Unit - IV

"Curve generation for blobby object or Curve generation :- blobbiness of object"

Problem in true-curve generation approach :-

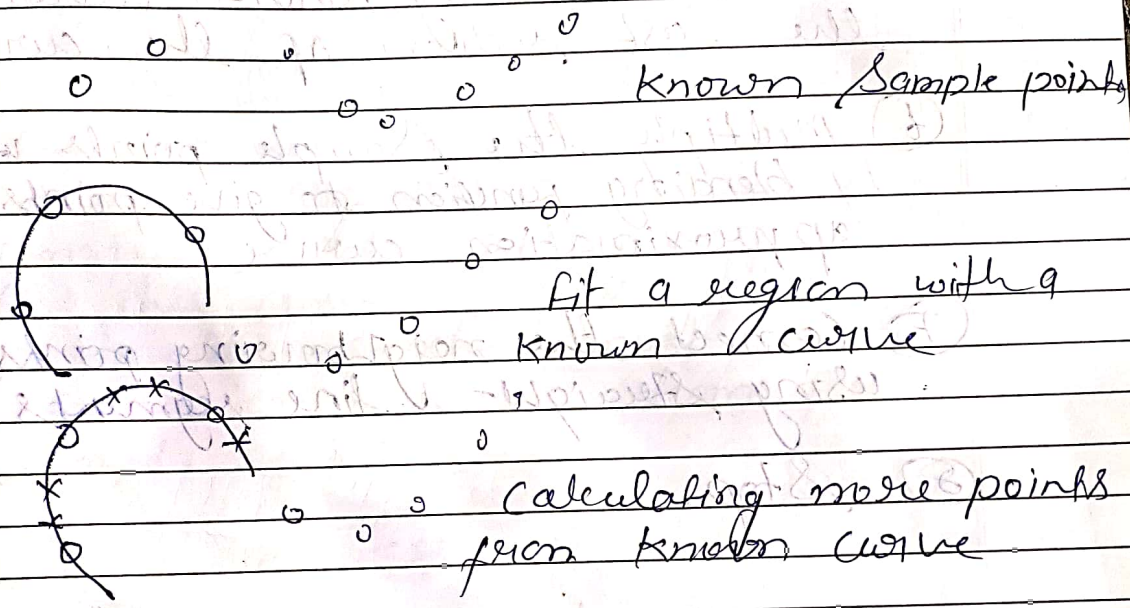
(1) To specify a curve, we need more information than just it's endpoints. this may mean that a different display file structure should be used.

(2) Another problem can arise from transformation. A line segment when scaled is still a line segment, but other curves may behave differently for example, a circle when scaled in only one direction becomes an ellipse.

(3) if we wished to clip arcs to some window boundry a new clipping algorithm would have to be developed.

Interpolation OR **Approximation** :- instead of using readymade algorithm we are approximating the curve by small straight lines. For this we have to use interpolation technique.

A Spline is a flexible strip used to produce a smooth curve through a designated set of points several small weights are distributed along the length of the strip to hold it in position on the drafting table as the curve is drawn.



Interpolating algorithm:-

1. get the Sample points
2. get the intermidiate values of u to determine intermediate points
3. Calculate blending function values

for middle section of the curve

(4) Calculate blending function values for first section of the curve

(5) Calculate blending function values for the last section of the curve

(6) Multiply the sample points by blending function to give points on approximation curve

(7) Connect the neighbouring points using straight-line segments

(8) Stop.

To generate a curve by approximate method, we are using parametric form which can adjusting parameter

$$x = f_x(u)$$
$$y = f_y(u)$$
$$z = f_z(u)$$

[parametric form]

* to find the point between the sample points. $(u_5) = ?$

$$P(u) = \sum_{i=1}^{n} P_i B_i(u)$$

Here we are using parametric form because it considers all three direction equally and it allows multiple values so that curve can cross itself.

$(x_1, y_1, z_1), (x_2, y_2, z_2) \text{ ----- } (x_n, y_n, z_n)$

as we are using parametric form the equation of the curve should be $x = f_x(u)$ it means we have to construct some functions for this curve As this curve is passing from n sample points the function must be the sum of n terms, one term for each sample point

there for

$$f_x(u) = \sum_{i=1}^{n} x_i B_i(u)$$

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

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

here for each function $f_x(u)$, $f_y(u)$, $f_z(u)$ we are using one common function $B_i(u)$ this function is called Blending function.

using these bending function with n sample points, we can construct a curve which passes through the n sample points

$$X = x_1 B(q) + x_2 B(q) + \dots + x_n B(q)$$

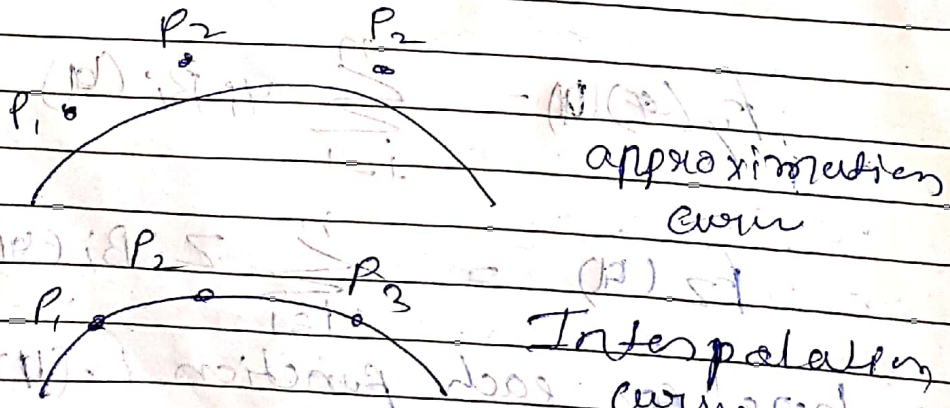
$$Y = y_1 B(q) + y_2 B(q) + \dots + y_n B(q)$$

$$Z = z_1 B(q) + z_2 B(q) + \dots + z_n B(q)$$

~~B-Spline curves~~

(1) Bezier Curve :-

(i) It is a spline curve

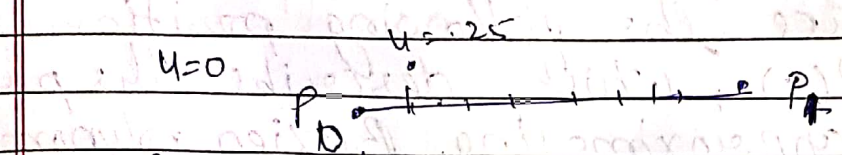


polynomial → $P(x) = a_0 x^n + a_1 x^{n-1} + a_2 x^{n-2}$

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

(i) Bezier curve is a approximate curve.

(ii) Bezier curve is a parametric curve



Control point
Sample point

(iii) we can get n degree polynomial with n+1 control point.

(iv) we can use CAD, typeface, Drawing use

(v) easy to implement.

