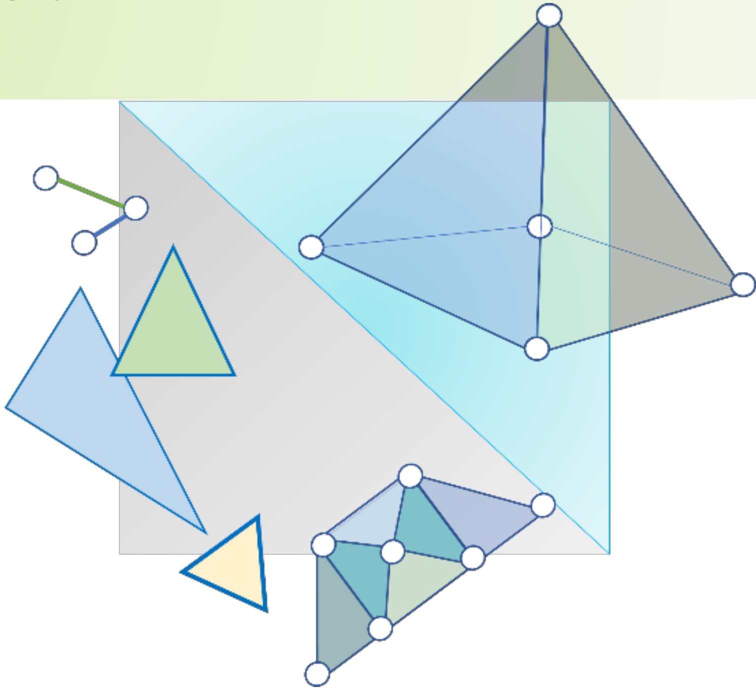
CITS3003 Graphics & Animation



Introduction
and
Admin Matters

Content

- Introduction to the Unit
- Introduction to Computer Graphics
- Introduction to OpenGL

Teaching Team

Naeha Sharif
Unit Coordinator & Lecturer

Room 1.05, First Floor
CSSE building

Consultation Hour
11:30am - 12:30pm Wednesdays

Email:
cits3003-csse@uwa.edu.au

Expect a response within 48-72hrs

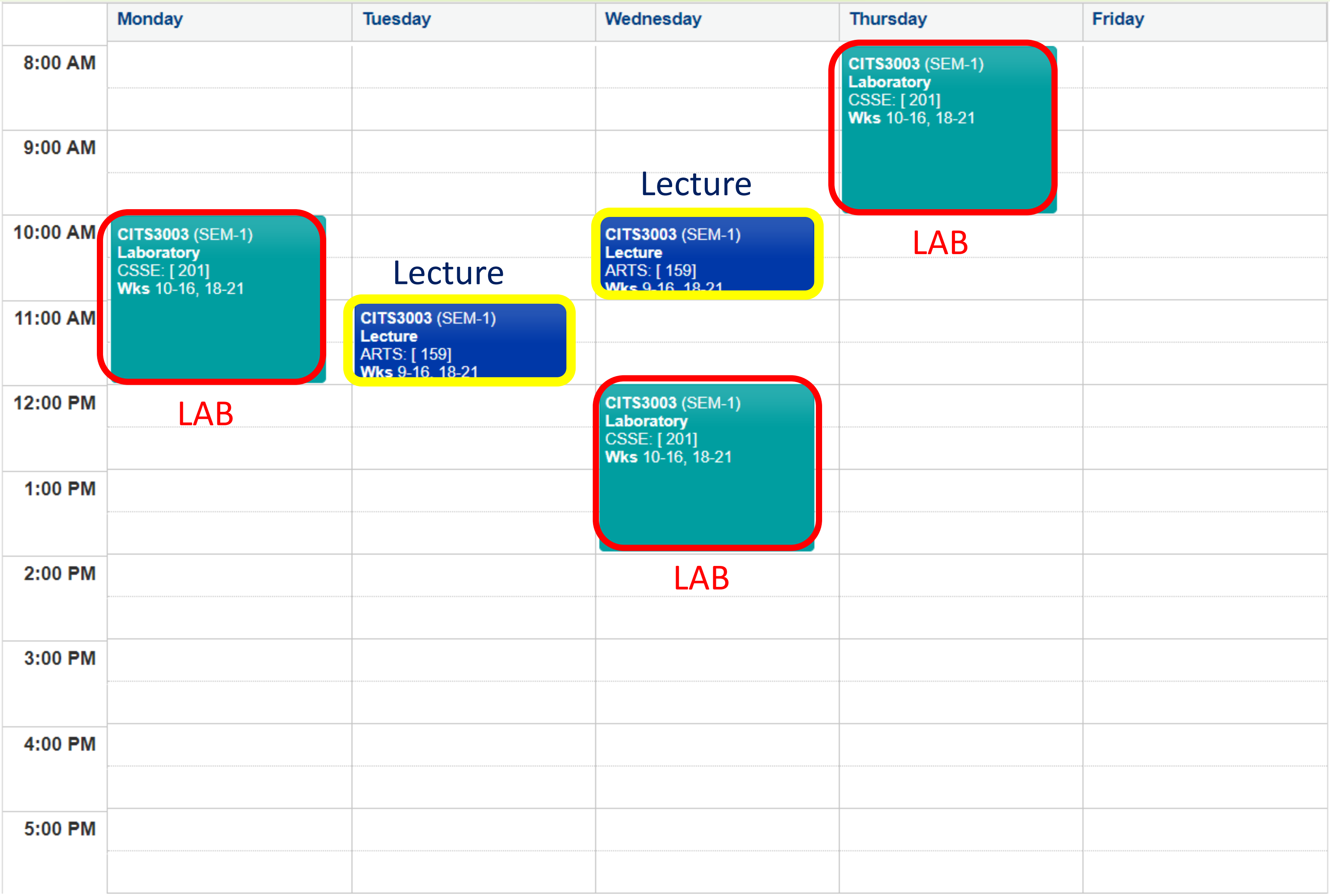
Khanh **Tyler** **Shane**

Facilitators

Labs
- Monday: 10:00am - 12:00pm
- Wednesday: 12:00pm - 2:00pm
- Thursday: 8:00am - 10:00am

Note: All CITS3003 labs will be conducted in CSSE-201 this semester

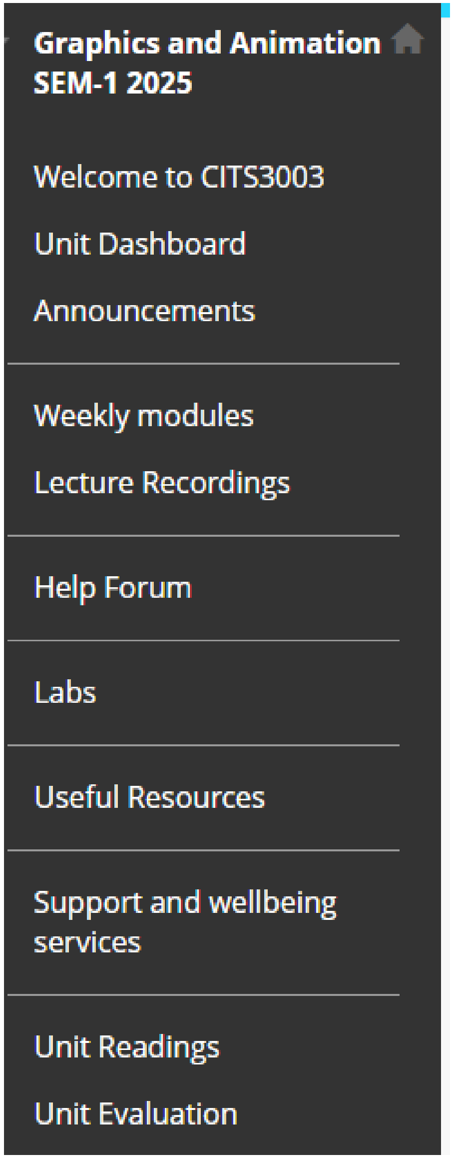
Timetable



https://timetable.applications.uwa.edu.au/draft.html

Other Admin Matters

- Lectures and lab material will be on LMS
  - Check regularly for announcements and updates
  - Lectures uploaded every teaching week
- Check the useful resources tab
- Help forum is available on LMS. All queries related to labs/project should be posted there. Students are encouraged to help each other. However, sharing solutions/partial solutions of assessments is not allowed.
- Email (cits3003-csse@uwa.edu.au) should only be used for issues which cannot be discussed on the help forum



