INDIAN INSTITUTE OF ENGINEERING SCIENCE AND TECHNOLOGY, SHIBPUR

classmate
Date
Page

B. TECH 5TH SEMESTER EXAMINATION, DECEMBER 2021

Computer Graphics [CS 3121]

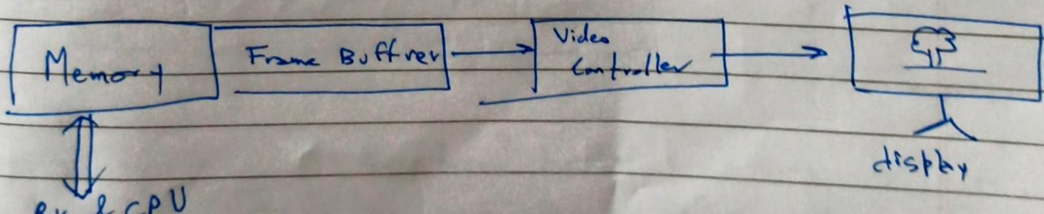
Name: Abhiroop Mukherjee
Examination Roll. No.: 510519109
G-Suite ID: 510519109.abhirup@students.iiests.ac.in
No. of sheets uploaded: 18

1, 2, 3, 4, 5, 6

Q1) a) i) Frame Buffer

→ Frame Buffer is a part of random access memory, containing data which is used to drive the ~~Video~~ Display Processor.

→ Frame Buffer ~~gone~~ contains data about the pixels that ~~will be~~ & is ~~supposed~~ suppoused to be shown by the display



ii) Resolution

→ resolution means the number of pixels that is shown by the display in a frame

→ it is generally written in the form of 'width × height' format

Eg 1920 × 1080 → 2073600 pixels displayed in a frame.
(1920 = width, 1080 = height)

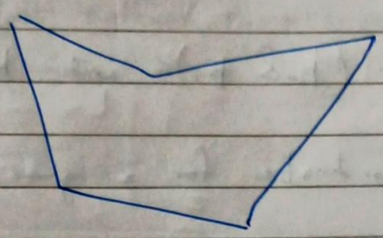
iii) Bit depth

→ The no. of bits used to define a pixel is called the bit depth.

Eg Grayscale → 8 bit depth
RGB → 24 (8×3) bit depth

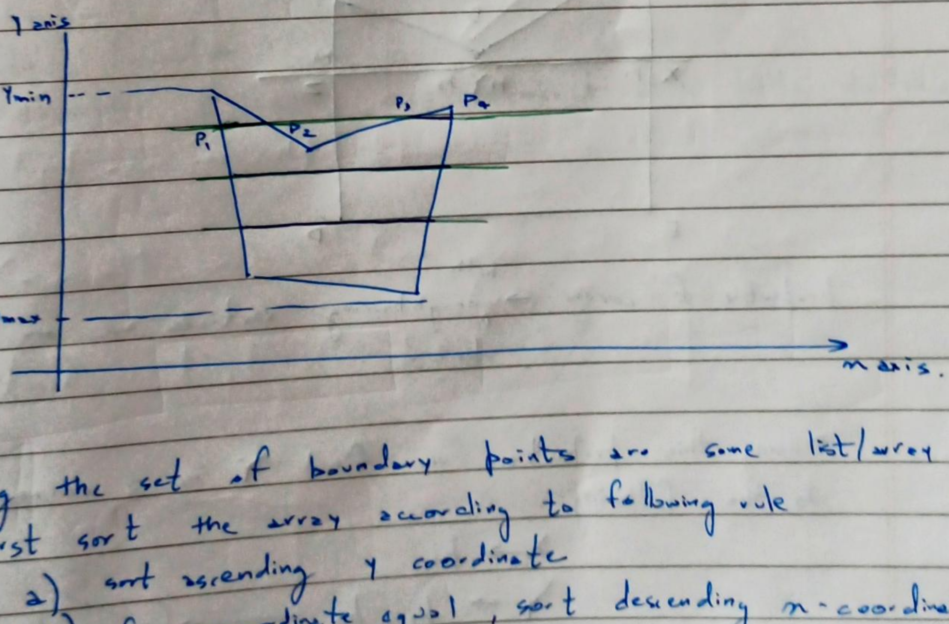
Q1) b) Scan line Polygon fill

→ consider the following shape [given only boundary points]



→ our objective is to fill this polygon using scanline polygon fill

→ To do this, we do the following



(i) assuming the set of boundary points are some list/array, we first sort the array according to following rule

a) sort ascending y coordinate

b) if y-coordinate equal, sort descending x-coordinate.

