Transformări

Mihai-Sorin Stupariu

Sem. al II-lea, 2022 - 2023

Exemple de transformări. Funcții OpenGL

Formalizare - introducerea celei de-a patra coordonate

Manevrarea transformărilor

Funcții standard pentru transformări în OpenGL

- `glTranslate*(t);` Translația $\mathrm{T_t}$ de vector $\mathrm{t} = (t_1, t_2, t_3)$

Funcții standard pentru transformări în OpenGL

- `glTranslate*(t);` Translația $\mathrm{T_t}$ de vector $\mathrm{t} = (t_1, t_2, t_3)$

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \overset{\mathrm{T_t}}{\mapsto} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} t_1 \\ t_2 \\ t_3 \end{pmatrix}.$$

Funcții standard pentru transformări în OpenGL

- `glTranslate*(t);` Translația $\mathrm{T_t}$ de vector $\mathrm{t} = (t_1, t_2, t_3)$

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \stackrel{\mathrm{T_t}}{\mapsto} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} t_1 \\ t_2 \\ t_3 \end{pmatrix}.$$

- `glScale*(s);` Scalarea σ_{s} de factor $\mathrm{s} = (s_1, s_2, s_3)$ (de-a lungul celor trei axe, centrul scalării fiind în origine - punct fix al transformării)

Funcții standard pentru transformări în OpenGL

- `glTranslate*(t);` Translația $\mathrm{T_t}$ de vector $\mathrm{t} = (t_1, t_2, t_3)$

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \overset{\mathrm{T_t}}{\mapsto} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} t_1 \\ t_2 \\ t_3 \end{pmatrix}.$$

- `glScale*(s);` Scalarea σ_{s} de factor $\mathrm{s} = (s_1, s_2, s_3)$ (de-a lungul celor trei axe, centrul scalării fiind în origine - punct fix al transformării)

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \overset{\sigma_{\mathrm{s}}}{\mapsto} \begin{pmatrix} s_1 & 0 & 0 \\ 0 & s_2 & 0 \\ 0 & 0 & s_3 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}.$$

Funcții standard pentru transformări în OpenGL

- glTranslate*(t); Translația $\mathrm{T_t}$ de vector $\mathrm{t} = (t_1, t_2, t_3)$

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \overset{\mathrm{T_t}}{\mapsto} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} t_1 \\ t_2 \\ t_3 \end{pmatrix}.$$

- glScale*(s); Scalarea σ_{s} de factor $\mathrm{s} = (s_1, s_2, s_3)$ (de-a lungul celor trei axe, centrul scalării fiind în origine - punct fix al transformării)

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \overset{\sigma_{\mathrm{s}}}{\mapsto} \begin{pmatrix} s_1 & 0 & 0 \\ 0 & s_2 & 0 \\ 0 & 0 & s_3 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}.$$

- glRotate*(θ,u); Rotația $\mathbb{R}_{\mathrm{u},\theta}$ de unghi θ și axă dată de versorul u
  Rotația 2D $\mathbb{R}_{3,\theta}$ de axă Ox_3 (adică $\mathrm{u} = (0, 0, 1)$ și unghi θ (centrul rotației fiind în origine - punct fix al transformării) este dată de

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \overset{\mathbb{R}_{Ox_3,\theta}}{\mapsto} \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}.$$

Exemplu (1)

Fie aplicația afină f dată de

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \overset{f}{\mapsto} \begin{pmatrix} 2x_1 + 4x_2 \\ 3x_1 - x_2 \end{pmatrix}. \tag{1}$$

Obs. Utilizăm formalismul cu coloane pentru consistența lucrului cu matrice; aplicația se scrie și $f : \mathbb{R} \to \mathbb{R}, \; f(x_1, x_2) = (2x_1 + 4x_2, 3x_1 - x_2)$.

Exemplu (1)

Fie aplicația afină f dată de

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \stackrel{f}{\mapsto} \begin{pmatrix} 2x_1 + 4x_2 \\ 3x_1 - x_2 \end{pmatrix}. \tag{1}$$

Obs. Utilizăm formalismul cu coloane pentru consistența lucrului cu matrice; aplicația se scrie și $f : \mathbb{R} \to \mathbb{R},\ f(x_1, x_2) = (2x_1 + 4x_2, 3x_1 - x_2)$.

(i) Calculați $f(0,0), f(2,5), f(e_1), f(e_2)$.

