13/1/24

Graphics

Hidden Surface Removal

Two general approaches

- Image precision (Image এর pixel by pixel চেক কাজ করা।)
- Object precision (একটা object ধরে ধরে চিন্তা করা। Pixel হিসাবে হয় না।)

offline ও precision. image থেকে object থেকে pixel colour ~~করছি~~

Pixel এর সাথে সম্পর্ক নাই।

Reason for invisibility

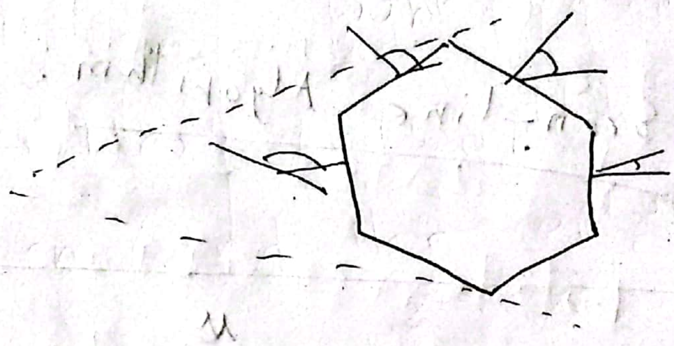
- Outside FOV — Efficiency and correctness
- Polygon is backfacing — correctness

- Occluded by other objects — correctness

→ Object precision এর comparison কম লাগে generally but এই comparison গুলো itself অনেক complex.

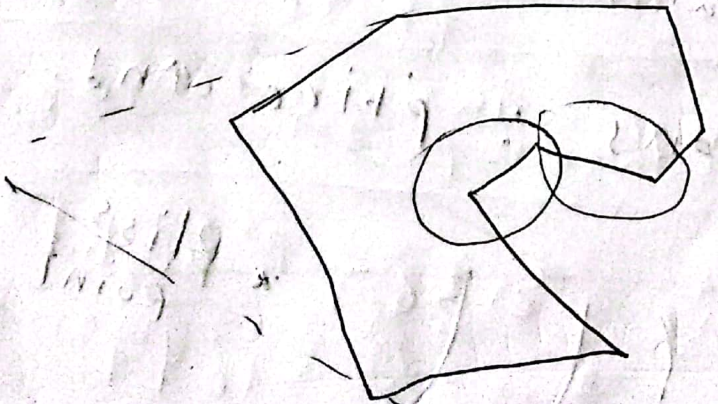
Back-face culling : → বাদ দেওয়া
→ conservative, extra দিতে পারে। কিন্তু কম দিবে না,

→ Surface normal . viewing vector
এইটা greater than 0 হইলে
বাদ।
[90° হলে রাখা/বাদ দেওয়া যায়]



Problem :

•



• Occlusion দূর করতে পারে না,

Painter's Algorithm:

→ Sorting এর প্রভৃতি সমূহের একটা use
করা হয়েছে না।

Z-buffer Algorithm:

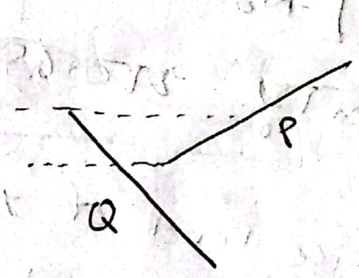
Scan-line Algorithm:

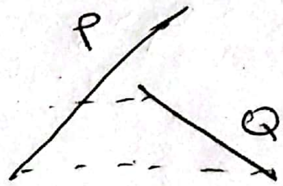
Graphics

List-Priority Algorithms

~~Depth~~ Object precision to sort, then image precision

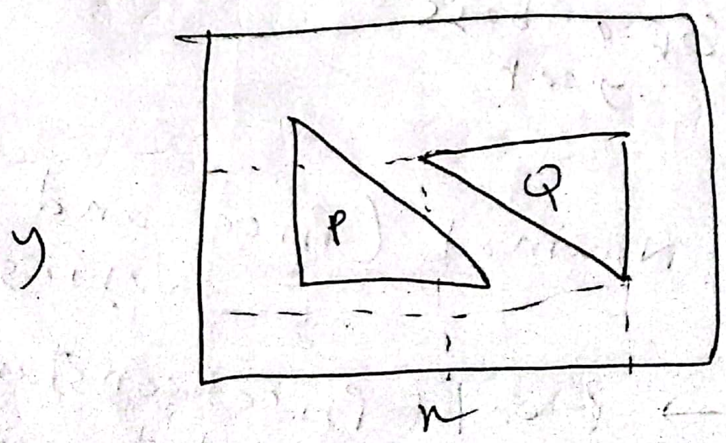
Depth-Sort Algorithm





P, Q এর মুছে দিবেন, But Q মুছে আসবেন না।

P মুছে দিবেন না। Q মুছে আসবেন।

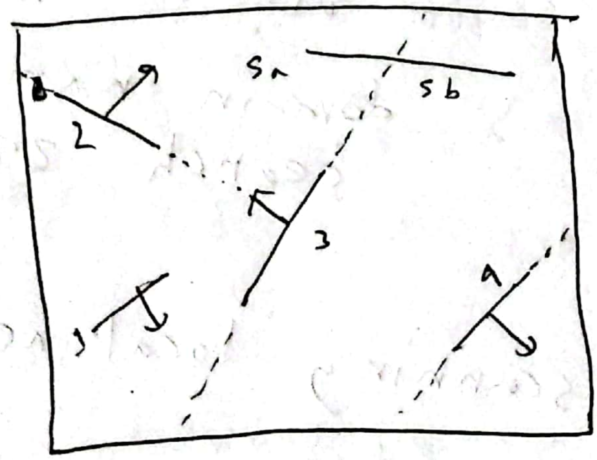


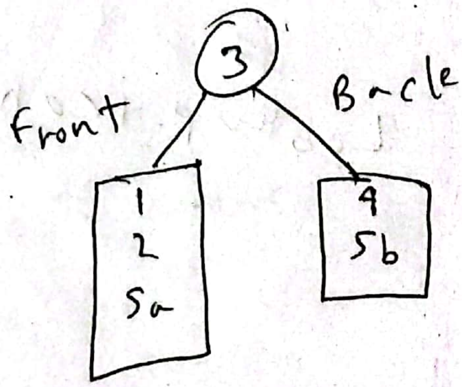
→ Projections on the screen are disjoint

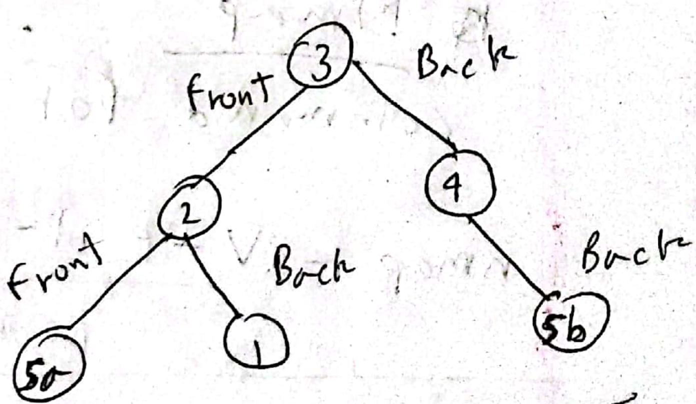
After 5 tests fails, we will test 3 and 4 again reversing the role of P and Q [1, 2 and 5 will still fail, so no need to check]

BSP Tree

BSP → Binary Space Partitioning







• In general BSP tree তে splitting হলে,

Graphics

Scan converting Lines

slope < 1 হলে, x একটা একটা করে বাড়বে

y choose করব,

slope > 1 হলে, y " " "

x choose " "

$$y = mx + c$$

when, $m < 1$

$$y_i = mx_i + c$$

$$y_{i+1} = m\,x_{i+1} + c$$

$$\therefore y_{i+1} = m(x_i + 1) + c$$

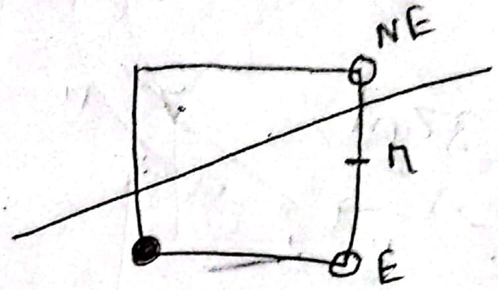
$$= mx_i + m + c$$

$$y_{i+1} = y_i + m$$

$\cancel{x_{i+1},\ \text{round}\ (y_i + m)}$

$$(x_{i+1}, y_{i+1}) = (x_i + 1, \text{round}(y_i + m))$$

Midpoint Line Algorithm:



$P = (x_p, y_p)$

$$y = \frac{dy\,x}{dx} + C$$

$$dx\,y = dy \cdot x + C \cdot dx$$

$\therefore$

$$F(x, y) = ax + by + c$$

$\therefore \; a = dy, \quad b = -dx, \quad c = dx \cdot B$

$$d = f(M) = f(x_p + 1, y_p + \tfrac{1}{2})$$

$dyx - dxy + dx \cdot B = 0$

$\leftarrow x_p+1+1$ হওয়াতে extra dy

আগেরটা x_p+1 এর জন্য

if E chosen,

$$d_{new} = f(x_p+1+1, y_p+\tfrac{1}{2}) \quad \Big| \quad d = f(M)$$

$$= f(x_p+1, y_p+\tfrac{1}{2}) + dy \cdot 1$$

$$= d_{old} + dy$$

if NE chosen,

$$d_{new} = d_{old} + dy - dx$$

$$d_{start} = f(x_0+1, y_0+\tfrac{1}{2})$$

$$= f(x_0, y_0) + dy - \frac{dx}{2}$$

$$= dy - \frac{dx}{2}$$

2 দিয়ে গুণ
দিয়ে দিব, তাহলে
সব calculation
এ 2 আসবে,

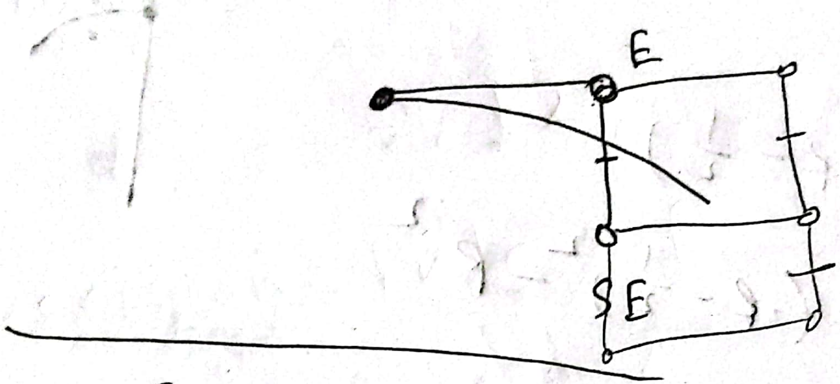
$2ax + 2by + 2c = 0$

m>1 হলে
y এর জন্য
same algo
চালানো যায়,
অথবা
x=y এর
সাপেক্ষে
reflection
নেওয়া যায়
so, points
উল্টায় যাবে

Graphics

20/1/24

Midpoint circle algorithm.



