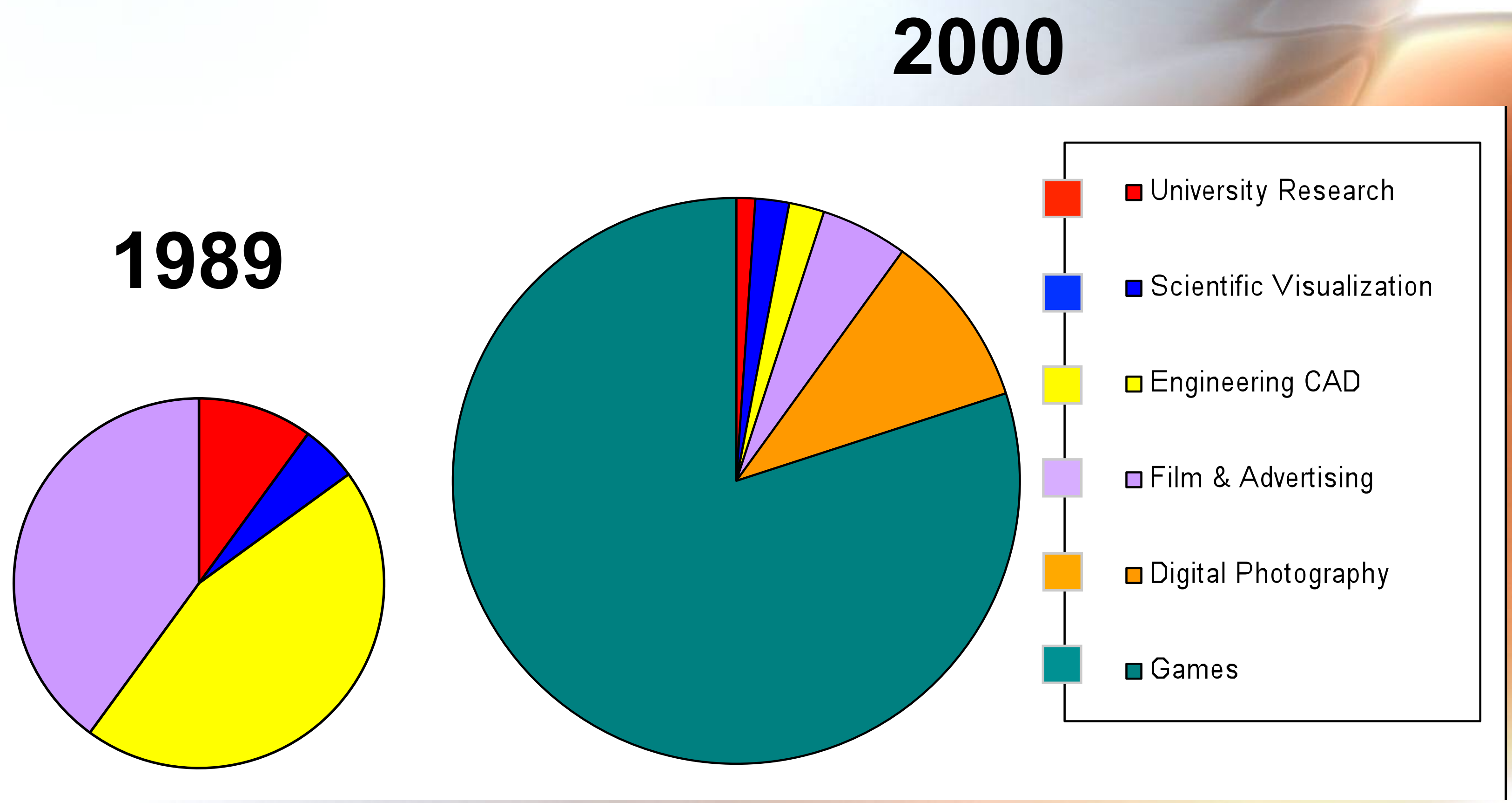
Computer Graphics

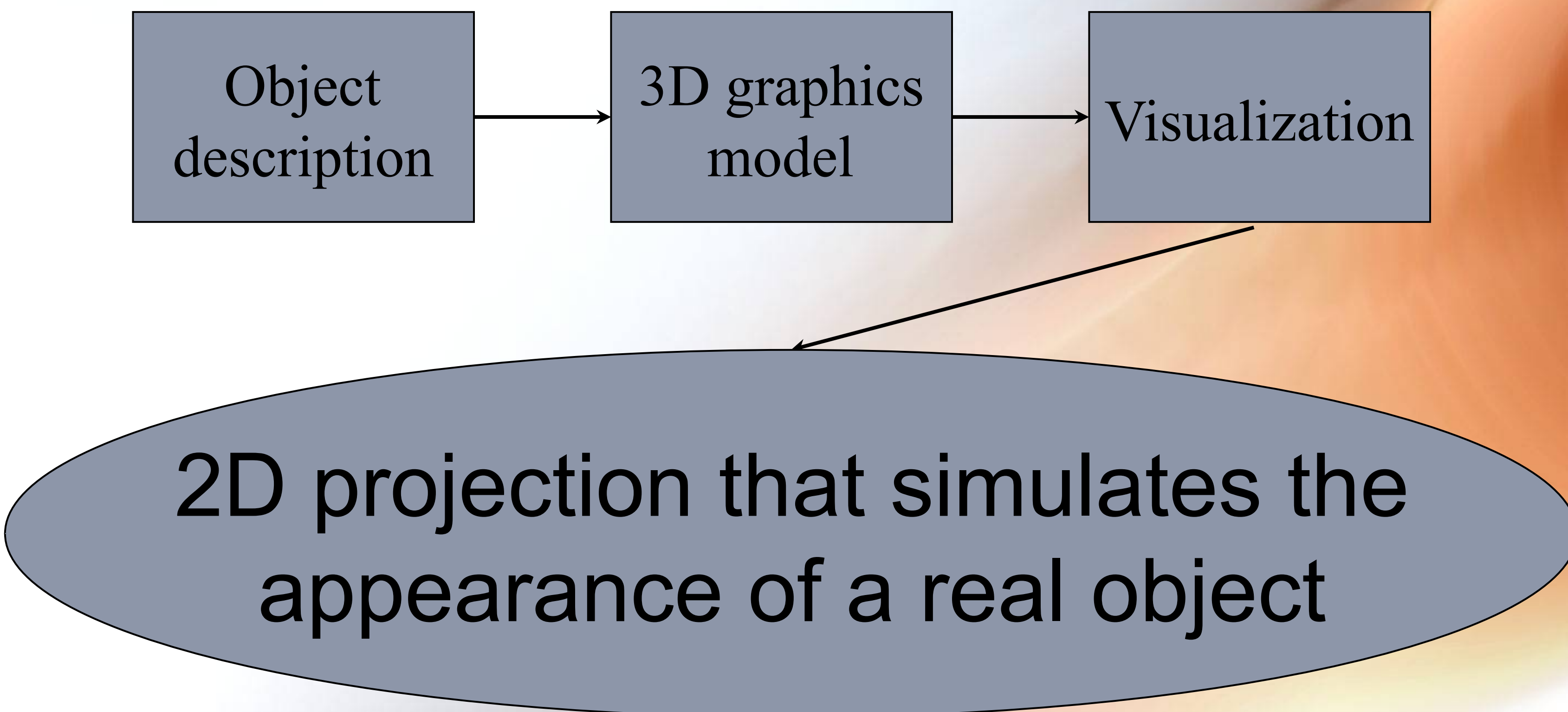
3D graphics, raster and colors

CS312 – Fall 2010

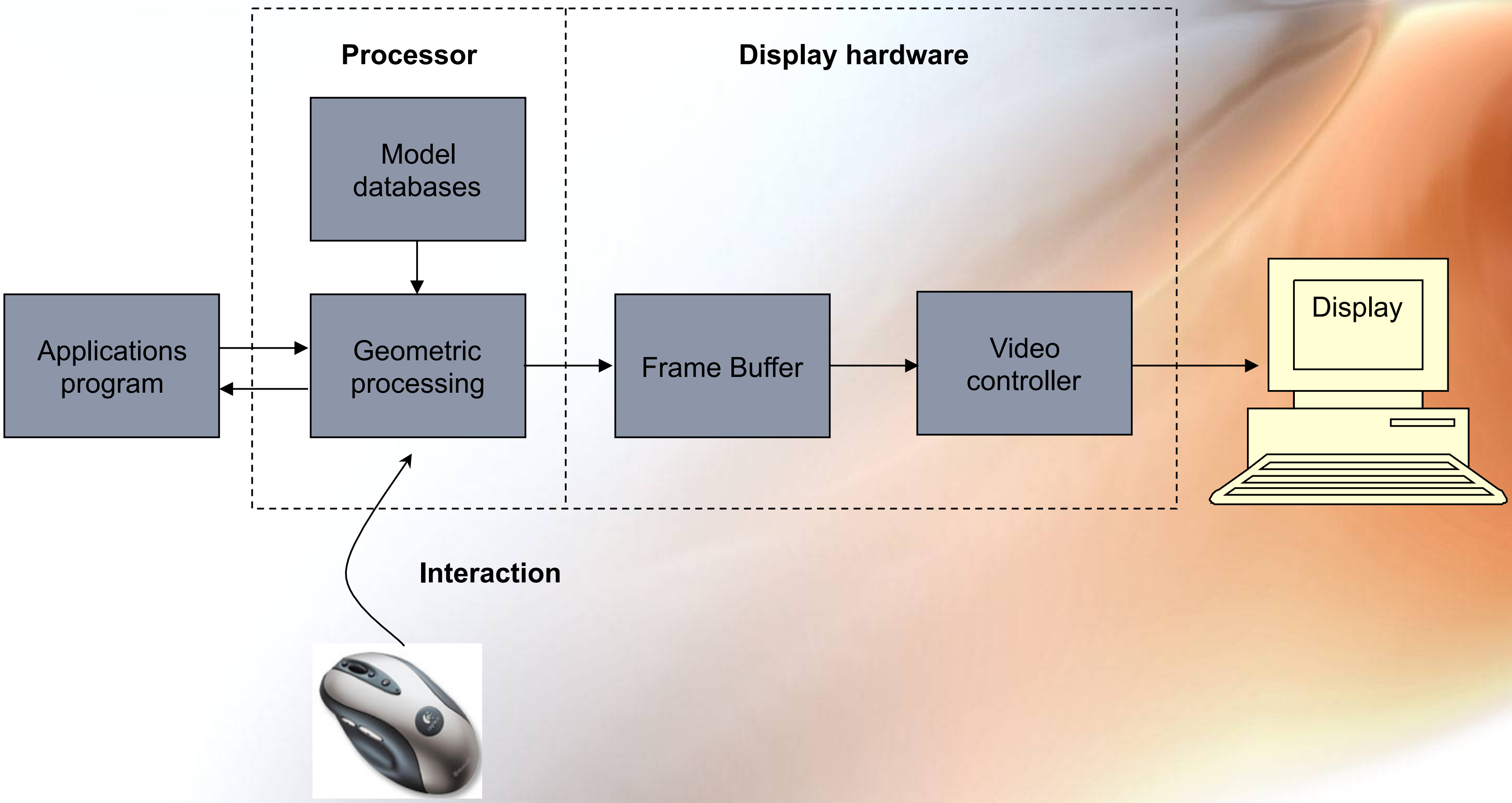
Shift in CG Application Markets 1989-2000



3D Graphics

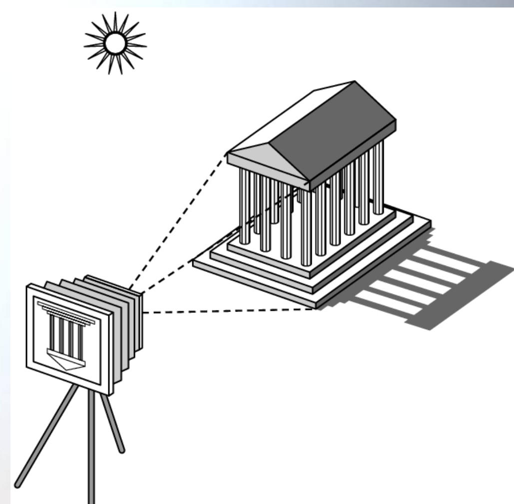

Object description
3D graphics model
Visualization
2D projection that simulates the appearance of a real object

Graphics System



Elements of Image Formation

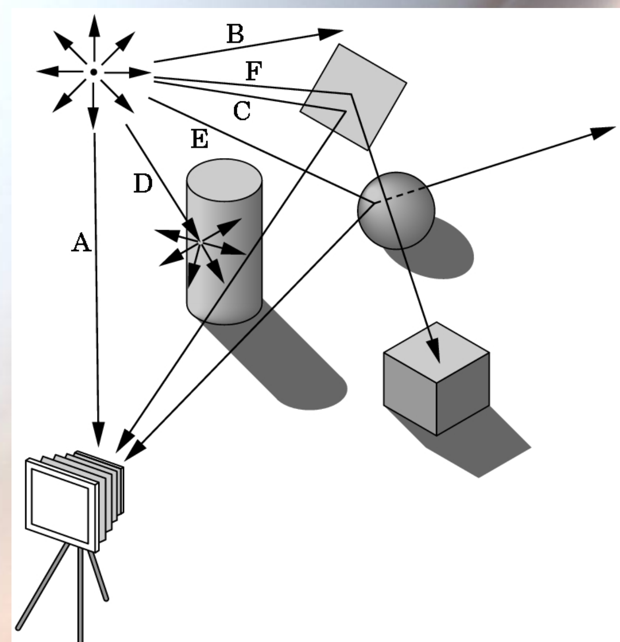
- Objects
- Viewer
- Light source(s)



- Attributes that govern how light interacts with the materials in the scene
- Note the independence of the objects, the viewer, and the light source(s)

Ray Tracing

One way to form an image is to follow rays of light from a point source and determine which rays enter the lens of the camera.

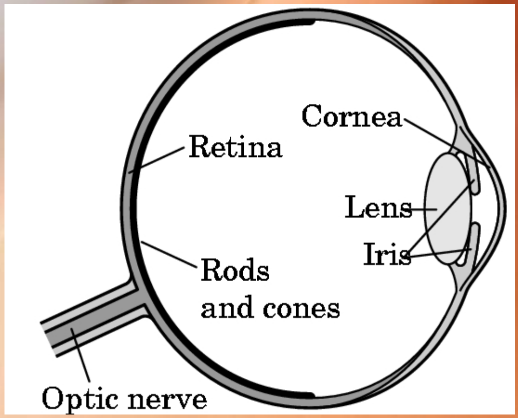


Luminance and Color Images

- Luminance Image
  - Monochromatic
  - Values are gray levels
  - Analogous to working with black and white film or television
- Color Image
  - Has perceptional attributes of hue, saturation, and lightness
  - Do we have to match every frequency in visible spectrum?

