	12	16		
Clipping	21 + 3a	18 + 2a	12	
Perspective xform		2	1	
Viewport mapping	2	2		
Total	51 + 3a + 6L	66 + 2a + 9L	13	L

130 + 5a + 16L instructions/pixel

Per-span Operations

Rasterization: 3+a
Additions/span

Per-pixel Operations

Step	Addition	Multiplication	Division	Read	Write
Rasterization	2+a				
Z-test	1			1	
Buffer Clear					2(HxV)
Texture Mapping				1 / 2 / 4 / 8	
Persp. Division			2		
LOD Calculation	7	10	1		
Filtering	0 / 2 / 6 / 14	0 / 1 / 3 / 7			
Update					2
Total	6.5 / 7.5 / 9.5 / 13.5 + a	5 / 5.5 / 6.5 / 8.5	1.5	1.5 / 2 / 3 / 5	1 + 2(HxV)

x0.5

13 / 14.5 / 17.5 / 23.5 + a instructions/pixel
2.5 / 3 / 4 / 6 + 2(HxV) memory accesses per pixel

Summary: Operations

- **Per vertex: 130+5a+16L**
- **Per triangle: 23 + 10a**
- **Per span: 3 + a**
- **Per pixel: 13 / 14 / 17.5 / 23.5 + a**

Performance Model (1)

$$T = n_t \cdot T_t + n_v \cdot T_v + n_s \cdot T_s + n_p \cdot T_p + T_0$$

- T Total rendering time [sec]
- T_t Rendering time per triangle [sec/triangle]
- T_v Rendering time per vertex [sec/vertex]
- T_s Rendering time per span [sec/span]
- T_p Rendering time per pixel [sec/pixel]
- T_0 Rendering time overhead [sec]

- n_t Number of triangles
- n_v Number of vertices
- n_s Number of spans
- n_p Number of pixels

Performance Model (2)

- **Linear approximation of the true performance characteristics**
  - Ignores 2nd order effects
    - Cache/page misses
    - Contention for shared resources like busses or memory
  - Linear model is useful in many practical applications
    - Performance characterization, e.g. benchmarking
    - Performance prediction, e.g. adaptive systems

$$T = n_t \cdot T_t + n_v \cdot T_v + n_s \cdot T_s + n_p \cdot T_p + T_0$$

Data Representation

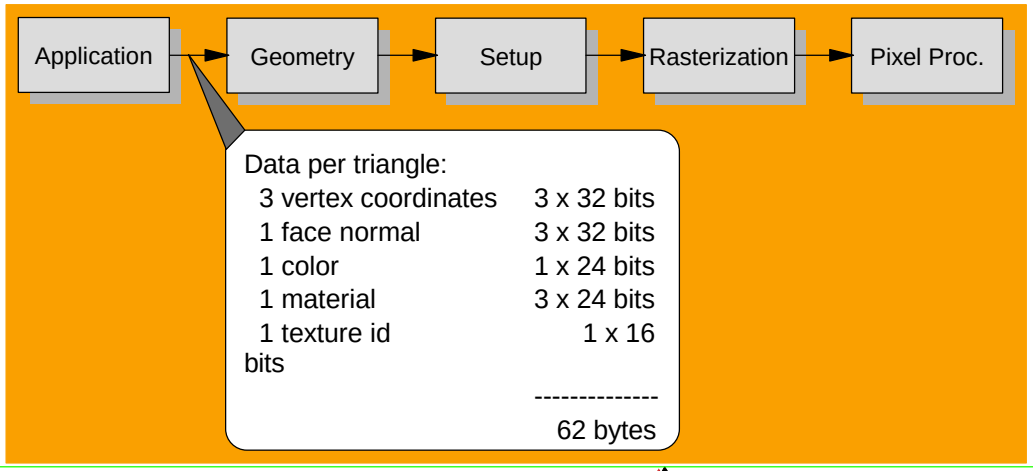
- **Colors: Integer**
  - 24 bits (8-8-8)
  - 8 bits (color index mode)
- **Depth: Integer**
  - 16, 24, 32 bits
- **Texture coordinates: Integer**
  - 16 bits
- **Texture values: Integer**
  - 1, 4, 8 bits (color indices)
  - 16, 24 (5-6-5, 8-8-8)
- **Slopes: Fixed point**
  - Integer: Maximum pixel address
  - Fraction: Maximum pixel address

Bandwidth in the Raster Pipeline

- **As data travel through the raster pipeline, data are described at increasingly lower levels of abstraction**
  - Application: High-level objects, procedural objects
  - Geometry pipeline: triangles, polygons, text, ...
  - Setup stage: Triangles
  - Rasterization: Simple triangles, edges, slopes
  - Pixel processing: Pixel fragments
  - Frame buffer: Pixels
- **Therefore, the same information is described less concisely**
  - The bandwidth in the raster pipeline must increase along the pipeline in order to avoid pipeline stalls

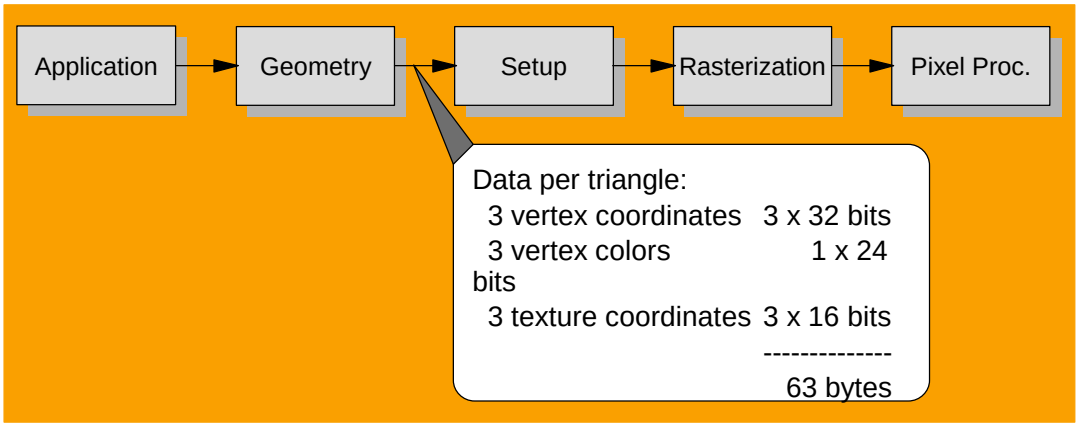
Bandwidth: Application - Geometry

- Computational demands depend on the application
- Send scene description, i.e. objects, materials, lights
- Less bandwidth for triangle strips !



