Week 01:
Image Formation

CS-537: Interactive Computer Graphics
Dr. Chloe LeGendre
Department of Computer Science

For academic use only.

Some materials from the companion slides of Angel and Shreiner, “Interactive Computer Graphics, A Top-Down Approach with WebGL.”

Objectives

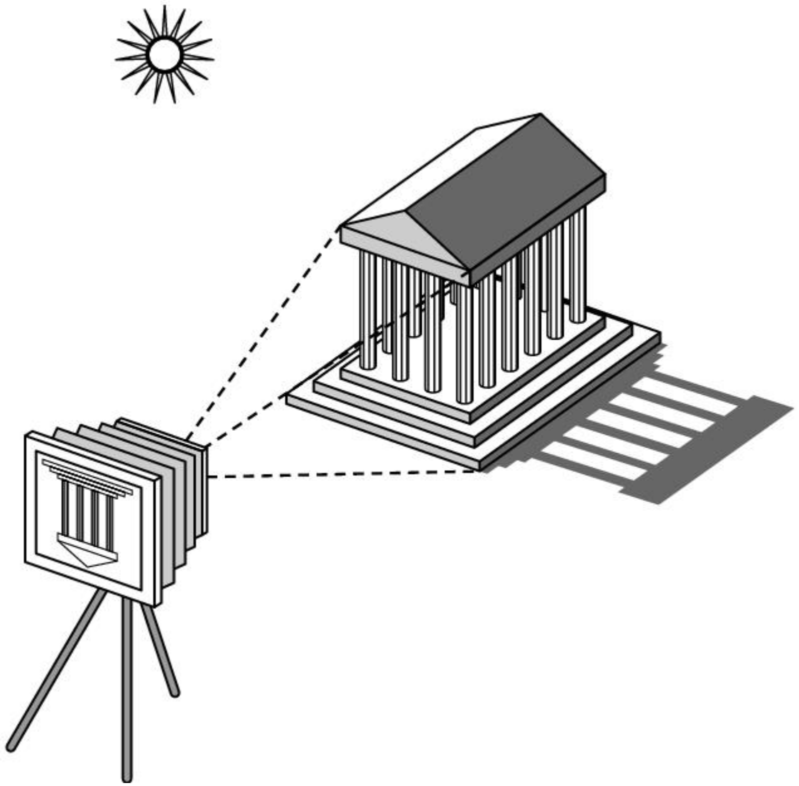
- Describe fundamental image formation notions
- Describe the physical basis for image formation:
  - Light
  - Color
  - Human perception
- Define a synthetic camera model
- Describe other potential models

Image Formation

- In computer graphics, we form images which are generally two dimensional (2D) using a process analogous to how images are formed by physical imaging systems, which could include:
  - Cameras
  - Microscopes
  - Telescopes
  - The human visual system
- The objects represented in images formed by these physical imaging systems are those of the real world
- In computer graphics, we'll want to form images of virtual objects represented in computer memory

