COM3037-337
Computer Graphics

Dr. Emin Emrah ÖZSAVAŞ
eminemrahozsavas@gmail.com

https://drive.google.com/drive/folders/1oaPvvv8eo3JzmxD6nwWo8bbAabJCkjhA?usp=sharing

Asst. Metehan ÜNAL
metehan.unal@ankara.edu.tr
Asst. Mert ÇALIŞ
calism@ankara.edu.tr

The slides of the course are based on the slides by Prof. Edward Angel (author of the textbook)

Introduction

Aim:

To research

*** Theory**

*** Application area**

To apply what we've learned - practice

Introduction

Scope:

Topic: Fundamentals of computer graphics

Adobe Photoshop, Light Scape, 3D Studio Max, Unity3D, Unreal SDK, …

Graphics programming using WebGL

Syllabus

Hafta Week	Konular Topics
1	Giriş, Grafik Sistemleri ve Modelleri, Grafik Programlama Introduction, Graphics Systems and Models, Graphics Programming
2	Giriş, Grafik Sistemleri ve Modelleri, Grafik Programlama Introduction, Graphics Systems and Models, Graphics Programming
3	Dönüşümler Transformations
4	Dönüşümler Transformations
5	Görüntüleme Viewing
6	Işıklandırma ve Gölgelendirme Lighting and Shading
7	Işıklandırma ve Gölgelendirme Lighting and Shading

Syllabus

8	Doku Kaplama, Çıkıntı Eşleme Texture Mapping, Bump Mapping
9	Kırpma Clipping
10	Kırpma Clipping
11	Tarama Dönüşümü Scan Conversion
12	Tarama Dönüşümü Scan Conversion
13	Gizli Yüzeylerin Kaldırılması Hidden Surface Removal
14	Genel Gözden Geçirme General Review

Introduction

Textbooks:

MAIN TEXTBOOK:
Interactive Computer Graphics: A Top-Down Approach with WebGL, 7th Ed., Edward Angel and Dave Shreiner, Pearson Education, 2015.

Supplementary books:
** Hearn and Baker, Computer Graphics with OpenGL, Pearson / Prentice Hall.*
** Foley, van Dam, Feiner, and Hughes, Computer Graphics – Principles and Practice, Addison-Wesley.*

Introduction

Other supplementary books:

* OpenGL www.opengl.org
  - The Red Book
    - Sheriner, Sellers, Kessenich, and Licea-Kane, *OpenGL Programming Guide*, Eighth Edition, *Addison-Wesley.* https://www.cs.utexas.edu/users/fussell/courses/cs354/handouts/Addison.Wesley.OpenGL.Programming.Guide.8th.Edition.Mar.2013.ISBN.0321773039.pdf
  - The Blue Book
    - The OpenGL Reference Manual: The Official Reference Document to OpenGL, Version 1.4 (4th Edition), Addison-Wesley. http://www.glprogramming.com/blue/

Introduction

Web:

- www.cs.unm.edu/~angel/
- www.cs.unm.edu/~angel/WebGL/7E
- www.opengl.org
- get.webgl.org
- www.khronos.org/webgl
- www.chromeexperiments.com/webgl
- Mozilla Developer Network WebGL Tutorial

Introduction

Grading:

Practical Assignments (will replace the midterm exam)	30 %
FINAL	80 %

Introduction

Prerequisites:

* **Programming knowledge**
  - **Basic data structures**
  - **Linked lists**
  - **Arrays**
* **Geometry**
* **Simple linear algebra**

Introduction

Computer graphics (CG):

* An information presentation technology

* Deals with rendering

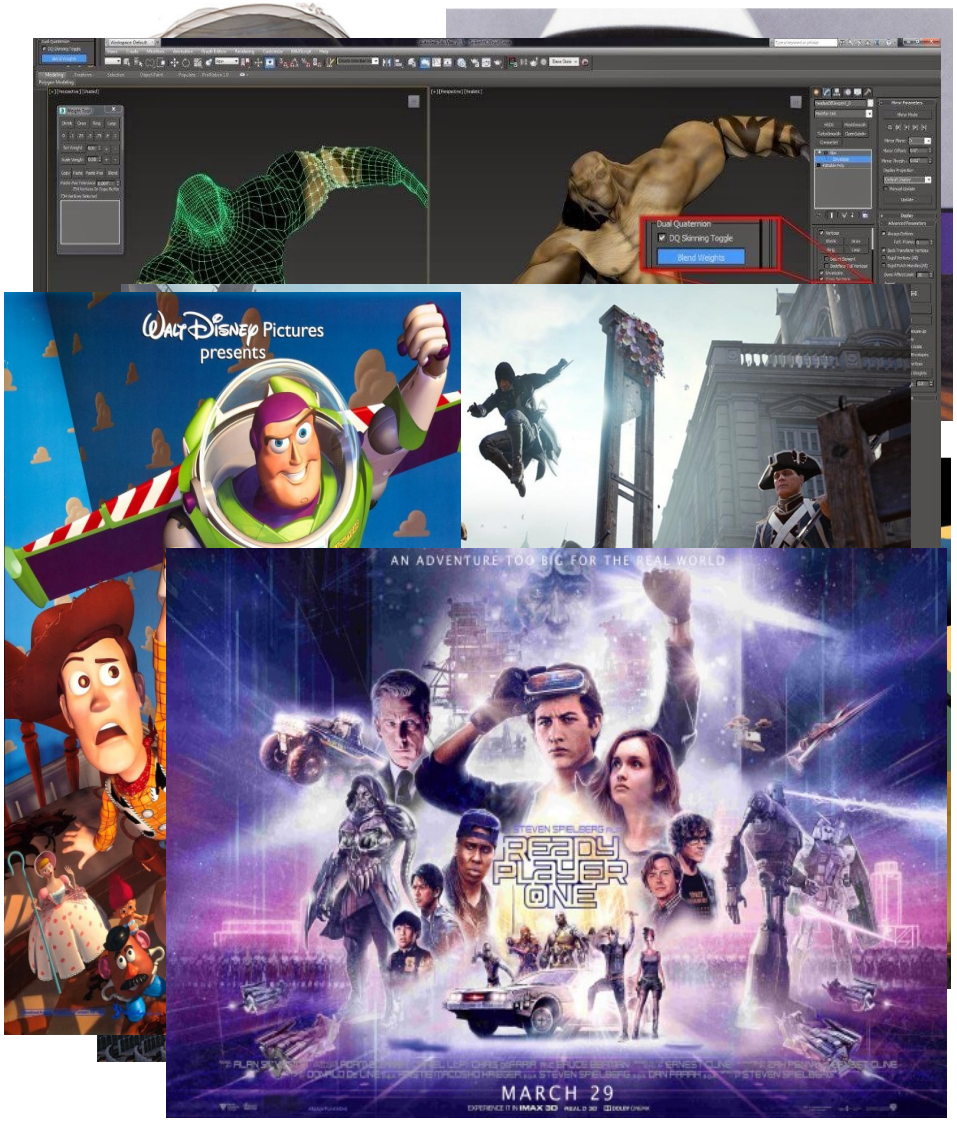
* Covers anything not related to text (also some issues about text)

