Graphics

Jungchan Cho
Dept. of Software, Gachon University

Many slides from Edward Angel and Dave Shreine

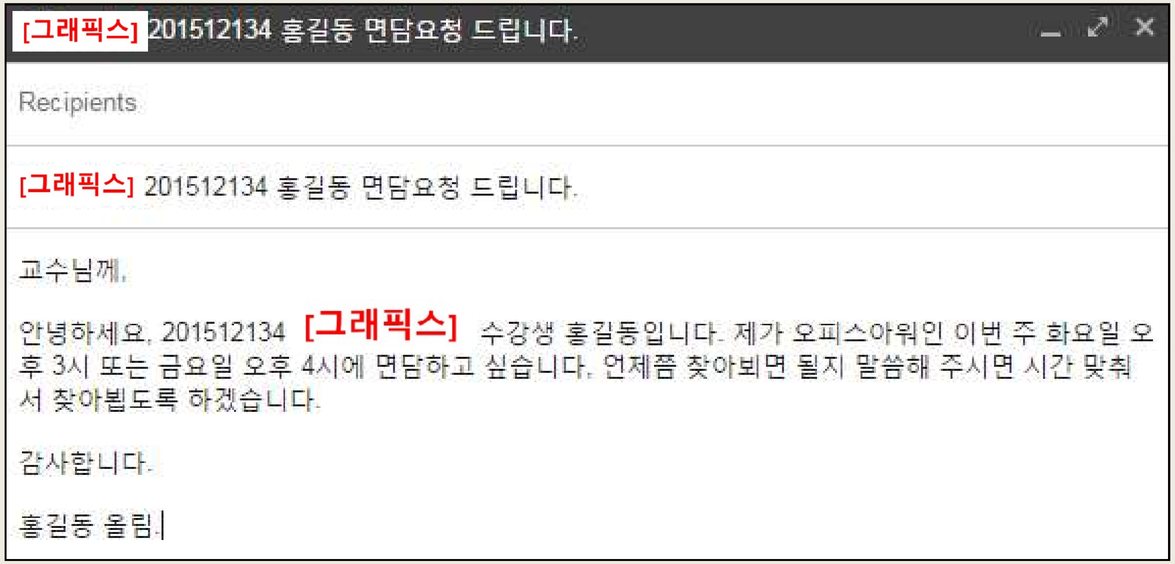
General Information

- Instructor
  - Prof. Jungchan Cho (조정찬)
  - Contact: 산학협력관, #319,
  - thinkai@gachon.ac.kr
  - https://sites.google.com/view/visual-ai
  - Office hours: 매주 화요일 목요일 사전 조율 후 가능

- Undergraduate TA
  - TBD (To-Be-Determined)
  - Office hours: TBA (Announced)

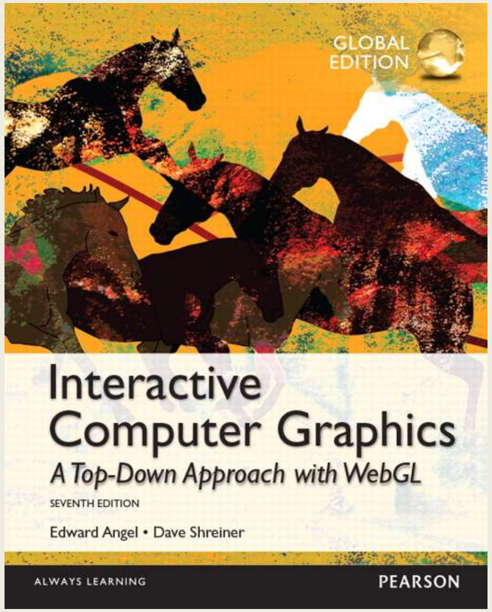
Contacting…

- <u>All contacts via E-mail:</u> with **[그래픽스]** header in mail title



Textbook

- Course Textbook:
  - Interactive Computer Graphics: A Top-Down Approach with WebGL **(7th Edition)**, Edward Angel and Dave Shreiner, Pearson Global Edition



Grading

평가요소	성적 평가방법	비율
출석	학교 규정에 의한 출석미달은 F 임	10
중간고사	중간고사 출석시험	30
기말고시	기말고사 출석시험	30
레포트	프로그래밍과제, 수업퀴즈	15
그룹 프로젝트	프로젝트	15
기타		0
기타 2		0
합 계		100

Objectives

- Broad introduction to Computer Graphics
  - Software
  - Hardware
  - Applications
- Top-down approach
- Shader-Based WebGL
  - Integrates with HTML5
  - Code runs in latest browsers

Prerequisites

- Good programming skills in C (or C++)
- Basic Data Structures
  - Linked lists
  - Arrays
- Geometry
- Simple Linear Algebra

Why is this course different?

- Shader-based
  - Most computer graphics use OpenGL but still use fixed-function pipeline
  - does not require shaders
  - Does not make use of the full capabilities of the graphics processing unit (GPU)
- Web
  - With HTML5, WebGL runs in the latest browsers
  - makes use of local hardware
  - no system dependencies

References

- Interactive Computer Graphics (7th Edition)
- The OpenGL Programmer's Guide (the Redbook) 8th Edition
- OpenGL ES 2.0 Programming Guide
- WebGL Programming Guide
- WebGL Beginner's Guide
- WebGL: Up and Running
- JavaScript: The Definitive Guide

Outline: Part 1

- Introduction
- Text: Chapter 1
  - What is Computer Graphics?
  - Applications Areas
  - History
  - Image formation
  - Basic Architecture

Outline: Part 2

- Basic WebGL Graphics
- Text: Chapter 2
  - Architecture
  - JavaScript
  - Web execution
  - Simple programs in two and three dimensions
  - Basic shaders and GLSL (Open**GL** **S**hading **L**anguage)

Outline: Part 3

- Interaction
- Text: Chapter 3
  - Client-Server Model
  - Event-driven programs
  - Event Listeners
  - Menus, Buttons, Sliders
  - Position input

Outline: Part 4

- Three-Dimensional Graphics
- Text: Chapters 4-6
  - Geometry
  - Transformations
  - Homogeneous Coordinates
  - Viewing
  - Lighting and Shading

Outline: Part 5

- Discrete Methods
- Text: Chapter 7
  - Buffers
  - Pixel Maps
  - Texture Mapping
  - Compositing and Transparency
  - Off-Screen Rendering

Outline: Part 6

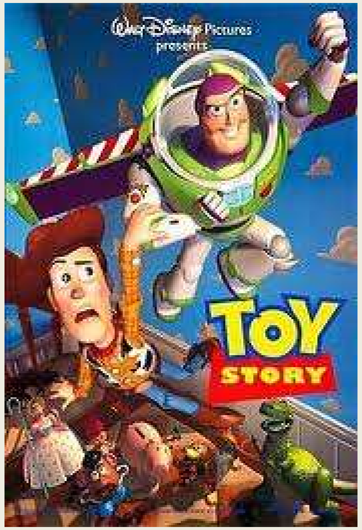
- Hierarchy and Procedural Methods
- Text: Chapters 9-10
- Tree Structured Models
  - Traversal Methods
  - Scene Graphs
  - Particle Systems
  - Agent Based Models

Outline: Part 7

- Advanced Rendering
- Text: Chapter 12

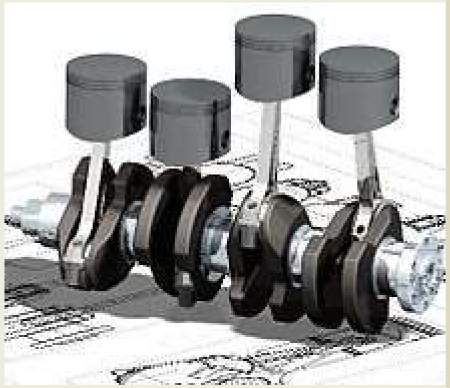
What is computer graphics?

- Graphics created by using computers
  - Representation and manipulation of image data by a com puter with help from specialized software and hardware



Toy Story

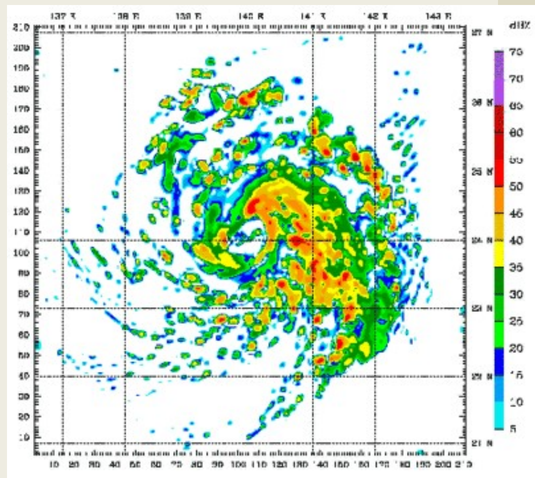
Dragonage II



Compute Aided Design

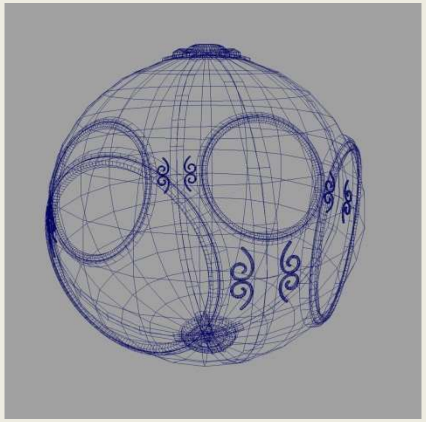
What is computer graphics?