(vi) it has global control

Bezier curve have a number of properties that make them highly useful and convenient for curve and surface design. a Bezier curve section can be fitted to any number of control points. the number of control points to be approximated and their relative position determine the degree of Bezier polynomial. as with the interpolation splines, a Bezier curve can be specified with boundry condition with a characterizing matrix or with blending function.

Suppose we are given n+1 control points $P_k(x_k, y_k, z_k)$ with k varying from 0 to n. These co-ordinate points can be blended to produce the following position vector $P(u)$, which describe the path of an approximating Bezier polynomial function between P_0 and P_n

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

The Bezier blending function $BEZ_{k,n}(u)$ are the Bernstein polynomials.

$$BEZ_{k,n}(u) = C(n,k)\, u^k (1-u)^{n-k} \quad \text{(IV)}$$

where the $C(n,k)$ is the binomial coefficient

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

$(P-1)$ = n
P = control points
k = 0 — n

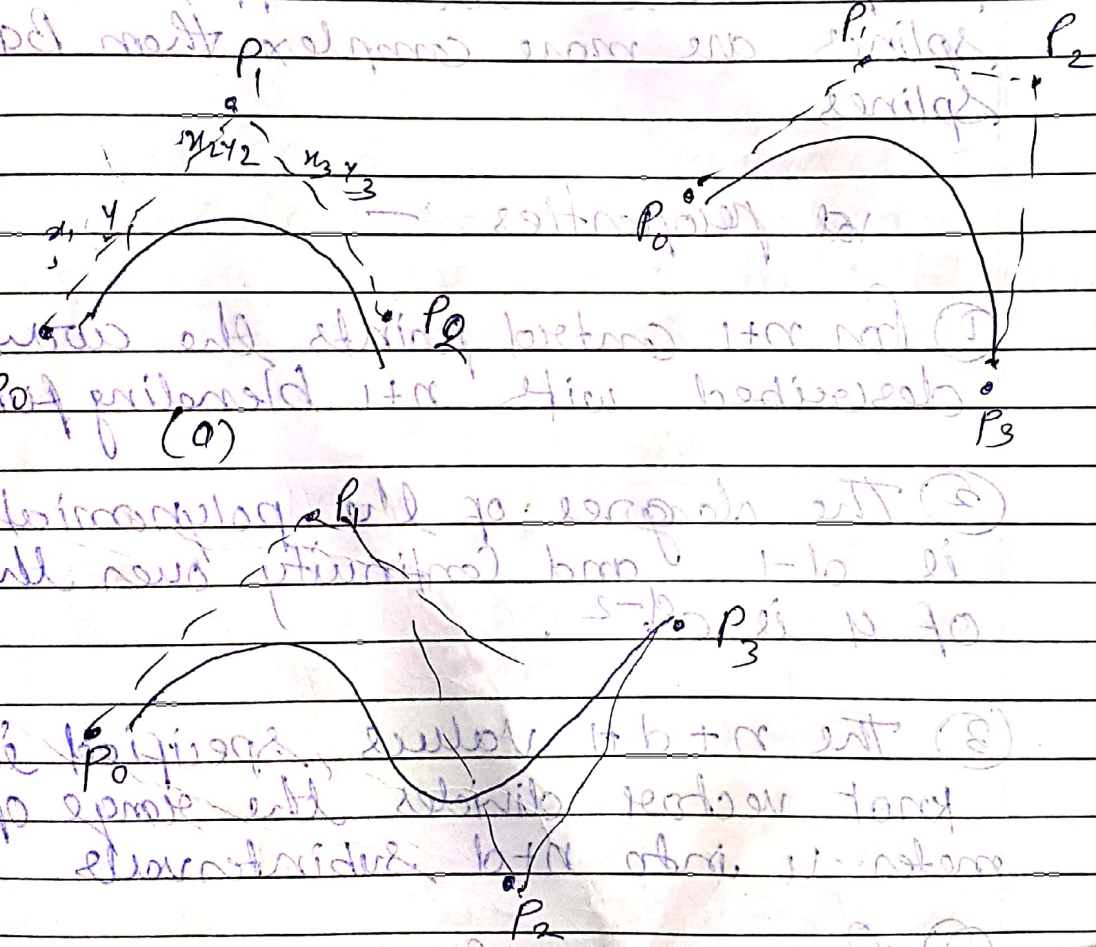
Equivalently we can define Bezier blending function with the recursive equation:-

$$BEZ_{k,n}(u) = (1-u)\, BBZ_{k,(n-1)}(u) + u\, BBZ_{k-1,(n-1)}(u) \quad k \ge 1$$

$$x(u) = \sum_{k=0}^{n} x_k BEZ_{k,n}(u) +$$

$$y(u) = \sum_{k=0}^{n} y_k BEZ_{k,n}(u)$$

$$z(u) = \sum_{k=0}^{n} z_k BEZ_{k,n}(u)$$



B-Spline curve :-

B-spline have 2 advantages over Bezier Splines

(1) The degree of B-spline polynomial can be set independent of the number of the control points and

(2) B-spline allow local control over the shape of a spline curve or surface. the trade-off is that B-splines are more complex than Bazier splines

properties :-

(1) For n+1 control points the curve is described with n+1 blending function.

(2) The degree of the polynomial curve is d-1 and continuity over the range of u is C^{d-2}.

(3) The n+d+1 values specified in the knot vector divides the range of parameter u into n+d subintervals.

(4) Each blending function $B_{i,d}$ is defined over d subintervals of total range of u.

(5) segments of curve.
(6) it has local control.

(7) each section of the splice curve is influenced by d control points.

(8) with knot values labeled as ($u_0, u_1, \ldots, u_{n+d}$) the resulting B-spline curves is defined only in the interval from knot value u_{d-1} up to knot value u_{n+1}

(9) Any one control point can affect the shape of almost d curve section.

We can write a general expression for the calculation of coordinate position along a B-Spline curve in a blending function formulation as -

$$P(u) = \sum_{k=0}^{n} P_k B_{k,d}(u)$$

k = each segment influenced by k point

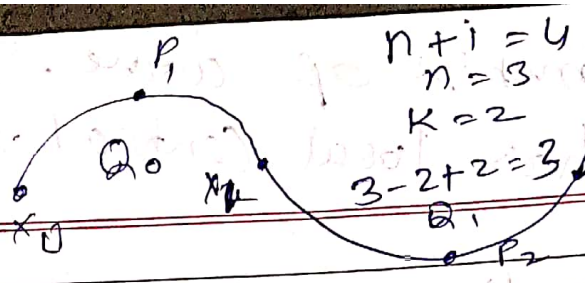
$B_{k,d}$ = B-spline Blending fun

$u_{min} \leq u \leq u_{max}$

$2 \leq d \leq n+1$

where the P_k are an input set of $n+1$ control points. There are several differences between this B-spline formulation and that for Bezier splines. The range of parameter u now depends on how we choose the B-spline parameter and the B-spline blending function ~~over subinterval of~~ $B_{k,d}$ are

K = Control point knot values



polynomials of degree d-1, where parameter d can be chosen to be integer value in the range from 2 up to the number of control points n+1, local control for B-spline is achieved by defining the blending function over subintervals of the total range of u.