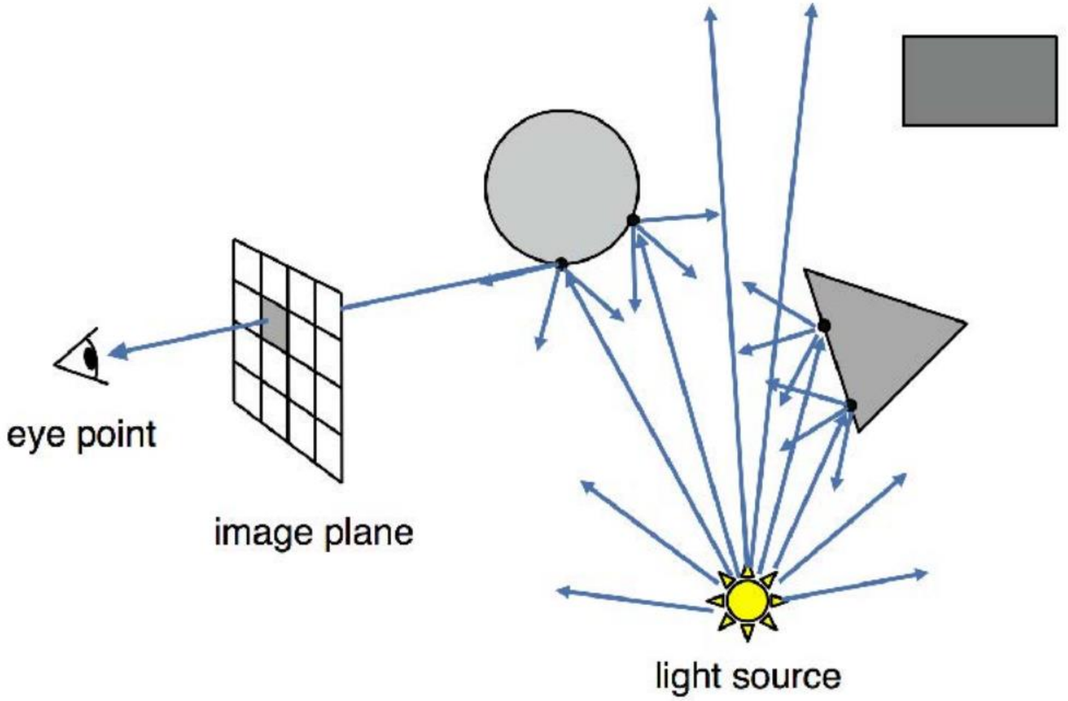
Elements of Image Formation

- To form an image with a physical imaging system, you need:
  - One or more objects
  - A viewer (“observer”)
  - One or more light sources
- In the real world, objects are made of materials that have properties that govern how light interacts with them (shiny? dull? red? blue?)
- Light reflects off the objects and towards the camera, forming the image
- Note: the objects, viewer, and light sources are independent of each other in the model



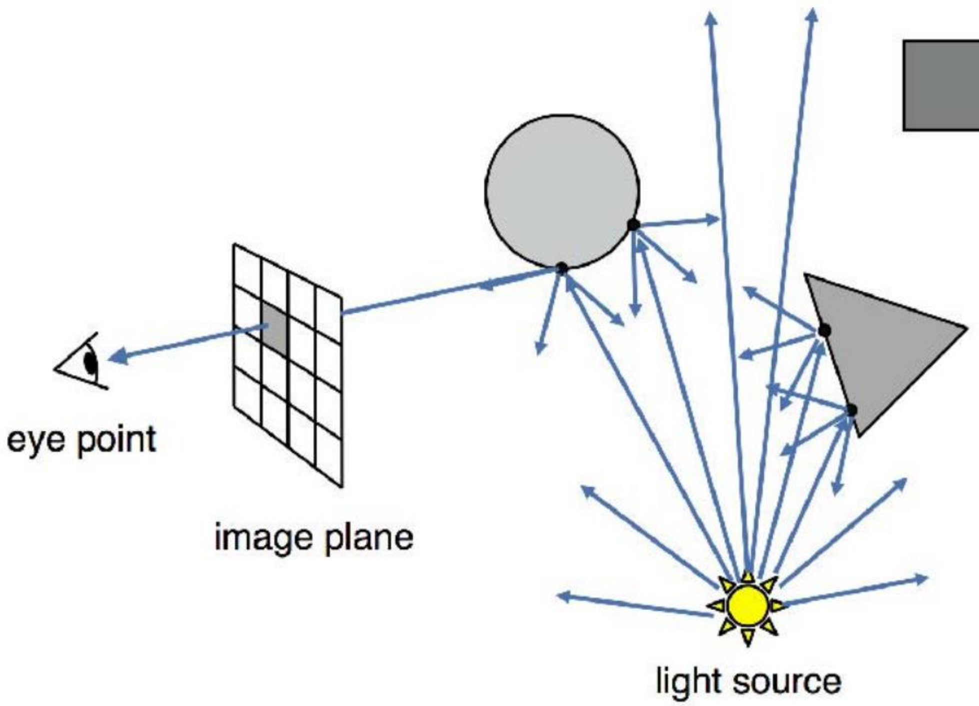
Geometric Optics

- We need to know how light interacts with the scene objects and is reflected towards the camera.
- One way to form an image is to follow rays of light from a point source, finding which rays enter the lens of the camera.
- However, each ray of light may have multiple interactions with objects before being absorbed or going to infinity.