Other Admin Matters

Prerequisites

- CITS1401 Computational Thinking with Python
- or CITS2002 Systems Programming
- or CITS2401 Computer Analysis and Visualisation
- or CITX1401 Computational Thinking with Python

Project and Labs

- Labs will be running every week, starting from week#2.
- Lab sheets will be provided (along with the solutions) on LMS
- Lab#1-5 are not assessed but it is important to complete them to be able to complete the project.
- Lab#6 is assessed and will be released in week#07
- Project will be released in week#07

Assessments

- The assessments will consist of:
  - 15%: Mid-semester test (week 07)
  - 20%: Project (due in week 12)
  - 5%: Lab#06 (due in week 12)
  - 60%: Final exam

Mid-semester test and Final exam will be conducted in a face-to-face format (paper-based)

Other matters

Some general advice:

- Attend lectures regularly
- Attempt all the lab excercises in a timely manner
- Consult supplementary material for deeper understanding
- Start working on the project as soon as it is released
- Seek help early

Commendations:

- Highest total score
- Class participation
- Creativity
- Help forum

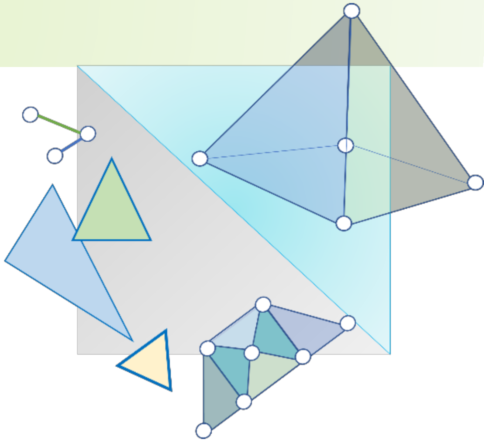
Breakdown of Lectures

1. Introduction & Image Formation
2. Programming with OpenGL
3. OpenGL: Pipeline Architecture
4. OpenGL: An Example Program
5. Vertex and Fragment Shaders 1
6. Vertex and Fragment Shaders 2
7. Representation and Coordinate Systems
8. Coordinate Frame Transformations
9. Transformations and Homogeneous Coordinates
10. Input, Interaction and Callbacks
11. More on Callback
12. Hidden Surface Removal

Mid-semester Test

13. Computer Viewing
14. Shading
15. Shading Models
16. Shading in OpenGL
17. Texture Mapping
18. Texture Mapping in OpenGL
19. Hierarchical Modelling
20. 3D Modelling: Subdivision Surfaces
21. Animation Fundamentals, Quaternions and Skinning
22. Guest Lecture
23. Tutorial

Computer Graphics



Computer graphics is a field that is concerned with all aspects of creating and manipulating visual content using a computer.

- hardware tools
- Software tools

Image Courtesy: Disney New Lion King

Computer Graphics

Applications

Displaying simulations

Computer Games

Image source

Scientific visualisations

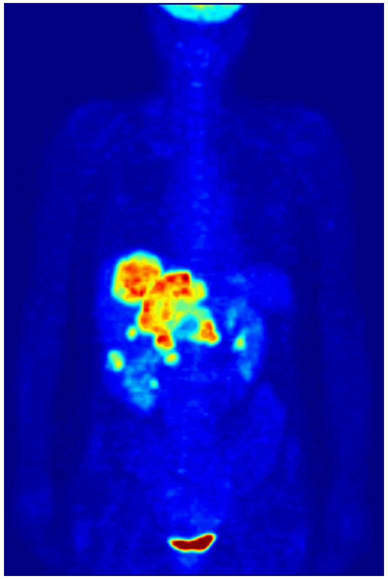


Image source

Movies

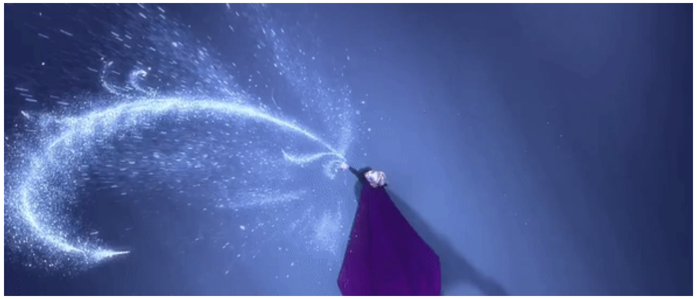


Image source

Virtual Reality

Image source

About CITS3003

- Computer Graphics has many aspects:
  - Computer Scientists create graphics programs and tools (e.g., Blender, Maya, photoshop)
    - Includes C/C+, shader programming, maths, linear algebra, etc.,
  - Artists use Computer Graphics packages to create photorealistic/creative pictures – (does not involve maths or programming)

About CITS3003

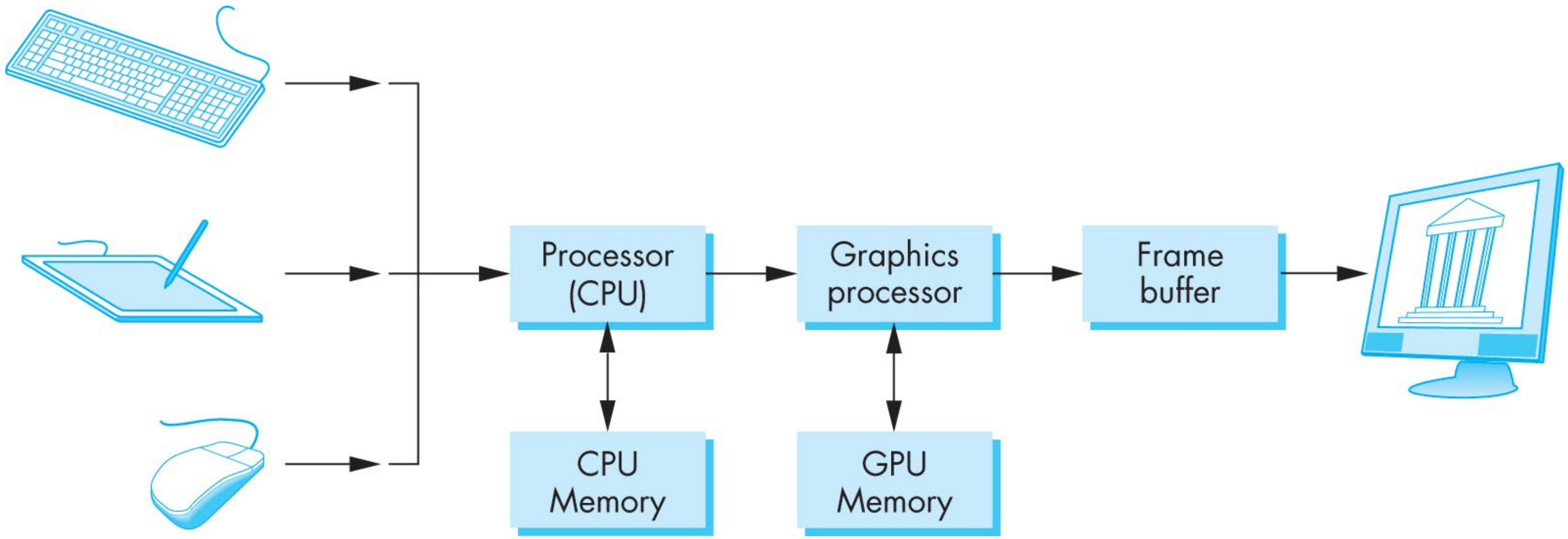
CITS3003 teaches fundamentals of computer-generated three-dimensional graphics and animation.