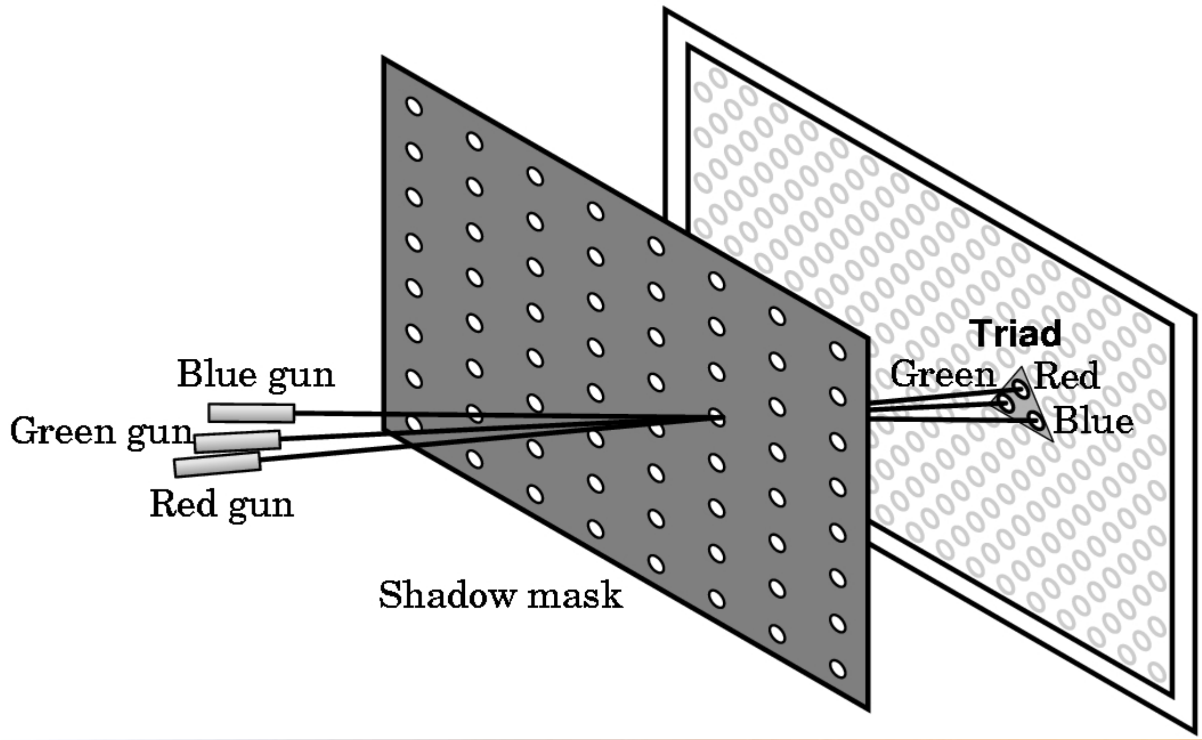
Three-Color Theory

- **Human visual system has two types of sensors**
  - **Rods: monochromatic, night vision**
  - **Cones**
    - **Color sensitive**
    - **Three types of cones**
    - **Only three values (the *tristimulus* values) are sent to the brain**



- **Need only match these three values**
  - **Need only three *primary* colors**

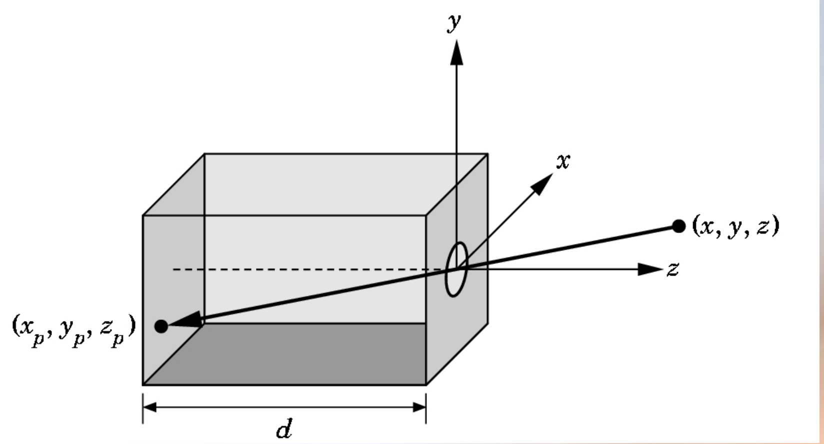
Shadow Mask CRT



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

Pinhole Camera

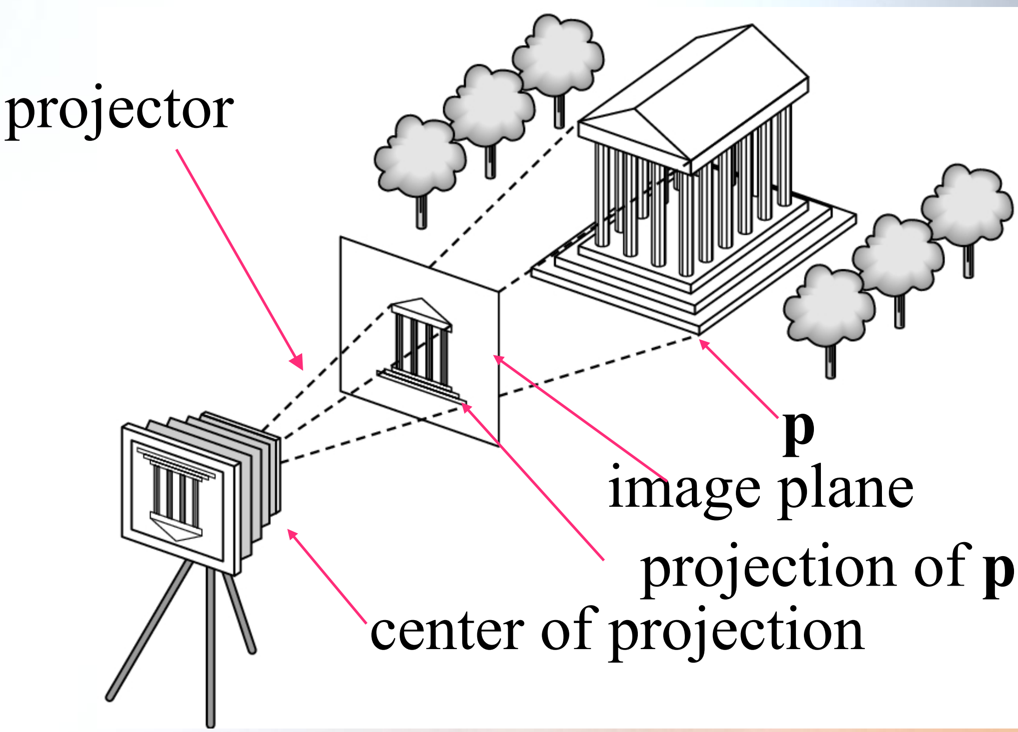


Use trigonometry to find projection of point at (x,y,z)