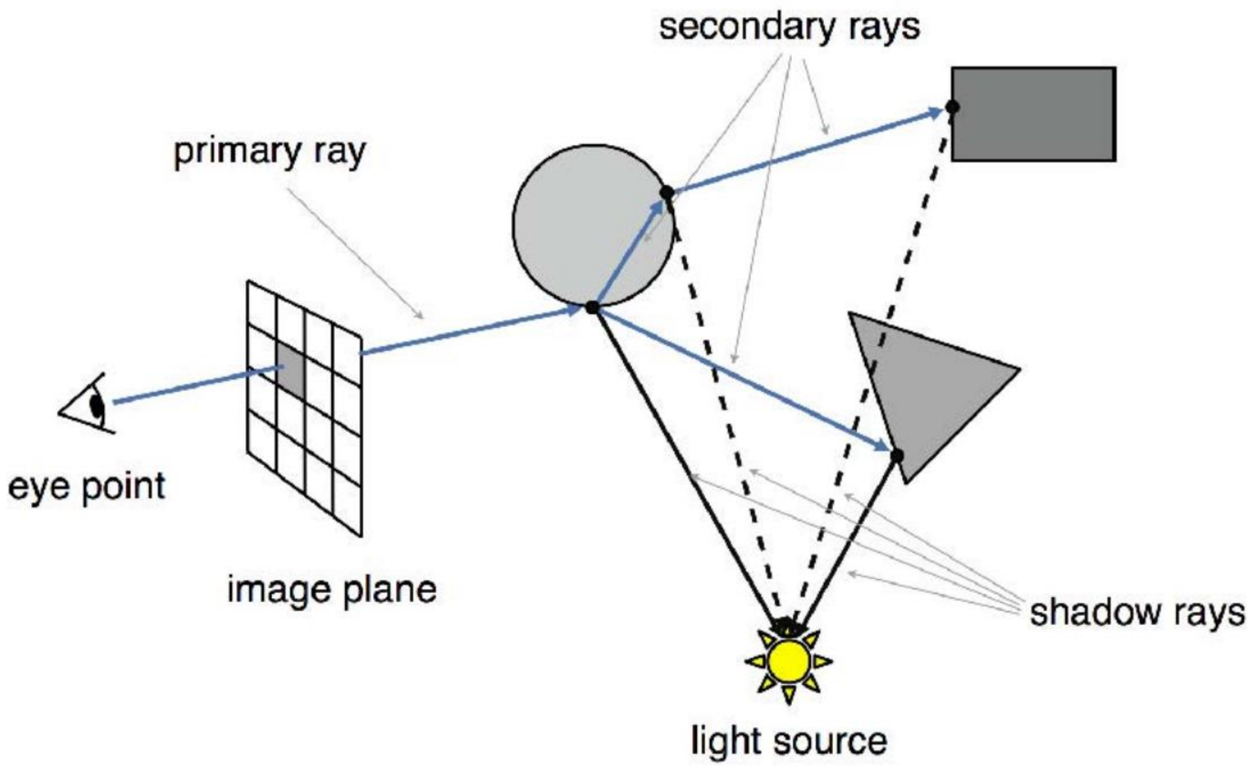
Ray Tracing: Light-Oriented (Forward Ray Tracing)

- Starting with the light source and tracing potential rays towards the scene objects.
- Drawback: only a fraction of rays will hit the image plane, so this is potentially inefficient.