* Many fields such as computer vision and animation emerged from CG

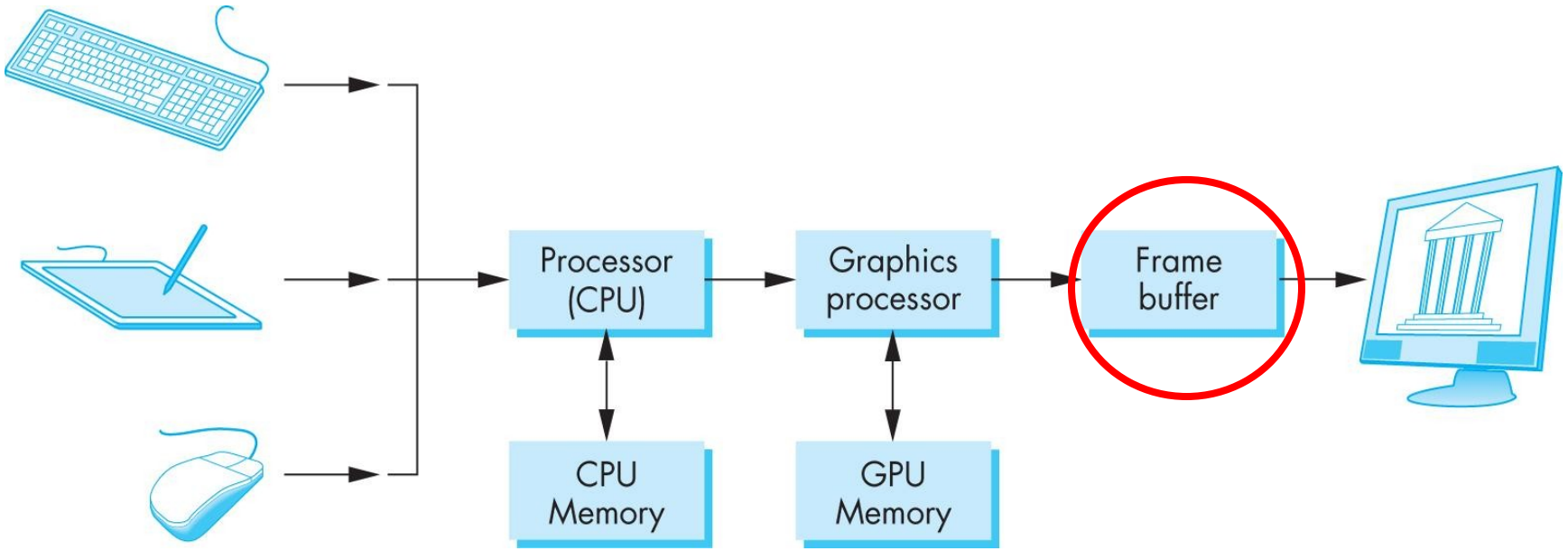
Introduction

Applications of CG

- Information visualization: maps, building plans, etc.
- Graphical user interfaces(GUI)
- Medical imaging (CT, MRI, PET …)
- Computer Aided Design (CAD)
- Simulators
- Games
- Educational software
- Animations (TV, movies, advertisements)
- Virtual Reality – VR
- Augmented Reality – AR
- Mixed Reality – MR



Basic Graphics System



Input device

Output device

Resolution
Depth (precision)
Raster / rasterization / scan conversion

Basic Graphics System

Raster: array of pixels (stored in FB)

Graphics systems are raster-based

Concept of depth:

1-bit-deep ⇒ black and white (binary)

8-bit-deep ⇒ $2^8 = 256$ colors (gray scale / monochrome)

24-bit-deep ⇒ RGB-color system:

red, green, and blue channels

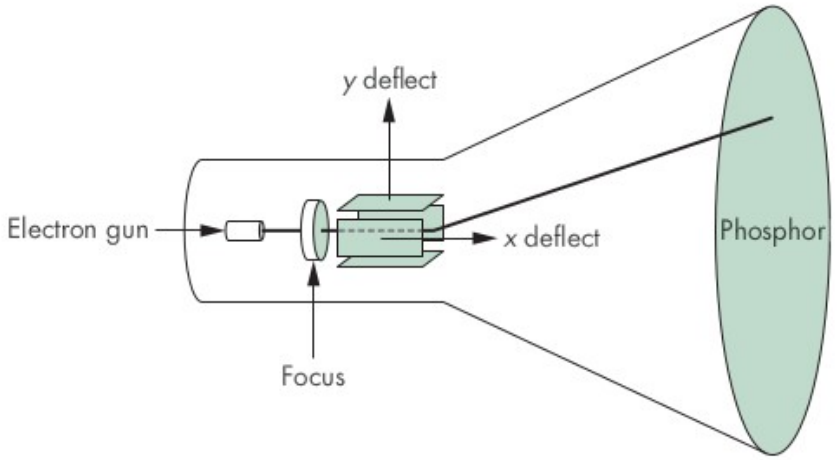
for each channel 256 shades

Basic Graphics System

Output devices:

* **Cathode-ray tube; CRT**

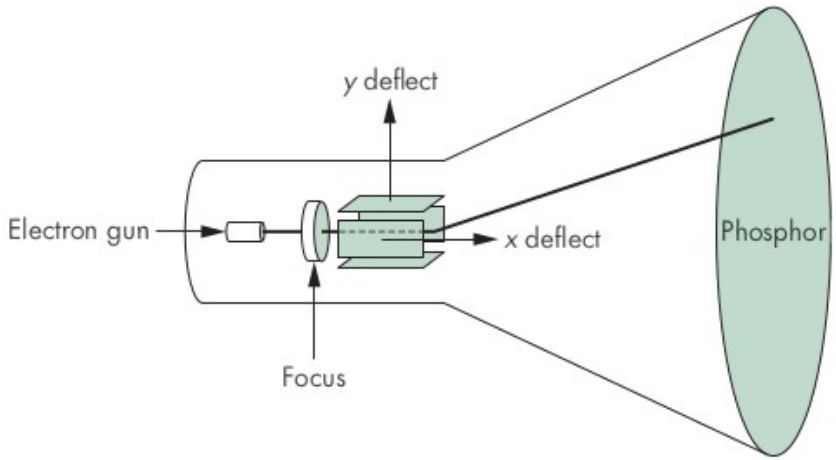
- **Cone-shaped**
- **Have an electron gun (narrow end)**
- **Have a phosphor layer (wide end)-Anode**
- **Cathode (heated thin filament) in electron gun**
- **No air in the tube (vacuum)**



