COMPUTER GRAPHICS

UNIT 1

INTRODUCTION

Syllabus

* Applications of computer graphics
* A graphics system
* Images: physical and synthetic
* Imaging systems.
* The synthetic camera model
* The programmer's interface.
* Graphics architectures.
* programmable pipelines.
* performance characteristics

* Sierpinski gasket
* programming two-dimensional applications.

7 Hours.

~~APPLICATIONS OF~~

Computer Graphics is concerned with all aspects of producing pictures or impages using a computer.

APPLICATIONS OF COMPUTER GRAPHICS

* Applications of computer graphics are divided into four major areas,
  - i) Display of information
  - ii) Design
  - iii) simulation and animation
  - iv) user interfaces.

Display of information

* Architects, mechanical designers, and draftspeople uses graphics system to generate useful information like floor plans of building etc
* Graphics can be used to develop and manipulate maps to display geographical and celestial information.
* Workers in the field of statistics uses graphics system (computer plotting packages) for generating plots that aid the viewer in understanding the information in a set of data.
* 3-D data generated by CT (computed Tomography), MRI (magnetic resonance imaging), ultrasound, and positron-emission-Tomography (PET) using computer graphics
* Field of scientific visualization provides graphical tools that helps researchers interpret the vast quantity of data that they generate
  - → working on supercomputer

Design

* Today, the use of interactive graphical tools in computer-aided design (CAD) pervades fields including as architecture, mechanical engg. the design of VLSI circuits, and the creation of characters for animations.
* In all these applications, graphics are used in number of ways (distinct ways)
for ex; in a VLSI design, the graphics provide an interactive interface b/w the user and the design package, usually by means of such tools as menus and icons.

Simulation and Animation

* once graphics systems evolved to be capable of generating sophisticated images in real time, engineers and researchers began to use them as simulators.
one of the most important uses has been in the training of pilots
* the success of flight simulators led to the use of computer graphics for animation in television, motion picture, and advertising industries.
  - (time varying behaviour of real and simulated objects)

user interfaces.

* Most applications that run on PC and workstations have user interface that rely on desktop window system to manage multiple simultaneous activities.
* These user interface are working with computer graphics.

A GRAPHICS SYSTEM

* There are five major elements in a graphics system.
  1. Input devices
  2. processor
  3. Memory
  4. Frame Buffer
  5. Output devices.

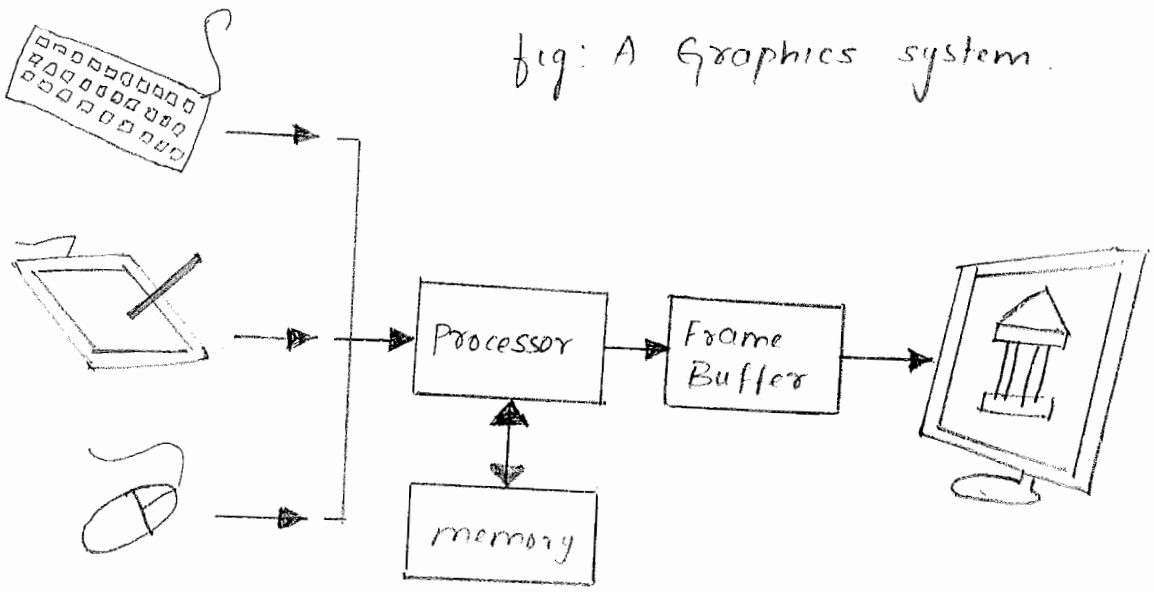


fig: A Graphics system.