Blending function for B-spline curves are cox-deboor recursion formulas.

$$B_{k,1}(u) = \begin{cases} 1, & \text{if } u_k < u < u_{k+1} \\ 0, & \text{otherwise} \end{cases}$$

$$B_{k,d}(u) = \frac{u - u_k}{u_{k+d-1} - u_k} B_{k,d-1}(u) + \frac{u_{k+d} - u}{u_{k+d} - u_{k+1}} B_{k+1,d-1}(u)$$

where, each blending function is defined over d subintervals of the total range of u. The selected set of the subinterval

Numerical for bazier curve :-

~~A Bazier curve pases through the point (1, 1) &~~

The Cordinate of four control points relative to a curve are given by P_1 (2, 2), P_2 (3, 3) P_3 (3, 3) & P_4 (3, 2)

write eqn of bazier curve also find the co-ordinate pixel of curve for $U = 0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}$ also plot bezier curve on graph.

$P_1 = x_1 = 2$ $\quad$ $P_2 = x_2 = 2$

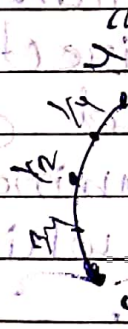
$y_1 = 2$ $\quad$ $y_2 = 3$

$P_3 = x_3 = 3$ $\quad$ $P_4 = x_4 = 3$

$y_3 = 3$ $\quad$ $y_4 = 2$

eqn =

$$x = x_4 U^3 + 3x_3 U^2 (1-U) + 3x_2 (1-U)^2$$



ILLUMINATION MODELS :-

"A model for the interaction of light with a surface is called an illumination model.

Light Source :-

light - emitting source
light - reflecting source

light emitting sources are like bulb and and sun

light repleting sources indirect wall of room.

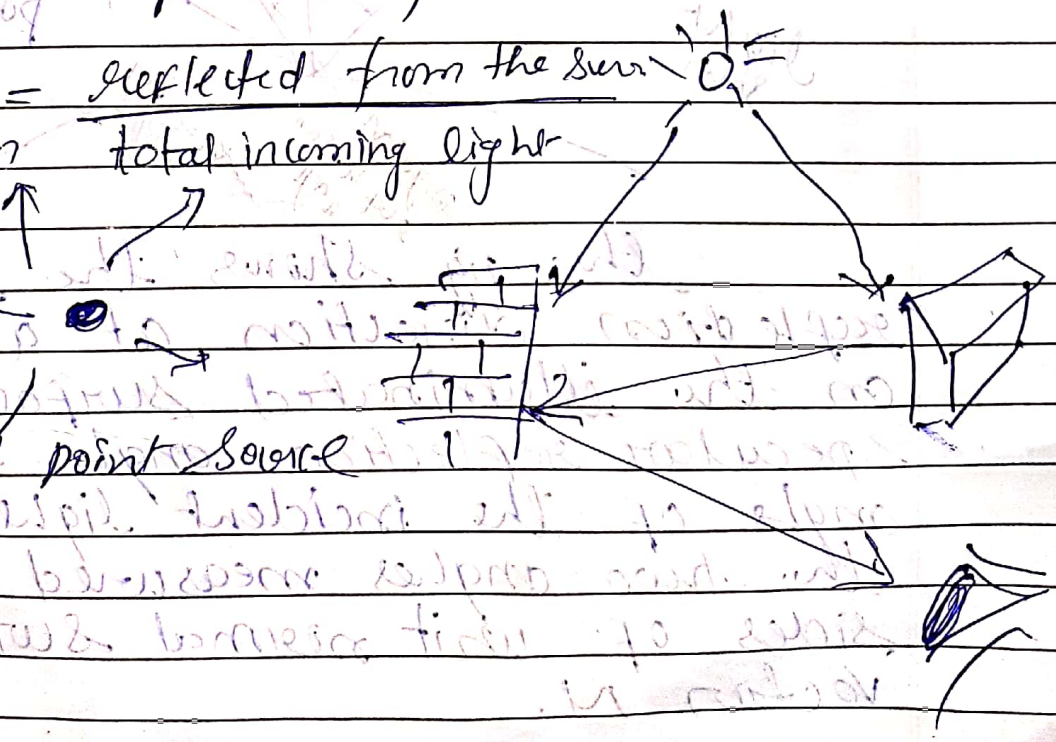
(1) Diffuse illumination :- / diffuse reflecti

We assume that our objects is illuminated by light b which does not come from any particular source but which comes from all direction & that is ambient light. when such illumination is uniform from all direction, the illumination is called diffuse illumination. Actually diffuse illumination is a background light, which is repleted from the wall.

Floors and ceiling we will assume that there is much light going up as there is going down and that there is same amount goint left: when we assume that going up, down, right and left same amount then we can say that the reflection are constent over each surface of abject and they are independent of viewing direction. Such reflection is called diffuse reflectia

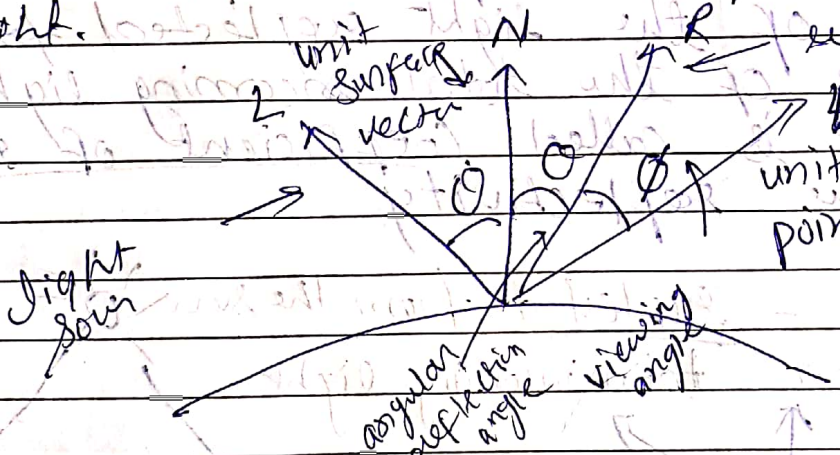
In practice, when object is illeminated, some part of light energy is absorbed by the surface of the abject while the rest of is repleted. The ratio of the light reflected from the surface of the total incoming light to the surface is called coefficient of reflection or the reflectivity

Coff of reflection = reflected from the surface / total incoming light



Specular reflection :-

When we look at illuminated shiny surface such as polished metal or a person's forehead we see a highlight, or bright spot at certain viewing direction, this phenomenon called specular reflection, is the result of total or near total reflection, of the result incident light in a concentrated region around the specular reflection angle. due to specular reflection surface appears to be not in it's original colour. but white the colour of incident light.