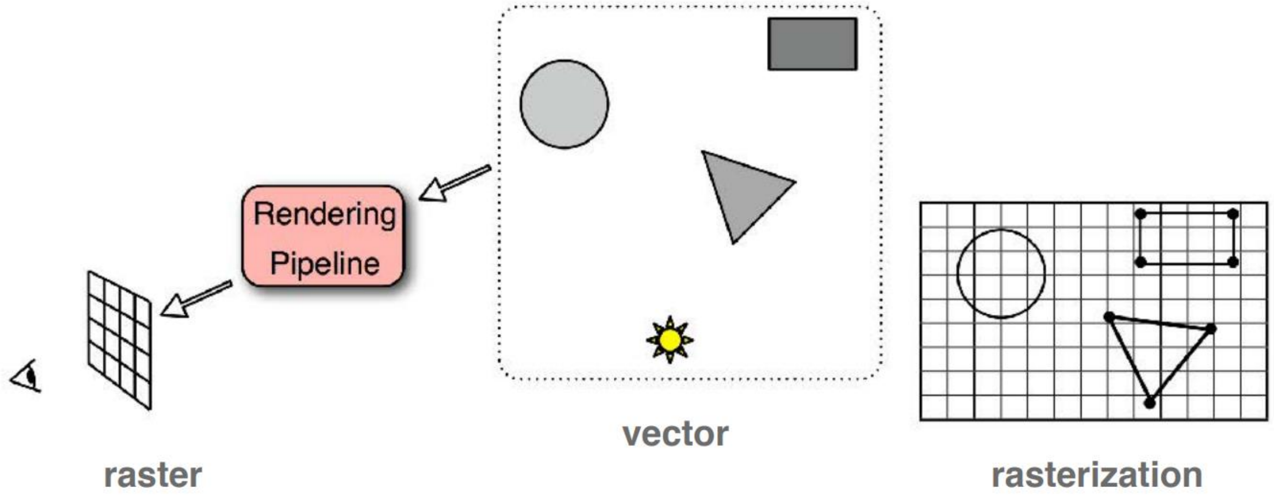
Ray Tracing: Eye-Oriented (Backwards Ray Tracing, or just “Ray Tracing”)

- Starting from the camera, trace rays through each pixel from the eye point, towards the objects in the scenes, and then from the objects towards the lights
- Prevents you from needing to trace unnecessary paths.



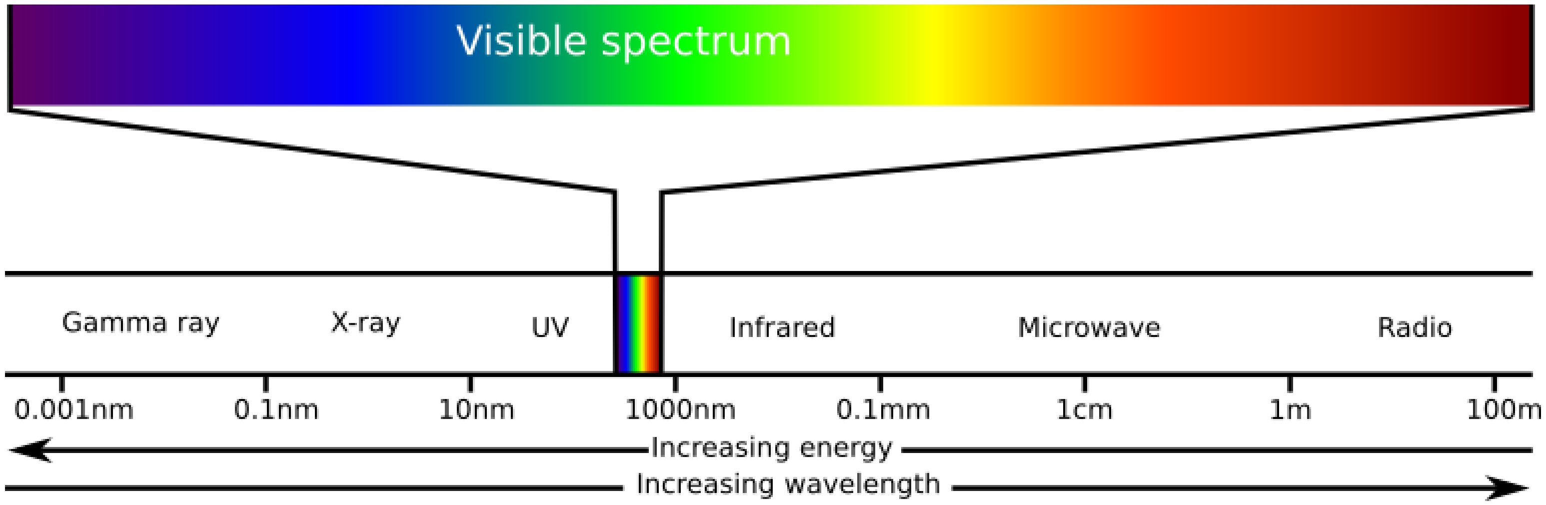
Object Oriented (Forward Rendering)