$x_p = -x/z/d$ $y_p = -y/z/d$ $z_p = d$

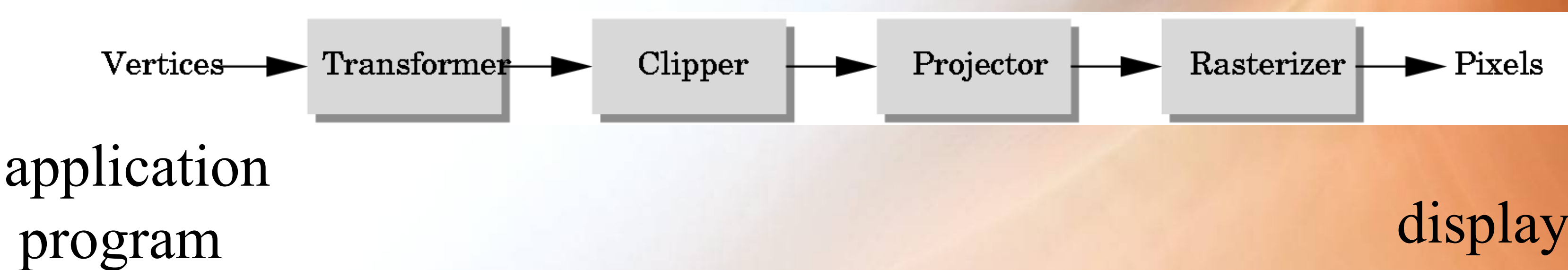
These are equations of simple perspective

Synthetic Camera Model



Practical Approach

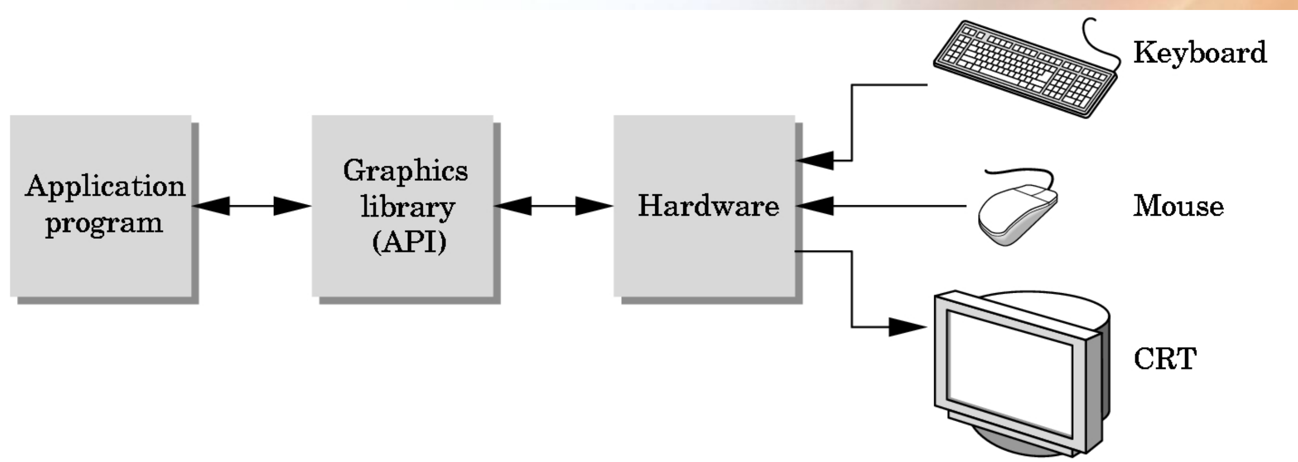
- Process objects one at a time in the order they are generated by the application
  - Can consider only local lighting
- Pipeline architecture



- All steps can be implemented in hardware on the graphics card

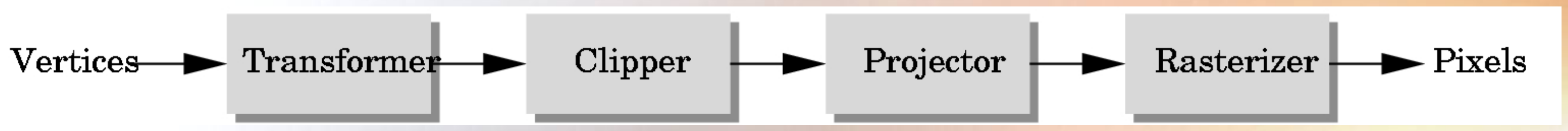
The Programmer's Interface

- Programmer sees the graphics system through a software interface: the Application Programmer Interface (API)



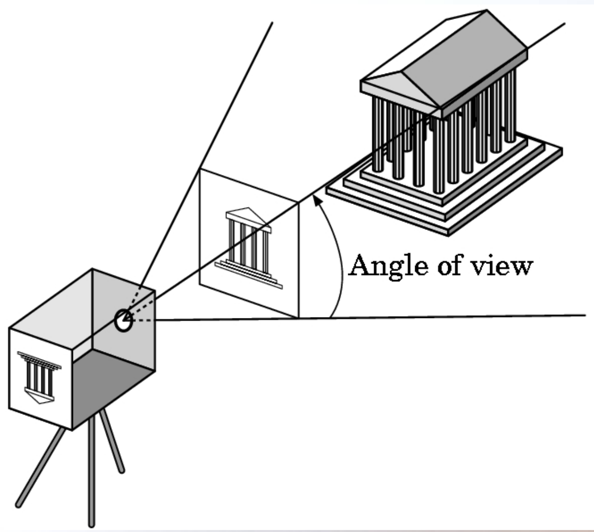
Following the Pipeline: Transformations

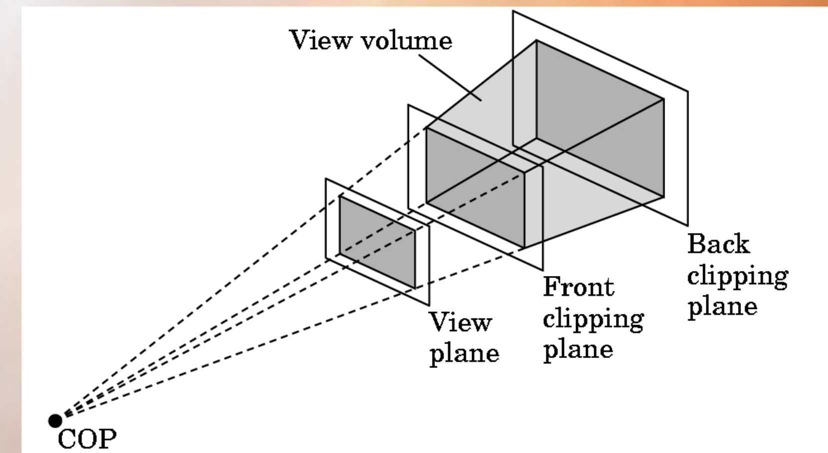
- Much of the work in the pipeline is in converting object representations from one coordinate system to another
  - World coordinates
  - Camera coordinates
  - Screen coordinates
- Every change of coordinates is equivalent to a matrix transformation



Clipping

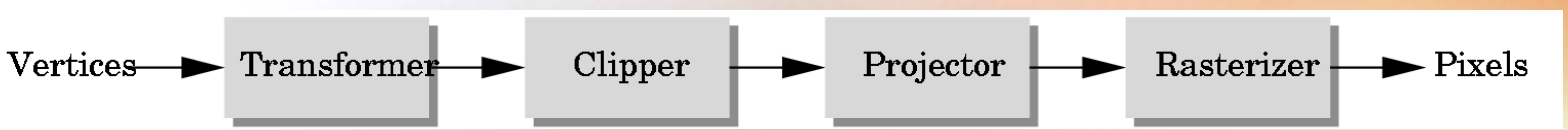
- Objects that are not within the viewing volume are said to be *clipped* out of the scene