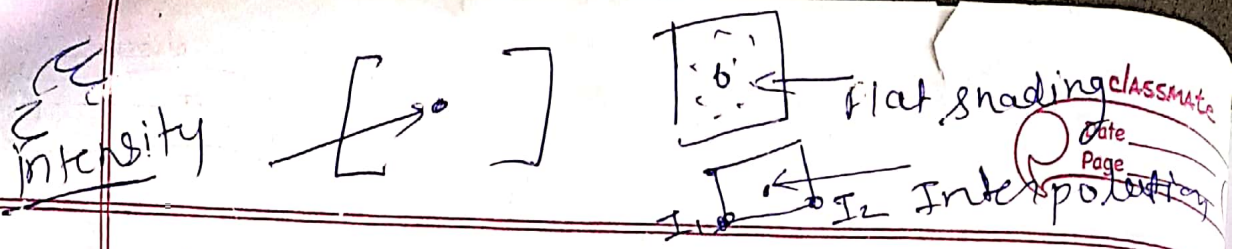


The fig shows the specular reflection direction at a point on the illuminated surface. The specular reflection angle equals the angle of the incident light with the two angles measured on opposite sides of unit normal surface vector N.

R is unit vector in the direction of ideal specular reflection. L is the unit vector directed toward the point light source and V is the unit vector pointing to the viewer from the surface position.

The angle ϕ between vector R and vector V is called viewing angle. For an ideal reflector (perfect mirror) incident light is reflected only in the specular reflection direction. In such case, we can see reflected light only when vectors V and R coincide their $\phi = 0$

Specular reflection angle = angle of the incident light



Shading :- to calculate pixel intensity after projection

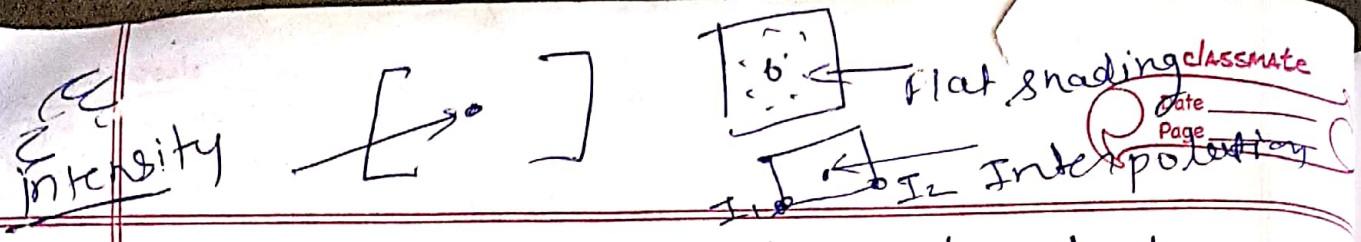
Gouraad shading

Gouraad shading is intensity-interpolation shading developed by Henri gouraad and generally referred to gouraad shading. The polygon surface is displayed by linearly interpolating intensity values across the surface. Intensity values for each polygon are matched with the value of adjacent polygons along the common edge. Thus, eliminates the intensity discontinuities that can occur in flat shading.

Each polygon surface is rendered with gouraad shading by performing following calculations.

① determine the average unit normal vector at each polygon vertex.
② apply an illumination model to each polygon vertex to determine the vertex intensity.
③ Linearly interpolate the vertex intensity over the surface of the polygon.

We can obtain a normal vector at each polygon vertex by averaging the surface normals of all polygons sharing



Shading :- to calculate pixel intensity after projection

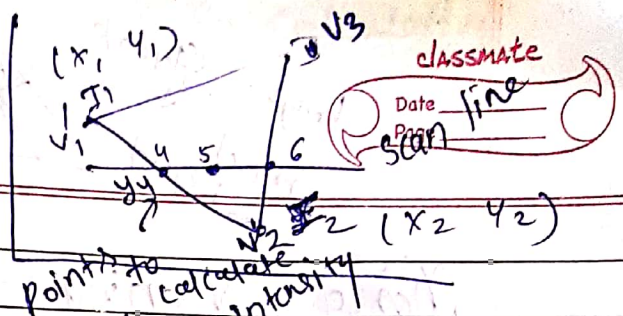
Gouraad shading

Gouraad Shading is intensity - interpolation shading, developed by Henri gouraad and generally referred to gouraad shading the polygon surface is displayed by linearly interpolating intensity values across the surface. intensity values for each polygon are matched with the value of adjacent polygons along the common edge. Thus, eliminates the intensity discontinuities that can occur in flat shading.

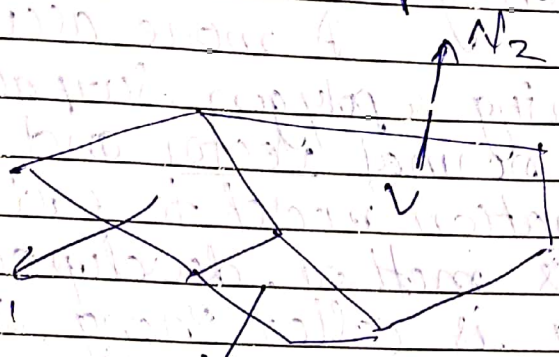
Each polygon surface is rendered with gouraad shading by performing following calculations.

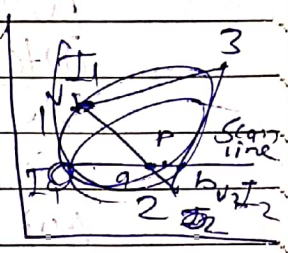
1. determine the average unit normal vector at each polygon vertex.
2. apply an illumination model to each polygon vertex to determine the vertex intensity.
3. Linearly interpolate the vertex intensity over the surface of the polygon.

We can obtain a normal vector at each polygon vertex by averating the surface normals of all polygons sharing



that vertex.





I_1 = intensity
I_2 = intensity

To compute normal vector

(1) $N_v = \frac{N_1 + N_2 + N_3}{|N_1 + N_2 + N_3|}$ $\quad N_v = \sum_{i=1}^{n} N_i$ $\quad (V_1, V_2)$ $\quad \left|\sum_{i=1}^{n} N_i\right|$

(2) apply illumination model to calculate intensity over the average normal vertices

For Calculate Intensity

(3) $I_4 = \frac{y_4 - y_2}{y_1 - y_2} I_1 + \frac{y_1 - y_4}{y_1 - y_2} I_2$

(3) ~~The mach band effect (also as~~

Disadvantages :- (1) Highlights on the surface are sometimes displayed with anamalous shapes. (2) The linear intensity interpolation can result bright or dark intensity streak to appear on the surface these bright or dark intensity streaks are called mach band

Advantages :- (1) gouraud shading can be combined with a hidden surface algorithm to fill in the visible polygon along each scan line

(2) it removes the intensity discontinuities exist in constant shading model

(interpolation scheme of rendering)

Phong Shading :-

A more accurate method for rendering a polygon surface is to interpolate normal vector and then apply the illumination model to each surface point, this method developed by phong Bui thoung is called phong shading or normal - vector interpolation shading interpolates the surface normal vector N, instead of the intensity

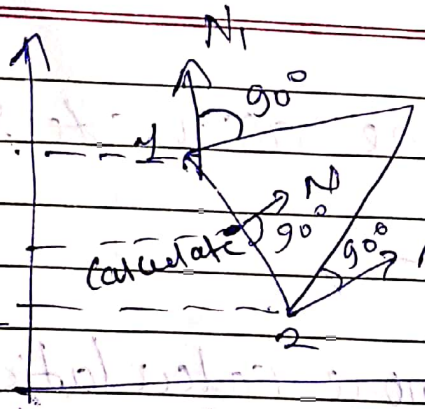
By performing following steps, we can display polygon surface using phong shading

1. determine the average unit normal vector at each polygon vertex.
2. Linearly interpolate the vertex normals over the surface of the polygon.
3. Apply an illumination model along each scan line to determine projected pixel intensity for the surface points.