Basic Graphics System

Output devices:

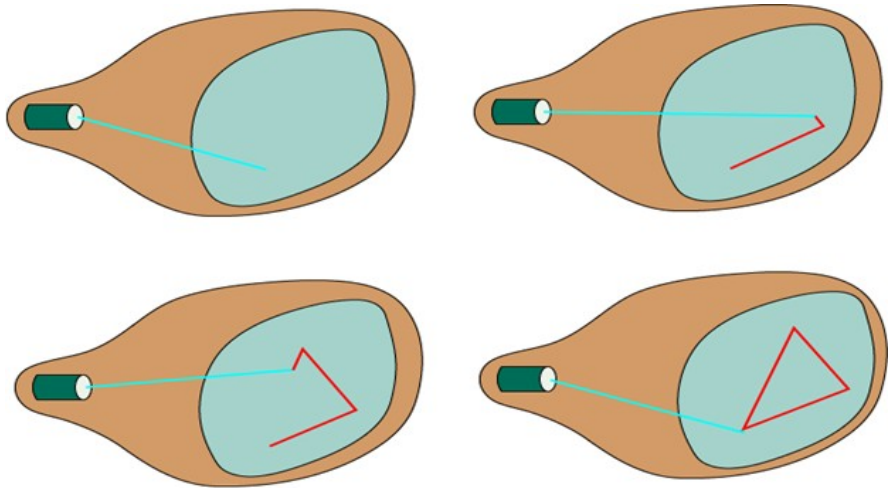
* **Cathode-ray tube; CRT - How it works?**
  - **Heating the cathode makes electrons move in the vacuum towards the anode as a beam of light.**
  - **Electrons hit the phosphor layer, shine and give out light for a short time (radiation).**
  - **Horizontal and vertical deflection coils provide radiation for every points (on the screen).**



Basic Graphics System

Output devices:

* **Cathode-ray tube; CRT**
  - ***Refresh Rate:*** **For a stable image (no flickering), the same beam traversal should be repeated at a certain rate (should be refreshed).**
  - ***Random-scan or vector scan or calligraphic CRTs:*** **early CRT screens, the beam passed over the shape to be drawn.**



Basic Graphics System

Output devices:

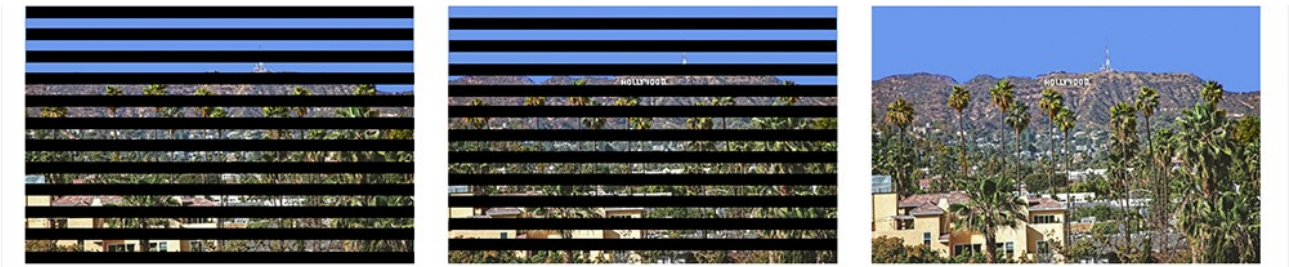
* **Cathode-ray tube; CRT**

- ***Raster-scan CRTs:*** line by line scanning, progresses line by line and transfers the pixels in the frame buffer to the screen. Two types:

  Non-Interlaced: (or progressive scan) screens display lines consecutively: 1-2-3-4-5-6 …, scanning line by line.

  Interlaced: odd and even rows (fields: half of image) are refreshed alternately, display odd lines first: 1-3-5 ...; then even lines: 2-4-6 ...

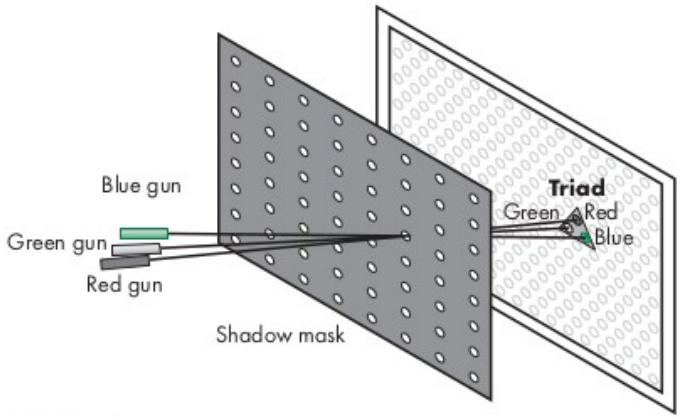
e.g. 60 Hz refresh rate: One field in 1/60th seconds (interlaced)
One image in 1/60th seconds (non-interlaced)



Basic Graphics System

Output devices:

* **Cathode-ray tube; CRT**
  - ***RGB CRTs* have 3 electron guns (each channel has a gun).**
  - **Light beam passes through a perforated shadow mask.**
  - **Every pixel on the screen has three subdots (triad).**
  - **Shadow mask ensures that the beam hits only the dot(s) of the desired color (activates these points separately).**
  - **Combination of the subdots is the color appears on the screen (main pixel color).**
  - **Color variations are obtained by varying the intensity levels of the three electron beams.**