- Computer graphics deals with all aspects of creating images with a computer
  - Hardware – PC, gaming console, smartphone, …
  - Software – Maya (on top of OpenGL), Lightwave, …
  - Application – movie, computer gaming, design, art, …



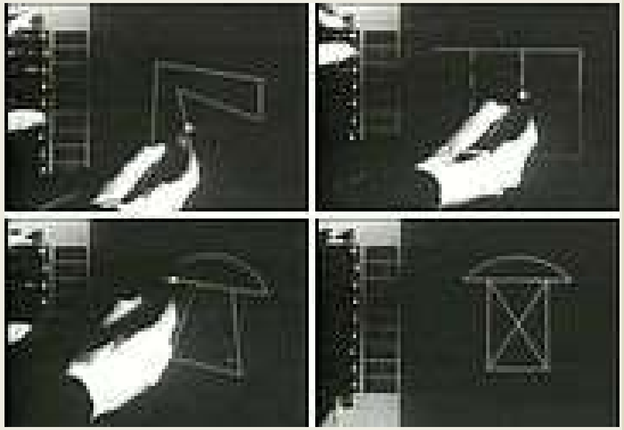
History: 1960-1970

- Wireframe graphics
- SAGE (Semi-Automatic Ground Environment)
- Sutherland's Sketchpad



Wireframe

SAGE Project



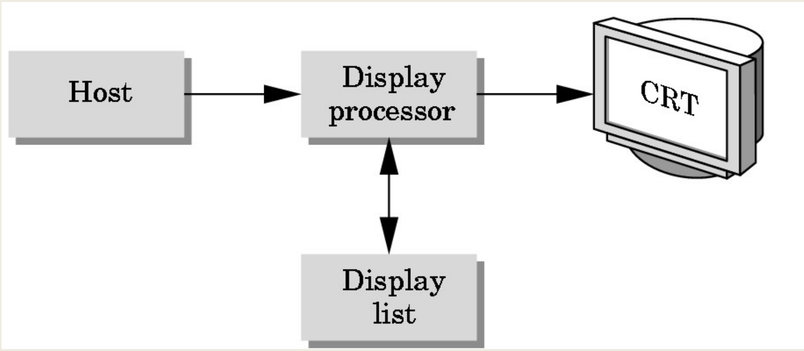
Sutherland's Sketchpad

History: 1960-1970

- Sketchpad
  - Ivan Sutherland's PhD thesis at MIT
  - Recognized the potential of man-machine interaction
    - Even today, many standards of computer graphics interfaces has been inherited from Sketchpad
    - E.g. "select box, location and size" instead of drawing four lines
  - Software to draw objects by using CRT and light pen
    - Display something
    - User moves light pen
    - Computer automatically generates new display in real-time
  - Sutherland also created many of now common algorithms for computer graphics

History: 1960-1970

- Display processor (DPU)
  - Special purpose computer to refresh display



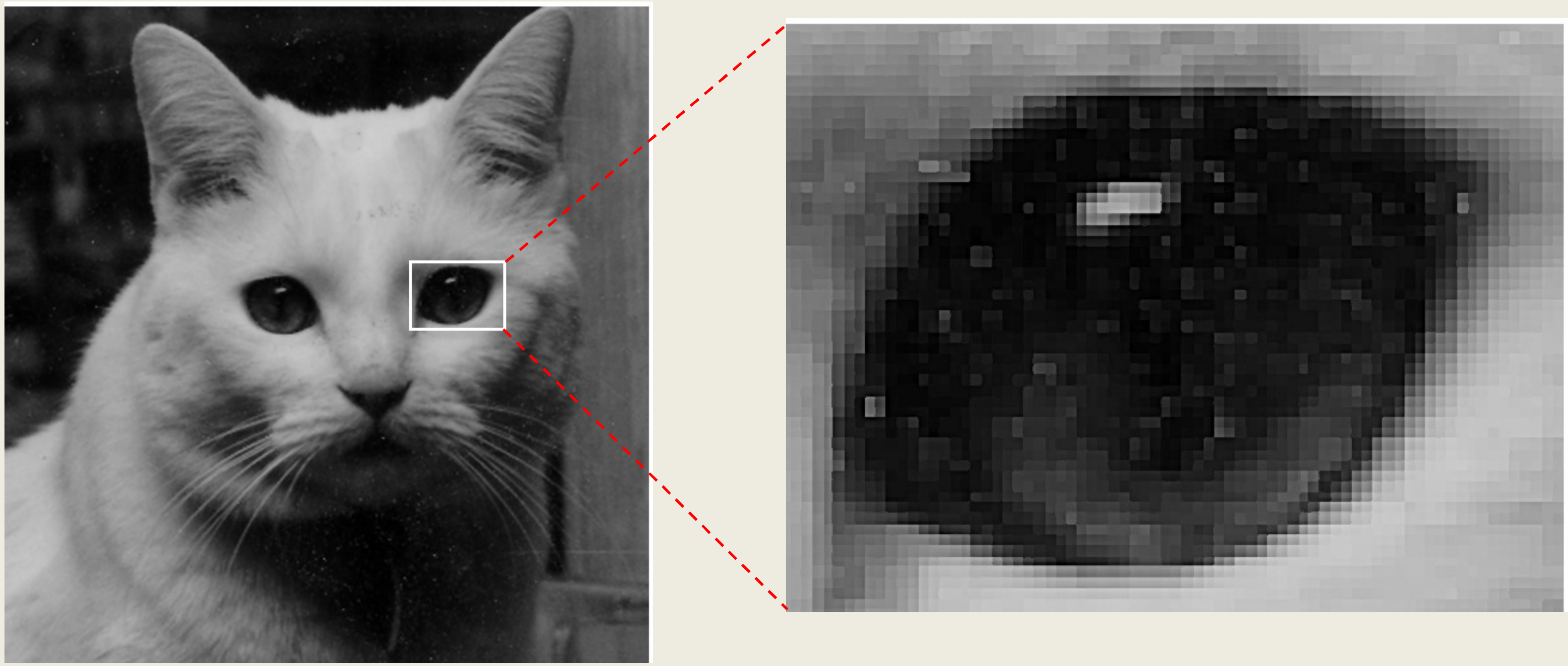
  - Graphics stored in display list on display processor
  - Host compiles display list and sends to DPU

History: 1970-1980

- Raster graphics
- Beginning of graphics standards
  - GKS (Graphical Kernel System)
    - First ISO standard for low-level computer graphics (1977)
    - Set of drawing features for 2D vector graphics
- New hardware: workstation and PC

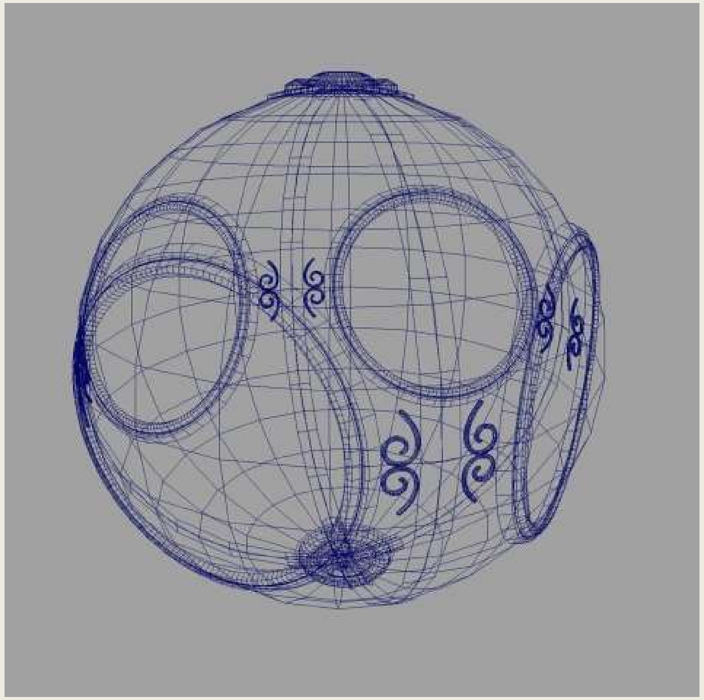
History: 1970-1980

- Raster graphics
  - Image produced as an array (*raster*) of picture elements (*pi xels*) in *frame buffer*

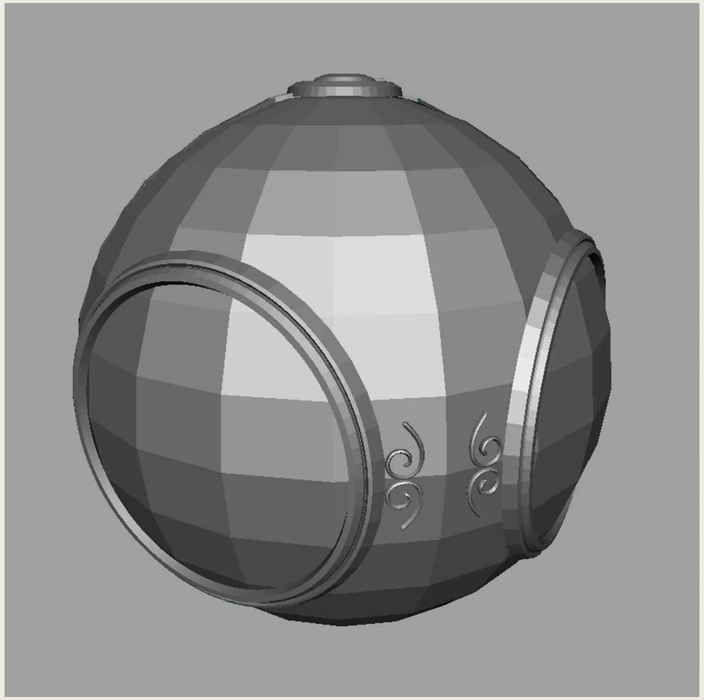


History: 1970-1980

- From wireframe graphics to filled polygons



Wireframe



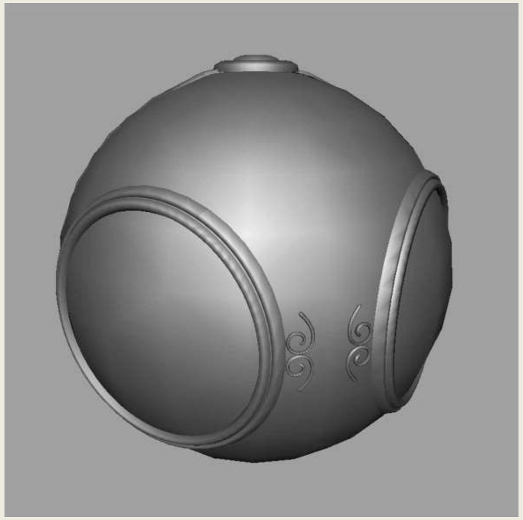
Filled polygons

History: 1970-1980

- New hardware: workstation and PC
  - Workstation
    - Special purpose machine
    - Networked connection: client-server model
    - Dedicated graphics peripherals
      - Display processor, frame buffer, …
  - PC (personal computer)
    - General purpose machine
    - Non-dedicated graphics peripherals
- Currently, there is no distinction between them

