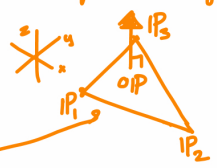
From continuous model to (triangle) mesh

① FLAT-SHADING

→ determine ONE color per triangle



$$P = \frac{1}{3}\sum_{1}^{3} P_i$$

$$n = \frac{(P_2 - P_1) \times (P_3 - P_1)}{\|(P_2 - P_1) \times (P_3 - P_1)\|}$$

⇒ II (p):

⇒ Two ways to define normal n:

BETTER.

Ⅰ $$n = \frac{\frac{1}{k}\sum_{i=1}^{k} n_i}{\left\|\frac{1}{k}\sum_{i=1}^{n} n_i\right\|}$$

Takes into account relative size of triangle

Ⅱ $$n = \frac{\frac{1}{k}\sum_{i=1}^{k} n_i^{norm}}{\left\|\frac{1}{k}\sum_{i=1}^{n} n_i^{norm}\right\|}$$

NOTE: "Outward normals" / "up-down" / "orientation"

Implication for Δ-culation:

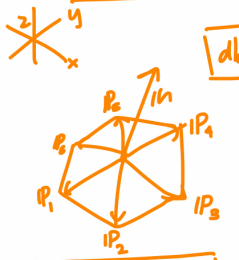
② GOURAUD-SHADING

→ "Compute Colors @ vertices of triangle & 'mix' them"

FIRST: NORMALS required @ vertices of each triangle !!!

→ Algo: Use PLATELE of a mesh vertex to "estimate" a normal

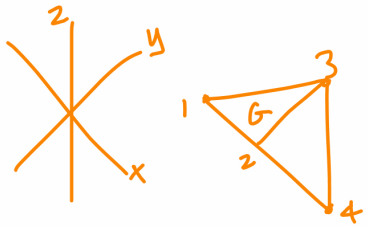
PLATELET



$$d_i = P_i - P$$

$$n_i = d_i \times d_{i+1} \quad \text{Ⅰ}$$

$$n_i^{norm} = \frac{d_i \times d_{i+1}}{\|d_i \times d_{i+1}\|} \quad \text{Ⅱ}$$



Geometry

4

x1 y1 z1

x4 y4 z4

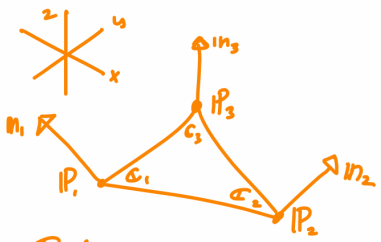
Topology

2

1 2 3

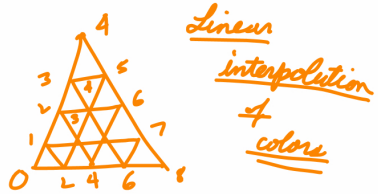
3 2 4

Having normal estimates @ each vertex in △ mesh can compute color value for all mesh vertices!



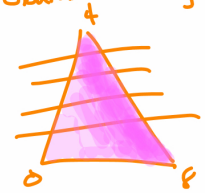
$C = (r_i, g_i, b_i)$

Desired Effect?

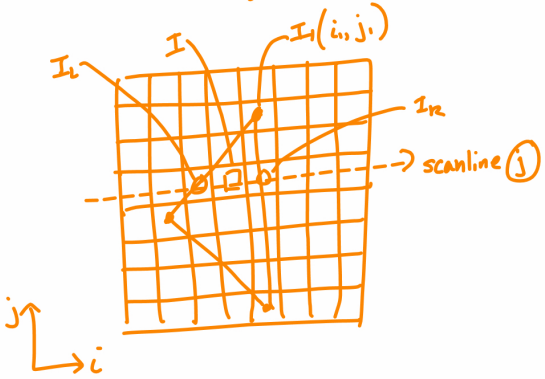


Linear interpolation of colors

Gouraud shading



Gouraud-shading performed as part of RASTERIZATION



$$I_L = \frac{j - j_2}{j_1 - j_2} I_1 + \frac{j_1 - j}{j_1 - j_2} I_2$$

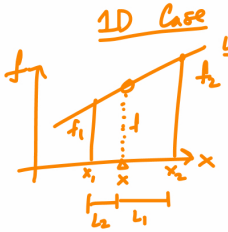
$$I_R = \frac{j - j_3}{j_1 - j_3} I_1 + \frac{j_1 - j}{j_1 - j_3} I_3$$

$$I = \frac{i_R - i}{i_R - i_L} I_L + \frac{i - i_L}{i_R - i_L} I_R$$

Note: Linear interpolation

1D Case

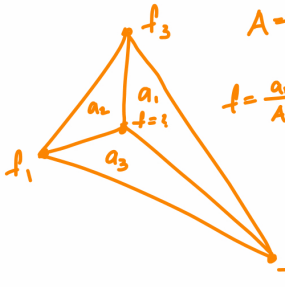
Line "Linear"



$$f(x) = \frac{L_1}{L} f_1 + \frac{L_2}{L} f_2$$

$$= \frac{x_2 - x}{x_2 - x_1} f_1 + \frac{x - x_1}{x_2 - x_1} f_2$$

2D Case



$$A = a_1 + a_2 + a_3$$

$$f = \frac{a_1}{A} f_1 + \frac{a_2}{A} f_2 + \frac{a_3}{A} f_3$$

"Linear interpolation over a triangle"

Half-toning

→ Binary output device can only "produce" two outputs:

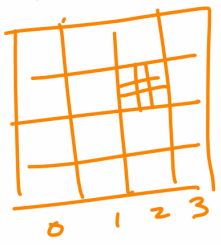
⇒ USE Larger MEGA_PIX to 'simulate' multiple intensity levels

2^9

3 × 3 MEGA-PIX

⇒ 10 intensity level

SCREEN



3 colors

2^{18}