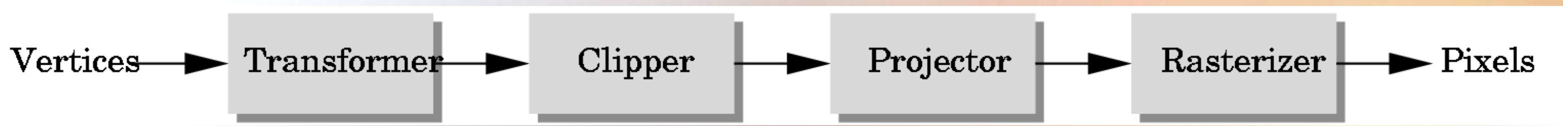
Projection

- Must carry out the process that combines the 3D viewer with the 3D objects to produce the 2D image
  - Perspective projections: all projectors meet at the center of projection
  - Parallel projection: projectors are parallel, center of projection is replaced by a direction of projection



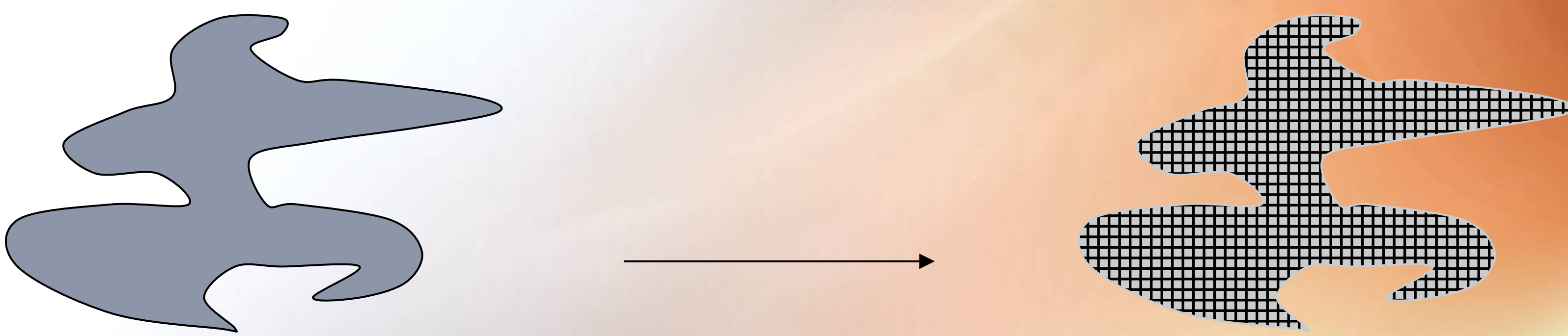
Rasterization

- If an object is visible in the image, the appropriate pixels in the frame buffer must be assigned colors
  - Vertices assembled into objects
  - Effects of lights and materials must be determined
  - Polygons filled with interior colors/shades
  - Must have also determined which objects are in front (hidden surface removal)



Rasterization

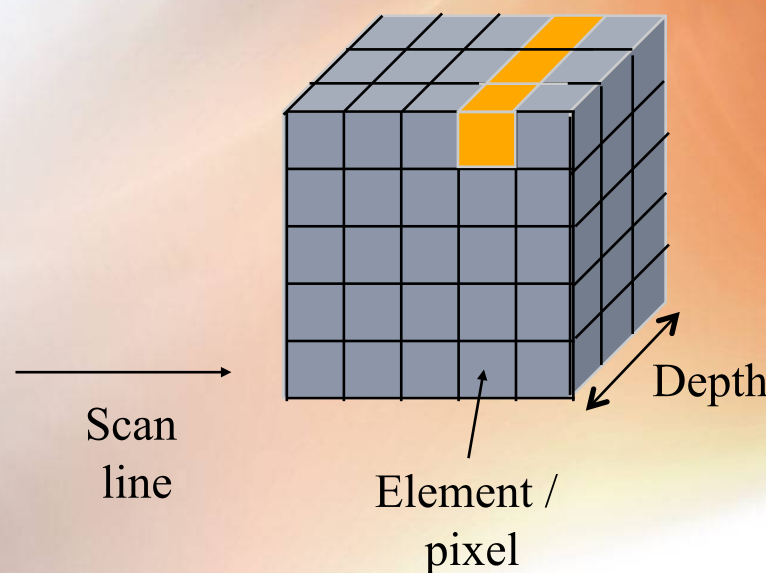
The process of transforming geometric shapes into discrete raster grids.



Raster Graphics

- **Spatial resolution= elements×scan lines**
  e.g. 5 × 5 (below)
- **Intensity or color resolution = 2^{Depth}**

here, each pixel can have any of   2^3 = 8 possible values.