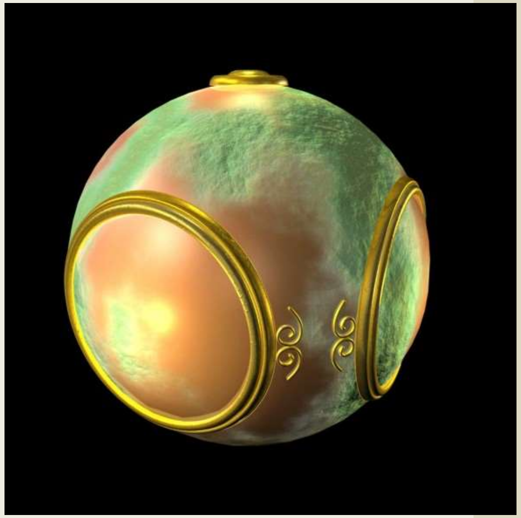
History: 1980-1990

- Realism comes to computer graphics
  - *Shading* –smoothness with considerations of lighting
  - *Environment mapping* – reflection from environments
  - *Bump mapping* – bumps and wrinkles on surface



Shading

Environment mapping



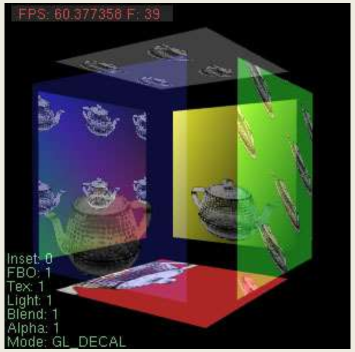
Bump mapping

History: 1980-1990

- Special purpose hardware
  - Silicon Graphics geometry engine
    - *VLSI implementation* of graphics pipeline
- Industry-based standards
  - PHIGS (Programmer's Hierarchical Interactive Graphics System)
    - API standard for rendering 3D computer graphics
    - Implemented as stand-alone systems (DEC PHIGS, IBM graPHIGS, SunPHIGS, ...)
  - Renderman – network distributed rendering system
- Networked graphics: X Window System

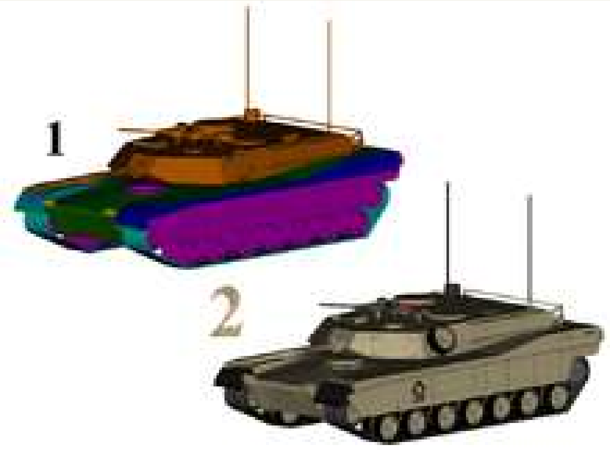
History: 1990-2000

- OpenGL (Open Graphics Library)
  - Cross-language, cross-platform API for 2D and 3D comput er graphics
  - Developed by Silicon Graphics
  - Widely used in computer-aided design (CAD), scientific vis ualization, flight simulation and video games
  - Managed by the non-profit technology consortium – Khro nos Group



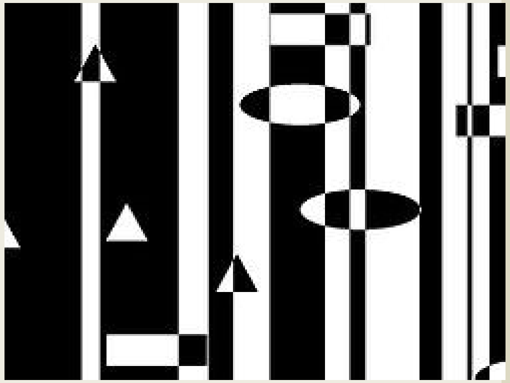
History: 1990-2000

- New hardware capabilities
  - Texture mapping
  - Blending of multiple sources (c.f. transparency)
  - Stencil buffers for visibility control



Texture mapping

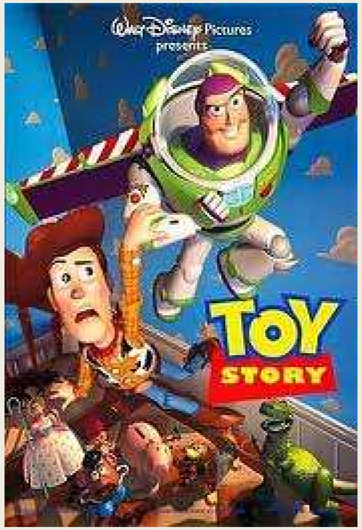
Alpha blending



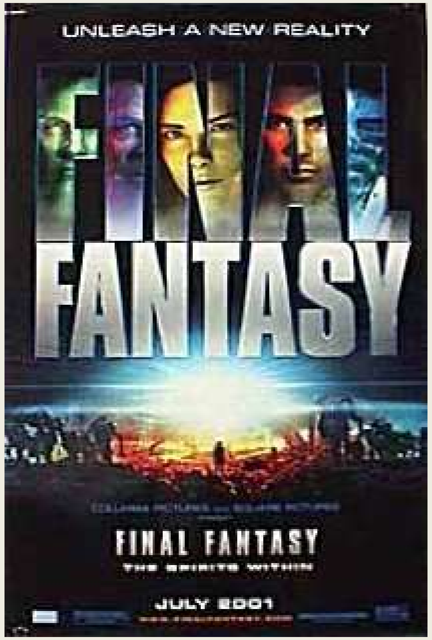
Stencil buffer

History: 1990-2000

- Completely computer-generated full-length movies are successfully launched
  - Computer-generated actors replaced human actors



Toy Story (1995)



Final Fantasy Movie (2001)

History: 2000-

- Photorealism
- Graphics cards for PCs dominate market
  - NVidia, ATI (now AMD)
- Game consoles and game players determine directio n of market
  - PlayStation 3, Xbox 360, Nintendo Wii
- Mobile devices get powerful graphics hardware
  - Galaxy S3, IPhone 5, ...
- Become popular in movie industry
  - Maya, Lightwave, ...

Bonus

- Screenshots of computer games

Wolfenstein 3D (1992)

Doom (1993)

Quake 1(1996)

Quake 2 (1997)

Quake 3 (1999)

Quake 4 (2005)

Bonus

- Screenshots of computer games

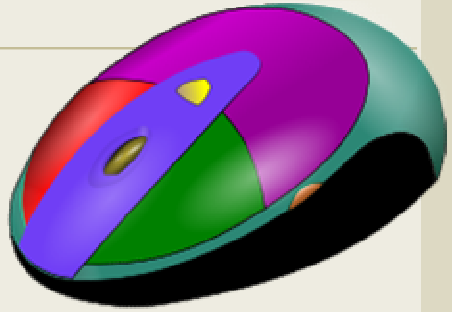
Crysis 2 (2009)

Mass Effect 2 (2010)

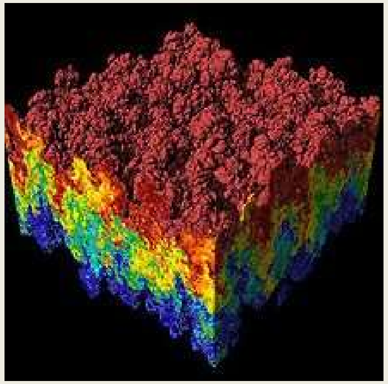
Battlefield 3(2011)

Elder scrolls V (2011)

Applications



- Computer-aided design (CAD)
  - Process of design and documentation
- Computer simulation
  - Simulates an abstract model of system
  - E.g. weather forecasting, typhoon



- Information visualization
  - A way to visualize complex behaviors
- Digital art
  - Artistic works using digital technology

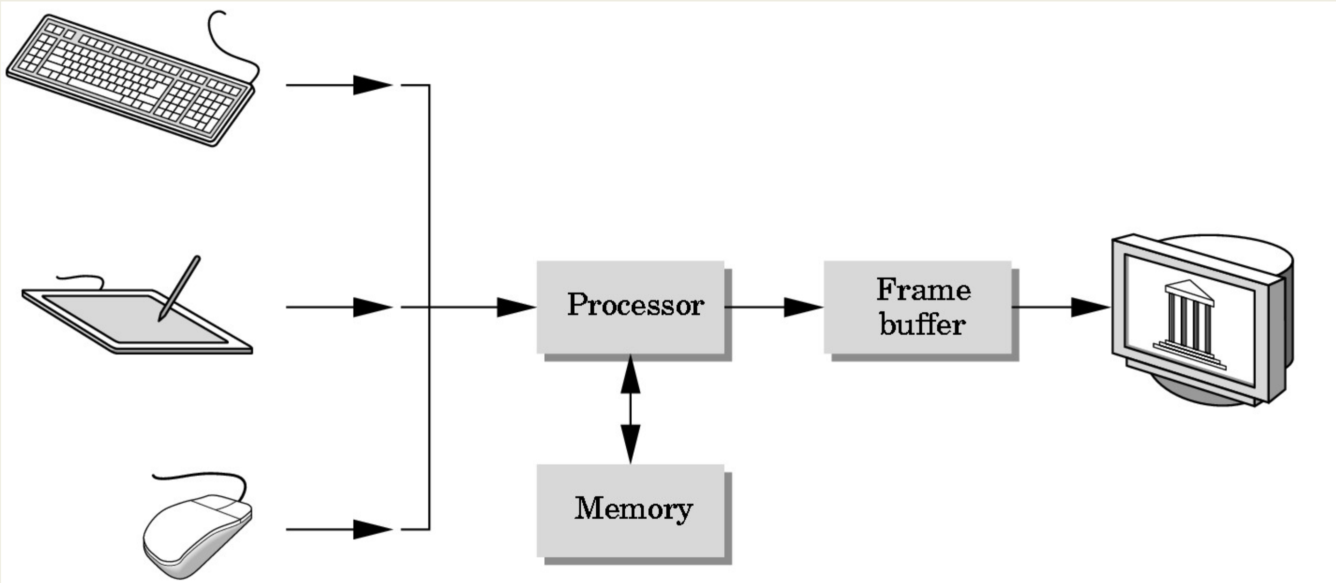
Applications

- Video games
  - Improved reality through stunning 2D / 3D graphics
- Virtual reality
  - Computer-simulated environments
  - Simulate physical presence in places
- Augmented reality
  - Real-world environment
  - Elements are augmented by computer-generated sensory data