- It introduces OpenGL (Graphics library) for writing interactive graphics programs.

CITS3003 is:

- not about using software packages like Photoshop, Maya, GIMP
- not a comprehensive course on OpenGL, as only limited parts of the library are covered
- not a game development unit

A Graphics System



Output device

Input devices

Image formed in frame buffer

Rendering

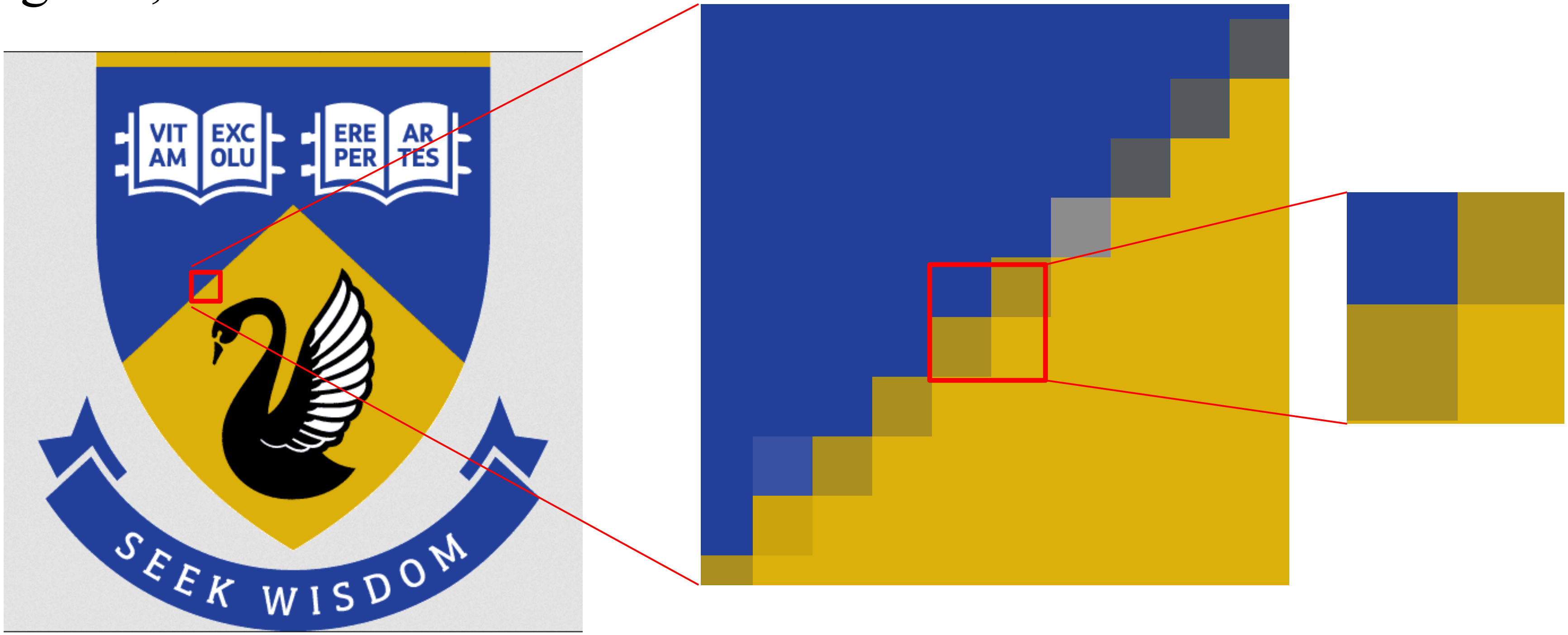
In general, one of the basic tasks of 3D graphics is *producing 2D images* of the three-dimensional world.

Fundamentally, ***rendering*** is a process that takes as its input a set of objects and produces as its output an array of pixels.

- the whole process of producing an image is referred to as rendering the scene.

Raster Image

A raster image is simply a 2D array that stores the *pixel value* for each pixel—usually a color stored as three numbers, for red, green, and blue.



Color Images

- Color Image
  - Has perceptional attributes of hue, saturation, and lightness

<u>Hue</u>
another word for color
(wavelength dependent)

<u>Saturation (Chroma)</u>
the intensity or purity of hue
(100% pure = no addition of gray)

<u>Lightness (Value)</u>
relative degree of black/white

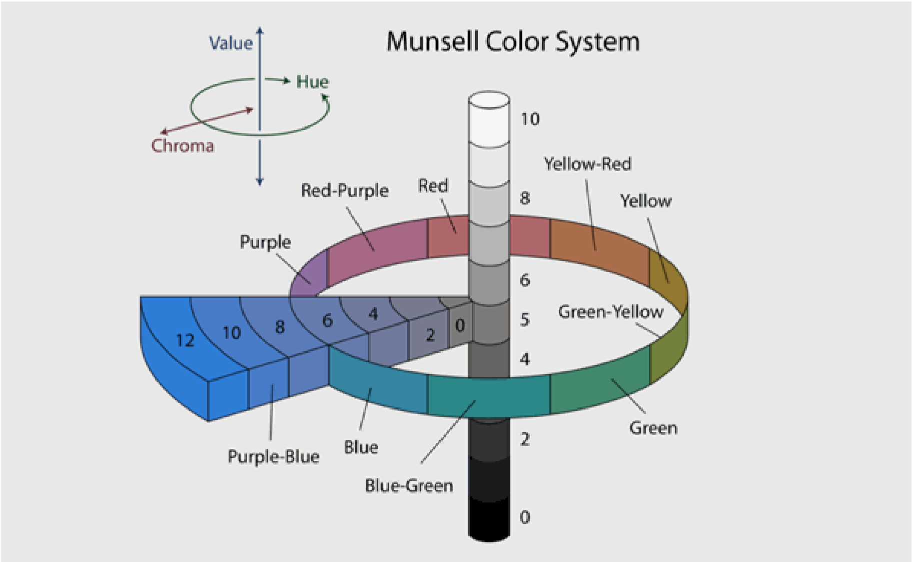
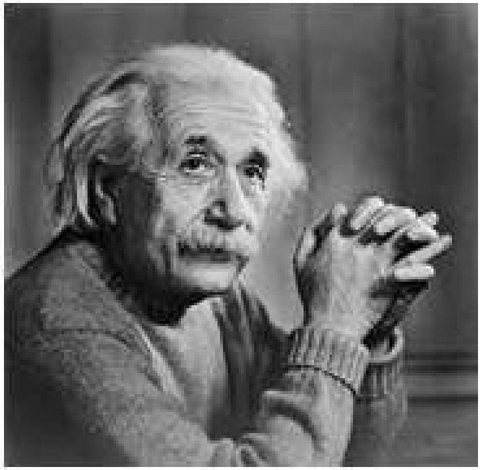