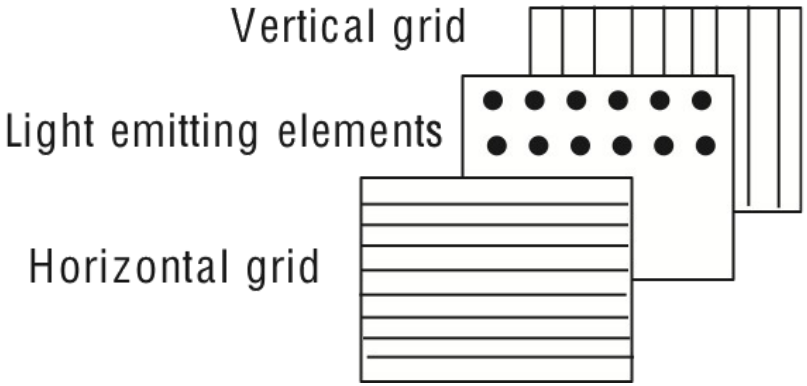
Basic Graphics System

Output devices:

* Flat-screen technologies

- ***LED:*** **Light Emitting Diodes are the light source.**
- ***LCD:*** **Liquid-Crystal Displays, fluorescent is the light source.**
- ***Plasma Panel:*** **ionized gas is the light source.**

They control individual light emitting or transmitting elements with a 2D grid.
The intersection of a column in a vertical grid and a line in a horizontal grid is applied a different level of energy and radiation is provided at that point (pixel).



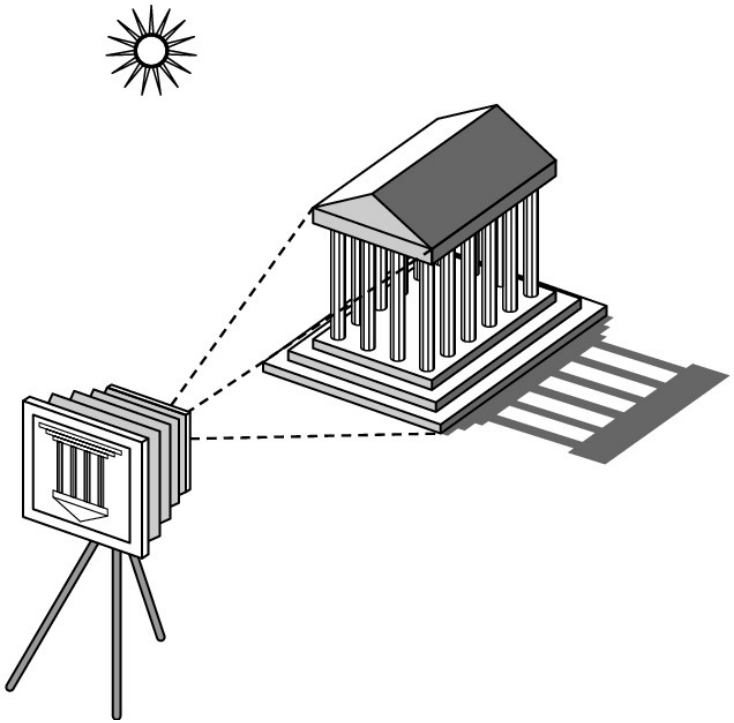
Basic Graphics System

Input devices: Keyboard, mouse, joystick …

Basic Graphics System

Image formation:

Physical (real) imaging systems: Camera, microscope, telescope, human visual system



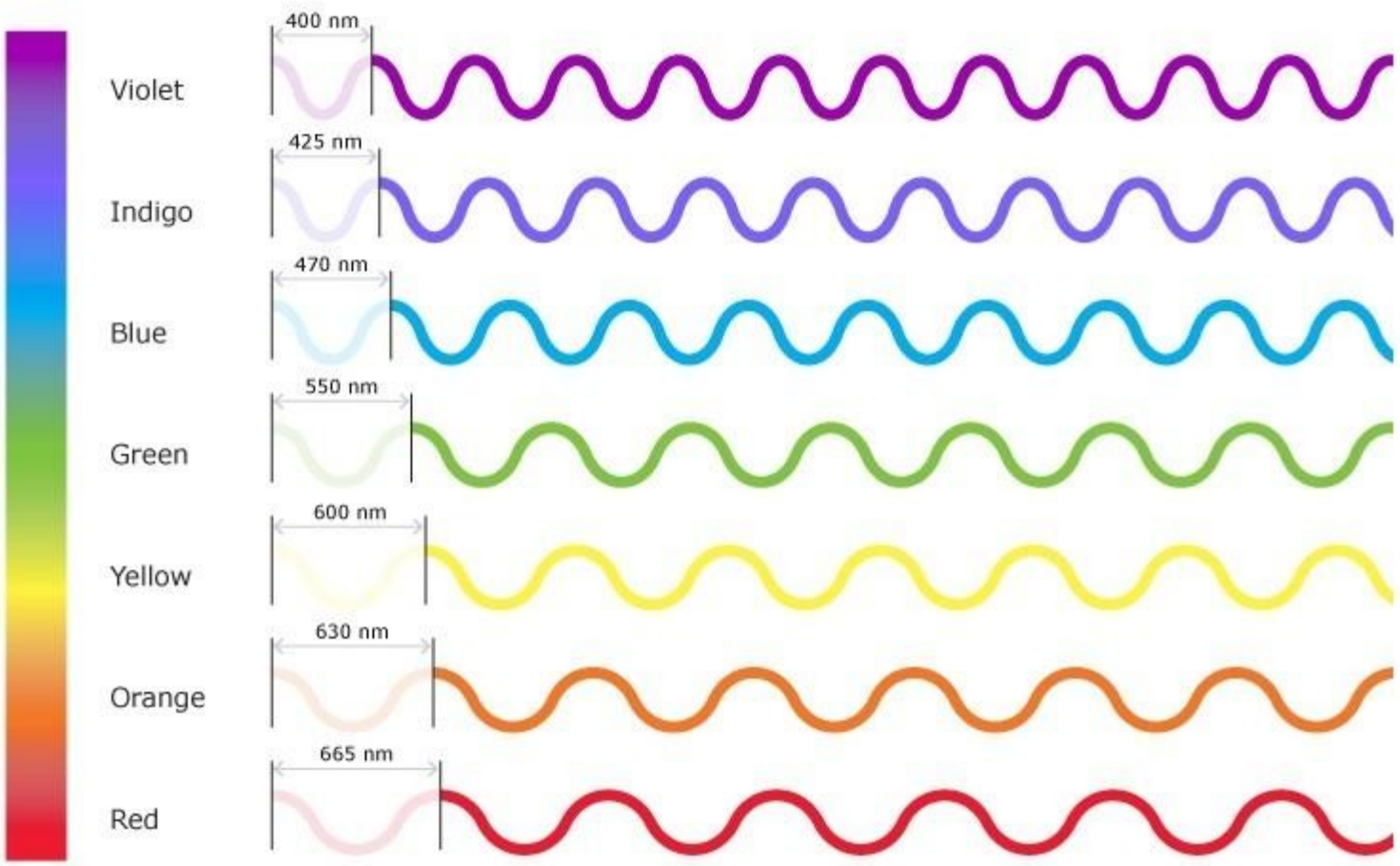
Needed for (elements of) image formation:

- Objects
- Viewer
- Light source(s).