(ii) using sorted list find y_{max} & y_{min}

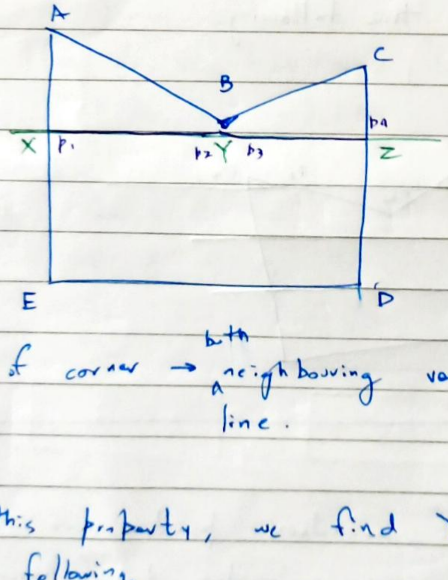
(iii) now loop y from y_{max} to y_{min}.

→ if no. of x-co-ordinate for a given y even, there is no corner, we fill between two points alternatively
Eg fill b/w (p_0 & p_1) and b/w (p_2 & p_3)

→ if no. of x-coordinate for a given y odd, there is corner (~~method~~

(iv) in the end, the polygon will be filled

Corner detection



property of corner → both neighbouring vertices are on one side of line.

→ using this property, we find Y is a corner point so we do the following

X → p1
Y → p2, p3 → (assing two points to corner)
Z → p4

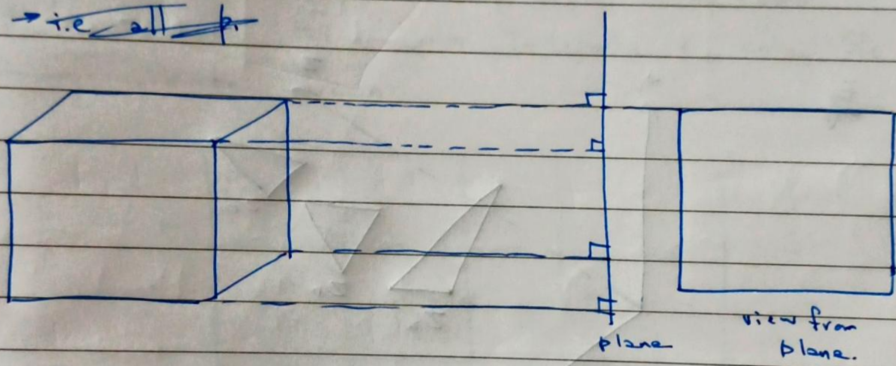
→ and then we draw alternatively i.e b/w (p_1 & p_2) & b/w (p_3 & p_4)

(A)

Q2) a) i) Orthogonal Projection.

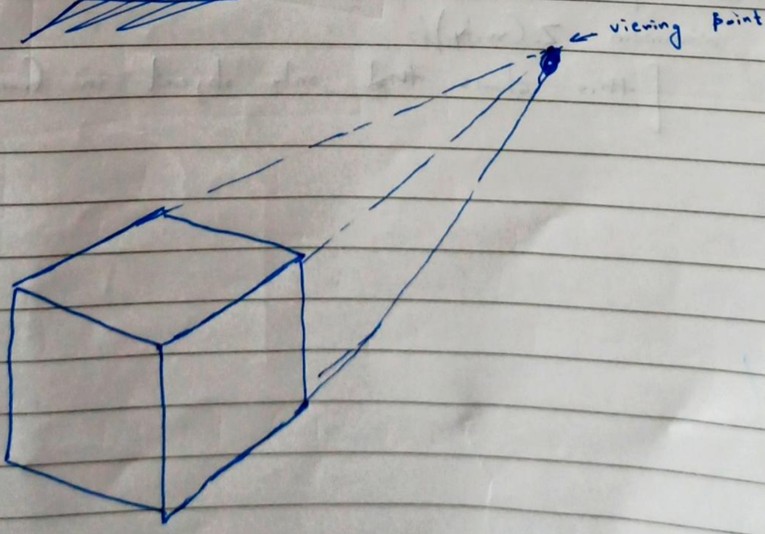
→ Orthographic Projection is a kind of parallel projection where the projecting lines emerge parallelly from the object surface and incident perpendicularly at projecting plane

→ ~~i.e all pr~~



ii) Perspective Projection

→ All projecting rays intersect at a finite distance.



