Transformări grafice 2D

Translatie: $P(x,y) \xrightarrow{\text{translatie}} P'(x',y')$

$x' = x + tx$

$y' = y + ty$

$v = tx\vec{i} + ty\vec{j}$ vector de translație

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} tx \\ ty \end{bmatrix}$$

Rotatie: $P(x,y) \xrightarrow[\text{în jurul originii}]{\text{rotație de unghi } \theta} P'(x',y')$

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

$y' = \cancel{y}\sin\theta + y\cos\theta$

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

Scalare: $P(x,y) \xrightarrow{\text{scalare}} P'(x',y')$

$x' = sx \cdot x$

$y' = sy \cdot y$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} sx & 0 \\ 0 & sy \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$sx = sy$ -> scalare uniformă

! Pt a putea trata toate transformările prin înmulțiri de matrici se utilizează coordonate omogene. Un punct (x,y) este reprezentat în coordonate omogene printr-o infinitate de vectori 3D. -> (tx, ty, t)

- Dacă interpretăm (tx, ty, t) într-un spațiu 3D, ele reprezintă punctele unei drepte ce trece prin $(x, y, 1)$ și $(0,0,0)$.
- Fiecare punct omogen reprezintă o linie în spațiul 3D.
- 2 tripleți în coord omogene repr. același punct dacă sunt multiple unul altuia $(x, y, w) \to (\frac{x}{w}, \frac{y}{w}, 1)$

Coordonate Omogene

Translatie

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & tx \\ 0 & 1 & ty \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Rotatia

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

Scalarea

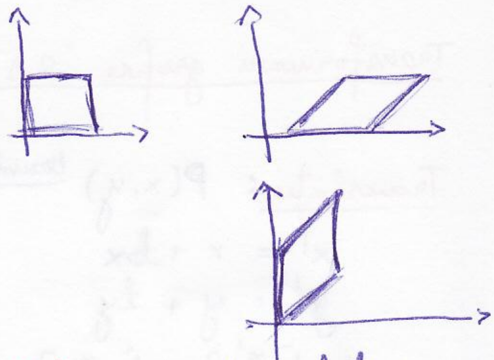
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} sx & 0 & 0 \\ 0 & sy & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- Repr. pct. în coord. omogene este deosebit de utilă pentru compunerea transformărilor geometrice
- Matricea corespunzătoare unei transformări compuse = produsul matr. transf elementare care o compun.
- Ordinea în care se plasează aceste matrice în produs este de la dreapta la stânga pt transf. repr cu vectori coloană și invers pt vectori linie
- Ex: $\begin{bmatrix} r_{11} & r_{12} & tx \\ r_{21} & r_{22} & ty \\ 0 & 0 & 1 \end{bmatrix}$, cu $\begin{bmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{bmatrix}$ ortogonală $(r_{11}r_{12} + r_{21}r_{22} = 0)$ corespunde unei transformări care păstrează unghiurile și lungimile (rigid body)
- O secvență de translații și rotații complexe are ca rezultat o transf rigidă
- Prin compunerea unui nr. arbitrar de rotații și scalări urmate de translații => transformări afine.

Shearing (forfecare)

- de-a lungul axei Ox $\quad SH_x = \begin{bmatrix} 1 & a & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

$$SH_x \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x + ay \\ y \\ 1 \end{bmatrix}$$



- $SH_y = \begin{bmatrix} 1 & 0 & 0 \\ b & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \Rightarrow SH_y \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y + bx \\ 1 \end{bmatrix}$

Compunerea mai multor transformări 2D

- Rotația de unghi θ a unui obiect în jurul $C(x_c, y_c)$, un pct. arbitrar
  1. Translație de vector $(-x_c, -y_c)$
  2. $R(\theta)$ în jurul originii
  3. $T(x_c, y_c)$ la loc

Matricea transf.: $T(+x_c, +y_c) \cdot R(\theta) \cdot T(-x_c, -y_c) =$

$$\begin{bmatrix} 1 & 0 & x_c \\ 0 & 1 & y_c \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -x_c \\ 0 & 1 & -y_c \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & x_c(1-\cos\theta) + y_c \sin\theta \\ \sin\theta & \cos\theta & y_c(1-\cos\theta) - x_c \sin\theta \\ 0 & 0 & 1 \end{bmatrix}$$

- nu contează ordinea de compunere pt: T-T, Sc-Sc, Rot, Rot, Sc. uniformă-Rot

Oglindirile

- față de axa Oy $\quad P(x, y) = P'(x', y') \quad \begin{cases} x' = -x \\ y' = y \end{cases}$

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

- față de axa Ox

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

- față de origine

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

Transformări de vizualizare 2D e o aplicație dintr-un sistem de coordonate în altul ce trebuie să pună în corespondență fiecărui punct al unui desen un punct al suprafeței de afișare. Se specifică un dreptunghi cu laturile // cu axele sist. de coordonate logice, numit fereastră, ale cărui puncte sunt puse în corespondență cu cele ale unui dreptunghi din suprafața de afișare, numit vizor.

Transformarea de vizualizare 2D se numește și transformarea fereastră-vizor

$$\frac{x_v - x_{vmin}}{x_{vmax} - x_{vmin}} = \frac{x_f - x_{fmin}}{x_{fmax} - x_{fmin}} \Rightarrow x_v = \frac{x_{vmax} - x_{vmin}}{x_{fmax} - x_{fmin}} (x_f - x_{fmin}) + x_{vmin} =$$

$$= \frac{x_{vmax} - x_{vmin}}{x_{fmax} - x_{fmin}} \cdot x_f - \underbrace{\frac{x_{vmax} - x_{vmin}}{x_{fmax} - x_{fmin}} \cdot x_{fmin} + x_{vmin}}_{t_x = -S_x \cdot x_{fmin} + x_{vmin}}$$

$$\begin{cases} x_v = S_x \cdot x_f + t_x \\ y_v = S_y \cdot y_f + t_y \end{cases} \qquad S_y = \frac{y_{vmax} - y_{vmin}}{y_{fmax} - y_{fmin}} \qquad t_y = -S_y \cdot y_{fmin} + y_{vmin}$$

Când fereastra și vizorul sunt dreptunghiuri care nu au același raport de aspect, scalarea realizată prin transformarea fereastră-vizor este neuniformă. $s_x \neq s_y$, ceea ce poate conduce la deformarea reprezentării care apare în vizor față de forma reprezentată în fereastră.
Pt. a evita acest lucru, se poate considera o scalare uniformă de factor $\min(s_x, s_y)$.

Transformări grafice 3D

Coordonate omogene: $(x, y, z) \longleftrightarrow (tx, ty, tz, t)$

$(x, y, z, w) \to \left(\frac{x}{w}, \frac{y}{w}, \frac{z}{w}\right)$; pt $w = 0$ „punct la infinit" - direcții

Mulțimea punctelor carteziene ce corespund punctelor omogene formează subspațiul tridimensional definit prin ecuația $w = 1$ al spațiului omogen 4D

$$T(t_x, t_y, t_z) = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad S(s_x, s_y, s_z) = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Rezultatul înmulțirii unui nr. arbitrar de matrice de rotație, translație sau scalare este o matrice de forma

$$\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} R_{3\times3} & T_{3\times1} \\ O_{1\times3} & 1_{1\times1} \end{bmatrix}$$

$R_{3\times3}$ → corespunde unei transf ortogonale

Forfecare (shear)

$$SH_{xy}(sh_x, sh_y) = \begin{bmatrix} 1 & 0 & sh_x & 0 \\ 0 & 1 & sh_y & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad SH_{yz}(sh_y, sh_z) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ sh_y & 1 & 0 & 0 \\ sh_z & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad SH_{xz}(sh_x, sh_z) = \begin{bmatrix} 1 & sh_x & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & sh_z & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Compunerea transformărilor 3D

Proprietăți ale matricilor ortogonale cu det 1 (cum sunt $R_{3\times3}$ - produsul unui nr arbitrar de rot)
- vectorii linie sunt un sistem de 3 vectori ortogonali, care, prin aplicarea matricei M, vor fi transformați în versorii $\vec{i}, \vec{j}, \vec{k}$ ai referențialului
- vectorii coloană repr. un sistem de versori ortogonali identici cu cei ce rezultă prin aplicarea matricei M asupra $\vec{i}, \vec{j}, \vec{k}$

Rotația unui punct P în jurul unei drepte oarecare $P_1(x_1, y_1, z_1) - P_2(x_2, y_2, z_2)$

1) $T_1(-x_1, -y_1, -z_1)$
2) $R_y(\alpha) \Rightarrow P_1P_2$ în xOy
3) $R_z(\beta) \Rightarrow P_1P_2$ coincide cu Oy

$$\cos\alpha = \frac{OB}{OA} = \frac{x_2 - x_1}{\sqrt{(x_2-x_1)^2 + (z_2-z_1)^2}} \qquad \sin\alpha = \frac{AB}{OA} = \frac{z_2 - z_1}{\sqrt{(x_2-x_1)^2 + (z_2-z_1)^2}}$$

$$\cos\beta = \frac{OC}{OP_2''} = \frac{y_2 - y_1}{\sqrt{(x_2-x_1)^2 + (y_2-y_1)^2 + (z_2-z_1)^2}}$$

$$\sin\beta = \frac{CP_2''}{OP_2''} = \frac{\sqrt{(x_2-x_1)^2 + (z_2-z_1)^2}}{\sqrt{(x_2-x_1)^2 + (y_2-y_1)^2 + (z_2-z_1)^2}}$$

4) $R_y(\theta)$
5) $R_z(-\beta)$
6) $R_y(-\alpha)$
7) $T_2(x_1, y_1, z_1)$

• Transf. care sunt rez. unui nr. arbitrar de compuneri de transformări de rotație, scalare, forfecare, urmate de o transformare de translație sunt transf. afine.

