Computer Graphics

1.1 machine used — graph/chart — pics (picture/image)
↳ end product

tool

Q Computer Graphics is the creation of Pictures with the help of a computer. The end product of the CG is a picture. it may be a business graph, drawing and engineering.

1.2 Application
① CAD - computer aided design (design, 2D, 3D,
② Presentation - (education, business, science).
③ Computer Art - Drawing, cartoon.
④ Entertainment - motion Picture, game, movie
⑤ Education - education model
⑥ Graphical User Interface : Click.
⑦ Training - Specialised system (simulators)

1.3 Hardware & Software

Input Device
- → Touch Panel
- → Light Pens
- → Graphics tablets [voter id sign system]
- → Mouse
- → Joystick
- → Keyboard
- → Image Scanner (একটা glass screen) এর উপর যা দেওয়া হয় তাই screen এ উঠে

Output devices
- → CRT (cathod Raytube)
- → Vector Scan Display (Random Scan Display)
- → Raster Scan Display
- → Coloured Monitors
- → LCD, TFT Screen
thin fill transistor

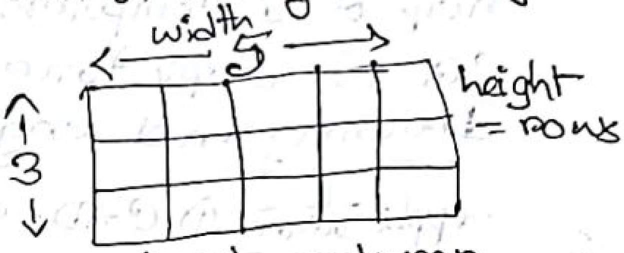
Software (2D)
- → Photoshop
- → Coral DRAW
Logo, poster
- → Maya 3D (Autodesk Maya)
- → CAD

2.1

Pixel: → Picture element, basic building block
→ represected by a dot or square
→ basic building block of a digital image

Resolution: Width × height

Megapixel: $\frac{\text{Resolution}}{1 \text{ million}}$



Aspect Ratio: weight : Height
→ Maintains a balance between the appearance of an image on the screen

$\frac{1920}{1080} = 16:9$

Width × height
1920 × 1080
Column × Rows

$\frac{2138400 \text{ pixel}}{1 \text{ million}}$

$= 2.1384$ megapixel

Q: Grayscale Image: 8bpp
Aspect Ratio = 6:2
Resolution = 480000

① # of Row?
② # of Col?
③ Size of image?

$\cancel{\frac{6x}{2x}}$ $\cancel{6x \times 2x = 480000}$ $\cancel{8x^2}$

$\frac{W}{H} = \frac{6}{2}$

$W = 3H$

$3H = \frac{480000}{H}$

$H^2 = \frac{48000}{3}$

$= 400$

$W \times H = 480000$

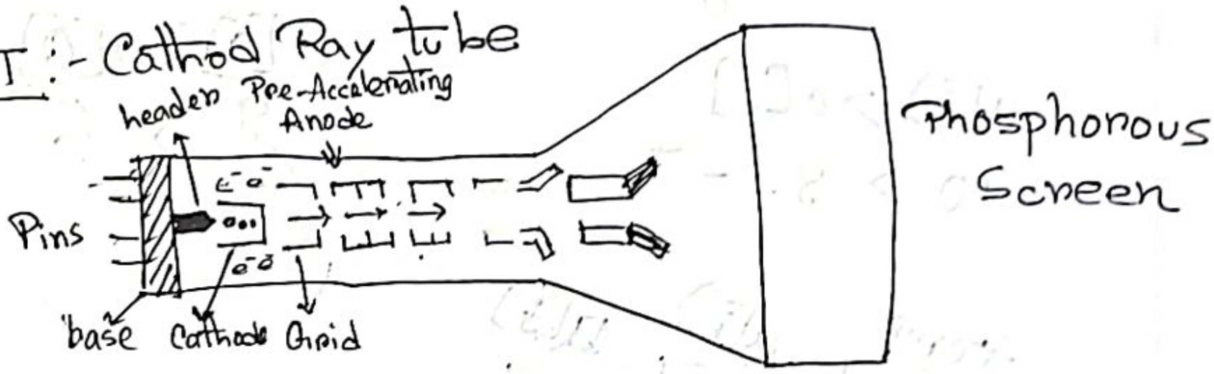
$\therefore W = \frac{480000}{H}$

$W = 1200$

bit per pixel

Size = Row × Col × bpp
= (400 × 1200 × 8) bits
= (400 × 1200) byte
= $\frac{400 \times 1200}{1000}$ kb.
= 480 kb

CRT :- Cathod Ray tube



electric signal to light signal

GIF - Graphic Interchange formate :- 8 bit color - 256
JPEG - Joint photographic export graph
↳ 24 bit color → lossy compression technique
improved
PNG → Protable Network graphic → 24 bit Color → browser Not supp-ort
↳ lossless compression

TIFF → Tagged image File formate → Desktop publis
↳

5. Calculate (2, 2) and (6,6).

$m = \frac{y_2 - y_1}{x_2 - x_1} = \frac{6-2}{6-2} = \frac{4}{4} = 1$

x	y	Round	Pixel
2	2		(2, 2)
3	3		(3,3)
4	4		(4,4)
5	5		(5,5)
6	6		(6,6)

PADIM

CMD

Computer Graphics - CH-1

What: Refers to creation, manipulation, and display of images and visual content using computer technology. It involves the use of algorithms and mathematical calculations to create and modify images, animations and 3D model on a computer screen or via a printer, or plotter.

Computer graphics can be divided into two main areas:

Raster Graphics: Created by manipulating individual pixels.

Vector Graphics: Created by using mathematical calculation to create and modify images, animations and 3D models on a computer screen.

Applications of Computer graphics :-
1) Video games, 2) Movie special effects,
3) medical imaging, 4) architecture visualization,
5) Data visualization, 6) Virtual & augmented Reality.
7) Computer Aided design, 8) Presentation graphics
9) Entertainment, 10) Education & Training.
11) Image processing 12) GUI
13) Geographic info system, GPS, 14) Deepfakes.

Scan-line Conversion Algorithm
Line drawing algorithm - DDA, Midpoints,
Circle " " - midpoint
Clipping Algorithm - Cohen-Sutherland, Cyrus-beck
Polygon - Clipping Algorithm - Sutherlan-Hodgman
Transformation - translation, Rotation, Scaling
Projection - Prespective Projection, Parallel-projection
Color model - RGB, HSV, HSL

RGB: 0-255
HSV: hue, saturation, value
HSL: hue, saturation, lightness

hue = 0 to 360
saturation = purity = 0% - 100%
value = brightness = high = white
low = black

lightness = light = 0% = black,
100% = white

CRT Display   chat GPT

narrow at one end & opens to a flat screen in the other hend.

phosphor dots : E.CFDH

E-beam Gun → Control Grid → Focusing System → Deflection Coils → High Positive V coating.

Advantage
1. greater resolution
2. less cost
3. widest viewing angle

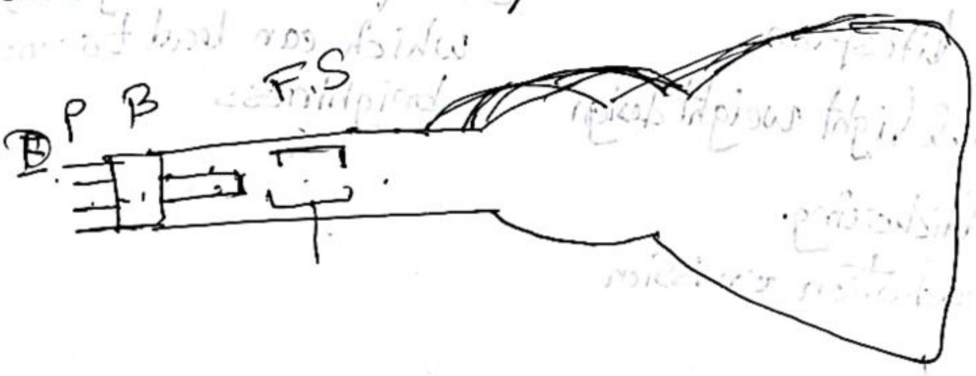
disadvantage
1. Thickness largerthan LCD, Plasma
2. Less view area compared to monitor size
3. Cannot be used for smaller displays like watches, calculator, portable device.
4. Fragile & bulky

Flat Panel Display:-

Emissive Display - emmitters.

21
CSF3R41 C
3R11
51 SP
61

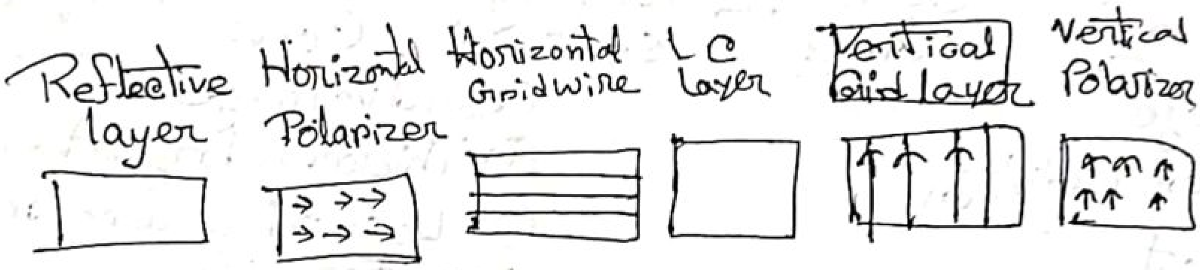
Non-emissive display. (non-emmitters)



LCD - Liquid Crystal ~~diode~~ display :-

① thin, flat, ② uses very small amounts of electric power

③ Color display by incorporating color filters.



Twisted Nematic Cell.

Parpendicular - homeotropic

Parallel - homogeneous

~~LCD~~ Advan

① Consume less power

② Can be operated on voltages as low as 2 to 3V and are easily driven by MOS IC drivers.

② long lifespan

③ Thin & light weight design

④ No flickering.

⑤ No radiation emission

Dis

① Cannot be seen in dark

② limited view angle

③ Operated in limited temp. range

④ Requires backlighting, which can lead to uneven brightness

CG $y = \frac{3}{5} \times 3 + 1$ $= \frac{2 \cdot 34}{7}$ $m = \frac{3}{5}$ $b = \frac{15}{5} - \frac{3}{5} \times 2$

Scan Conversion.

$(x_0, y_0), (x_1, y_1)$ x_1, y_1 $(\underset{x_0}{2}, \underset{y_0}{5})$ $(7, 8)$

$y = m\boxed{x} + b$

$m = \frac{y_1 - y_0}{x_1 - x_0} = \frac{\Delta y}{\Delta x}$ $\frac{3}{5}$ $\frac{6}{7}$

$\boxed{b} = y_0 - m x_0$

$x \Rightarrow$ independent $y = mx + b$

$y \Rightarrow$ dependent $x = \frac{y - b}{m}$

x	y	y (round)	pixel
2	$\frac{5}{7}$		
3			
4			

DDA

$(x_0, y_0), (x_1, y_1)$.

$m = \frac{y_1 - y_0}{x_1 - x_0}$

if $(-1 < m < 1)$

$x_{k+1} = x_k + 1$

$y_{k+1} = y_k + m$

else ~~$y_2 = y_1 + m$~~

$y_{k+1} = y_k + 1$

$x_{k+1} = x_k + \frac{1}{m}$

x	y	Round	pixel

Bresenham Algorithm

(x_0, y_0) end (x_n, y_n).

$\Delta x = x_n - x_0$

$\Delta y = y_n - y_0$

$P_k = 2\Delta y - \Delta x$

if $P_k < 0 \Rightarrow P_{k+1} = P_k + 2\Delta y$

$x_{k+1} = x_k + 1$

$y_{k+1} = y_k$

else $P_k \geq 0 \Rightarrow P_{k+1} = P_k + 2\Delta y - 2\Delta x$

$x_{k+1} = x_k + 1$

$y_{k+1} = y_k + 1$

x_{k+1}	y_{k+1}	P_k	P_{k+1}
x_0	y_0	Initial	—
— —	—	— —	— —
—	— —	— —	— —

CG

Applications:-

① Computer vision:- Extracting info from the contents of an input image or video frame eg: face recognition

② Computer graphics:- creating image from scratch using computer, eg: animated movie

③ Digital image processing:- Processing raw input image to perform different operations, eg:- apply filter on an image.

④ Movies / Special effects

⑤ Video games and simulation

⑥ CAD - CAM & design

⑦ Architecture

⑧ Virtual reality

⑨ Medical imaging

⑩ Education

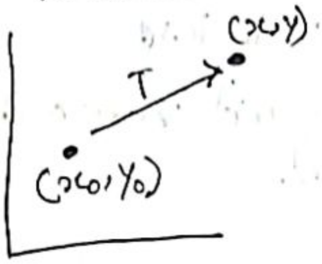
⑪ Geographic info systems & GPS

⑫ Any display

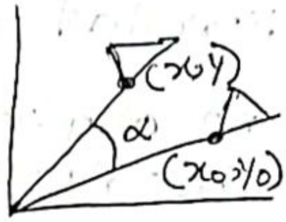
⑬ Deepfakes

Transformation

⇒ Translation



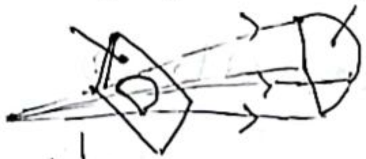
⇒ Rotation



⇒ Scaling

⇒ Perspective projection

object



sight line

⇒ Parallel projection

Perspective projection

Color Model
RGB
HSV
HSL

Illumination Models
⇒ Global illumination
⇒ Diffuse "

Shading:-
~~White~~ flat
Gouraud
Phong

L-2 Primary o/p device in graphics system is a video monitor

A display device is a device for visual ~~or~~ presentation of images (including text)/ video acquired, stored or transmitted in various forms.

analog vs Digital ⇒ tube vs flat panel display,

Analog display devices
- Oscilloscope tube
- T V CRTs

Digital display devices
- LED
- LCD
- Thin-Flim Transistor LCD
- PDP (Plasma display panels)

Other
- Electronic paper
- Nanoelectronics

Active display devices: emits electromagnetic radiation

Passive display ": reflect or modulate light

Pixel Resolution:- no. of pixel per unit video display

Video Graphics Array (VGA).

Display size:- distance diagonally

Viewing angle:- angle from which screen from side

- It is larger for CRT as compared to LCD

Response time:- Minimum time to change a pixel's color or brightness.

Persistence:- How long a phosphor continue to emit light

Aspect Ratio: Ratio of vertical points to horizontal points to produce equal length lines in both direction on screen

17-5-2020

Brightness: amount of light emitted from display

$\boxed{\text{CRT}}$ Glass tube narrow at one end & opens to a flat screen at other end.

Single electron gun for single-color monitor

display screen covered with ~~a~~ tiny ~~posphor~~ phosphor dots

Components of CRT

E-Gun → Control Grid → Focusing System → Deflection Coil → High positive coating

CRT

Advantages

1. Greater Resolution
2. Widest viewing angle
3. Cheap Compared to LCD, PLASMA display

Disadvantages

1. Thickness much larger
2. Cannot be used for smaller display like watches, calculators, portable devices
3. View area is less than offered monitor size
4. More fragile & bulky

Flat Panel Display:- Reduced volume, weight and power requirement compared to CRT. 2 types → LED, PDP

1. Emissive display :- Convert electrical energy to light.
2. Non-emissive " :- Use optical effects to convert sunlight or light from some other source into graphics pattern : LCD.

LCD :- thin, flat with any amount of pixel arrayed in front of light source / reflector. → ① Transmissive display (rear light) ② Reflective display (external light)

① Twisted nematic cell helical

Advantages

① Consumes only microwatts of power over a thousand times less than LED

② Can operate on voltages as low as 2 to 3 V and easily driven by MOS IC drivers

limitations:-

① Can not be seen in dark.

② Have limited viewing angle.

③ Can be operated in limited temperature range.

