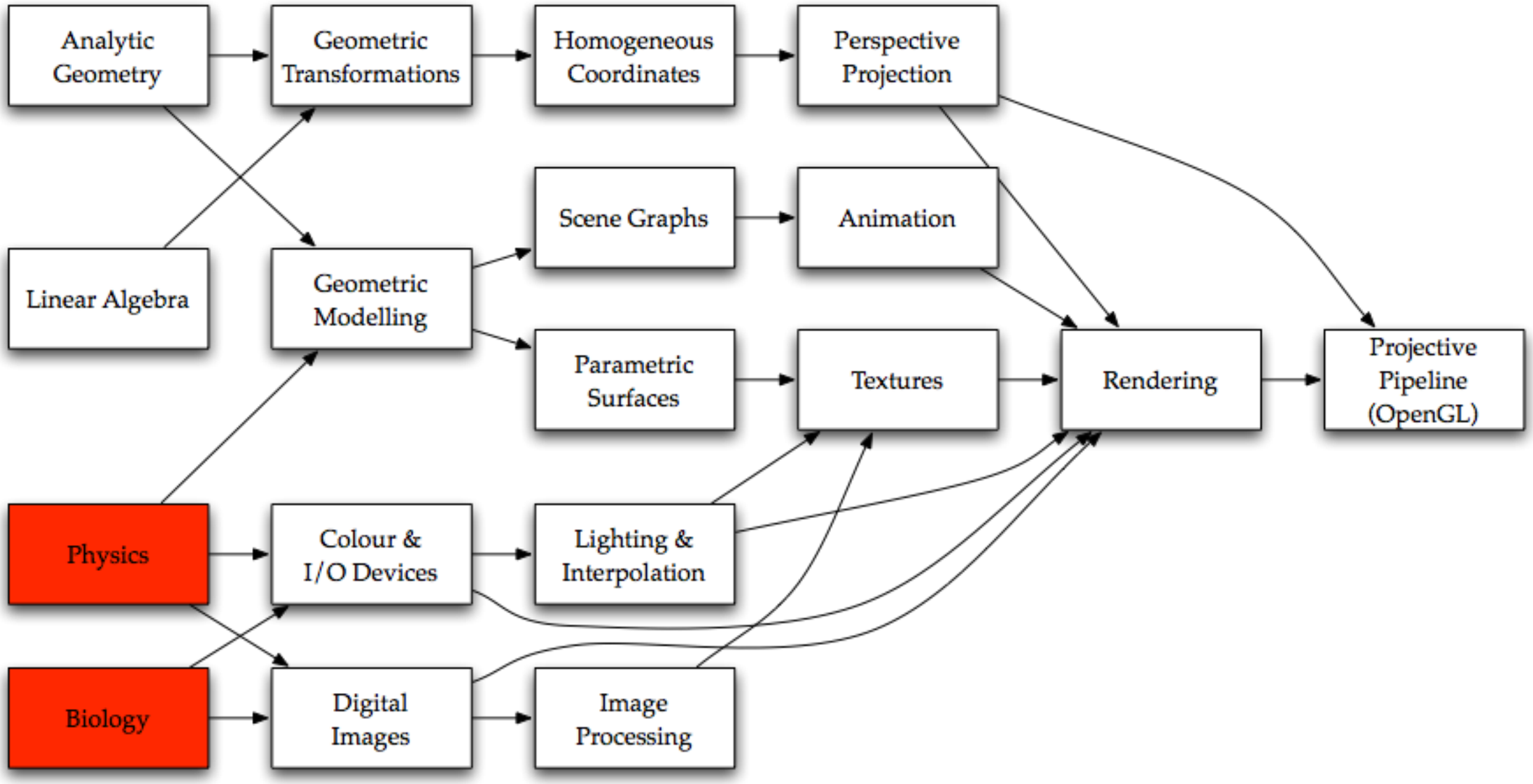
Physics & Biology of Vision

COMP 30020: Intro Computer Graphics

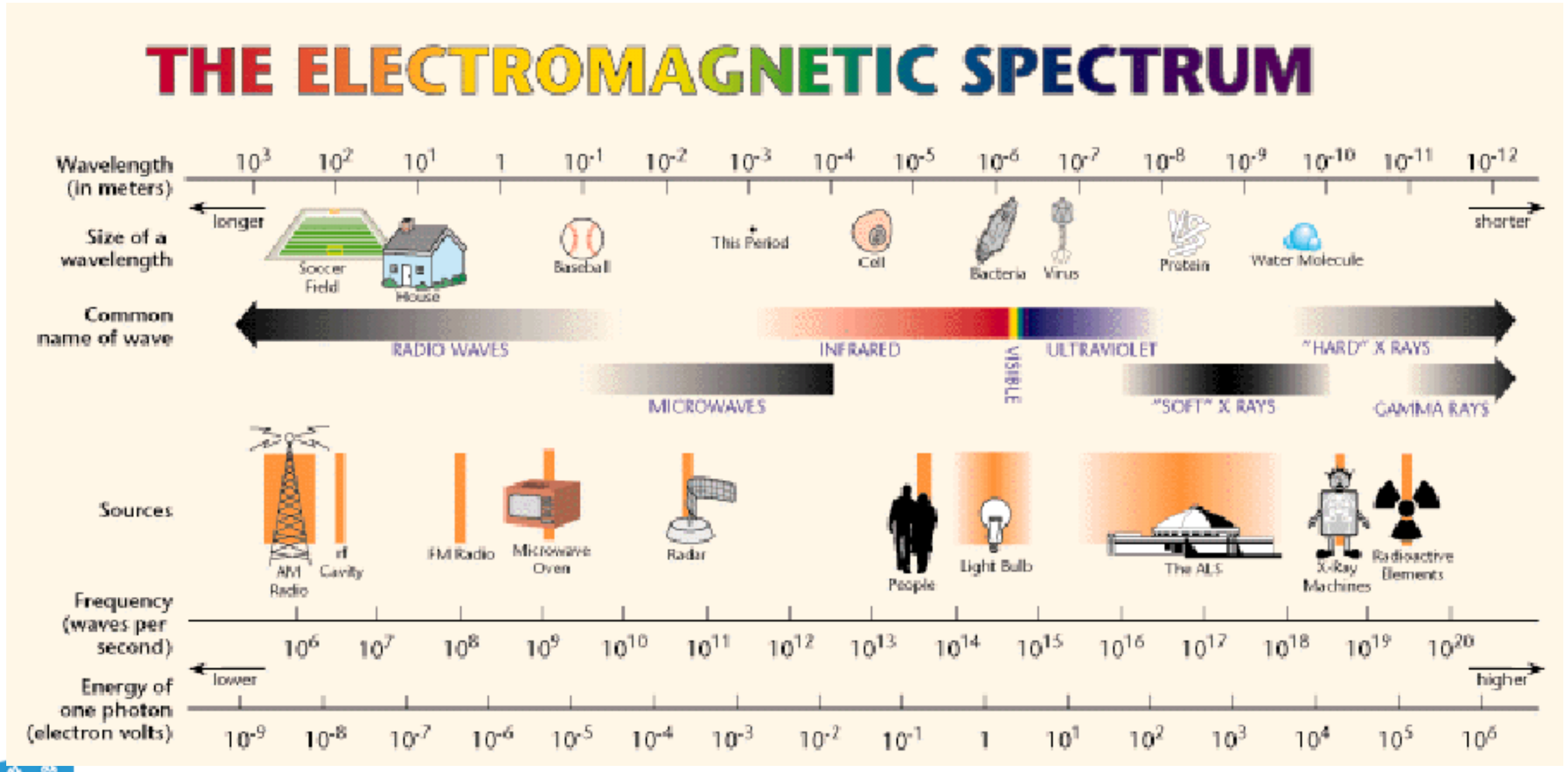
Block Diagram



Physics of Light

- What is Light?
- What is Colour?
- What is White Light?
- How does Light behave?

What is Light?



Waves or Particles?

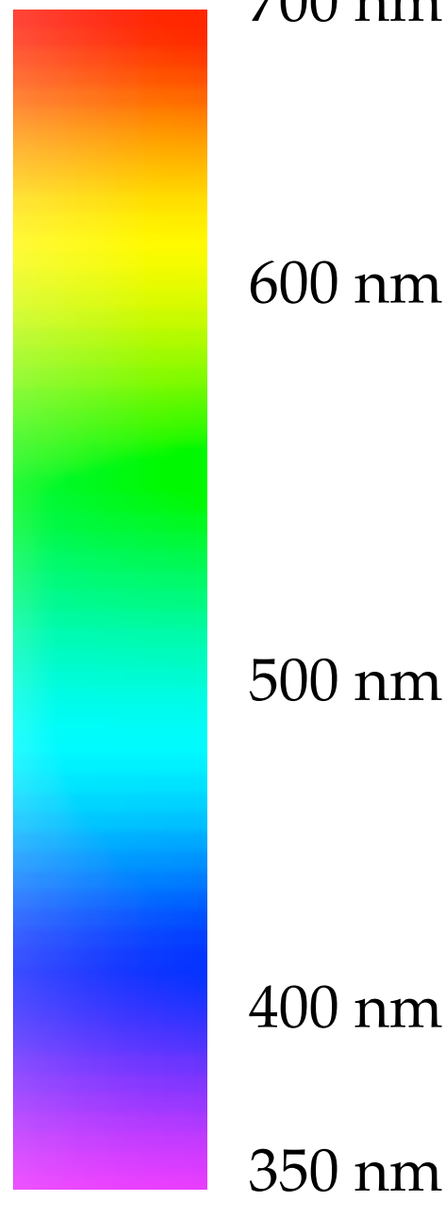
- Major argument in science:
  - is light particles or waves?
- Actually, it's both
  - but we will mostly treat it as particles
  - sometimes we will treat it as waves

Particles & Rays

- Photons are emitted by a source
- Travel at 2.99 x 10^{11} m/s
- Apply Newtonian mechanics
- We can model them with rays

Wavelength & Colour

- Photons carry energy with them
  - Proportional to “wavelength”
  - “Wavelength” is “colour”
- Our eyes can see:
  - wavelengths 350 - 700 nm
  - all the colours of the rainbow