Exemplu (1)

Fie aplicația afină f dată de

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \overset{f}{\mapsto} \begin{pmatrix} 2x_1 + 4x_2 \\ 3x_1 - x_2 \end{pmatrix}. \tag{1}$$

Obs. Utilizăm formalismul cu coloane pentru consistența lucrului cu matrice; aplicația se scrie și $f : \mathbb{R} \to \mathbb{R},\ f(x_1, x_2) = (2x_1 + 4x_2, 3x_1 - x_2)$.

(i) Calculați $f(0,0), f(2,5), f(e_1), f(e_2)$.

(ii) Scrieți relația (1) sub forma matriceală. Ce observați? (legătura cu $f(e_1), f(e_2)$).

Exemplu (1)

Fie aplicația afină f dată de

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \stackrel{f}{\mapsto} \begin{pmatrix} 2x_1 + 4x_2 \\ 3x_1 - x_2 \end{pmatrix}. \qquad (1)$$

Obs. Utilizăm formalismul cu coloane pentru consistența lucrului cu matrice; aplicația se scrie și $f : \mathbb{R} \to \mathbb{R},\ f(x_1, x_2) = (2x_1 + 4x_2, 3x_1 - x_2)$.

(i) Calculați $f(0,0), f(2,5), f(e_1), f(e_2)$.

(ii) Scrieți relația (1) sub forma matriceală. Ce observați? (legătura cu $f(e_1), f(e_2)$).

(iii) Cum procedăm dacă ținem cont de a treia coordonată?

Calcule

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \overset{f}{\mapsto} \begin{pmatrix} 2x_1 + 4x_2 \\ 3x_1 - x_2 \end{pmatrix}; \qquad f(x_1, x_2) = (2x_1 + 4x_2, 3x_1 - x_2)$$

(i) Calculați $f(0,0), f(2,5), f(e_1), f(e_2)$.

$f(0,0) = (0,0)$; $\begin{pmatrix} 0 \\ 0 \end{pmatrix} \mapsto \begin{pmatrix} 0 \\ 0 \end{pmatrix}$

$f(2,5) = (24,1)$; $\begin{pmatrix} 2 \\ 5 \end{pmatrix} \mapsto \begin{pmatrix} 24 \\ 1 \end{pmatrix}$

$f(e_1) = f(1,0) = (2,3)$; $\begin{pmatrix} 1 \\ 0 \end{pmatrix} \mapsto \begin{pmatrix} 2 \\ 3 \end{pmatrix}$

$f(e_2) = f(0,1) = (4,-1)$; $\begin{pmatrix} 0 \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} 4 \\ -1 \end{pmatrix}$

Calcule

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \overset{f}{\mapsto} \begin{pmatrix} 2x_1 + 4x_2 \\ 3x_1 - x_2 \end{pmatrix}; \qquad f(x_1, x_2) = (2x_1 + 4x_2, 3x_1 - x_2)$$

(ii) Scrieți f folosind reprezentarea matriceală. Ce observați? (legătura cu $f(e_1), f(e_2)$).

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \overset{f}{\longmapsto} \begin{pmatrix} 2x_1 + 4x_2 \\ 3x_1 - x_2 \end{pmatrix} = \begin{pmatrix} 2 & 4 \\ 3 & -1 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

$f(e_1)$ $f(e_2)$

coloanele acestei matrice A_f sunt exact vectorii $f(e_1)$ și $f(e_2)$

Calcule

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \overset{f}{\mapsto} \begin{pmatrix} 2x_1 + 4x_2 \\ 3x_1 - x_2 \end{pmatrix}; \qquad f(x_1, x_2) = (2x_1 + 4x_2, 3x_1 - x_2)$$

(iii) Cum procedăm dacă ținem cont de a treia coordonată?

$A_f = \begin{pmatrix} 2 & 4 \\ 3 & -1 \end{pmatrix}$; vrem să luăm în considerare și a 3^a coordonată

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \longmapsto \begin{pmatrix} 2x_1 + 4x_2 \\ 3x_1 - x_2 \\ x_3 \end{pmatrix} \rightsquigarrow \underset{(3\times 3)}{\tilde{A}_f} = \begin{pmatrix} 2 & 4 & 0 \\ 3 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 2x_1 + 4x_2 \\ 3x_1 - x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 2 & 4 & 0 \\ 3 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$$

Exemplu (2)

Aceleași cerințe pentru aplicația afină f dată de

$$f(x_1, x_2) = (2x_1 + 4x_2 + 5, 3x_1 - x_2 - 2) \tag{2}$$

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \mapsto \begin{pmatrix} 2x_1 + 4x_2 + 5 \\ 3x_1 - x_2 - 2 \end{pmatrix} = \begin{pmatrix} 2x_1 + 4x_2 \\ 3x_1 - x_2 \end{pmatrix} + \begin{pmatrix} 5 \\ -2 \end{pmatrix} =$$

$$= \begin{pmatrix} 2 & 4 \\ 3 & -1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} 5 \\ -2 \end{pmatrix}$$

Proprietăți - de reținut!