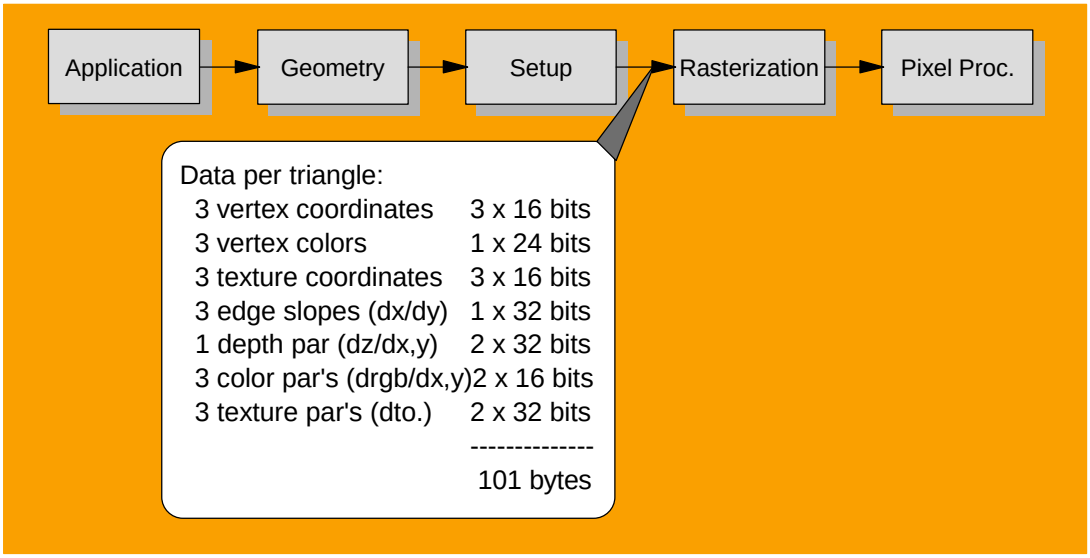
Bandwidth: Geometry - Setup

- Transformed, projected, lit and texture-mapped vertices
- Less bandwidth for triangle strips !



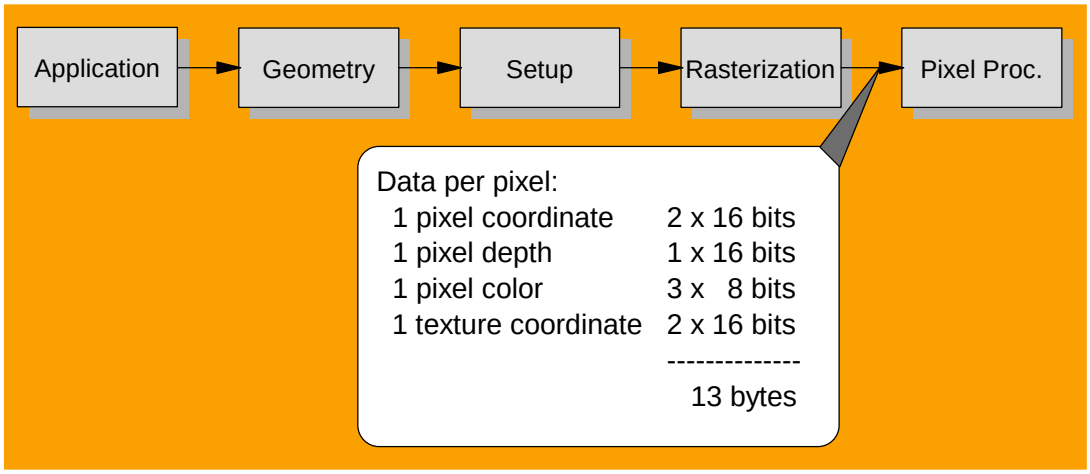
Bandwidth: Setup - Rasterization

- Compute edge slopes and plane equations



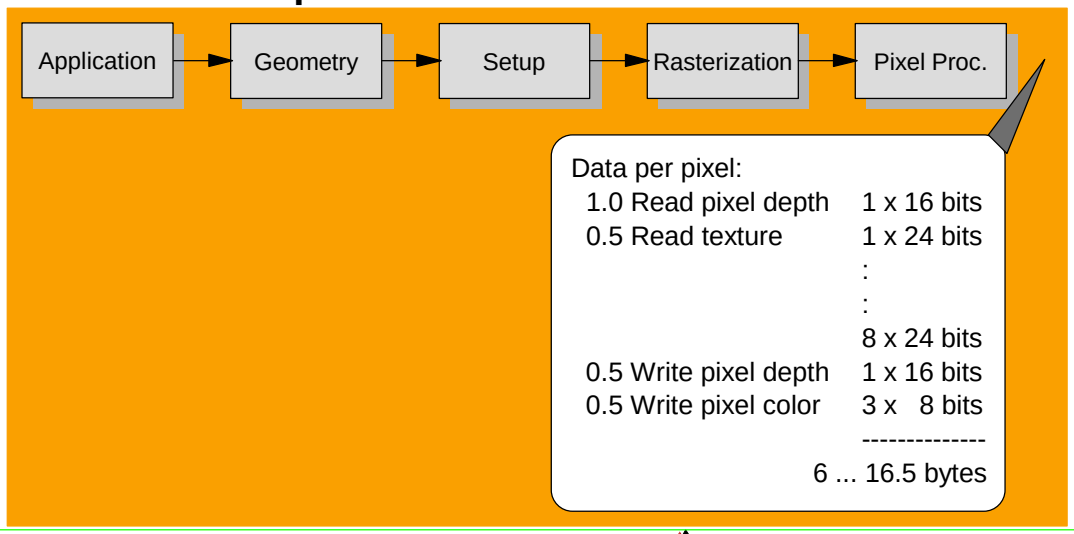
Bandwidth: Rasterizer - Pixel Proc.

- Generate pixels covered by triangle

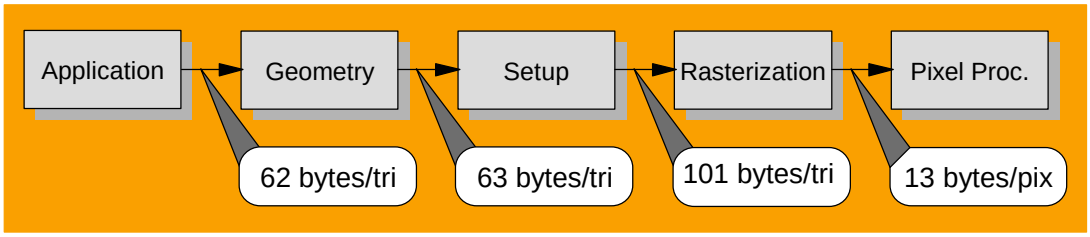


Bandwidth: Pixel Proc. - Memory

- Access texture memory and store into frame buffer
- Assume 50% pass on z-test



Bandwidth: Summary



Example: Description

- Gouraud shading w/ Phong lighting
- 2 light sources
- Z-buffering
- Texturing
  - Perspective correction
  - True LOD calculations
- a=7 (RGBzrst)
- Window: 640x480
- 24 bit depth
- 24 bit color
- 24 texture values
- 100k triangles/scene
  - 1.2 vertices/triangle (10 triangles/strip)
  - 50 pixels/triangle
  - 10 spans/triangle
- 120k vertices/scene
- 5M pixels/scene
- 1M spans/scene

Example: MIPS and MBytes/sec

- For triangle processing
  - 9.3 M instructions
- For vertex processing
  - 23.64 M instructions
- For span processing
  - 10 M instructions
- For pixel processing
  - 100 / 105 / 122.5 / 152.5 M instructions
- Total
  - 33 M FP instructions
  - 110-160 million fixed point instructions
- Pixel memory accesses
  - Read: 7.5 / 10 / 15 / 25 MBytes
  - Write: 6.8 MBytes
- Refresh rate: 30 Hz
  - 1 GFLOPS
  - 3.3 - 4.8 GIPS
  - 430 - 955 MBytes/sec

Performance Bottlenecks

- **Geometry-limited**
  - Vertex computations
  - Mostly compute limited
  - Lights, lighting model
  - Shading model
- **Setup-limited**
  - Triangle computations
  - Mostly compute limited
  - Number of attributes
  - Number of independent triangles
- **Pixel-limited**
  - Mostly memory bandwidth limited
  - Screen resolution
  - Possible starvation by screen refresh
  - Texturing filtering and blending modes