Basic Graphics System

Light: made up of wavelengths of light, each wavelength is a particular color.

Visible light: electromagnetic waves we can see.



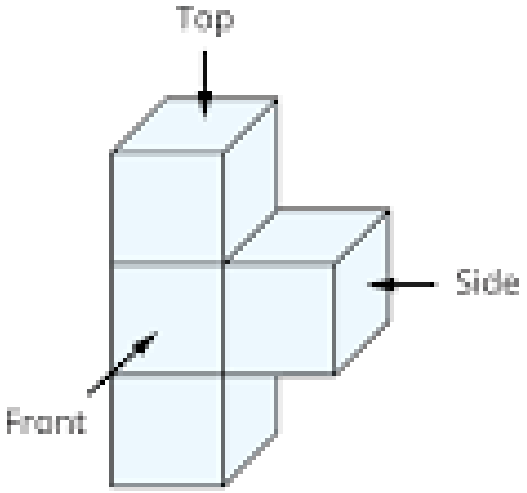
Basic Graphics System

Colors of objects:

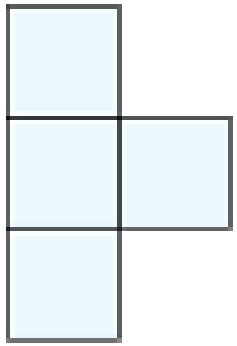
Basic Graphics System

Image formation:

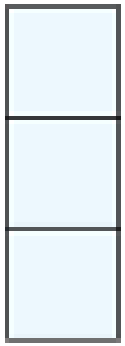
* Like real imaging systems, we try to form 2D images (create virtual objects) in CG.
* To do so we have to specify the positions of geometric objects (points, lines, polygons) and their vertices.



A solid made of cubes



Front view



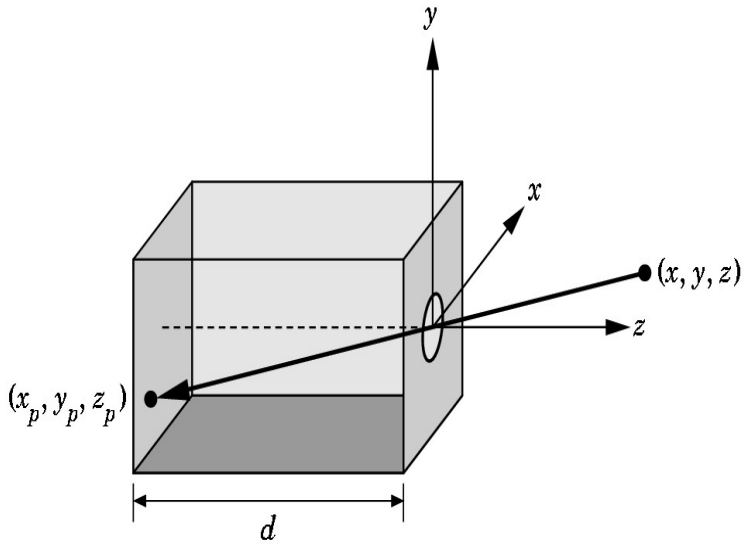
Side view

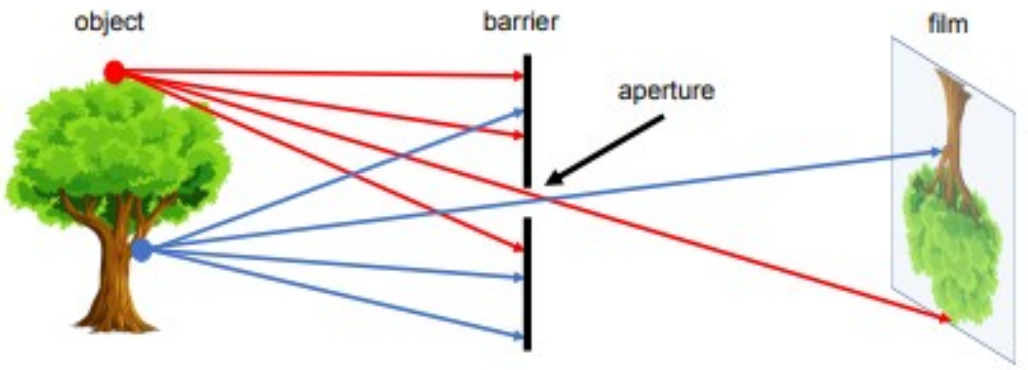
Top view

Basic Graphics System

The Pinhole camera: a lensless camera model

* An opaque box with a small hole.
* Barrier is like the surfaces of the box.
* Aperture or center of projection is the pinhole or center of the camera.
* Film is image, retinal or projection plane.
* The distance (d) between the image plane and the pinhole is the focal length.
* Light from the object passes through the pinhole, reflects an inverted image on the opposite side (projection plane) of the box.

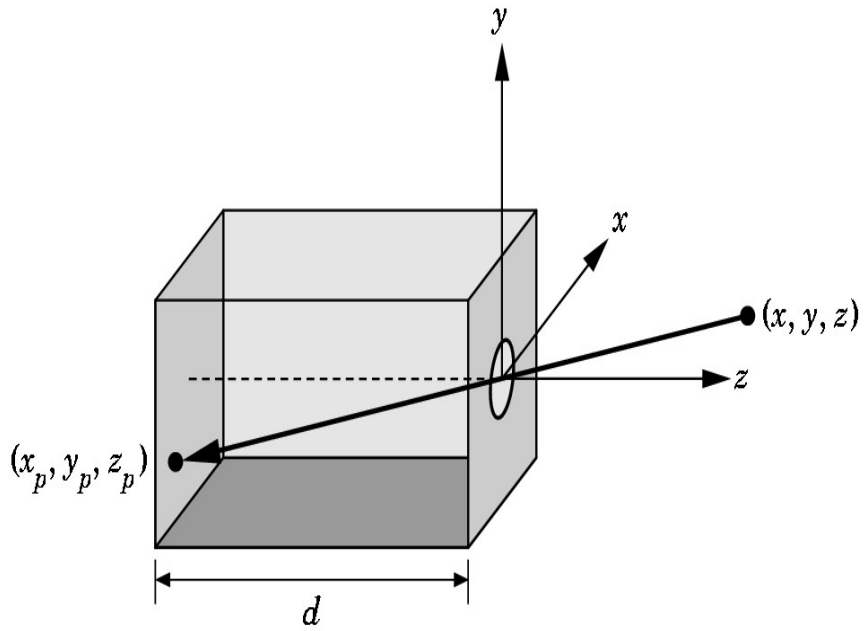




Basic Graphics System

The Pinhole camera:

* This coordinate system is the camera reference system or camera coordinate system.
* The ray from point (x, y, z) is projected to the projection point.
* Law of similar triangles (to find the coordinates of the projection point)
* In an idealized model, the color on the film plane at this point is the color of the point (x, y, z).