(i) Pentru f aplicație afină 2D dată de

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \stackrel{f}{\mapsto} \begin{pmatrix} a_{11}x_1 + a_{12}x_2 \\ a_{21}x_1 + a_{22}x_2 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \qquad (3)$$

au loc relațiile

$$\begin{pmatrix} 0 \\ 0 \end{pmatrix} \stackrel{f}{\mapsto} \begin{pmatrix} 0 \\ 0 \end{pmatrix}; \qquad \begin{pmatrix} 1 \\ 0 \end{pmatrix} \stackrel{f}{\mapsto} \begin{pmatrix} a_{11} \\ a_{21} \end{pmatrix}; \qquad \begin{pmatrix} 0 \\ 1 \end{pmatrix} \stackrel{f}{\mapsto} \begin{pmatrix} a_{12} \\ a_{22} \end{pmatrix}$$

Coloanele matricei sunt exact $f(e_1)$, $f(e_2)$.

Proprietăți - de reținut!

(i) Pentru f aplicație afină 2D dată de

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \overset{f}{\mapsto} \begin{pmatrix} a_{11}x_1 + a_{12}x_2 \\ a_{21}x_1 + a_{22}x_2 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \tag{3}$$

au loc relațiile

$$\begin{pmatrix} 0 \\ 0 \end{pmatrix} \overset{f}{\mapsto} \begin{pmatrix} 0 \\ 0 \end{pmatrix}; \qquad \begin{pmatrix} 1 \\ 0 \end{pmatrix} \overset{f}{\mapsto} \begin{pmatrix} a_{11} \\ a_{21} \end{pmatrix}; \qquad \begin{pmatrix} 0 \\ 1 \end{pmatrix} \overset{f}{\mapsto} \begin{pmatrix} a_{12} \\ a_{22} \end{pmatrix}$$

Coloanele matricei sunt exact $f(e_1)$, $f(e_2)$.

(ii) Pentru f aplicație afină 2D dată de

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \overset{f}{\mapsto} \begin{pmatrix} a_{11}x_1 + a_{12}x_2 \\ a_{21}x_1 + a_{22}x_2 \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} \tag{4}$$

Proprietăți - de reținut!

(i) Pentru f aplicație afină 2D dată de

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \overset{f}{\mapsto} \begin{pmatrix} a_{11}x_1 + a_{12}x_2 \\ a_{21}x_1 + a_{22}x_2 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \tag{3}$$

au loc relațiile

$$\begin{pmatrix} 0 \\ 0 \end{pmatrix} \overset{f}{\mapsto} \begin{pmatrix} 0 \\ 0 \end{pmatrix}; \qquad \begin{pmatrix} 1 \\ 0 \end{pmatrix} \overset{f}{\mapsto} \begin{pmatrix} a_{11} \\ a_{21} \end{pmatrix}; \qquad \begin{pmatrix} 0 \\ 1 \end{pmatrix} \overset{f}{\mapsto} \begin{pmatrix} a_{12} \\ a_{22} \end{pmatrix}$$

Coloanele matricei sunt exact $f(e_1)$, $f(e_2)$.

(ii) Pentru f aplicație afină 2D dată de

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \overset{f}{\mapsto} \begin{pmatrix} a_{11}x_1 + a_{12}x_2 \\ a_{21}x_1 + a_{22}x_2 \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} \tag{4}$$

au loc relațiile

$$\begin{pmatrix} 0 \\ 0 \end{pmatrix} \overset{f}{\mapsto} \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}; \begin{pmatrix} 1 \\ 0 \end{pmatrix} \overset{f}{\mapsto} \begin{pmatrix} a_{11} \\ a_{21} \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}; \begin{pmatrix} 0 \\ 1 \end{pmatrix} \overset{f}{\mapsto} \begin{pmatrix} a_{12} \\ a_{22} \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