Q2) b) Z-buffer Algorithm

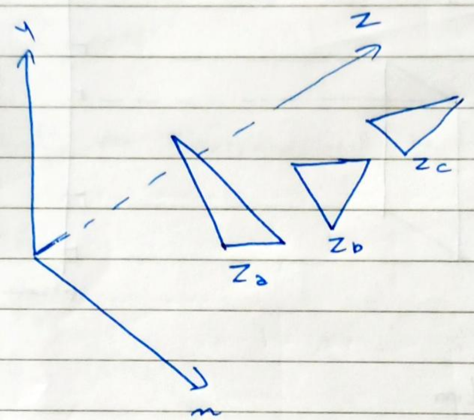
→ for every point that is to be drawn, we store two things about the point

i) The color Intensity → $f(x, y, z)$

ii) depth, i.e distance from the Viewing plane

↳ $Z(x, y) = z$

Eg



→ initially we define bit depth for every pixel to be drawn as ∞

→ and ~~for~~ for every pixel, we draw $f(x, y, z)$ which have minimum $Z(x, y)$

[this shows that only object in front is visible]

Eg

10			
9	9		
7	7	6	

Z_a.

12	10	11	
	10		

Z_b

15	16	16	16
	15	15	

Z_c

→ assume this is the ~~bt~~ depth map of the ~~figure~~ object A, B, C and each have their color Intensity, using Z_a, Z_b, Z_c following will be drawn

A	B	B	C
A	A	C	
A	A	A	

→ wher A → $f(x,y,z)$ of shape A
B → $f(x,y,z)$ of shape B
C → $f(x,y,z)$ of shape C

classmate

Date

Page

Q3) a) i) World Coordinate

→ World coordinate system is original coordinate system ~~which~~ in which the object in question was drawn

ii) A Window

→ A world-coordinate area selected for a display is called a window

→ It consists of a visual area containing some of the graphical user interface of ~~a~~ the program it belongs to and is framed by a window decoration

iii) View port

→ A view port is a polygon viewing region ~~in~~. It is expressed in rendering device specific co-ordinates.

→ Eg: pixels for a screen co-ordinate

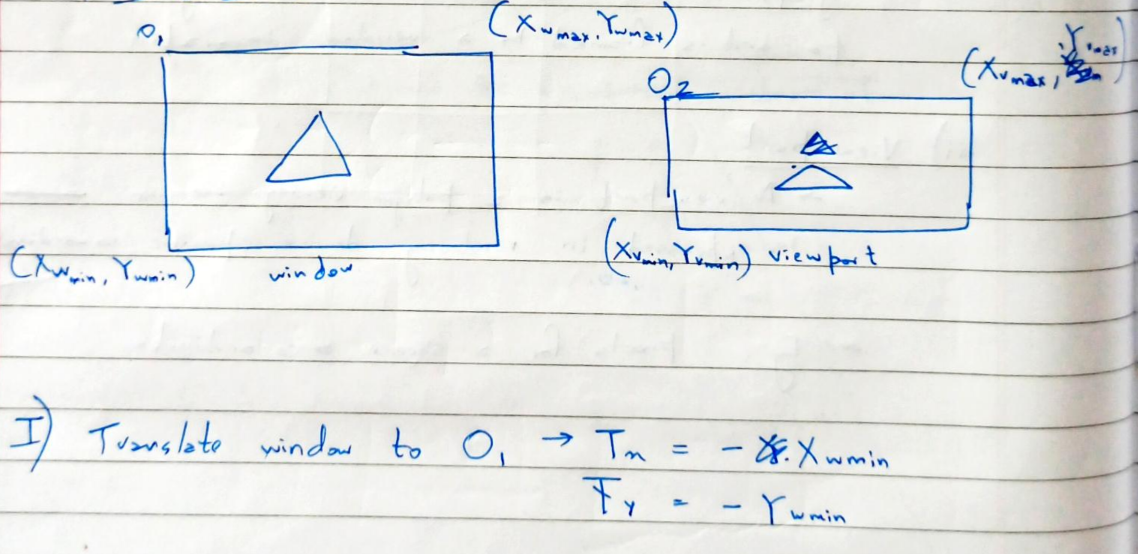
Q3) b) I) Window to ViewPort mapping by Normalization.