Image from (https://vanseodesign.com/web-design/hue-saturation-and-lightness/)

Luminance Images

- Luminance Image
  - Monochromatic
  - Values are gray levels
  - Analogous to working with black and white film or television



Introduction to Image Formation

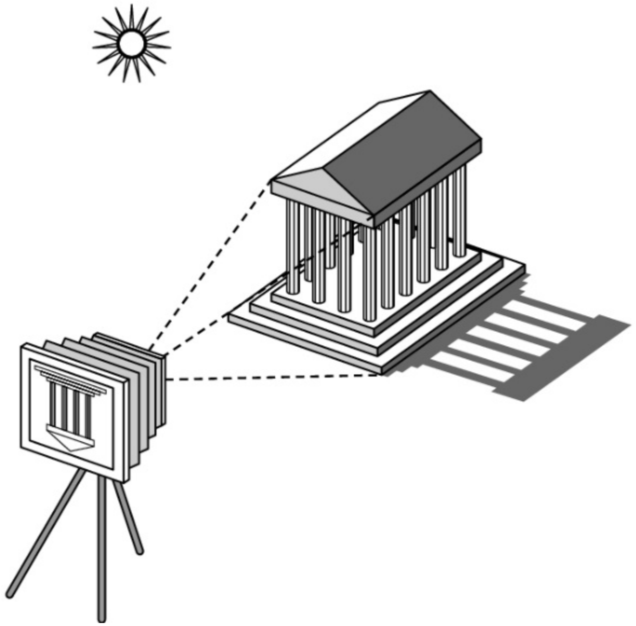
Image credits

Image Formation

- In computer graphics, we form/create images using a process analogous to how images are formed by physical imaging systems
  - Cameras
  - Microscopes
  - Telescopes
  - Human visual system

Elements of Image Formation

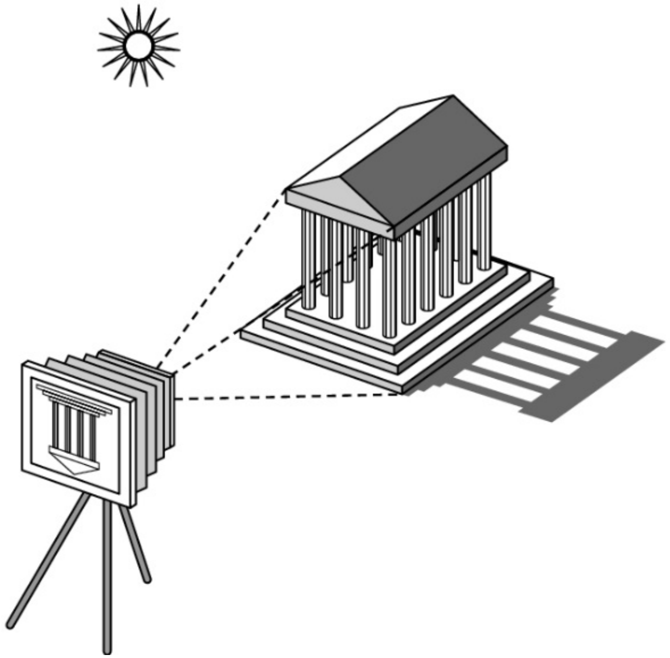
1. Objects
2. Viewer
3. Light source(s)



*Note the **independence** of the objects, the viewer, and the light source(s)*

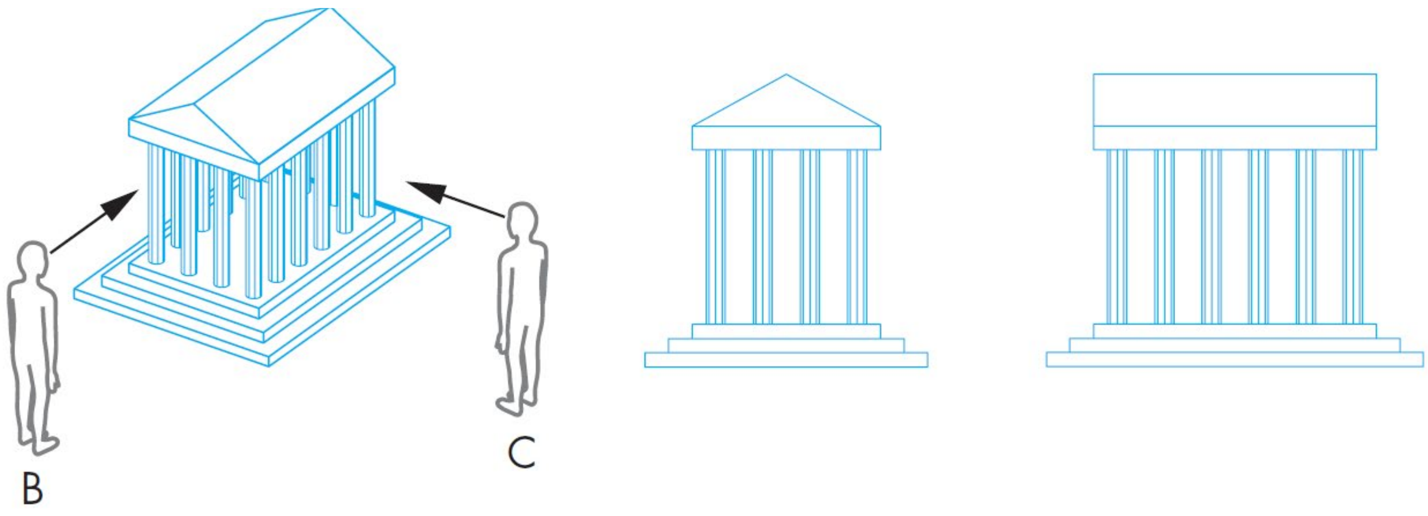
Objects

- A set of locations (vertices) in space is sufficient to define or approximate most objects



Viewer

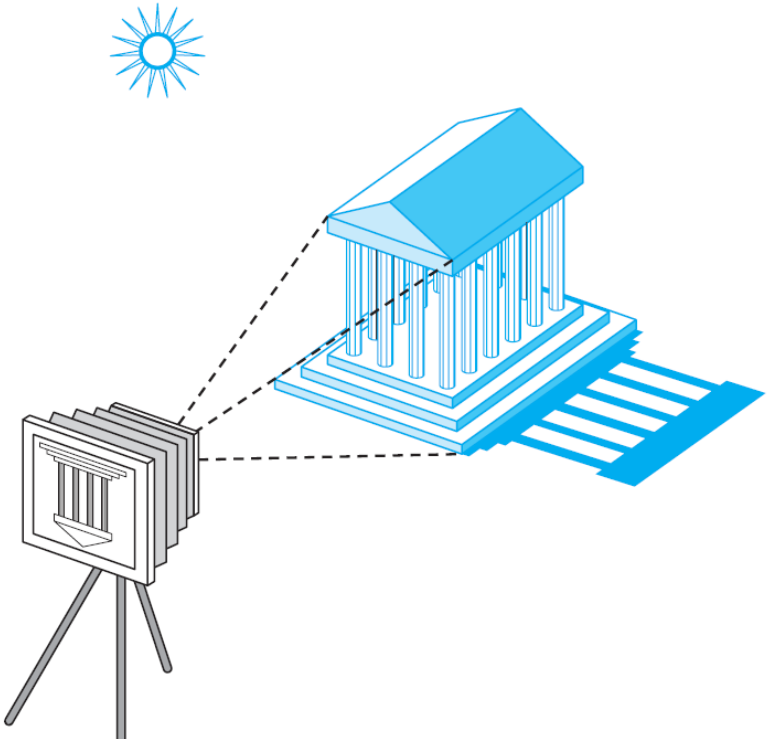
- To form an image, we must have someone or something that is viewing our objects, be it a human, a camera, or a digitizer. It is the **viewer** that forms the image of our objects.