• Obiectele sunt definite într-un sistem de coord. utilizator care este convențional un sistem drept. În sist. de vizualizare utilizate în multe aplicații grafice se folosesc sist. de ref. stângi.

• Transformarea unui SR drept în SR stâng se face cu $M_{RL} = \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&-1&0\\0&0&0&1\end{bmatrix}$ (oglindire față de xOy)

• Transf. care mută un pct dintr-un sist. de coord în altul.

Pt. a determina ac. transf. a unui punct dintr-un sist de coord A în altul B ($M_{B\leftarrow A}$), se inversează matricea care aduce axele lui A peste ale lui B, transformarea făcându-se relativ la referențialul A.

Oglindirea față de un plan oarecare

• există oglindiri față de xOY $\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&-1&0\\0&0&0&1\end{bmatrix}$, yOZ $\begin{bmatrix}-1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}$, xOZ $\begin{bmatrix}1&0&0&0\\0&-1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}$

• oglindirea față de un plan oarecare este o transf. compusă care include una din transf. simple de oglindire, precum și transf. elementare de rotație și translație pt. a alinia normala planului dat cu o axă a sist. de coord

• considerăm planul de oglindire specificat printr-un punct $P(x_0, y_0, z_0)$ și vectorul normal la plan $\vec{N}$.

• procedura de obținere a transf. de oglindire față de planul dat este:

1) translație P în origine: $T(-x_0, -y_0, -z_0)$
2) aliniere vect. $\vec{N}$ la axa Oy
3) oglindire față de xOZ
4,5) invers 2 și 1.

Transformări de proiecție

• sunt transf. ale unor puncte specificate relativ la un sist. de coordonate de dimensiune n, prin puncte într-un sist. de coord de dim. $k < n$.

• proiecția unui obiect 3D este definită cu ajutorul unor drepte numite proiectori sau raze de pr., care pornesc dintr-un punct numit centru de proiecție, trec printr-un punct arbitrar al obiectului și intersectează un plan de proiecție. Astfel de proiecții se numesc proiecții geometrice (proiectorii sunt linii drepte, nu curbe) planare (pr. se face pe un plan, nu pe o supr. curbă).

• proiecții geometrice plane
- de perspectivă (distanța dintre centrul de pr. și planul de pr. finită) - se specifică coord centrului de pr
- paralele (distanța infinită) - se specifică coef. directori ai direcției de pr.

~~Mărimea~~

• mărimea proiecției perspectivă a unui obiect variază invers proporțional cu dist. de la obiect la centrul de proiecție; liniile paralele nu se proiectează tot ca linii paralele (în general), iar unghiurile nu sunt păstrate decât în cazul fețelor paralele cu planul de proiecție;

• proiecțiile liniilor paralele care nu sunt paralele cu planul de proiecție converg către un punct din acel plan, numit pct. de convergență

• dacă liniile proiectate sunt paralele cu una din axele sist de coord, punctul spre care converg se numește punct de convergență principal

• proiecțiile perspectivă se clasifică după nr. de pct. de convergență principale: 1, 2 sau 3

Proiecții paralele – **ortografice** (dir. de pr. este normală la planul de pr.)
– **oblice** (dir. de pr. nu e perpendiculară pe planul de pr.)

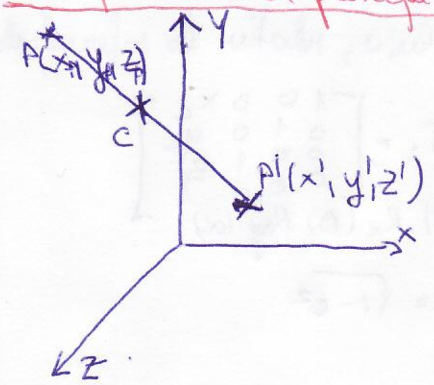
Proiecția ortografică

- **frontală, de sus, laterală** (planul de pr. e perpendicular pe o axă principală)
- **axonometrică** (pl. de pr. nu e perpendicular pe o axă principală)
  - *ex*: pr. izometrică -> dir. de proiecție face unghiuri egale cu fiecare dintre axele pr.
  - -> dir este $(dx, dy, dz) \Rightarrow |dx| = |dy| = |dz|$

Proiecții oblice

- **Proiecția cavalieră**: unghiul format de dir. de proiecție cu planul de proiecție este 45°
  (ex. pt xOy: $\left(\frac{\sqrt2}{2}, \frac{\sqrt2}{2}, -1\right)$ sau $\left(\frac{\sqrt2}{2}, \frac12, -1\right)$)
- **Proiecția cabinet**: unghiul format de dir. de proiecție cu planul de proiecție $= \operatorname{arctg} 2 \approx 63{,}4$
  (ex. pt xOy: $\left(\frac{\sqrt2}{4}, \frac{\sqrt2}{4}, -1\right)$ sau $\left(\frac{\sqrt3}{4}, \frac14, -1\right)$)

Transformarea de proiecție de perspectivă de centru $C(x_c, y_c, z_c)$ în plan de proiecție xOy



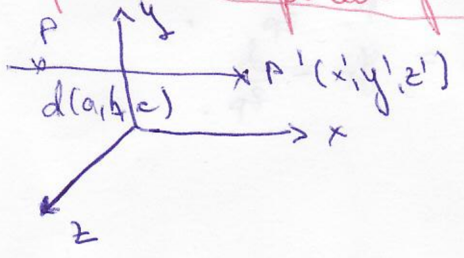
dreapta $d = PC$: $\dfrac{x - x_c}{x_p - x_c} = \dfrac{y - y_c}{y_p - y_c} = \dfrac{z - z_c}{z_p - z_c}$

pt x', y', z', $z' = 0 \Rightarrow \begin{cases} z' = 0 \\ x' = x_c - \dfrac{x_p - x_c}{z_p - z_c} \cdot z_c \\ y' = y_c - \dfrac{y_p - y_c}{z_p - z_c} \cdot z_c \end{cases}$

Transf. lui (x, y, z) prin pr. de $C(x_c, y_c, z_c)$ și plan $\Pi (z=0)$ este $(x', y', 0)$ unde $\begin{cases} x' = x_c - \dfrac{x - x_c}{z - z_c} \cdot z_c \\ y' = y_c - \dfrac{y - y_c}{z - z_c} \cdot z_c \end{cases}$

Dacă $x_c = y_c = 0$, $z_c = -d \Rightarrow \begin{cases} x' = +d \dfrac{x}{z+d} = \dfrac{x}{\frac{z}{d} + 1} \\ y' = +d \dfrac{y}{z+d} = \dfrac{y}{\frac{z}{d} + 1} \\ z' = 0 \end{cases}$ $M_{pr} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/d & 1 \end{bmatrix}$

Transformarea de proiecție paralelă pe planul $z=0$ având direcția de proiecție $d(a, b, c)$



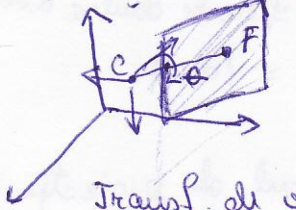
$\begin{cases} x = x_p + a \cdot t = x_p - \frac{a}{c} \cdot z_p \\ y = y_p + b \cdot t = y_p - \frac{b}{c} \cdot z_p \\ z = z_p + c \cdot t \Rightarrow t = -\frac{z_p}{c} \\ z = 0 \end{cases}$ $M_{pr} = \begin{bmatrix} 1 & 0 & -\frac{a}{c} & 0 \\ 0 & 1 & -\frac{b}{c} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

Transformarea de vizualizare 3D

- Un sist. de vizualizare 3D realizează transformările necesare pentru a duce un punct din spațiul de coord. 3D al utilizatorului într-un plan de vizualizare.
- Un sist. de viz. 3D se compune din:
  - **transformare de vizualizare** din sist. de coordonate utilizator în sist. camerei de luat vederi, în care originea e în poz. camerei de luat vederi
  - **transformare de proiecție** – proiectează punctele 3D din spațiul de vizualizare într-un spațiu cu 2D numit plan de vizualizare.
  - uneori, un sist. de vizualizare specifică un **volum de vizualizare**, care este submulțimea punctelor din sp. de coord. utilizator care se includ în procesul de vizualizare
  - se poate specifica și o **fereastră** în planul de vizualizare

Ex: $\begin{cases} C(x_c, y_c, z_c) - \text{camera} \\ F(x_f, y_f, z_f) - \text{punctul către care punctează camera} \\ \theta - \text{unghiul de răsucire al camerei în jurul } CF \\ d - \text{dist. dintre } C \text{ și pl. de vizualizare} \end{cases}$

Se folosește o proiecție de perspectivă de centru de proiecție C.