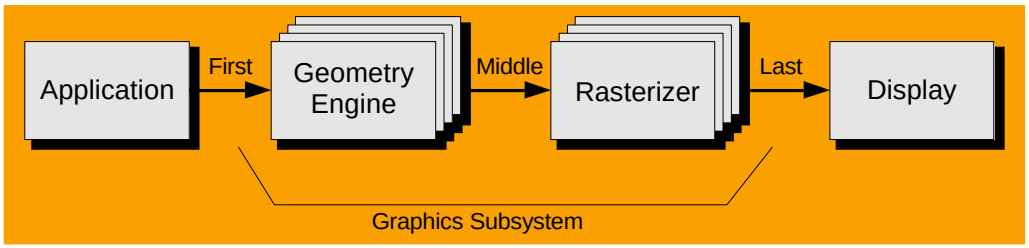
Parallel architectures to widen the bottlenecks

Parallel Raster Architectures

- **Sorting**
- **Data Parallelism**

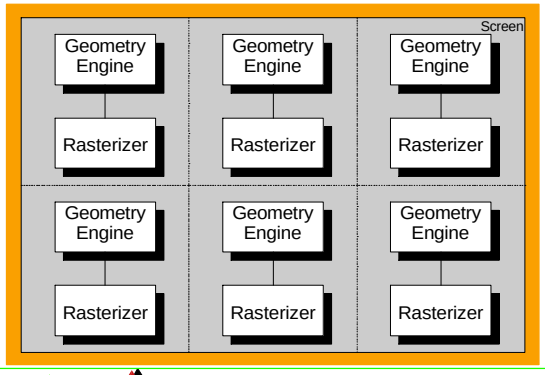
Sorting Classification [Molnar91]

- Geometry processing and rasterization implemented with multiple processors
- Objects are sorted and assigned to these processors
- Classification according to where in the pipeline sorting is done



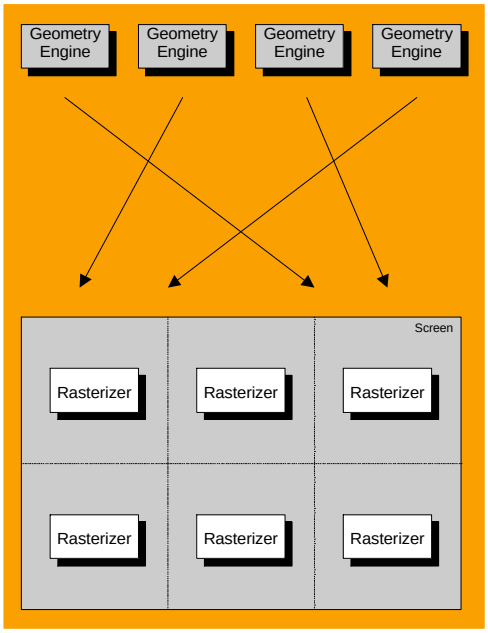
Sort-First (Dynamic object sort)

- **Principle**
  - One rendering engine per screen region
  - Polygons are assigned randomly
  - Geometry engines determine screen region and re-distribute objects
  - Objects overlapping several regions are assigned to multiple rendering engines
- **Properties**
  - Poor load balancing due to scene statistics
  - Mostly independent operation
  - Boundaries: complications + extra processing

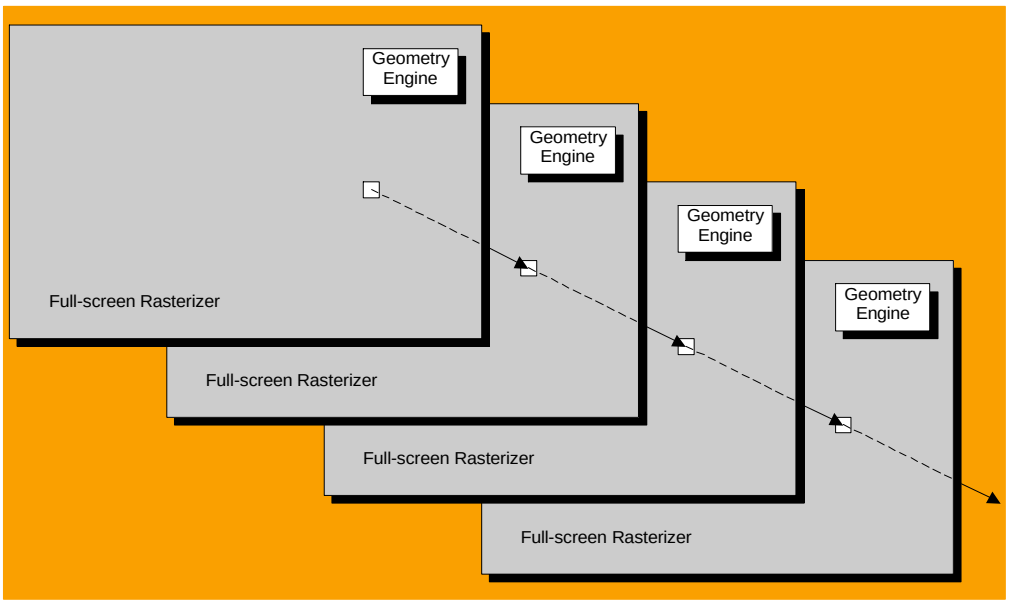


Sort-Middle (Screen subdivision)

- **Principle**
  - Rasterizers are statically assigned to screen regions
  - Objects are randomly assigned to geometry engines
  - GE transfer transformed objects to proper rasterizer
- **Properties**
  - Distributed database
  - Doesn't maintain object ordering
  - High latency due to sorting
  - Good load balancing among geometry engines
  - Load imbalance among rasterizer
  - Many-to-many communication

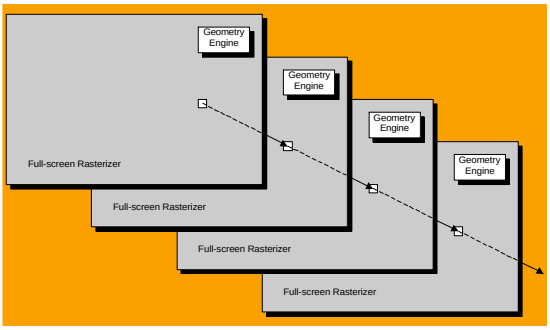


Sort-Last (Image compositing) (1)



Sort-Last (Image compositing) (2)

- **Principle**
  - Statically coupled geometry engine and rasterizer
  - Each rendering engine covers the entire screen area
  - Objects are distributed to the rendering engines randomly
  - The partial images are combined in a pixel compositing network
- **Properties**
  - Automatic load balancing
  - Only local communication
  - Linear scaling
  - Distribute data base
  - High-speed compositing with z-buffer complex and costly
  - Problems with anti-aliasing



Data Parallelism Classification

- **Geometry processing and rasterization implemented with multiple processors**
- **Data are objects and pixels**
- **Classification according to whether multiple pixels or multiple objects are processed in parallel**

Image-space Parallelism

- Pixels or screen regions are processed in parallel
- Objects are processed sequentially
- Pixel storage, i.e. frame buffer, is required since objects are processed in arbitrary order

Object-space Parallelism

- Primitives or groups of primitives (objects) are processed in parallel
- Pixels are generated sequentially
- Pixel storage, i.e. frame buffer, is not absolutely necessary because pixels can be generated in scan order.
  - However, is often included because processors cannot maintain perfect pixel synchronism.