The first step in the phong shading is same as first step in the Gouraud shading. In the second step the vertex normals are linearly interpolated over the surface of the polygon.

Rendering

(3) apply illumination model for calculating intensity of projected pixel.



(1) determine average normal vector at each vertex
(2) linearly interpolate per pixel normal across surface

Calculating of interpolation of surface normals along a polygon edge.

As shown in figure, the normal vector N for the scan line interaction point along the edge bet'n vertices 1 and 2 can be obtained by vertically interpolating between edge endpoint normals.

$$N = \frac{y - y_2}{y_1 - y_2} N_1 + \frac{y_1 - y}{y_1 - y_2} N_2$$

Like gouraud shading, here also we can use incremental methods to evaluate normals between scan lines and once the surface normals are evaluate the surface intensity at the point is determined by applying the illumination models.

Advantages :-

(1) it displays more realistic highlights on a surface
(2) it greatly reduces the mach-band effect

3. it gives more accurate result

Disadvantages :-

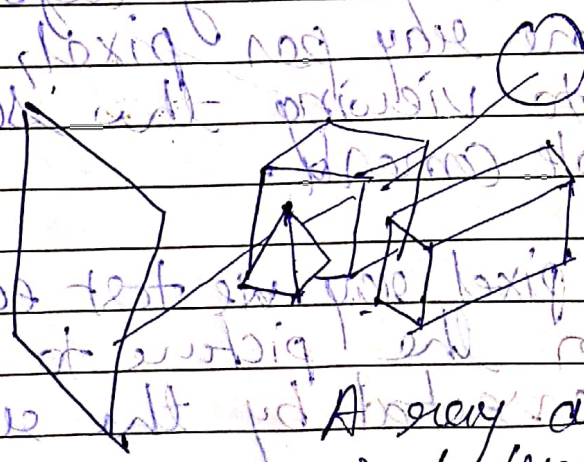
1. it requires more calculations and greatly increases the cost of shading steeply.

Gouraad Shading	Phong Shading
(1) in this, we apply an illumination model to each vertex for calculating the vertex intensity	(1) In this, we apply an illumination model to each scan line for calculating projected pixel intensities for surface point
(2) linearly interpolate the vertex intensities over the surface of the polygon	(2) linearly interpolate the vertex normals over the surface of polygon
(3) gouraad shading takes less time than phong shading	(3) Phong shading takes more time than gouraad shading
(4) it requires less calculations	(4) it requires more calculation

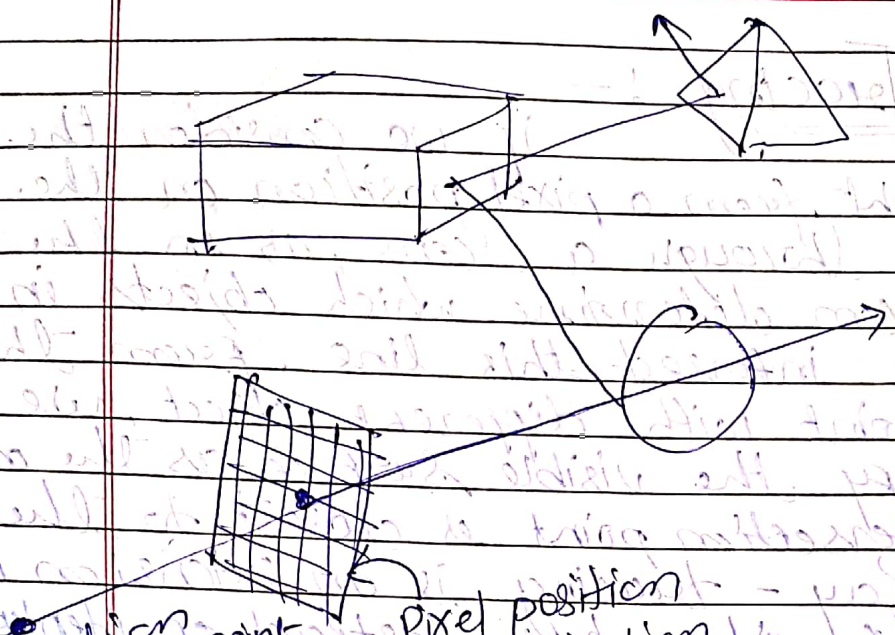
Ray - Tracing :-

if we consider the line of sight from a pixel position on the view plane through a scene, as in the fig. We can determine which objects in the scene intersect this line. From the insection point with different object, we can identify the visible surface as the one whose intersection point is closest to the pixel. Ray - tracing is an extension of this basic idea. Here, instead of identifying for the visible surface for each pixel, we continue to bounce the ray around the picture. When the ray is bounding from one surface to another surface it contributes the intensity for that surfaces. This is a simple and powerful rendering technique for obtaining global reflection and transmission effects.

Basic ray



A ray along the line of sight from a pixel position through a scene



Ray tracing algorithm:- First setup a co-ordinate system with the pixel position designated in the xy plane and projection reference point lie on the Z-axis. From the centre of projection we then determine a ray path that passes through the centre of each screen pixel position.

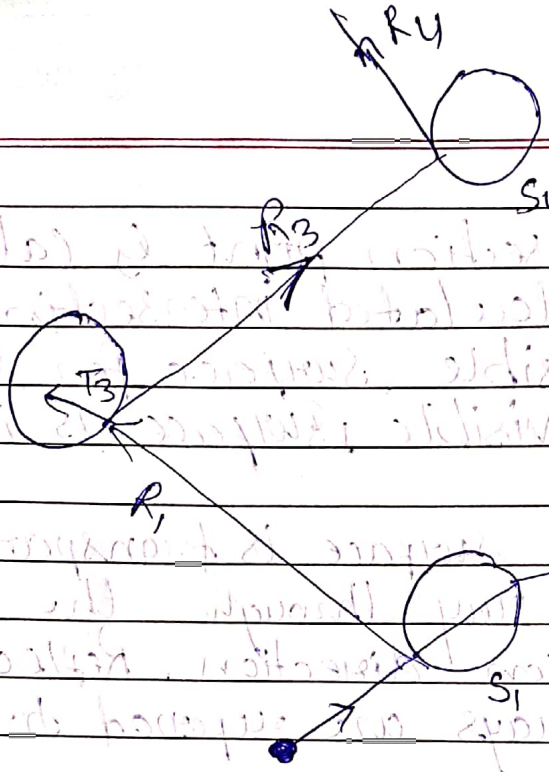
In the basic ray tracing algorithm we consider one ray per pixel, which is equivalent to viewing the scene through a pinhole camera.