$M(x_p+1, y_p-\frac{1}{2})$

$f(x,y) = x^2+y^2-R^2$

$d = f(M)$

$d=0$, on the circle any of E, SE

$d>0$, outside the circle, SE

$d<0$, inside the circle, E

if E chosen,

$M_E(x_p+2, y_p-\frac{1}{2})$

$d_{new} = f(M_E)$

$= (x_p+2)^2 + (y_p-\frac{1}{2})^2 - R^2$

$= (x_p+1)^2 + 2(x_p+1) + 1 + (y_p-\frac{1}{2})^2 - R^2$

$= d_{old} + (2x_p+3)$

if ~~NE~~ SE chosen,

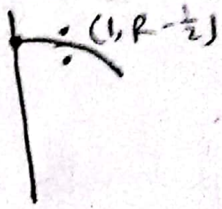
$n_{SE}(x_p+2,\ y-\frac{1}{2}-1)$

$$d_{new} = (x_p+2)^2 + (y-\tfrac{1}{2}-1)^2 - R^2$$

$$= (x_p+1)^2 + (2x_p+3) + (y-\tfrac{1}{2})^2 - 2(y-\tfrac{1}{2}) + 1 - R^2$$

$$= (x_p+1)^2 + (y-\tfrac{1}{2})^2 - R^2 + (2x_p+3) - 2y_p + 2$$

$$= d_{old} + 2x_p - 2y_p + 5$$



$$d_{start} = f(1, R-\tfrac{1}{2})$$

$$= 1^2 + (R-\tfrac{1}{2})^2 - R^2$$

$$= \frac{5}{4} - R$$

Say, $h = d - \frac{1}{4}$

$\therefore d = h + \frac{1}{4}$

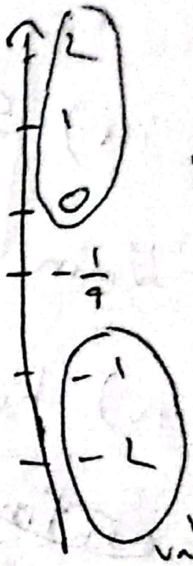
$$h + \frac{1}{4} = \frac{5}{4} - R$$

$\therefore h = 1 - R$

$h = -\frac{1}{4} \rightarrow$ ON circle

$h > -\frac{1}{4} \rightarrow$ outside circle

$h < -\frac{1}{4} \rightarrow$ inside circle



So, 0 নিয়ে compare করলেই হয়। As all values will be integer

more optimization :

$\Delta E = 2x_p + 3$

$\Delta SE = 2x_p - 2y_p + 5$

$d_{new} = d_{old} + \Delta E_{old} / \Delta SE_{old}$

$\Delta E_{new} = \Delta E_{old} / \Delta SE_{old} + \text{constant}$

$\Delta SE_{new} = \Delta SE_{old} / \Delta E_{old} + \text{Constant}$

Polygon scan conversion:

i) fraction হলে ভেতরেরটার দিকে round করবে।

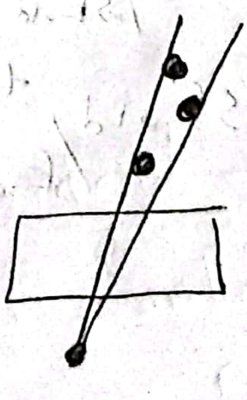
ii) Integer intersection হলে
left গুলো and bottom গুলো priority.

iii) Shared vertex হলে, যেগুলো
toggle হবে, বাকিগুলো না।
ymin হলে

iv) horizontal line এ ও (iii)
~~continue~~ হবে। কিন্তু horizontal line
এর বেলায় ymin/ymax নাই।

Problem:

Sliver →



এই জায়গায় হবে না, আবার অনেক জায়গা না, gap হতে পারে, এটার জন্য anti-aliasing.

Lighting and Shading

Illumination
- Local illumination
- Global illumination

Depends on → Light sources
~~Surface~~ properties
Surface

Light source
- Ambient Light
- Diffuse Light
- Spot Light

Ambient Light:

World / scene এ light source নাই, But light এর effect আছে, light reflect হয়ে

→ Global (সূর্য থেকে reflect হয়ে আসছে)

→ Local (যেহেতু একটি light আছে, ছড়িয়ে, বিভিন্ন surface এ reflect হয়ে exist করছে system এ)

System এ নাই

System এ আছে

Diffuse Light:

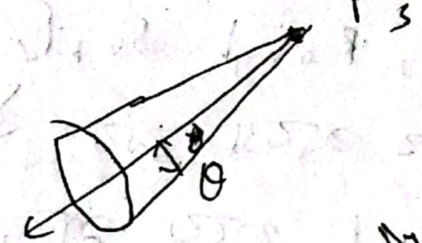
→ Point Source
(সব দিকে equally emit করে)

$$i(p, p_0) = \frac{1}{|p - p_0|^2} I(p_0)$$

→ Directional source
Source অনেক দূরে। তাই ray গুলো Parallel.

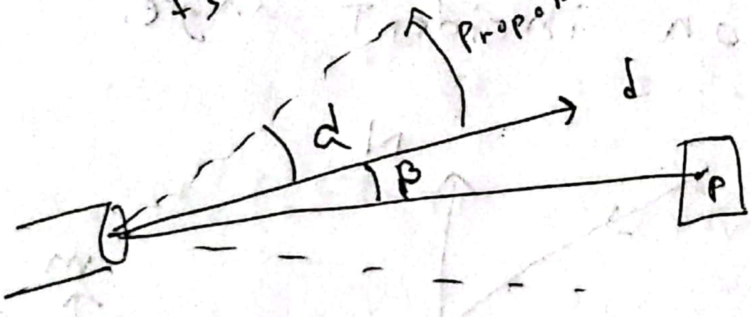
Intensity depends on angle between surface normal and light.

Spot light:



Proportionally change হয়। Then α হবে ধরে direct 0.