pixels and the Frame buffer (core element of graphic system)

* presently almost all graphics systems are raster based. A picture is produced as an array (the raster) of picture elements, or pixels within the graphics system. pixel is the smallest component of an image.
* Pixels are stored in a part of memory called Frame buffer. Its resolution (no. of pixels in the frame buffer) determines the detail that you can see in the image.
* Depth or precision of the frame buffer is defined as the number of bits that are used for each pixel. It determines the properties such as how many colors can be represented on a given system.

ex: 1-bit deep frame buffer allows only two colors.
8-bit deep frame buffer allows $2^8 = 256$ colors.

per pixel
if depth = 24 bits or more, such a systems are called Full color systems or True color systems or RGB-color systems.

* Frame buffer usually is implemented with special types of memory chips that enable fast redisplay of contents of frame buffer.

* In simple systems, there may be only one processor, the CPU of the system, which must do both the normal processing & graphical processing.
The main graphical function of a processor is to take specifications of graphical primitives (such as lines, polygons generated by application programs and to assign values to the pixels in the frame buffer that best represent these entities.
For ex; A triangle is specified by its three vertices, but to display its outline by the three line segments connecting the vertices, the graphic system must generate a set of pixels that appear as line segments to the viewer.

Defn: The conversion of geometric entities to pixel color and locations in the frame buffer is known as rasterization or scan conversion.

(or) It is the process of converting graphical primitives (or images) into pixel representation (or frame buffer representation)

* In earlier graphic system, frame buffer was part of standard m/m that could directly accessed by CPU.
Today, almost all graphic systems are characterized by special-purpose graphics processing units (GPU) which can be either on motherboard of the system or on graphic card
the frame buffer is accessed through GPU & may be included in the GPU.

Output Devices.

* For many years, the dominant type of display has been the cathode ray Tube (CRT)
* Fig below shows simplified picture of a CRT.

Working:

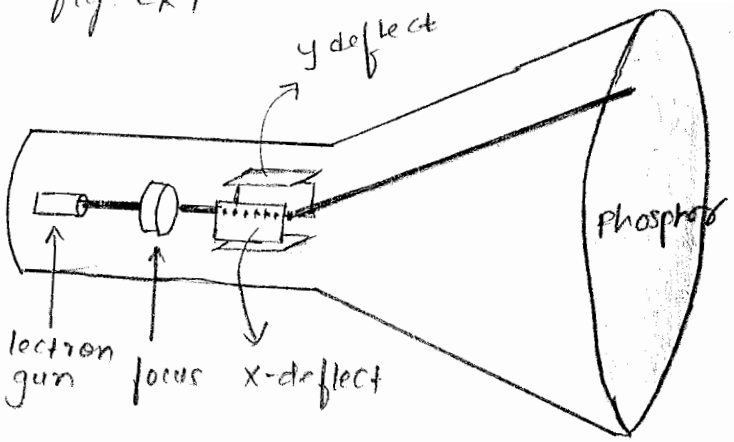


fig: CRT

* when electrons strike the phosphor coating on the tube, light is emitted. The direction of beam is controlled by two pairs of deflection plates.
* The output of computer is converted to voltages across the x and y deflection plates (by digital to analog converters)
* If the voltages steering the beam change at a constant rate, the beam will trace a straight line, visible to a viewer. Such a device is known as random scan, calligraphic, or vector CRT, because the beam can be moved directly from any position to any other position.

Refreshing.

* A typical CRT will emit light for only a short time (- few milliseconds) after phosphor is excited by electron beam. Therefore, for a human to see a flicker free image, the same path must be retraced or refreshed, by the beam at a sufficiently high rate (refresh rate). In US → 60 cycles per sec or 60 Hz.
  rest of the world → 50Hz.

* In raster system, the graphics system takes pixels from the frame buffer and displays them as points on the surface of the display in one of two fundamental ways:

i) Non interlaced or progressive display.
   Here, pixels are displayed row by row, or scanline by scanline, at the refresh rate.

ii) Interlaced display.
   Here odd rows and even rows are refreshed alternatively. This method is used in commercial TV.
   In an interlaced display operating at 60Hz, the screen in redrawn in its entirety only 30 times per second, although visual system is tricked into ~~the~~ thinking the refresh rate is 60Hz rather than 30Hz.

Shadow-mask CRT (color CRT)

* color CRTs have 3 different colored phosphors (red, green, and blue), arranged in small groups. one common style arranges the phosphors in triangular groups called Triads, each triad consisting of three phosphors, one of each primary.

* Most color CRTs have three electron beams, corresponding to the three types of phosphors. In the shadow-mask CRT, a metal screen with small holes (shadow mask) ensures that an electron beam excites only phosphors of the proper color.