Spectral Distribution

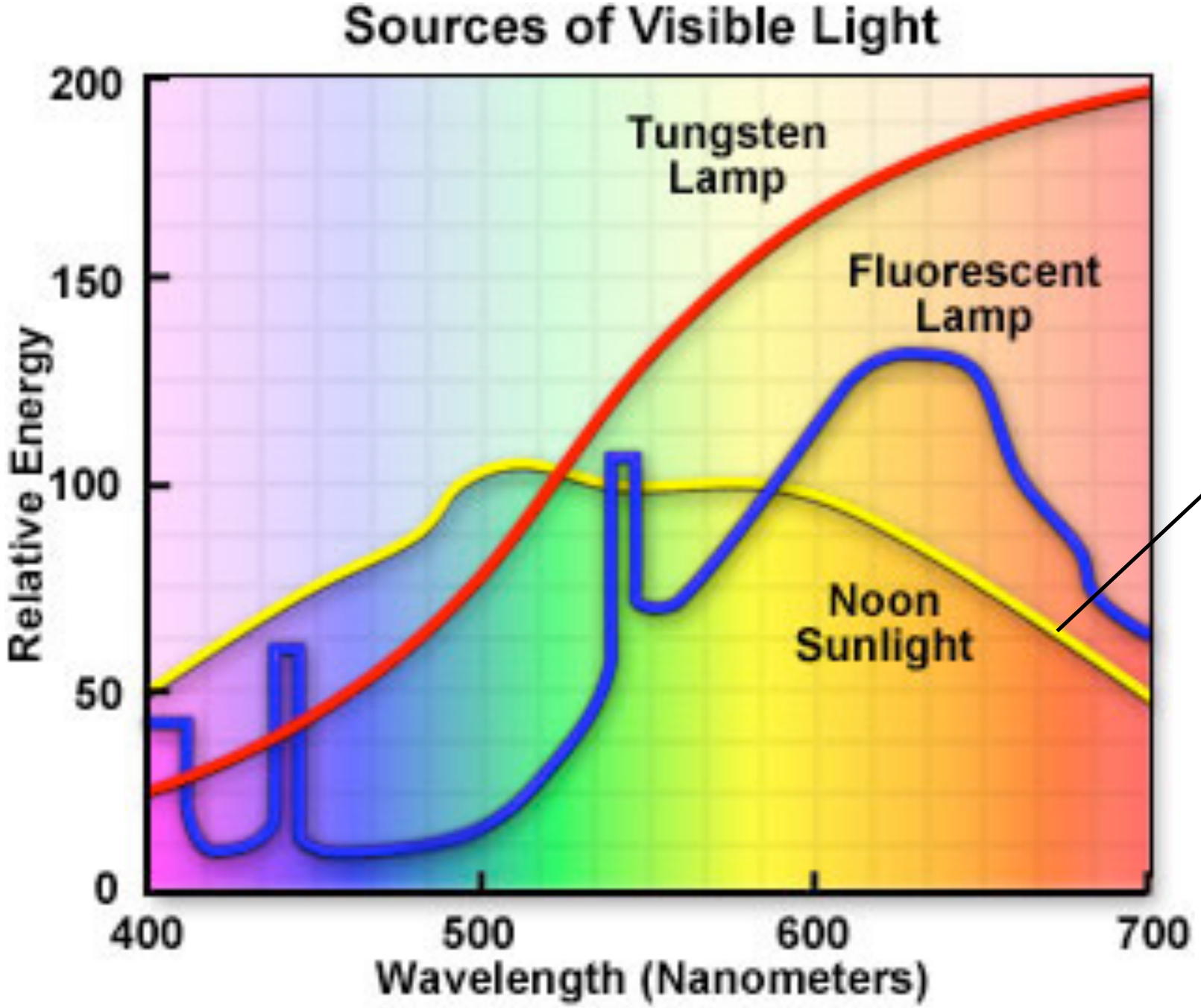


Figure 2

"White" light

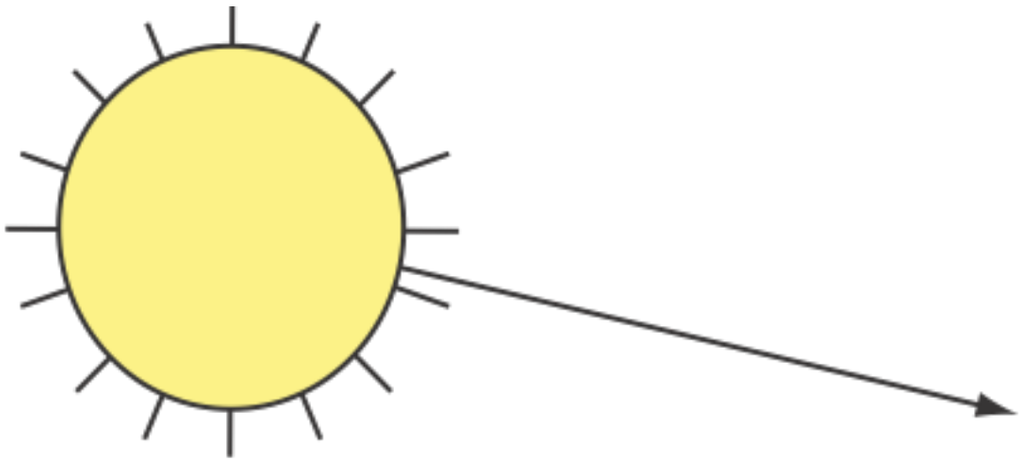
mputer Graphics

Physics of Light

- Light is:
  - emitted
  - reflected
  - absorbed
  - Also but not covered in this course
    - refracted
    - diffracted