I_s



empirical constant

$$P = I\cos^{\varepsilon}(\beta)$$

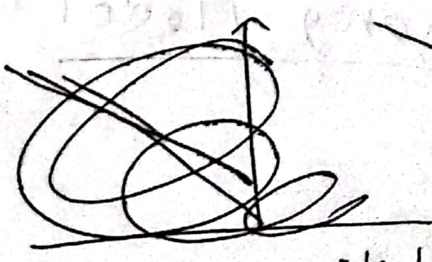
Surface

- → Specular (Ideal mirror)
- → Diffuse (No shiny part.)
- → Translucent (Transparent)

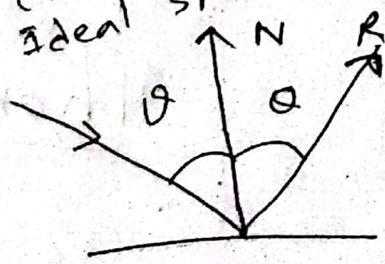
Reflection

- → Specular

যে দিকে আলো ফেলা প্রতিফলন এর উপরে depend করবে।



(fully on the plane) Ideal specular

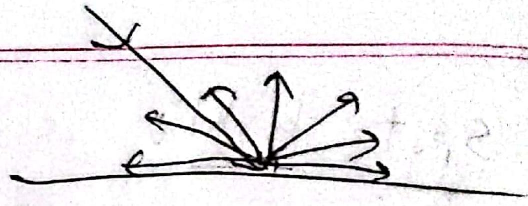


Non-ideal (Not fully on the plane)

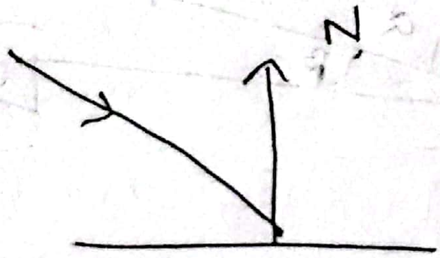
উল্টা দিক দিয়ে যেদিকে যাব না।

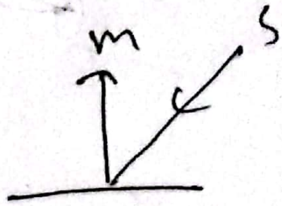
R এর আশেপাশে।

↳ Diffused



সবদিকে সমানভাবে ছড়ায়, Rough Surfaces.
দর্শকের অবস্থানের উপর depend করে না, Depends on incident angle.



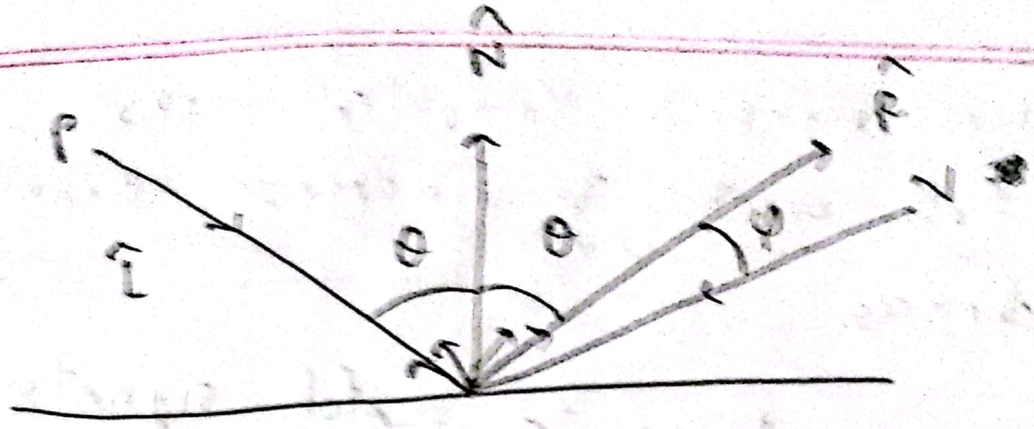


$$I_d = I_s \, \rho_d \left(\frac{\vec{s} \cdot \vec{m}}{|\vec{s}||\vec{m}|} \right)$$

Source intensity

surface dependent constant.

Phong Model



Diffuse

$I_d \propto \cos\theta$

$= I_p k_d \cos\theta$

$= I_p k_d (\hat{L} \cdot \hat{N})$

Specular Reflection

$I_s \propto (\cos\varphi)^k$

$= I_p k_s (\cos\varphi)^k$

$= I_p k_s (\hat{R} \cdot \hat{V})^k$

k_d and k_s surface property.