Basic Graphics System

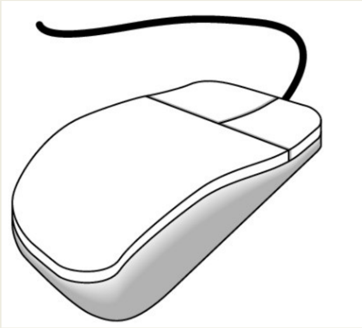
Graphics system

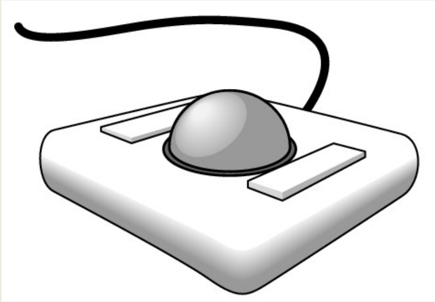
- Basic graphics system
  - Input devices, processor, memory, frame buffer, output de vices



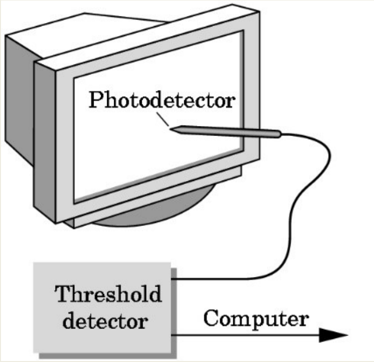
Graphics system

- Input devices
  - Mouse & trackball
    - Converts *relative* movement into two orthogonal directions

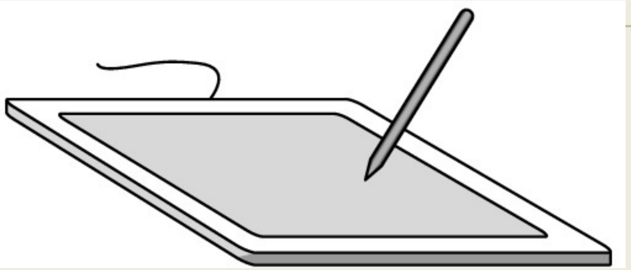




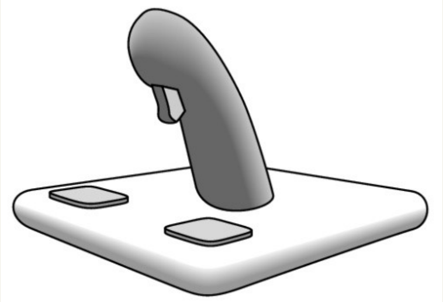
  - Light pen
    - Used in Sutherland's Sketchpad
    - Contains light-sensing device
    - Converts lights into *absolute* position



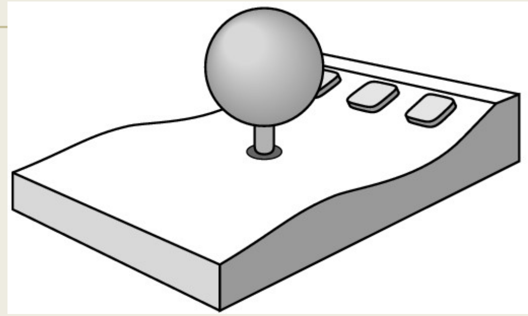
Graphics system



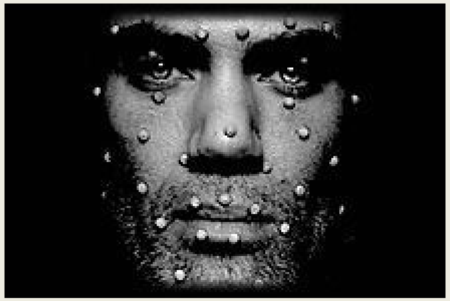
- Input devices
  - Data tablet
    - Provides absolute position as light pen
    - Sometimes detects how strong light pen is pressed
  - Joystick
    - Motion of stick is converted into two orthogonal directions
    - *Variable-sensitivity device* – degree of input strength
    - Well suited for flight simulators and game controllers



Graphics system

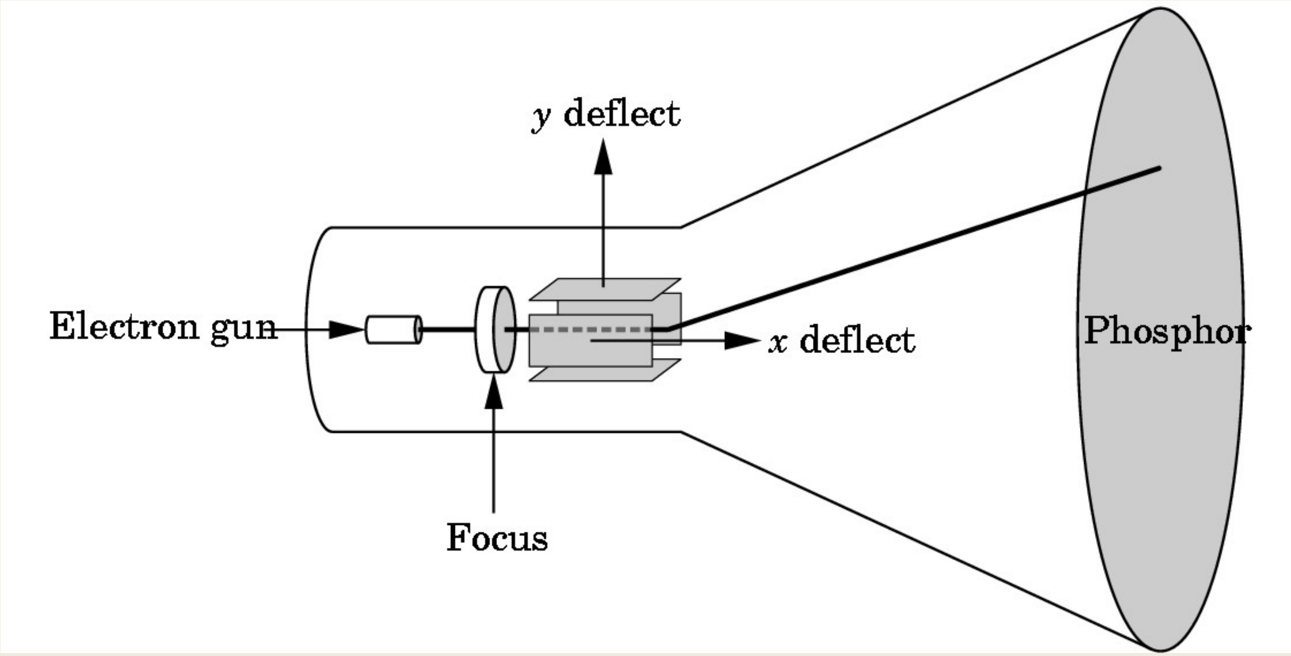


- Input devices
  - Space ball
    - 3-dimensional input device
    - Pressure sensors in the ball measures the forces by the user
    - 6 degrees of freedom – up/down, left/right, front/back, 3 twists
  - Motion capture device
    - Array of cameras capture lights from small spherical dots



Graphics system

- Output devices
  - CRT (cathode ray tube) (1897~)
    - Vacuum tube containing an electron gun and a fluorescent screen



Graphics system

- Output devices
  - LCD (liquid crystal display)
    - Flat panel display uses the liquid crystals
    - Energy efficient than CRTs
    - Needs backlight
  - Plasma display
    - Flat panel display uses electronically charged ionized gases
    - Brighter, wider color gamut, easier to be produced in large size th an LCDs
    - Needs more energy than LCDs
    - Image burn-in – image without movement can burn your TV
    - Large pixel pitch – hard to develop high resolution screen

Graphics system

- Output devices
  - LED (light-emitting diode) display
    - Flat panel display uses light-emitting diodes
    - LED display is similar to LCD (e.g. backlight)
    - Reduces gap between plasma display
  - OLED (organic light-emitting diode) display
    - Flat panel display uses organic LED
    - Works without backlight
    - Lighter weight, better power efficiency, brighter, wider viewing a ngle than LCDs
    - Possibility of flexible OLED
    - Expensive, organic materials dies – blue OLED: 14,000 hours