Thin Film Transistor (TFT)

TFT LCD -> Active Matrix LCD
↳ both flat panel display & projectors.

special field effect transistor made by depositing thin films for metallic contacts semicondor active layer & dielectric layer

Advantage
1) Fast response time
2) Lower cost of production
3) Less noise

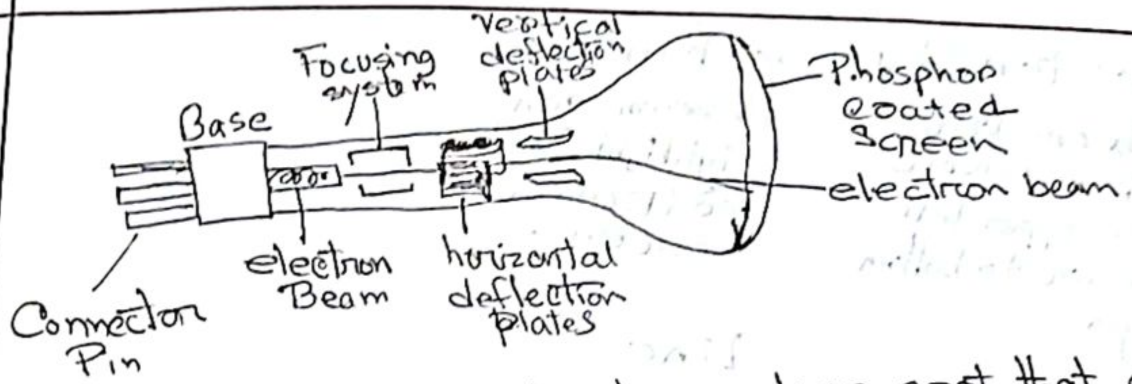
Disadvantage
1) Limited viewing angle especialy in vertical direction
2. Unable to display full color

PD/ Plasma display :- Flat panel display, suitable and used for large TV display

Advantage:-
1) Supports large display
2) Overall thickness of monitor is less than 10 cm. & can be installed on a wall.
3) Faster response time
4) Greater Color Spectrum
5) Wider viewing angle.

Disadvantage:-
1). Costly compared to CRT, LCD
2) Burn-in problem Static images can leave a permanent mark on screen.

CRT



Pixel: smallest size object or colour spot that can be displayed and addressed on a monitor.

True Color:- 3byte info - Red, Green & blue.

Memory in MB =

Raster is a matrix covering the screen area and is composed of raster lines

On a black-white system with one bit per pixel, the frame buffer is commonly called a bitmap.

Bitmap:- monochrome system, each bit is 1 or 0

Raster-scan; Pixmap :- for Color monitors.

Random-scan :- Picture is stored in a display list,
↳ electron beam directly draws the picture in any specific order.

Refresh rate depends upon the size of file

Raster ~~Ras~~ vs Random

Raster	Random
E-beam traces entire screen from upper left corner to bottom right	Ebeam can highlight Random position on screen.
Pixel	Line
Frame Buffer	Display file
High Color depth	Lesser color and shades
Supports animation	Not Supporting animation
Entire screen Refresh	Only selected portion are redrawn
Shadow Masking	Beam Penetration

Pseudo Code $x=0,\ y=r,\ d=1-r$

```
while (x<y)
    if (d<0) # East
        d = d + 2x + 3
        x = x + 1
    else # SE
        d = d + 2x - 2y + 5
        x = x + 1
        y = y - 1
```

Digital differential Analyzer.

$y = mx + c$

DDA $m = \frac{\Delta y}{\Delta x}$

if $-1 < m < 1$

$x = x + 1$

$y = y + m$

else

$y = y + 1$

$x = x + \frac{1}{m}$

Bresenham Algorithm

$\Delta x, \Delta y$

$P_k = 2\Delta y - \Delta x$

$P_k \geq 0$, $P_{k+1} = P_k + 2\Delta y - 2\Delta x$

$x = x + 1$

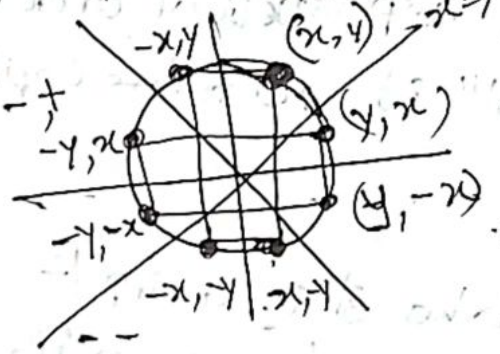
$y = y + 1$

$P < 0$ $P_{k+1} = P_k + 2\Delta y$

$x = x + 1$

$y = y$

$P_k \quad P_{k+1} \quad x_{k+1} \quad y_{k+1}$



translation

$A + T$

Rotation

$$[x' \ y'] = [x \ y] \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$$

Scaling

$$[x' \ y'] = [x \ y] \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix}$$

$x' = x . S_x$

$y' = y . S_y$

homogeneous co-ordinate for translation

$$T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{bmatrix}$$

for rotation

$$R = \begin{bmatrix} \cos x & \sin x & 0 \\ -\sin x & \cos x & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

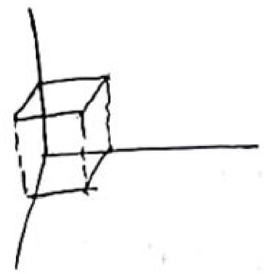
for Scaling

$$S = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

CH-10 3D Object Representation

Plane equation: $Ax + By + Cz + D = 0$

3 points (x_1, y_1, z_1)
(x_2, y_2, z_2)
(x_3, y_3, z_3).



If $Ax + By + Cz + D = 0$ → on plane
$\neq 0$ not on plane
$> 0 \rightarrow x, y, z$ is outside of plane
$< 0 \rightarrow$ inside of plane

$$Ax + By + Cz = -D$$

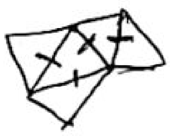
$$\frac{A}{D}x_1 + \frac{B}{D}y_1 + \frac{C}{D}z_1 = -1$$

$$\frac{A}{D}x_2 + \frac{B}{D}y_2 + \frac{C}{D}z_2 = -1$$

$$\frac{A}{D}x_3 + \frac{B}{D}y_3 + \frac{C}{D}z_3 = -1$$

Polygon mesh:-

every mesh line joins 2 polygon.. triangle used to represent any surface



$$① \; -\begin{vmatrix} y_1 & 1 & z_1 \\ y_2 & 1 & z_2 \\ y_3 & 1 & z_3 \end{vmatrix}$$

By Cramers rule

$$D = -\begin{vmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{vmatrix}$$

$$A = \begin{vmatrix} 1 & y_1 & z_1 \\ 1 & y_2 & z_2 \\ 1 & y_3 & z_3 \end{vmatrix} \overset{②}{=} -\begin{vmatrix} y_1 & y_2 & y_3 \\ 1 & 1 & 1 \\ z_1 & z_2 & z_3 \end{vmatrix}$$

$$B = \begin{vmatrix} x_1 & 1 & z_1 \\ x_2 & 1 & z_2 \\ x_3 & 1 & z_3 \end{vmatrix} \quad C = \begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix}$$

$$A = \begin{vmatrix} 1 & y_1 & z_1 \\ 1 & y_2 & z_2 \\ 1 & y_3 & z_3 \end{vmatrix}$$

$$= -\begin{vmatrix} y_1 & 1 & z_1 \\ y_2 & 1 & z_2 \\ y_3 & 1 & z_3 \end{vmatrix}$$

$$= -\begin{vmatrix} y_1 & y_2 & y_3 \\ 1 & 1 & 1 \\ z_1 & z_2 & z_3 \end{vmatrix}$$

$$= -\{y_1(z_3 - z_2) - y_2(z_3 - z_1) + y_3(z_2 - z_1)\}$$

$$= y_1(z_2 - z_3) + y_2(z_3 - z_1) + y_3(z_1 - z_2)$$

$$\therefore B = \begin{vmatrix} \cancel{x_1} & \cancel{x_2} & \cancel{x_3} \\ \cancel{1} & \cancel{1} & \cancel{1} \\ \cancel{z_1} & \cancel{z_2} & \cancel{z_3} \end{vmatrix}$$

$$= \cancel{x_1(x_3 - x}$$

এত change করার লাগেনা। যে line ধরে করাতে চাই ঐ ~~লাইন~~ ~~soo~~ ঐটা রেখে করলেও হয়।

So, no need to switch places

$$B = \begin{vmatrix} x_1 & 1 & z_1 \\ x_2 & 1 & z_2 \\ x_3 & 1 & z_3 \end{vmatrix} \leftarrow B = \begin{vmatrix} x_1 & 1 & z_1 \\ x_2 & 1 & z_2 \\ x_3 & 1 & z_3 \end{vmatrix}$$

$$= z_1(x_2 - x_3) - z_2 \cdot (x_3 - x_1) + z_3(x_1 - x_2).$$

$$= -\begin{vmatrix} x_1 & z_1 & 1 \\ x_2 & z_2 & 1 \\ x_3 & z_3 & 1 \end{vmatrix}$$

$$= \begin{vmatrix} z_1 & x_1 & 1 \\ z_2 & x_2 & 1 \\ z_3 & x_3 & 1 \end{vmatrix}$$

$$= -\begin{vmatrix} z_1 & 1 & x_1 \\ z_2 & 1 & x_2 \\ z_3 & 1 & x_3 \end{vmatrix}$$

$$C = \begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix}$$

$$= x_1(y_2 - y_3) - x_2(y_3 - y_1) + x_3(y_1 - y_2).$$

$$D = \begin{vmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{vmatrix}$$

$$= x_1 (y_2 z_3 - z_2 y_3) - x_2 (z_1 y_3 - y_1 z_3) + x_3 (y_1 z_2 - y_2 z_1)$$

quadric surface
2nd degree

Sphere, Ellipsoid, Torus

Sphere: $x^2 + y^2 + z^2 = r^2$

$x = r\cos\theta\cos\phi$

$y = r\cos\phi\sin\theta$

$z = r\sin\phi$

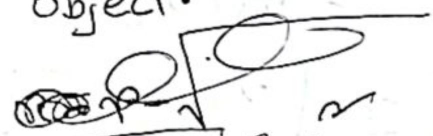
Ellipse: $\left(\frac{x}{r_x}\right)^2 + \left(\frac{y}{r_y}\right)^2 + \left(\frac{z}{r_z}\right)^2 = 1$

$x = r_x \cos\phi\cos\theta$

$y = r_y \cos\phi\sin\theta$

$z = r_z \sin\phi$

Torus: take an ellipse or circle which is rotated with some angle along axis at constant then we get a nutshell shaped object.



$r - \sqrt{\left(\frac{x}{r_x}\right)^2 + \left(\frac{y}{r_y}\right)^2} + \left(\frac{z}{r_z}\right)^2 = 1$

Parametric eqn:-

$x = r_x(r + \cos\phi)\cos\theta$

$y = r_y(r + \cos\phi)\sin\theta$

$z = r_z \sin\phi$

$-\pi \le \theta \le \pi$

$-\pi < \phi \le \pi$

angular phy

Superquadric are formed by incorporating additional parameters into the quadric equations to provide increased fexibility for adjusting object shapes.

The number of additional parameters used is equal to dimension of object. ∴ One parameter for curves & 2 parameters for surface.

<u>Super ellipse</u> :- $\left(\frac{x}{r_x}\right)^{2/s} + \left(\frac{y}{r_y}\right)^{2/s} = 1 \quad s = 1.$

$x = r_x \cos^s \theta$

$-\pi \leq \theta \leq \pi$

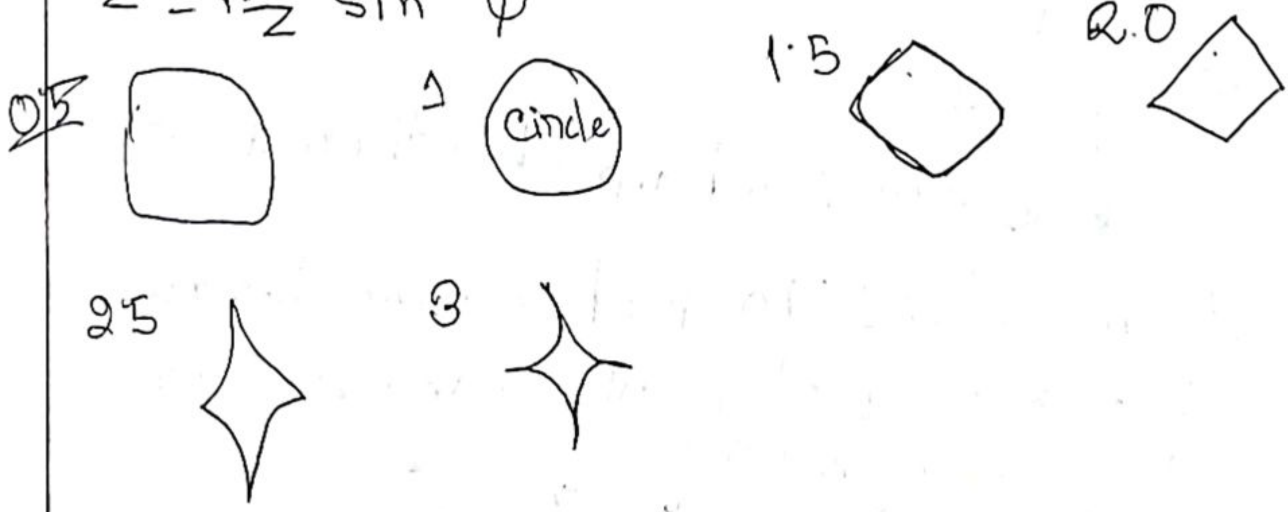
$y - r_y \sin^s \theta$

<u>Superellipsoid</u> :- $\left[\left(\frac{x}{r_x}\right)^{2/s_2} + \left(\frac{y}{r_y}\right)^{2/s_2}\right]^{s_2/s_1} + \left(\frac{z}{r_z}\right)^{2/s_1} = 1$

$x = r_x \cos^{s_1}\phi \cos^{s_2}\theta \qquad -\pi/2 \leq \phi \leq \pi 2$

$y = r_y \cos^{s_1}\phi \sin^{s_2}\theta \qquad -\pi \leq \theta \leq \pi$

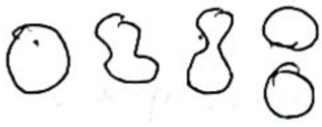
$z = r_z \sin^{s_1}\phi$

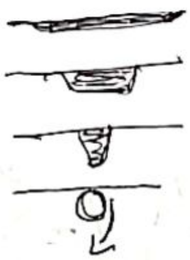


Blobby Objects:- Some object do not maintain and fixed shape, but change their surface characteristics in certain motion or when in proximity to other objects

Example: Molecular structure, melting objects, muscles in human body and water droplets.

Water Molecular Structure→





$$F(x,y,z) = \sum_k b_k e^{-a_k r_k^2} - T$$

$$r_k^2 = \sqrt{x_k^2 + y_k^2 + z_k^2}$$

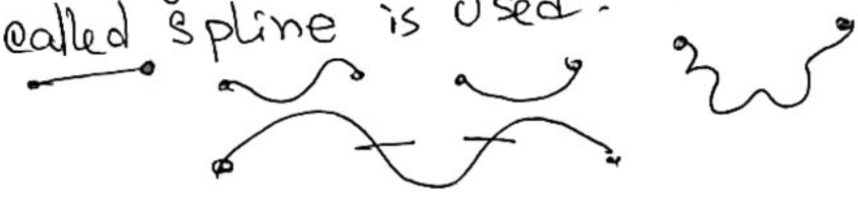
T = thourshold value.

a, b = r the paramater used to adjust amount of blobiness.

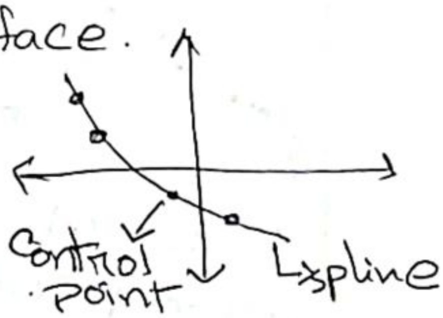
① Bump ② Dent

If b is +ve then bump, b⊖ then dent

Spline Curve: To produce smooth curve through a set of points a fexible strip called spline is used.

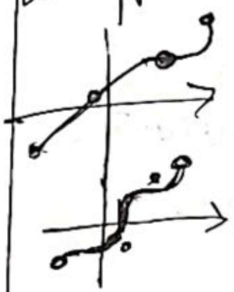


Spline: A spline curve is mathematical representation for which it is easy to built an interface that allows users to design and control the shape of complex curve & surface.



The spline curve is defined, modified and manipulated by ~~the~~ control point (the set of co-ordinate position which indicates the general shape of the curve)

2 types of Spline



→ Interpolating / Interpolation Spline → curve passes through all the control point → animation & camera motion

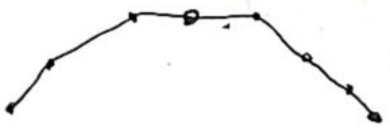
→ Approximating / approximation spline. → does not pass through all control point

I.S: Must take all the control points in consideration

A.S:- Not compulsory to take all control points in curve.

Bazier Curve:- Is a spline curve

→ Approximation ✓ Bazier is Approximation spline curve



→ Interoplation

Bezier curve ① is an approximation spline curve.

② Is a parametric curve

t = % of travelled location.