Rotații 2D

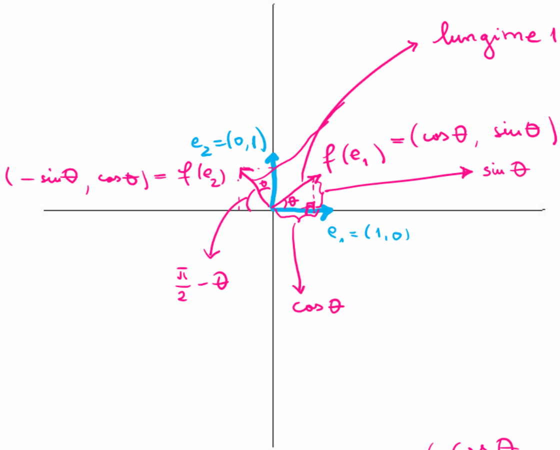
Rotația 2D (cu axa de rotație u $= (0, 0, 1)$) de unghi θ, cu centrul în origine are matricea 2×2 asociată dată de

$$\begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix},$$

iar matricea 3×3 este

$$\begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Figura



Dacă $f = R_{(0,0,1),\theta}$ (în plan) $\Rightarrow$ $M_f = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$

Concluzii intermediare

- Pentru o transformare 2D se poate considera și a treia coordonată x_3, ca fiind invariantă (adică nu își modifică valoarea) și se poate introduce o reprezentare matriceală corespunzătoare.

Concluzii intermediare

- Pentru o transformare 2D se poate considera și a treia coordonată x_3, ca fiind invariantă (adică nu își modifică valoarea) și se poate introduce o reprezentare matriceală corespunzătoare.
- Cum se poate proceda pentru a reprezenta cât mai uniform și termenii de gradul 0 (componenta translațională)?

Vârfuri și direcții - rolul celei de-a 4 coordonate

- **Motivație:** Este necesar un cadru în care transformările să fie reprezentate în mod uniform și compunerea lor să fie ușor de descris: **folosind "coordonate omogene" și considerând 4 coordonate**.

Vârfuri și direcții - rolul celei de-a 4 coordonate

- **Motivație:** Este necesar un cadru în care transformările să fie reprezentate în mod uniform și compunerea lor să fie ușor de descris: **folosind "coordonate omogene" și considerând 4 coordonate**.
- Coordonatele omogene verifică proprietatea fundamentală

  $$[\alpha : \beta : \gamma : \delta] = [\lambda\alpha : \lambda\beta : \lambda\gamma : \lambda\delta], \quad \forall \lambda \neq 0, (\alpha, \beta, \gamma, \delta) \neq (0, 0, 0, 0).$$

  De exemplu, $[2 : 3 : -1 : 4] = [4 : 6 : -2 : 8] \neq [4 : -1 : 3 : 2]$.

Vârfuri și direcții - rolul celei de-a 4 coordonate

- **Motivație:** Este necesar un cadru în care transformările să fie reprezentate în mod uniform și compunerea lor să fie ușor de descris: **folosind "coordonate omogene" și considerând 4 coordonate**.
- Coordonatele omogene verifică proprietatea fundamentală

  $$[\alpha : \beta : \gamma : \delta] = [\lambda\alpha : \lambda\beta : \lambda\gamma : \lambda\delta], \quad \forall \lambda \neq 0, (\alpha, \beta, \gamma, \delta) \neq (0, 0, 0, 0).$$

  De exemplu, $[2 : 3 : -1 : 4] = [4 : 6 : -2 : 8] \neq [4 : -1 : 3 : 2]$.
- Unui **vârf** de coordonate (x_1, x_2, x_3), notate și (x, y, z) i se asociază **coordonatele omogene**

  $$[x_1 : x_2 : x_3 : 1] \text{ (sau } [x : y : z : 1]).$$

Vârfuri și direcții - rolul celei de-a 4 coordonate

- **Motivație:** Este necesar un cadru în care transformările să fie reprezentate în mod uniform și compunerea lor să fie ușor de descris: **folosind "coordonate omogene" și considerând 4 coordonate**.
- Coordonatele omogene verifică proprietatea fundamentală

  $$[\alpha : \beta : \gamma : \delta] = [\lambda\alpha : \lambda\beta : \lambda\gamma : \lambda\delta], \quad \forall\lambda \neq 0, (\alpha, \beta, \gamma, \delta) \neq (0,0,0,0).$$