Graphics system

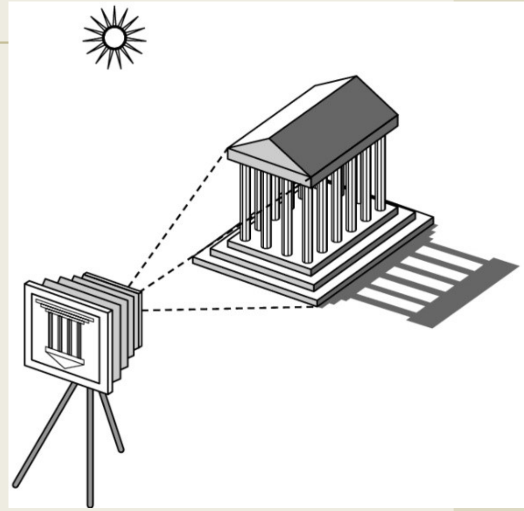
- Frame buffer
  - A picture is produced as an array (raster) of picture eleme nts (pixels)
  - Pixels are stored in a part of memory called *frame buffer*
  - Two specifications for frame buffer
    - Resolution: number of pixels – (width x height)
    - Depth (or precision) - bits per color component
      - 8 bits / color component = 24 bits / pixel – *true-color system*
      - High dynamic range (HDR) applications more than 24-bits
  - Sometimes, frame buffer means multiple buffers
    - Three color buffers for R, G, B components + 3D depth (z-axis)

Graphics system

- Processor
  - Earlier system may have only one processor
    - CPU (central processing unit) does both (normal / graphical)
  - More than two processors are usual now
    - CPU does normal processing
    - GPU (special purpose graphics processing units) does graphical processing
      - Most graphics primitives are implemented via hard-wired logics
      - Graphic card embeds frame buffer and GPU directly accesses it
      - GPU has multiple cores inside it - GeForce GTX590 (1024 cores)

Image Formation

Image formation



- Elements of image formation
  - 1) Objects, 2) viewer and 3) light sources
  - They are *independent* each other

1) Object

  - Exists in space independent of viewer
  - In most graphics systems, object is defined (approximatel y) as a set of locations in space called as *vertices*
    - E.g. Line can be specified by two vertices
    - E.g. Polygon can be specified by an ordered list of vertices
    - E.g. Sphere can be specified by two vertices giving its center and a ny point on its circumference

Image formation

2) Viewer

- Someone who views objects
- Person, camera, digitizer, microscope, ...
- Image (2D) is formed
  - On the back of the eye in human visual system
  - In the film plane in camera
- Obtained image is different according to viewer's position

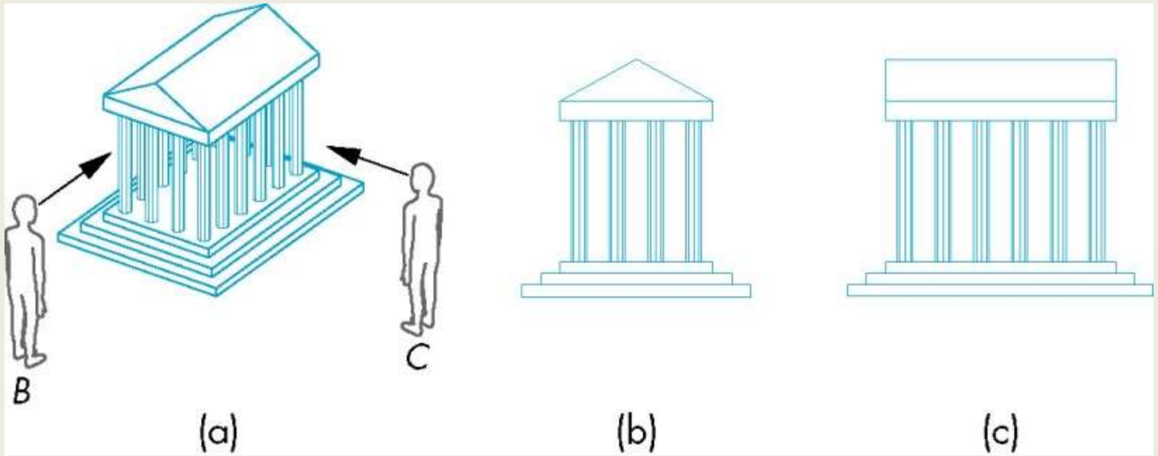
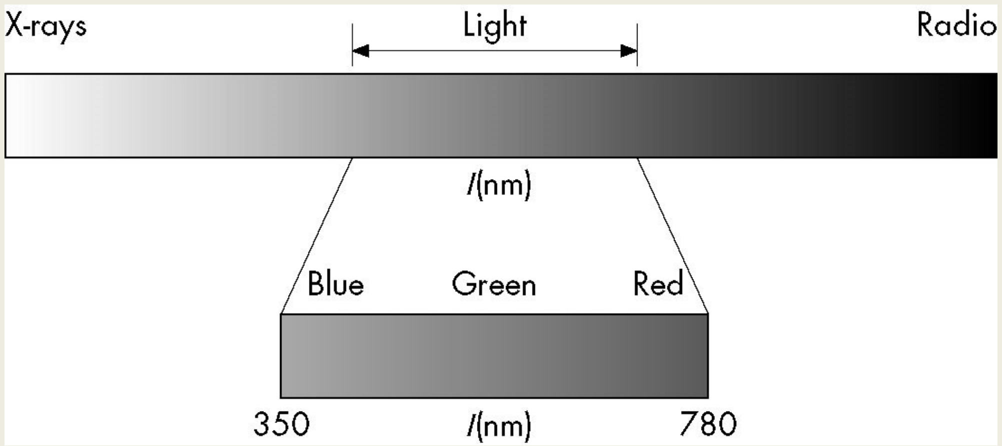


Image formation

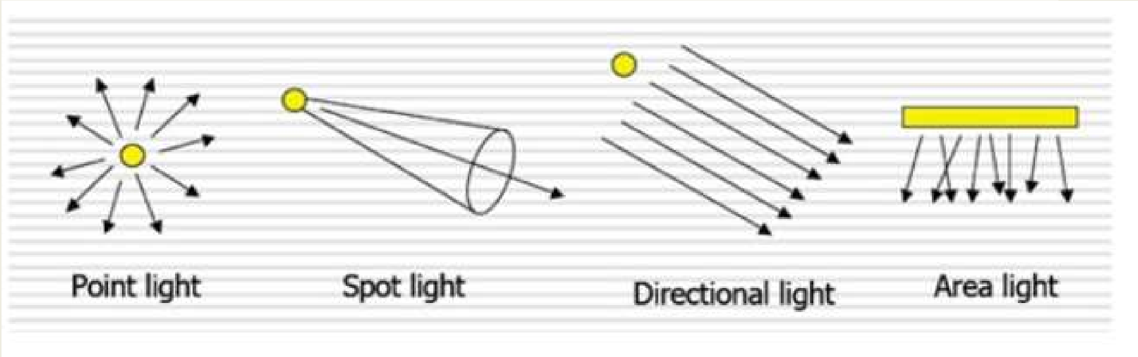
3) Light

- Electromagnetic radiation
- Characterized by its wavelength
  - Wavelengths of visible spectrum (called as light): are in the range of 350 to 780 nm



Lights and Materials

- Types of lights
  - Point sources vs distributed sources
  - Spot lights
  - Near and far sources
  - Color properties



- Material properties
  - Absorption: color properties
  - Scattering
    - Diffuse
    - Specular

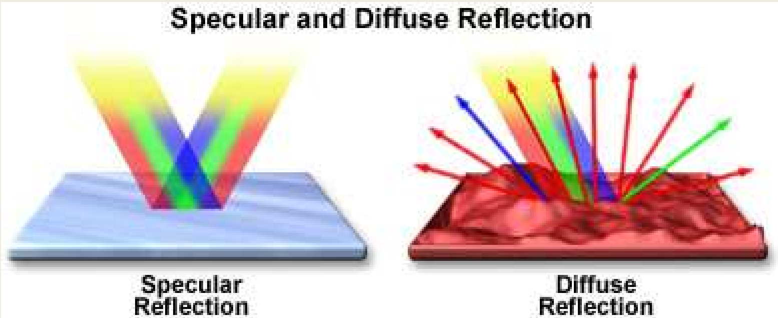
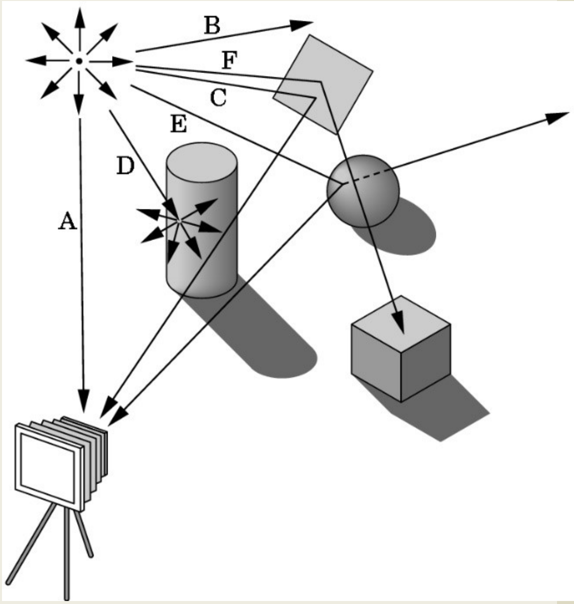


Figure 1

Image Source
https://slideplayer.com/slide/4635649/
http://olympus.magnet.fsu.edu/primer/java/reflection/specular/index.html

Image formation

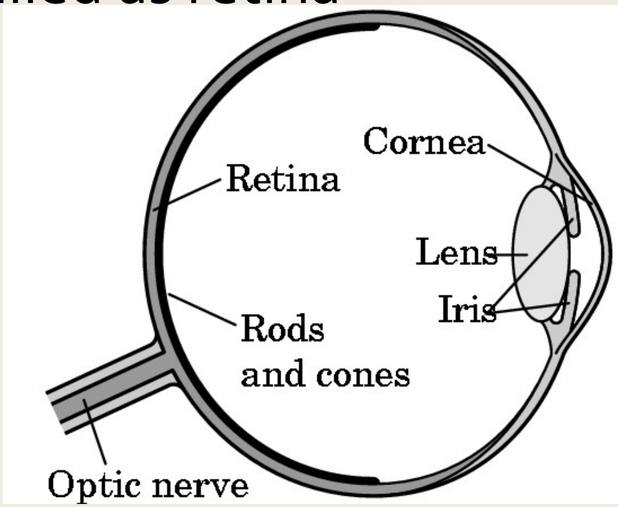
- Ray tracing (or photon mapping)
  - A *ray* is a semi-infinite line that emanates from a point an d travels to infinity in a particular direction
  - Rays of light emanating in all directions from point source
  - *Collect all rays reaching film plane*
    1. Directly come from the source
    2. Indirectly come through the objects (reflection, absorption, …)
  - Good approximation of the world
  - Not good for real-time processing