For each pixel ray we test each surface which is tested in the picture to determine if it is intersected by the ray.

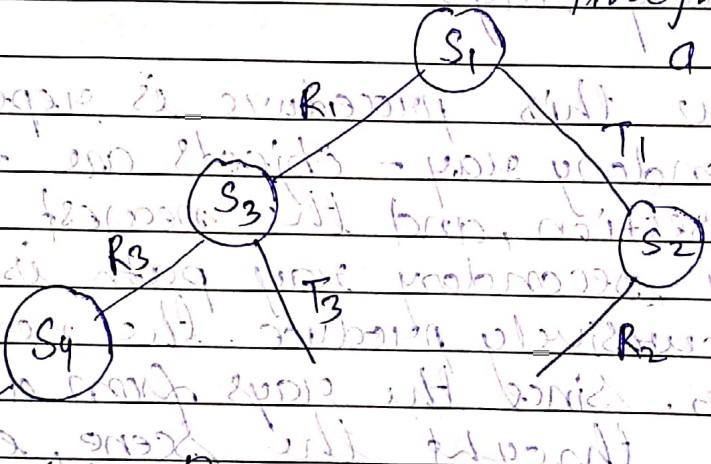
if Surface intersection point is calculated the smallest calculated intersection distance identifies the visible surface for that pixel. once the visible surface is ident

If the surface is transparent we also send a ray through the surface in the refraction direction. Reflection and refraction rays are referred to as Secondary rays.

Now this procedure is repeated for each secondary ray - objects are tested for intersection, and the nearest surface along a secondary ray path is used to recursively produce the next generation paths. Since the rays from a pixel ricochet through the scene, each successively interested surface is added to a binary ray-tracing tree, as shown in fig. we use left branches in tree to represent reflection paths. and the right branches represent transmission path. maximum depth of the ray-tracing trees can be set as a user point, or it can be found by the amount of storage available. Then the path in the tree is terminated if it reaches the preset maximum of it or if the ray strike a light source



(i) Reflection and refraction Ray paths through a scene for a screen pixel



(ii) Binary ray-tracing tree for the paths shown in (a)

then the intensity assigned to pixel is found by accumulating the intensity contri.

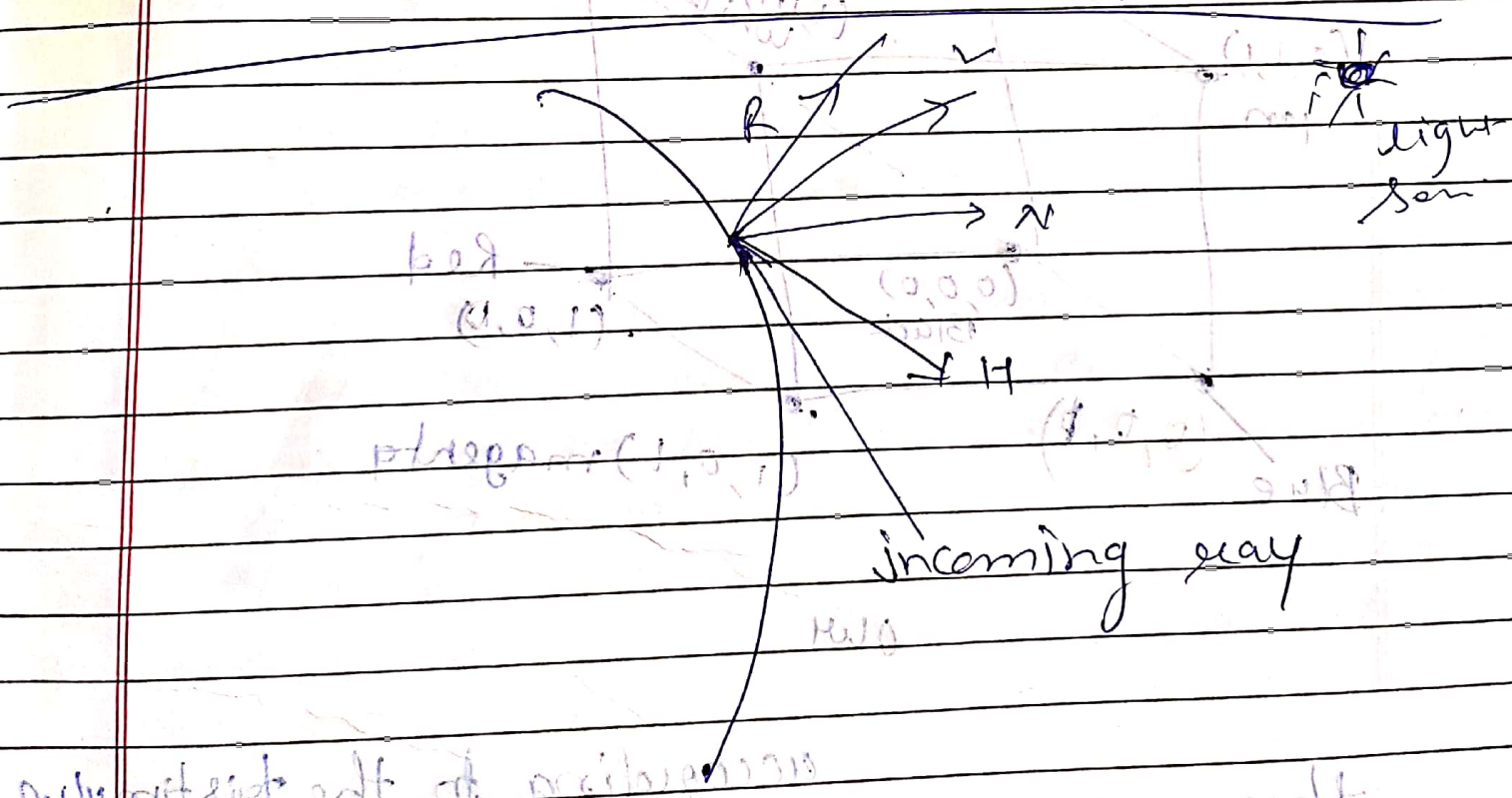
(1) one ray per pixel

Numericals

Pg(84, 91, 100, 103)

- Line drawing, circle drawing
- Transformation (only translation, rotation & scaling)
- Window to viewport
- Line clipping, polygon clipping
- 3-D Transformation
- Projection (Parallel & Perspective)
- Bezier curve & B spline curve.