$$x_v = x_{v_{min}} + (x_w - x_{w_{min}}) \left[\frac{x_{v_{max}} - x_{v_{min}}}{x_{w_{max}} - x_{w_{min}}} \right]$$

$$Y_v = y_{v_{min}} + (y_w - y_{w_{min}}) \left[\frac{y_{v_{max}} - Y_{v_{min}}}{Y_{w_{max}} - Y_{w_{min}}} \right]$$

II) Matrix method

i) ~~Translate wi~~



I) Translate window to O_1 $\rightarrow$ $T_x = -X_{wmin}$

$T_y = -Y_{wmin}$

$$\text{transformation matrix} = \begin{bmatrix} 1 & 0 & -X_{wmin} \\ 0 & 1 & -Y_{wmin} \\ 0 & 0 & 1 \end{bmatrix} = T$$

ii) scale window

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

where
$$S_x = \frac{x_{vmax} - \cancel{x_{wmin}}\, x_{vmin}}{x_{wmax} - x_{wmin}}$$

$$S_y = \frac{y_{vmax} - y_{vmin}}{Y_{wmax} - y_{wmin}}$$

iii) Translate to viewport

$T_x = X_{vmin}$

$T_y = Y_{vmin}$

$$\text{Transformation matrix} = \begin{bmatrix} 1 & 0 & x_{vmin} \\ 0 & 1 & Y_{vmin} \\ 0 & 0 & 1 \end{bmatrix} = T'$$

So full transformation $\rightarrow$ $P_v = TST'P_w$

$P_v = T S T' P_w$

$$= \begin{bmatrix} 1 & 0 & -X_{wmin} \\ 0 & 1 & -Y_{wmin} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & x_{vmin} \\ 0 & 1 & y_{vmin} \\ 0 & 0 & 1 \end{bmatrix} P_w$$

$$= \begin{bmatrix} S_x & 0 & -X_{wmin} \\ 0 & S_y & -Y_{wmin} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & x_{vmin} \\ 0 & 1 & y_{vmin} \\ 0 & 0 & 1 \end{bmatrix} P_w$$

$$\begin{bmatrix} x_v \\ y_v \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & 0 & S_x x_{vmin} - X_{wmin} \\ 0 & S_y & S_y Y_{vmin} - Y_{wmin} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ 1 \end{bmatrix}$$

$\therefore \; x_v = x_{vmin} + (x_w - x_{wmin}) S_x$

$y_v = y_{vmin} + (y_w - y_{wmin}) S_y$

$\therefore$ hence both are equal

classmate

Date

Page

Q2) a) Bezier Curve

→ from De Casteljau's Recursive algorithm:

(i) Two Points formula.

$$P = (1-t)\,P_i + t\,P_{i+1} \qquad \left(P = \begin{bmatrix} x \\ y \end{bmatrix} \right)$$

ii) Three Points formula

$$P = (1-t)\,P_k + t\,P_{k+1}$$

where $P_k = (1-t)\,P_i + t\,P_{i+1}$

$P_{k+1} = (1-t)\,P_{i+1} + t\,P_{i+2}$

$$\therefore P = (1-t)\left[(1-t)\,P_i + t\,P_{i+1}\right] + t\left[(1-t)\,P_{i+1} + t\,P_{i+2}\right]$$

$$= (1-t)^2\,P_i + 2t(1-t)\,P_{i+1} + t^2\,P_{i+2}$$

similarly

iii) four Points formula.

$$P = (1-t)^3\,P_i + 3(1-t)^2 t\,P_{i+1} + 3(1-t)\,t^2\,P_{i+2} + t^3\,P_{i+3}$$

→ This is very similar to binomial expansion, so we can write

$$P = \sum_{i=0}^{n} P_i\,B_i^{\,n-1}(t)$$

where $B_i^{\,n}(t) = {}^nC_i\,(1-t)^{n-i}\,t^i$

is called Bezier basis function.