Image vs. Object Parallelism

Properties	Image Space	Object Space
Screen resolution	More processors	Faster processors
Pixel throughput	More processors	Faster processors
Object throughput	Faster processors	More processors
Number of objects	Faster processors	More processors
Max number of processors	Number of pixels	Number of objects
Primitive types	Fixed, all processors handle the same type	Configurable, new processor for new primitive type

Summary: Graphics Hardware

- **Structure of display devices: CRT and LCD**
- **Basics of graphics architectures**
- **Screen refresh and frame buffer access problem**
- **Performance modeling and bandwidth issues**
- **Basics of parallel graphics architectures**

Further Reading

- **Displays:**
  D. Bosman**,** *Display Engineering*, North Holland, 1989

- **Frame Buffer Issues**
  Mary C. Whitton, *Memory Design for Raster Graphics Displays*, IEEE Computer Graphics & Applications, March 1984

Color

Color

- **Color Specification**
  - Perceptual
  - Physical and Psychophysical
  - Color Spaces

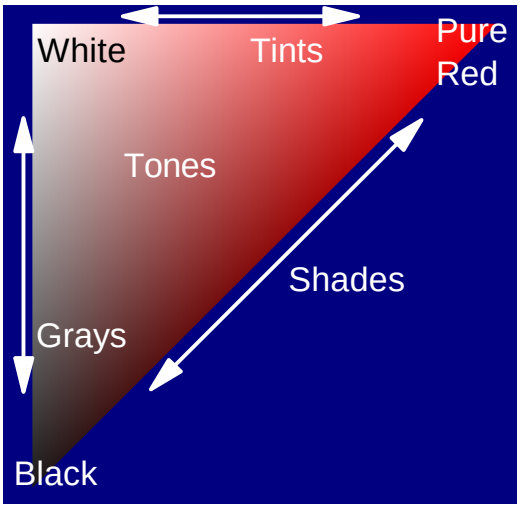
Perceptual Color Specification (1)

- **Color based on perception typically distinguishes between 3 components**

- **Hue**
  - Based on the dominant wavelength
  - The "actual" color, e.g. red, yellow, brown
- **Lightness**
  - The (achromatic) intensity of the color
    - Brightness: For luminous objects (light emitter), e.g. CRT
    - Lightness: For reflecting objects, e.g. paper, wood.
- **Saturation**
  - Distance of the color from a gray of equal intensity
  - Saturated are more brilliant than unsaturated colors
  - Unsaturated color contain more white, e.g. pastels

Perceptual Color Specification (2)

- **Artists often use slightly different terms:**
  - Pure Pigments
    - Saturated colors
  - Tints
    - Add white pigments to pure color
    - Reduces saturation
    - Increase lightness
  - Shades
    - Add black pigments to pure color
    - Reduces lightness and saturation
  - Tones
    - Add black and white pigments

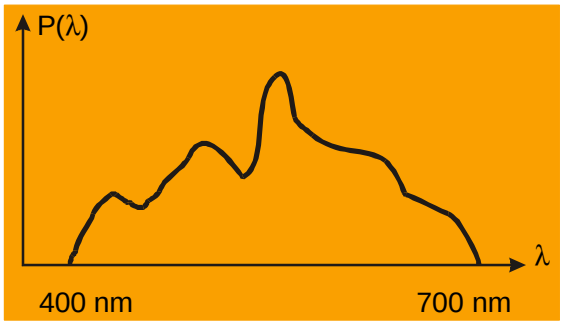


Perceptual Color Specification (3)

- **Textual Specification**
  - Mix of pure colors, e.g. "Yellowish Green"
  - Shades of pure colors, e.g. "Dark Blue"
  - In reference to shared (?) perception, e.g. "Dusty Blue"
- **By Reference to Samples**
  - Often used in printing, graphic design and interior design (paint chips), e.g. Munsell color-order system
  - Fairly elaborate color catalogs
- **All perceptual color specifications are subjective**
  - Differ between observers
  - Color impression depends on lighting and surrounding colors

Physical Color Specification (1)

- **Colorimetry**
- **Color is described by a spectrum**
  - Spectral energy distribution *P(l)* determines the color
  - Same *perceived* color can be created by several spectra called *metamers*
- **Spectra are continuous or discrete**
  - Only one spectral line, are called *monochromatic*
  - Constant energy across the entire spectrum: *achromatic*



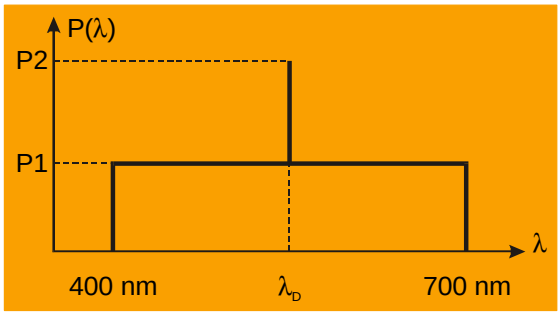
Physical Color Specification (2)

- **Color in physical terms is defined by**
  - Dominant wavelength
  - Excitation purity
  - Luminance
- **There is loose relationship between the terms used in both specification systems:**

Perceptual	Colorimetry
Hue	Dominant Wavelength
Saturation	Excitation Purity
Lightness	Luminance
Brightness	Luminance

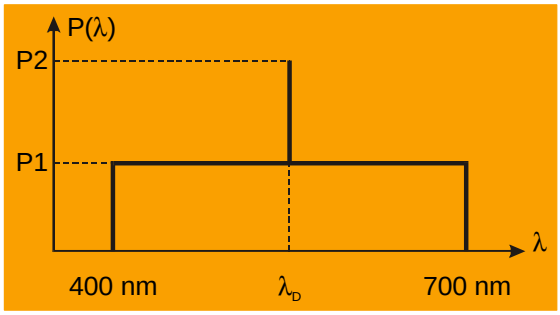
Dominant Wavelength

- **The color we "see"**
  - Wavelength of monochromatic stimulus that, when mixed with an achromatic stimulus, matches the given color.
  - Could be a metamer of the actual color.
  - Not always the wavelength with the largest amplitude !!
- **Some colors do not have a dominant wavelength**
  - Then, it is possible to match a color by adding an amount of the complementary color the achromatic stimulus, e.g. Pink.



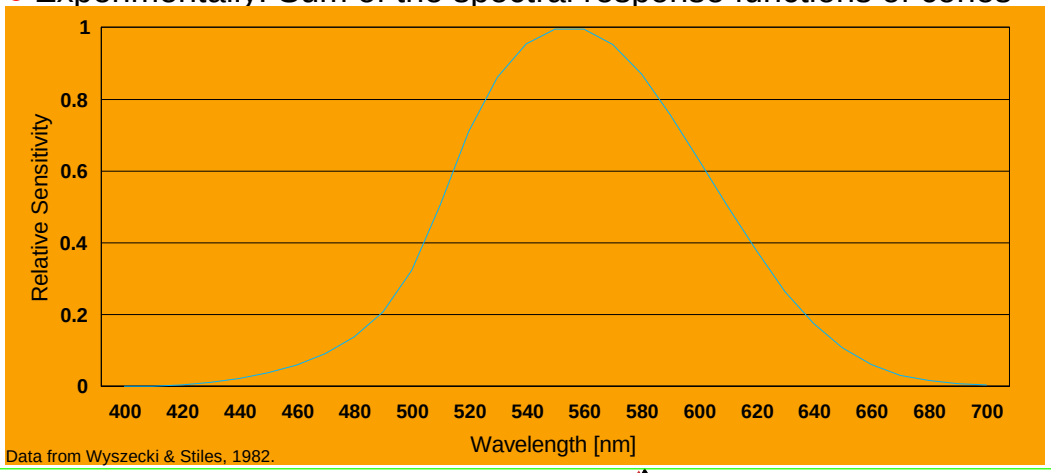
Excitation Purity & Luminance

- **Excitation Purity**
  - Saturation of the color
  - Determined by the size of P1 and P2
    - P1 = P2: Excitation purity is 0%
    - P1 = 0: Excitation purity is 100%.
- **Luminance**
  - Intensity
  - Measured as the area under spectral energy curve, taking into account both P1 and P2.
  - Area is weighted by luminous efficiency function to account for human visual system.