Imaging System

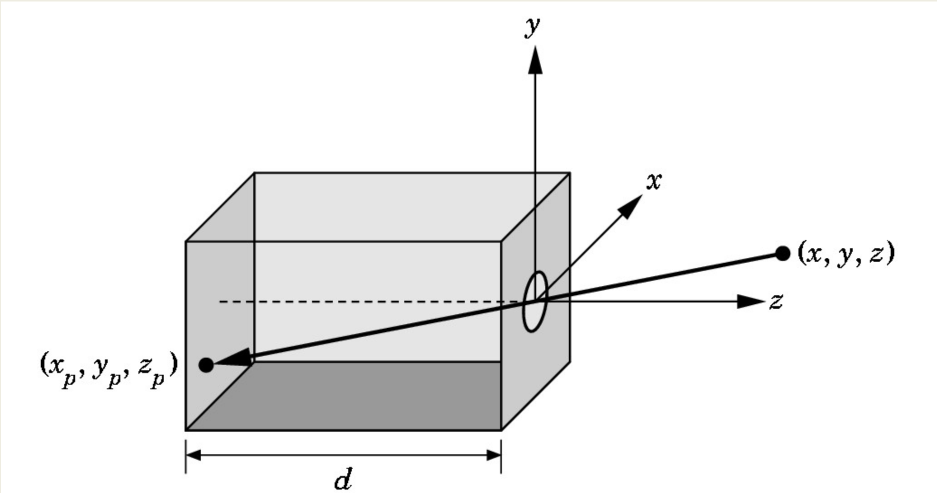
Imaging system

- Human visual system
  - Light enters the eye through lens and cornea
  - Iris controls the amount of light entering the eye
  - Image is formed on 2D structure called as retina
  - Two sensors: *rods* and *cones*
    - Rods: single type for brightness
    - Cones: three types for color
      - Reason why we use three primary colors (e.g. RGB)



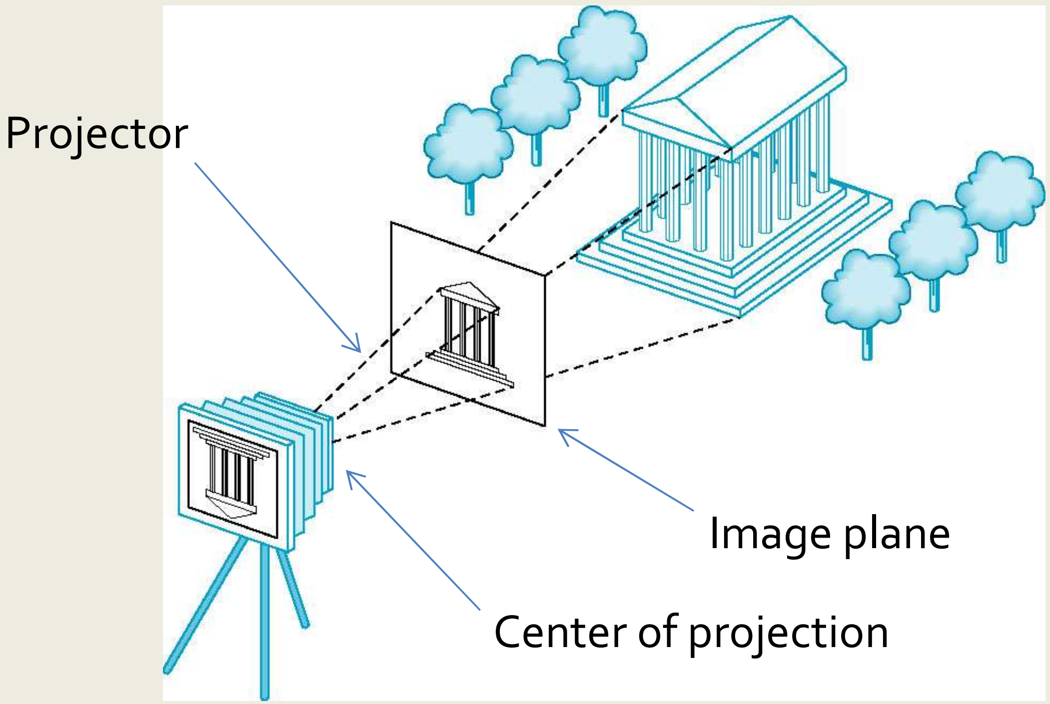
Imaging system

- Pinhole camera
  - Box with a small hole in the center of one side
  - Film is placed on the side opposite the pinhole
  - Relation between positions
    - $x_p = -\frac{xd}{z},\ y_p = -\frac{yd}{z},\ z_p = -d$



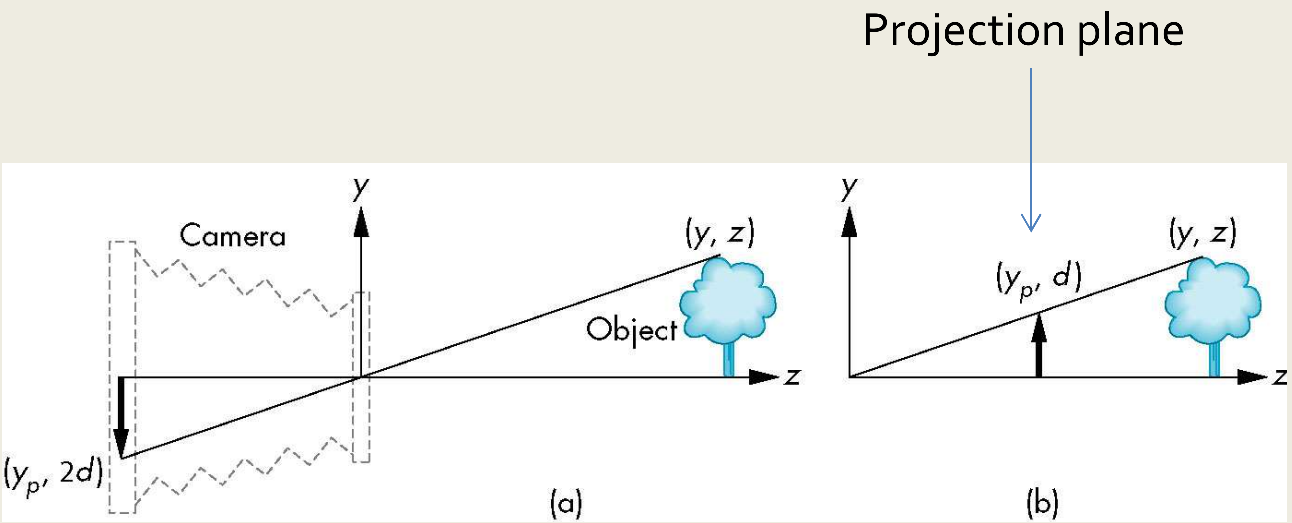
Synthetic camera model

- Paradigm to create computer generated image
  - Similar to forming an image using an optical system



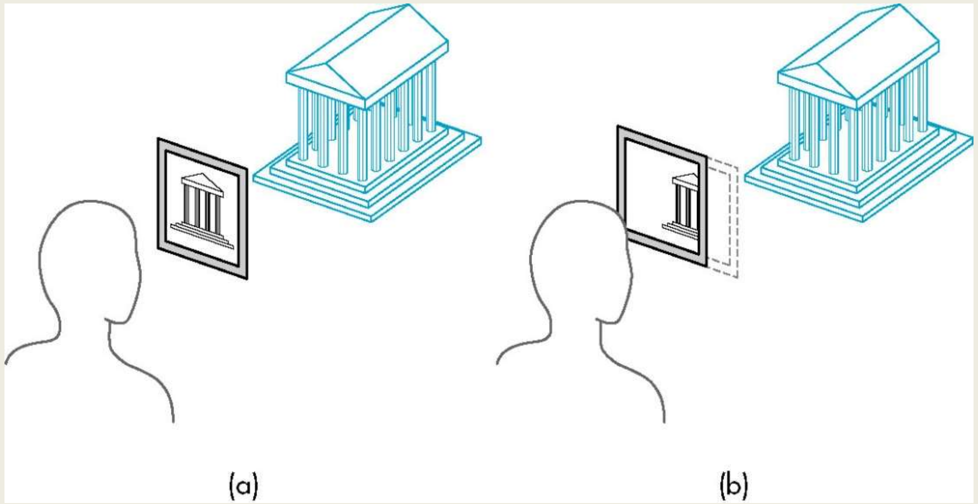
Synthetic camera model

- Define virtual image plane in front of the lens
  - Called as projection plane
  - No need to consider "flipped" property of objects



Synthetic camera model

- Clipping window
  - Not all objects can be imaged onto film plane
  - Defines viewing angle located in projection plane
  - We can determine which objects will appear in final image



Additive and Subtractive Color

- Additive color
  - Form a color by adding amounts of three primaries
    - CRTs, projection systems, positive film
  - Primaries are Red (R), Green (G), Blue (B)
- Subtractive color
  - Form a color by filtering white light with cyan (C), Magenta (M), and Yellow (Y) filters
    - Light-material interactions
    - Printing
    - Negative film

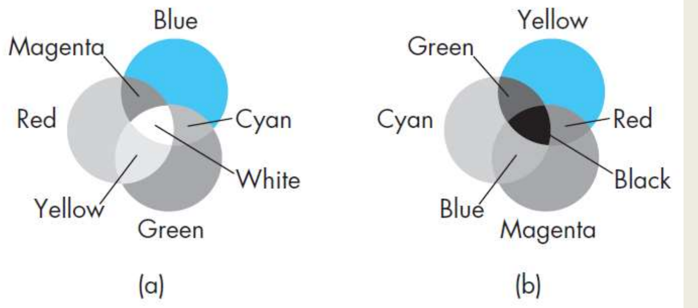


FIGURE 2.25 Color formation. (a) Additive color. (b) Subtractive color.