$u =$

$0 \le u \le 1$

③ We can get n degree polynomial (constent variable) with $n+1$ control point

$P(x) = a_0 x^n + a_1 x^{n-1} + a_2 x^{n-2} + \ldots$

$P(x) = n$ highest degree

$P_1(x) = x^3 + x + 1$, highest degree 3

④ We can use Bezier Curve in CAD, typeface & drawing use.

degree
2 = binomial
3 = quadratic
4 = Cubic

⑤ It is easy to implement

Quadratic bezier curve derivation :-

3 control point

Let parametric curve u, $0 \le u \le 1$ and Q_0 & Q_1 points are lying btn

$P_0 \to P_1$ & $P_1 \to P_2$.

$Q_0 = (1-u)P_0 + uP_1$ --- ①

$Q_1 = (1-u)P_1 + uP_2$ --- ②

Suppose $C(u)$ is point betn $Q_0 \to Q_1$

$$P(t) = \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{bmatrix}$$

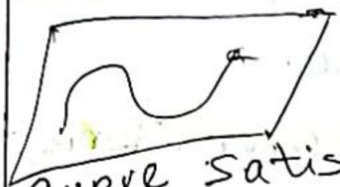
$$C(u) = (1-u) Q_0 + u Q_1$$

$$= (1-u)(1-u) P_0 + (1-u) u P_1 + (1-u) u P_1 + u^2 P_2$$

$$= (1-u)^2 P_0 + 2 P_1 (u)(1-u) + u^2 P_2$$

$C(u) = (1-u)^2 P_0 + 2u(1-u) P_1 + u^2 P_2$ ie Quadratic equation.

Convex hull is enclosed polygon which passes through the outermost control points



Curve satisfies convex hull property

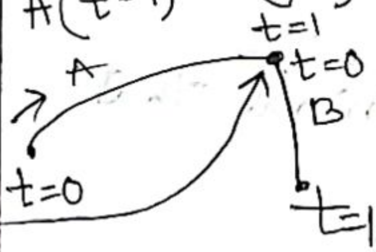
Doesn't satisfy Convex hull property

Parametric Continuity:-

OPC = Order Parametric continuity

① Zero OPC

$A(t=1) = B(t=0)$

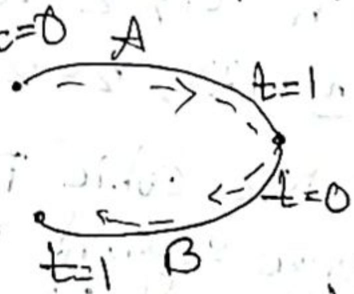


just connect 2 curves at a point that's it. C^0

no smoothness

② First OPC

$A'(t=1) = B'(t=0)$



1st order derivative of A & B, i.e. A' & B'

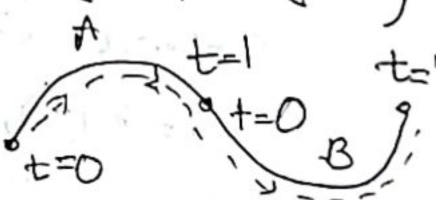
progressively smoother then zero OPC.

C^1

(mirror)

③ Second OPC.

$A''(t=1) = B''(t=0)$



2nd Order derivative of A & B i.e A'' & B''

Most smooth

highest Continuity

C^2

→ cubic polynomial parametric
~~Spline Representation~~ C^1 & C^2 Continuity

Spline: Is a flexible strip used to produce a smooth curve through a ~~deg~~ designated set of points

Splines curves:- A curve drawn in this manner

book-329 Spline Specification:- 3 equivalent method for specifying a particular spline representation

1. We ~~de~~ state the set of boundary conditions that are imposed on the spline.
2. We can state the matrix that characterizes the spline.
3. We can state the set of blending function or (basis function). ~~that determine how specified a~~

Represent in x, y, z co-ordinate -
$x = x(t)$, $y = y(t)$, $z = z(t)$.

A parametric cubic polynominal ~~that~~ defined curve segment is:

$$P(t) = [x(t) \quad y(t) \quad z(t)]$$

$$\left.\begin{aligned} x(t) &= a_x t^3 + b_x t^2 + c_x t + d_x \\ y(t) &= a_y t^3 + b_y t^2 + c_y t + d_y \\ z(t) &= a_z t^3 + b_z t^2 + c_z t + d_z \end{aligned}\right\} \text{Eq-1}$$

usually $t = [0 \ldots 1]$

$u(t) = a_u t^3 + b_u t^2 + c_u t + d_u$

1. Boundary Condition:- Starting, $x(0)$, ending $x(1)$ in $x(t)$ and on the parametric first derivatives $x'(0)$ & $x'(1)$.

$$P(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \\ d_x & d_y & d_z \end{bmatrix} = T.C$$

T : The Row matrix of Parameter t

Co-efficient Matrix

2. After step-1, ~~By~~ doing farther work on the developed matrix. $C = M_{spline} \cdot M_{geometric}$

M_{geom} = 4 element Column matrix

M_{spline} = 4 by 4 matrix that transform geometric constraints values to polynomial Co-efficient.

$P(t) = T.C$ $\quad x(t) = T \cdot M_{spline} \cdot M_{geom}$

$$P(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} M_{00} & M_{01} & M_{02} & M_{03} \\ M_{10} & M_{11} & M_{12} & M_{13} \\ M_{20} & M_{21} & M_{22} & M_{23} \\ M_{30} & M_{31} & M_{32} & M_{33} \end{bmatrix} \begin{bmatrix} P_{0x} & P_{0y} & P_{0z} \\ P_{1x} & P_{1y} & P_{1z} \\ P_{2x} & P_{2y} & P_{2z} \\ P_{3x} & P_{3y} & P_{3z} \end{bmatrix}$$

$$= T M_{spline} \begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix}$$

Control points or Control vector

M_{spline}:- translates geometric info to co-efficients.

③ Blending function, B_k

$$Q(t) = B_0(t)P_0 + B_1(t)P_1 + B_2(t)P_2 + B_3(t)P_3$$

$$Q(t) = \sum_{k=0}^{3} B_k(u)P_k$$

P_k = Control point

Cubic Spline Interpolation method

x	x_0	x_1	------	x_n
y	y_0	y_1	------	y_n

Assum $f(x)$,

⇒ $f(x)$ is a linear polynomial outside the interval (x_0, x_n)

⇒ $f(x)$ is a cubic polinomial in each of the subinterval.

⇒ $f'(x)$ & $f''(x)$ are continuous at each point.

Now, as $f(x)$ is cubic in each subinterval of ~~so~~ $f''(x)$ will be linear

$$f''(x) = \frac{x - x_{i+1}}{x_i - x_{i+1}} f''(x_i) + \frac{x - x_i}{x_{i+1} - x_i} f''(x_{i+1})$$

$Q(u) = \sum_{i=0}^{n} P_i \, B_{i,n}$ $B_{i,n} \rightarrow$ Basis

To ensure a smooth transition from on section of piecewise polynomial curve to the next we can impose various

C – Parametric Condition } Continuity

G – Geometric "

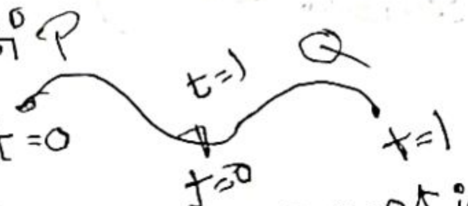
C^1

$P'(t=1) \neq Q(t=0)$

GCC:-

zero order :- Same as C^0

G^0

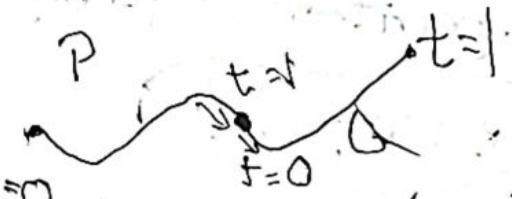


two curves sections must have same co-ordinate position

G^1 : It may be equal to C^1 when k=1 only.

$P'(x,y,z) = Q'(k*x, k*y, k*z)$

$C' \Rightarrow G'$ but $G' \neq C'$ k=2,3,..



tangent vector direction same.

Magnitude may or may not be equal

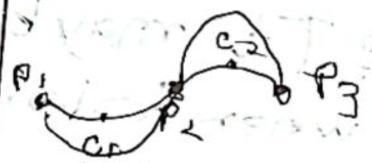
tangents are proportional

G^2 : Both first and second parametric derivatives of the two curve sections are proportional at their boundary point and tangent vector direction is same

Magnitude may or may not be same

$P''(x,y,z) = Q''(k*x, k*y, k*z)$

$C^2 \Rightarrow G^2$ but $G^2 \neq C^2$ k=1 the magnitudes same



With geometric continuity, the curve is pulled towards the section with greater tangent value

tangent

Positional (not tangent).

Tangental (not Position).

Curvature (take all in Consideration)

topic-8

Single parametric cubic spline segment

$P(t) = B_1 + B_2 t + B_3 t^2 + B_4 t^3$

$P(t) = \sum_{i=1}^{4} B t_1^{i-1} \quad t_1 \le t \le t_2$

$P(t) = [x(t), y(t), z(t)]$ Cartesian

or $[r(t), \theta(t), z(t)]$ Cylindrical

$[r(t), \theta(t), \phi(t)]$ Spherical

topic-9: Natural Cubic Spline

Given, $t_0 < t_1 < \dots < t_n$ we define the cubic spline with $S(x) = S_i(x)$ for $t_i \le x < t_{i+1}$

$S_i(x) = a_i x^3 + b_i x^2 + c_i x^3 + d_i \quad i = 0, 1, 2, \dots i-1$

Total number of unknowns $= 4n$

Requirement: S, S', S'' are all continuous

eqn		# of equation
① $S_i(t_i) = y_i$	$i = 0, 1, \dots n-1$	n
⓪ $S_i(t_{i+1}) = y_{i+1}$	$i =$ " "	n
ⓘⓘⓘ $S_i'(t_{i+1}) = S_{i+1}'(t_{i+1})$	$i = 0, 1 \dots n-2$	$n-1$
ⓘⓥ $S_i''(t_{i+1}) = S''(i+1)$	i "	$n-1$
ⓥ $\underbrace{S_0''(t_0) = 0,\ S_{n-1}''(t_n) = 0}_{\text{Natural}}$		2 (free variable)
		$4n$

polynomial of degree

$S_i = 3$

$S_i' = 2$

$S_i'' = \underline{1}$ - linear

① $S_i''(x)$ - lagrenge

② $\int S_i''(x)$

$z_0 = z_n = 0,\ z_i = S''(t_i)\quad i = 1, 2, \ldots, n-1$

z_i' are unknown

$h_i = t_{i+1} - t_i$ Lagrange form for S_i''

$$S_i''(x) = \frac{z_{i+1}}{h_i}(x - t_i) - \frac{z_i}{h_i}(x - t_{i+1})$$

$$S'(x) = \frac{z_{i+1}}{2h_i}(x - t_i)^2 - \frac{z_i}{2h_i}(x - t_{i+1})^2 + C_i - D_i$$

$$S_i(x) = \frac{z_{i+1}}{6h_i}(x - t_i)^3 - \frac{z_i}{6h_i}(x - t_{i+1})^3 +$$

$$C_i(x - t_i) - D(x - t_{i+1})$$

$$D_i = \frac{y_i}{h_i} - \frac{h_i}{6} z_i$$

$$C_i = \frac{y_{i+1}}{h_i} - \frac{h_i}{6} z_{i+1}$$

92000
8000
10000
100K

$$S'_{i-1}(t_i) = S'_i(t_i) \quad i = 1, 2, \ldots, n-1$$

$$S'_i(t_i) = -\frac{z_i}{2h_i}(-h_i)^2 + \underbrace{\frac{y_{i+1} - y_i}{h_i}}_{b_i} - \frac{z_{i+1} - z_i}{6} h_i$$

$$= -\frac{1}{6} h_i z_{i+1} - \frac{1}{3} h_i z_i + b_i$$

unknown vector

$$S'_{i-1}(t_i) = \frac{1}{6} z_{i-1} h_{i-1} + \frac{1}{3} z_i h_{i-1} + b_{i-1}$$

$$H\vec{z} = \vec{b}$$

$$H = \begin{bmatrix} 2(h_0+h_1) & h_1 & & & \\ h_1 & 2(h_1+h_2) & h_2 & & \\ & h_2 & 2(h_2+h_3) & h_3 & \\ & & \ddots & \ddots & \ddots \\ & & h_{n-3} & 2(h_{n-3}+h_{n-2}) & h_{n-2} \\ & & & h_{n-2} & 2(h_{n-2}+h_{n-1}) \end{bmatrix}$$

H = tri-diagonal,
symmetric
diagonal dominent

$2[h_{i-1} + h_i] > |h_i| + |h_{i-1}|$ which is unique sol^n

$$\vec{z} = \begin{pmatrix} z_1 \\ z_2 \\ z_3 \\ \vdots \\ z_{n-2} \\ z_{n-1} \end{pmatrix} \quad \vec{b} = \begin{pmatrix} 6(b_1 - b_0) \\ 6(b_2 - b_1) \\ 6(b_3 - b_2) \\ \vdots \\ 6(b_{n-2} - b_{n-3}) \\ 6(b_{n-1} - b_{n-2}) \end{pmatrix}$$

* Set matrix vector equation & solve for z_i

* Compute $S_i(x)$ using z_i's

Hermite Interpolation

A Hermite Spline is an interpolating piecewise cubic polynomial with a spe[…] tangent at each control point

$P(u)$ is a parametric cubic point function for curve section betw[…] control points P_k & P_{k+1}

1st point — tangent

$P(0)=P_k$ $\quad P(1)=P_{k+1}$ $\quad P'(0)=DP_k$ $\quad P'(1)=D[…]$

last point

$$P(u)=[x(u),\ y(u),\ z(u)]$$

Spline specification

$$x(u)=a_x u^3+b_x u^2+c_x u+d_x \qquad 0\le u\le […]$$

$$x(u)=\begin{bmatrix}u^3 & u^2 & u & 1\end{bmatrix}\begin{bmatrix}a_x\\ b_x\\ c_x\\ d_x\end{bmatrix}=UC$$

$3u^2\quad 2u\quad 1\quad 0$

0

1

$$\begin{bmatrix}0&0&0&1\\1&1&1&1\\0&0&1&0\\3&2&1&0\end{bmatrix}\begin{bmatrix}P_k\\ P_{k+1}\\ DP_k\\ DP_{k+1}\end{bmatrix}$$

P_k — $x\,0$

P_{k+1} — 1

Diff P_k then P

DP_{k+1} — " P_{k+1} "

Hermite matrix

$$\begin{bmatrix} P_k \\ P_{k+1} \\ DP_k \\ DP_{k+1} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix}^{-1} \begin{bmatrix} P_k \\ P_{k+1} \\ DP_k \\ DP_{k+1} \end{bmatrix}$$

M_H

$$= \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} P_k \\ P_{k+1} \\ DP_k \\ DP_{k+1} \end{bmatrix}$$

4×4 4×1

4×1

$$= \begin{bmatrix} 2P_k - 2P_{k+1} + DP_k + DP_{k+1} & -3P_k + 3P_{k+1} - 2DP_k - DP_{k+1} & + DP_k & + P_k \end{bmatrix}_{4\times1}$$

$$P(u) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \cancel{(2P_k - 2P)} . M_H \begin{bmatrix} P_k \\ P_{k+1} \\ DP_k \\ DP_{k+1} \end{bmatrix}$$

$4\times1\times4$

1×4

$$\cancel{= \begin{bmatrix} u^3 \\ u^2 \\ u \\ 1 \end{bmatrix} \begin{bmatrix} 2P_k - 2P_{k+1} + DP_k + DP_{k+1} & -3P_k + 3P_{k+1} - 2DP_k - DP_{k+1} & DP_k & P_k \end{bmatrix}}$$

1×4

4×4

1×4

4×1 4×1

1×4

$$= \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix}_{1\times4} \begin{bmatrix} 2P_k - 2P_{k+1} + DP_k + DP_{k+1} \\ -3P_k + 3P_{k+1} - 2DP_k - DP_{k+1} \\ DP_k \\ P_k \end{bmatrix}_{4\times1}$$

$$= 2P_k u^3 - 2P_{k+1} u^3 + Dp_k u^3 + Dp_{k+1} u^3 - 3P_k u^2 + 3P_{k+1} u^2$$
$$- 2Dp_k u^2 - Dp_{k+1} u^2 + Dp_k u + P_k$$

$$= P_k (2u^3 - 3u^2 + 1) + P_{k+1} (-2u^3 + 3u^2) + DP_k (u^3 - 2u^2 + u)$$
$$DP_{k+1} (u^3 - u^2) -$$

$$= P_k H_0(u) + P_{k+1} H_1(u) + Dp_k H_2(u) + Dp_{k+1} H_3(u)$$

The polynomial $H_k(u)$ for $k = 0,1,2,3$, are refered to as blending function.