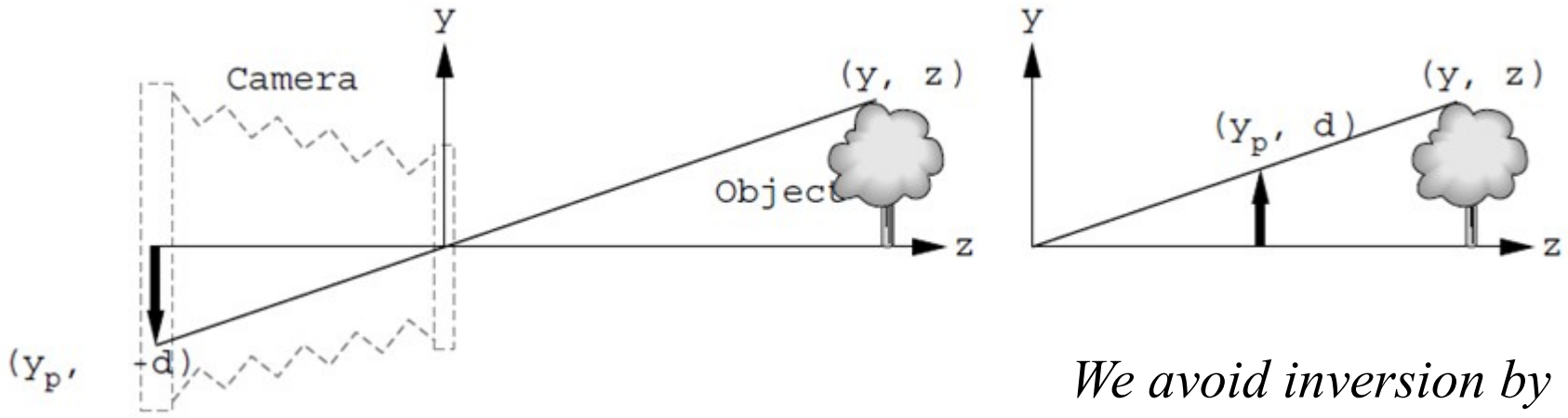


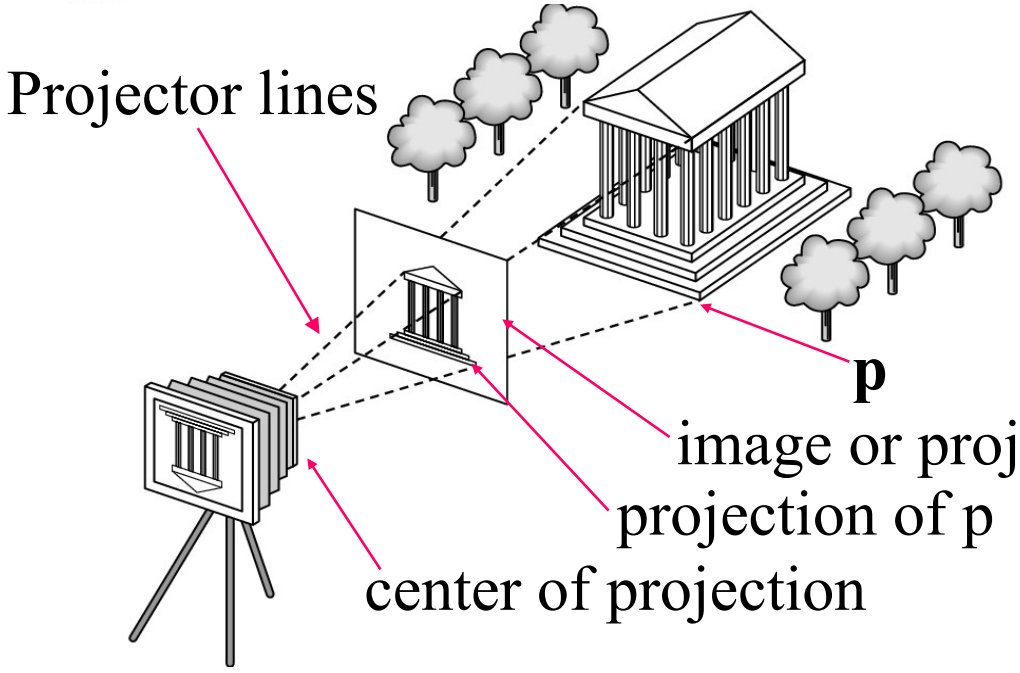
Projection of the point (x,y,z) :

$x_p = -dx/z$ $\quad y_p = -dy/z$ $\quad z_p = -d$

Basic Graphics System

The Synthetic camera model: is used to mimic the behaviour of a real camera instead of pinhole model.