Models and Architectures

Objectives

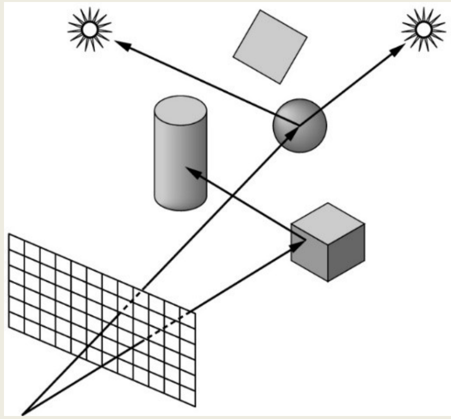
- Learn the basic design of a graphics system
- Introduce pipeline architecture
- Examine software components for an interactive graphics system

Image Formation Revisited

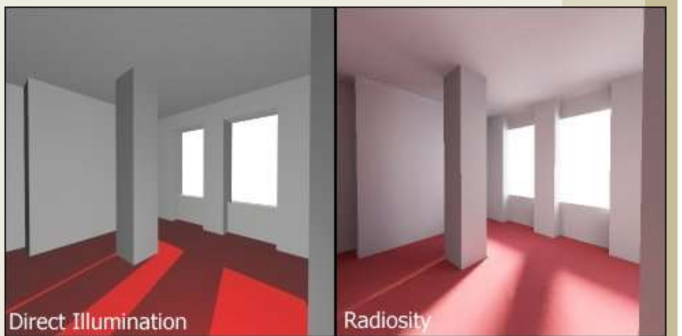
- Can we mimic the synthetic camera model to design graphics hardware software?
- Application Programmer Interface (API)
  - Need only specify
    - Objects
    - Materials
    - Viewer
    - Lights
- But how is the API implemented?

Physical Approaches

- **Ray tracing**: follow rays of light from center of projection until they either are absorbed by objects or go off to infinity
  - Slow, Great specular, approx. diffuse
    - View dependent
  - Can handle global effects
    - Multiple reflections
    - Translucent objects
  - Must have whole data base available at all times

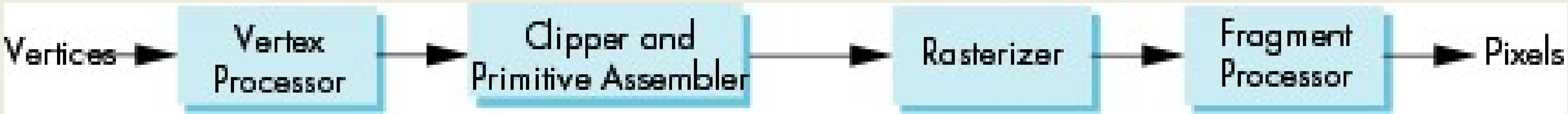


- **Radiosity**: Energy based approach
  - Very slow, Great diffuse, specular ignored
    - View independent, mostly-enclosed volumes
- Advanced hybrids
  - Combine them



Graphics architectures

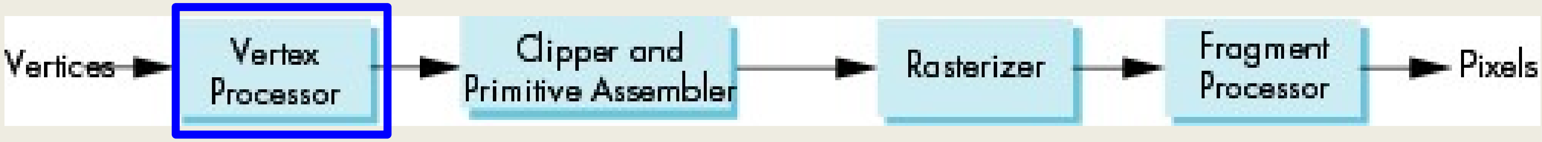
- Most graphics architecture uses *pipelining*
  - Similar to an assembly line in a car plant
    - Cons: significant delay between starts and ends
    - Pros: throughput (number of produced cars in a given time) is much higher than if single team builds each car
- Graphics pipeline
  - Vertex processing, clipping and primitive assembler, rasterizer and fragment processing
  - All steps can be implemented in hardware



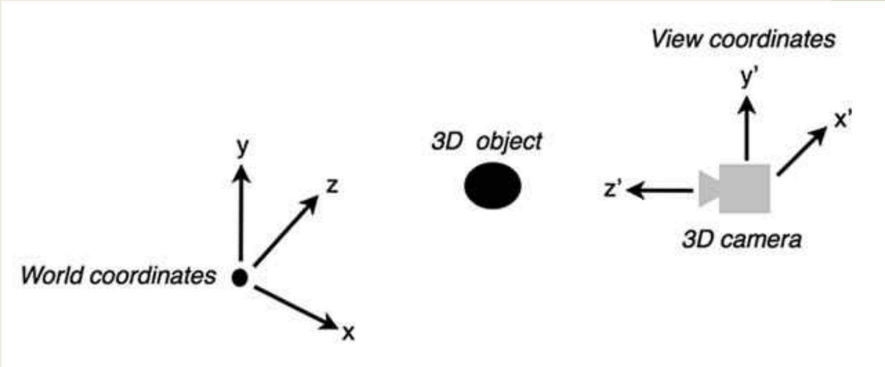
application program

display

Vertex Processing

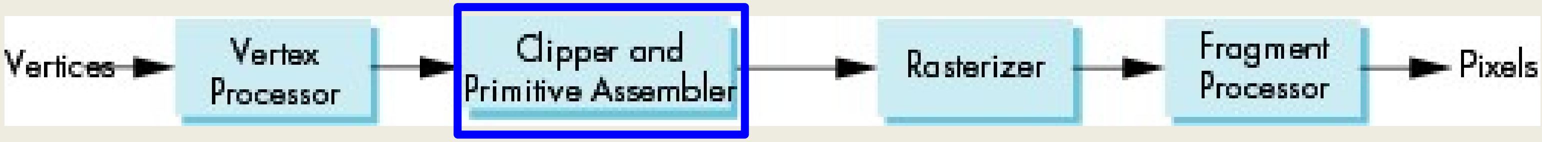


- Much of the work in the pipeline is in converting object representations from one coordinate system to another
  - Object coordinates
  - Camera (eye) coordinates
  - Screen coordinates

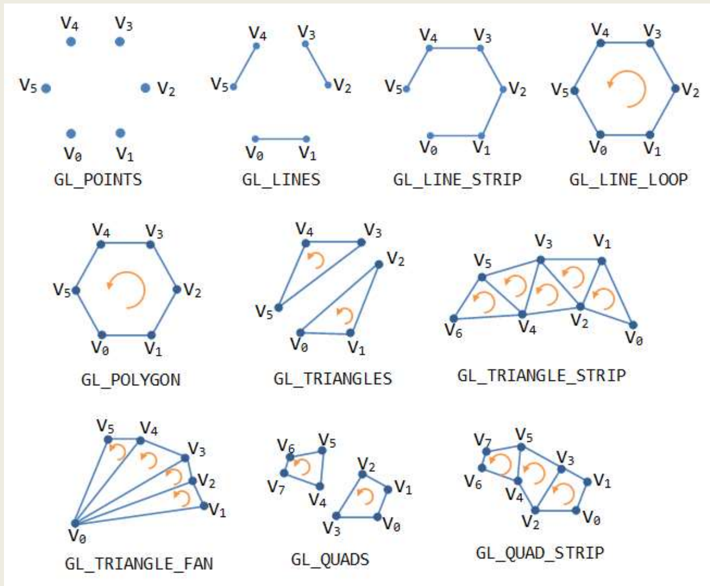


- Every change of coordinates is equivalent to a matrix transformation
- Vertex processor also computes vertex colors

Primitive Assembly



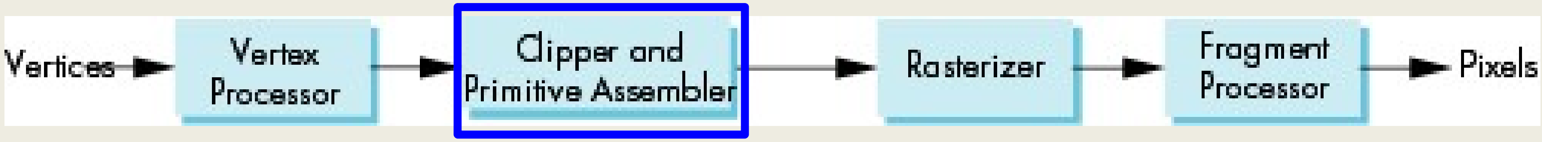
- Mainly due to the computational efficiency
- Vertices must be collected into geometric objects before clipping and rasterization can take place
  - Line segments
  - Polygons
  - Curves and surfaces



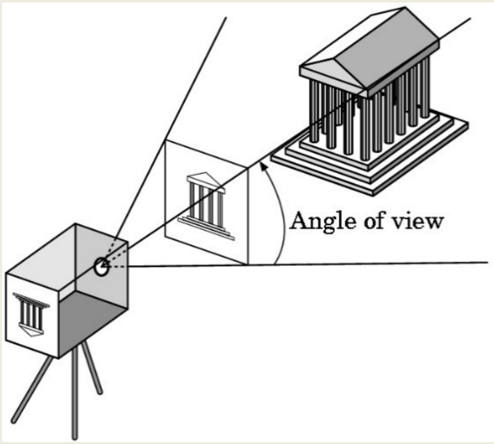
OpenGL Primitives

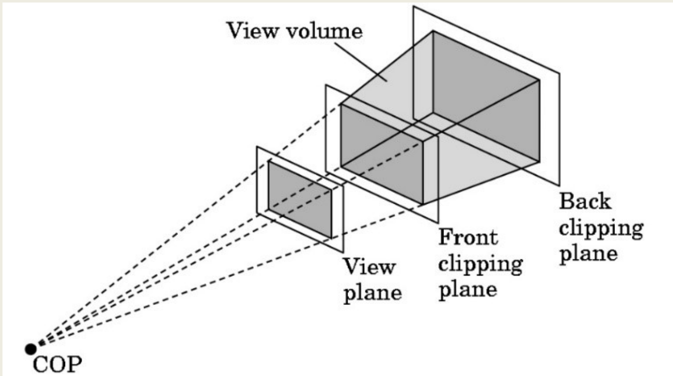
https://www.ntu.edu.sg/home/ehchua/programming/opengl/CG_BasicsTheory.html