Blending function

→ Curves, interpolation, apporimation, blending function

Curve:- ① used to create high resolution graphics
② Specified by parametric equation
③ Easily scaled & without losing smoothnes
④ Double back (go in & out) & cross (zigzag)

Implicit

Not directly expressed

$-f(x,y) = 0$

Test if point is on curve

Explicit

Directly expressed

$y = f(x)$

Cannot represent vertical line

Single Valued

Parametric

Curves having parametric form

$P(t) = f(t), g(t)$
2D

$P(t) = x(t), y(t)$

$[a, b]$ - interval

$[0, 1]$

App that can

Approximation → Complex curve

$x = f_x(u)$ Interpolation : Construct new data points
$y = f_y(u)$ within range ~~of~~ only
$z = f_y(u)$ Blending function

$(x_1, y_1, z_1) \ldots (x_n, y_n, z_n)$

$$F_x(u) = \sum_{i=1}^{n} x_i \underset{\uparrow}{B_i}(u)$$

$B_i(u)$ ← strenght to pull that curve in the given direction

$$F_y(u) = \sum_{i=1}^{n} y_i B_i(u)$$

$$f_z(u) = \sum_{i=1}^{n} z_i B_i(u).$$

$u = (-1)$ $B_1(u) = 1$. and 0 for $u = 0, 1, 2, \ldots, n-2$

$u = 0$ $B_2(u) = 1$ and 0 for $u = -1, 1, \ldots n-2$

$\vdots$

$u = n-2$ $B_n(u) = 1$ and 0 for $u = -1, 0, \ldots, n-1$

$B_1(u) = 1,\ u = -1\ ,\ 0$ for $u = 0, 1, 2; \ldots n-2$

$$B_1(u) = \frac{u(u-1)(u-2) \ldots [u(n-2)]}{(-1)(-2) \ldots [1-n]}$$

$$B_i(u) = \frac{(u+1)\,u\,(u-1) \ldots [u-(i-3)][u-(i-1)] \ldots [u-(n-2)]}{(i-1)(i-2)(i-3) \ldots (1)(-1)(i-n)}$$

Lagrange, $$y(x) = \frac{y_1(x-x_2)(x-x_3)(x-x_4)}{(x_1-x_2)(x_1-x_3)(x_1-x_4)} + \frac{y_2(x-x_1)(x-x_3)(x-x_4)}{(x_2-x_1)(x_2-x_3)(x_2-x_4)} + \frac{y_3(x-x_1)(x-x_2)(x-x_4)}{(x_3-x_1)(x_3-x_2)(x_3-x_4)} + \frac{y_4(x-x_1)(x-x_2)(x-x_3)}{(x_4-x_1)(x_4-x_2)(x_4-x_3)}.$$

$x_4 = 4$ | $x = u$: -1 0 1 2

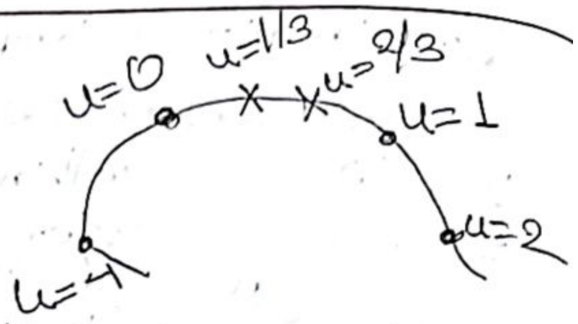
1 2 3 4

1st -1 $B_1(u) = \dfrac{u(u-1)(u-2)}{(-1)(-2)(-3)}$

2nd 0 $B_2(u) = \dfrac{(u+1)(u-1)(u-2)}{1(-1)(-2)}$

3rd 1 $B_3(u) = \dfrac{(u+1)(u)(u-2)}{2 \cdot 1 \cdot (-1)}$

4th 2 $B_3(u) = \dfrac{(u+1)(u)(u-1)}{3 \cdot 2 \cdot 1}$



$x = x_1 B_1(u) + x_2 B_2(u) + x_3 B_3(u) + x_4 B_4(u)$

y y_1 y_2 y_3 y_4

z z_1 z_2 z_3 z_4

So, we are calculating for each 4 points.

$(1, 2, 3, 4)$ after calculating for 1 to 2 discard 1 & put another no. at end —

$(2, 3, 4, 5)$

1 2 1 2

-1 0

Linear Spline: Only C^0 continuous

$P(t) = (1-t)P_0 + tP_1$

Cardinal Spline : use neighbouring point

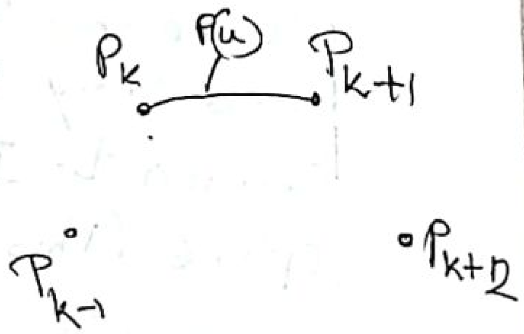
Scale ↑ (1) bumpy   Catmull-Rom spline.

Scale ↓ (0) straight line .

Cardinal spline are interpolating piecewise cubics with specified endpoint tangents at the boundary of each curve section.

We do not need to give values for endpoint tangent.

ok 34 → The value of the slope at a control point is calculated from the co-ordinates of the two adjacent control points.



4 consecutive control points are used to ~~represent~~ specify cardinal spline section completely.

The middle 2 points are the section endpoints and the other two points are used in the calculation of the endpoint slopes.

If we take $P(u)$ as the representation for the parametric cubic point function for the curve section between control points P_k and P_{k+1}. Then the four control points from P_{k-1} to P_{k+1} are used to set the boundary condition for the cardinal spline section as —

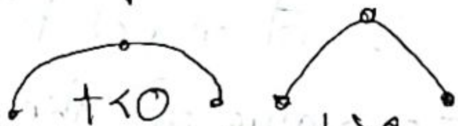
$P(0) = P_k$ $P'(0) = \frac{1}{2}(1-t)(P_{k+1} - P_{k-1})$

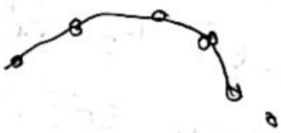
$P(1) = P_{k+1}$ $P'(1) = \frac{1}{2}(1-t)(P_{k+2} - P_k)$

Cardinal - No Interpolating but C^2 Continuous.

Thus the slopes at control points P_k and P_{k+1} are taken to be proportional, respectively to the chord $P_{k-1}P_{k+1}$ and P_kP_{k+2}. the parameter t is called the tension parameter since it controls how loosely or tightly the cardinal spline fits the input control points.

The shape of a cardinal curve for very small and very large value of tension t. If $t=0$ then its Catmull-Rom spline. (this curves are called)





4x4 16 eqn's

$$P(u) = \underbrace{[u^3 \; u^2 \; u \; 1]}\, M_c \begin{bmatrix} P_{k-1} \\ P_k \\ P_{k+1} \\ P_{k+2} \end{bmatrix}_{4\times 1} \qquad s = \frac{1-t}{2}$$

$$M_c = \begin{bmatrix} -s & 2-s & s-2 & s \\ 2s & s-3 & 3-2s & -s \\ -s & 0 & s & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \qquad s = \frac{1-t}{2}$$

Expanding the matrix equation $P(u)$ to Polynomial form, we have -

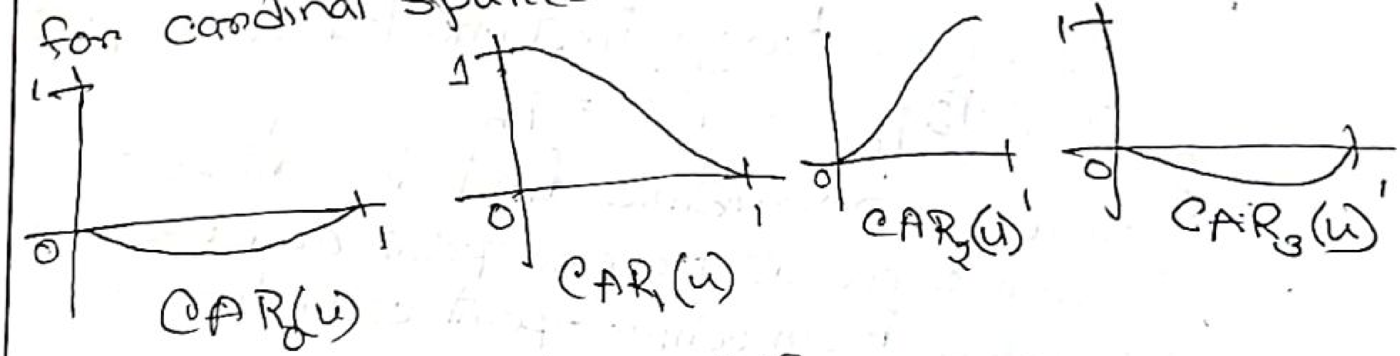
$$\cancel{P(u) = [-su^3 + u^2(2-s) + u(s-2) + s}$$

$$P(u) = \begin{bmatrix} -su^3 + 2su^2 - us & (2-s)u^3 + (s-3)u^2 + 1 & (s-2)u^3 + (3-2s)u^2 + us & u^3 s - u^2 s \end{bmatrix} \begin{bmatrix} P_{k-1} \\ P_k \\ P_{k+1} \\ P_{k+2} \end{bmatrix}$$

$$= P_{k-1}(-su^3 - 2su^2 - su) + P_k[(2-s)u^3 + (s-3)u^2 + 1] + P_{k+1}[(s-2)u^3 + (3-2s)u^2 + su] + P_{k+2}(u^3 s - s u^2)$$

$$= P_{k-1} CAR_0(u) + P_k CAR_1(u) + P_{k+1} CAR_2(u) + P_{k+2} CAR_3(u)$$

where the polynomials $CAR_k(u)$ for $k = 0, 1, 2, 3$ are the cardinal blending func. The basis functions for cardinal splines with $t = 0$



For $t = 0$ and $s = 0.5$

Bazier Curve:- Used to various CAD system, in general graphics packages (such as GL on Silicon Graphics system).

Bazier curve section can be fitted into any number of control points. For general Bazier curves, blending functions specification is most convenient.

n+1 control points $P_k = (x_k, y_k, z_k)$ with k varying from 0 to n. These coordinate points can

$$P(u) = \sum_{k=0}^{n} P_k \, BEZ_{k,n}(u) \cdot \quad 0 \leq u \leq 1$$

Bezier blending function:

$$BEZ_{k,u}(u) = C(n,k) u^k (1-u)^{n-k}$$

$C(n,k)$ are binomial coefficient:

$$C(n,k) = \frac{n!}{k!(n-k)!}$$

book-332

→ Properties of Bezier Curves –

(1) It passes through First and last control point

$P(0) = P_0$ & $P(1) = P_n$

(2) Values of parametric first derivatives of Bezier curve at the end point can be calculated from control-point co-ordinates as

$$P'(0) = -nP_0 + nP_1$$

$$P'(1) = -nP_{n-1} + nP_n$$

(3) the slope at the beginning of the curve is along the line joining the first two control points.

The slope at the end of the curve is along the line joining the last two end points

. \

③ The Parametric second derivatives of a bezier curve at the endpoints are calculated as -

$$P''(0) = n(n-1)\left[(P_2 - P_1) - (P_1 - P_0)\right]$$

$$P''(1) = n(n-1)\left[(P_{n-2} - P_{n-1}) - (P_{n-1} - P_n)\right]$$

④ The bezier curve lies within the convex hull of the control points. (convex polygon boundary)

⑤ The Bezier blending functions are all positive and sum always 1. $\sum_{k=0}^{n} BEZ_{k,n}(u) = 1$

⑥ The convex hull property for a Bezier curve ensures that the polynomial smoothly follows control points without erratic oscillation.

⑦ Bazier curves are Approximation Spline curve

⑧ " " is Parametric Curve.

⑨ Cut n degree Polynomial with n+1 control point

$\cancel{P(x) = a}$

Fractal Geometry :-

(So previous ones had equations and were Euclidean -geometry). But natural objects- mountain clouds has irregular or fragmented features.

Characteristics :- ① Infinite detail at every point
② A certain self-similarity betn the object parts and overall features of object

→ Describe Fractal object with a procedures that specifies a repeated operation for producing the detail in the object subparts.

→ zoom in euclidean shape → smooth out.
→ zoom in fractal object → see more detail in magnification as in original view.

→ Self similarity property of a object can take different forms, depending on the choise of fractal representation.

⇒ The amount of variation in the object detail with a number called Fractal dimension aka fractional dimension, which is basis for name "fractal"

Types of Fractal -

Self-similar:- fractals have parts as scaled down version of entire object

Self-Affine:- have parts formed with different scaling parameters in different co-ordinate directions

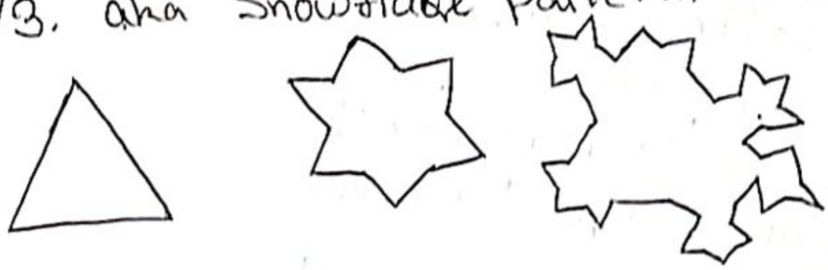
Invarient:- formed with non-linear transformation
Self squaring fractals, [self-inversing fractal - formed with inversion procedure.

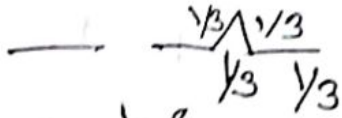
Fractal Dimension:- Measure of roughness or fragmentation of object

⇒ More fractal dimension = more jagged looking object

⇒ Using some iterative procedure we calculate fractal dimension D.

Koch Curve:- Each straight line is replace with four equal sized lines of scaling factor 1/3. aka Snowflake pattern. Fractal dimension = 1.2619





$$D = \frac{\ln 4}{\ln 3} = 1.261$$

Self Similar - tree, shrubs, plants.

Self affine - terrain, water, clouds.

Invarient - Mandelbrot &

Applications:- ① Modeling natural structures - geographic terrain, mountain, plant structure, clouds, vegetables.
② Study chaotic phenomenon
③ Fractal art
④ Space research
⑤ Medical Science
⑥ Fluid mechanics
⑦ Image Compression
⑧ telecommunication

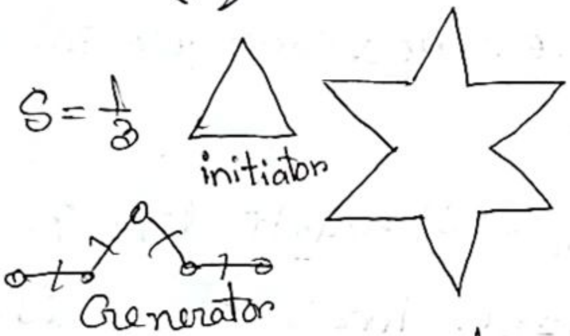
Fractal object is generated by repeatedly applying a specified transformation function to points within a region of space. If $P_0=(x_0,y_0,z_0)$ is a selected initial point, each iteration of a transformation fun^n -

$P_1=F(P_0)$, $P_2=F(P_1)$, $P_3=F(P_2)$

$$D=\frac{\ln(n)}{\ln(1/s)} \quad n\cdot s^{D}=1 \qquad \sum_{k=1}^{n} s_k^{D}=1$$

↓ scaling factor

$S=\frac{1}{3}$



deterministic

$D>1$: fractal curve lies completely within 2D plane

$D\approx1$: smoother fractal curve

$D=2$: Peano curve

$2<D<3$: curve self-intersects.

$D>1$: Spatial Fractal curve

$D>2$: without self-intersecting.

$D=3$: Surface fill volume of space

$D>3$: Overlapping coverage of volume

$3<D\leq4$: Fractal Solid

$D>D$: Self-overlapping object [water vapor density, temp+ within region of space

Invarient ~~Factorial~~ fractals

Characteristics	Self-Squaring	Self-Inverse
Definition	Can be decomposed into smaller copies of themselves.	Fractals that are their own inverse under a certain transformation.
Iterated Func System	Required.	Not Required
Construction	Based on IFS and the concept of iterated transformation	Based on a specific mathematical property of self-inverse transforma-tion.
Shape	Exhibit self-similar pattern at different ~~sc~~ scales, each ~~at~~ iteration squaring no. of copies	Doesnot exhibits self-similar pattern, instead are their own inverse under particular transformation
	Menger Sponge Glynn-Julia Julia set	Phoenix fractal

Convex hull:- The convex polygon boundary that encloses a set of control points is called convex hull. ~~The~~ It provides a measure for the derivation of a curve or surface from the region bounding the control points. Some splines are bounded by the convex hull, thus ensuring that the polynomial smoothly follow the control points without erratic oscillation.

created by specifying the endpoints.

a continuous line that is composed of one or more connected straight line segment.