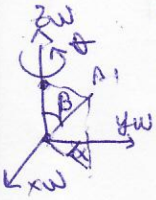
$$\bar{v} = (a, b, c)$$

$$a = \frac{x_f - x_c}{\sqrt{(x_f - x_c)^2 + (y_f - y_c)^2 + (z_c - z_f)^2}} \qquad b = \frac{y_f - y_c}{''} \qquad c = \frac{z_f - z_c}{''}$$

Transf. de vizualizare va trebui să calculeze coordonatele punctului P_v în sist (x_v, y_v, z_v) față de sist. de coordonate de vizualizare (al camerei) cunoscând coordonatele punctului $P_w(x_w, y_w, z_w)$ față de sist de coord utilizator.

$$P_v = M_{v \leftarrow w} A_w$$

$M_{v \leftarrow w} = T_{vw}^{-1}$ → aduce axele sist w peste axele lui v, relativ la referențialul



$$T_{vw} = T_1 \cdot T_2 \Rightarrow T_{vw}^{-1} = T_2^{-1} T_1^{-1}$$

T_1 - matr de transf. care aduce Ox_w peste C $\qquad T_1 = \begin{bmatrix} 1 & 0 & 0 & x_c \\ 0 & 1 & 0 & y_c \\ 0 & 0 & 1 & z_c \\ 0 & 0 & 0 & 1 \end{bmatrix}$

T_2 - așează Oz_w peste versorul CF $\qquad T_2^{-1} = R_z(\theta) R_x(\beta) R_y(\alpha)$

$$\cos\alpha = \frac{b}{\sqrt{1-c^2}} \qquad \sin\alpha = \frac{a}{\sqrt{1-c^2}} \qquad \cos\beta = c \qquad \sin\beta = \sqrt{1-c^2}$$

$$T_2^{-1} = R_z(\theta) \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c & -\sqrt{1-c^2} & 0 \\ 0 & \sqrt{1-c^2} & c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{b}{\sqrt{1-c^2}} & \frac{-a}{\sqrt{1-c^2}} & 0 & 0 \\ \frac{a}{\sqrt{1-c^2}} & \frac{b}{\sqrt{1-c^2}} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{vw}^{-1} = T_2^{-1} T_1^{-1} = R_z(\theta) \cdot \begin{bmatrix} \frac{b}{\sqrt{1-c^2}} & \frac{-a}{\sqrt{1-c^2}} & 0 & 0 \\ \frac{ac}{\sqrt{1-c^2}} & \frac{bc}{\sqrt{1-c^2}} & -\sqrt{1-c^2} & 0 \\ a & b & c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot T_1^{-1}$$

Transf. de vizualizare: centru în origine, plan $z_v = d$, (x_v, y_v, z_v)

$$\frac{x_v - 0}{x_p - 0} = \frac{y_v - 0}{y_p - 0} = \frac{z_v - 0}{z_p - 0}, \; z_v = d \Rightarrow \begin{cases} x_v = x_p \cdot \frac{d}{z_p} \\ y_v = y_p \cdot \frac{d}{z_p} \end{cases}$$

Algoritmi fundamentali în grafica raster pt. decuparea primitivelor 2D

Algoritmi de decupare (clipping)

În cazul pachetelor de programe pt afișarea de imagini raster există 2 etape ale procedurii
1) conversia primitivelor grafice în mulțimi de pixeli (rasterizare sau scan-conversion)
2) decuparea primitivelor grafice față de frontiera (de obicei dreptunghiulară) a regiunii care va fi afișate

- Cea mai simplă / ineficientă tehnică de clipping constă în efectuarea în întregime a scan-conversiei unei primitive și scrierea în frame buffer numai a pixelilor vizibili
- Coordonatele (x, y) ale fiecărui pixel sunt verificate ca să se încadreze între limitele $(x_{min}, y_{min}) - (x_{max}, y_{max})$ ale regiunii dreptunghiulare de afișat
- Acest tip de clipping este folosit numai în cazul sistemelor grafice în care aceste teste se fac prin hardware.
- Cea mai eficientă tehnică de decupare, utilizată în special pt decuparea liniilor drepte, este efectuarea decupării înainte de rasterizare.
- Rutina de decupare va genera numai pixelii porțiunii de primitivă grafică rămase după decupare (se numește și decupare analitică / pre-clipping)

Algoritm pt. decuparea liniilor față de o frontieră dreptunghiulară

1) Se determină dacă segmentul e integral în interiorul dreptunghiului (nu trebuie clipping)
2) Dacă linia nu poate fi acceptată trivial, ea e intersectată cu toate cele 4 laturi ale dreptunghiului-frontieră și se caută determinarea punctelor de intersecție care se află chiar pe latura dreptunghiului de decupare.
   - dacă nu există astfel de puncte, se produce rejectarea segmentului 
   - dacă sunt 2 puncte, se unesc 
   - dacă e 1 punct de intersecție, se unește acesta cu punctul dintre P și Q care e interior dreptunghiului de decupare (adică P sau Q) 

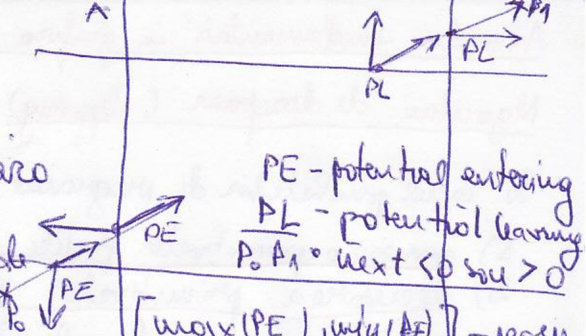
Algoritm de decupare Cohen-Sutherland

- $P(x_0, y_0) - Q(x_1, y_1)$ segment PQ se decupează față de $(x_{min}, y_{min}) - (x_{max}, y_{max})$
- la început se fac teste pt. acceptarea/rejectarea trivială a segm. PQ (nu se decupează sau nu se desenează).
- în acest scop, planul e împărțit în 9 regiuni de dreptele $x = x_{min}$, $x = x_{max}$, $y = y_{min}$, $y = y_{max}$
- unui punct care e plasat într-o regiune i se asociază un cod binar format din 4 biți $b_1 b_2 b_3 b_4$ astfel. $b_1 = 1$ dacă $y > y_{max}$, $b_2 = 1$, dacă $x > x_{max}$, $b_3 = 1$, $y < y_{min}$, $b_4 = 1$, $x < x_{min}$

1001	1000	1100
0001	0000	0100
0011	0010	0110

- segmentul e rejectat dacă cod (P) & cod (Q) ≠ 0000
- dacă cod(P) & cod (Q) = 0. Presupunem cod(P) ≠ 0000, $b_1 = 0 \Rightarrow PQ \cap y = y_{max}$; $b_2 \neq 0 \Rightarrow PQ \cap x = x_{max}$.
- se decupează și se reia cu segmentul rămas
- dacă segmentul nu a putut fi rejectat/acceptat trivial atunci biții nenuli corespund axelor pe care PQ le intersectează.

Algoritmul de decupare parametrică Cyrus – Beck



- Algoritmul folosește descrierea parametrică a liniei de decupat și a liniilor ce alcătuiesc frontiera poligonului de decupare
- determină valoarea parametrului t ce corespunde intersecției segm. P_0P_1 și fiecare dintre laturile (drepte) poligonului de decupare
- Fie n = nr de laturi ale poligonului de decupare
  - laturile sunt considerate în sens trigonometric
  - alg-ul va determina n valori ale lui t corespunzătoare intersecției dintre P_0P_1 și laturi și apoi prin comparații simple va selecta dintre ele două valori ce corespund extremităților porțiunii din P_0P_1, care e interioară poligonului de decupare
- punctul generic situat pe segmentul $P_0P_1 = p(t) = P_0 + (P_1 - P_0)t$
- Fie E_i una dintre muchiile (cons. as) a polig. de decupare, $\bar{n}_i$ versorul normalei ext. la muchie și P_{E_i} un pct arbitrar al muchiei E_i
- Punctul $P(t)$ va fi interior poligonului de decupare dacă vectorul $\overline{P(t)} - \overline{P_{E_i}}$ formează cu $\bar{n}_i$ un unghi $> \pi/2$ și $P(t)$ e ext. poligonului de pe frontieră $\bar{n}_i \perp (\overline{P(t)} - \overline{P_{E_i}})$ (prod scalar $= 0$)
- determinăm $\cap$ dintre E_i și P_0P_1 din ec. $\bar{n}_i(\overline{P(t)} - \overline{P_{E_i}}) = 0$, $\Rightarrow$

$$\Rightarrow \boxed{t = \frac{n_i(P_1 - P_{E_i})}{\overline{N_1} \cdot \overline{P_0P_1}}}$$

- pt a avea o valoare validă a lui t, $|\bar{n}_i| \neq 0$ și $|\overline{P_0P_1}| \neq 0$ și $\bar{n}_i \perp \overline{P_0P_1}$
- valorile lui t care nu $\in [0, 1]$ nu vor fi luate în considerare.
- Se sortează valorile t ale intersecțiilor valide între E_i și P_0P_1. Se determină cea mai mare valoare a lui $t \in [0, 1]$ care corespunde unei extremități de tip PE, fie ea t_e. Se determină cea mai mică valoare a lui $t \in [0, 1]$ care corespunde unei intersecții de tip PL, fie ea t_L. Dacă $t_L < t_e$, se produce rejectarea segmentului P_0P_1. Dacă $t_L > t_e$, se afișează porțiunea cuprinsă între punctele $P(t_e)$ și $P(t_L)$.