- Scene is composed of geometric structures with the building block of a triangle. Each triangle is projected, colored depending on some contribution of scene light sources, and painted on the screen.
- Most of this class will focus on this type of rendering, using WebGL, a graphics library for web programming.
- Why? Historically, ray tracing was not well-suited for real-time / interactive applications due to compute constraints (too slow), but this is changing in 2020 (state-of-the-art).



Light

- *Light* is the part of the electromagnetic spectrum that causes a reaction in our visual systems
- Generally these are wavelengths in the range of about 350-750 nm (nanometers)
- Longer wavelengths appear as reds and short wavelengths as blues
- Light sources like light bulbs, lamps, the sun each emit electromagnetic radiation



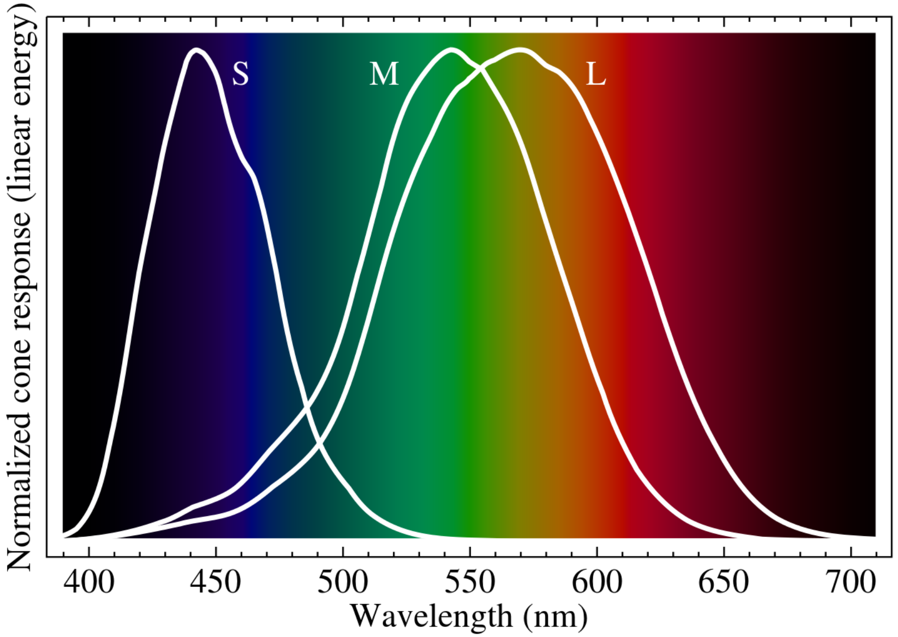
Luminance and Color Images

- Luminance Image
  - Monochromatic
  - Values in the image are gray levels
  - Analogous to working with black and white film or TV
- Color Image
  - Has the perceptual attributes of hue, saturation, and lightness
  - Do we have to match every frequency in the visible electromagnetic spectrum?
    - No - we can make some assumptions.