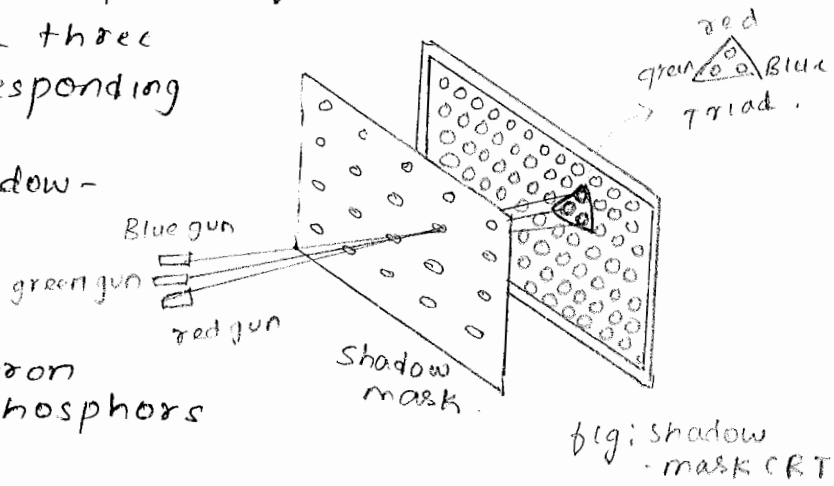
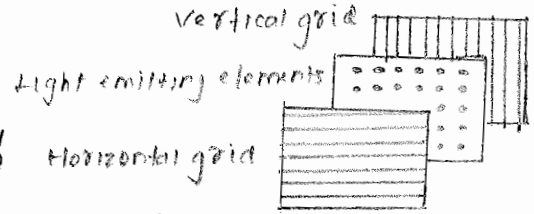


fig: shadow-mask CRT

Flat-screen Technologies.

* Flat panel monitors are are inherently raster.
* There are three technologies available.
  - → Light emitting diodes (LEDs)
  - → Liquid-crystal Display (LCD)
  - → Plasma panels.
* All these uses 2-D grid to address individual light emitting elements.
* Fig. shows a generic flat-panel display.
* By sending electric signals to the proper wire in each grid, the electric field at a location, determined by the intersection of two wires, can be made strong enough to control the corresponding element in the middle plate.



* Middle plate in an LED panel contains LEDs that can be turned on & off by electric signals sent to the grid.
* In an LCD display, electrical field controls the polarization of the liquid crystals in the middle panel, thus turning on & off the light passing through the panel.
* A plasma panel uses the voltage on the grids to energize gases embedded ~~with~~ b/w the glass panels holding the grid The energized gas becomes a glowing plasma

Aspect ratio.

* It is the width to height ratio of the frame buffer.
* until recently, most displays had a 4:3 aspect ratio.
* VGA resolution → 640 × 480 resolution.
  ie $\frac{640}{480}$ aspect ratio
* XGA → 1024 × 768.
* HDTV → 16:9.

Input Devices.

* common i/p devices are ; mouse, joystick, & the data tablet. Each provides positional information to the system, and each usually are equipped with one or more buttons to provide signals to the processor.
* mouse → pointing devices, used for selection of an image, cut, copy, paste etc
* keyboard → used for inserting text based information.

IMAGES: PHYSICAL AND SYNTHETIC

note: Any object is always 3-Dimensional.
Image is always 2-Dimensional.
ie. An object from 3D is converted into an image in 2-D representation.

Objects and viewers

* Two basic entities that are part of any image-formation process are : objects, and viewers.

objects : physical structure (or instance) of an image.

* We form objects by specifying the positions in space of various geometric primitives (eg points, Lines, & polygons)
* In most graphics system, the vertices are sufficient to define most objects. eg: Line can be specified by two vertices.

viewers : one who forms the image of the objects.

* objects exists in real world & it is 3D. viewer sees an object in 2D (image)
* eg for viewer : Human, camera, digitizer.

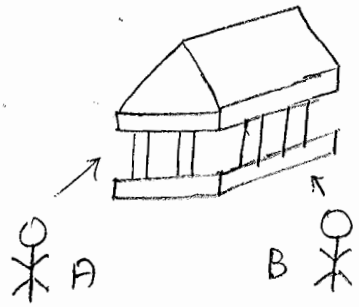
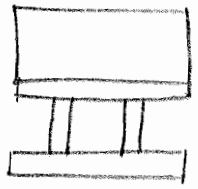


fig (a)