Control graph :- A polyline connecting the sequence of control points for an approximation spline is usually displayed to remind a designer of the control-point ordering. This set of connected line segments is often referred to as the control graph of the curve.

Other name of the series of straight-line sections connecting the control points is the order specified are control polygon and characteristics polygon

① Convex hull shapes (the dashed lines) for

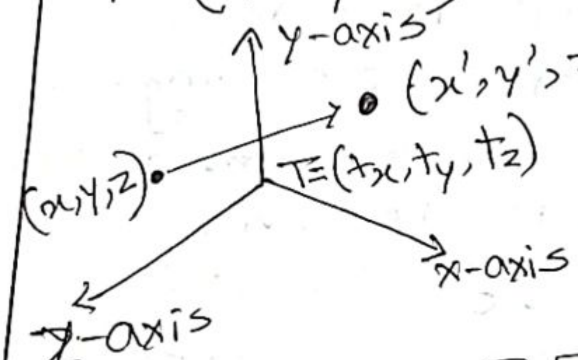
② Control graph

For 2 different set of points.

CH-11: 3D Geometric & Modeling transformation

3D Translation :-

$T = (t_x, t_y, t_z)$



$x' = x + t_x$

$y' = y + t_y$

$z' = z + t_z$

3D point

$\begin{bmatrix} x \\ y \\ z \end{bmatrix}$

in column

$P = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$

homogenious 4D

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$P' = T \cdot P$

When we are transforming we need to represent in 3D homogeneous co-ordinate representation.

~~3D Scaling~~ 3D Rotation:-

Must designate an axis of rotation. And the angle of Rotation.

Positive rotation angle → Rotation counterclockwise

3D Rotation about x-axis, the x co-ordinate of

$x' = x\cos\theta - y\sin\theta$

$y' = x\sin\theta + y\cos\theta$

$z' = z.$

3D rotation about x-axis, X-coordinate of the position vector doesn't change, the transformation with angle θ:

$$[T] = \begin{array}{c|cccc} & x & y & z & \\ x & 1 & 0 & 0 & 0 \\ y & 0 & \cos\theta & -\sin\theta & 0 \\ z & 0 & \sin\theta & \cos\theta & 0 \\ & 0 & 0 & 0 & 1 \end{array}$$

x এর নিচে

diagonal = 1

$$\begin{array}{c|cccc} & x & y & z & \\ x & 1 & 0 & 0 & 0 \\ y & 0 & & & \\ z & 0 & & & \\ & 0 & & & 1 \end{array}$$

Rotation is Right hand sense, that is clockwise as one looks outward from ORIGN in the positive direction along the rotation axis. Right hand thumb rule

Transformation Matrix for rotation by angle ϕ about y-axis.

$$[T] = \begin{array}{c|cccc} x & \cos\phi & 0 & \sin\phi & 0 \\ y & 0 & 1 & 0 & 0 \\ z & -\sin\phi & 0 & \cos\phi & 0 \\ & 0 & 0 & 0 & 1 \\ \hline & x & y & z & \end{array}$$

Sine term reversed sign to maintain R.H.T

Rotate φ about z-axis

$$[T] = \begin{array}{cccc} x & y & z & \\ \cos\varphi & -\sin\varphi & 0 & 0 \\ \sin\varphi & \cos\varphi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}$$

$x\ y\ z \quad -90°$

$$A = (3,2,1) \xrightarrow{x\text{-axis}} A^* = (3,1,-2)$$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 3 \\ 2 \\ 1 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 3 \\ 1 \\ -2 \\ 1 \end{bmatrix}$$

Rotation axis -2 co-ordinate

Rotation in 5 steps –

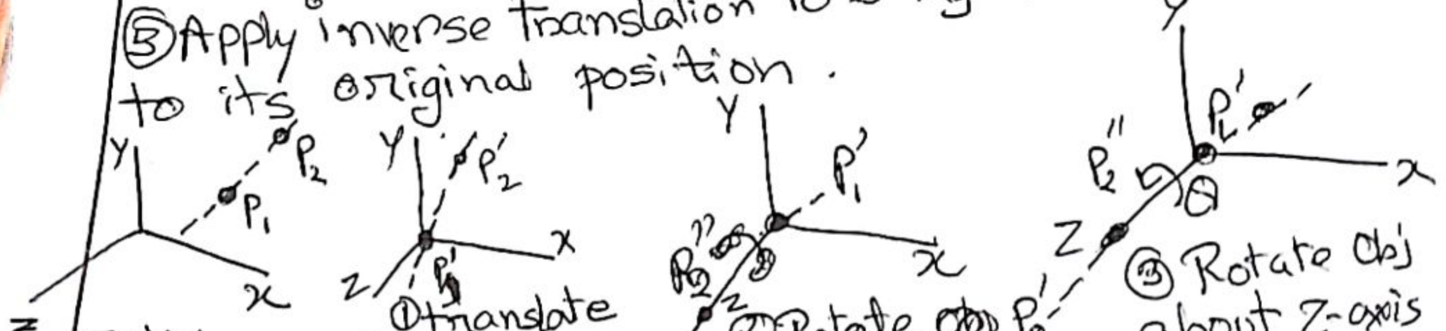
① translate object so that the rotation axis passes through the co-ordinate origin.

② Rotate the object so that the axis of rotation coincides with one of the co-ordinate axes.

③ Perform specified rotation about that co-ordinate axis.

④ Apply inverse rotation to bring rotation axis back to its original orientation

⑤ Apply inverse translation to bring rotation axis back to its original position.

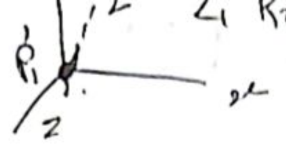


Initial

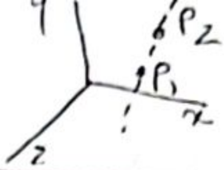
① translate P_1 to Origin

② Rotate obj P_2 ~~around axis~~ onto Z-axis

③ Rotate obj about Z-axis



4. Rotate Axis to Original Orientation



5. Translate Rotation axis to original pos.

Scaling

Change size of object: Scaling transformation relocates a point with relation to the origin.

normal –

$$P' = \begin{vmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & S_z \end{vmatrix}$$

homogeneous co-ordinate –

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$P' = S \cdot P$

$x' = S_x \cdot x$

$y' = S_y \cdot y$

$z' = z \cdot S_z$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} =$$

Rotation

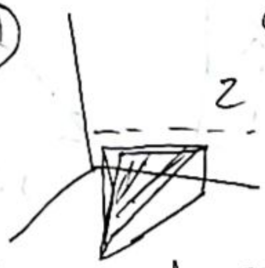
$V = P_2 - P_1 = (x_2 - x_1, y_2 - y_1, z_2 - z_1)$ $\quad u = \frac{V}{|V|} = (a, b, c)$

$a = \frac{x_2 - x_1}{|V|}, \quad b = \frac{y_2 - y_1}{|V|}, \quad c = \frac{z_2 - z_1}{|V|}$

(a) Original position of object

(c) Rotate object through angle θ

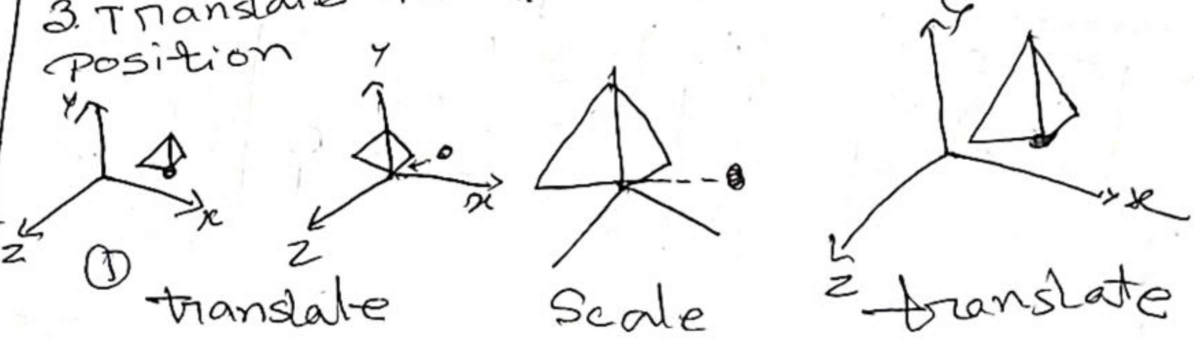
(b) translate Rotation axis onto x-axis.



(d) translate rotation axis to original position.

Scaling representation

1. Translate fixed point to origin
2. Scale object relative to the origin using formula
3. Translate the fixed point back to original position



$$T(x_f, y_f, z_f) \cdot S(s_x, s_y, s_z) \cdot T(-x_f, -y_f, -z_f) =$$

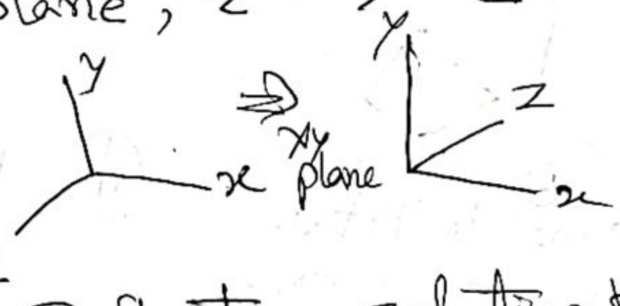
$$\begin{bmatrix} s_x & 0 & 0 & (1-s_x)x_f \\ 0 & s_y & 0 & (1-s_y)y_f \\ 0 & 0 & s_z & (1-s_z)z_f \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Reflection:- Occurs through a plane.

For pure reflection determinant of the reflection matrix is identically -1

Reflect through xy plane, $z \rightarrow -z$

$$RF_z = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



Reflection relative to xy plane

plane
yz (−1)
xz (−1)
xy (−1)

Shears- to modify object shapes.

shear relative to z-axis.

$$SH_z = \begin{bmatrix} 1 & 0 & a & 0 \\ 0 & 1 & b & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

parameters a, b assign any value

z-axis shearing

$z' = z$

$x' = x + z\,Sh_x$

$y' = y + z\,Sh_y$.

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} 1 & 0 & Sh_x \\ 0 & 1 & Sh_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

x-axis

$x' = x$

$y = y + x\,Sh_y$

$z = z + x\,Sh_z$.

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ Sh_y & 1 & 0 \\ Sh_z & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

y-axis

$y' = y$

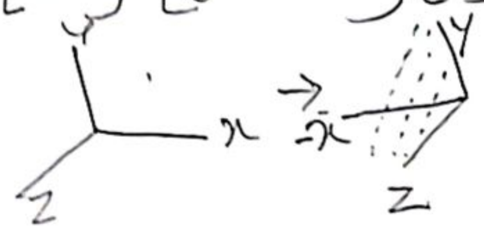
$x' = x + y\,Sh_x$

$z' = z + y\,Sh_y$

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} 1 & Sh_x & 0 \\ 0 & 1 & 0 \\ 0 & Sh_z & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

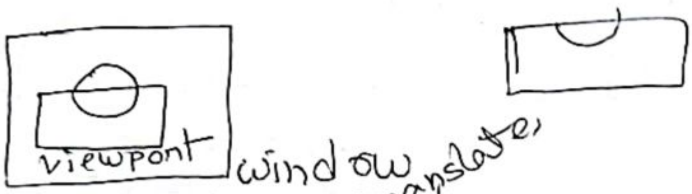
Reflection through yz plane

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$



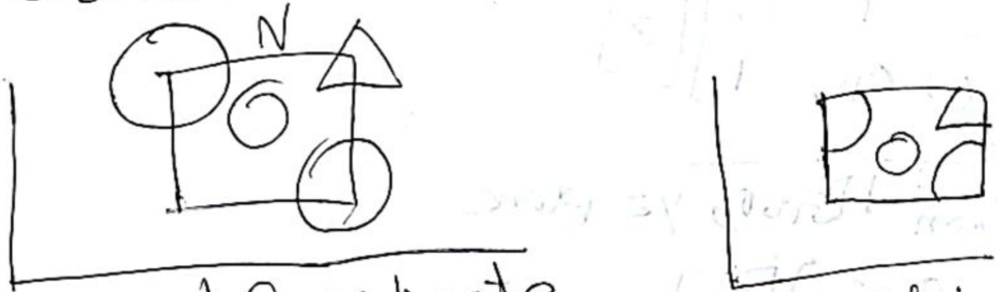
CH-10

Viewing Pipeline



Modeling Co-ordinate → Modeling transformation → World Co-ordinate → Viewing transformation → Viewing Co-ordinate → Projection transformation → Projection Co-ordinate → Work station transformation → Device Co-ordinate.

General 3D transformation pipeline, from modeling coordinate to final device coordinates.



World Coordinate

View.

view port
view up vector

Line of sight:- Refers to the visibility of an object or a point from a certain viewpoint and is used to determine which part of a scene should be rendered or displayed on the screen. [Casting algorithm], [Binary space partitioning tree]

Modeling Co-ordinate - Analogous to taking photograph
↳ objects.

Modeling transform :- Change the shape of models to ~~being~~ ~~......~~ transform to world co-ordinate.

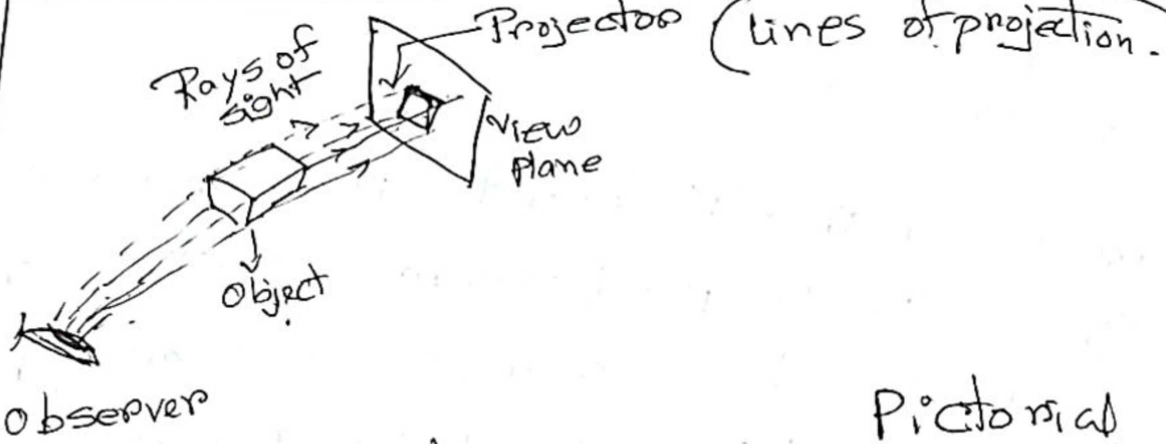
V.T → set position within 2D and also to do orientation (set camera)

Project. T → Set picture - [Focus

Viewport transformation → [photo taken]
↓
device

3D viewing Parameters -

Projection:-

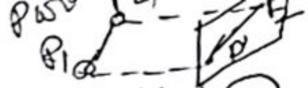


Projection types

- Parallel Projection
  - Oblique
    - Cavilier
    - Cabinet
  - Orthographic ⊥
    - Multiview
      - 1st Angle Proj
      - 2nd
      - 3d
      - 4th
    - Axonometric
      - Isometric
      - Dimetric
      - Trimetric
- Perspective Pro. — Pictorial (Realistic)
  - 1 Point
  - 2P
  - 3P

Parallel Proj:- Co-ordinate position are transformed to the view plane along parallel lines

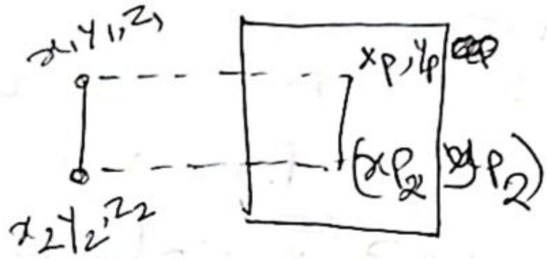
Perspective Proj:- Obj position are transformed to the view plane along lines that converge to a point called projection reference point or center projection. The projected view of an object is determined by calculating intersection of projection lines with view planes.



Parallel Projection.