$n \rightarrow$ no. of control points / degree or curve

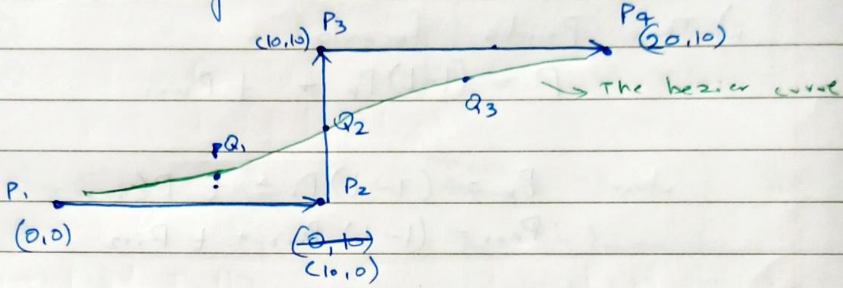
Q2) b) quartic Bezier curve $\rightarrow n = 3$

$\therefore$ for t in range 0 to 1 with incremen d : (d is small)

$P = (1-t)^3 P_i + 3(1-t)^2 t \; P_{i+1} + 3(1-t)t^2 P_{i+2} + t^3 P_{i+3}$

draw(P)

So Eg: consider following control points



$\Rightarrow$ so ~~for~~ with small increments of t, we will get points that will be plotted.

Eg $t = 0.2$ $[1-t = 0.98]$

$P = 0.941192 \begin{bmatrix} 0 \\ 0 \end{bmatrix} + 0.057624 \begin{bmatrix} 10 \\ 0 \end{bmatrix} + 1.176\times10^{-3} \begin{bmatrix} 10 \\ 10 \end{bmatrix} + 8\times10^{-6} \begin{bmatrix} 20 \\ 10 \end{bmatrix}$

$= \begin{bmatrix} 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 5.76 \\ 0 \end{bmatrix} + \begin{bmatrix} 0.01176 \\ 0.01176 \end{bmatrix} + \begin{bmatrix} 0.00016 \\ 0.00008 \end{bmatrix}$

$= \begin{bmatrix} \cancel{0.06}792 \\ \cancel{0}.01182 \end{bmatrix} = \begin{bmatrix} 6.792 \\ 1.182 \end{bmatrix}$

draw (6, 1) Q_1

classmate

Date
Page

Eg $t = 0.5$

$$P = 0.125 \begin{bmatrix} 0 \\ 0 \end{bmatrix} + 0.375 \begin{bmatrix} 10 \\ 0 \end{bmatrix} + 0.375 \begin{bmatrix} 10 \\ 10 \end{bmatrix} + 0.125 \begin{bmatrix} 20 \\ 10 \end{bmatrix}$$

$$= \begin{bmatrix} 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 3.75 \\ 0 \end{bmatrix} + \begin{bmatrix} 3.75 \\ 3.75 \end{bmatrix} + \begin{bmatrix} 2.5 \\ 1.25 \end{bmatrix}$$

$$= \begin{bmatrix} 10 \\ 5 \end{bmatrix}$$

Eg $t = 0.75$

$$\cancel{P = 0.42 \begin{bmatrix} 0 \\ 0 \end{bmatrix}}$$

$$P = 0.015 \begin{bmatrix} 0 \\ 0 \end{bmatrix} + 0.14 \begin{bmatrix} 10 \\ 0 \end{bmatrix} + 0.421 \begin{bmatrix} 10 \\ 10 \end{bmatrix} + 0.423 \begin{bmatrix} 20 \\ 10 \end{bmatrix}$$

$$= \begin{bmatrix} 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 1.4 \\ 0 \end{bmatrix} + \begin{bmatrix} 4.21 \\ 4.21 \end{bmatrix} + \begin{bmatrix} 8.46 \\ 4.23 \end{bmatrix}$$

$$= \begin{bmatrix} 14.07 \\ 8.44 \end{bmatrix} \rightarrow \text{draw } (14, 8)$$

Q6) 2) Illumination model

→ when we shine light ~~int~~ into an object, the object might get illuminated by the following ways.

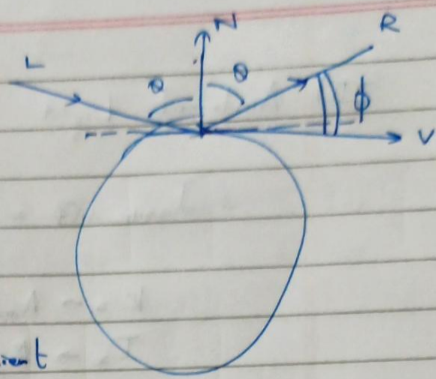
i) Object Reflects light
ii) Object Diffuses light
iii) Ambient light

~~Re~~

Reflected light

- from Light Emitting Source
  - Specular Reflection → I_{spec}
  - Diffuse Reflection → I_{diff}
- from Light Reflection Surfaces → I_{diff_amb}