  De exemplu, $[2 : 3 : -1 : 4] = [4 : 6 : -2 : 8] \neq [4 : -1 : 3 : 2]$.
- Unui **vârf** de coordonate (x_1, x_2, x_3), notate și (x, y, z) i se asociază **coordonatele omogene**

  $$[x_1 : x_2 : x_3 : 1] \text{ (sau } [x : y : z : 1]).$$

- Unei **direcții** date de vectorul (v_1, v_2, v_3), i se asociază **coordonatele omogene**

  $$[v_1 : v_2 : v_3 : 0].$$

Vârfuri și direcții - rolul celei de-a 4 coordonate

- **Motivație:** Este necesar un cadru în care transformările să fie reprezentate în mod uniform și compunerea lor să fie ușor de descris: **folosind "coordonate omogene" și considerând 4 coordonate**.
- Coordonatele omogene verifică proprietatea fundamentală

$$[\alpha : \beta : \gamma : \delta] = [\lambda\alpha : \lambda\beta : \lambda\gamma : \lambda\delta], \quad \forall \lambda \neq 0, (\alpha, \beta, \gamma, \delta) \neq (0, 0, 0, 0).$$

  De exemplu, $[2 : 3 : -1 : 4] = [4 : 6 : -2 : 8] \neq [4 : -1 : 3 : 2]$.
- Unui **vârf** de coordonate (x_1, x_2, x_3), notate și (x, y, z) i se asociază **coordonatele omogene**

$$[x_1 : x_2 : x_3 : 1] \text{ (sau } [x : y : z : 1]).$$

- Unei **direcții** date de vectorul (v_1, v_2, v_3), i se asociază **coordonatele omogene**

$$[v_1 : v_2 : v_3 : 0].$$

- **Ilustrare:** codul sursă `04_C_2_coord_omogene.cpp`.

Exemple

- Punctul $(1,2,0)$ are coordonatele omogene

$$[1:2:0:1] = [3:6:0:3] = [-2:-4:0:-2] = [5:10:0:5] \neq [1:2:0:-1] \text{ etc.}$$

- Direcția $(2,2,0) \longrightarrow [2:2:0:0] = [4:4:0:0]$ etc.

Transformări – reprezentare matriceală

- Fie $f : \mathbb{R}^3 \to \mathbb{R}^3$ o aplicație afină arbitrară a lui $\mathbb{R}^3$, dată prin

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \overset{f}{\mapsto} \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}. \tag{5}$$

(a da o aplicație afină f revine la a da matricele $(a_{ij})_{i,j}$ și $(b_i)_i$.

Transformări – reprezentare matriceală

- Fie $f : \mathbb{R}^3 \to \mathbb{R}^3$ o aplicație afină arbitrară a lui $\mathbb{R}^3$, dată prin

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \overset{f}{\mapsto} \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}. \tag{5}$$

  (a da o aplicație afină f revine la a da matricele $(a_{ij})_{i,j}$ și $(b_i)_i$.
- Lui f îi corespunde o matrice 4×4

$$\mathcal{M}_f = \begin{pmatrix} a_{11} & a_{12} & a_{13} & b_1 \\ a_{21} & a_{22} & a_{23} & b_2 \\ a_{31} & a_{32} & a_{33} & b_3 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{6}$$

Transformări – reprezentare matriceală

- Fie $f : \mathbb{R}^3 \to \mathbb{R}^3$ o aplicație afină arbitrară a lui $\mathbb{R}^3$, dată prin
$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \overset{f}{\mapsto} \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}. \qquad (5)$$
(a da o aplicație afină f revine la a da matricele $(a_{ij})_{i,j}$ și $(b_i)_i$.
- Lui f îi corespunde o matrice 4×4
$$\mathcal{M}_f = \begin{pmatrix} a_{11} & a_{12} & a_{13} & b_1 \\ a_{21} & a_{22} & a_{23} & b_2 \\ a_{31} & a_{32} & a_{33} & b_3 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \qquad (6)$$
- **Principiu:** Dat un vârf / o direcție având coordonate omogene reprezentate de un vector coloană
$$\mathrm{v} = \begin{pmatrix} X_1 \\ X_2 \\ X_3 \\ X_0 \end{pmatrix},$$
aplicarea transformării f generează un nou vector coloană, și anume
$$M_f \cdot \mathrm{v}.$$

Exemple

- În momentul apelării funcției `glTranslate3f(`t_1, t_2, t_3`)`, OpenGL generează (și manevrează) matricea 4×4

$$M_{\mathrm{T_t}} = \begin{pmatrix} 1 & 0 & 0 & t_1 \\ 0 & 1 & 0 & t_2 \\ 0 & 0 & 1 & t_3 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Exemple