Projection Reference point

~~Orthogonal~~

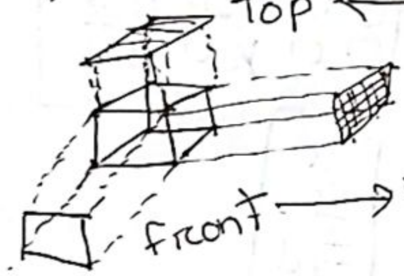


3D → 2D

Parallel projection line → Projection vector

Projection vector parpendicular to viewplane called <u>Orthographic</u> $x_p = x$, $y_p = y$

No Parpendicular → <u>Oblique</u>.



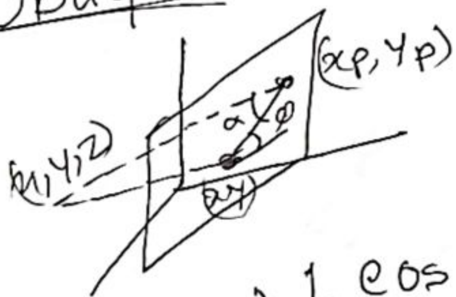
types of Orthographic

Orthograph.

$x_p = x$

$y_p = y$

Oblique



$$x_p = x + L\cos\varphi$$

$$y_p = y + L\sin\varphi$$

$$\begin{bmatrix} x_p \\ y_p \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & L_1\cos\varphi & 0 \\ 0 & 1 & L_1\sin\phi & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

~~Display~~.

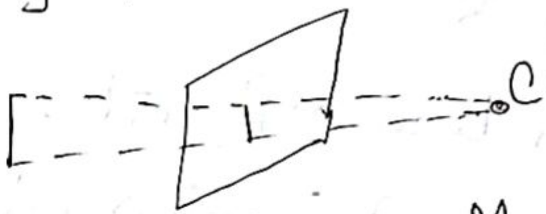
Display more than one face of an object - axonometric

isometric -

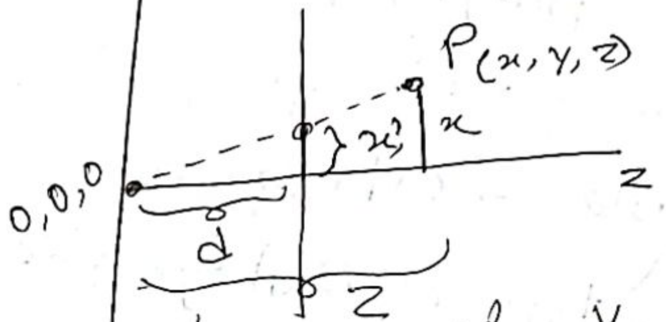
Perspective Projection :-

the lines converge to one point → Cent
of projective

far - small
near - larg



$$M_{per} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/d & 0 \end{bmatrix}$$



$$\begin{bmatrix} x' \\ y' \\ z' \\ z/d \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/d & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} x \\ y \\ z \\ z/d \end{bmatrix}$$

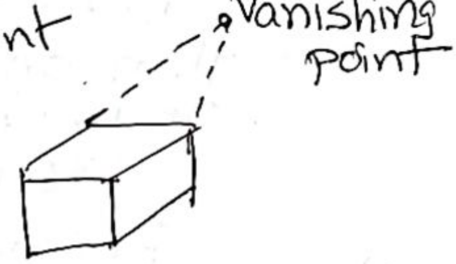
$$\frac{x'}{d} = \frac{x}{z} \qquad \frac{y'}{d} = \frac{y}{z}$$

$$\therefore x' = \frac{xd}{z}$$

$$\therefore y' = \frac{y}{z/d}$$

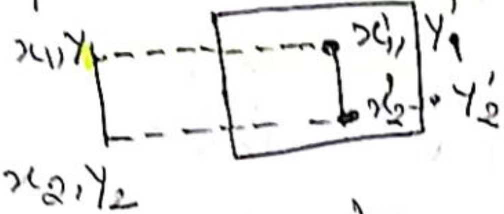
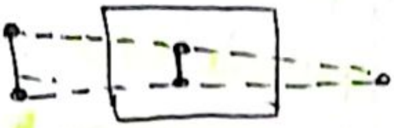
$$z' = z$$

1 point



2point

3point

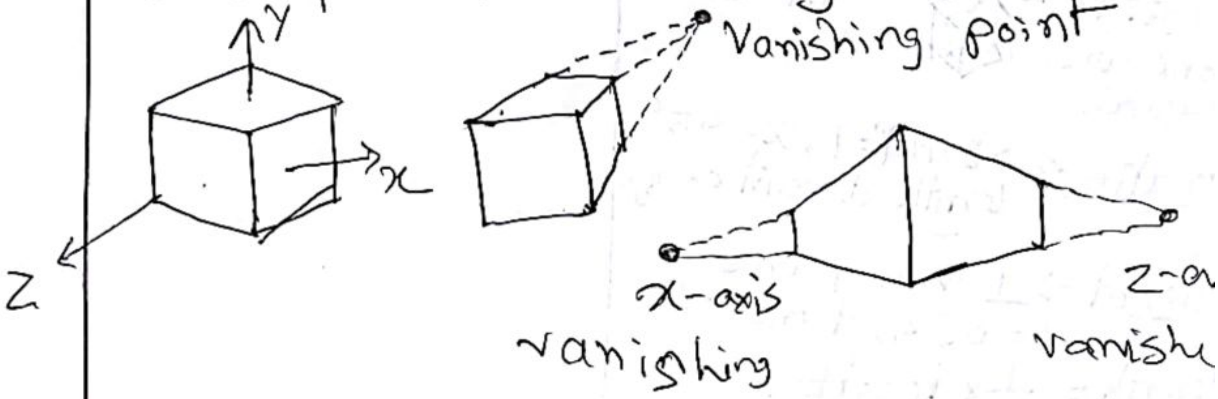
	Parallel	Perspective
1.	Co-ordinate positions of the object are transformed into the view plane along the parallel line  	Co-ordinate positions of the object are transformed into the view plane along with the point called center of projection 
2.	Relative properties of object are maintained.	It does not maintain R.P
3.	Accurate view of object	4. No accurate view of object. Far objects small, near objects large
4	Doesnot give the realistic view	Give realistic view
5.	types এর photo আগের page এ আছে। এখানে just define থিওরি - Isometric -  $A = B = C$ angles Dimetric -  any 2 angles equal $(A = B) \neq C$ $A \neq (B = C)$ $(A = C) \neq B$ Trimetric -  none are equal Cavalier $\rightarrow \tan\alpha = 1, \alpha = 45°$ length doesn't change Cabinet $\rightarrow \perp$ view plane $\alpha = 63.4, \tan\alpha = 2$ length $= \frac{1}{2} \times$ length Engineering drawing, Architecture drawing	1. point Projection 2 " " 3 " " Building design, rail track.

Vanishing Point :-

When a three dimentional object is projected onto a view plane using perspective transformation equations, any set of parallel lines in the object that are not parallel to the plane are projected into converging lines. Parallel lines that are parallel to the view plane will be projected as parallel lines. The point at which a set of Projected parallel lines appears to converge is called a vanishing point

age-456

Vanishing point for any set of lines that are parallel to one of the principal axes of an object is reffered to as a principal vanishing point.



Frustom - portion of a solid that lies btn 2 parallel planes cutting the solid (cone, pyramid) (piece cut off)

Orthographic Parallel Projection :- Parallel projection with projection vector that defines the direction for the projection line. When the projection is perpendicular to the view plane we call Oro' Pall PrD.

Axonometric Ortho P. :- The form of orthographic projection that display more than one face of an object.

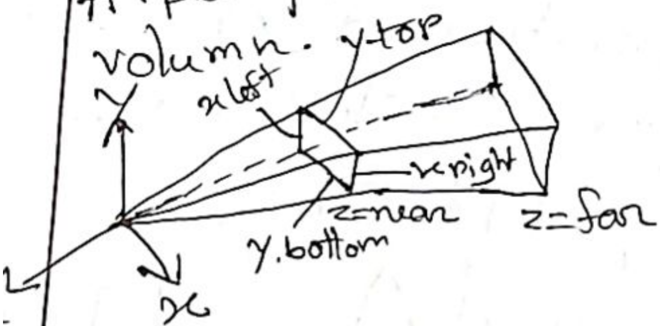
View Window OR Projection window :- Also known as viewport. Is a rectangular area on a computer screen ~~that~~ displays a portion of a larger image or ~~scea~~ scene. It is used to show a specific part of a 2D/3D scene ~~that~~ has been created or imported into a graphics application.

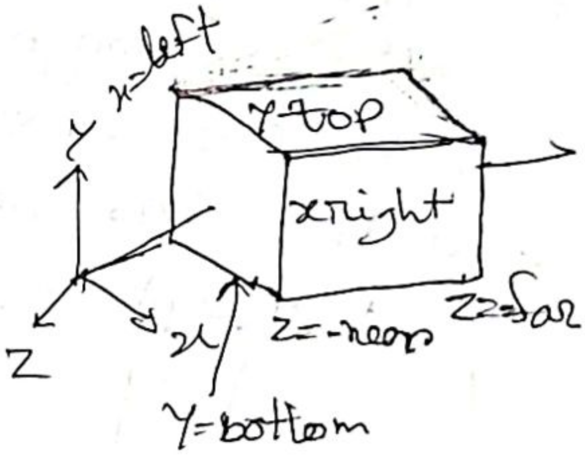
View plane: It is an area ~~of~~ where world co-ordinate system which is projected into viewing plane.

Projector / Projection vector: Rays that start from the object scene and used to create an image of the object on viewing or view plane

View Volumn: The part of the part world that is visible in the image. It is used for → ① Clipping objects which will not project onto the image
② Restricting the domain of z for visibility calculations.

A perspective view volumn.





CH-13 : Visible -Surface Detection Method.

In Realistic graphics displays, it is important to identify those parts of a scene that are visible ~~for~~ from a chosen viewing position

Classification of V.S.D.M. -

deal with Object definition directly ⇒ Object-space method

or with the projected image of object ⇒ Image-space method

O.S.M:- Compares objects and parts of objects to each other within the scene definition to determine which surface as a whole, we should ~~tabl~~ label as visible.

I.S.M:- Visibility is decided Point by point at each Pixel position on the projection plane.

⇒ Most visible-surface algorithm uses $\boxed{\text{ISM}}$.
but
OSM. method can be used effectively to locate visible surfaces in some cases.

Line-display Algo:- use O.S.M. to identify visible lines in wireframe displays,

Image-space visible-surface algorithm can be adapted easily to visible-line detection.

hidden – Surface Elemination

The visible-surface detection algorithms most use sorting and coherence method to improve performance.

Sorting is used to facilitate depth comparison by ordering the individual surfaces in a scene according to their distance from the view plane

Coherence method are used to take advantage of regularities in a scene.

Algorithm Selection criteria –

① Particular Application
② Complexity of scene involved
③ Type of object to be displayed
④ Available equipment
⑤ Static or animated display are to be generated

Back face detection :- (Object space method) uses

OSM :-

M (x, y, z) viewing direction V

make 90° angle on surface

Normal vector N(A, B, C)

$Ax + By + Cz + D < 0$ ← point inside the polygon

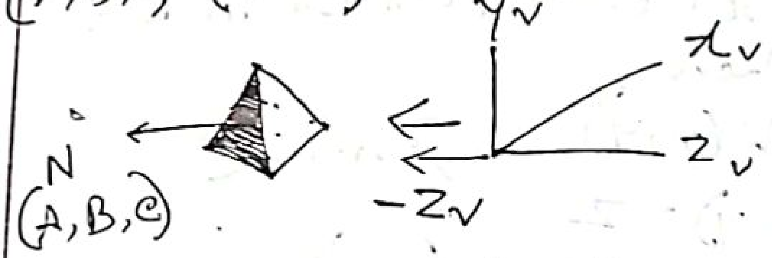
$V \cdot N > 0$ ← that particular Surface will be backface

$V \cdot M \leq 0$ ← Front face

$V = (0, 0, V_z)$ [parallel to z]

$N \cdot V = V_z \cdot C$ if $C > 0$ / $C < 0$

$(A, B, C) \cdot (0, 0, V_z)$ (Right handed system



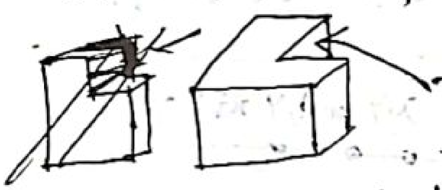
$N(0, 0, V_z)$

If $V \cdot N > 0$ backface if $C = +ve$ $C \times (-z_v) \le 0$ Front face

if $C = -ve$ $(-C) \times (-z_v) > 0$ Backface

Drawback

Partial hidden surface problem



this area is not hidden completely. So, there may rise issues.

concave polyhedron

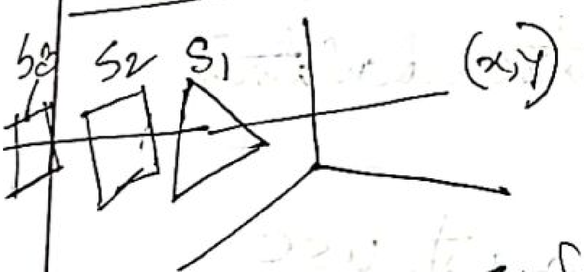
Depth Buffer method: (uses image space method

Calculate intensity

Algorithm

→ buffer (x, y) → z value

→ refresh buffer (x, y) → Intensity of (x, y).



→ z-co-ordinate will be normalized

$0 < z < 1$

Back clipping plane

Front clipping plane

S_1 will be visible surface.

will use 2 buffer

Good for opaque object Intensity

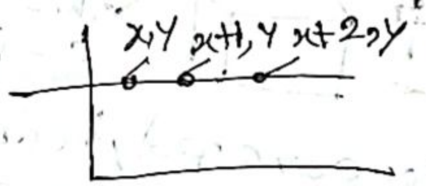
(I) Initialize both the buffer i.e.
buffer $(x,y) = 0$, refresh $(x,y) = I_{background}$

(II) Calculate Z-value for each position in the surface and then
if $z > depth(x,y)$ surface
$depth(x,y) = z$
$refresh(x,y) = I_{sur}(x,y)$

(III) After processing all the surface we will get visible surface in depth (x,y) and intensity value in surface (x,y).

buffer Algorithm

$$Ax + By + Cz + D = 0$$



$$z = \frac{-Ax - By - D}{C}$$

$$z' = \frac{-A(x+1) - By - D}{C} = \frac{-Ax - By - D - A}{C}$$

$$= z - A/C$$

$$\therefore z_{i+1} = z_i - \frac{A}{C}$$ [iterative calculation]

Disadvantage :-

(i) In transparent we can not use z buffer algo.

~~Painter's Algorithm (Depth Sorting)~~.

Advan

1) Easy implementation
2) No sorting of the surface in a scene.
3) For opaque.

A-Buffer Method: Antialised, area average, accumulation buffer.

2 fields :- ① depth field, ② Intensity field

depth field: real no. +ve, -ve

$d < 0 \rightarrow$ Multiple surface

$d > 0 \rightarrow$ Only effect of intensity of a single surface

Intensity field: (x, y), Linked List

pointers storing how many surface intensity are on that point

d>0	I

Pixel dont get overlaped only one Surface

d<0	•	→	sur 1	→	sur 2

Pixel gets overlaped by multiple surface.

opaque

transparent

Surface data:

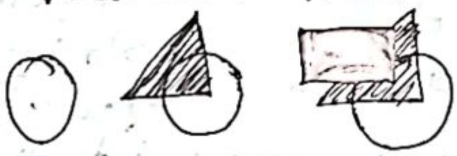
1) Surface identifier
2) transparency percentage
3) Opacity
4) RGB color
5) Other randering Parameter
6) Percent of area coverage
7) pointer to next surface.

(depth Sorting algorithm) compare object — take project of obj.

Painter's Algorithm:- Both O.S.M & I.S.M

Step ① Sorting the surfaces according to the z-value (decreasing order) for both Object-space & image space.

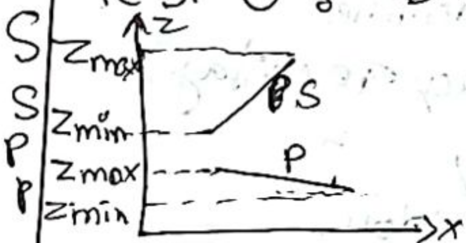
② Scan conversion of sorted surface
↳ Image space - We will store point by point the pixel intensity in the frame buffer.



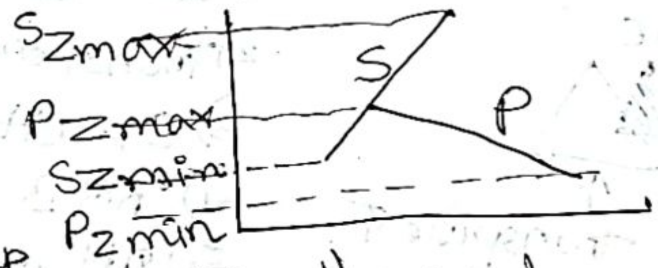
also called Depth Priority / Priority algorithm

We need to do some tests to find if overlap occurs-

Test 0: Depth Overlap.