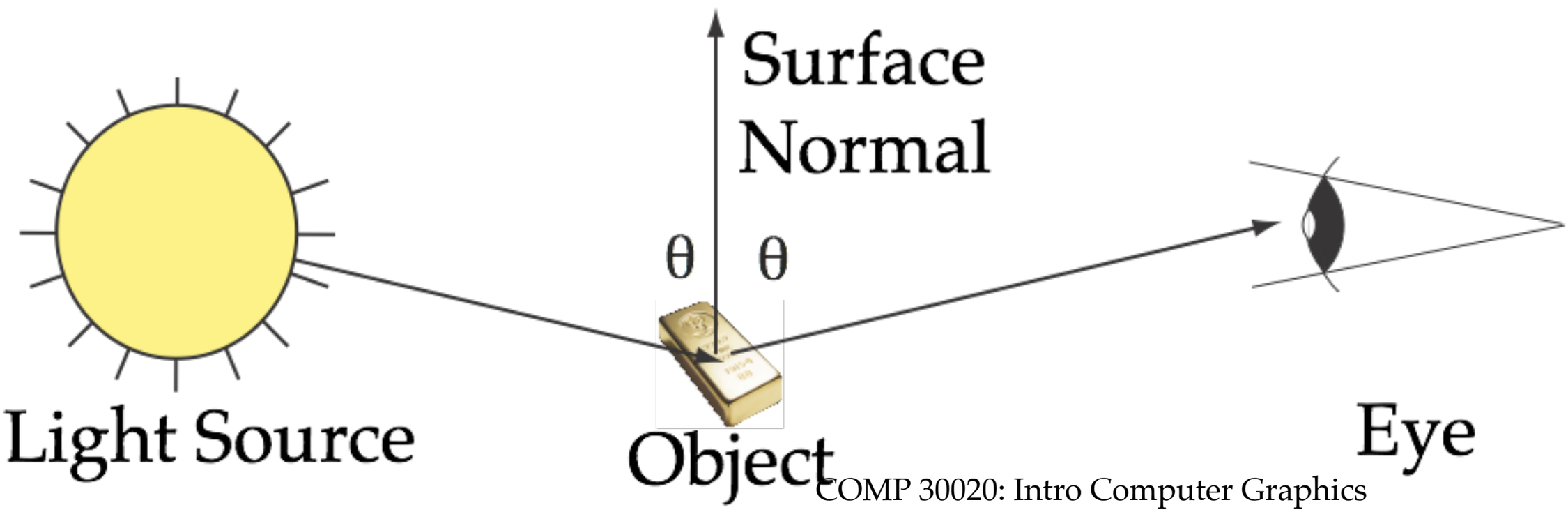
Emission

- Atoms emit photons to lose energy
- Wavelength depends on which atom



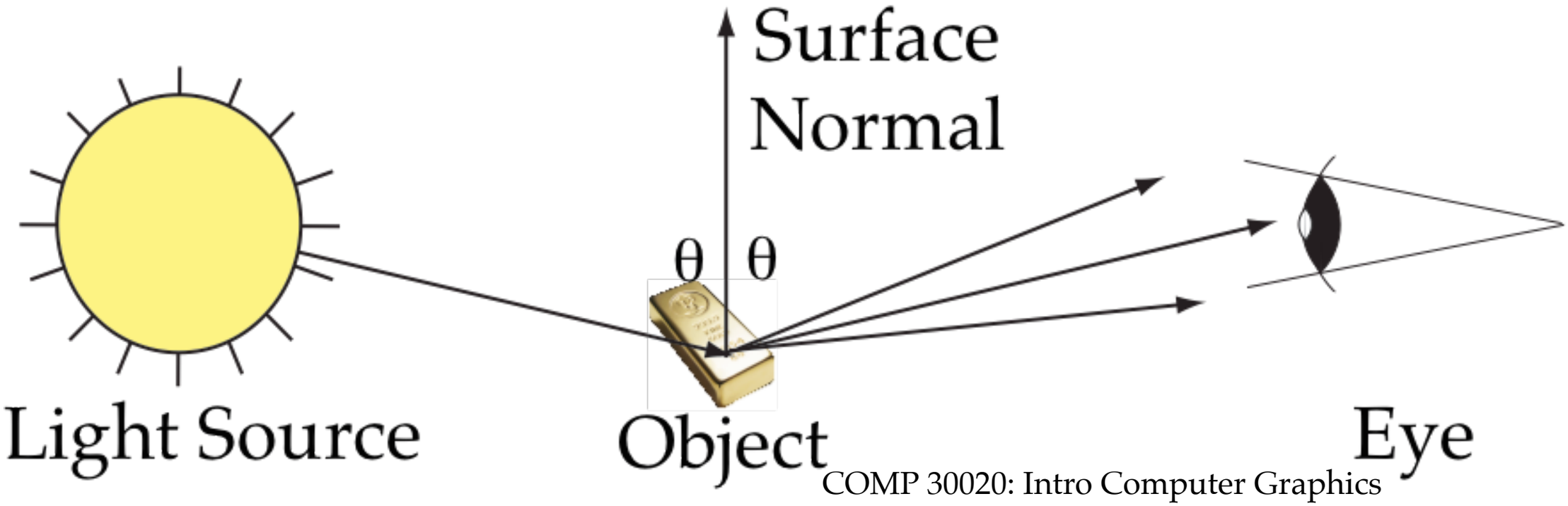
Reflection

- A photon strikes a surface
- By Newtonian mechanics, it bounces
- Angle of incidence = angle of reflection