A

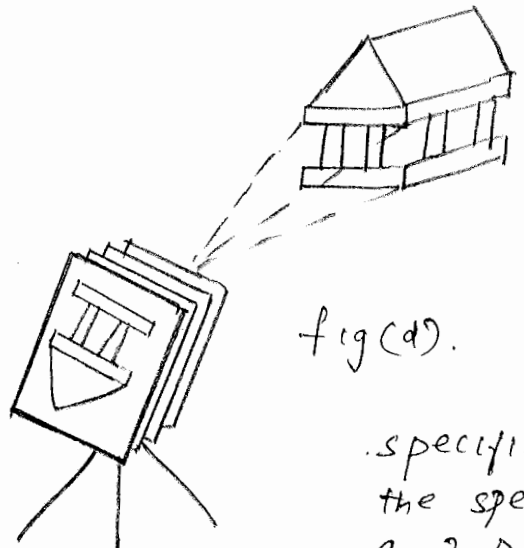
fig (b)



B

fig (c).

* Fig b shows the image seen by ~~per~~ human A. Fig c. shows the image seen by human B.
* Fig d shows the image seen by camera.



camera system

fig (d).

Image Formation

It is the process by which the specification of the object is combined with the specification of the viewer to produce a 2-Dimensional image.

Light and images

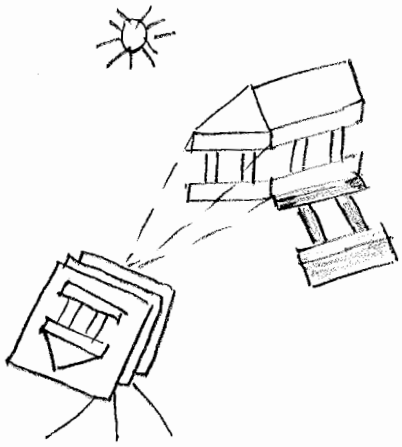
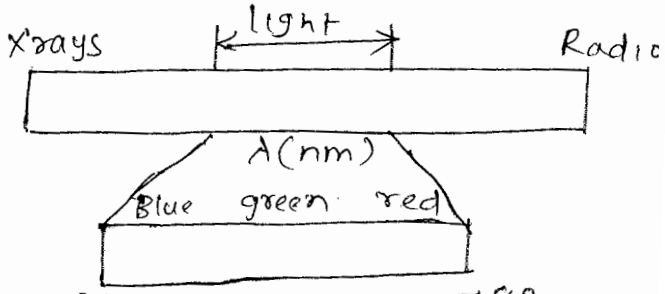


Fig: A camera system with an object and

* If there are no light sources, the objects would be dark, and there would be nothing visible in our object
* Light from the source strikes various surfaces of the object, & portion of reflected light enters the camera through the lens.
* Light is a form of electromagnetic radiation.
* portion of this spectrum is visible ~~for~~ & is called visible spectrum.
* visible spectrum has wavelengths in the range of 350 to 780 nm & is called (visible) Light.



note: Electromagnetic radiations (or energy) travels in the form of waves & measured in the form of wavelength.
The electromagnetic spectrum includes radiowaves, infrared, & a portion is our visual system.

image formation models

* Ray tracing and photon mapping are image formation techniques that are based on light source, reflection, and luminousity of the object.

IMAGING SYSTEMS

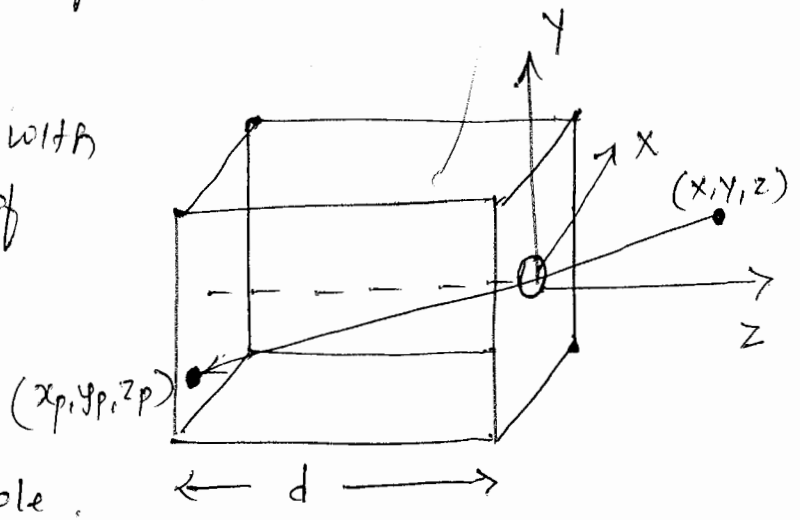
* Two physical imaging systems are
  - → pinhole camera
  - → Human visual system.

The pinhole camera

IMAGING SYSTEMS.