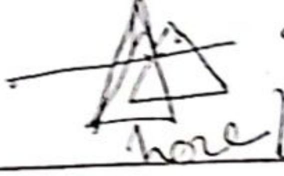
No depth overlap



Depth overlap

If Depth overlap do test 1:-

Continue ⊛



Depth Compare → take nearest one

$\boxed{\text{Searched \& got a bit diffent test case}}$

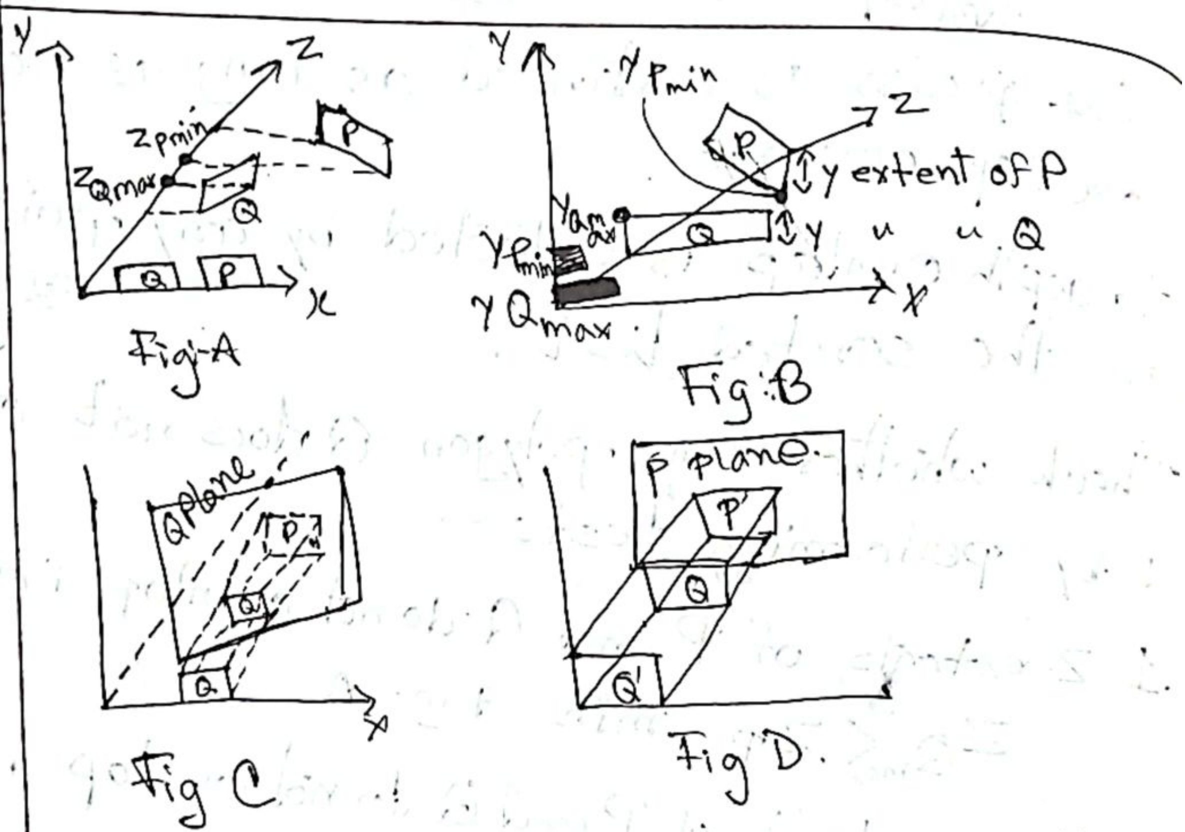
The process is continued as long as no overlap occurs.

If depth overlap is detected by any point in the sorted list.

আড়াল করে

Check whether any polygon Q does not obsure P by performing tests –

1. Z-extents of P and Q do not overlap i.e.
   $Z_{Q_{max}} < Z_{p\,min}$ Fig- A
2. The y-extent of P and Q do not overlap. Fig B.
3. The x-extents of P and Q do not overlap
4. Polygon P lying entirely on the opposite side of Q's plane from the view port (Fig : C)
5. Polygon Q lying entirely on the same size of P's plane as the viewpont
6. The projection of polygon P and Q onto xy screen dont overlap.



If all the 5 test fail, P actually obsure Q. and test whether Q can be scan-converted before P. So repeat 4 & 5 for Q.

If tests also fail-
No order in which P and Q can be scan converted correctly and we have to split ~~P~~ either P or Q into two polygon.

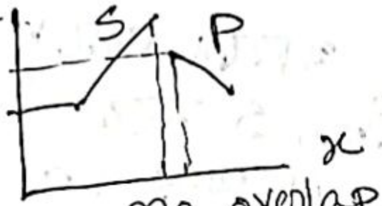
Algorithm

1. Sort all polygon in order of decreasing depth
2. Determine all polygons Q (preceding P) in the polygon list whose z-extents overlap –
3. Continue

→ Bounding rectangle

Test 1:

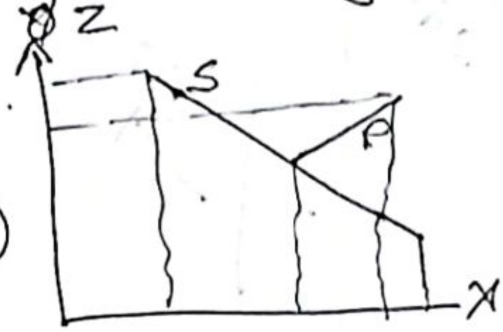
Case 1:



no overlap

Here along z i.e depth we get overlap but in xy plane no overlap. So, scan S first then P.

Case 2: (depth overlap)



Bounding rectangle of P is completely inside S. So, we scan P first then S

Case 3:

in this case P is slightly/Partially obscured by S. So, we do test 2

Now

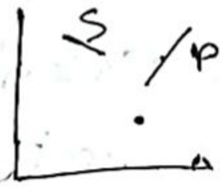
If either of the direction x/y show no overlap, then the two planes cannot obscure one other.

Test 2 & 3 with an "inside-outside" polygon test.

For inside →

Test 2: Check if surfaces is outside of surface P (relative to view plane)

S completely outside



$Ax + By + Cz + D > 0$

For S ↗ then S is outside of P

মানে হলো দেখতে হবে P কি completely S এর front এ আছে কিনা। থাকলে আগে P print হবে then S হবে।

আর যদি completely সামনে না থাকে i.e. overlap ও থাকে then test-3 তে যাবো।

P completely inside

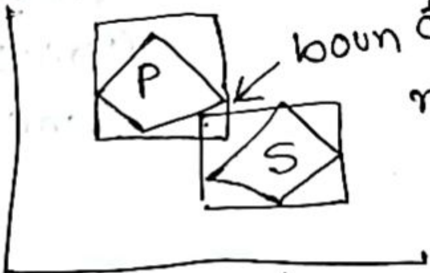
Test 3: if P is complety inside the surface of S. $Ax + Bx + Cx + D < 0$

here P is completely inside of S. So, just scan convert S only.

Test 4:

Boundary rectangle overlap but projection don't then use line equation and test for intersection

$y = mx + c$



bounding rectangle

If intersection then don't Scan convert

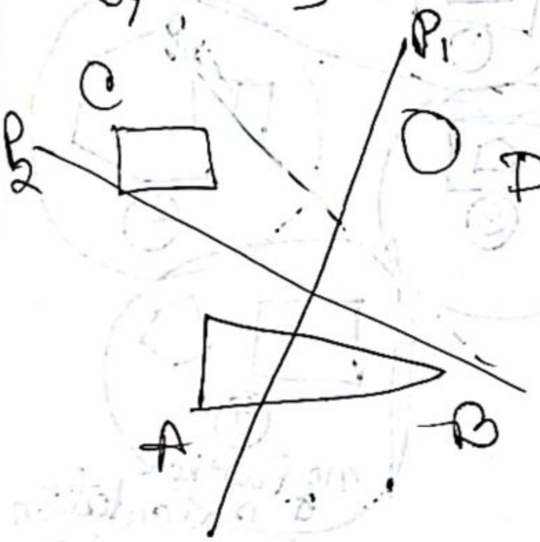
If NO intersection then scan convert S.

BSP Tree method - Binary Space Partitioning

For determining object visibility by painting surfaces onto the screen from back to front, as in the Painter's algorithm.

useful when view reference point changes, but the object in a scene are at fixed position.

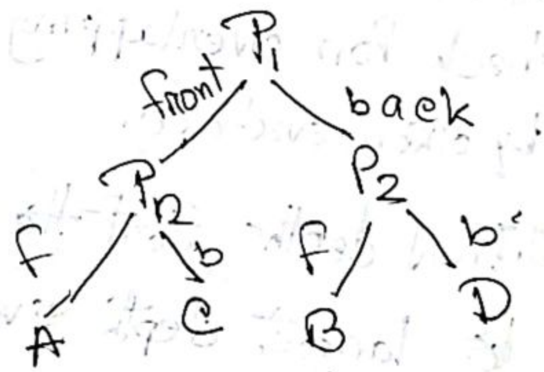
① We need to partition the whole surface by using partitioning plane.



Partition Plane

P_1	A, C	Front
	B, D	Back
P_2	A, B	front
	C, D	Back.

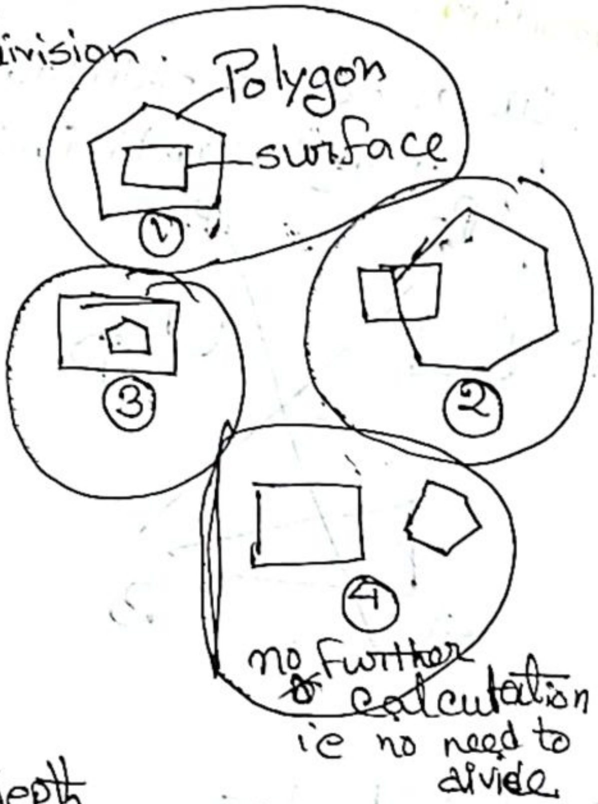
Front = left edge
Back = Right edge



Area Subdivision Method → I.S.M.

If surface present in an area then divide that area.
No surface no division.

If after consecutive divisions we get the area of the smallest divison is 1 pixel then don't do any further division.



Surface types :-

1. Surrounding surface
2. Overlapping "
3. Inside "
4. Outside "

For ① & ② we check for overlapping by other surface.

Find depths, Sort the depth
largest depth value surface be scanned first.

Z value lowest is considered first

the surface

1. Surrounding:- Completely encloses the area
2. Overlapping : some inside & some portion outside

[4] No further subdivision of a specified area are needed if one of the following Condition is true:-

Conditions -

[1] All surfaces are outside surface with respect to the area

[2] Only one inside, overlapping or surrounding surface is in the area

[3] A surrounding surface obscure all other surfaces within the area boundaries.

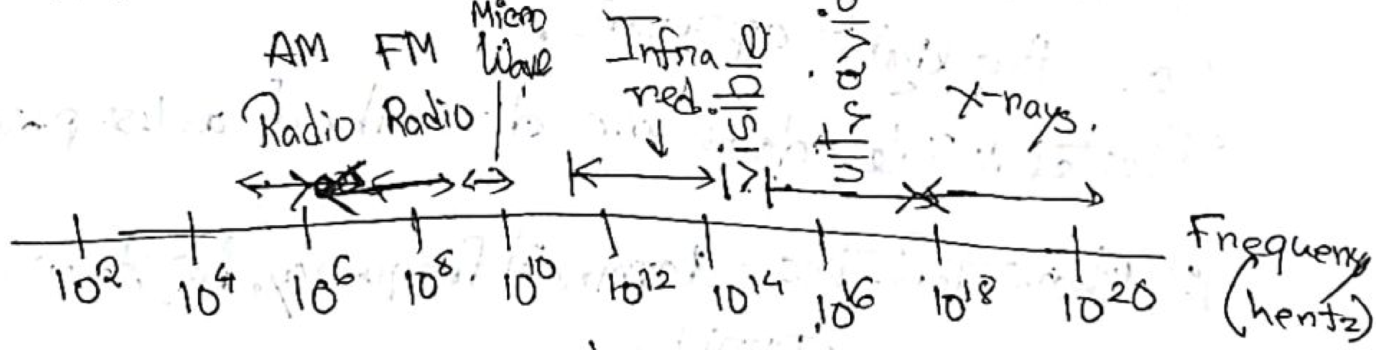
CH-15 Color Models & Color Application.

Properties of light :- Each Frequency value within the visible band corresponds to a distinct color. At the low frequency end is a red color (4.3×10^{14} Hz) and the highest frequency we can see is a violet color (7.5×10^{14} hertz).

Spectral colors range from the reds through orange and yellow at the low-frequency end to greens, blues and violet at high end.

Red → Orange → Yellow → Green → Blue → Violet

লা ~~কমলা~~ হ স নী বে



Electromagnetic spectrum

$c = \lambda f$

If low frequency are predominant in reflected light the object is described as red.

We say the perceived light has a dominant frequency (or dominant wavelength) at the red end of spectrum.

3 The DOMINANT FREQUENCY IS CALLED HUE (color of light or simply the color of the light.

2 basic sense

উজ্জ্বলতা

1 → Brightness - The perceived intensity of light.

Intensity - the radient energy emitted per unit time, per unit solid angle per unit projected area of the source.

Radiant energy is related to luminance of the source.

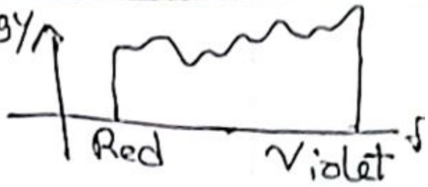
2 → Purity or Saturation of light :- Purity describes how washed out or how "pure" the color of the light appears.

Pastel & Pale colors are described as less pure.

3 Characteristics - dominant frequency, brightness, purity

used to describe different properties we perceive in a source of light

Chroma parameter - Hue & Saturation
Luma " - Luminance.



Chromaticity : is used to refer collectively to 2 properties describing color characteristics
(1) Purity & (2) Dominant frequency.
Saturation   Hue

Energy emitted by a White-light has a distribution over the visible frequency.

When dominant source present that frequency is dominant

E_D = Energy density of dominant light
E_W = Energy of white light

Purity = $E_D - E_W$

Purity → 100% when $E_W = 0$
Purity = 0% when $E_W = E_D$.

* If the two color sources combine to produce white light, they are referred to as Complementary colors. eg - Red + CYAN, GREEN + Magenta, blue + Yellow.

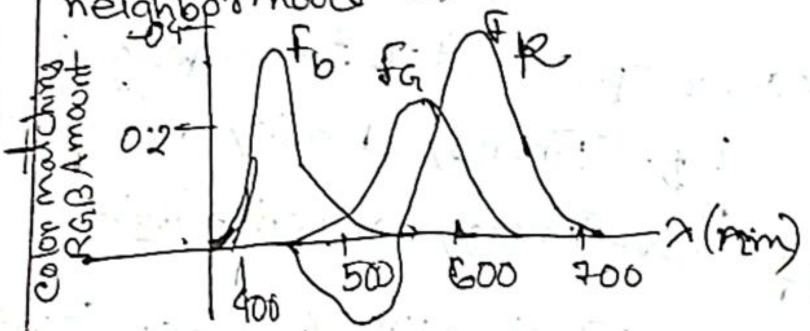
Color Models used to describe Combination of light in terms of dominant frequency (HUE) usin 3 colors. to obtain a reasonably wide range of colors called COLOR GAMUT for that model.

3 colors - Primary Color.

area near
proximity in space.
আওতা

Colors in vicinity of 500 nm can only be matched by "subtracting" an amount of Red light from combination of blue & green.

RGB color monitor cannot display color in neighborhood of 500 nm.