Color

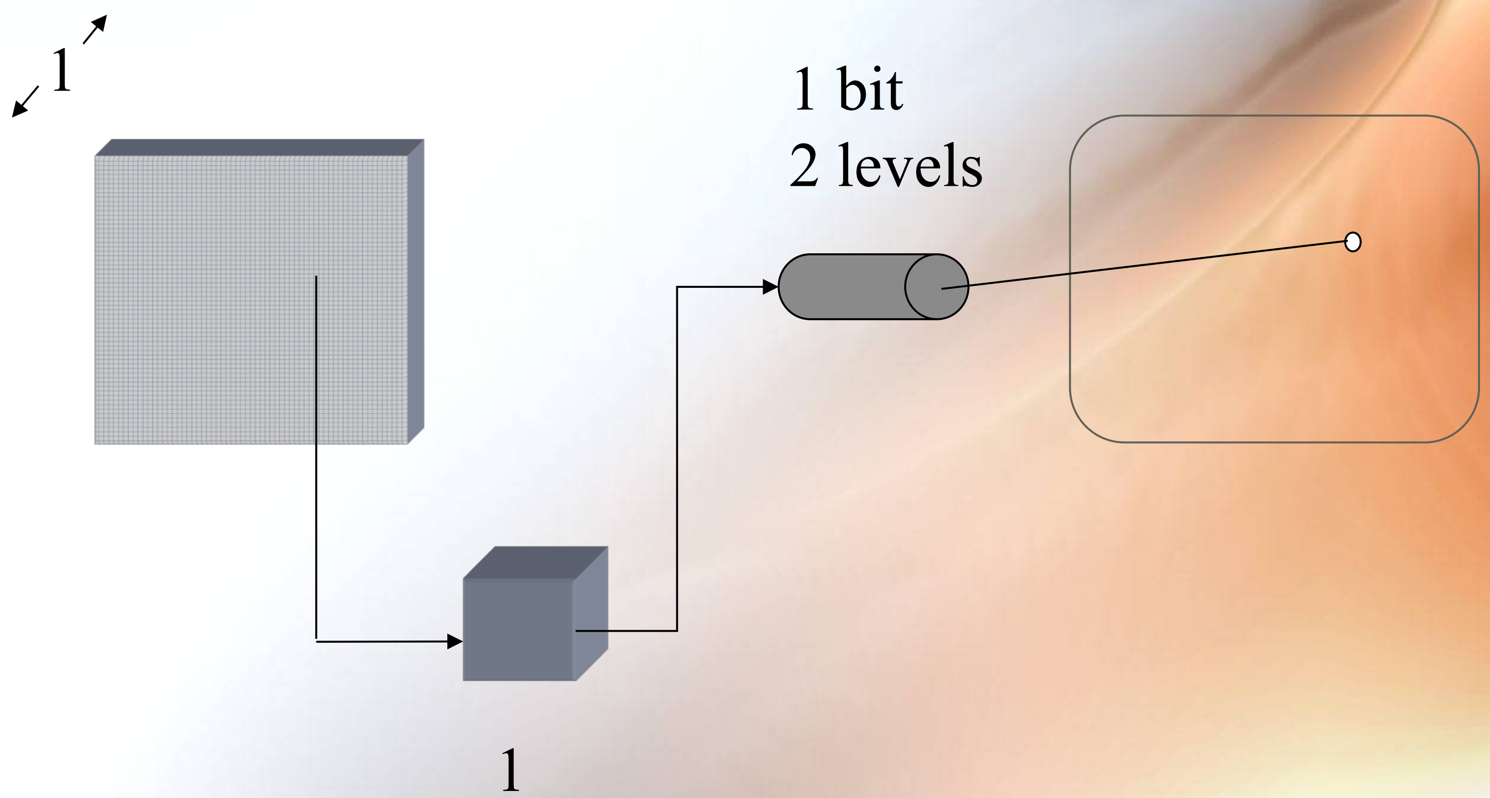
$C = rR + gG + bB$

- Minimum number of levels for just noticeable intensity difference is 100, or about 7 bits.
- Thus, use at least 8 bits per color (R, G, B) (10-24 for photographic quality)

Frame Buffer Architectures

- The raster image is stored in a “frame buffer” memory.
- The frame buffer is built from one or more “bit planes” --two-dimensional arrays of bits.
- This memory is usually peripheral to the host on a video card.