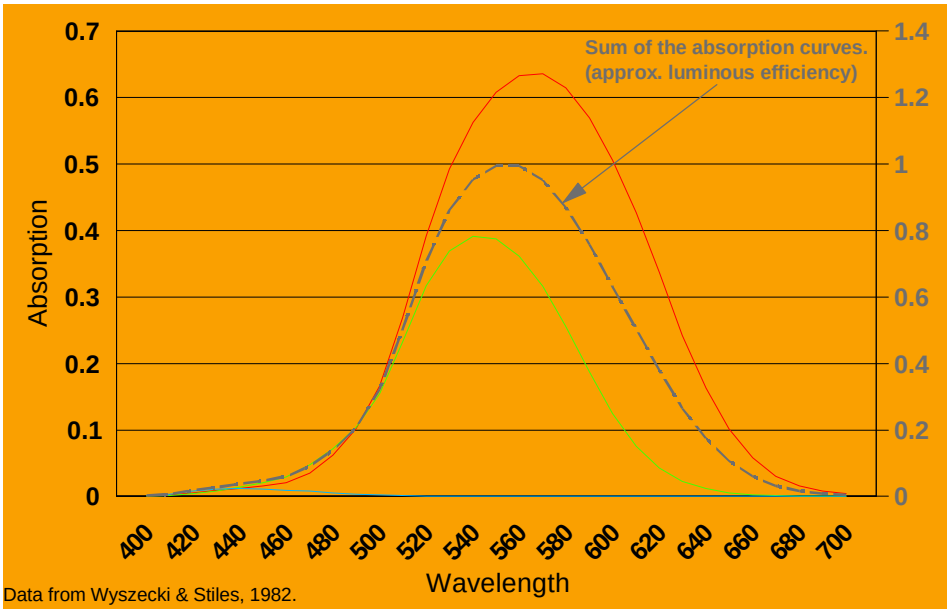
Luminous Efficiency

- **Perceived brightness for light of constant luminance**
  - Sensitivity of the human visual system to different wavelengths
  - Peak sensitivity at 555 nm (yellow-green)
  - Experimentally: Sum of the spectral response functions of cones



Data from Wyszecki & Stiles, 1982.

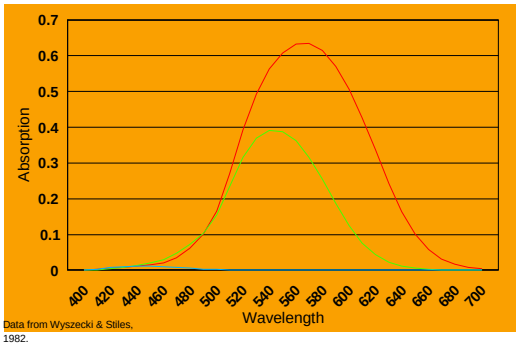
Spectral Response of Cones (1)



Data from Wyszecki & Stiles, 1982.

Spectral Response of Cones (2)

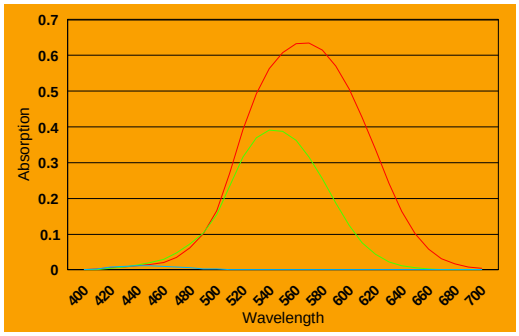
- **The human eye contains 3 types of cones**
  - Tristimulus response
  - Maximum sensitivity at 440 (blue), 550 (yellow-green) and 570 (yellow) nm resp.
  - Blue sensitivity is much lower than for green and red
  - The tristimulus theory is based on experiments with primates and psychophysical studies, e.g. color-blindness.



Data from Wyszecki & Stiles, 1982.

Spectral Response of Cones (2)

- **The spectral sensitivity functions are filters**
  - Incoming signals, i.e. colors, must be multiplied by by the filter function
  - Each receptor (cone) integrates over all received wavelength (after the filter)
  - These three filtered signal are the basis for color perception



$$R = \int c(\lambda) \cdot r(\lambda) \cdot d\lambda$$

$$G = \int c(\lambda) \cdot g(\lambda) \cdot d\lambda$$

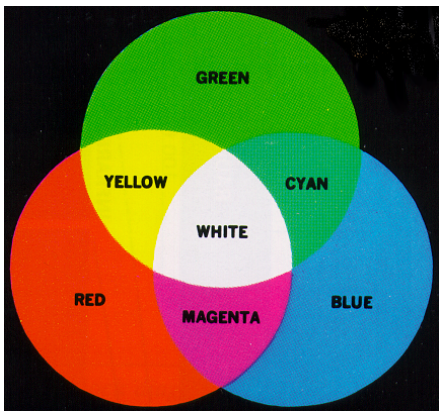
$$B = \int c(\lambda) \cdot b(\lambda) \cdot d\lambda$$

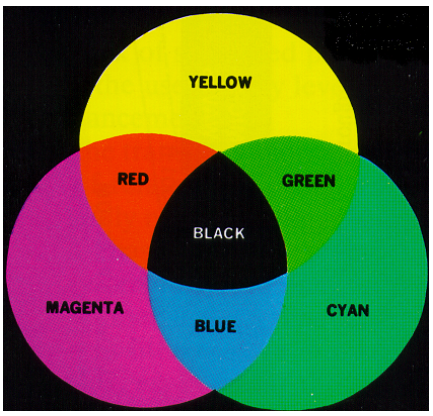
Metamers

- **The tristimulus theory provides an explanation of metamers**
  - Spectra that create the same signals in the photorecptors are perceived as the same color, although the spectra may differ !
  - Monochromatic (spectral) lights create unique responses.
  - Therefore, two spectral colors cannot be metamers for each other.
  - Note: Metamers are a solely perceptual effect. Instruments can distinguish between metameric lights.

Color Mixing

- **Tristimulus theory suggests that all colors can be created by stimulating the three types of cones to different extents.**
  - This is the principle of additive and subtractive color mixing