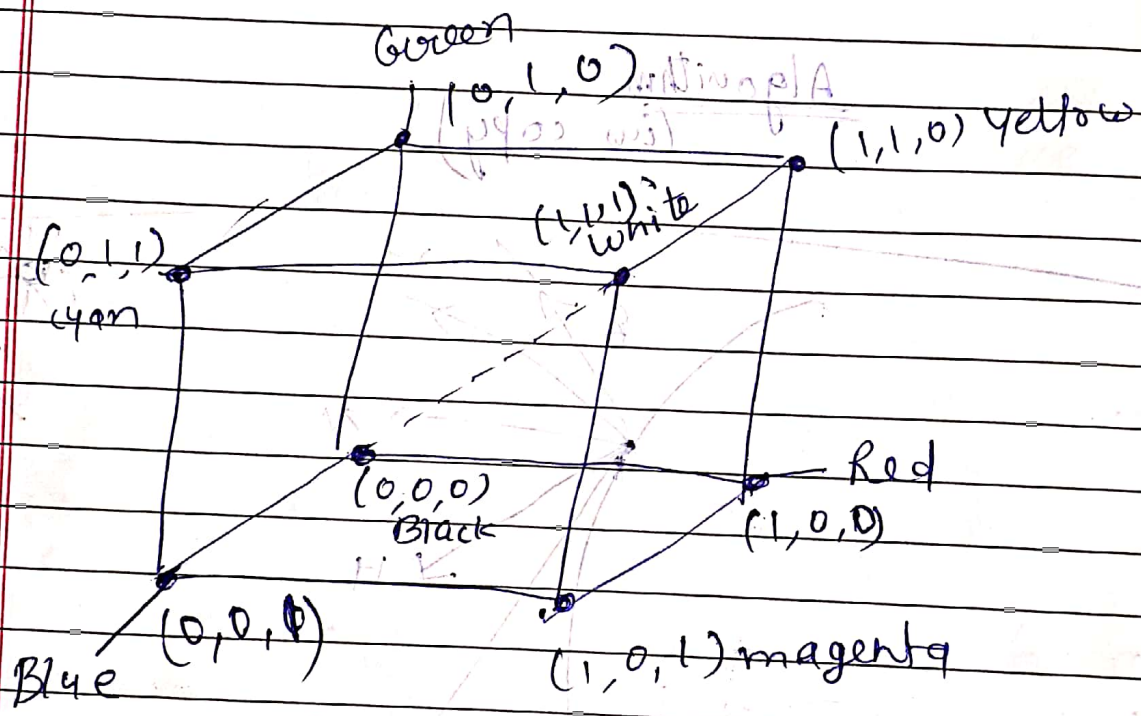
Algorithm (in copy)



Colour models :-

1. RGB
2. YIQ
3. CMY
4. HSV
5. CMYK

(1) RGB :- (Red, green, blue)



OR

according to the tristimulus theory of vision, our eyes preseive color through the stimulation of three visual pigments in the cones of retina. at wavelength of about 630 nm (red), 530 nm (green) and 450 nm (blue). these visual pigment have a peak sensitivity By comparing the different intensities

in a light source, we get the colour of light. This theory of vision is applied to the video monitors using the three color primaries, red, green and blue referred to as the RGB colour model.

as shown in fig. this model can be represented by unit cube defined R, G and B axes. The (0, 0, 0) co-ordinate that is origin represent white. primary colours are the vertices other than on the axes represent the complementary colour for each of the primary colours.

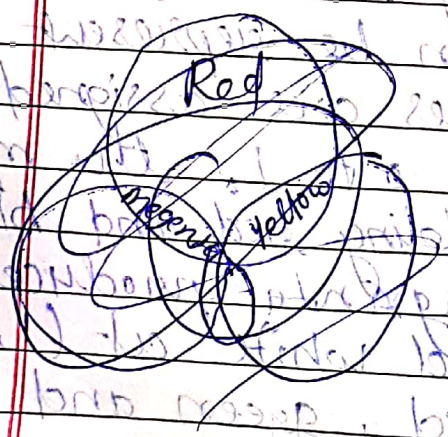
~~in this model, intensities of the primary colours are added~~

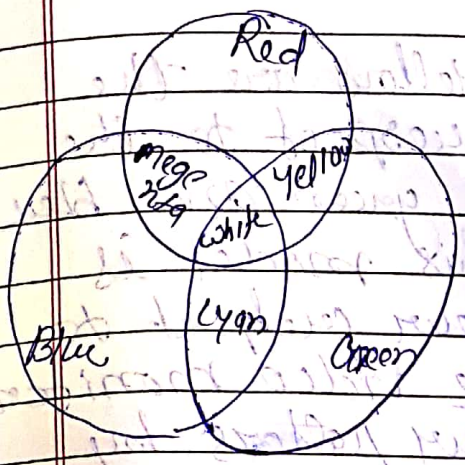
Primary colour intensities are add to produce other colour. each colour point within the cube bound can be represent as triple (R, G, B) where values are assigned within the range from 0 to 1. the magenta vertex is obtained by adding red and blue the vertex representing magenta produces the triplet (1, 0, 1) and white at (1, 1, 1) is the sum of the red, green and blue vertices.

Additive Vs Subtractive Colour models:-

"Since additive colour models display colours as result of light being transmitted or (added) the total absense of light would be perceived as black."

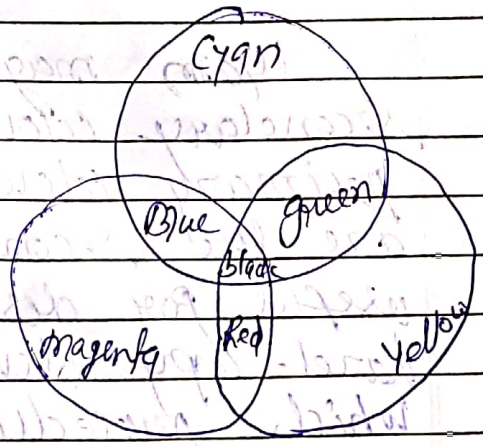
"Subtractive colour models display colour as a result of light being absorbed (subtract) by the printing inks. As more inks is added less light is reflected"





RGB colour model additive colour model for computer displays uses light to display colour. Colour result from transmitted light

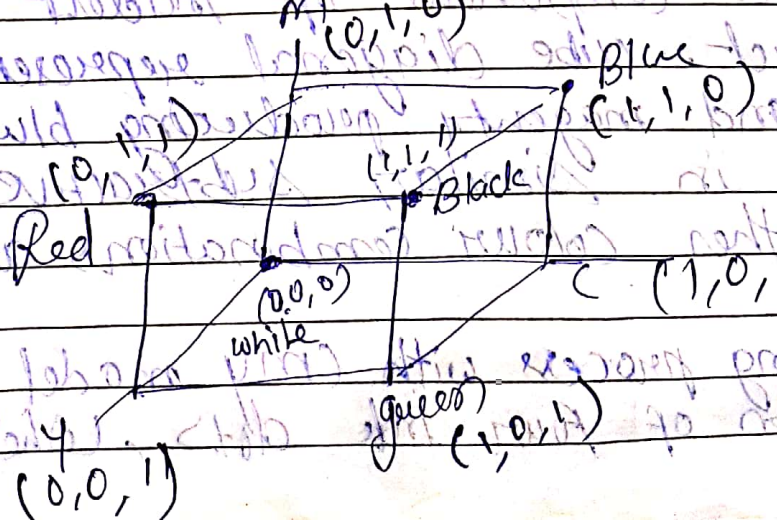
Red + Green + Blue ⟹ White



CMY colour model Subtractive colour model for printed material uses ink to display colour. colours result from reflected light

Cyan + magenta + Yellow = Black

(2) CMY (Cyan, magenta, Yellow)



Cyan magenta and Yellow are the secondary colours with respect to the primary colours of red, green and blue are the secondaries. This model is useful for describing colour output to hard-copy devices. Unlike video monitor which produces a colour pattern by combining light from the screen phosphors hard-copy devices such as plotters produce a colour picture by coating a paper with colour pigments.