Light and Images

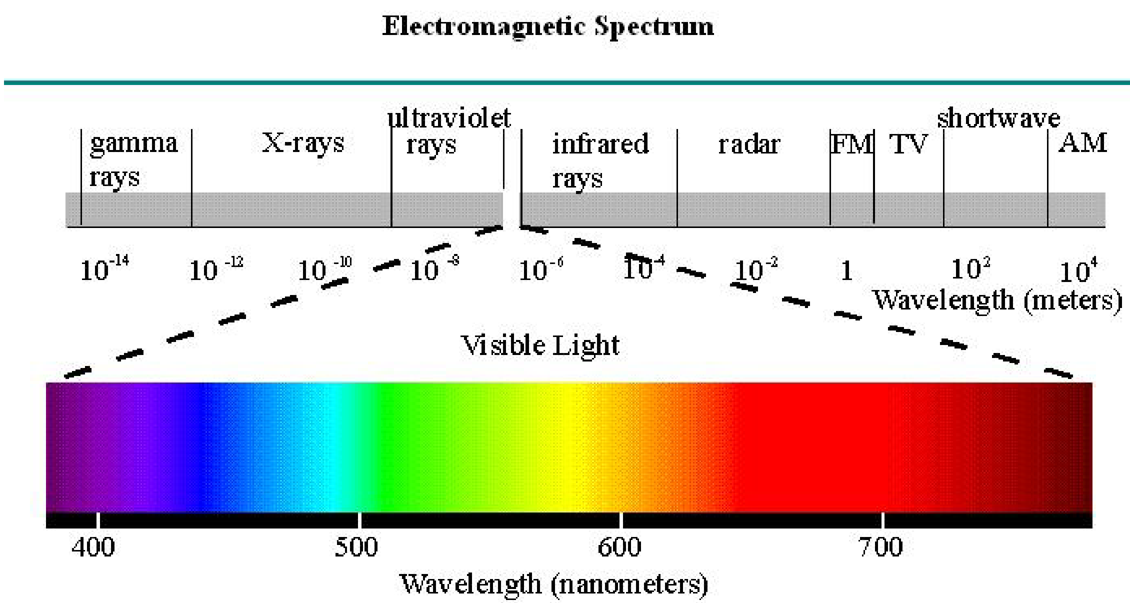
- If there were no light sources, the objects would appear dark
- Light is the part of the electromagnetic spectrum that causes a reaction in our visual system

The details of the interaction between light and the surfaces of the object determine how much light enters the camera.

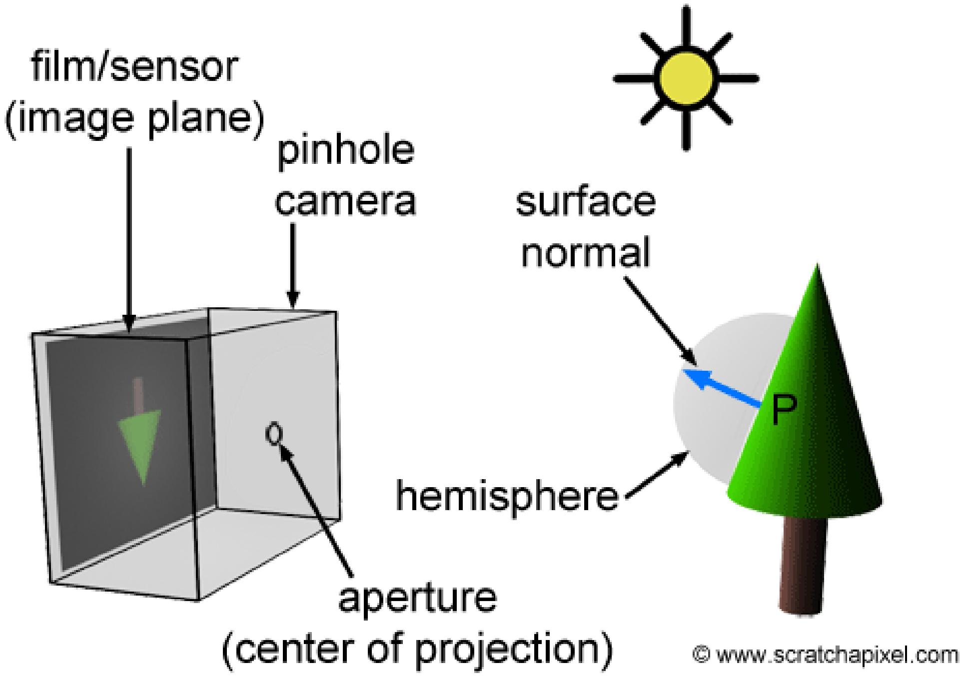


Light and Images

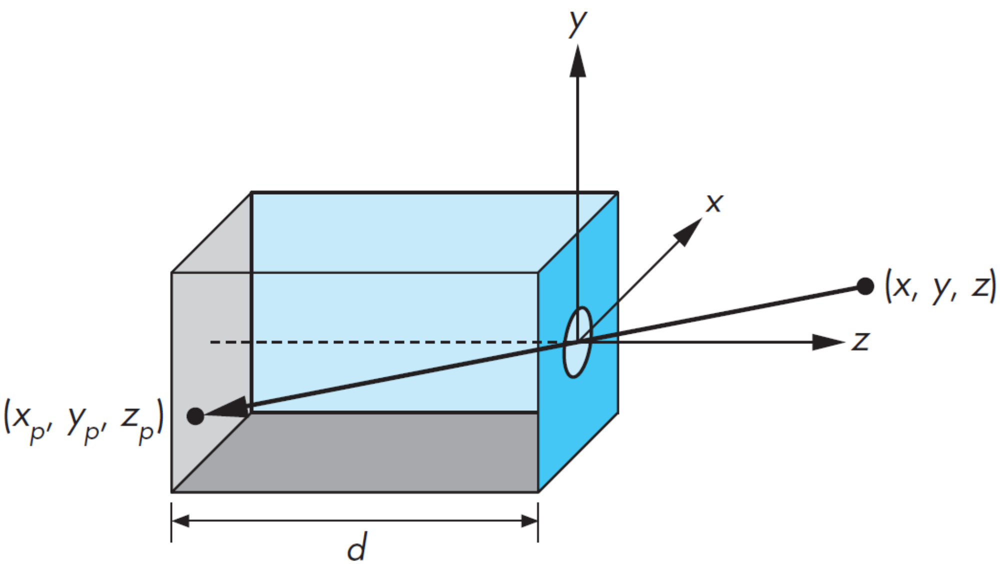
- If there were no light sources, the objects would appear dark
- Light is the part of the electromagnetic spectrum that causes a reaction in our visual system
- Generally, these are wavelengths in the range of about 350-750 nm (nanometers)
  - Long wavelengths appear as reds and short wavelengths as blues