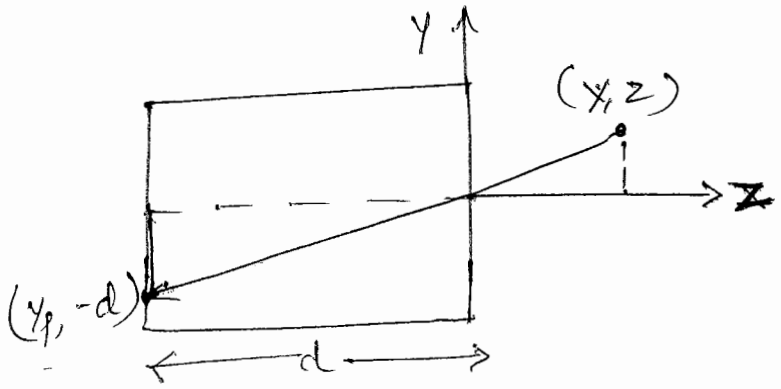
↳ pinhole camera
↳ Human visual system.

pinhole camera.



* pinhole camera is a box with small hole in the center of one side of the box.
The film is placed inside the box on the side opposite to pinhole, at a distance d from pinhole.
(hole will be so small)

* Fig shows sideview of pinhole camera.



To calculate where the image of the point (x,y,z) is on the film plane;
Two triangles are similar as shown above.

$$\therefore \frac{y_p}{y} = \frac{-d}{z} \Rightarrow \boxed{y_p = -\frac{y}{z/d}}$$

A similar calculation using top view yields

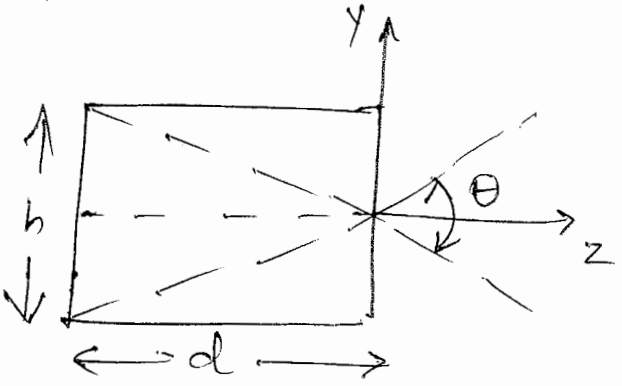
$$\boxed{x_p = \frac{x}{z/d}}$$

the point $(x_p, y_p, -d)$ is called the projection of the point (x, y, z)

The field, or Angle, of view

It is the angle made by the largest object that our camera can image on its film plane.

angle of view $\theta = 2\tan^{-1}\frac{h}{2d}$.



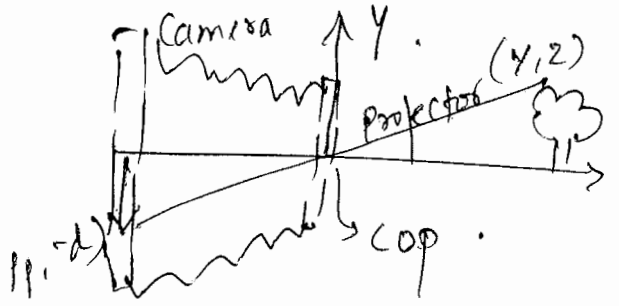
The ideal pinhole camera has an infinite depth of field. ↳(d).

Disadv of pinhole camera.

1. pinhole is so small that it admits only a single ray from a point source. Almost no light enters the camera.
2. cannot be adjusted to have different angle of view (ie no zoom in or zoom out)

Some definitions.

1. projector: We find the image of an point on the object on the virtual image plane by drawing a line called projector, from the point to the center of lens or centre of projection (COP)

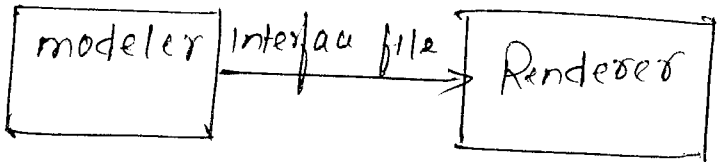


In our synthetic camera,

2. projection plane: virtual image plane that we have moved in front of the lens is called projection plane.

The modeling - Rendering paradigm.

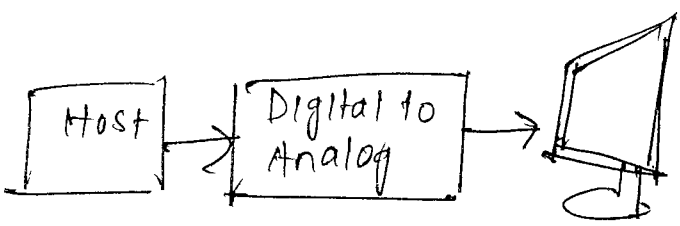
* Image formation is a two-step process (as shown in fig)
  - → modelling
  - → Rendering



* We might implement the modeler and renderer with different software and hardware.
* modeling involves designing and positioning our objects. & we dont work with detailed images of the objects here.
* Rendering involves adding light sources, material properties, and a variety of other detailed effects to form a production quality image.