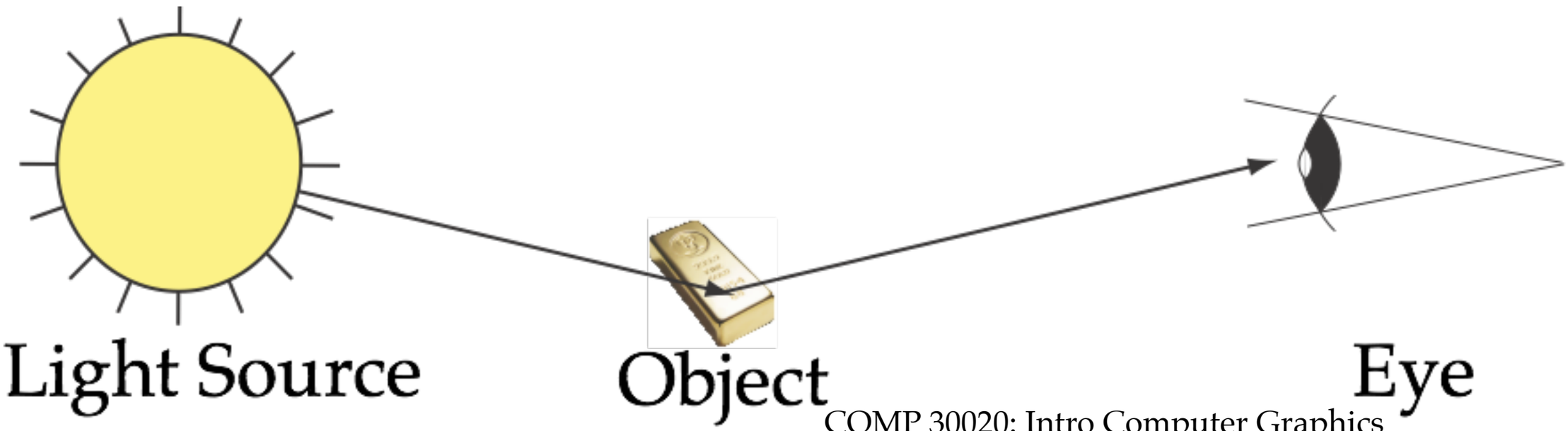
More Reflection

- Few surfaces are entirely smooth
- Reflection angle is somewhat random
- We'll come back to this



Absorption

- Photons can be absorbed by atoms
- Energy is transferred to atom
- Basis of vision, photosynthesis, warmth