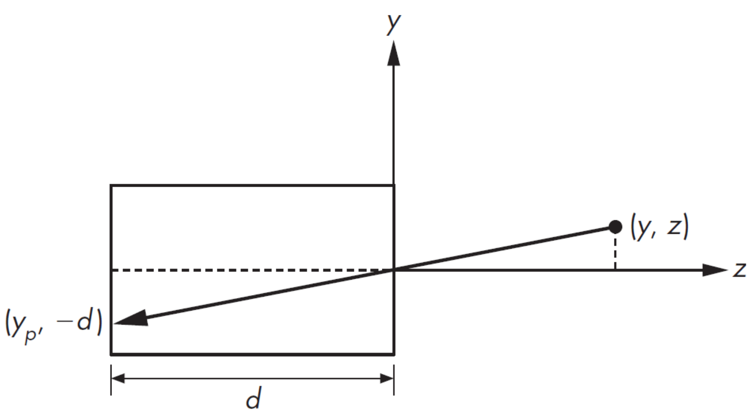


Imaging System



Pinhole Camera





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

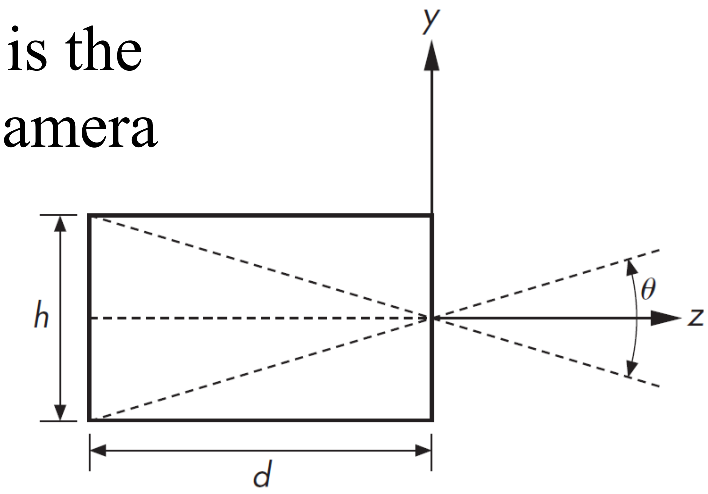
$$x_p/z_p = x/z \qquad y_p/z_p = y/z \qquad z_p = -d$$

- These are equations of simple perspective
- The point $(x_p, y_p, -d)$ is called the **projection** of the point (x, y, z).

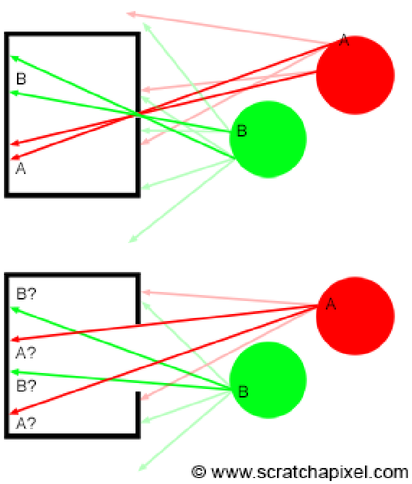
Pinhole Camera (cont..)

- The **field, or angle of view** of our camera is the angle made by the largest object that our camera can image on its film plane.

$$\theta = 2\tan^{-1}\frac{h}{2d}$$



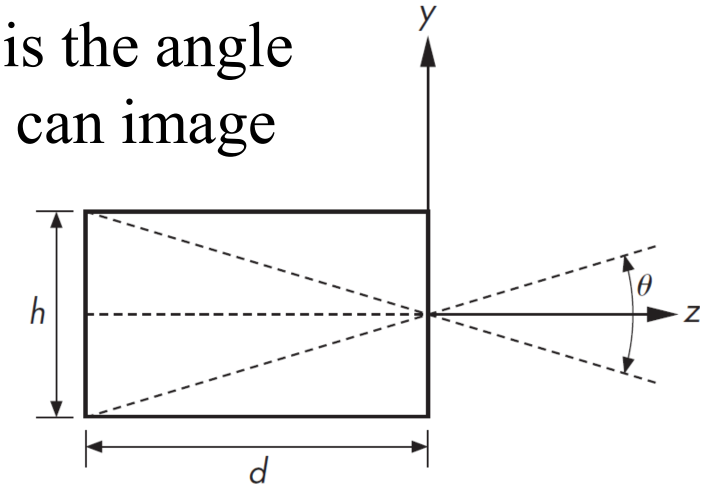
- The ideal pinhole camera has an infinite **Depth Of Field (DOF)**
  - DOF is the distance between the nearest and the farthest objects that are in acceptably sharp focus in an image



Learn more: https://www.scratchapixel.com/lessons/3d-basic-rendering/3d-viewing-pinhole-camera/how-pinhole-camera-works-part-1.html

Pinhole Camera (cont..)

- The **field, or angle of view** of our camera is the angle made by the largest object that our camera can image on its film plane.



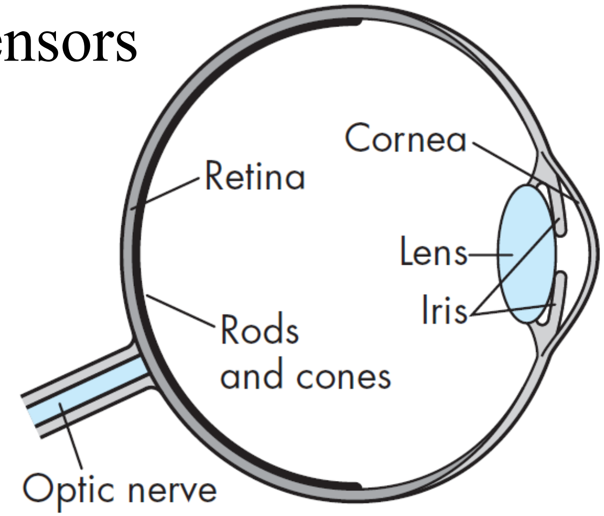
$$\theta = 2\tan^{-1}\frac{h}{2d}$$

- The pinhole camera has two disadvantages:
  – It admits only a single ray from a point source—almost no light enters the camera.
    - Long exposure time
  – The camera cannot be adjusted to have a different angle of view

Learn more: https://www.scratchapixel.com/lessons/3d-basic-rendering/3d-viewing-pinhole-camera/how-pinhole-camera-works-part-1.html

Human Visual System

- The human visual system has two types of sensors
  - Rods (up to 125M)
    - Monochromatic, night vision
  - Cones (6M+)
    - Color sensitive
    - Three types of cones
    - Only three values (the ***tristimulus*** values) are sent to the brain



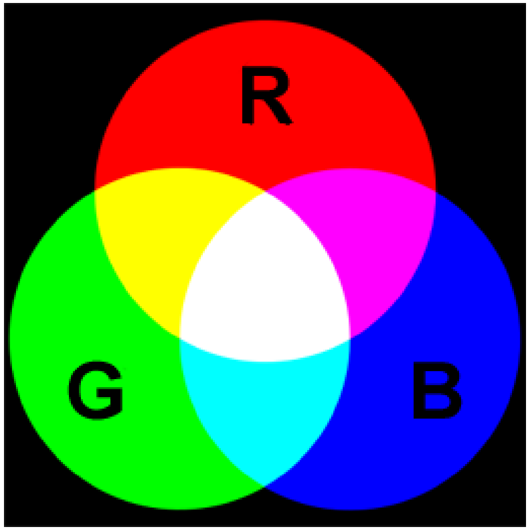
- That is, we need only match these three values