GRAPHICS ARCHITECTURES.

Early graphics system: They used gen. purpose comp. with std von newmann architecture. It had single processor that processes single instr^n at a time Display in these systems was based on a calligraphic CRT display

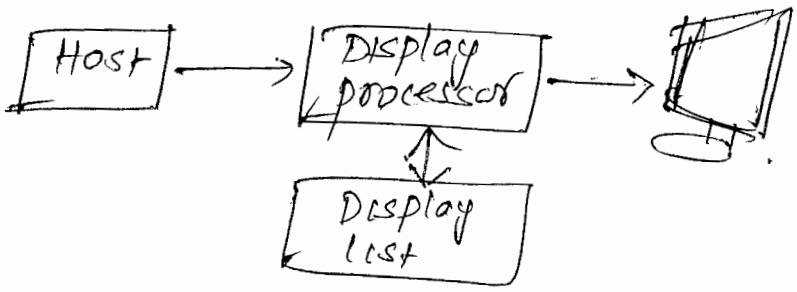


Job of Host comp was to run the applic^n prg and to compute end points of the line segments. CRT display includes necessary circuitory to generate lineseg. connecting two points.

Disadv: * Want to do both Job
* SLOW (single instr^n at a time)

Display processors. (random based)

* fig shows Display processor architecture.



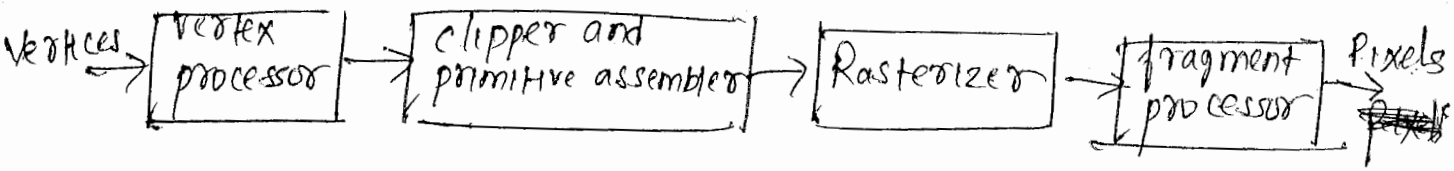
* relieves the gen. purpose processor from task of refreshing the display continuously.
* these display processors included instructions to display primitives on the CRT.
* instructions to generate the image could be assembed once in the host & sent to the display processor, where they were stored in display processors own m/m as display list or display file (random based)
  - ↳ Adv.

* Disadv.
  - → complex images can not be displayed (∵ difficult to convert them into instructions).
  - → Transformation is difficult.

Pipeline Architectures. (or)

(Graphics pipeline)

* fig shows four major steps in imaging ~~but~~ process in case of geometric or graphic pipeline.



* 4 major steps in imaging process.
  1. vertex processing.
  2. clipping and primitive assembly.
  3. Rasterization.
  4. Fragment processing.

note: each object comprises a set of geometrical primitives, each primitive comprises a set of vertices.

Vertex processing.

* It carries two functions.
  → coordinate transformations
  → compute color for each vertex.
* We can represent each change of coordinate systems by a ~~vector~~ matrix. We can represent successive changes in coordinate systems by multiplying, or concatenating the individual matrices into a single matrix.
* Assignment of vertex colors can be simple or complex.

clipping and primitive assembly.

* checks whether the matrix is the limit of our window and assemblies into appropriate format so that it fits to our window. If the image is too large - it clips it.
* o/p of this stage is a set of primitives whose projections can appear in the image.

Rasterization.

* Here the primitives ~~the~~ represented in terms of their vertices are processed to generate pixels in the frame buffer.
* o/p of the rasterizer is a set of fragments for each primitive.

Fragment processing.

* Takes the fragments & updates the pixels in the frame buffer to accomodate transformations.

Computer Graphics.

Q: Explain 7 major groups of Graphic functions.

1. primitive functions.
2. Attribute functions.
3. Viewing functions.
4. Transformation functions.
5. Input functions.
6. Control functions.
7. Query functions.

primitive functions.

* They define the low level objects or atomic entities that our system can display.
Depending on the API, the primitives can include points, line segments, polygons, pixels, text, etc...

Attribute functions.