Cyan is formed by adding blue and green light. When white light is reflected from cyan coloured ink, the reflected light must have no red component that is red light is absorbed by the ink.

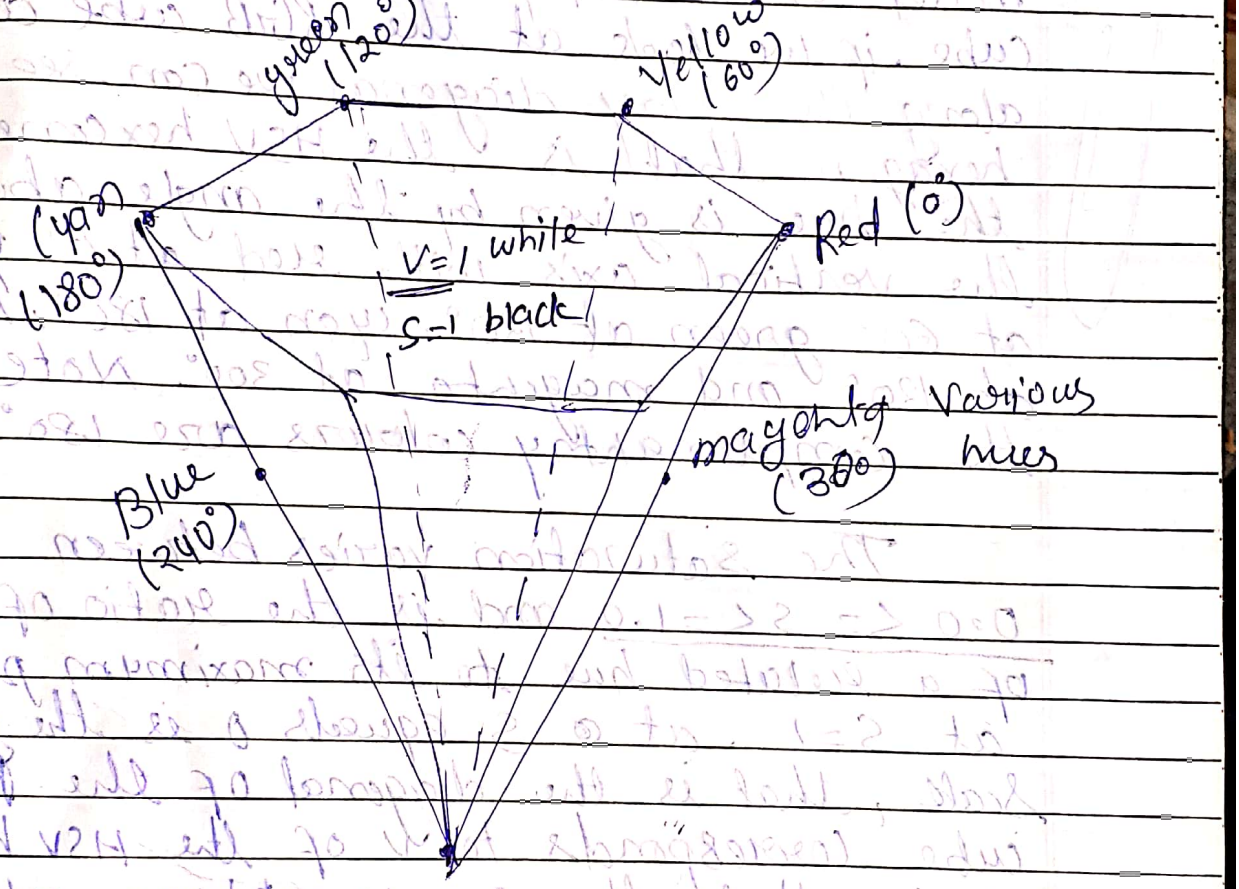
magenta ink ~~subs~~ subtract green component from incident light and yellow subtract the blue component. A unit cube representation of a CMY model is shown in fig.

here the point origin represents white where point (1, 1, 1) represents black since all the component of incident light are subtract. cube diagonal representing of cyan and magenta producing blue light while in similar subtractive process the other colour combinations are obtained

Printing process with CMY model produce collection of four, ink dots, coherences

RGB monitors produces collection of three ink do phosphorous dots, one dot is used for

HSV :—



The colour models representation is derived from the RGB colour model. View the cube here user specifies a colour and the amount of white and black to add to the colour to obtain different shades, tones and tints.

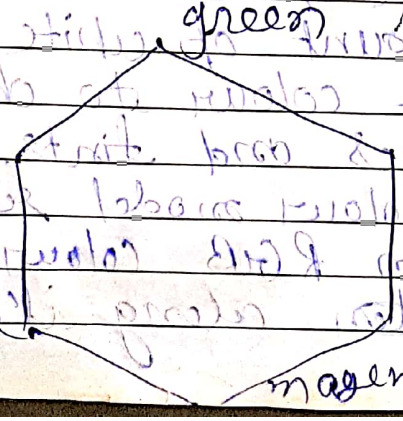
The colour model representation is derived from RGB colour model. View the cube often along it's sides, giving

7

us a shape of hexagon. This boundary represents the various hues and used as a top of HSV hexagon.

The 3D representation of the HSV model is derived from the RGB model cube. if we look at the RGB cube along the gray diagonal we can see a hexagon that is the HSV hexcone the hue is given by the angle about the vertical axis with red at 0° yellow at 60° green at 120°, cyan at 180°, blue at 120° and magenta at 300°. Note that the complementary colours are 180° apart.

The saturation varies between 0.0 <= S <= 1.0 and is the ratio of purity of a related hue to its maximum purity at S=1. at S equals 0 is the gray scale, that is the diagonal of the RGB cube corresponds to V of the HSV hexcone. Notice that the complementary colour (red + cyan, blue + yellow, green + magenta) are diagonally opposite.



So, to choose a colour we do the following:

Select pure hue (specifies) H and sets S = V = 1) to add black decrease V and or to add white decrease S.

For example:

- Pure green H = 120° S = V = 1
- dark green H = 120° S = 1 V = 0.40
- Light green H = 120° S = 0.3 V = 1.0