  →Need only three *primary* colors- trichromatic color vision

Color

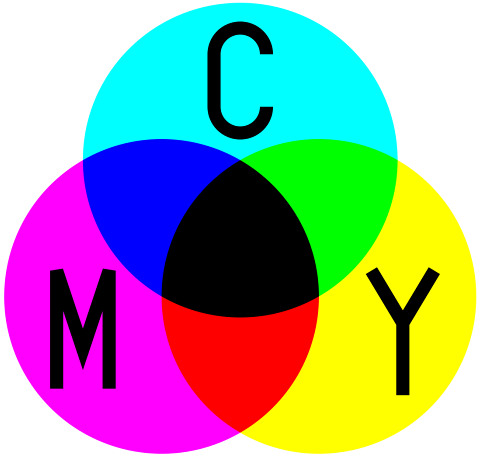
Additive color

- Form a color by adding amounts of three primaries
  - CRTs, projection systems, positive film
- Primaries are Red (R), Green (G), Blue (B)



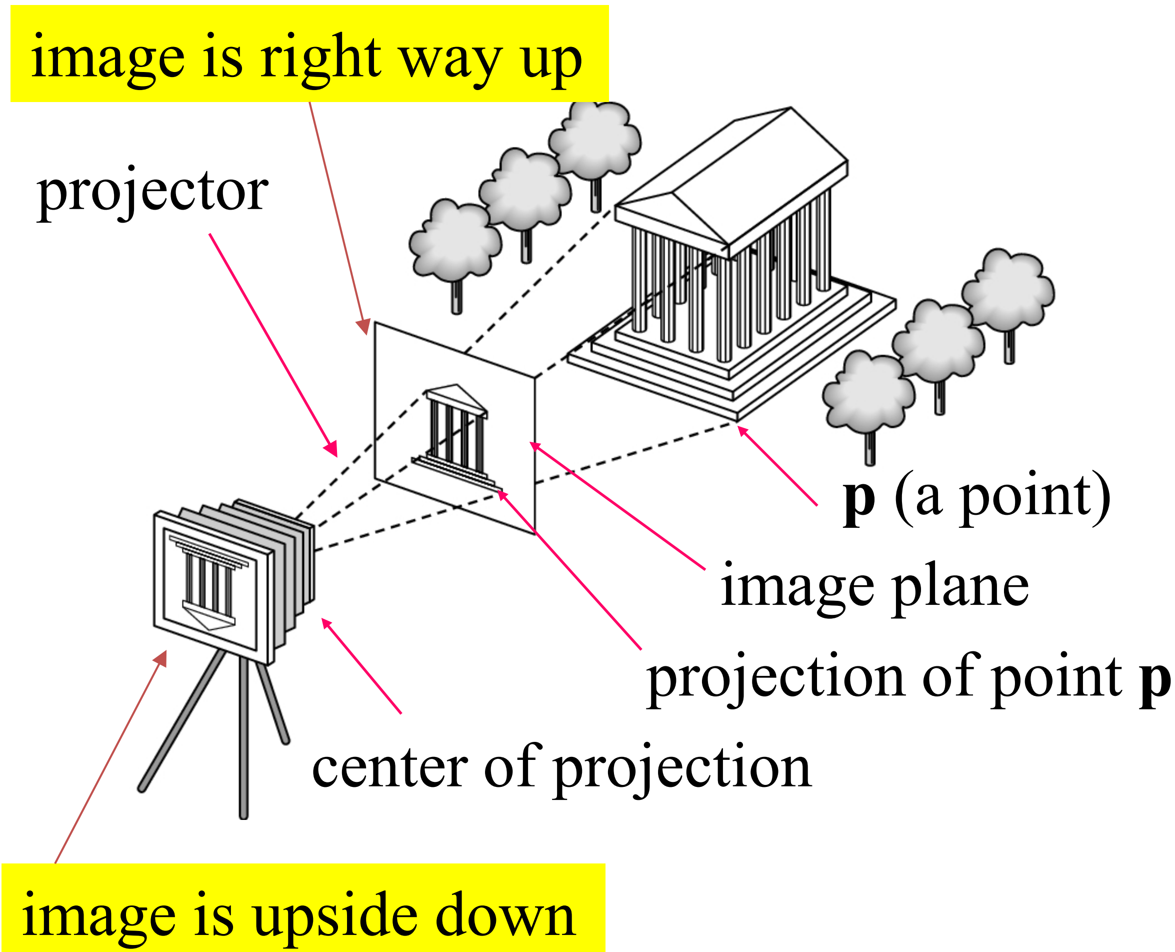
Subtractive color

- Form a color by filtering white light with cyan (C), Magenta (M), and Yellow (Y) filters
  - Light-material interactions
  - Printing
  - Negative film



https://en.wikipedia.org/wiki/Subtractive_color

Synthetic Camera Model



- OpenGL uses the synthetic pin hole camera model
- Since the image of the object is flipped relative to the object on the back of the camera, we draw another plane in front of the lens.
- With this synthetic camera model, the object is the right way up.

Synthetic Camera Model

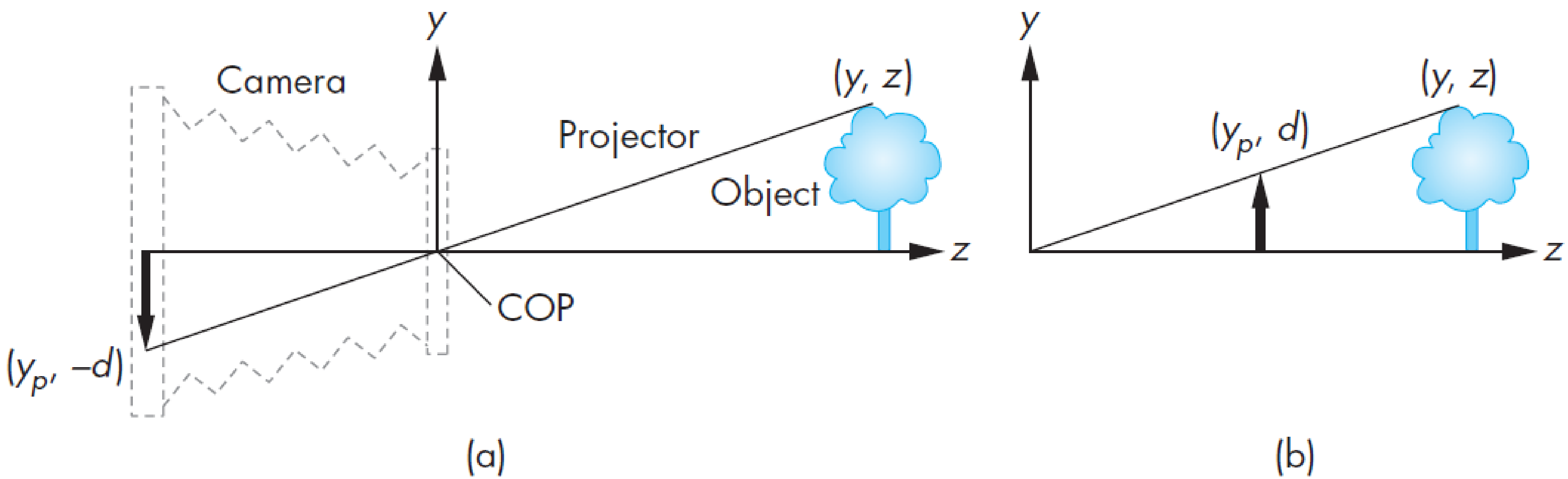


FIGURE 1.24 Equivalent views of image formation. (a) Image formed on the back of the camera. (b) Image plane moved in front of the camera.