* Gives attributes to our primitives.
They allow us to perform operations ranging from chosing the color with which we display line seg, to picking a pattern to which to fill the inside of a polygon, to selecting a typeface for the titles on a graph.

Viewing functions (specifies various views)

* It describes the synthetic camera.
ie camera's position & orientation.
This process will not only fix the view but also allow us to clip out objects that are too close or too far away.

Transformation functions.

* Allows us to carry out transformations of objects, such as rotation, translation, & scaling.

input functions

* For interactive applications, we need these to deal with the diverse forms of i/p that characterize modern graphic systems.
These functions deal with devices such as k/b, mice & data tablets.

control functions

* enable us to
  - -> communicate with the window system.
  - -> ~~to~~ initialize our programs.
  - -> deal with any errors that takes place during the execution of our program.

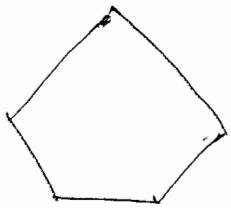
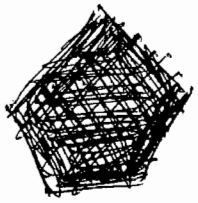
query functions.

* Allow us to retrieve information about o/p device (or system) for eg: how many colors are supported, or size of the display.
* Also allow us to know the camera parameters or values in the frame buffer.

Q: Explain polygon basics & different types of polygons in open GL.

* polygon is an object that has
  - -> A border that can be described by Line Loop.
  - -> well defined interior.
* performance of graphics system is characterised by no. of polygons per second that can be rendered.
* we can render ~~a polygon~~
  - -> only its edges
  - -> its interior with solid color or pattern.

* big: methods of rendering a polygon

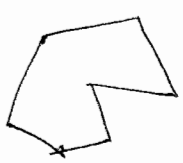




etc ...

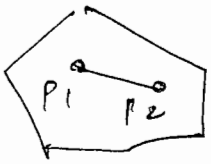
* There are 3 properties of a polygon.
  - → simplex
  - → convex
  - → flat.

→ cost of testing whether a polygon is simple or not is very high.

In 2D, if no two edges of a polygon cross each other, we say its a simple polygon.

eg   → non simple

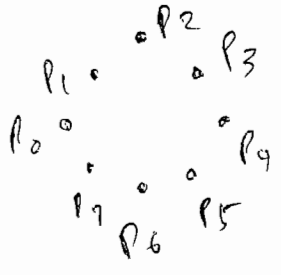
An object is ~~con~~ convex if all points on the line segment b/w any two points inside the object or on its boundary, are inside the object

eg 

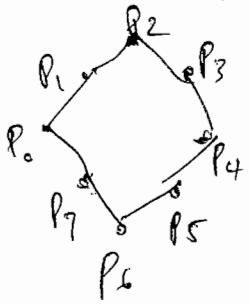
Flat:
↳ It means what is our dimension (2D or 3D)

Polygon Types

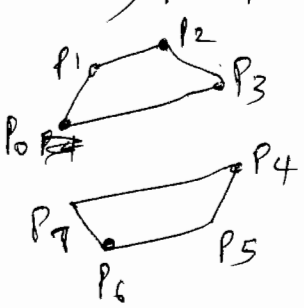
1.) GL_POINTS



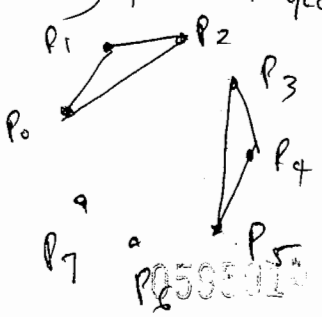
2.) GL_POLYGON



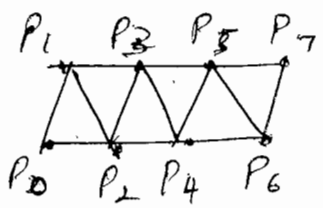
3) GL_QUADS

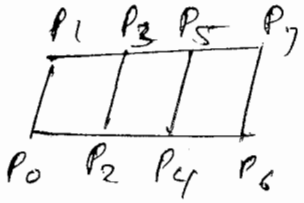


4.) GL_TRIANGLES

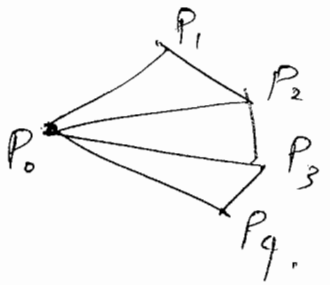


GL_TRIANGLE_STRIP . GL_QUADSTRIP





GL_TRIANGLE_FAN



polygons (GL_POLYGON)

* successive vertices define line segments, & a line segment connects the final vertex to the first.
* Interior is filled according to the state of relevant attributes.
* most graphics systems allow us to fill the polygon with color or pattern or to draw the lines around the edges but not to do both.
* we use the func. glPolygonMode to tell what we want.
* To do both, we have to render it twice, once in each mode.