- În momentul apelării funcției `glTranslate3f(`t_1, t_2, t_3`)`, OpenGL generează (și manevrează) matricea 4×4

$$M_{\mathrm{T_t}} = \begin{pmatrix} 1 & 0 & 0 & t_1 \\ 0 & 1 & 0 & t_2 \\ 0 & 0 & 1 & t_3 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

- În momentul apelării funcției `glScale3f(`s_1, s_2, s_3`)`, OpenGL generează (și manevrează) matricea 4×4

$$M_{\sigma_{\mathrm{s}}} = \begin{pmatrix} s_1 & 0 & 0 & 0 \\ 0 & s_2 & 0 & 0 \\ 0 & 0 & s_3 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Exemple

Fie $f(x_1, x_2) = (2x_1 + 4x_2 + 5, 3x_1 - x_2 - 2)$.

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \longmapsto \begin{pmatrix} 2 & 4 & 0 \\ 3 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} 5 \\ -2 \\ 0 \end{pmatrix}$$

$$\Downarrow$$

$$\underset{(4\times 4)}{M_f} = \begin{pmatrix} 2 & 4 & 0 & 5 \\ 3 & -1 & 0 & -2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Completare

Pe lângă funcțiile standard din biblioteca gl(Core), transformările de modelare pot fi apelate în codul sursă indicând explicit matricea 4×4 asociată. Prin convenție, elementele sunt introduse pe coloane, de sus în jos, de la stânga la dreapta

$$\begin{pmatrix} 0 & 4 & 8 & 12 \\ 1 & 5 & 9 & 13 \\ 2 & 6 & 10 & 14 \\ 3 & 7 & 11 & 15 \end{pmatrix}.$$

Exemplu: codurile sursă `05_C_1_matrice.cpp` și `05_C_2_exemple_transformari.cpp`

Stiva de matrice

- Apelarea funcției glMatrixMode(GL_MODELVIEW) (explicație: OpenGL mai utilizează și matricele de proiecție) - v. glMatrixMode(GL_PROJECTION).

Stiva de matrice

- Apelarea funcției `glMatrixMode(GL_MODELVIEW)` (explicație: OpenGL mai utilizează și matricele de proiecție) - v. `glMatrixMode(GL_PROJECTION)`.
- Stivă de matrice

M_1 matrice curentă (standard, implicit este $\mathbb{I}_4$, identitatea)

M_2

$\vdots$

Stiva de matrice

- Apelarea funcției `glMatrixMode(GL_MODELVIEW)` (explicație: OpenGL mai utilizează și matricele de proiecție) - v. `glMatrixMode(GL_PROJECTION)`.
- Stivă de matrice

  M_1 matrice curentă (standard, implicit este $\mathbb{I}_4$, identitatea)

  M_2

  $\vdots$

- Stiva se modifică

Stiva de matrice

- Apelarea funcției `glMatrixMode(GL_MODELVIEW)` (explicație: OpenGL mai utilizează și matricele de proiecție) - v. `glMatrixMode(GL_PROJECTION)`.
- Stivă de matrice

  M_1 matrice curentă (standard, implicit este $\mathbb{I}_4$, identitatea)
  M_2
  $\vdots$

- Stiva se modifică
  - "pe verticală" - funcții specifice

Stiva de matrice

- Apelarea funcției `glMatrixMode(GL_MODELVIEW)` (explicație: OpenGL mai utilizează și matricele de proiecție) - v. `glMatrixMode(GL_PROJECTION)`.
- Stivă de matrice

  M_1 matrice curentă (standard, implicit este $\mathbb{I}_4$, identitatea)

  M_2

  $\vdots$

- Stiva se modifică
  - "pe verticală" - funcții specifice
  - "pe orizontală" (în vârf) - legată de compunerea transformărilor

Stiva de matrice

- Apelarea funcției `glMatrixMode(GL_MODELVIEW)` (explicație: OpenGL mai utilizează și matricele de proiecție) - v. `glMatrixMode(GL_PROJECTION)`.
- Stivă de matrice

  M_1 matrice curentă (standard, implicit este $\mathbb{I}_4$, identitatea)

  M_2

  $\vdots$

- Stiva se modifică
  - "pe verticală" - funcții specifice
  - "pe orizontală" (în vârf) - legată de compunerea transformărilor
- Exemplu: codul sursă `04_C_1_animatie.cpp`

Funcții pentru manevrarea stivei de matrice pe verticală

- `glPushMatrix( );`

  ↓ Mută toate matricele din stivă în jos cu o poziție (modificarea se face de jos în sus); în vârful stivei rămâne aceeași matrice (se va regăsi de două ori în stivă).