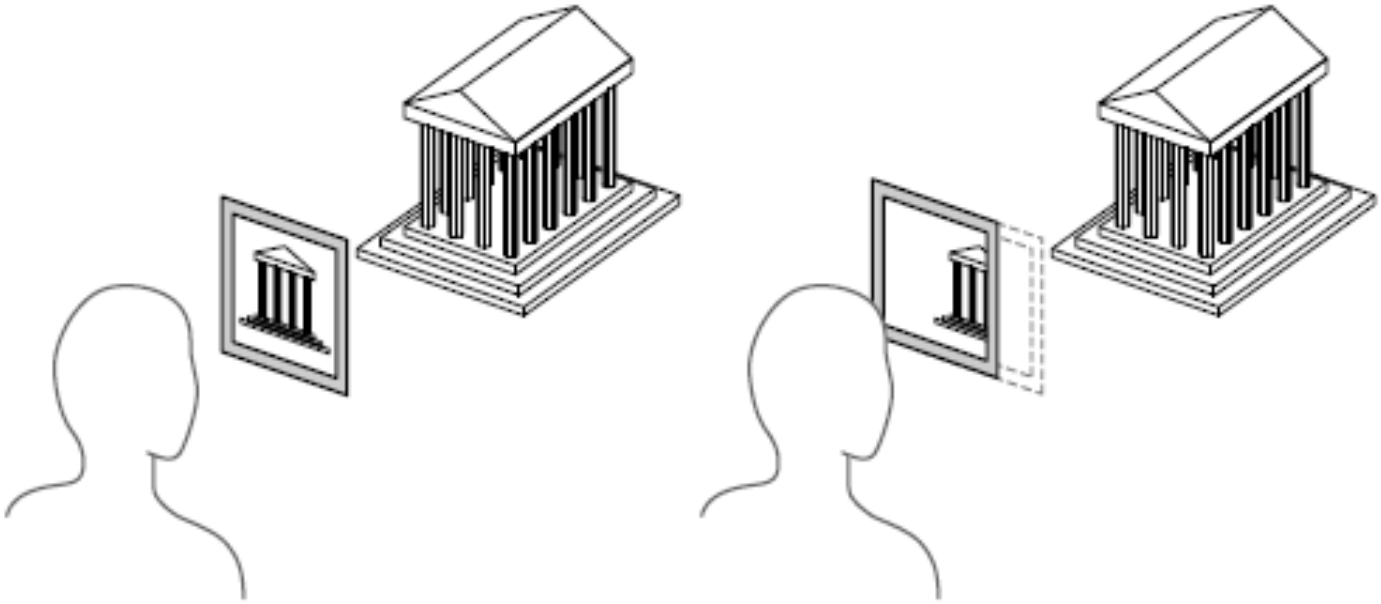




We avoid inversion by placing the film plane (the image or projection plane) in front of the camera.

Basic Graphics System

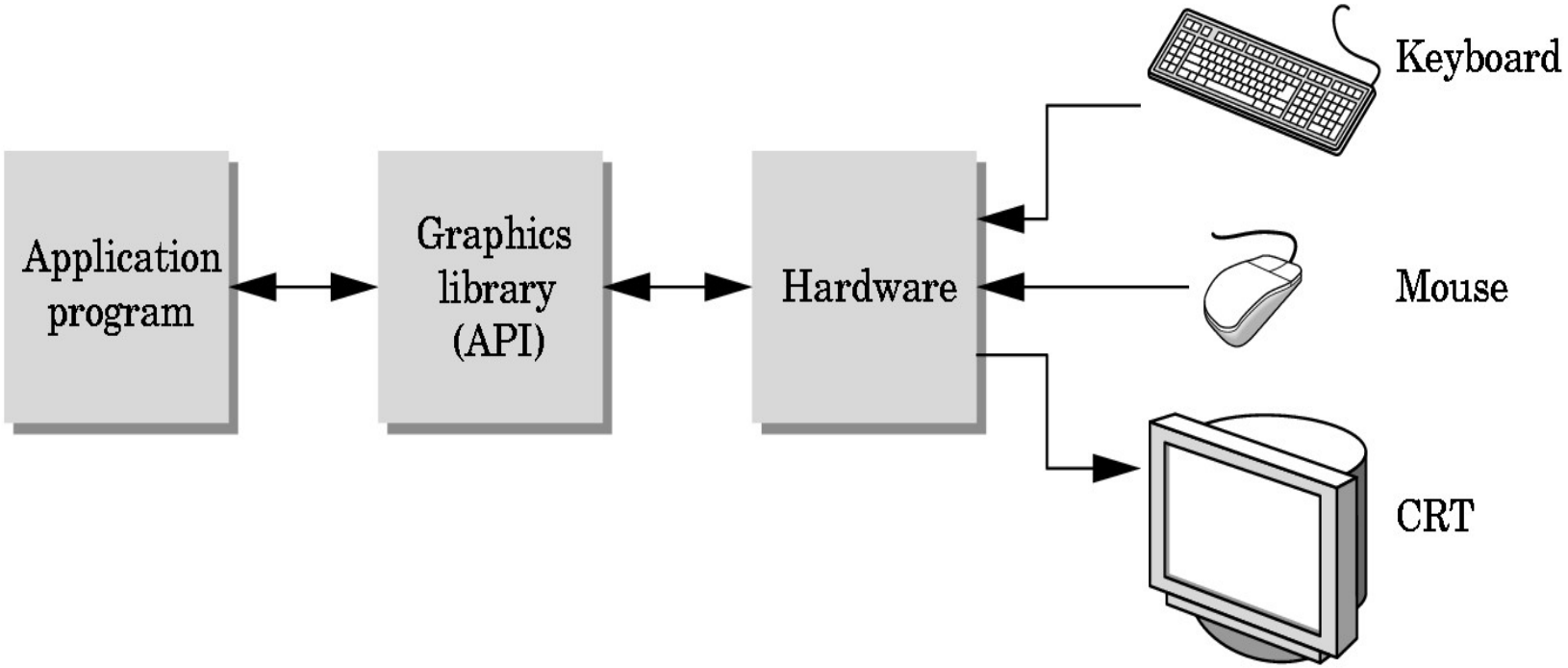
The Synthetic camera model:



The clipping rectangle or clipping window determines the size of the image.

Basic Graphics System

The Programmer's interface:



- A graphical application programmer interacts with the graphics system through function specifications (APIs) in a graphics library.