(I) Specular Reflection



$$I_{spec} = K_S I_L \cos^{n_s} \phi$$

where $K_S \rightarrow$ specular reflection coefficient

$n_S \rightarrow$ specular reflection parameter

$I_L \rightarrow$ Intensity of ~~reflected~~ incident light

$L \rightarrow$ incident ray vector

$N \rightarrow$ surface normal

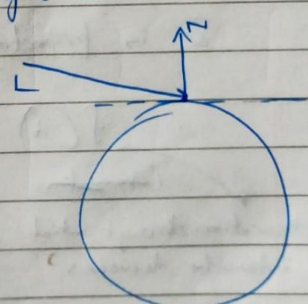
$R \rightarrow$ reflected ray vector.

$V \rightarrow$ vector in the direction of viewer.

$\theta \rightarrow$ angle of incident/ reflection

$\phi \rightarrow$ ~~and~~ angle between reflected light & viewer.

(II) diffuse reflection



$$I_{diff} = K_d I_L \cos \theta$$

$K_d \rightarrow$ diffuse reflection coefficient
$(0 \leq K_d \leq 1)$

$I_L \rightarrow$ intensity of incident light

$L \rightarrow$ incident ray vector.

$N \rightarrow$ normal

$\theta \rightarrow$ angle of incidence

III) Ambient Light

$$I_{amb-diff} = K_a I_a$$

$K_a \rightarrow$ Ambient Reflection coeffient $(0 \leq k_a \leq 1)$

$I_a \rightarrow$ ~~A~~ Intensity of ambient light

$\therefore$ $I_{reflection} = I_{spec} + I_{diff} + I_{amb-diff}$

$= K_s I_L \cos^{n_s} \phi + K_d I_L \cos \theta + K_a I_a$.

Intensity Attenuation

→ we know that as light travels, it's intensity decreses & is proportional to $1/d^2$

$$\therefore f(d) = \frac{1}{a_0 + a_1 d + a_2 d^2}$$

(~~Attenuation~~

(a_1, a_2, a_3 constant)

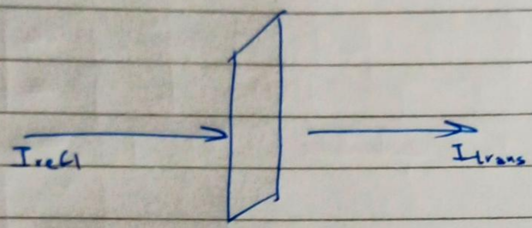
$d \rightarrow$ dist travelled by light

the ration through which intensity decreses

$$\therefore I_{refl} = f(d) \left[I_{spec} + I_{diff} \right] + I_{amb-diff}$$

Transparency

~~I~~ → as light goes through a transparent medium, intensity of output light is as follows



$I = (1-k_t) I_{refl} + k_t I_{trans}$

final intensity of light

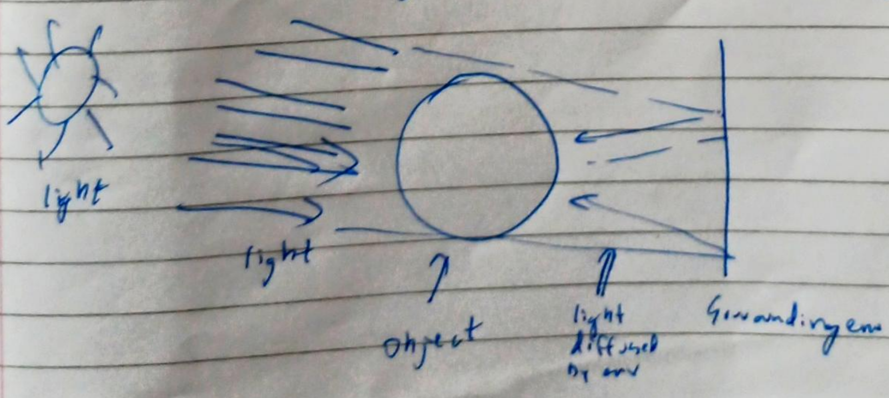
k_t → Transparency coefficient $(0 \le k_t \le 1)$

$1-k_t$ → opacity factor.

Q6) b) Ambient Light

→ As we shine light in to object, not all lights are shined on object, so light goes to surronding enviornment & gets reflected / diffused

→ Those light can also ~~get~~ shine the object and hence those lights are called ambient lights



18