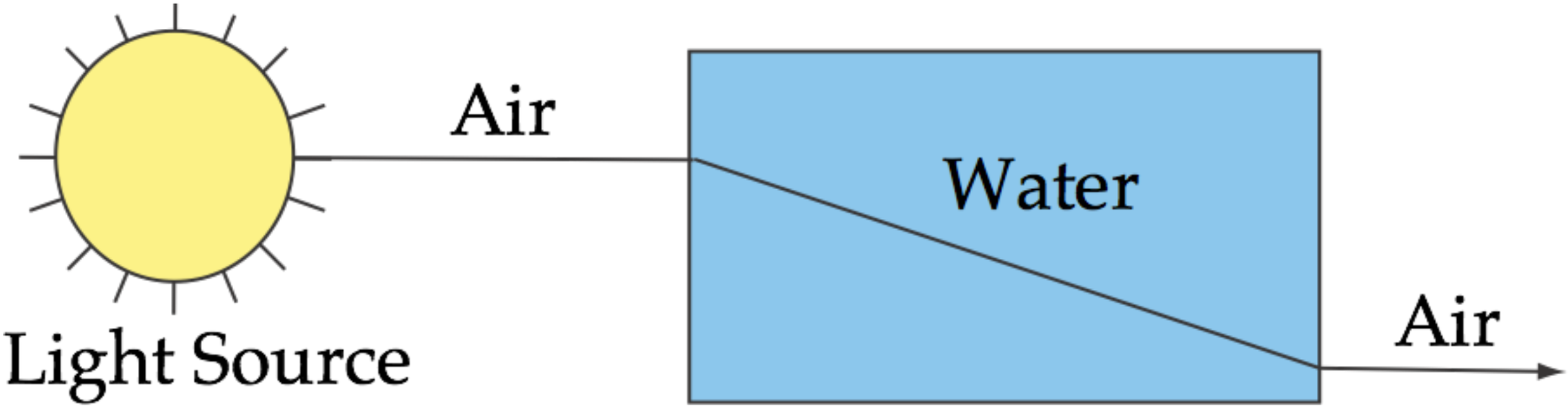


Absorption & Colour

- Why is this buttercup yellow?
- It reflects mostly yellow photons

Refraction

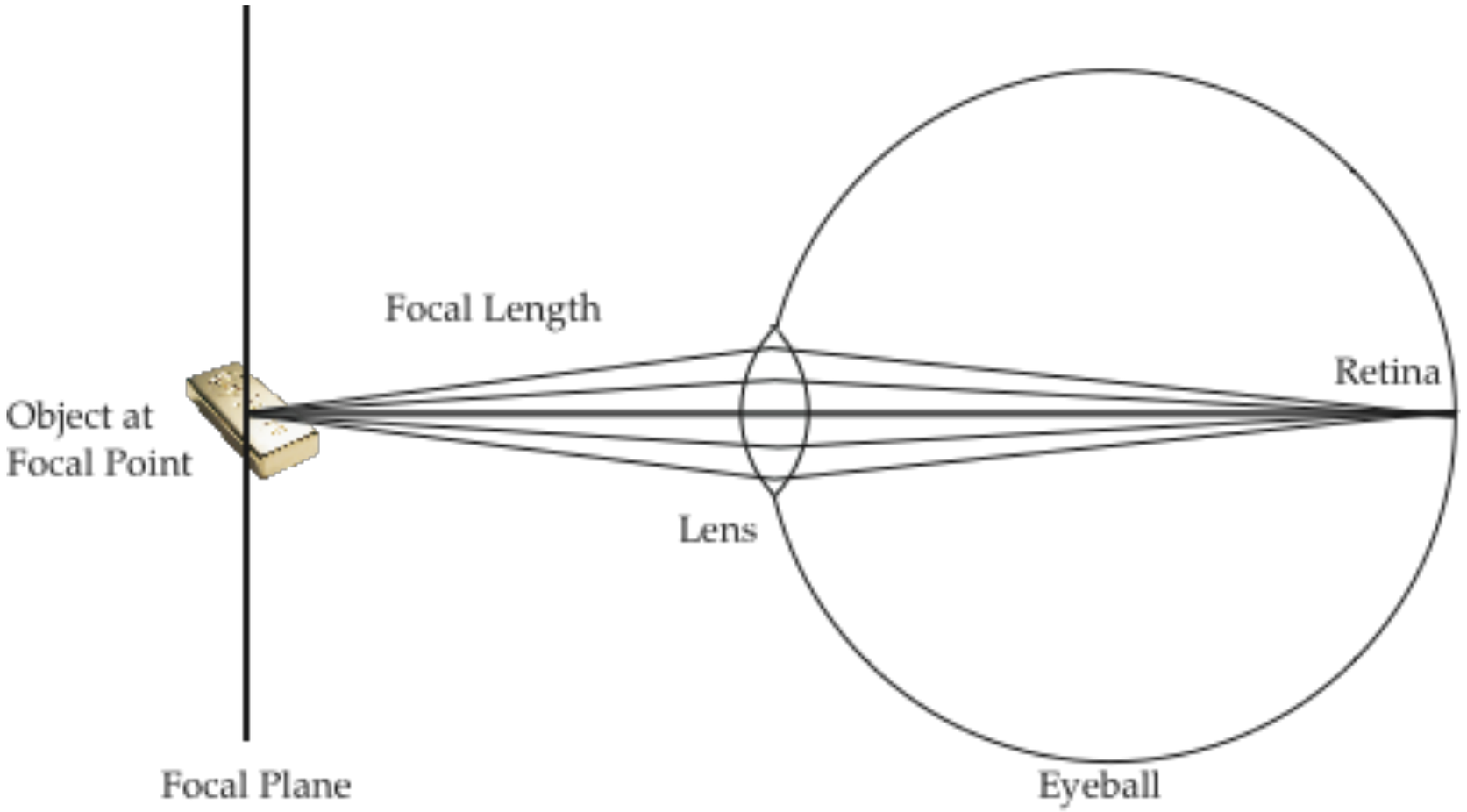
- Light bends when it changes media
- How much depends on the material
- And on the wavelength of the light