Generarea prin puncte a primitivelor grafice (rasterizare)

Algoritmul DDA (Digital Differential Analyzer)

- algoritmul de rasterizare pt. linii
- calculează coordonatele pixelilor aflați în nodurile unei grile 2D care sunt cei mai apropiați de o dreaptă teoretică infinit de subțire.
- în cazul liniilor cu pantă $m \in [-1, 1]$ va fi aprins un singur pixel pe fiecare coloană de puncte a display-ului
- $m \notin [-1, 1]$ – un singur pixel pe linie
- Linie DDA $vect((x_0, y_0), (x_1, y_1))$; panta $m = \frac{\Delta y}{\Delta x} = \frac{y_1 - y_0}{x_1 - x_0}$,

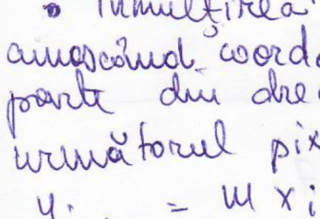
  ec: $y_i = m x_i + b$
- vor trebui aprinși pixelii cu coordonatele $(x_i, y_i) = (x_i, round(m x_i + b))$
- această abordare ar implica o adunare, o înmulțire și o conversie de format pt. fiecare pixel.
- înmulțirea poate fi eliminată prin următorul calcul incremental: cunoscând coordonatele unui pct de raster selectat pt. a face parte din dreapta rasterizată la iterația i la iterația $(i+1)$ următorul pixel va fi selectat va avea coord. $x_{i+1} = x_i + 1$,

  $y_{i+1} = m x_{i+1} + b = m x_i + m + b = m(x_i + \Delta x) + b = m x_i + m \Delta x + b =$
  $= y_i + m \Delta x, \quad \Delta x = 1$
- dacă $|m| > 1$, la fiecare iterație valoarea lui y va crește cu o unitate, iar valoarea lui x_{i+1} va fi în $round(x_i + \frac{1}{m})$.

Bresenham: un algoritm mai rapid pt generarea de linii

- lucrează doar cu nr. întregi, evitând utilizarea lui round la fiecare iterație
- considerăm o dreaptă cu $m \in [0, 1)$ (în primul octant; algoritmul se va modifica pt. a trasa drepte în ceilalți octanți)

Demonstrație:



- Pp. că la o iterație oarecare, algoritmul a selectat punctul $P(x_p, y_p)$ pt a face parte din dreapta rasterizată.
- La iterația următoare, deoarece dreapta are o pantă $m \in [0, 1)$, va fi selectat unul din pixelii E sau NE pt. a face parte din dreapta rasterizată. (ambii pixeli se află pe coloana $x = x_p + 1$)
- Dacă pct. M, situat la 1/2 NE-E, este în partea stângă (adică deasupra) a dreptei teoretice, va fi selectat E, altfel va fi selectat NE.
- Eroarea = distanța pe verticală între pixelul ales și dreapta teoretică; eroarea $\leq 1/2$
- Fie dr. $F(x, y) = ax + by + c = 0$ (ecuație implicită)
- În cazul unei drepte care trece prin 2 puncte (x_0, y_0) și (x_1, y_1) ecuația explicită e $y = \frac{dy}{dx} + B$; $dy\,x - dx\,y + B\,dx = 0$
- Pt pct aflate:
  - la dreapta (dedesubtul) dreptei $F(x, y) > 0$
  - la stânga dreptei $F(x, y) < 0$
  - pe dreaptă $F(x, y) = 0$

- la pasul anterior am selectat $P(x_p, y_p)$
- la pasul acesta, calculăm $F(x_M, y_M)$, unde $d = F(x_M, y_M) = dy\,(x_p + 1) - dx\,(y_A + 1/2) + b\,dx$

$d > 0 \Rightarrow NE \rightarrow M''$

$d < 0 \Rightarrow E \rightarrow M'$

$d = 0 \Rightarrow$ oricare, E

$x_{M'} = x_p + 2 = x_{M''}$

$y_{M'} = y_p + \frac{1}{2}$

$y_{M''} = y_p + \frac{3}{2}$.

$$\begin{cases} d_{M'} = d + dy = d + a \\ d_{M''} = d + dy - dx = d + a + b \end{cases}$$

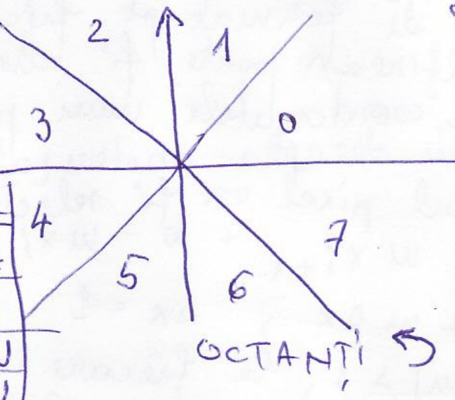
$\rightarrow$ primul pixel al dreptei $= (x_0, y_0)$

- valoarea inițială a lui $d = F(x_0 + 1, y_0 + 1/2) = a(x_0 + 1) + b(y_0 + 1/2) + c = \overset{=0}{F(x_0, y_0)} + a + b/2 \Rightarrow d_{start} = a + \frac{b}{2} = dy - \frac{dx}{2}$
- Ca să nu împărțim la 2, considerăm $F(x, y) = 2(ax + by + c)$

$d_{start} = 2dy - dx = 2a + b$

$i_{poz} = 2(dy - dx)$

$i_{neg} = 2dy$



OCTANTUL	dstart	ipos	ineg	pct	sel
0	$2dy - dx$	$2dy$	$2(dy - dx)$	E	NE
1	$-2dx + dy$	$\cancel{+}2(dy - dx)$	$-2dx$	NE	N
2	$-dy - 2dx$	$-2dx$	$-2(dy - dx)$	N	NW
3	$-2dy - dx$	$-2(dy + dx)$	$-2dy$	NW	W
4	$-2dy + dx$	$-2dy$	$-2(dy - dx)$	W	SW
5	$2dx - dy$	$2(dx - dy)$	$2dx$	SW	S
6	$dy + 2dx$	$2dx$	$2(dy - dx)$	S	SE
7	$2dy + dx$	$2(dy + dx)$	$2dy$	SE	E

Generarea cercurilor prin puncte

- cel mai ineficient: prin rezolvarea ec. implicite $x^2 + y^2 = r^2$ (în primul cadran) $y = \sqrt{r^2 - x^2} \rightarrow$ dacă s-a generat $(x, y) \in$ cadranul 1, atunci pct. $(-x, y)$, $(-x, -y)$, $(x, -y)$ fac parte din cerc și vor fi generate.
- Mai puțin ineficient:
  - generăm microvectori cu vf. de coord. $r\cos\theta$, $r\sin\theta$
  - cercul se va aproxima cu un poligon regulat

```
moveTo (r, 0)
for (i=1; i<=N; i++)
    lineTo ((int) r * cos (2πi/N), (int) r * sin (2πi/N))
```

- Eficient
  - lucrează cu numere întregi: Bresenham, Mitchener
  - se generează pct din al 2-lea octant
  - celelalte puncte se obțin prin 8 simetrii: (x, y), $(x, -y)$, $(-x, -y)$, $(-x, y)$, (y, x), $(-y, x)$, $(-y, -x)$, $(y, -x)$

Pp. că la o iterație oarecare a fost selectat pt. a face parte din cerc pct $P(x_p, y_p)$.

$F(x,y) = x^2 + y^2 - r^2 = 0$

Dacă $M \in int(C)$, $F(x_M, y_M) < 0$ va fi selectat E, altfel SE, $F(x_M, y_M) > 0$.

$d = F(x_M, y_M) = (x_p+1)^2 + (y_p - \frac{1}{2})^2 - r^2$

$d < 0 \Rightarrow E$

$d_{next} = (x_p+2)^2 + (y_p - \frac{1}{2})^2 - r^2 = d_{current} + \underbrace{2x_p + 3}_{i_{neg}}$

$d > 0 \Rightarrow SE$

$d_{next} = (x_p+2)^2 + (y_p - \frac{3}{2})^2 - r^2 = d_{current} + 2x_p + 3 - 2(y_p + \frac{1}{2}) + 1$

$= d_{current} + \underbrace{2x_p - 2y_p + 5}_{i_{pos}}$

- i_{neg} și i_{pos} nu mai sunt constante
- $d_{start} = F(x_0 + 1, y_0 - \frac{1}{2}) = 2x_0 + 1 - y_0 + \frac{1}{4} = \frac{5}{4} - R$

  $h = d - \frac{1}{4} \Rightarrow h = 1 - R$

  $d < 0 \Rightarrow h < -\frac{1}{4}$ dar cum lucrăm cu valori întregi, testăm doar $h < 0$

Elipsă (algoritmul lui DaSilva)

- elipsa cu semiaxele paralele cu Ox, Oy: $F(x,y) = b^2x^2 + a^2y^2 - a^2b^2 = 0$

  $\rightarrow$ alg. pt. primul sector (cadran)
- $grad\, F(x,y) = \frac{\partial F}{\partial x}\vec{i} + \frac{\partial F}{\partial y}\vec{j} = 2b^2x\vec{i} + 2a^2y\vec{j}$
- în pct care separă reg I și reg II, gradientul are panta 1
- în regiunea I, grad. are componenta pe y > comp pe x

  (e mai mult vertical vectorul) $2a^2y > 2b^2x$
- în această regiune, la iterația următoare a algoritmului se va alege între punctele situate în E sau SE punctului ales la it. curentă
- alegerea se face în funcție de semnul lui $F(x_p + 1, y_p - \frac{1}{2})$
- Analog, în cazul reg II, la iterația următoare a alg se va alege între pct. plasate la S și SE față de punctul P ales la iterația curentă
- la reg I, $d_{old} = F(x_p+1, y_p - \frac{1}{2}) = b^2(x_p+1)^2 + a^2(y_p - \frac{1}{2})^2 - a^2b^2$

$d_{new} = b^2(x_p+2)^2 + a^2(y_p - \frac{1}{2})^2 - a^2b^2.$

E $d_{new} = d_{old} + b^2(2x_p + 3) = d_{old} + \Delta est$

SE $d_{new} = d_{old} + \Delta est + a^2(-2y_p + 2) = d_{old} + \Delta sest$

$\Delta sest = b^2(2x_p + 3) + a^2(-2y_p + 2)$

- a, b întregi $F(1, b - \frac{1}{2}) = b^2 + a^2(b - \frac{1}{2})^2 - a^2b^2 = b^2 + a^2(-b + \frac{1}{4})$

Primitive de suprafață

- se referă la umplerea continuă sau folosind un șablon repetitiv a unei zone 2D specificate prin conturul său.
- când zona care trebuie umplută este definită prin culoarea pixelilor de pe contur și umplerea acestei zone începe dintr-un punct (seed) ale cărui coordonate sunt param. ai fct.
- un caz particular - umplerea poligonală: frontiera zonei umplute este un poligon specificat prin vârfuri (utilizat și în algoritmii de ascundere de suprafețe)
- Etape:
  I Se determină intersecția dreptei de rasterizare cu muchiile poligonului
  II - Se sortează intersecțiile crescător după x
  III - Desenarea pixelilor care se află între o intersecție de ordin impar și una de ordin par
  IV - Trece la urm. dreaptă de rasterizare, goto I

Reprezentarea curbelor și suprafețelor

①* De cele mai multe ori, suprafețele sunt reprezentate folosind clase de poligoane sau suprafețe parametrizate. Suprafețele pot fi folosite și pt a reprezenta corpuri solide. Una din metodele de repr. a solidelor se numește boundary representation.