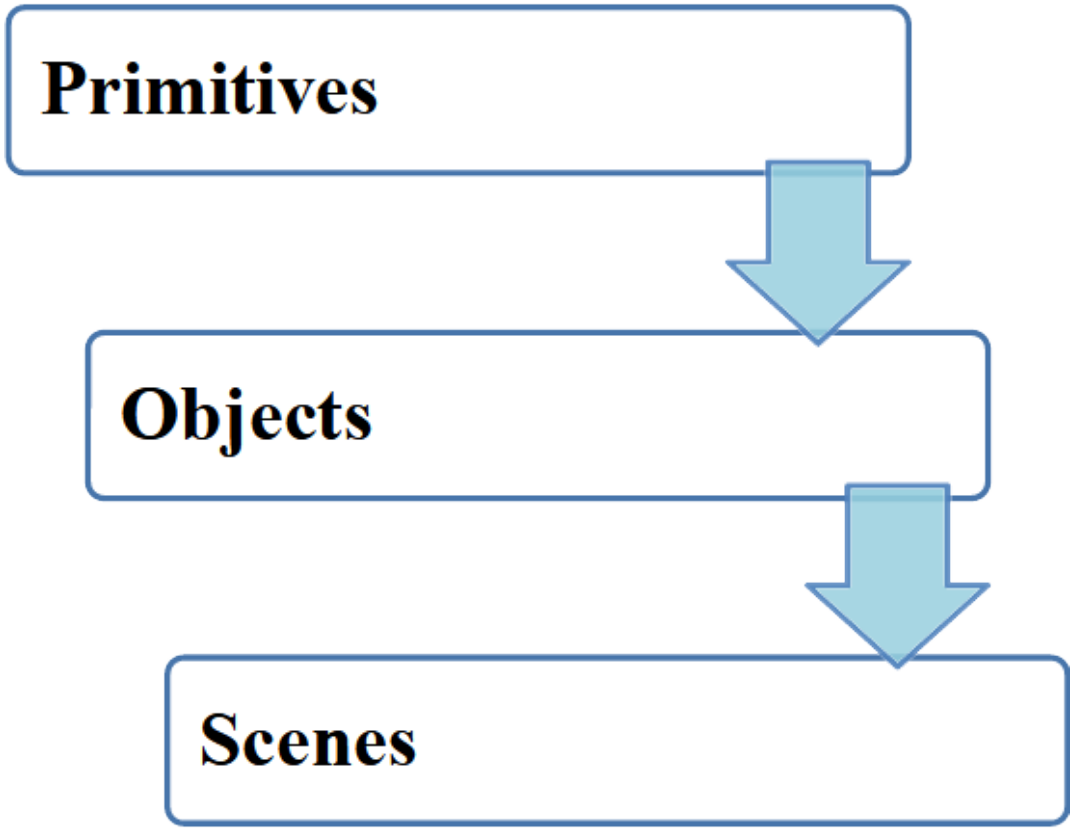
- APIs like OpenGL and Direct3D use the synthetic camera model.

- We need API functions to specify elements of image formation: *objects (geometry and other properties), a viewer, and light sources*

Basic Graphics System

Object geometries: are defined by sets of vertices.

Most APIs provide similar sets of primitive objects (points, line segments, and triangles) that can be displayed rapidly on the hardware.



Basic Graphics System

Object geometries:

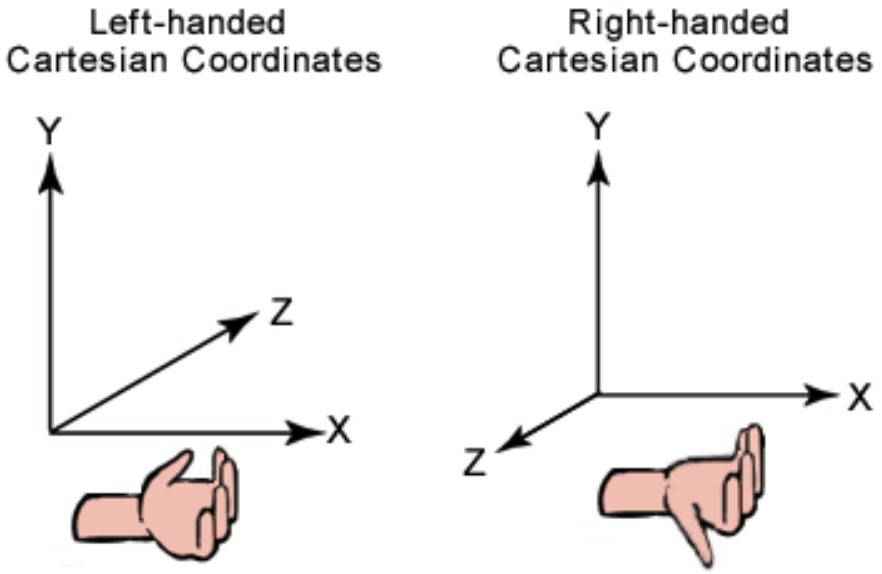
First the objects to be displayed must be described.

- *A sphere is specified by its centre position and its radius.*
- *A box is specified by the positions of its vertices (corners).*

Basic Graphics System

Object geometries: 3-D graphics applications use two types of Cartesian coordinate systems:

- the positive x-axis points to the right, the positive y-axis points up
- point the fingers of either your left or right hand in the positive x direction and curl them into the positive y direction. The direction your thumb points, *either toward or away from you*, is the direction that the *positive z-axis* points for that coordinate system.



Microsoft Direct3D uses a left-handed, OpenGL uses a right-handed coordinate system.

Basic Graphics System

The Graphics pipeline:

- A set of data processing elements connected in series, where the output of one element is the input of the next one.
- The elements of a pipeline are often executed in parallel or in time-sliced fashion *(e.g. to begin calculations for a new vertex, the previous vertex's calculations are not waited to finish.)*.
- Increases system throughput if same computation stages are carried out on different data again and again.
- Graphics cards mostly have pipelines built into the GPU.

Basic Graphics System

The Graphics pipeline:

- Vertex processor: coordinate transformations and color computations are done on each vertex.
- Clipping: determines parts of objects that will be in the view *(must be done at a primitive level, not vertex level, so we must also assemble vertices into primitives)*.
- Rasterization: primitives from the 2nd stage are still represented with vertices, they are converted to fragments *(potential pixels, operations can still happen on fragments whereas no operations can further happen on pixels)*.
- Fragment processor: takes fragments as input and updates the pixels in the framebuffer *(while pixels contains color/intensity information only, fragments contains more than color information, such as depth, point size, etc.)*.

Basic Graphics System

The Graphics pipeline:

- Fixed-function pipeline:

* will work on ancient hardware, and can reduce the amount of code which you need to write.

* graphics implemented through a set of pre-defined functions, such as transform and lighting (glRotatef, glLightfv, etc.).

* we are limited to what those functions can provide, very limited compared to what can be achieved with shaders.

Basic Graphics System

The Graphics pipeline:

- Programmable pipeline:

* graphics hardware is programmable through shader programs (implemented in a shader language e.g. OpenGL Shading Language-GLSL)

* shader programs run on the GPU (an opportunity to implement things that are difficult or impossible to do using the fixed function pipeline).

Basic Graphics System

WebGL vs OpenGL:

WebGL	OpenGL
It is mainly used to run in the browser for web applications	It is mainly used in desktop applications, do need native drivers
It is programmed in JavaScript programming	It is written in C language
It has fewer features comparatively	It has many features
There is no fixed-function pipeline (programmable pipeline)	There is a fixed function pipeline
https://www.khronos.org/webgl/	Opengl.org