Color Images and Tristimulus Values

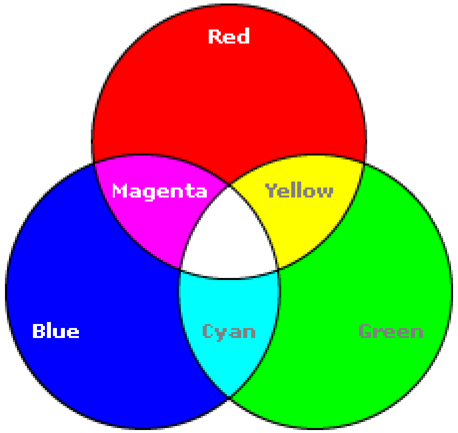
The human visual system has two types of “sensors”

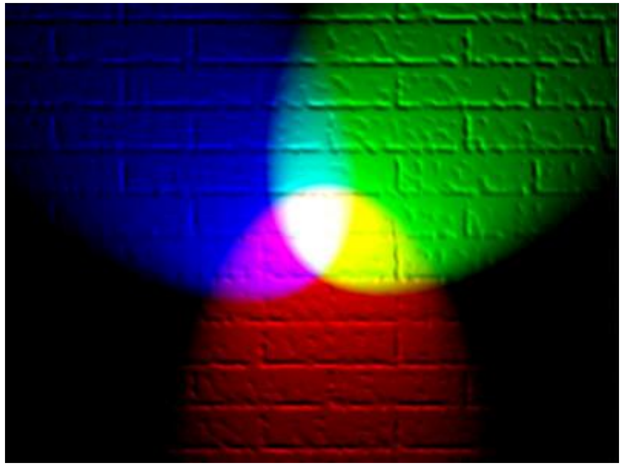
- **Rods**: Monochromatic, for night vision, and provide a general cue for how bright/dark parts of a scene are
- **Cones**: Color sensitive (sensitive to different parts of visible spectrum)
  - Three types: S, M, L (short/medium/long wavelength)
  - Rather than a full spectrum, three values are sent to the brain as the visual signal, corresponding to the integration of the fully spectral signal by the sensitivity of the cone cells (plot on the right). These are called the “tristimulus values”
  - For graphics, this means that if we want to represent a color of a surface or a light source, we only need three primary colors corresponding to the S, M, L cone response functions.

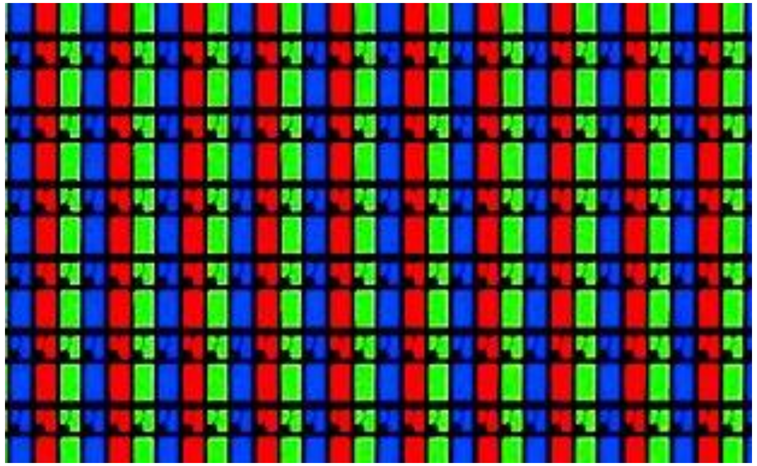


Additive and Subtractive Color Models

- Additive color
  - Form a color by adding amounts of three primaries together
  - Example: LCD panel (liquid crystal display), made of red, green, and blue light sources
  - Primaries: Red, green, and blue light, matching the S, M, L cones
  - In additive color models, the primaries summed together produce white.