যেগুলো diffuse এবং যেগুলো specular

Ambient component $= I_a k_a$

$I = I_a k_a + I_p [k_d \max\{(\hat{L} \cdot \hat{N}), 0\} + k_s \max\{(\hat{R} \cdot \hat{V})^k, 0\}]$

~~When a source~~

Graphics

Color of a surface হইতেছে আসল light এ যে colour। সেটা হয়।

- Colour of a surface 30% red, 95% green and 25% blue

$$K_{ar} = k_{dr} = 0.3k$$
$$K_{ag} = k_{dg} = 0.95k$$
$$K_{ab} = k_{db} = 0.25k$$

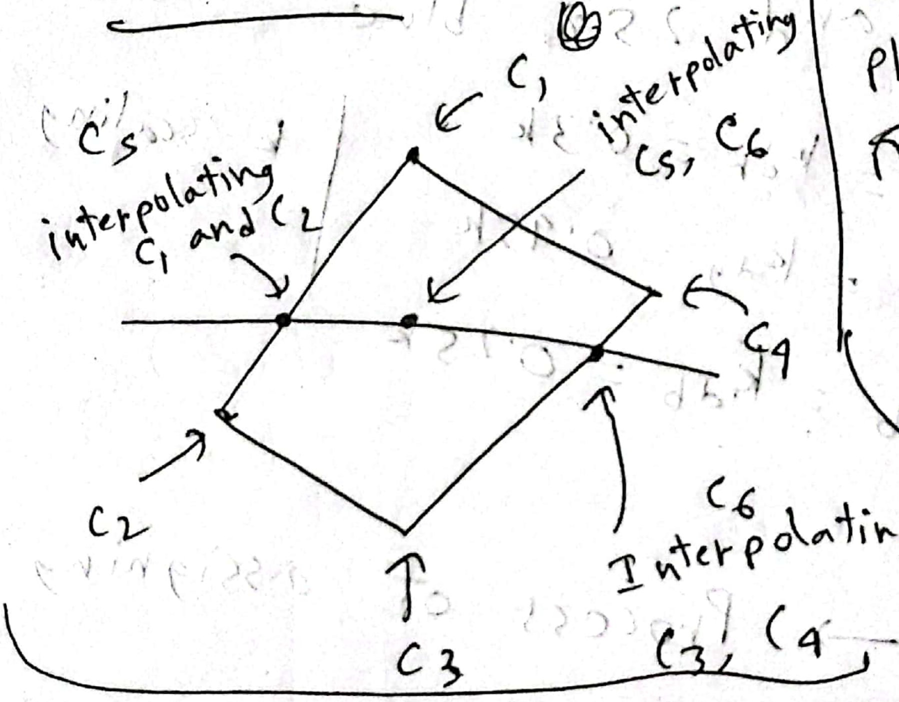
k scaling factor

Shading → Process of assigning colors to

Shading
- Flat (Polygon এ সব same colour)
- Smooth
  - Gouraud
  - Phong (Shading, not model)

Curve surface এর ক্ষেত্রে polygon whole surface কে represent করে না। অনেক ছোট ছোট Polygon দিয়ে তৈরি।

Gourand



Flat এ, কোনো একটা Point এ Phong model দিয়ে colour বের করে Full polygon এ same colour.

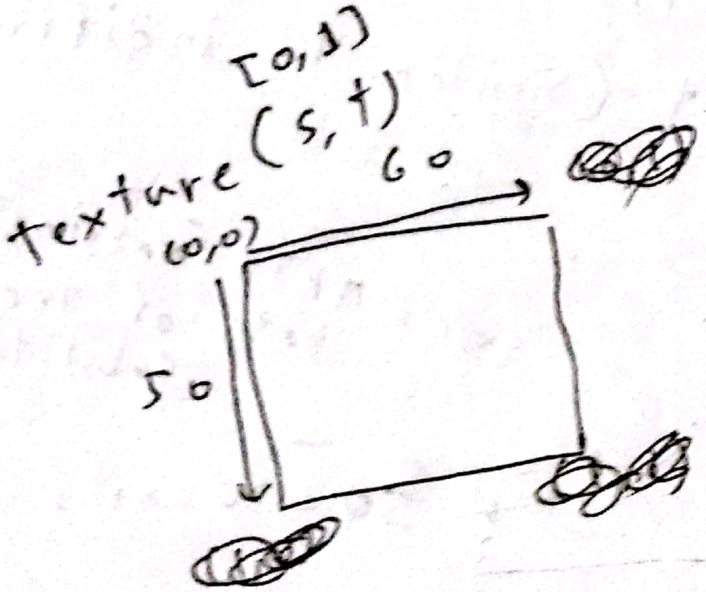
4 vertex এ colour বের করবো। আগে normal বের করে করবো ওই 4 vertex এ।