Clipping

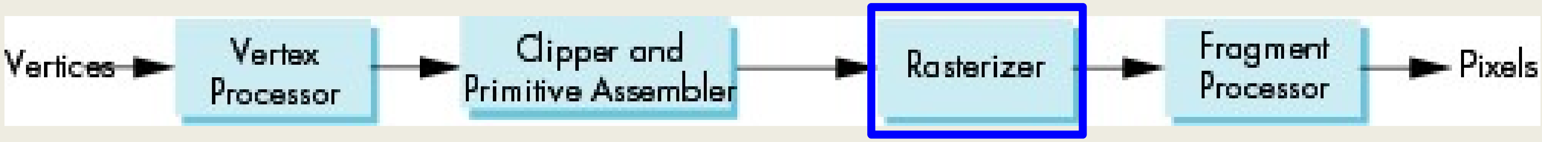


- Just as a real camera cannot "see" the whole world, the virtual camera can only see part of the world or object space
  - Objects that are not within this volume are said to be *clipped* out of the scene

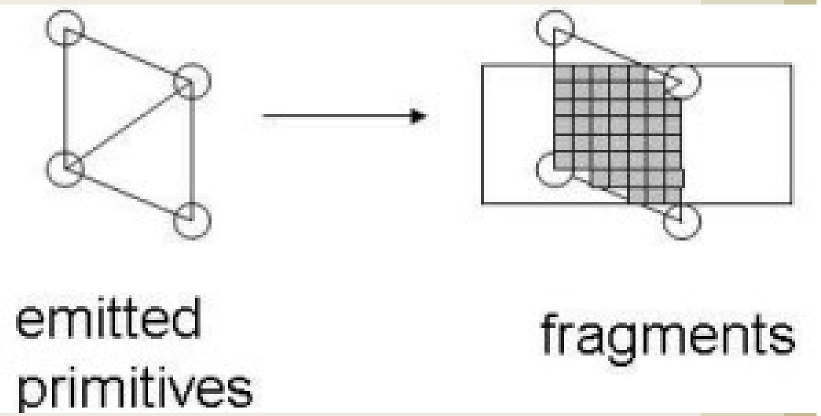




Rasterization

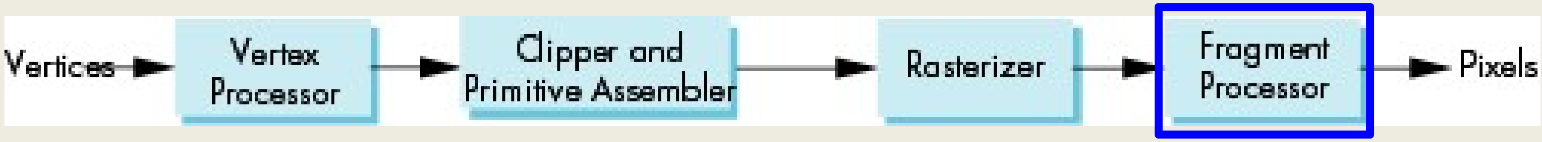


- If an object is not clipped out, the appropriate pixels in the frame buffer must be assigned colors
- Rasterizer produces a set of fragments for each object
- Fragments are "potential pixels"
  - Have a location in frame buffer, color, and depth attributes

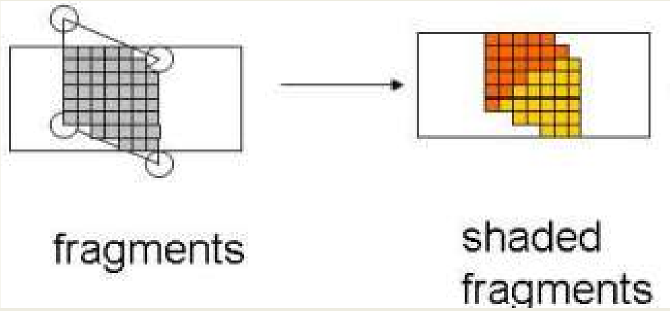


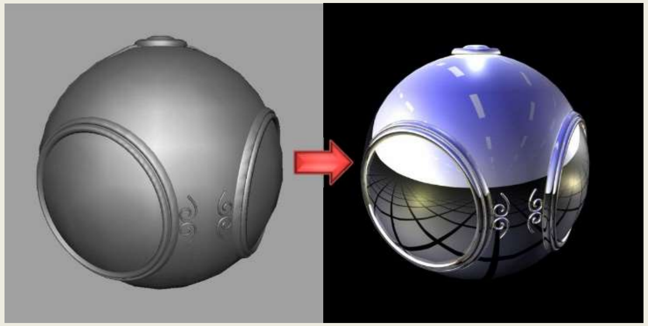
- Vertex attributes are interpolated over objects by the rasterizer

Fragment Processing



- Fragments are processed to determine the color of the corresponding pixel in the frame buffer
- Colors can be determined by texture mapping or interpolation of vertex colors
- Fragments may be blocked by other fragments closer to the camera
  - Hidden-surface removal





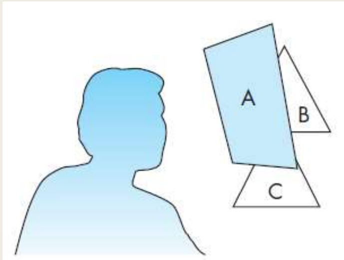
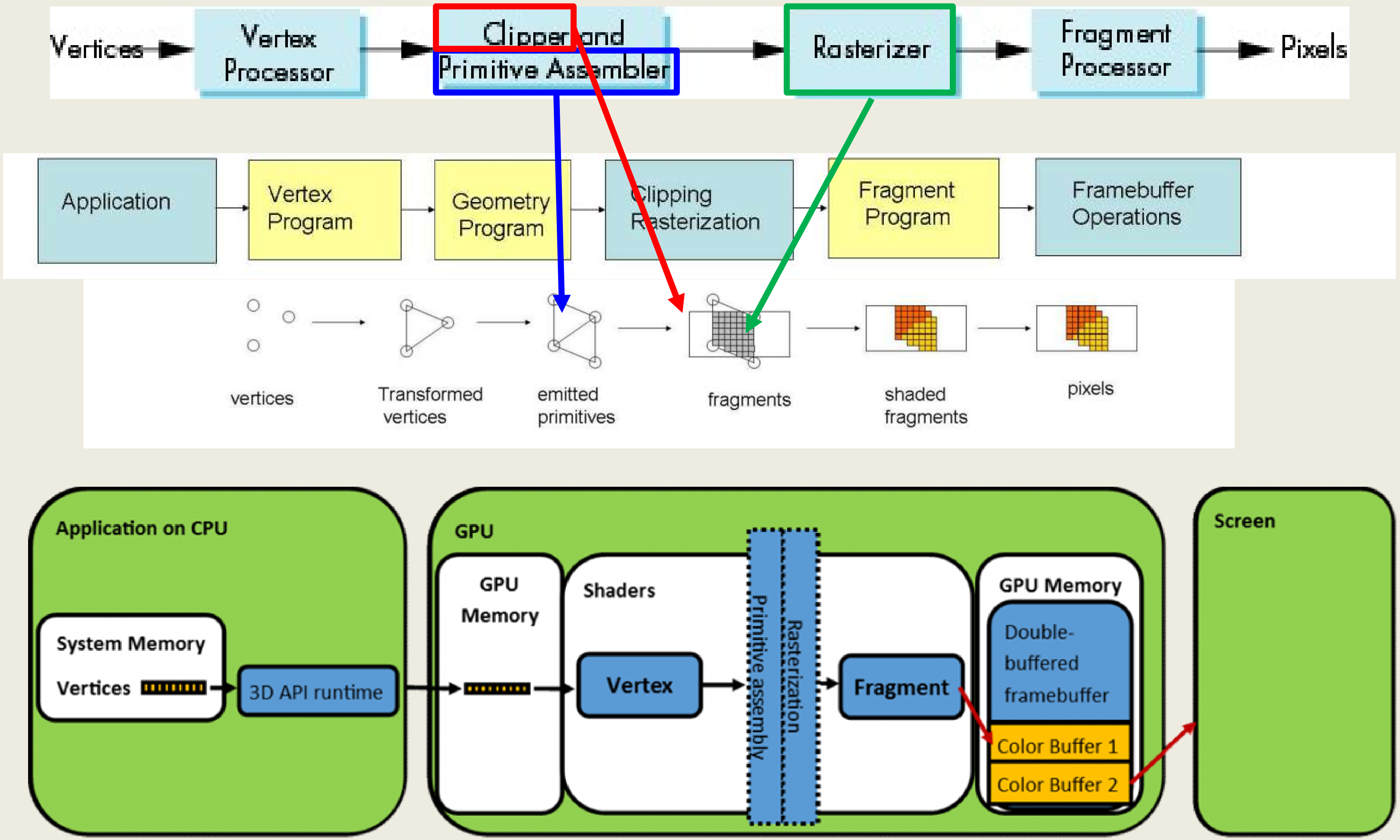


FIGURE 2.43 The hidden-surface problem.



- Vetter et al.," Non-rigid multi-modal registration on the GPU. In Medical Imaging 2007: Image Processing," *International Society for Optics and Photonics*, vol. 6512, pp. 651228, Mar. 2007.
- https://www.gamedev.net/articles/programming/graphics/introduction-to-the-graphics-pipeline-r3344/