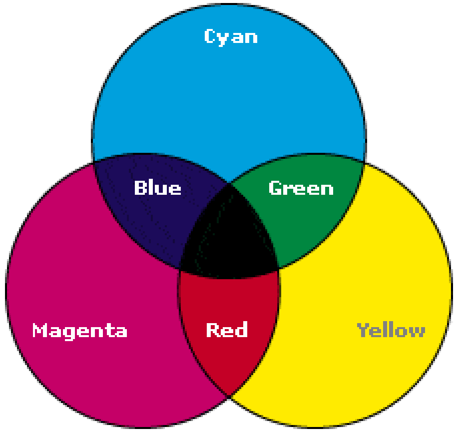


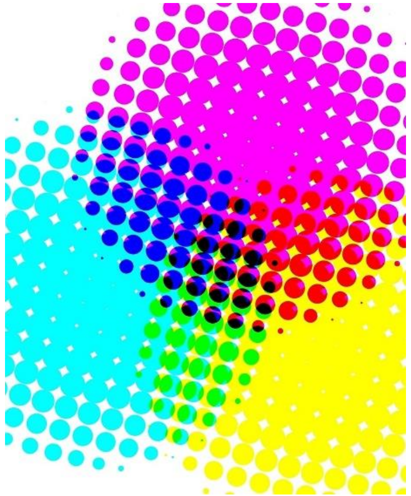


Microscope close-up image of an LCD, with RGB light sources. One “pixel” is represented by a group of three lights: one each of RGB. Example: turning on RG produces yellow when viewed from far away.

Additive and Subtractive Color Models (II)

- Subtractive color
  - Possible to create with light: form a color by filtering white light with cyan (C), magenta (M) and yellow filters (Y)
  - A more intuitive example: printing with ink
  - In subtractive color models, the primaries summed together produce black.





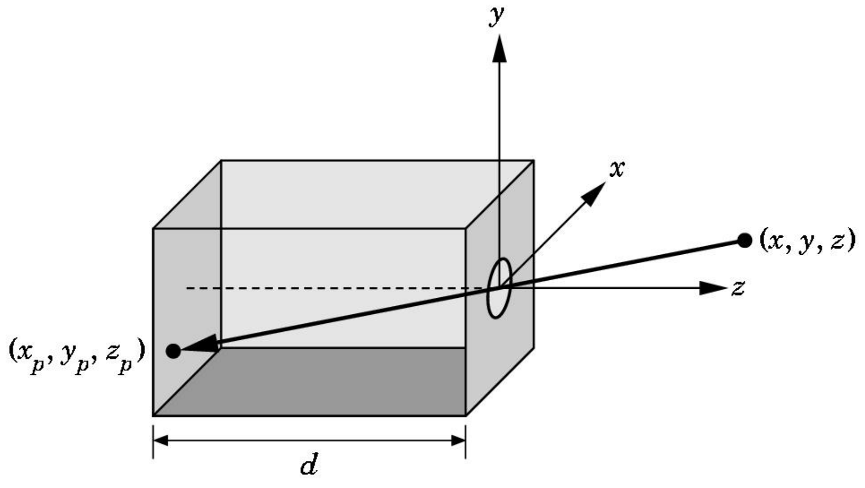
Printing systems use C, M, Y inks to create colors.

(They also usually add black ink, since it appears frequently, and adding the CMY pigments found in most inks creates grey, not black.)