Black and White Frame Buffer with 1 Bit Plane



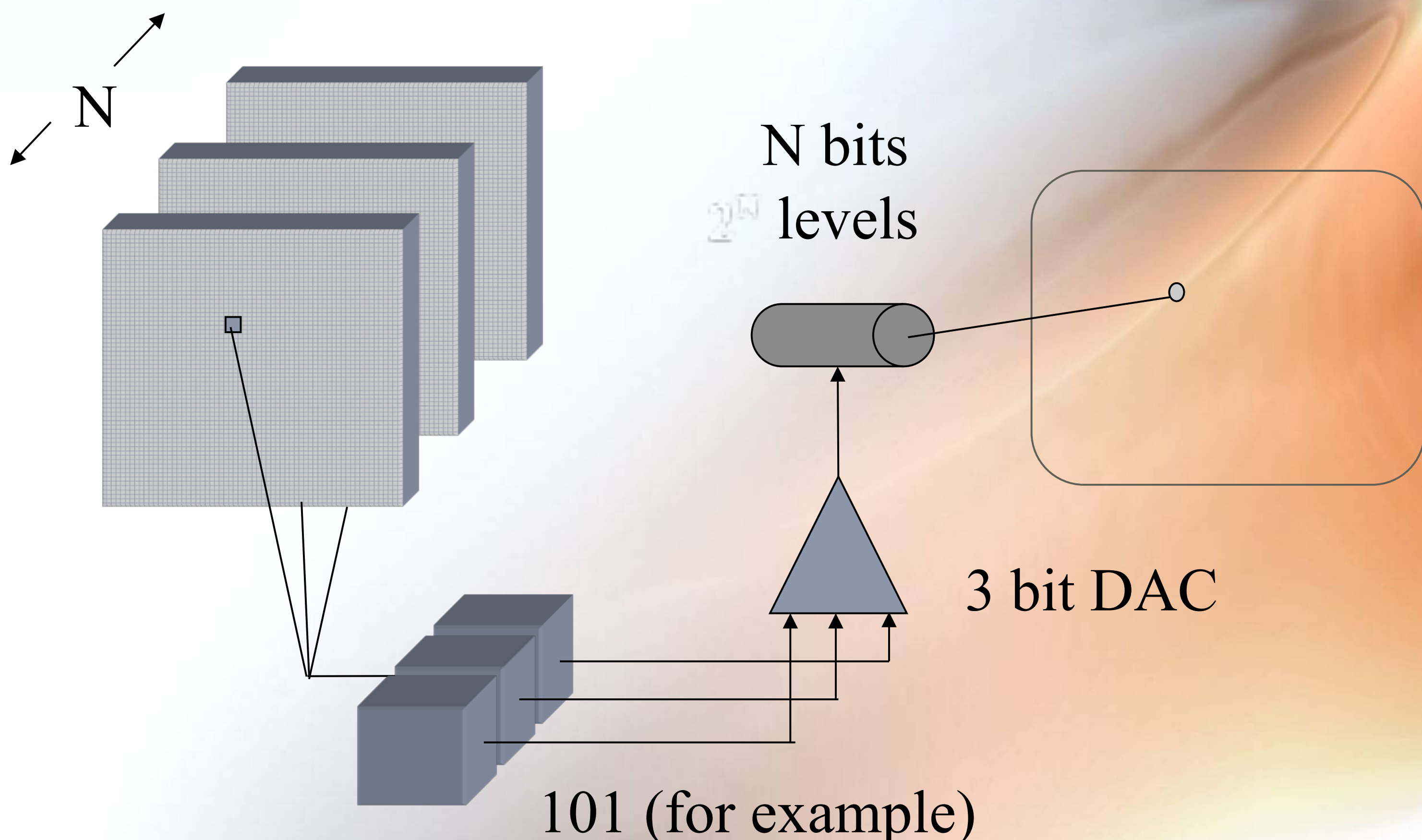
Frame Buffer Configurations

N: number of bit planes

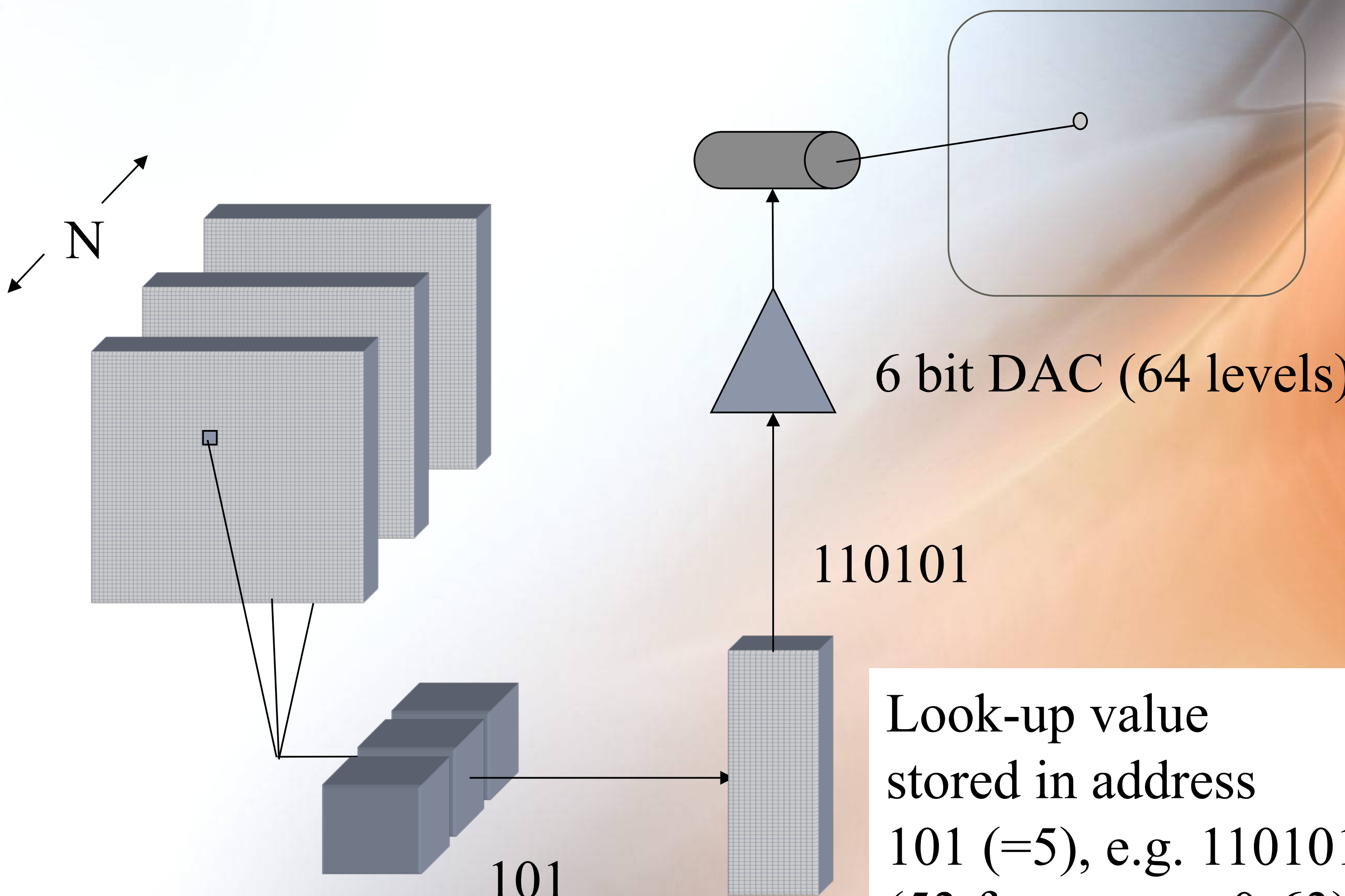
=> 2^N intensity levels

- **Can drive a digital-to-analog converter directly, or use these values as an index into a “lookup table” (“color map”, etc.)**

Grey Scale Frame Buffer with N Bit Planes

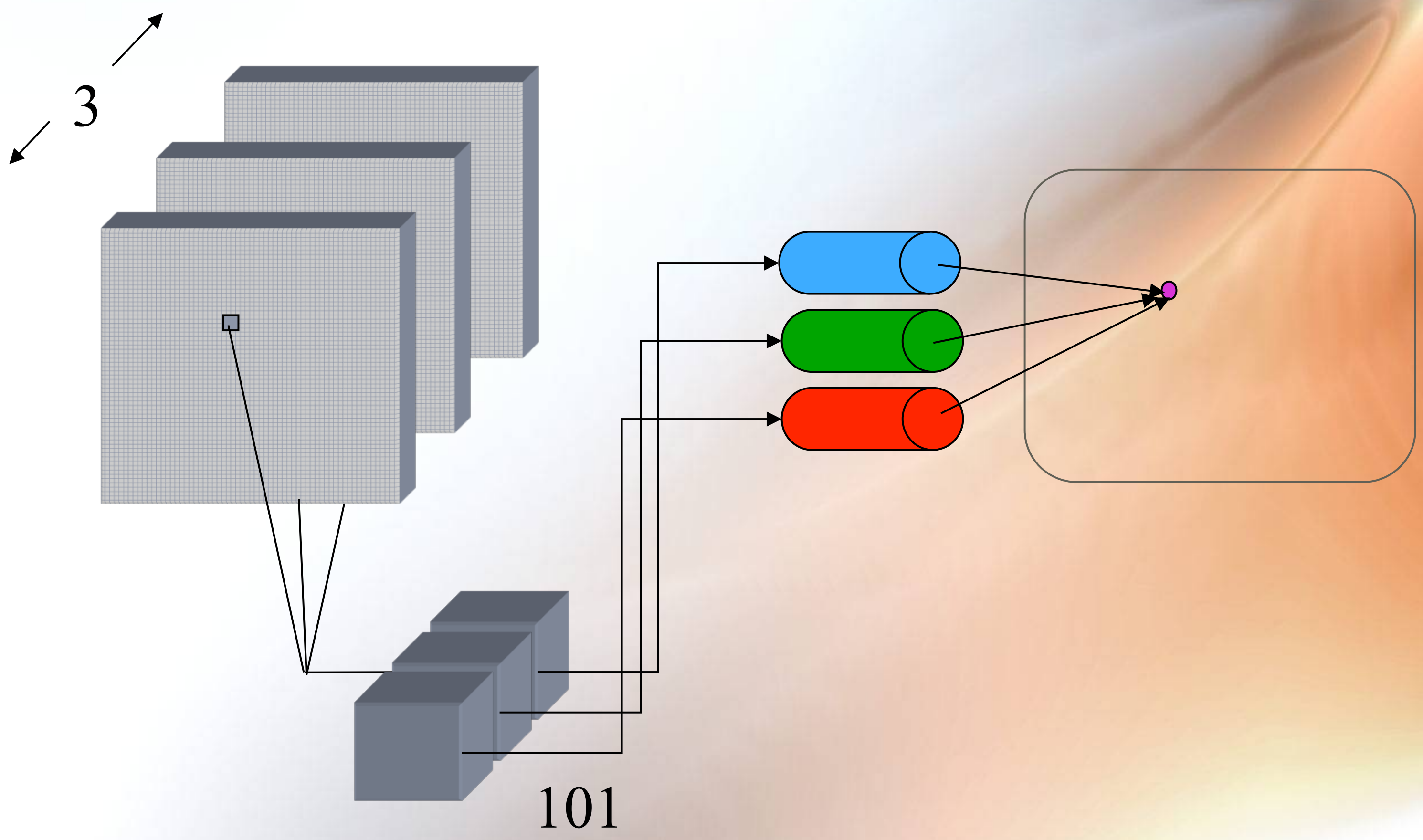


Grey Scale Frame Buffer with Look-Up Table



Look-up value stored in address 101 (=5), e.g. 110101 (53 from range 0-63)