②* Clasa de poligoane (polygonal mesh) este o colecție de suprafețe plane având fiecare o frontieră de formă poligonală.
Acest gen de repr. poate aproxima și suprafețe curbe (ca sferele noastre).

① În spațiu, se pot defini curbe polinomiale parametrizate (x, y, z)
$x = p(t)$, $y = q(t)$, $z = r(t)$, unde p, q, r - polinoame de o sg. var.

② Repr. explicită a unei curbe $y = f(x)$, $z = g(x)$

③ Repr. implicită $f(x, y, z) = 0$ (de ex. $x^2 + y^2 + z^2 - r^2 = 0$ e repr. unei sfere) ③

④* Coef. acestor polinoame se aleg a.î. curba să aibă forma dorită (polin. p, q, r au de obicei gradul 3 și curba s.n. cubică parametrizată)

⑤* Supr. curbe pot fi modelate prin petice de supr. polinomială parametrizată, unde pct. generic are ec. $x = p(u, v)$, $y = q(u, v)$, $z = r(u, v)$.
În general, o suprafață curbă oarecare poate fi aproximată printr-un nr. ~~de~~ petice parametrizate mai mic decât nr. unor petice poligonale care ar realiza o aproximare cu același grad de acuratețe.
De cele mai multe ori, polinoamele p, q, r au gradul 3 în fiecare din cele 3 variabile. Supr. corespunzătoare sunt numite bicubice și sunt alcătuite din petice bicubice parametrizate.

O clasă poligonală este o colecție de muchii, vârfuri și poligoane conectate a.î. fiecare muchie să fie partajată de cel mult 2 poligoane, o muchie să conecteze 2 vârfuri și fiecare vârf să fie partajat de cel puțin 2 muchii

Operații tipice (asupra unei clase poligonale)

① Det. muchiilor incidente unui vârf dat

② Det. poligoanelor adiacente unui vf sau muchii date

③ Det. vârfurilor conectate de o muchie

④ Det. tuturor muchiilor unui poligon

⑤ Afișarea ~~unei~~ clasei poligonale

⑥ Identificarea eventualelor erori de reprezentare (se numesc erori topologice)

Regula lui Euler: $V - M + B - 2 = 0$.

a) Repr. explicită a unei clase poligonale.

Fiecare poligon e repr. prin lista coordonatelor vf. sale

- vârfurile ~~sunt~~ stocate în listă în ordinea în care sunt întâlnite la parcurgerea trigonometrică a poligonului
- muchiile sunt plasate între oricare 2 vârfuri succesive, precum și între primul și ultimul
- repr. este eficientă pentru un singur poligon, dar în cazul claselor poligonale duplică vârfuri și nu repr. explicit muchiile partajate

b) pointeri într-o listă de vârfuri

- fiecare vf. e menționat o singură dată
- vârfurile se pot modifica foarte ușor
- cererea de tip 2 e greu de satisfăcut

c) muchii explicite

- pointeri într-o listă de muchii
- lista vârfurilor e ca la b)
- fiecare muchie se repr. prin pointeri la cele 2 vf.
- unele repr. mai pun și pointeri care partajează o muchie (winged edge)

• Determinarea normalei

- se aleg 3 vârfuri ale polinomului (P_1, P_2, P_3) și se calculează $\overline{P_1P_2} \times \overline{P_1P_3}$
- dacă produsul este nul (P_1, P_2, P_3 coliniari) se vor alege alte 3 vârfuri.
- dacă poligonul are mai mult de 3 vf., ele pot să nu fie riguros coplanare; se poate determina un plan π care să treacă prin vf poligonului a.î. să fie minimizată $\sum_{i=1}^{n} |d_i|$ suma dist de la pct la plan

$(\pi): Ax + By + CZ + D = 0$

- coef. A, B, C trebuie să fie proporționali cu ariile cu semn ale proiecției poligonului pe planul $\underset{A}{yOz}$, $\underset{B}{xOz}$, $\underset{C}{xOy}$

Curbe de interpolare / aproximare

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

- Polinom Lagrange (de interpolare) $L_{n-1}(x) = \sum_{i=1} y_i \prod_{j=1}^{n} \frac{x - x_i}{x_i - x_j}$

- polinomul Lagrange e o funcție puternic oscilantă

Proprietăți de aproximare

① existența unor pct de control ce det. forma curbei

② curba e cuprinsă în întregime în poligonul de înfășurătoare convexă al pct de control (convex hull)

③ Controlul local (curba Spline) și global (curba Bezier) asupra formei curbei

Dacă un proiectant modifică poziția unui punct de control, forma curbei se poate schimba local (într-o vecinătate a pct) sau global (pe întregul d.def al curbei).

④ Ordinul de continuitate

- curbele sunt formate din mai multe „segmente de curbă", în pct de racordare ale acestor segmente de curbă se pot pune condiții de continuitate

-> clasa C_1: vectorii tangentei și modulele lor sunt identice

-> clasa C_2: doar orientarea tangentei

- de obicei în descrierea curbelor de aproximare se folosesc curbe parametrice, cu parametrii $\in [0, 1]$

- 3 tipuri de curbe de aproximare: Hermite, Bezier, Spline

Curbe Bezier de grad n

• Def: curba al cărui punct generic este $P(u)$, în funcție de coord a $n+1$ punct de control P_i, prin relația $P(u) = \sum_{i=0}^{n} A_i B_{in}(u) \quad 0 \le u \le 1$

$B_{in}(u) = C_n^i u^i (1-u)^{n-1}$ ← polinom Bezier

• Ec. parametrice ale curbei Bezier în cazul bidirecțional:

$x(u) = \sum_{i=0}^{n} x_i B_{in}(u)$

$y(u) = \sum_{i=0}^{n} y_i B_{in}(u) \quad 0 \le u \le 1$

• Punctele de control ale unei curbe Bezier de ~~grad~~ ord n satisfac urm prop:

① cele 2 puncte de control extreme (p_0, p_n) aparțin curbei

② în pct de control extreme, curba Bezier e tangentă laturilor corespunzătoare ale poligonului de control

③ la concatenarea a 2 curbe Bezier pot fi asigurate cu ușurință cerințele de cont. de ord 1 în pct. de racordare (laturile poligonului de control adiacente punctului de control final comun al celor 2 curbe să fie situate în prelungire.

• În cazul unui poligon de control convex, curba e în interiorul acestui poligon

• Controlul asupra formei curbei este global: schimbăm poziția unui pct de control => schimbăm forma întregii curbe.

• O suprafață Bezier este definită prin următoarea ec. param:

$$(*)\quad P(u,v) = \sum_{i=0}^{m}\sum_{j=0}^{n} P_{ij}\, B_{i,m}(u)\, B_{j,n}(v), \quad 0 \le u, v \le 1$$

• Forma acestei suprafețe este determinată de poziția punctului de control P_{ij} $(0 \le i \le n,\ 0 \le j \le n)$. P_{ij} sunt vârfuri ale unui poliedru de control

• În ecuația (*), polinoamele Bernstein B_{im} și B_{jn} au gradele m, resp n.
• Petice bicubice parametrizate

Algoritmi pt. vizualizarea realistă a scenelor 3D

• Algoritmii de rendering descriu procesul de obținere a unei reprezentări colorate realist a unui obiect pornind de la o reprezentare a unui model 3D a obiectului:

Etapele alg. de rendering:

① Producerea unui model care conține toată informația necesară pt. a produce o reprezentare realistă a obiectului (ex: o plasă de poligoane)
② Aplicarea de transformări liniare modelului corpului 3D
③ Eliminarea poligoanelor auto-obturate (back face culling)
④ Decuparea poligoanelor rămase față de frontiera unui volum de vizualizare
⑤ Rasterizarea poligoanelor: cu alg. de tip scan-line
⑥ Aplicarea de alg. pt. eliminarea suprafețelor ascunse
⑦ Colorarea realistă a pixelilor interiori fiecărui poligon, utilizând scheme interpolative sau incrementale.

Tehnici pt. a dezvolta imagini realiste:
• alg. de det a supr. vizibile
• alg. de shading prin interpolarea culorilor
• modele de iluminare locală, surse punctuale cu suprafață
• modelarea suprafețelor curbe și a prop. de materiale
• aplicarea texturilor și umbrelor
• modelarea transparenței / reflexiei
• metode de iluminare globală - Ray Tracing, Radiosity.