CIE (Commision International de l'Eclairage)

If all frequencies are absord, Color = black

light bounced off by object :- reflected light

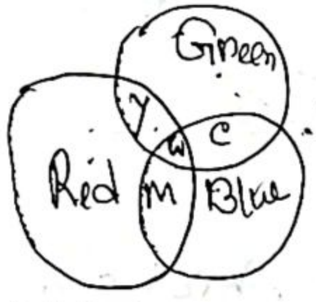
Color :- is the combination of frequencies reflected.

Eliminate (-ve) value color matching and other problems associated with selecting a set of real primaries.

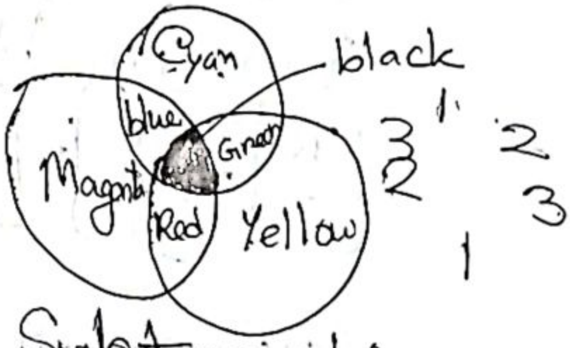
Additive Color	Subtractive Color
Adding together lights of different primary colors	By subtracting light of different colors from black ink
If all colors are mixed with equal intensity, white is produced	If all color's are present, black is produced
If no color is mixed, black is produced	If all colors are substracted, white is produced.
Basic Colors:- Red, green, blue	Cyan, Magenta, Yellow
$R+G+B = W$ $G+B = Y$ $B+R = Mag$ $b+G = Cy$	$Y + mag = red$ $Y + C = G$ $M + C = Blue$

equal intensity is used here

M= Magenta



Additive

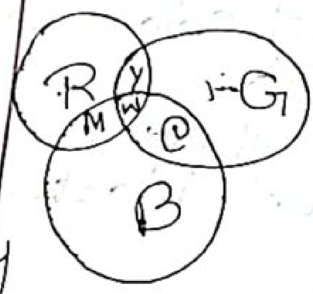


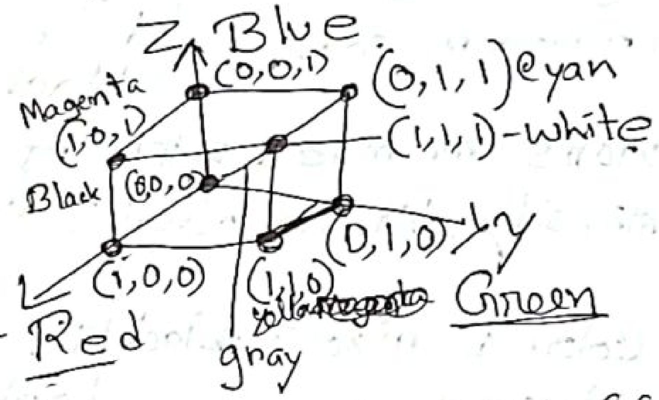
Subtractive

It ~~use~~ is result of transmitting light	It is result of reflected light.

RGB Model

Additive model, Red green blue, light is used to display color, color result of transmitted light





diagonal wise complementary color.

- Used in- T.V, PC monitor, camera, scanners
- Used for Web graphics, but ~~not~~ can not be used for print production
- It directly reflects the physical properties of "True-Color" display.

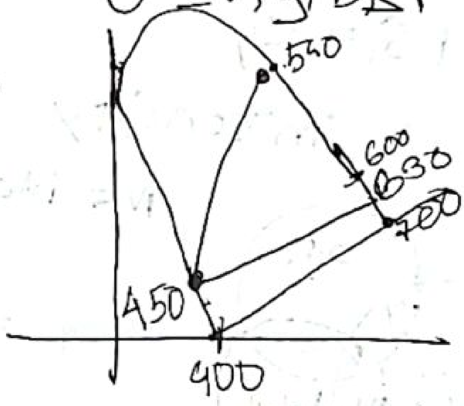
$0 \le r, g, b \le 1$

630 nm (RED)

530 nm (green)

450 nm blue.

$$\begin{bmatrix}R\\G\\B\end{bmatrix} = \begin{bmatrix}1\\1\\1\end{bmatrix} - \begin{bmatrix}C\\M\\Y\end{bmatrix}$$



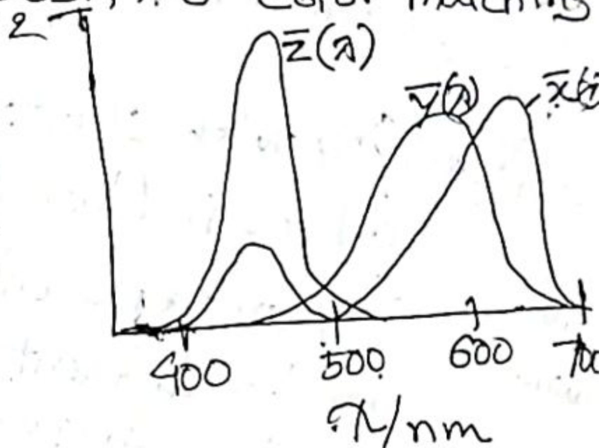
Color gamut → a colour space which can be created by a particular color model.

The entire range of colors & tones achievable by an imaging system

CIE Chromaticity Diagram

Commission Internationale de l'Eclairage (1931)

* Not a computer model but a theorietical one.
* Imagine X, Y, Z represent vectors in a 3-dimen additive color space.
* These X, Y, Z are imaginary primary colors.
* Mathematically defined as positive color matching functions.



* Additive method :

each color = a weighted sum of 3 imaginary primary colors

$C = aX + bY + cZ$

a, b, c designates amounts of standard primaries needed to match C.

* Normalized amounts are calculated as –

$$x' = \frac{a}{a+b+c}, \quad y' = \frac{b}{a+b+c}, \quad z' = \frac{c}{a+b+c}$$

$x' + y' + z' = 1$ and lies in range (0, 1)

2 given third easily found $\Rightarrow z' = 1 - (x' + y')$

⇒ To compute the actual amount from normalized amount →

* At least one actual amount of one of primary is given.
* Two normalized amounts and one actual amount.

For example, if x', y' & b are known then

$$\frac{x'}{y'} = \frac{a}{b}, \quad a = b \cdot \frac{x'}{y'}$$

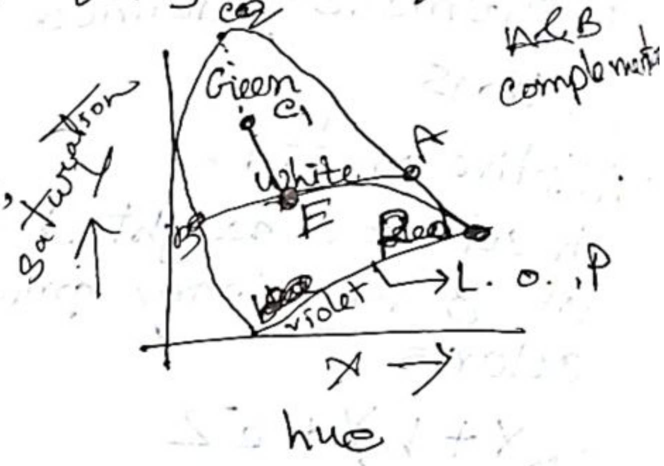
$$z' = 1 - (x' + y'), \quad \frac{y'}{z'} = \frac{b}{c} \text{ so, } c = b \cdot \frac{z'}{y'}$$

Chromaticity
→ hue → frequency dominant
→ Saturation (purity)

No. lumination

CIE Chromatic diagram is a plot of X vs Y for all visible color using x and y.

Only with 2 color differ in Chromaticity, they are represented by two different point in diagram



Color on boundary → Completely Saturated ↓ pure color

Connecting line b/t^n red & violet on purple line

get hue & saturation of a particular color, denoted by point c?
① join c & E using straight line. Extend the line towards boundary. (touches boundary at c2)

hue or dominant wavelength $= c2$

$$\text{Saturation or purity} = \frac{\text{length of line from c1 and E}}{\text{length of line from c2 and E}}$$

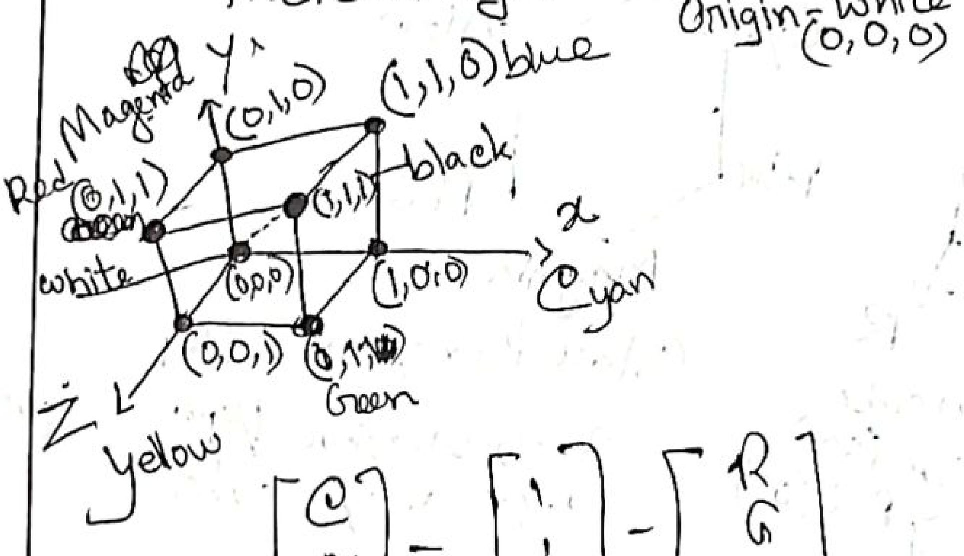
Advantage of CIE Chromaticity Diagram

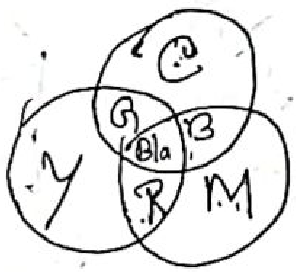
- Comparing Color gamuts for different set of primary color
- Identify Complementary colors
- Determining dominant wavelength & purity of color.

CMY Color Model Cyan, Magenta, Yellow

→ Useful for describing color output of hard-copy devices → eg. plotter produce a color picture by coating a paper with color pigment.

- Color by Reflected light, subtractive process.
  - Cyan has no red component. So, it substract red from incident light.





$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

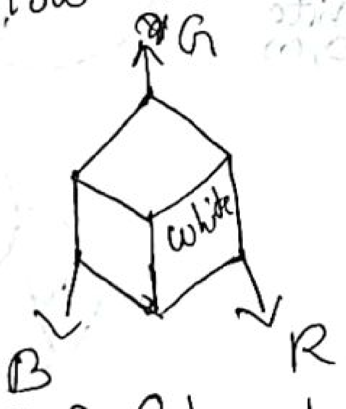
→ Cylindrical Color model (HSL is also one)

HSV → hue, Saturation Value

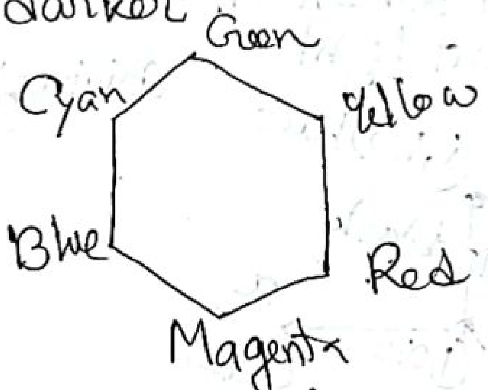
hue :- hue (the dominant frequency / wavelength)
color wheel → 0 to 360. Represents
basic color of pixel / image element.

Saturation:- Purity of Color.. 0% to 100%
It means how much white light
is added to hue to wash it off.
Full saturation - no white light.

Value :- Brightness of color represented
by a percentage of 0% to 100%
Represents amount of black or white
light added to hue. high value is brighter
low Value is darker

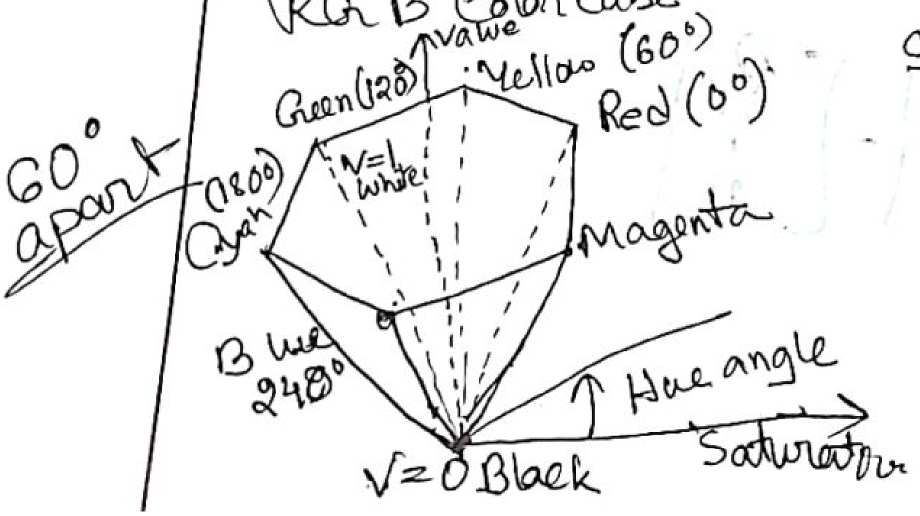


RGB Color cube



Color hexagon

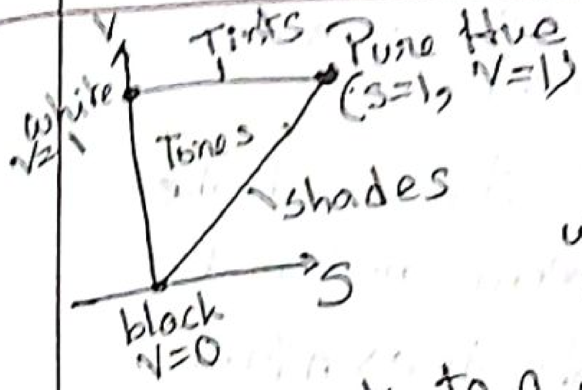
Complementary 180°



$S = 0.25 \Rightarrow$ hue 1/quarter pure

$S = 0$ gray scale

In the one section of hexagon color



Tones ⇒ Various tones are specified by adding both black and white, producing color points within the triangular cross-sectional area of the hexacone

⇒ Adding black to a pure hue decreases Value (V) down the side of hexcone. Thus, various Shades are represented with value S=1, & $0 \le V \le 1$

⇒ Adding white to a pure tone produces different Tints across the top plane of the hexcone, where parameter value V=1 and $0 \le S \le 1$

128 hues, 8 saturation level, 15 value.
HSV - 16384 colors available, 14 bit color storage pixel.
Color look up table

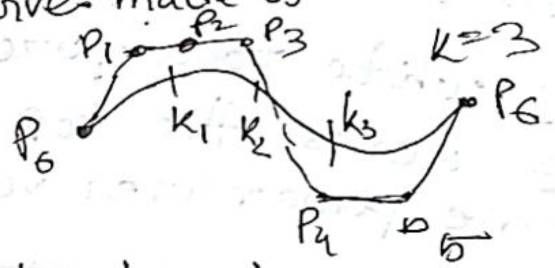
Characteristics of light:
1. Color:- Different color depending on different wavelength.
2. Intensity: Refers to brightness of light source
3. Direction:- Determines how light falls on objects and creates shadow & highlights
4. Shadow:- Created when light is blocked by an object, can help to give depth and dimensionality to a scene.

5. Reflection:- Reflects off surface, creating specular highlights and reflections.

6. Refraction: Light can refract or bend when it passes through different material.

7. Attenuation: Can lose intensity as it travels through space

B-Spline : after bazier Curve. ① made of n+1 Control points
Order of curve (k)



② Change on one control point changes that segment only. i.e. local control over curve

③ Used to draw open & closed curve

④ Gives us polynomial of degree k-1

$k=3 \quad P(x)=x^2$

$k=4 \quad P(x)=x^3$

⑤ $n-k+1$ segments ~~control points~~

$n=6,\ k=3$

control points $6+1=7$

Segments $= n-k+1 = 5$

$6-3+1$

0-1, 1-2, 2-3, 3-4, 4-5

⑥ Allow change of # of Control points without changing degree of polynomial

$0 \le u \le n-k+2$

$2 \le k \le n+1$

$$P(u) = \sum_{i=0}^{n} B_{ik}(u) P_k$$

$$P_k^i = (x_k^i, y_k^i, z_k^i)$$

$$x(u) = \sum_{i=0}^{n} B_{ik}(u) x_k$$