Advantages

The synthetic-camera model is the basis for a number of popular APIs, including OpenGL

It stresses the independence of objects, viewer, light sources (can model them separately).

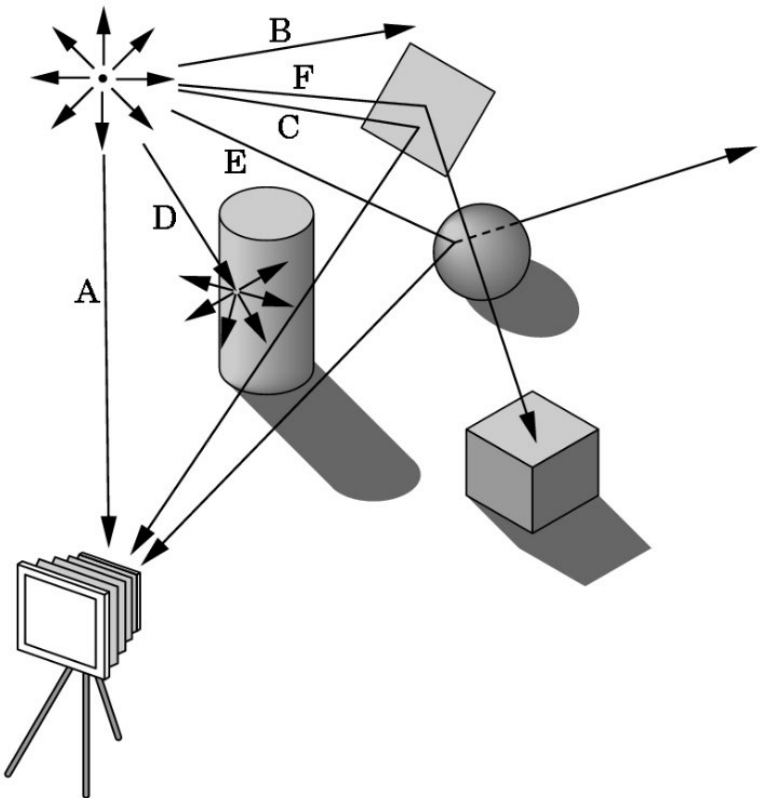
- Leads to simple software API
    - Can specify objects, lights, camera, attributes separately
    - Let implementation determine image by interaction
- Leads to fast hardware implementation
- Two-dimensional graphics becomes a special case of three-dimensional graphics

Ray Tracing: Physical Approach to Image Formation

Ray tracing and OpenGL represent two different methods of rendering images in computer graphics

Ray tracing is a rendering algorithm that simulates the physical behavior of light. It traces the rays of light from a source, finding which rays enter the camera lens.

However, rays of light may have multiple interactions with objects, get absorbed, or go to infinity.



Languages and Libraries

Graphics Libraries

Modern graphics programming is done using a graphics library/ set of libraries

- most common library for platforming independent graphics programming is called *OpenGL (Open Graphics Library)*.
- Using OpenGL with C++ requires configuring several libraries

We will use the following libraries:

- OpenGL / GLSL
- GLUT (window management)
- extension library

Some more libraries will be used in the lab#06 and the project

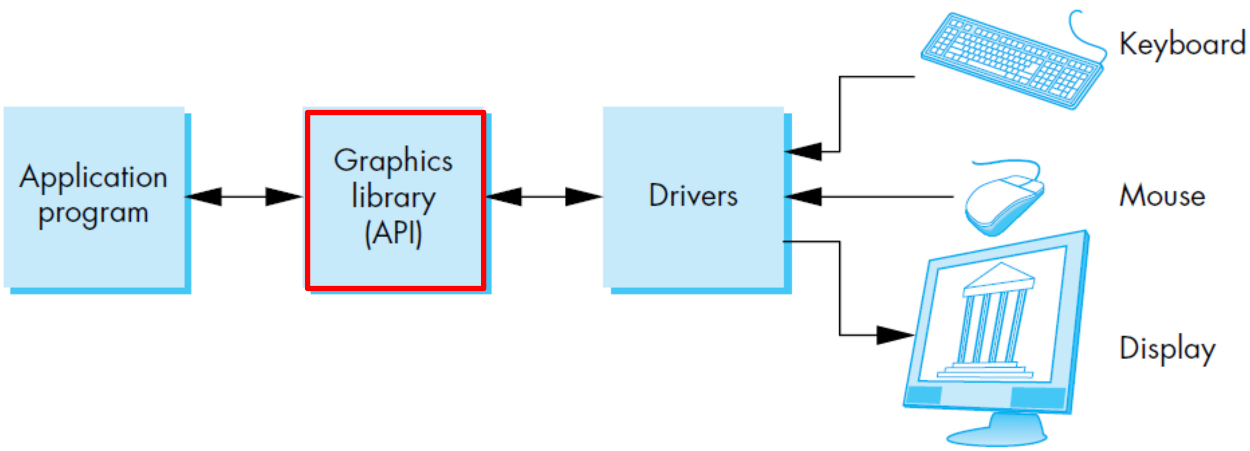
- *GLFW*
- *glm*
- *ImGui*

Introduction to OpenGL

What is OpenGL

OpenGL is a platform-independent Application Programmers' Interface (API) that

- Is close enough to the hardware to get excellent performance
- Provides a link between the low-level graphics hardware and the high-level application program that you write
- Is easy to use



- Most of the concepts related to OpenGL covered in week 01 are for introduction purpose.
- Many of these concepts will be repeated in more detail in the weeks to follow.

Variants of OpenGL

OpenGL ES

- Is suitable for embedded systems
- Version 1.0 is a simplified version of OpenGL 2.1
- Version 2.0 is a simplified version of OpenGL 3.1

- WebGL
  - Is a derivative of OpenGL ES version 2.0
  - Provides JavaScript bindings for OpenGL functions, allowing an HTML page to render images using any GPU resources available on the computer where the web browser is running

- WebGL and OpenGL ES are not included in the curriculum

Which Function is in which Library?

- You don’t need to memorize the functionalities of different OpenGL libraries
- Instead, you decide on your objects, lights and camera, then work out which OpenGL functions are required.
- Include libraries that contain your functions.
- For the practical issues you will have the OpenGL documentation to help.

https://docs.gl/

Further Reading

“Interactive Computer Graphics – A Top-Down Approach with Shader-Based OpenGL” by Edward Angel and Dave Shreiner, 6^{th} Ed, 2012

- Sec. 1.2 A graphics system
- Sec. 1.3 Images: Physical and Synthetic
- Sec. 1.4 Imaging Systems
- Sec. 1.5 The Synthetic Camera Model
- Sec. 1.6 The Programmer’s Interface

Acknowledgement

- It is important to acknowledge that this unit utilizes the resources developed and supplied by Edward Angel, Dave Shreiner, Gordon and V. Scott Gordon and John Clevenger, authors of the following textbooks:

  - "Interactive Computer Graphics – A Top-Down Approach with Shader-Based OpenGL" by Edward Angel and Dave Shreiner, 6th Ed, 2012
  - Computer Graphics Programming in OpenGL with C++, 2nd Ed, by V. Scott Gordon and John Clevenger

Choose the correct option(s)

ⓘ Start presenting to display the poll results on this slide.