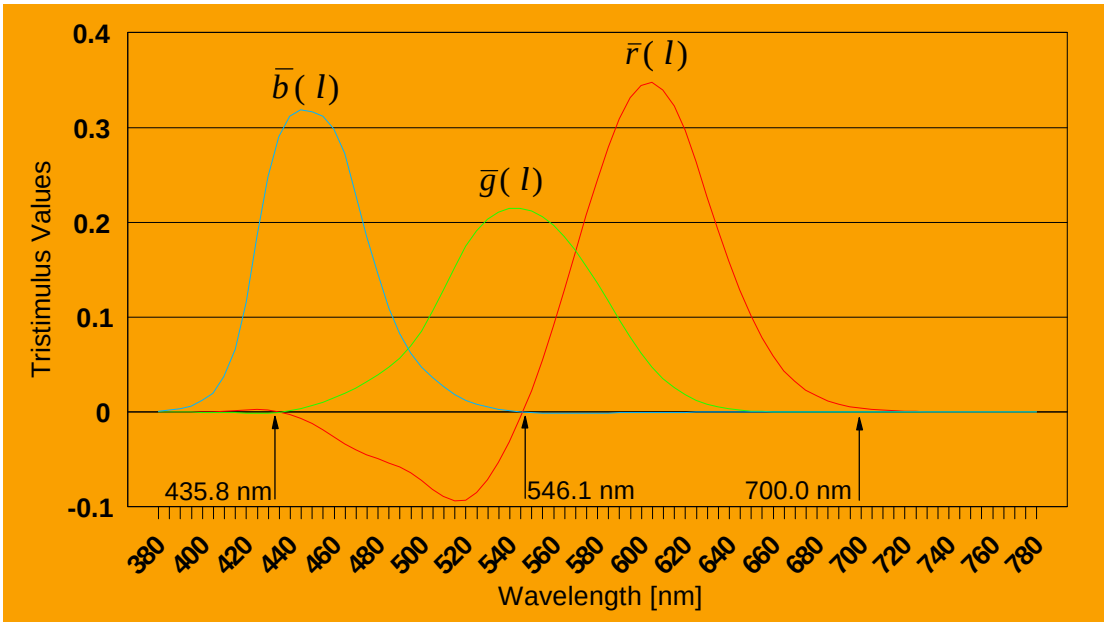




Tristimulus Color Matching

- **Tristimulus theory suggests that all colors can be created by stimulating the three types of cones to different extents.**
  - Many colors (e.g. CRT) can be generated that way ... *but not all* !
- **Color Matching Experiment**
  - Synthesize a sample color by additive mixing of 3 spectral colors
    - Blue: 435.8 nm, Green: 546.1, Red: 700 nm
    - Relative radiant power B:G:R = 72.1 : 1.4 : 1.0
  - Luminance for each color can be adjusted between -1 and +1
  - Luminance <0 means adding it to the sample
  - For a few colors, Red must be subtracted to match the sample color, i.e. Red had to be added to the color sample.
  - The experiment results in a set of wavelength-dependent color matching functions needed to match all monochromatic colors.

Tristimulus Color Matching Functions (1)



Tristimulus Color Matching Functions (2)

- **These functions can be normalized so that only 2 of the 3 functions are independent**
  - One can be computed given the two other functions
  - Typically, $r(\lambda)$ and $g(\lambda)$ are chosen as the independent functions
  - Plotting the two independent matching functions creates *(r,g)*-Chromaticity Diagram

$$r(\lambda) = \frac{\bar{r}(\lambda)}{\bar{r}(\lambda)+\bar{g}(\lambda)+\bar{b}(\lambda)}$$

$$g(\lambda) = \frac{\bar{g}(\lambda)}{\bar{r}(\lambda)+\bar{g}(\lambda)+\bar{b}(\lambda)}$$

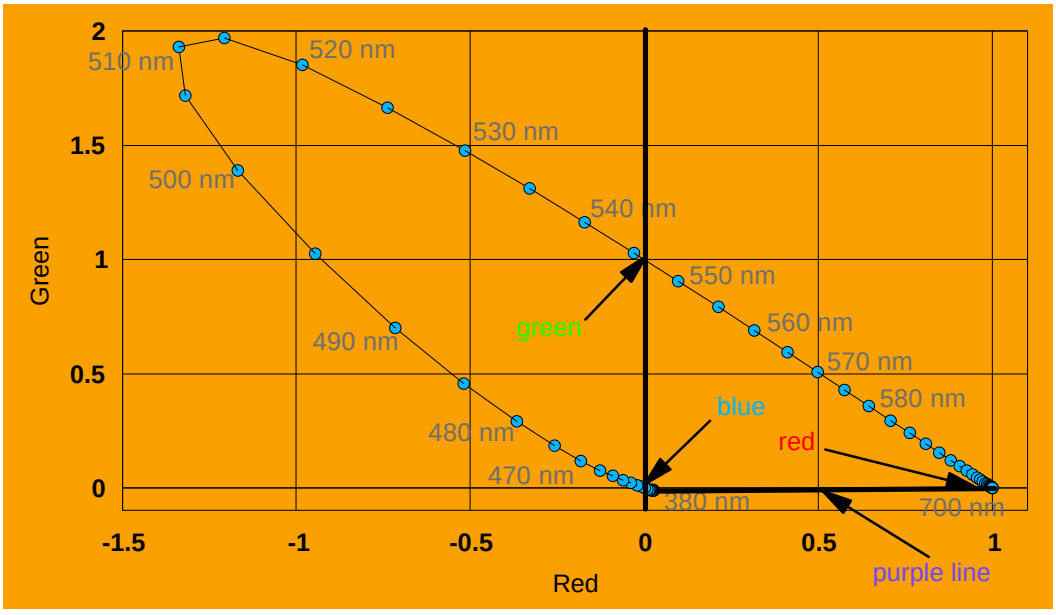
$$b(\lambda) = \frac{\bar{b}(\lambda)}{\bar{r}(\lambda)+\bar{g}(\lambda)+\bar{b}(\lambda)}$$

$$r(\lambda)+g(\lambda)+b(\lambda)=1$$

$$\Rightarrow$$

$$b(\lambda)=1-r(\lambda)-g(\lambda)$$

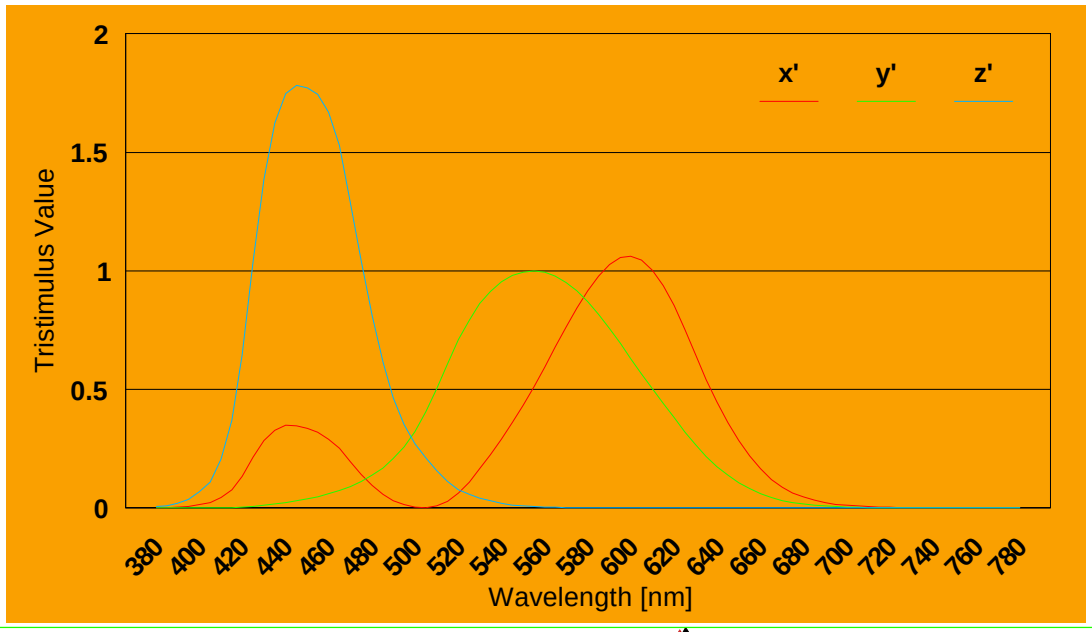
Tristimulus *(r,g)*-Chromaticity Diagram



CIE Color Matching Functions (1)

- **The negative contributions from the tristimulus color matching functions are inconvenient and non-intuitive**
- **CIE color matching functions have only positive weights**
  - CIE = Commission Internationale de l'Eclairage
  - Different primaries ***X***, ***Y*** and ***Z*** that replace R, G, and B
  - Corresponding color matching functions $x'(\lambda)$, $y'(\lambda)$, $z'(\lambda)$.
  - The primary *Y* was chosen, so that the color matching function $y'(\lambda)$ is identical to the luminous efficiency function.
  - Note, that $x'(\lambda)$, $y'(\lambda)$, $z'(\lambda)$ are not spectral distribution of the primary colors ***X***, ***Y***, ***Z***.
  - They simply functions that indicate how much of ***X***, ***Y***, ***Z*** is needed to match a sample color.