Funcții pentru manevrarea stivei de matrice pe verticală

- `glPushMatrix( );`

  ↓ Mută toate matricele din stivă în jos cu o poziție (modificarea se face de jos în sus); în vârful stivei rămâne aceeași matrice (se va regăsi de două ori în stivă).

- `glPopMatrix( );`

  ↑ Mută toate matricele din stivă în sus cu o poziție (modificarea se face de sus în jos); în particular matricea din vârful stivei dispare, iar cea de pe poziția 2 devine matrice curentă.

Observație

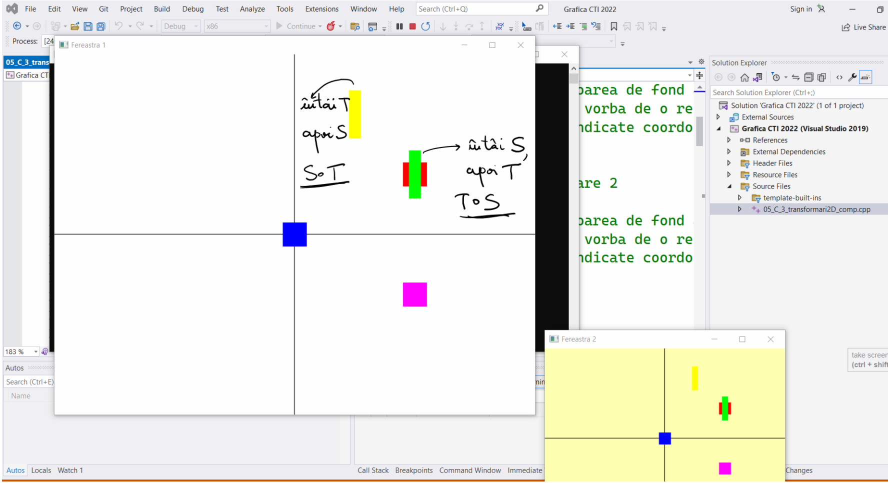
- **Observație importantă.** Scalarea și rotația date de `glScale` și `glRotate` au originea ca punct fix.

Observație

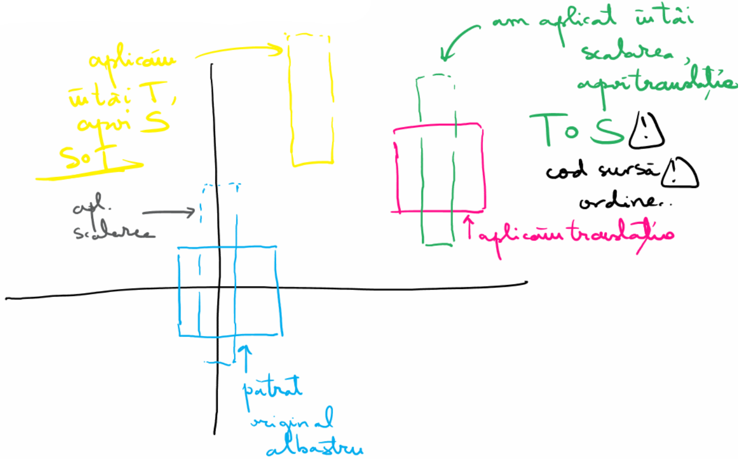
- **Observație importantă.** Scalarea și rotația date de `glScale` și `glRotate` au originea ca punct fix.
- De exemplu, dacă aplicăm `glScalef (0.5, 2.0, 1.0)` efectul este dat de regula

$$\begin{cases} x \mapsto 0.5x \\ y \mapsto 2y \\ z \mapsto z \end{cases} \quad \text{în particular} \quad \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \mapsto \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}.$$

Compunerea transformărilor - exemplu: codul sursă 05_C_3_transformari2D_comp.cpp



Compunerea transformărilor - exemplu: codul sursă 05_C_3_transformari2D_comp.cpp



Compunerea transformărilor

- Survine dacă în codul sursă sunt apelate succesiv mai multe transformări (fără a apela `glPushMatrix( );` `glPopMatrix( );`)

Compunerea transformărilor

- Survine dacă în codul sursă sunt apelate succesiv mai multe transformări (fără a apela `glPushMatrix( );` `glPopMatrix( );`)
- Exemplu (v. codul sursă `05_C_3_transformari2D_comp.cpp`)