An Example

- Note the reflections & refractions

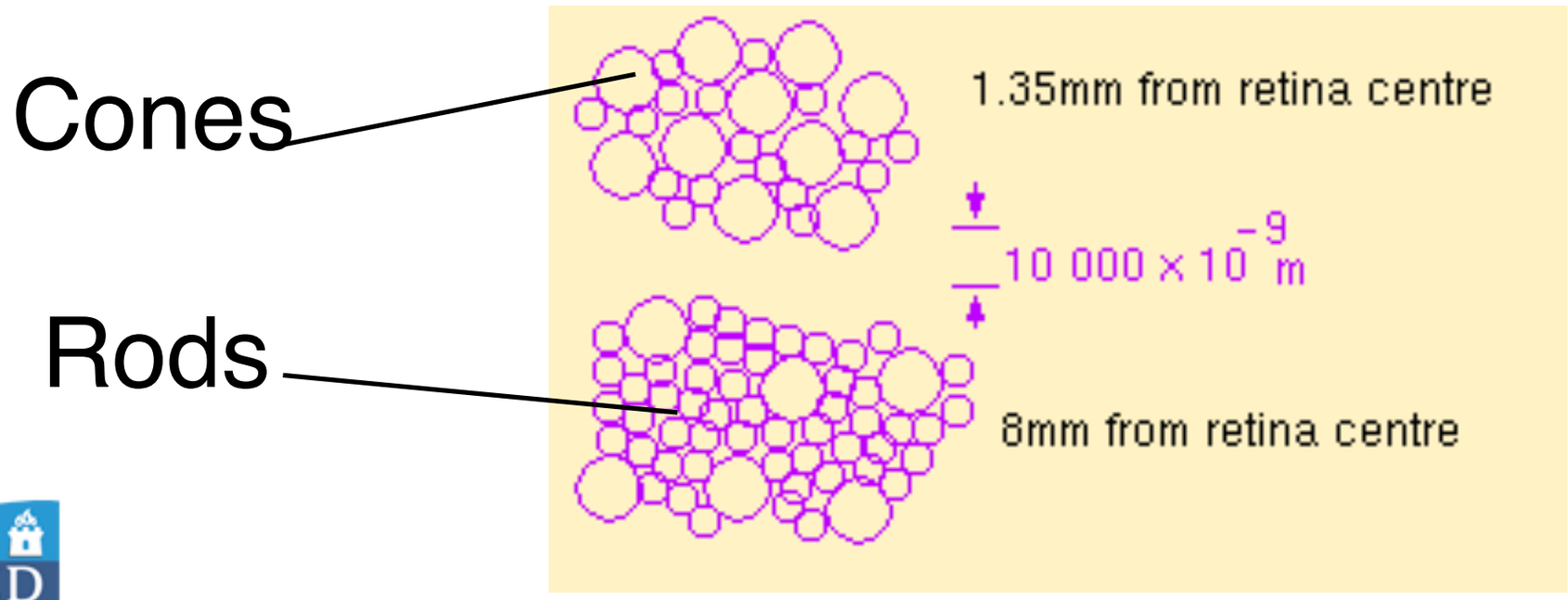
Human Vision



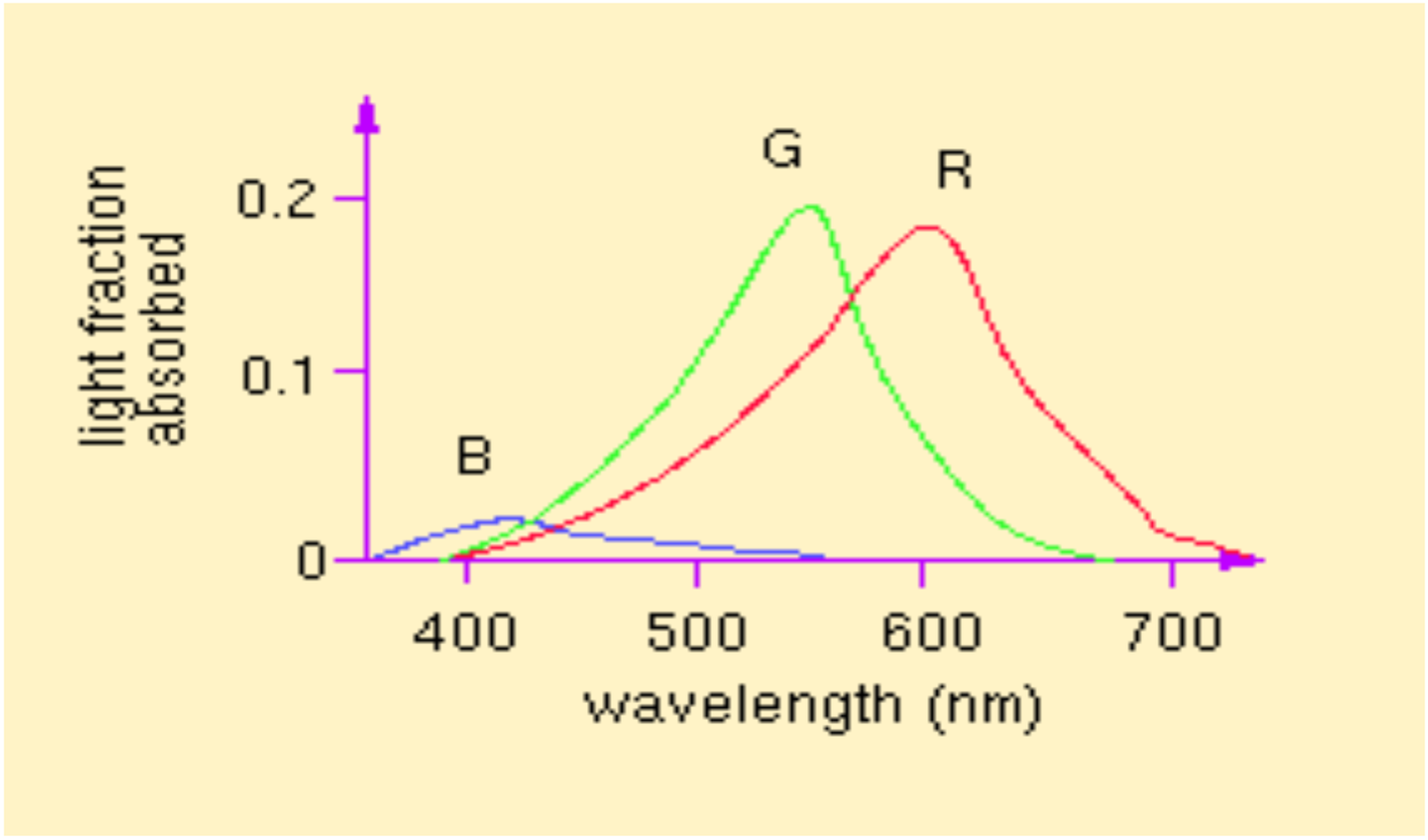
- Lens focusses light on retina
- Objects in focal plane are seen clearly

The Retina

- A patchwork of light-sensitive cells
  - Rods: low light conditions (B/W)
  - Cones: ordinary conditions (colour)