Phong:

~~...~~ 4 vertex এ নরমাল বের করবো, করবো। Then সব ~~...~~ pixel এ নরমাল

interpolate করবো। Then এই normal use করবো phong model দিয়ে colour at all pixels.

Texture
- → Bitmap
- → Procedural



27/1/24

Graphics

Texture:

- → Bitmap Texture
- → Procedural Texture

Shadow

Using texture / shadow buffer.

Texture → shape এবং position of shadow calculate করা অনেক tough.

Shadow buffer →

সুবিধা → Stationary objects run-time এ থাকলে এর অনেক calculation কমে। shadow buffer pre-process করে রাখা যায়.

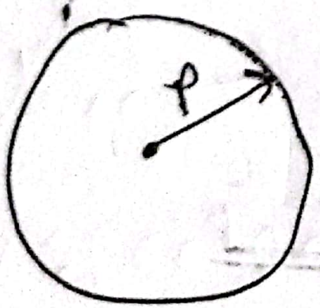
অসুবিধা → Memory intensive. প্রতি source এর জন্য একটা buffer.

Ray Casting and Ray Tracing

implicit → Parameter change করে direct point যাবে না।

explicit → যাবে

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

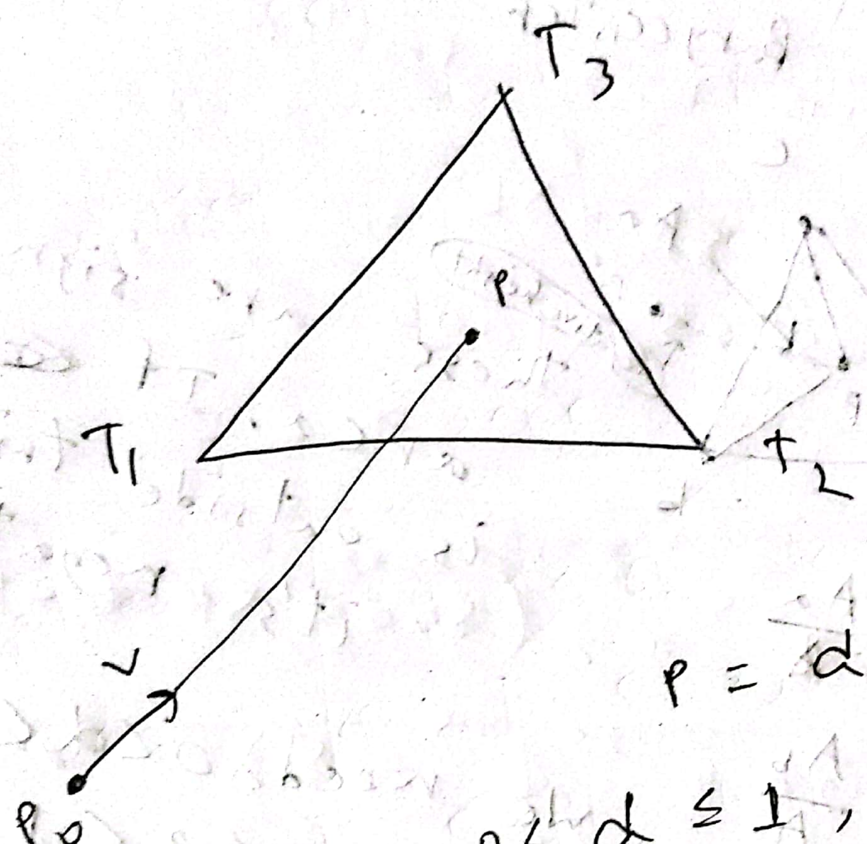


$$\vec{P}\cdot\vec{P}-r^2=0$$

...ware

lock করে দিয়েছে। (through ransomware)