Alg. de eliminare a liniilor / supr. ascunse

{ care lucrează în spațiul imagine (Image Precision Algorithms)
care lucrează în spațiul obiect (Object Precision Algorithms)

• Image Precision Alg. determină care dintre cele n obiecte este vizibil în fiecare dintre pixelii imagine:
pt fiecare pixel p al imaginii repetă:
* det. obiectul cel mai apropiat de observator
raza care pornește de la observator și trece prin pixelul P.
care este situat pe
* desenează pixelul P cu culoarea adecvată

• Object Precision Alg compară obiectele între ele, eliminând direct obiecte întregi sau porțiuni de obiect care nu sunt vizibile.
pt fiecare obiect o din scenă repetă:
* det. porțiunile din ob. care nu sunt obstruate de alte ob. sau de alte părți ale lui o
* desenează porțiunile vizibile cu culoarea adecvată

Algoritmi de determinare a vizibilității fețelor:

1) bazați pe lista de priorități: Z-buffer, Weiler - Atherton
2) alg. scan - linie - det. părțile vizibile ale fețelor prin baleierea imaginii care se afișează linie cu linie: Watkins
3) Alg bazați pe subdivizare: divizează recursiv fețele scenei până se pot stabili porțiunile vizibile ale lor: arbori BSP - binary space partition

Z - buffer

- lucrează în spațiul imagine
- e implementat hardware
- Alături de memoria video (Frame Buffer), care reține codul de culoare al fiecărui pixel al imaginii, algoritmul folosește și un buffer de memorie (numit Z - buffer) care conține valoarea de adâncime Z a obiectului din scenă a cărui imagine ce corespunde coordonatei poate fi observată într-un anumit pixel din spațiul imagine (nr de componente ale Z-bufferului coincide cu nr. de pixeli ai imaginii)
- Considerăm scena alcătuită din corpuri cu frontiera modelată prin plase de poligoane ⇒ fiecare poligon P va fi proiectat în planul de vizualizare, obținându-se poligonul P'. Punctele interioare ale P' vor fi aprinse cu culoarea corespunzătoare folosind un algoritm de rasterizare de tipul scan-linie.
- În timpul procesului de rasterizare, pixelul-imagine având coordonatele (x_i, y_i) va fi aprins numai dacă coordonata z_i care corespunde acestui punct (din poligonul inițial) este una mai apropiată de observator decât coordonata Z coresp. din Z - buffer.
- Inițial Z-bufferul e inițializat cu o valoare mică, iar valoarea tuturor punctelor din memoria video e inițializată la valoarea culorii de fond a scenei.
- pt. fiecare poligon P al scenei repetă:
  * proiectează P pe planul xOy => P'
  * pt fiecare pixel Q(x, y) interior P' repetă
    - pZ ← val. coord Z a poligonului P în pct prin a cărui proiecție se obț Q
    - dacă pZ ≥ Zbuffer [x, y]
      * Zbuffer [x, y] ← pZ
      * put pixel (x, y, culoare-poligon-în-pct-Q)
- se folosește proiecția ortografică pe xOy
- nu este necesară sortarea prealabilă a poligoanelor
- pt. a putea det. adâncimea unui punct al poligonului P, este necesară cunoașterea ecuației planului poligonului. Fie ea $Ax + By + Cz + D = 0 \Rightarrow Z = -\frac{Ax + By + D}{C}$
- Pt a mări eficiența calculelor de adâncime, se poate folosi coerența în adâncime imaginii (de obicei $\Delta x = 1$ pixel): cunosc $z_1(x_1, y_1)$ atunci pt $(x_1 + \Delta x, y_1) \Rightarrow z_2 = z_1 - \frac{A}{C}\Delta x$

• asemănător, pt a determina valoarea lui z coresp pct de start al unei linii de rasterizare $y + \Delta y$, cunosc $y_1 \Rightarrow z_2 = z_1 - \frac{B}{C}\Delta y$

• dacă proiecția folosită e paralelă cu o direcție (a, b, c) oarecare, înainte de a aplica z-buffer se va aplica fiecărui corp din scenă o secvență de transformări care să aducă dreapta (a, b, c) de-a lungul axei $Oz \to (0, 0, 1)$

• Algoritmul z-buffer poate lucra la fel de bine și asupra corpurilor care nu au frontiera poliedrală, trebuie să se poată determina coordonata z a unui punct al suprafeței ce are proiecția (x, y) dată.

• În cazul în care z-bufferul ar avea dimensiuni prea mari (există limitări ale memoriei interne a sistemului), se poate aplica repetat algoritmul asupra unor fâșii de imagine suficient de subțiri pt. a permite folosirea unui z-buffer.

• uneori, în programele de arhivare de imagini se salvează alături de o imagine corespunzătoare unei scene și z-bufferul ei, ceea ce permite inserarea ulterioară cu ușurință a noi obiecte în scenă.

Object-buffer

• variantă a z-buffer, de Atherton.

• se asociază fiecărui pixel (x, y) din imagine o listă ordonată după z a corpurilor care se proiectează peste punctul (x, y).

• în acest fel se pot realiza efecte cum ar fi transparența

Back face culling

• Eliminarea fețelor din spate sau auto-obturate ale unui poliedru

• Definirea fețelor suprafeței poliedrale (o plasă de poligoane) trebuie să fie făcută prin enumerarea vârfurilor feței în sens trigonometric

• În acest fel, calculul normalei ca produs vectorial a 2 din laturile (considerate vectori orientați) feței poligonale va produce normala exterioară.

• În sist de coordonate ale observatorului, fețele din spate ale poliedrului sunt caracterizate prin faptul că produsul scalar dintre normala exterioară la față și vectorul $\vec{OP}$ (o - pct de observare, P - un pct oarecare al feței) este pozitiv

• Algoritmul lucrează în spațiul obiect și realizează înjumătățirea numărului fețelor poliedrelor componente ale unei scene; numai fețelor neauto-obturate li se vor aplica eventual alți alg. de ascundere de suprafețe sau de shading

Algoritmul pictorului

• Ideea algoritmului este de a rasteriza poligoanele componente ale unei scene în ordinea descrescătoare a distanței lor față de observator.

• În cazul scenelor ale căror obiecte sunt situate în plane cu $z = ct$ (scene $2\frac{1}{2}D$), aplicarea alg. pictorului este evidentă

• În cazul general în care poligoanele componente ale scenei nu sunt situate în plane cu $z = ct$, pot apărea o serie de ambiguități generate de intersectarea sau suprapunerea pe axa z a poligoanelor. (se rezolvă prin descompunerea unui poligon al scenei în mai multe subpoligoane)

① Se sortează toate poligoanele scenei într-o listă în ordinea crescătoare a celei mai mici coordonate z a fiecărui poligon; aceasta este coordonata celui mai depărtat punct de observator pe Oz (se consideră proiecția ortogonală pe xOy)

② Rezolvarea ambiguităților care pot fi determinate de suprapunerea pe axa Oz a poligoanelor și împărțirea, dacă este cazul, a acestor poligoane

- Fie P poligonul aflat la extremitatea cea mai depărtată a listei sortate
  Înainte ca P să fie rasterizat, el va trebui testat în raport cu toate celelalte poligoane Q din listă ale căror extensii pe Oz se suprapun cu cea a lui P.

2.1 se verifică dacă extensiile pe x ale lui P și Q nu se suprapun (dacă nu, atunci se randează P și apoi Q)

2.2 analog pt extensiile pe y

2.3 se verifică dacă poligonul P (toate vf. sale) și poziția observatorului sunt situate de părți opuse față de planul poligonului Q. → se substituie coordonatele fiecărui vârf al lui P în ecuația planului lui Q și se testează ca semnul valorii obținute să fie contrar celui rezultat prin substituția coord obs. în ec. pl. lui Q

2.4 Se verifică dacă Q și obs. sunt de aceeași parte față de planul lui P.

2.5 Se verifică dacă proiecțiile lui P și Q în planul xOy nu se suprapun

⊛ dacă unul din testele 2.1 – 2.5 se termină cu succes, P este rasterizat

⊛ dacă toate testele se termină cu eșec, atunci P ar putea să-l obtureze pe Q; se va testa dacă Q nu ar putea fi rasterizat înaintea lui P. → nu se mai repetă testele 2.1, 2.2 și 2.5, dar sunt efectuate 2.3 și 2.4 cu rolul poligoanelor P și Q inversat → 2.3' și 2.4'

⊛ dacă cel puțin unul din testele 2.3', 2.4' se termină cu succes, poligonul Q va fi mutat în poziția celui mai depărtat poligon al scenei față de observator

⊛ Pt. a evita for-ul (loop), poligoanele care se află la începutul listei vor fi marcate; dacă în cazul unui poligon marcat testele 2.1 – 2.5 FAIL => EPIC FAIL, nu se mai încarcă 2.3' și 2.4' pt că ar fi FAIL și resp. poligon este descompus, iar părțile lui componente sunt reinserate în listă (comp. coord z ja desc. polig.)