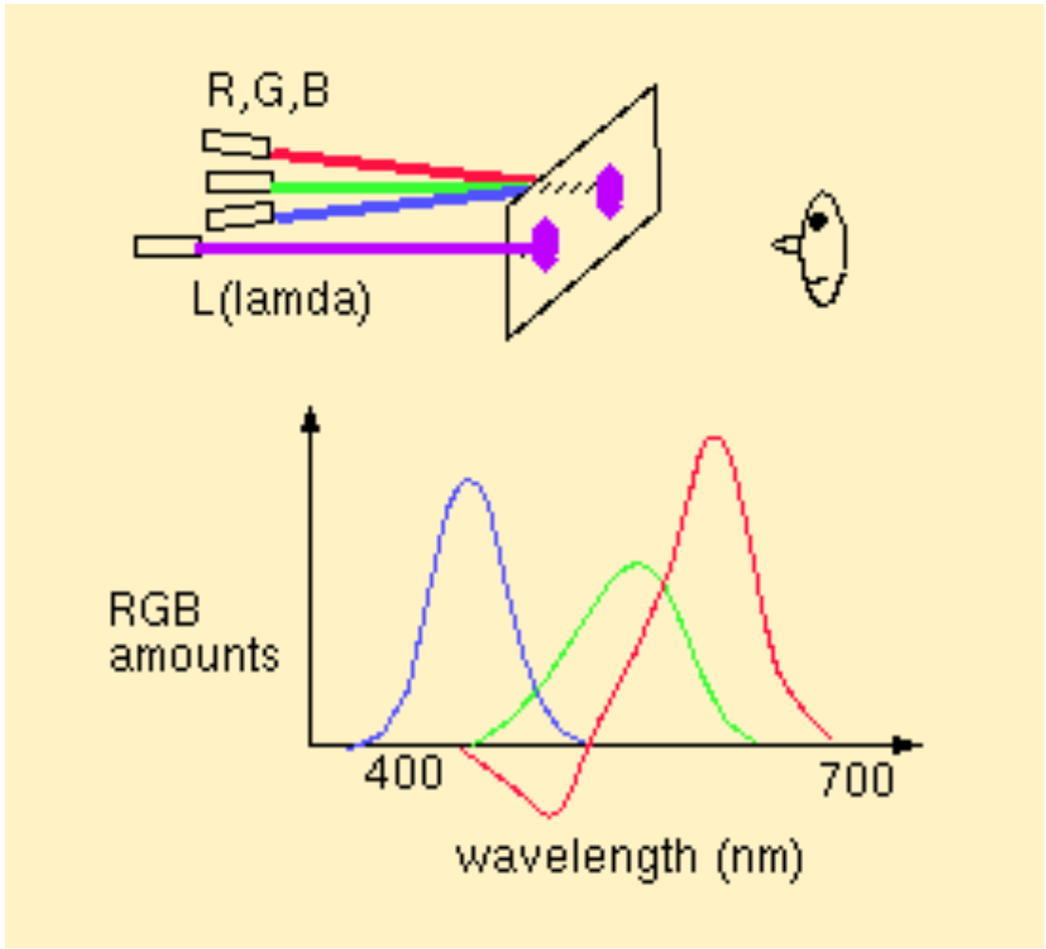


Cones & Colour



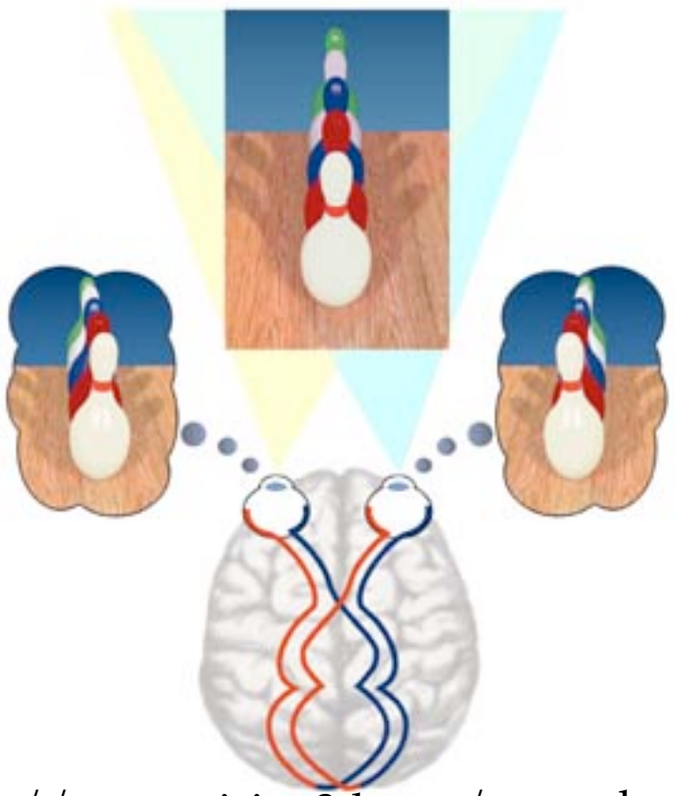
- Three types of cones
- Respond to different wavelengths:

Tri-stimulus Theory



- Mix R, G & B to get any colour desired
- Human can't tell the difference

Stereoscopic Vision



http://www.vision3d.com/stereo.html

- Two images at slightly different places
- Allow the brain to perceive depth
- The Brian also uses other tricks such as Focus and motion parallax

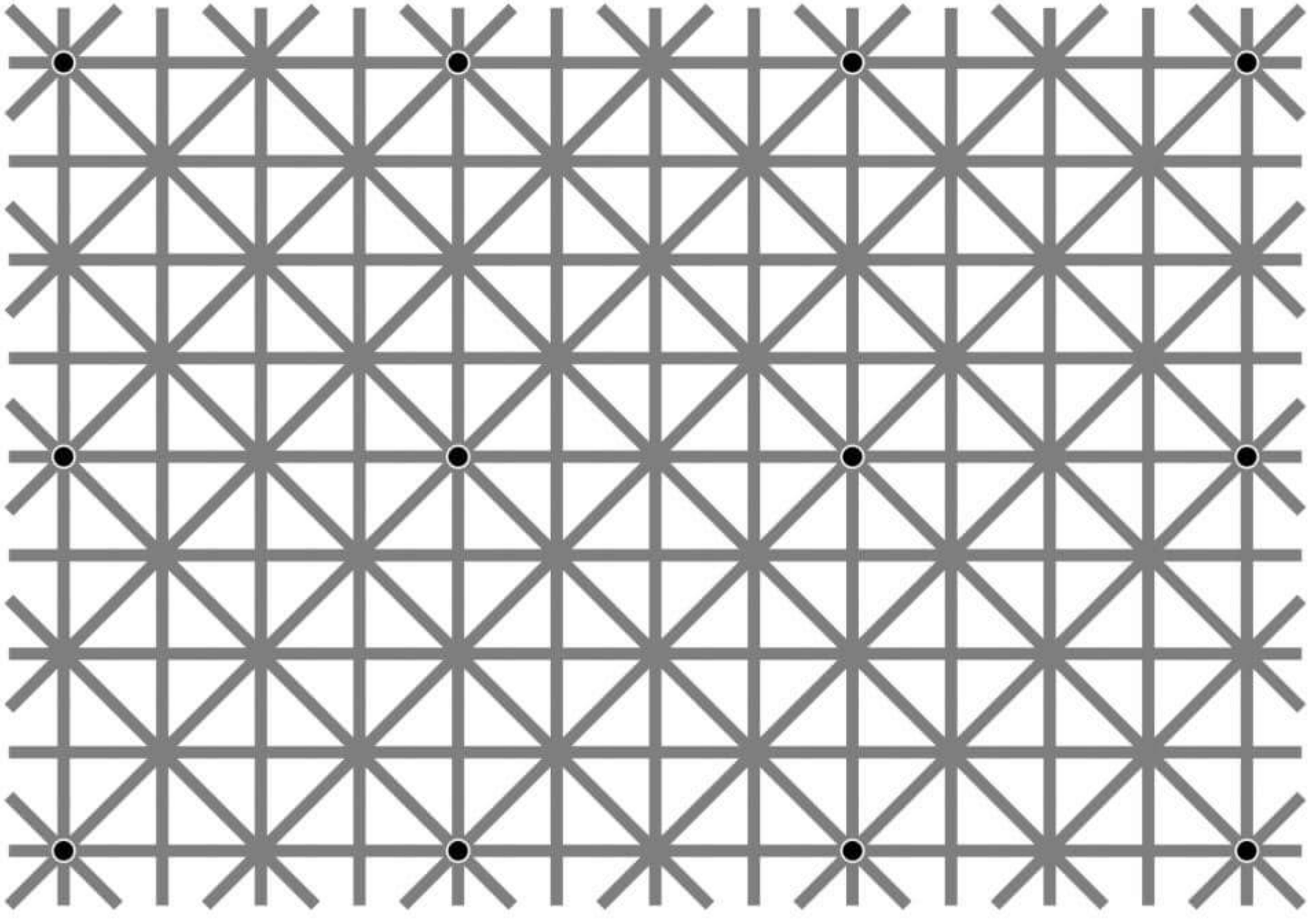
Limitations of the Eye

- The eye sees images at 24 Hz (+/-)
- Therefore, we render at 30 fps or better
  - On CRTs, double that to at least 60
  - We'll see why later
- This is why < 30 fps looks jerky

Limitations of the Eye

Only every see small parts of an image/World at the same time.