CIE Color Matching Functions (2)



CIE Color Matching Functions (3)

- **For a given spectral distribution *P(l)*, we compute the weights *X*, *Y*, *Z* for each of the matching functions as:**

$$X = k\int P(\lambda)\cdot\bar{x}(\lambda)\cdot d\lambda \quad Y = k\int P(\lambda)\cdot\bar{y}(\lambda)\cdot d\lambda \quad Z = k\int P(\lambda)\cdot\bar{z}(\lambda)\cdot d\lambda$$

  - For self-luminous objects: *k*=680 lm/W
  - For reflecting objects, *k* is chosen such that a bright white gives *Y*=100, i.e.

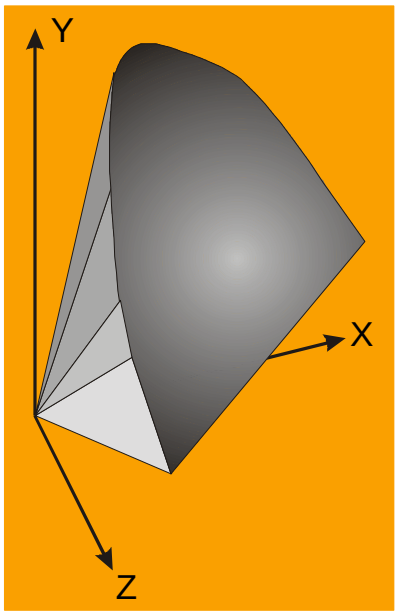
$$k = \frac{100}{\int P_w(\lambda)\cdot\bar{y}(\lambda)\cdot d\lambda}$$

- **Then each color *C* is described as**

$$\mathbf{C} = X\cdot\mathbf{X} + Y\cdot\mathbf{Y} + Z\cdot\mathbf{Z}$$

CIE Color Matching Functions (3)

- **The colors C are contained in the space defined by all valid values of *X, Y, Z*.**



Computer Graphics – Week 12

© Bengt-Olaf Schneider, 1999

CIE Color Matching Functions (4)

- **Similar to the tristimulus color matching functions, the CIE functions can be normalized**

$$x(\lambda) = \frac{\bar{x}(\lambda)}{\bar{x}(\lambda)+\bar{y}(\lambda)+\bar{z}(\lambda)}$$

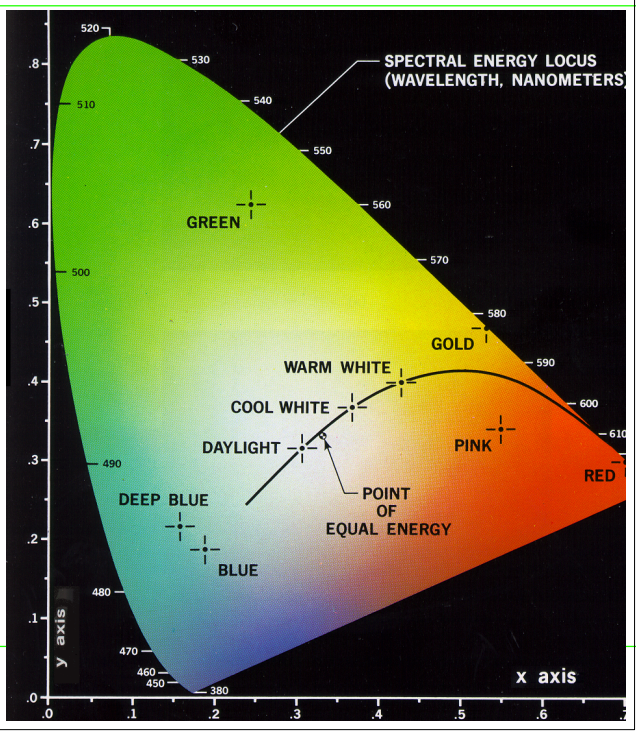
$$y(\lambda) = \frac{\bar{y}(\lambda)}{\bar{x}(\lambda)+\bar{y}(\lambda)+\bar{z}(\lambda)}$$

$$z(\lambda) = \frac{\bar{z}(\lambda)}{\bar{x}(\lambda)+\bar{y}(\lambda)+\bar{z}(\lambda)}$$

$$x(\lambda)+y(\lambda)+z(\lambda)=1$$

CIE Chromaticity Diagram (1)

- **The CIE chromaticity diagram is the projection of the plane *X+Y+Z=1* onto the *XY*-plane.**
  - Monochromatic colors along the border
  - Unsaturated colors in the interior
  - Several white points defined
    - Equal energy, x = y = z = 1/3
    - Sunlight
    - Different "white" lights



CIE Chromaticity Diagram (2)

- **Chromaticity diagram only contains dominant wavelenth and excitation purity**
- **Luminance information is excluded:**
  - Given two values, e.g. *x* and *y*, we can recover the third (here *z*):

$$x(l) + y(l) + z(l) = 1$$

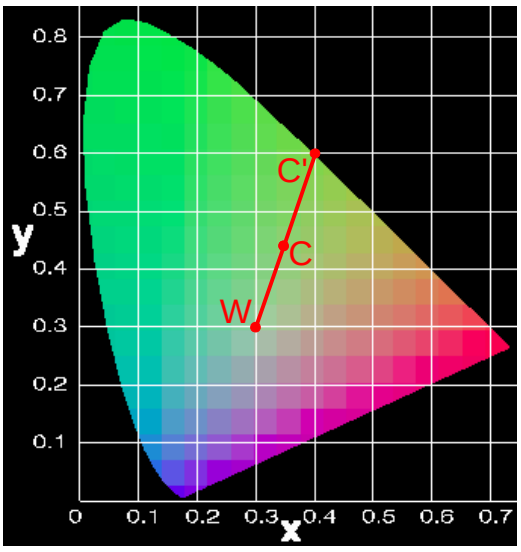
  - However, the original values *X, Y* and *Z* cannot be recoverd !
  - This requires one of the original values, e.g. *Y*. Then we get:

$$X = Y\frac{x}{y} \quad Y = Y \quad Z = Y\frac{1-x-y}{y}$$

  - Remember: The chromaticity contains all visibile colors of a given luminance ... not simply all visible colors !