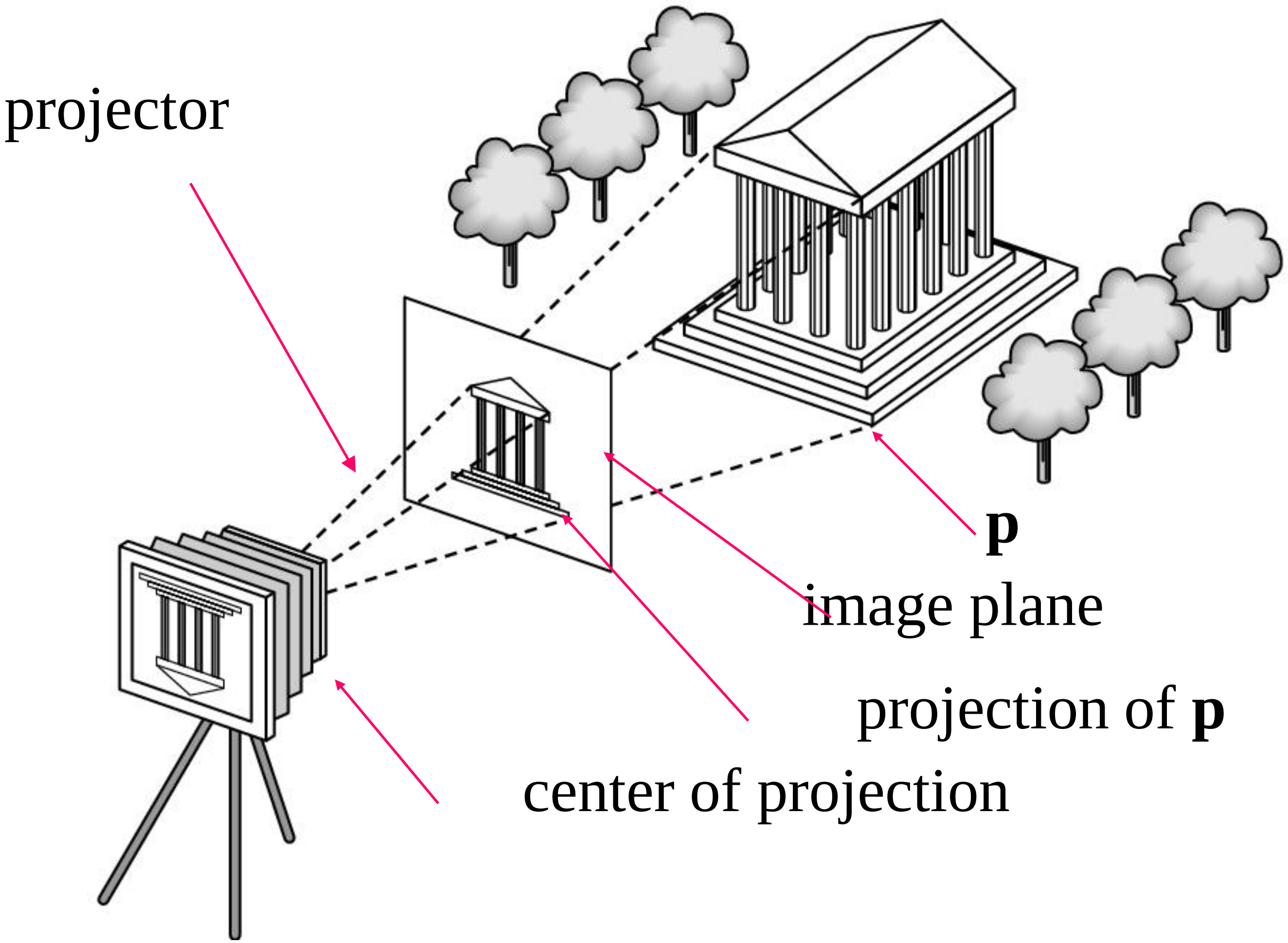
Image Formation: A Pinhole Camera Model

- Pinhole camera: box with a small hole at the center of one side, film placed at the back of the box
- Define point in space, [x, y, z], and seek to determine where it is imaged ("projected") on the film [x_p, y_p, z_p]
- Use trigonometry to find projection of point [x,y,z]

$$x_p = -x/(z/d) \quad y_p = -y/(z/d) \quad z_p = d$$



Synthetic Camera Model - terminology



Advantages to Synthetic Camera Model

- Separation of objects, viewer, light sources
- 2D graphics is a special case of 3D graphics
- Leads to simple / straightforward software API
  - Specify objects, lights, cameras, material attributes
  - Let the implementation determine the image
- Leads to fast hardware implementation using GPU