Graphics



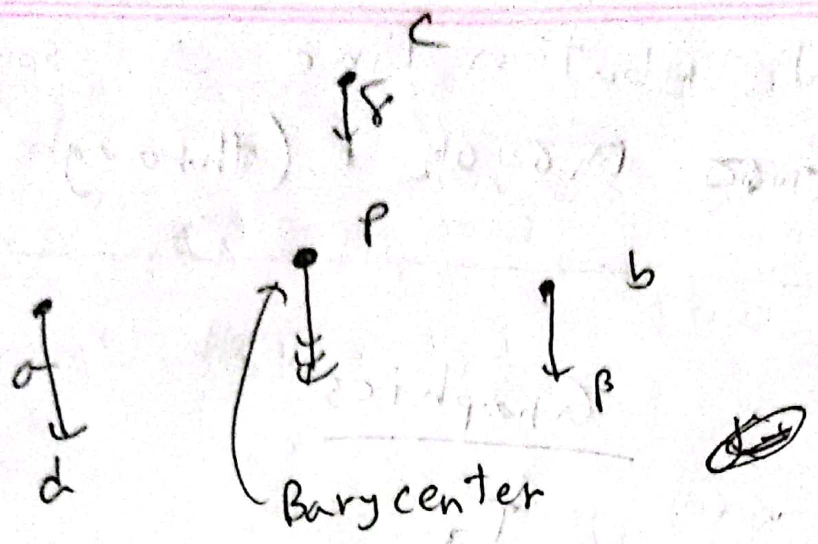
$$P = \alpha (T_2 - T_1) + \beta (T_3 - T_1)$$

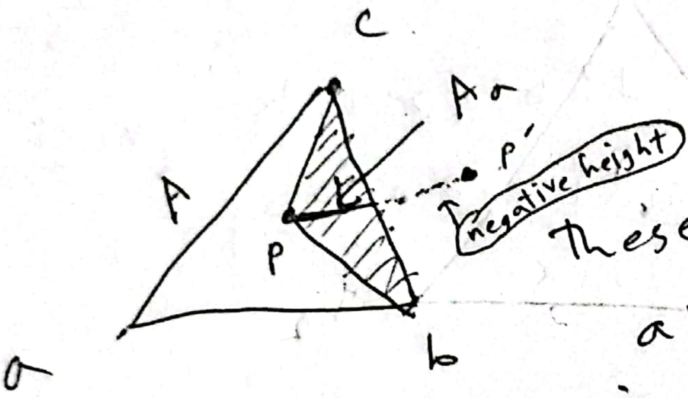
$$0 \le \alpha \le 1, \quad 0 \le \beta \le 1, \quad \alpha + \beta \le 1$$

$$P(\alpha, \beta, \gamma) = \alpha a + \beta b + \gamma c$$

with $\alpha + \beta + \gamma = 1$

এটা always true হতে হবে। For any plane





These are signed area. If ~~...~~ p is outside triangle, it's negative.

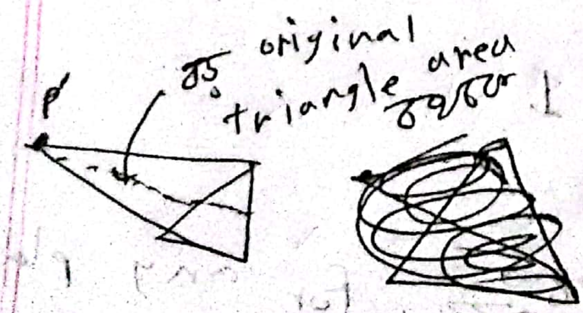
$$\alpha = \frac{A_a}{A}$$

$$\beta = \frac{A_b}{A}$$

$$\gamma = \frac{A_c}{A}$$

We need, $0 < \alpha < 1$,
$0 < \beta < 1$,
$0 < \gamma < 1$.

[Cause বাইরে p থাকলে, α, β, γ negative / greater than 1 হবে]



$\alpha + \beta + \gamma = 1 \longrightarrow \alpha = 1 - \beta - \gamma$

$\therefore P(\alpha, \beta, \gamma) = \alpha a + \beta b + \gamma c$

$\therefore P(\beta, \gamma) = (1 - \beta - \gamma) a + \beta b + \gamma c$

$= a + \beta (b - a) + \gamma (c - a)$

$P(t) = P(\beta, \gamma)$

$R_0 + t * R_d = a + \beta (b - a) + \gamma (c - a)$

(Solve separating x, y and z.)