GRAPHIC SYSTEMS AND MODELS

Computer Graphics Applications

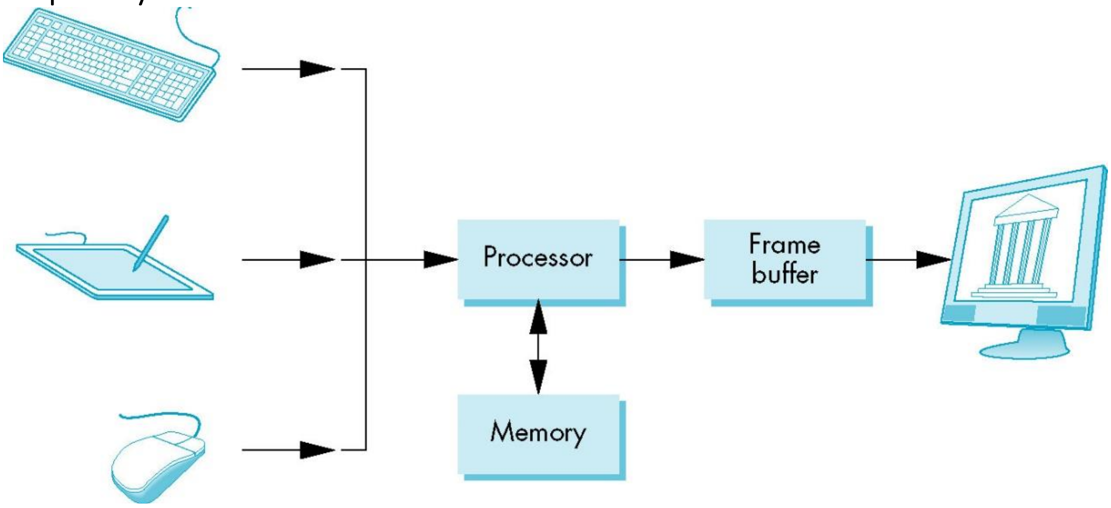
Information Display - Ways in which images are produced and colors. Example, Medical Images.

Design - CAD

Simulation and Animation - Generate complex real-time images in real time. Example, flight simulators. And VR!

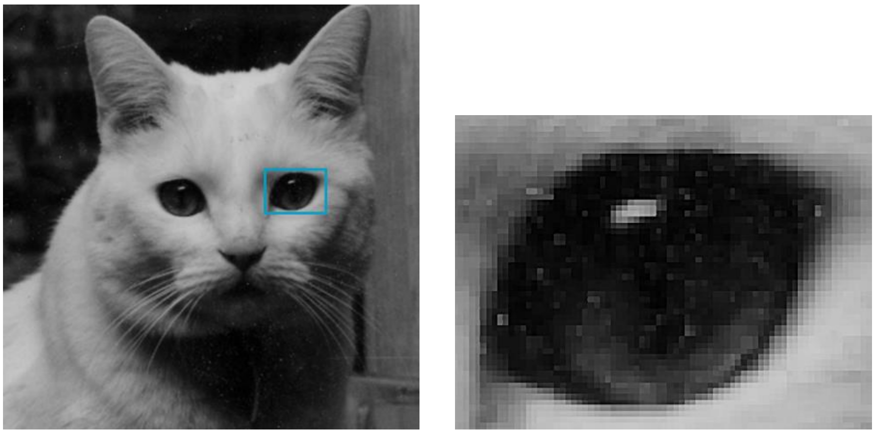
User Interface - The dominant interaction paradigm on computers is visual, windows, icons, pointing devices, etc.

Graphics System



Pixels and the Frame Buffer

Currently graphic systems are based on raster images (image produced as an array - the raster - of elements, the pixels.)



The pixels are stored in the frame buffer (special type of memory that allows for quick display of images).
Frame Buffer is characterized by: resolution (1024X768), depth (RGB).

Rasterization: The conversion of geometric entities to colored pixels and positions in the frame buffer.

Output and input peripherals
Monitors (+ refreshment) and mice (pointing devices)

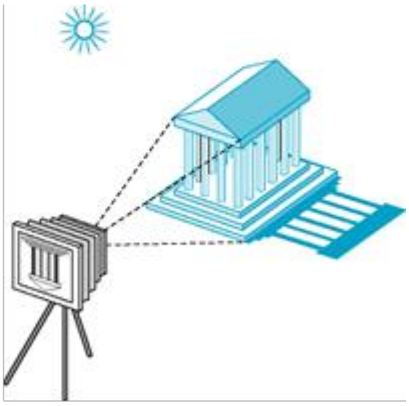
Image Formation

Modeling

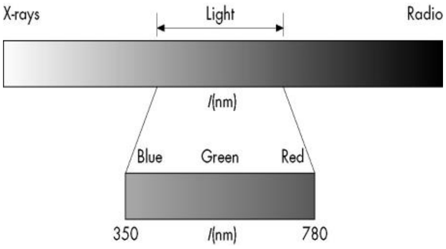
- Basic primitive shapes (cubes, spheres, cones, cylinders, etc.)
- Composite Objects + Transformations (translation, rotation and scale change)
- Parametric Objects
- Text

Lighting

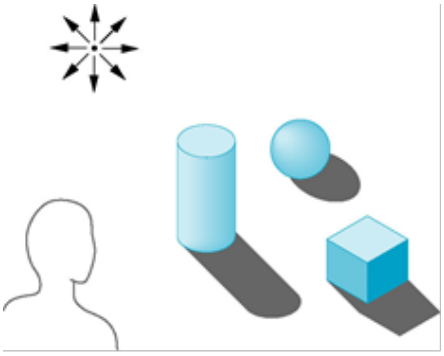
Illuminate the scene:



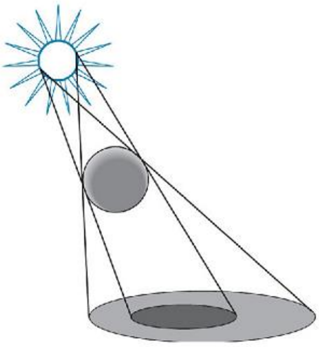
Spectrum:



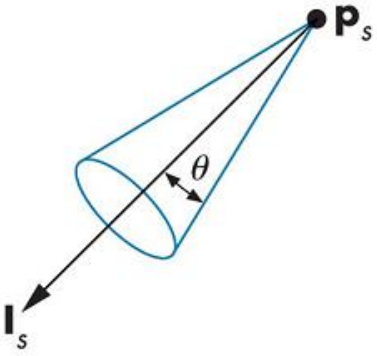
Point Light Source:

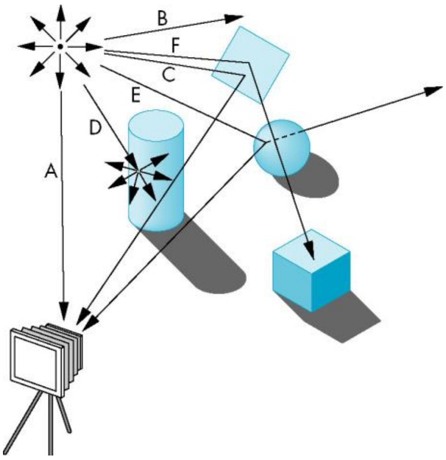


Non-point Light Source:



SpotLight:



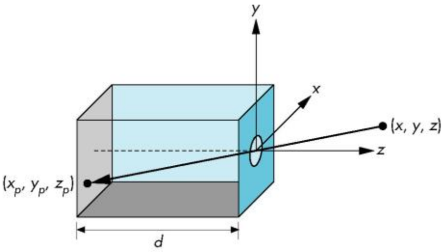


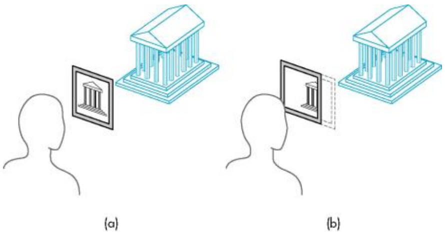
Radiosity (conservation of energy and dispersion of light energy)

Image Calculation (by approximation of physical phenomena):
Ray-tracing (Interactions between light rays).

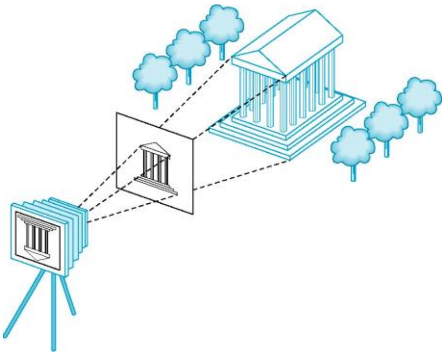
Camera

Pinhole camera. Positioned at the origin of the axes, infinite focus, different from wearing lenses.
Computer graphics systems produce images in which all objects are focused.





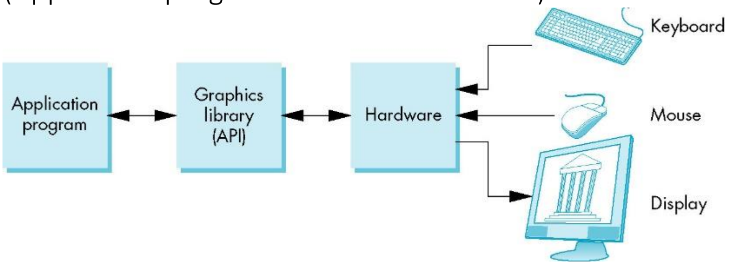
Synthetic camera (viewing plane in a certain position)



Synthetic camera (Need for clipping - clipping window)

Programming Interfaces

(Application programmer's interface - API)

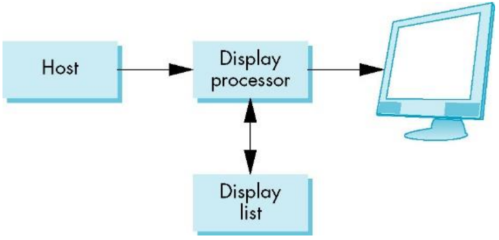


Pen-Plotter Model:
LOGO and PostScript (2D)

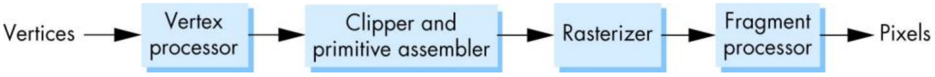
3D APIs:
OpenGL, DirectX, Java3D.
Based on the synthetic chamber model.
It is necessary to define: Objects, Views, Light Sources, Material properties.

Graphic Architectures

Drawing processors



Graphic pipeline



Sequence of operations to create an image from your geometric model.

Vertex processing: transformations and projections, matrix calculation and colors.

Cutout and primitives: transformation into graphic primitives (straight lines and polygons) and definition of the cutout volume.

Rasterization (discretization): transformation into fragments (grid, color, location)

Fragment processing: update the pixels in the framebuffer, i.e., write a color in a position in the framebuffer. There are a number of effects that must be addressed here (XOR, hidden fragments, textures, etc.)

Programmable graphic pipeline
Makes it possible to perform operations in real time.