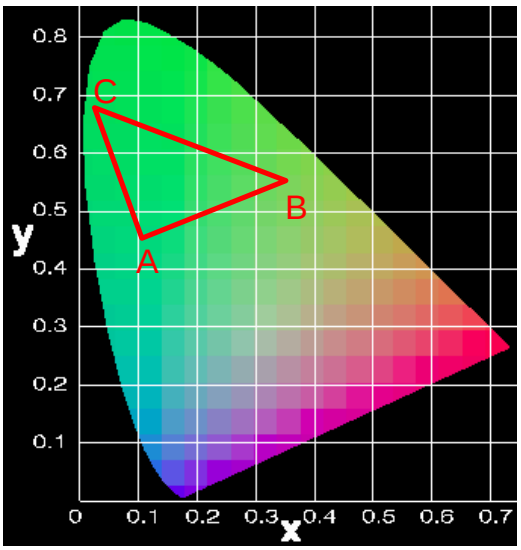
Using the CIE Chromaticity Diagram (1)

- **Dominant wavelenth**
  - Draw line from the white point *W* to the color *C* and continue to the intersection point *C'* with the border of the chromaticity diagram.
  - The intersection point is the dominant wavelenth.
  - Here: approx. 565 nm
- **Excitation purity**
  - The ratio of the *WC / WC'* is the excitation purity.
  - Here: approx. 50%



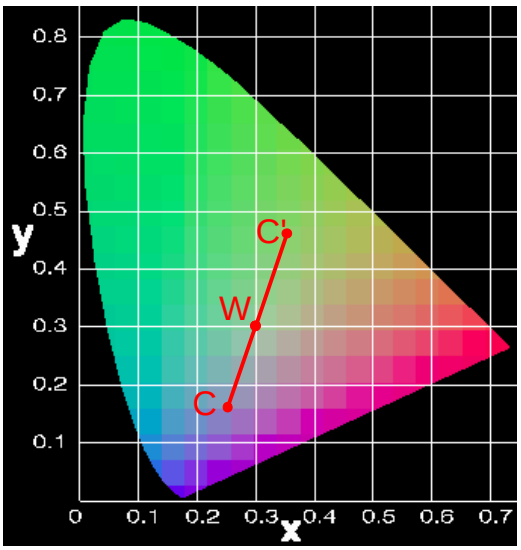
Using the CIE Chromaticity Diagram (2)

- **Mixing Colors**
  - All colors along a line in the chromaticity diagram can be created by mixing the colors at the endpoints, e.g.
  - Similarly, all colors in a convex polygon can be created by mixing the colors a the vertices.



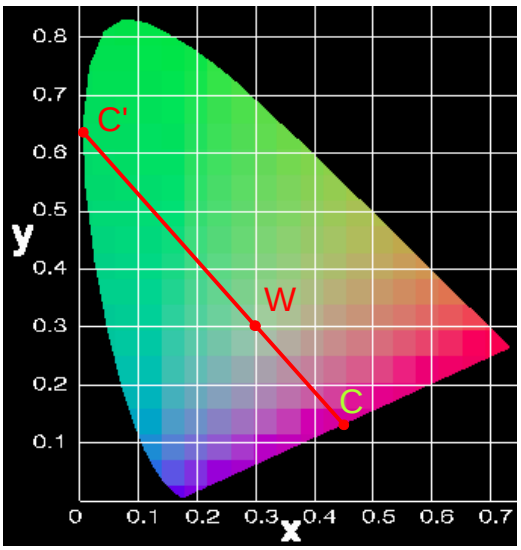
Using the CIE Chromaticity Diagram (3)

- **Complementary Colors**
  - Mixing complementary colors produces white light.
  - C and C' are complementary colors as the midpoint of their connecting line passes through the white point.
  - Note: White can be produced by either 2 colors or by a constant spectrum (also see additive color mixing).



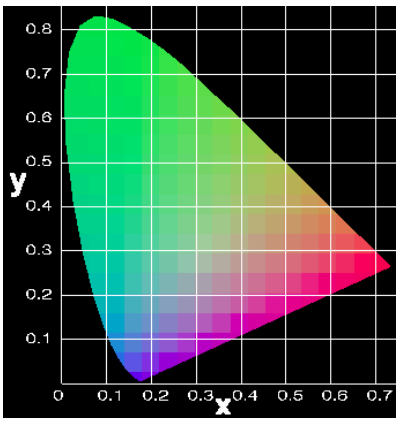
Using the CIE Chromaticity Diagram (4)

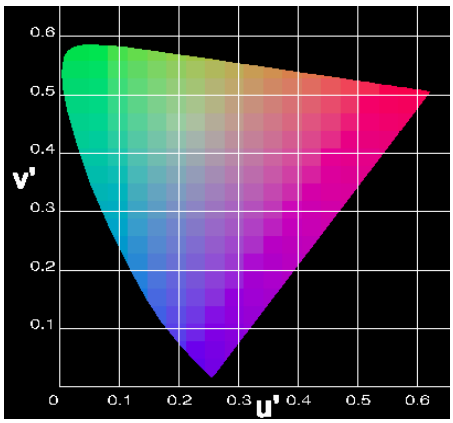
- **Non-spectral Colors**
  - Some colors do not have dominant wavelength
  - All colors that lie on a line from the white point towards the purple line
  - Complement of a spectral color (i.e. color with a dominant wavelength).
  - Excitation purity is defined as before.
  - Here:
    C is the complementary color of *C'* at approx. 505 nm.



Color Distances

- **Colors that visually differ by the same amount appear at different distances in the CIE chromaticity diagram**
  - Another CIE diagram, the CIE LUV defines a uniform color space
  - More intuitive prediction of color difference from color coordinates.



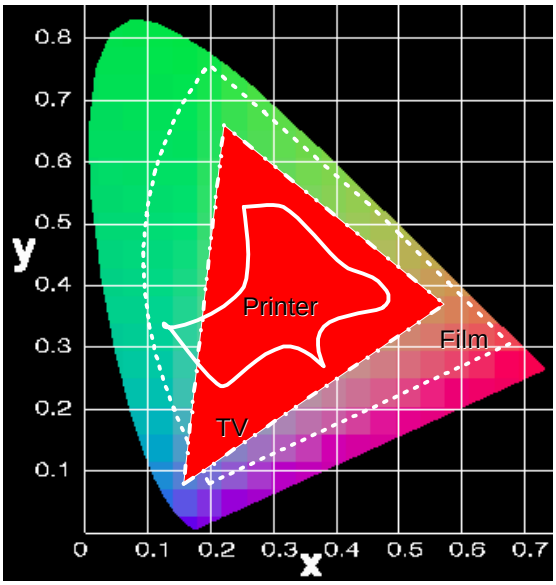


Color Gamuts (1)

- **The CIE chromaticity diagram can describe all visible colors (given a luminance)**
  - Therefore, it can be used to define, name and compare colors
  - We will use the CIE color specification to analyze other color specification schemes
  - Such specifications define a *gamut* in the space of visible color
  - Most gamuts specify colors by mixing several primaries and can therefore not represent all visible colors
    - Remember: Negative contributions are required to produce all colors !

Color Gamuts (2)

- **Color gamuts can be specified for different output devices**
  - To create matching images the use of colors has to restricted to the intersection of the color gamuts
  - Note 1:
    The printer gamut is small, resulting in dull reproduction of screen images
  - Note 2:
    All chromaticity diagrams in these charts are approximate



Summary: Color Specification

- **Perceptual and Physical Color Definition**
  - Correspondence between terms
- **Linear combinations of primary colors can not produce all visible colors**
  - Established using color matching experiments
  - Due to non-linearities of the human visual system
- **CIE Chromaticity Diagram**
  - Uses matching functions that are non-negative for all wavelengths
  - Can be used to define colors and evaluate color models

Further Reading

- **For Color Science:**
  - Guenther Wyszecki, W.S. Stiles, "Color Science: Concepts and Methods, Quantitative Data and Formulae", 2nd edition, John Wiley & Sons, 1982.
- **For YIQ Color Model:**
  - Keith Jack, "Video Demystified", HighText Publications, 1993.

Homework

- Review graphics hardware and color theory
- Prepare volume rendering (chapter 20.6)
- Review entire class material

Next Week ...

- Conclusion of color theory
  - Color models
  - Use of color
- Volume Rendering
- Q + A