Triangles & quadrilaterals (GL_TRIANGLES, GL_QUADS).

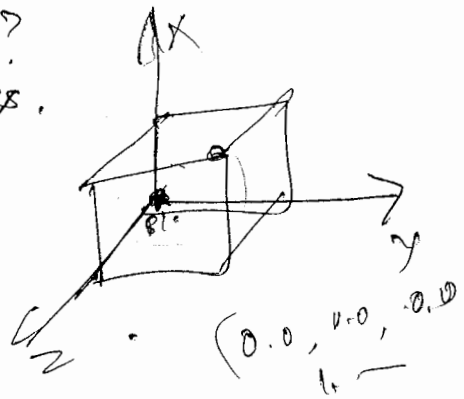
* These are special cases of polygons.
* Triangle → successive group of 3 vertices
* quadrilateral → 1 — 4
* using these types may lead to a rendering more efficient.

strips & fans (GL_TRIANGLESTRIP, GL_QUAD_STRIP, GL_TRIANGLE_FAN).

* In triangleStrip - each additional vertex is combined with the prev. two vertices to define new triangle,
* for quad strip - we combine two new vertices with prev. two vertices
* triangle fan - Here one point is fixed. next two points determine the first triangle, & subsequent triangles are formed from one new point, the prev point, & the fixed point.

Q: Explain RGB color mode & Index color mode.

~~RGB color~~. How color is handled in graphics system from programmer's perspective?
there are two diff approaches.
→ RGB-color model
→ Indexed-color model.



RGB-color model.

* Here, there are conceptually separated buffers for red, green, & blue images.
* Each pixels has separate red, green, and blue components that corresponds to locations in memory (refer fig)
* Typically, each pixel might consists of 24 bits (3 bytes) 1 byte for each of red, green & blue. Here we can specify 2^{24} different possible colors (referred as 16M colors) M denotes 1024^2.

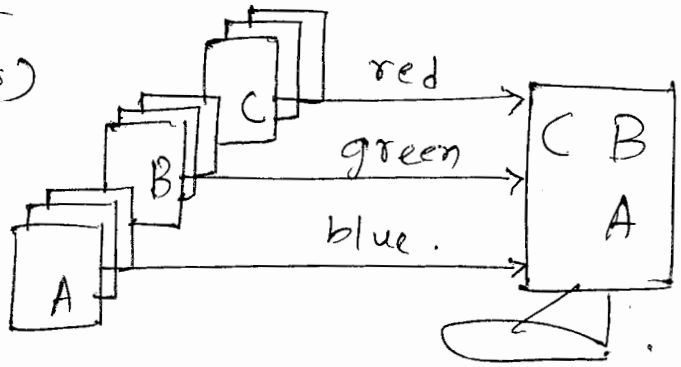


fig: RGB Color monitor.

* We specify color components in our API using color cube ~~as color components~~. as a number b/w 0.0 & 1.0 where 1.0 ⇒ max. (saturated), 0.0 ⇒ zero value of that primary.

Ex. to draw in red; we issue following fun. call.

```
glColor3f (1.0, 0.0, 0.0);
```

↳ sets current drawing color to red.

~~later,~~

* there is 4-color system also (RGBA). ~~system~~ 4th color (A, or alpha) is also stored in frame buffer as are the RGB values.

uses: for creating fog effects, combining images.

A = 0.0 ⇒ ~~opacity (opaque)~~ fully transparent

A = 1.0 ⇒ fully opaque.

eg: `glclearcolor (1.0, 1.0, 1.0, 1.0)`

defines an RGB-color clearing color as white & opaque.

Indexed-color model.

* used for limited-depth pixels.
* Here color is selected by interpreting pixels as indices into a table of colors rather than as color values.
* suppose ovre frame buffer has k bits per pixel. Each pixel value or index is an integer b/w 0 and $2^k - 1$. suppose that we can display colors with a precision of m bits (ie we can chose from 2^m reds, 2^m greens, an & 2^m blues) Hence we can produce any of 2^{3m} colors on the display, but the frame buffer can specify only 2^k of them.

we handle the specification thru' a user defined color look-up table that is of size $2^k \times 3m$ (refer fig)

I/p	Red	Green	Blue
0	0	0	0
1	2^m-1	0	0
:	0	2^m-1	0
:	1	1	1
2^k-1	1	1	1

once user has constructed this table, they can specify a color by its index, which points to appropriate entry in color-lookup table (refer below fig)