8 Color Frame Buffer (3 Bit Planes)



3 Bit Planes = 8 Colors

0 0 0 == BLACK

1 0 0 == RED

0 1 0 == GREEN

0 0 1 == BLUE

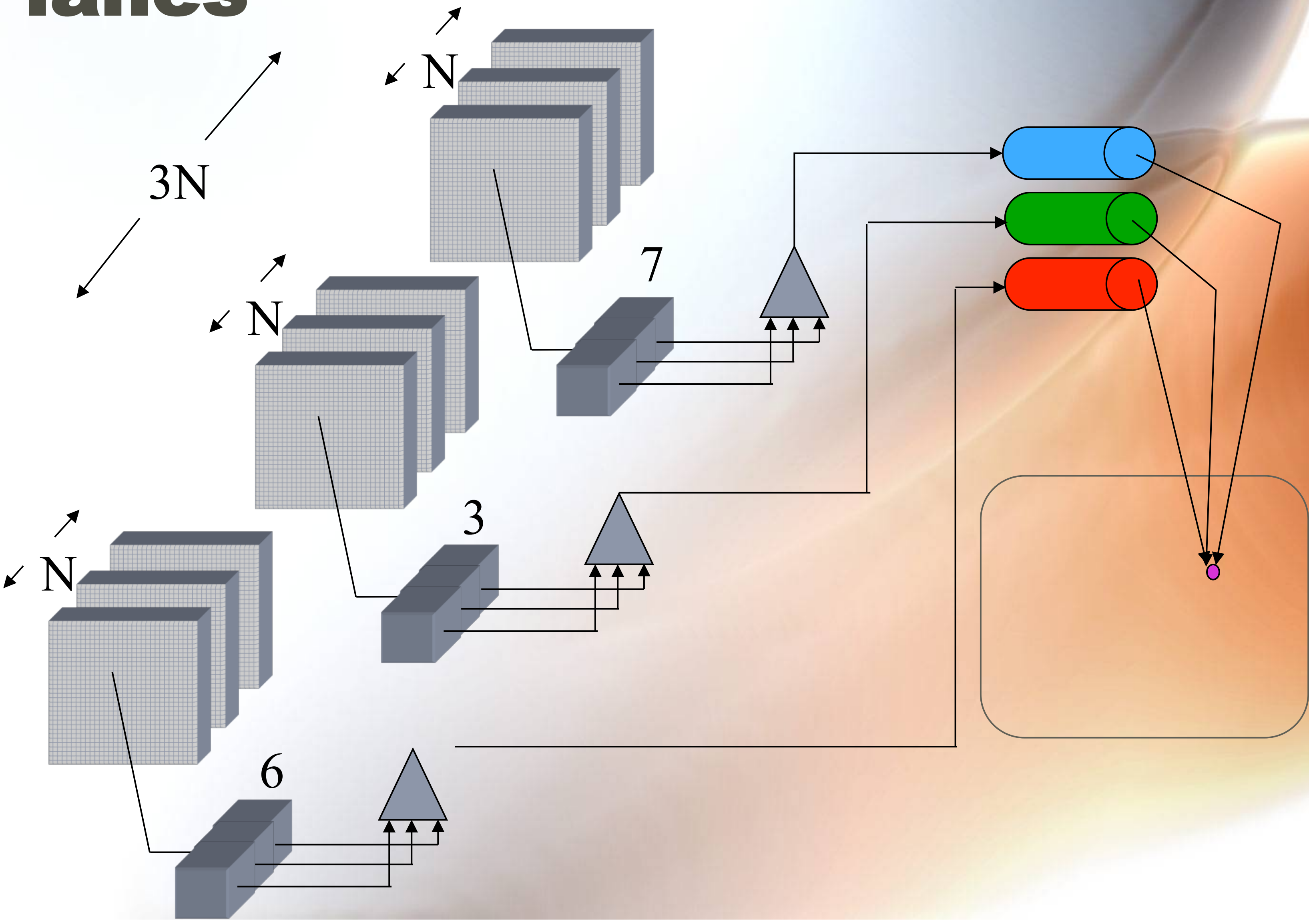
1 1 0 == YELLOW

0 1 1 == CYAN

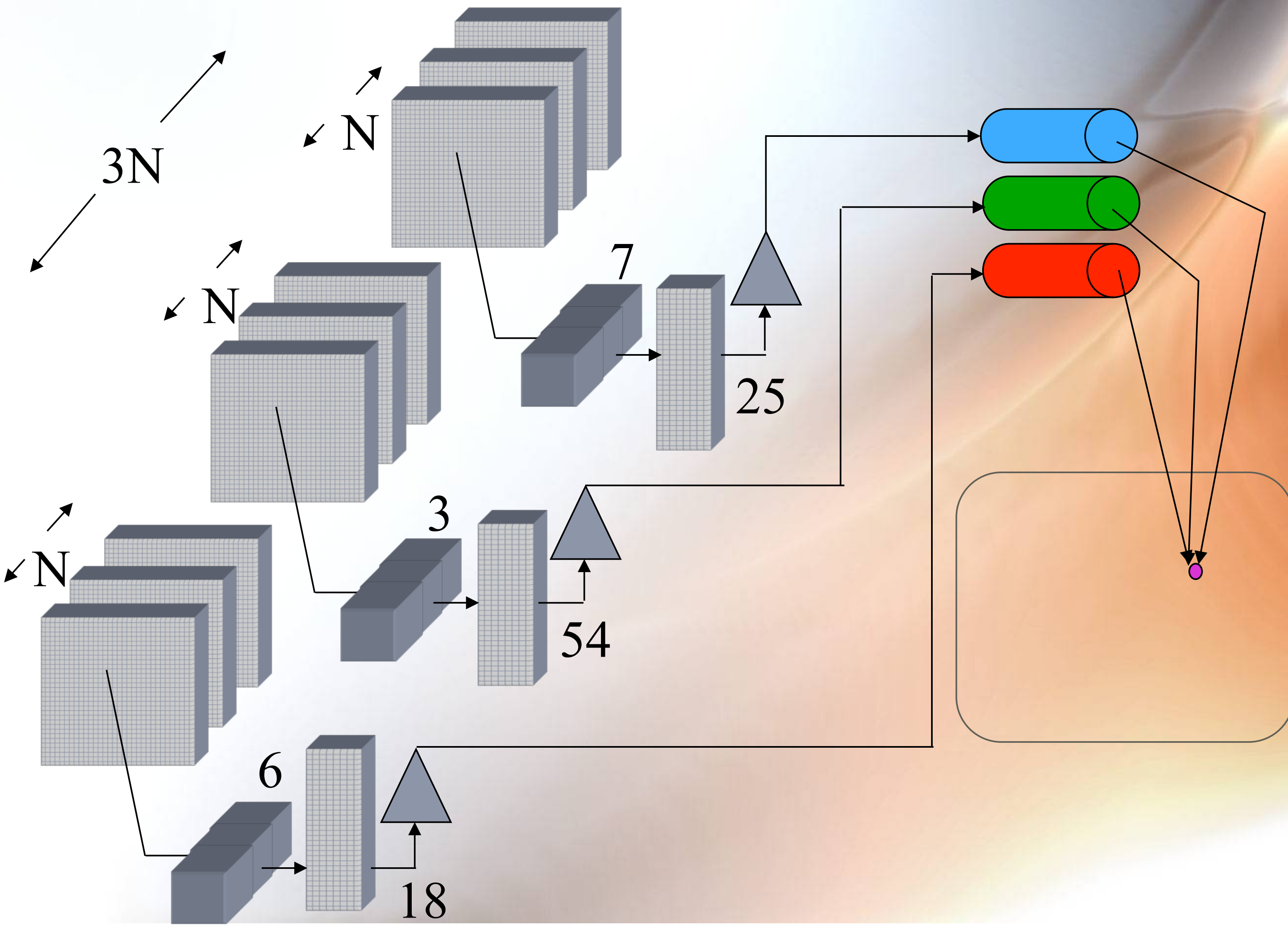
1 0 1 == MAGENTA

1 1 1 == WHITE

Color Frame Buffer with 3N Bit Planes



CFB with 3N Bit Planes and LUT



Newer Architectures

- Additional memory planes per pixel:
  - Z (depth)
  - Alpha blending
  - Stencils (to select areas for operations)
- Separate texture memory

Raster Graphics "Primitives"

- Points
- Lines
- Rectangles
- Circles or Conics
- Disks
- Polygons
- Characters or Special symbols

  G

  Ω
- Bit maps, sprites, or patterns