⊛ e un alg hibrid: operează atât în sp. obiect cât și în sp. imagine (rasterizarea)

Arbori BSP (Binary Space Partition)

- Structură independentă de pct de observare și care permite alg. de vizualizare rapidă a planelor de poligoane

  $P = \{p_1, p_2, \dots p_n\}$ - mulțime de poligoane
- se alege un poligon p_k, care va reprezenta rădăcina arborelui BSP.
- planul poligonului p_k va partiționa spațiu 3D în 2 semispații S_k și S_k'

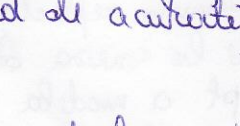
  $S_k = \{(x, y, z) \mid ax + by + cz + d > 0\}$
- Poligoanele lui $P \setminus \{p_k\}$ pot aparține în întregime lui S_k sau S_k'
- Dacă un polig din $P - \{p_k\}$ intersectează planul lui p_k, el va fi împărțit în 2 subpoligoane, unul în S_k, celălalt în S_k'
- Procesul de împărțire a submulțimii de poligoane continuă până la obținerea de submulțimi formate dintr-un singur element
- Afișarea corectă a unui poligon implică următoarea ordine de generare în memoria ecran:
  - ⊛ generarea tuturor poligoanelor aflate de cealaltă parte a planului poligonului p_k decât observatorul
  - ⊛ generarea lui p_k
  - ⊛ generarea tuturor poligoanelor aflate de aceeași parte

Modele de reflexie/iluminare

- Un model de reflexie descrie interacțiunea luminii cu suprafața unui obiect, în funcție de proprietățile suprafeței și natura luminii incidente => proiecția unui obiect în planul de vizualizare pare reală.
- Modelul de iluminare definește natura luminii emise de o sursă luminoasă, geometria distribuției intensității luminoase. Sursele luminoase sunt punctiforme.
- Scopul modelelor de reflexie este de a realiza redarea obiectelor 3D în spațiul-ecran 2D a.î. realitatea înconjurătoare să fie "imitată" cu un grad de acuratețe acceptabil
- Modele de reflexie
  - locale (iau în consid. doar interacțiunea dintre un pct al suprafeței și sursa luminoasă)
  - globale (iau în consid. și lumina care ajunge într-un pct. al suprafeței obiectului pe căi indirecte - prin reflexia sau transmisia radiației luminoase pe alte suprafețe ale scenei)

Modele de reflexie locale:

- reflexia difuză (lambertiană) - caracteristică suprafețelor mate.

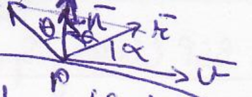


- ⊛ intensitatea luminii reflectate difuz de un element de arie infinit de mic centrat în P: $i = k_d\, i_p \cos\theta = k_d\, i_p\, \bar{n} \cdot \bar{l}$ $\quad \theta \in [0, \pi/2]$ pt supr ne-auto-obturate

  ↳ intensitatea sursei luminoase (i_p)

  ↳ coeficient de reflexie difuză (k_d)
- ⊛ uneori se ține seama și de lumina ambiantă, care e o cantitate de lumină constantă adăugată la cea reflectată: $i = k_a i_a + k_d\, i_p\, \bar{n} \cdot \bar{l}$

- reflexia speculară – caracteristică suprafețelor lucioase



$i_s = K_s i_p \cos^n \alpha = K_s i_p (\bar{r} \cdot \bar{v})^n$ (modelul lui Tui Bui Phong)

$\bar{r}$ – rad. reflectată
$\bar{v}$ – versorul de observare

n – exponent al reflexiei speculare, $n \in [1, 200]$ (se alege experimental)

$\overline{PQ} = \bar{n} \cos\theta$

$\bar{S} = \bar{n} \cos\theta - \bar{l} = \bar{r} - \bar{n} \cos\theta$

$\bar{R} = 2\bar{n} \cos\theta - \bar{l} = 2\bar{n}(\bar{n}\bar{l}) - \bar{l}$

(*) O altă formulare a modelului de reflexie Phong pt. reflexia speculară folosește vectorul $\bar{h}$ – versorul dir. bisectoare între direcția sursei, de versor $\bar{l}$ și cea a observatorului, de versor $\bar{v}$ $\quad \bar{h} = \dfrac{\bar{l} + \bar{v}}{|\bar{l} + \bar{v}|}$ în loc de $(\bar{r} \cdot \bar{v})^n \Rightarrow (\bar{l} \cdot \bar{n})^n$, $\cos\beta = \bar{l} \cdot \bar{h} \neq \cos\alpha = \bar{r} \cdot \bar{v}$

(*) e un model empiric; folosirea lui β e mai eficientă când sursa luminoasă și obs. sunt situate la infinit. -> în acest caz, direcția lui h este constantă pt orice P de pe suprafața obiectului

(*) În unele formulări, modelul Phong include, alături de componenta speculară și componenta lambertiană și ambiantă: $i = K_a i_a + K_d i_p (\bar{l} \cdot \bar{n}) + K_s i_p (\bar{r} \cdot \bar{v})^n$

(*) modelul Phong
- sursă luminoasă punctuală
- sursa și observatorul sunt de obicei considerate la infinit
- culoarea reflexiei speculare e considerată culoarea sursei luminoase (de ob. albă)
- termenul ambiant = constant

(*) alte modele: Blinn, Torrance – Sparrow, Cook – Torrance

(*) Acestea sunt modele teoretice / fizice, care consideră suprafața obiectului alcătuită din microfețe care au normalele distribuite în jurul unei normale medii

Modele de iluminare ale surselor luminoase – 3 elemente:
- geometria sursei (punctuală, liniară, dreptunghiulară)
- distribuția intensității luminoase a sursei
- distribuția spectrală a sursei

(*) Modelul lui Warn: Intensitatea sursei $= i_l (\bar{l}_N \cdot \bar{l})^s$
-> versorul dir. de la sursă la pct. ob. lumin.
-> indice de concentrare a sursei
-> direcție principală

(*) modelul este echivalent cu o suprafață care primește lumină dintr-o sursă punctuală și o reflectă specular conform modelului Phong.

Atenuarea sursei luminoase
- lumina reflectată de suprafața unui obiect poate fi atenuată în funcție de distanța de la sursa luminoasă până la punctul de interes de pe suprafața obiectului.
- pt. a modela această atenuare se introduce un factor de atenuare conform relației:

$$i = f_{at} \cdot i_p \cdot K_d (\bar{n} \cdot \bar{l}), \text{ unde } f_{at} = \min\left(\frac{1}{\underbrace{c_1 + c_2}_{\text{constante definite de ut. asociate cu sursa luminoasă}} d_e + c_3 d_e^2}, 1\right)$$

d_e -> distanța pe care lumina o parcurge de la sursă pct la supr. obiectului

Lumini și suprafețe colorate
- Luminile și suprafețele colorate sunt tratate scriind ecuații separate pt. fiecare componentă a modelului de culoare
- Culoarea difuză a unui obiect se reprezintă prin intensitățile reflectate (O_d) pt fiecare componentă (ex: (O_{dR}, O_{dG}, O_{dB})
- În acest caz, cele 3 componente ale luminii sursei luminoase (i_{pR}, i_{pG}, i_{pB}) sunt reflectate respectiv în proporțiile $K_d O_{dR}$, $K_d O_{dG}$, $K_d O_{dB}$. Comp. roșie $i_R = i_{aR} \cdot K_a \cdot O_{dR} + f_{at} \cdot i_{pR} \cdot K_d O_{dR} (\bar{n} \cdot \bar{l})$

Algoritmi de colorare realistă (shading)

a) pentru suprafețele modelate prin plase de poligoane

1. Shading constant (flat shading)

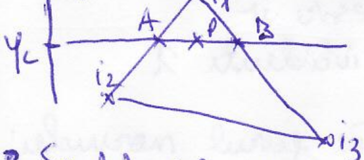
- se aplică un model de reflexie pe un singur punct din interiorul unei fețe poligon, iar culoarea calculată e utilizată pt. a umple întregul poligon.
- ac. metodă presupune că:
  - sursa luminoasă e la ∞ ($\bar{n} \cdot \bar{l}$ = ct pe fața poligonală)
  - observatorul e la ∞ ($\bar{n} \cdot \bar{v}$ = ct pe fața poligonală)
  - poligonul care e colorat nu e o aproximare a unei suprafețe curbe

2. Shading cu interpolarea culorii

- acest tip de metode (metode de shading incrementale sau interpolative) aplică modelele de reflexie poligoanelor din plasă, calculând intensitățile luminoase în vârfurile poligoanelor și apoi interpolând valorile de intensitate sau culoare din vârf în fiecare din punctele interioare ale poligoanelor
- cele mai răspândite metode de shading incremental: Gouraud, Phong

Metoda Gouraud

- Fie un poligon P al plasei. Se calculează normala la suprafață în fiecare din vârfurile poligonului.
- Dacă plasa poligonală aproximează o suprafață analitică, se pot calcula normalele în vârfurile poligonului pornind direct de la ecuațiile suprafeței
- Dacă nu se cunosc ecuațiile suprafeței aproximate, atunci normala la un vârf al plasei de poligoane se poate calcula prin medierea normalelor la poligoanele adiacente respectivului vârf.
- Se determină intensitatea luminii reflectate în fiecare vârf al plasei de poligoane (se aplică un model de reflexie)
- Se determină intensitatea fiecărui punct de pe muchiile poligonului prin interpolarea liniară a valorilor intensităților celor 2 vf de extremitate ale muchiei



$$\frac{i_A - i_1}{i_2 - i_1} = \frac{y_c - y_1}{y_2 - y_1} \Rightarrow i_A = i_1 + (i_2 - i_1)\frac{y_c - y_1}{y_2 - y_1}$$

$$\frac{i_B - i_1}{i_3 - i_1} = \frac{y_c - y_1}{y_3 - y_1} \Rightarrow i_B = i_1 + (i_3 - i_1)\frac{y_c - y_1}{y_3 - y_1} \quad (*)$$