This is why so many optical illusions are possible. The image below is static



Rendering Images

- How can we fake reality?
  - make the eye think something's there
- Goes back to the Renaissance
  - Brunelleschi, Alberti, da Vinci, &c.
- Apply mathematics to the world

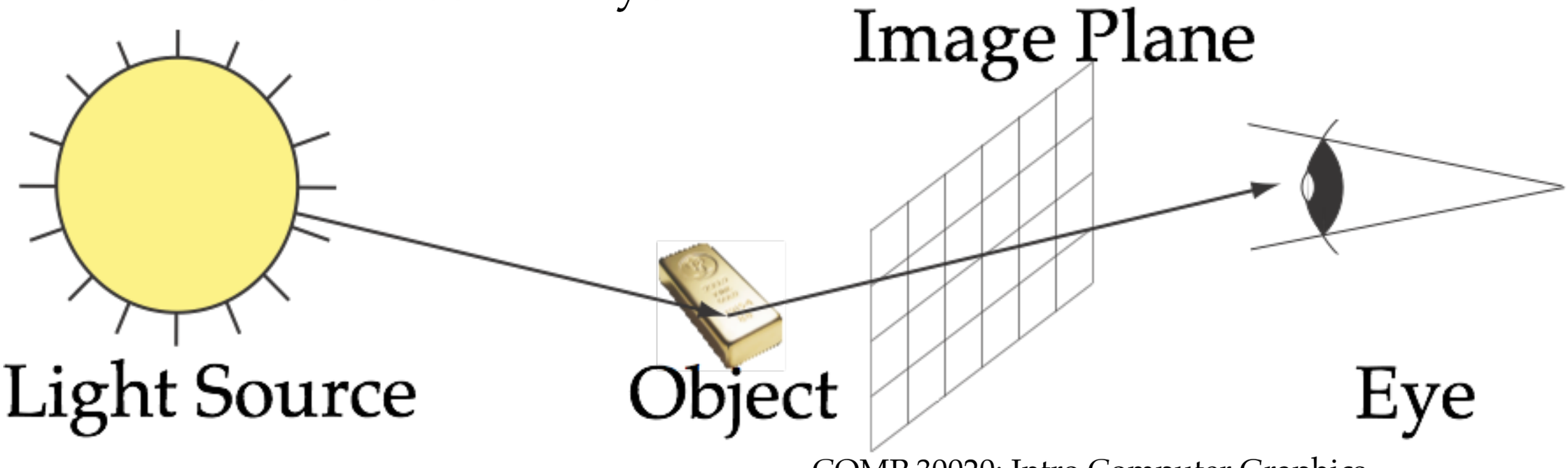
Basic Rendering

- Draw coloured image
  - Painters: copy what you see
  - CG: invent what you see
- Put in front of the eyes
- Look at it

Alberti's Window

Italian artist and architect Leon Battista (1435) described a method for creating realistic, three-dimensional representations on a two-dimensional canvas. He envisioned a painting as if viewed through an imaginary window, framing the scene for the viewer.

- Place glass sheet in front of the eye
- Draw what you see on the glass sheet
- Then take the sheet with you



Ideal Image

- We want to measure light everywhere
  - at each point (x,y) on retina
- So light intensity (brightness) is a function
  - I(x,y): intensity at retina (or A.'s window)
- Intensity is a continuous function
- Sadly, computers are discrete machines

Pixel Image

- Instead of a function, use an array

Digital Images

- A digital image is an array of intensities
- Assume image is 8-bit greyscale
  - each pixel is 0 ... 255 (Y in colour space)
- Each value is a sample point
  - $p_{ij} = I\left(x_i, y_j\right)$
  - actually an integral

$$p_{ij} = \int_{x_i - \frac{1}{2}\Delta x}^{x_i + \frac{1}{2}\Delta x} \int_{y_j - \frac{1}{2}\Delta y}^{y_j + \frac{1}{2}\Delta y} I(x,y)\, dy\, dx$$

Rendering Images

- Three popular/basic ways of doing this:
  - Raytracing (Renderman /movies)
  - Projective rendering (OpenGL)
  - Image based (Street View)

Ray Tracing (RenderMan, used in Movies)

Overview:
Ray tracing is a rendering technique that simulates the way light rays interact with objects in a scene to produce highly realistic images. It traces the path of light rays from the camera through each pixel in the scene and calculates how these rays reflect, refract, and interact with surfaces.

How It Works:
- A ray is cast from the camera (or eye) through a pixel and into the scene.
- The algorithm traces the ray's path as it intersects objects.
- When a ray hits an object, the shading and lighting of that point are calculated based on factors such as the material of the object, the light sources, and other objects in the scene (for shadows, reflections, and refractions).
- This process can involve recursive rays to handle reflections, transparency, and global illumination effects like caustics.

Applications:
- Movies and Animation: Ray tracing is widely used in Hollywood movies, especially in high-end rendering engines like Pixar's RenderMan or Arnold. The ability to simulate complex lighting and shadows is perfect for creating realistic visual effects and animations.
- Photorealistic Rendering: Ray tracing is preferred when high levels of realism are required, as it can accurately model reflections, refractions, and soft shadows.

Advantages:
- Produces highly realistic images.
- Handles reflections, refractions, shadows, and global illumination naturally.

Disadvantages:
- Computationally expensive and slower than other methods.
- Typically requires more time for rendering, especially for complex scenes.

Projective Rendering (OpenGL)

Overview:
Projective rendering (also known as rasterization) is a technique used to project 3D objects onto a 2D image plane by transforming the 3D coordinates into screen coordinates. This is the traditional method used by OpenGL, DirectX, and other real-time graphics APIs, particularly in games and interactive applications.

How It Works:
- 3D models are represented using a mesh of triangles (or other polygons).
- The geometry of the scene is projected onto the screen using matrix transformations (e.g., model, view, and projection matrices).
- The scene is then rasterized, meaning the 3D objects are converted into 2D images by shading each triangle, using interpolation to fill in pixel values between vertices.
- Lighting is handled using local illumination models (e.g., Phong or Gouraud shading), which approximate how light interacts with surfaces.

Applications:
- Real-Time Rendering: Projective rendering is the go-to method for real-time applications like video games, virtual reality (VR), and interactive simulations. It is fast and can handle complex scenes efficiently.
- OpenGL/DirectX: Used extensively in hardware-accelerated rendering pipelines in graphics engines like Unreal Engine, Unity, and OpenGL.

Advantages:
- Extremely fast and suitable for real-time applications.
- Efficient use of hardware resources like GPUs.

Disadvantages:
- Limited realism compared to ray tracing.
- Does not natively support global illumination, reflections, or complex lighting phenomena without additional techniques like shadow mapping or baked lighting.

Image-Based Rendering (IBR)

Overview:
Image-Based Rendering (IBR) is a technique that synthesizes new views of a scene using existing images, rather than relying on 3D models. It creates realistic views by interpolating between multiple photographs or previously rendered images.

How It Works:
- IBR uses photographs or previously rendered frames as input.
- The images are mapped onto simple geometry (e.g., planes) and transformed to match the camera's perspective.
- The technique often includes methods like texture mapping, view interpolation, or light field rendering to generate new views.

Applications:
- Virtual Tours and VR: IBR is used in applications like Google Street View and virtual reality (VR) experiences, where real-world environments are displayed with minimal 3D modeling.
- Non-Interactive Rendering: IBR can be used when capturing highly detailed environments where modeling every detail in 3D would be difficult or inefficient.

Advantages:
- Very realistic results since it uses real images.
- Efficient for specific applications where pre-rendered or real-world imagery is available.

Disadvantages:
- Limited interactivity and control over the scene compared to fully 3D techniques.
- Not suitable for dynamic scenes or environments with complex, moving objects.