- Se det intensitățile de pe fiecare punct al poligonului de pe dreapta de rasterizare curentă, prin interpolarea intensităților pct de la intersecția $y = y_c$ cu muchiile polig.

$$\frac{i_P - i_A}{i_B - i_A} = \frac{x_P - x_A}{x_B - x_A} \Rightarrow i_P = i_A + (i_B - i_A)\frac{x_P - x_A}{x_B - x_A}$$

- Schema Gouraud ia în considerare numai componenta difuză a modelului de reflexie în calculul intensităților la vârf, pt. că altfel forma petei speculare ar depinde prea mult de plasa poligonală
- De aceea, normalele la vârf se calculează o singură dată și sunt depozitate în structura de date a plasei poligonale.
- Calculele (*) se pot efectua incremental
- Pot apărea erori de reprezentare / vizualizare datorate efectului Mach (muchiile comune au porțiuni mai întunecate, dintr-un motiv dubios)
- Det. culorii e echivalentă cu 3 interpolări de intensitate

Metoda Phong - interpolarea vectorului normal

- Se det. normalele la vârf pt. fiecare din vf. poligonului
- Se det. normalele în fiecare din pct. unei muchii, prin interpolarea liniară a coeficienților normalelor la vârf
- Se det. normalele în fiecare din punctele interioare poligonului situate pe dreapta curentă de rasterizare prin interpolarea liniară a coef. normalelor situate la extremitățile segm. de rasterizare
- Se efectuează un calcul separat de intensitate pt. fiecare din normalele calculate
- Obs: Se consideră că sursa luminoasă și observatorul sunt plasate la infinit => intensitatea luminoasă într-un punct trebuie să depindă doar de normala interpolată.
- Obs: Modelul de reflexie folosit ia în considerare componenta speculară.
- Obs: Metoda Phong necesită mult mai multe calcule decât metoda Gouraud, dar dă rezultate mai bune
- Obs: Metoda Phong ia 50% din calculul total de rendering
- Obs: Metoda Gouraud e un standard la stațiile de lucru
- Obs: Au fost puse la punct tehnici pt. accelerarea calculelor metodei Phong: abordări geometrice (Bergman) și numerice (Bishop-Weimer)

Metoda de combinare Gouraud-Phong

- se aplică Gouraud la toate poligoanele plasei, iar Phong doar pe polig. supraluminate
- Partea cea mai importantă a unui model de reflexie locală este să calculăm cât din lumina incidentă este reflectat de o suprafață
- Calculul reflexiei luminii se reduce la a calcula o funcție BRIDF (Bidimensional Reflection Intensity Distribution Function)
- Parametrii funcției BRIDF: - direcția luminii incidente $\bar{l}$
  - direcția radiației propagate la ieșire $\bar{v}$
  - lungimea de undă a luminii incidente λ

raport dintre intensitatea luminii propagate la ieșire și i. rad. incidente

- $BRIDF(\bar{l}, \bar{v}, \lambda) \doteq \frac{i_{out}}{i_{in}}$
- Dacă funcția $BRIDF(\bar{l}, \bar{v}, \lambda)$ nu are o simetrie circulară în jurul normalei la suprafață, suprafața se numește anizotropică.

Modele de culoare

- specificare cantitativă a culorii într-un sistem de coordonate
- o culoare: 3-4 valori reale
- Modelul de culoare: descrierea sistemului de coordonate și descrierea unei mulțimi de culori care se numește spațiul culorilor și e o submulțime a spațiului descris de sist. de coordonate
- O culoare e caracterizată din punct de vedere al percepției de următoarele mărimi:
  - ⊛ nuanță (hue) - distinge între 2 culori diferite (mărimea fizică = lungimea de undă dominantă)
  - ⊛ saturație - măsura de alb amestecată într-o culoare pură (m. f = puritatea de excitare)
  - ⊛ luminozitate - intensitatea luminoasă percepută la reflectarea luminii de o suprafață col.
  - ⊛ strălucire - intensitatea luminoasă percepută de la un obiect-sursă luminoasă

Modelul RGB

- utilizat pt. specificarea culorii emise în cazul monitoarelor color
- sist. de coord este ortogonal
- spațiul culorilor are formă cubică (roșu 100, verde 010, albastru 001, alb 111, negru 000)
- modelul RGB e aditiv $C = \underbrace{rR + gG + bB}$ → culori primare; r, g, b → ponderi

Modelul CMY (cyan magenta yellow)

- complementele lui RGB
- alb = 000; cyan + yellow va absorbi roșu și albastru și va reflecta numai verde
- e un model substractiv - se indică ce se scoate din alb
- CMYK (black) -> variantă a CMY folosită la specificarea culorilor la imprimante
- în procesul de afișare prin imprimare, pigmenții C+M+Y nu dau un negru perfect, de aceea se folosește depunerea de pigment negru $K = \min(c, m, y)$

Modelul YIQ

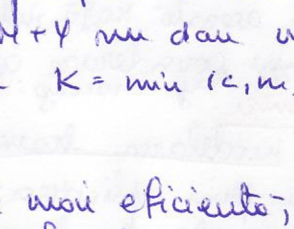
- utilizat în transmiterea imaginilor color TV
- recodificare a RGB a.î. să se obțină o transmisie mai eficientă, comprimată, și să se păstreze compatibilitatea cu codificarea imaginilor transmise alb-negru.
- subspațiul YIQ e un poliedru convex care corespunde cubului RGB care a suferit o transformare liniară.

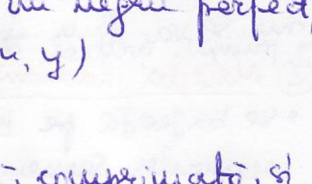
$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & 0.275 & 0.321 \\ 0.212 & 0.538 & 0.311 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

I, Q - semnale de crominanță → cod. cul. sist. color
Y - semnal de luminanță, e singura comp. transmisă la imag. alb/negru.

- sistemul vizual uman e mai sensibil la variația de luminanță față de schimbarea de nuanță/saturație ale unei culori
- de aceea, în formatele digitale de codificare a culorii care folosesc modelul YIQ, luminanța se codifică pe mai mulți biți față de I și Q.

Modelul HSV (hue, saturation, value = brightness)





- sist de coordonate e cilindric
- spațiul culorilor e o piramidă hexagonală regulată
- baza piramidei = planul V = 1
- H se măsoară în grade. 0° - roșu, 60° - galben, 120° - verde, 180° - cyan, 240° - albastru, 300° - magenta
- Spre deosebire de modelele RGB, CMY, YIQ, care sunt orientate către hardware, modelul HSV e orientat către utilizator, indicând modul în care el percepe culoarea,

Algoritmi de iluminare globală

① metoda drumului optim - Ray Tracing
② metoda radianței - Radiosity

- alg. de iluminare globală calculează culoarea într-un punct P al imaginii luând în considerare nu numai lumina emisă direct de o suprafață luminoasă care ajunge în P, ci și lumina ce ajunge în P în urma unor reflexii sau transmisii față de supr. altor obiecte ale scenei.

① Metoda Ray-Tracing

- Calculele de determinare a suprafețelor invizibile și de shading sunt între-pătrunse cu cele de determinare a umbrelor, razelor reflectate și razelor transmise.
- alg. trasează raze imaginare între un punct ce reprezintă ochiul observatorului și centrul fiecărui pixel din subspațiul imagine.
- raza e intersectată cu obiectele scenei, determinându-se punctul de intersecție cel mai apropiat de observator.
- În acest punct se determină, aplicând un model de reflexie locală (de obicei Phong) culoarea a obiectului în acel punct.
- raza care vine din ochiul observatorului = raza primară
- raza transmisă + reflectată = raze secundare
- Pt. determinarea umbrei, din pct P se duce câte o rază până la fiecare sursă luminoasă. Dacă această rază intersectează suprafața altui obiect, P va fi umbrit și nu se va lua în considerare contribuția sursei și la calculul culorii locale a lui P.
- primul articol de Ray-Tracing: Appel, Goldstein

② Metoda Radianței

- se bazează pe modelarea transferului de energie luminoasă între suprafețele corpurilor scenei prin aplicarea legii conservării energiei
- algoritmul care stă la baza metodei radianței lucrează în spațiul obiect
- suprafețele corpurilor scenei de vizualizat sunt descompuse în petice, pe fiecare petic este selectat un punct reprezentativ (de obicei centrul de greutate al petecului); peticele sunt de obicei dreptunghiulare)
- se determină radianța peticului în fiecare din punctele reprezentative.

E_i - energia luminoasă emisă de un petic de suprafață
R_i - reflexivitatea peticului = $\frac{\text{Rad care iese}}{\text{Rad care intră}}$
B_i - radianța peticului (energia ce părăsește unit. de suprafață în unit. de timp)

$$\begin{bmatrix} 1-R_1F_{11} & -R_2F_{12} & \cdots & -R_nF_{1n} \\ \vdots & & \ddots & \\ -R_nF_{n1} & -R_2F_{n2} & \cdots & 1-R_nF_{nn} \end{bmatrix} \boxed{\begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_n \end{bmatrix}} = \begin{bmatrix} E_1 \\ E_2 \\ \vdots \\ E_n \end{bmatrix}$$

F_{ij} = factori de formă, depind de geometria scenei
= fracțiunea din energia ce părăsește peticul j și care ajunge la peticul i
- se calculează în f.ț. de forma și orientarea relativă, precum și prezența unor petice obstructive.

- După determinarea radianțelor B_1 .. B_n ale punctelor reprezentative ale peticelor, se determină radianțele la vârf.
- Metoda a fost dezvoltată de Cohen și Greenberg