Compunerea transformărilor

- Survine dacă în codul sursă sunt apelate succesiv mai multe transformări (fără a apela `glPushMatrix( );` `glPopMatrix( );`)
- Exemplu (v. codul sursă `05_C_3_transformari2D_comp.cpp`)
- Fapt esențial: dat un vârf / o direcție având coordonate omogene reprezentate de un vector coloană v, aplicarea unei transformări f cu matrice M_f generează un nou vector coloană, și anume $M_f \cdot \mathrm{v}$. De exemplu, dacă aplicăm scalarea de factor (2,4,6) vârfului $(1,2,1)$ obținem vârful $(2,8,6)$. Matriceal, cu convenția utilizării vectorilor coloană:

$$\begin{pmatrix} 2 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 6 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 2 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 2 \\ 8 \\ 6 \\ 1 \end{pmatrix}.$$

Compunerea transformărilor

- Survine dacă în codul sursă sunt apelate succesiv mai multe transformări (fără a apela `glPushMatrix( );` `glPopMatrix( );`)
- Exemplu (v. codul sursă `05_C_3_transformari2D_comp.cpp`)
- Fapt esențial: dat un vârf / o direcție având coordonate omogene reprezentate de un vector coloană v, aplicarea unei transformări f cu matrice M_f generează un nou vector coloană, și anume $M_f \cdot$ v. De exemplu, dacă aplicăm scalarea de factor (2,4,6) vârfului $(1, 2, 1)$ obținem vârful $(2, 8, 6)$. Matriceal, cu convenția utilizării vectorilor coloană:

$$\begin{pmatrix} 2 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 6 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 2 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 2 \\ 8 \\ 6 \\ 1 \end{pmatrix}.$$

- **Regulă:** În momentul în care este apelată o transformare de modelare (definită printr-o funcție standard sau printr-o matrice), **matricea curentă** (din vârful stivei) este înmulțită la **dreapta** cu matricea corespunzătoare noii transformări, devenind matrice curentă.

Compunerea transformărilor (continuare)

- **Practic:** Dacă în codul sursă au fost apelate transformările
$$T_1, T_2, T_3 \ldots, T_n$$
în această ordine, una după alta, urmate de o primitivă (vârfuri), ordinea în care acționează asupra primitivei este
$$T_n, T_{n-1}, \ldots, T_3, T_2, T_1.$$
De fapt: transformarea apelată este compunerea
$$T_1 \circ T_2 \circ T_3 \circ \ldots \circ T_n.$$

Compunerea transformărilor (continuare)

- **Practic:** Dacă în codul sursă au fost apelate transformările

$$T_1, T_2, T_3 \ldots, T_n$$

  în această ordine, una după alta, urmate de o primitivă (vârfuri), ordinea în care acționează asupra primitivei este

$$T_n, T_{n-1}, \ldots, T_3, T_2, T_1.$$

  De fapt: transformarea apelată este compunerea

$$T_1 \circ T_2 \circ T_3 \circ \ldots \circ T_n.$$

- Fie $M_{T_1}, M_{T_2}, \ldots, M_{T_n}$ matricele 4×4 asociate acestor transformări.

`glLoadIdentity ( );`	vf. stivei	$\mathbb{I}_4$
apelare T_1	...	M_{T_1}
apelare T_2	...	$M_{T_1} \cdot M_{T_2}$
	...	
apelare T_n	...	$M_{T_1} \cdot M_{T_2} \cdot \ldots \cdot M_{T_n}$

Compunerea transformărilor (continuare)

- **Practic:** Dacă în codul sursă au fost apelate transformările
$$T_1, T_2, T_3 \ldots, T_n$$
în această ordine, una după alta, urmate de o primitivă (vârfuri), ordinea în care acționează asupra primitivei este
$$T_n, T_{n-1}, \ldots, T_3, T_2, T_1.$$
De fapt: transformarea apelată este compunerea
$$T_1 \circ T_2 \circ T_3 \circ \ldots \circ T_n.$$

- Fie $M_{T_1}, M_{T_2}, \ldots, M_{T_n}$ matricele 4×4 asociate acestor transformări.

`glLoadIdentity ( );`	vf. stivei	$\mathbb{I}_4$
apelare T_1	...	M_{T_1}
apelare T_2	...	$M_{T_1} \cdot M_{T_2}$
	...	
apelare T_n	...	$M_{T_1} \cdot M_{T_2} \cdot \ldots \cdot M_{T_n}$

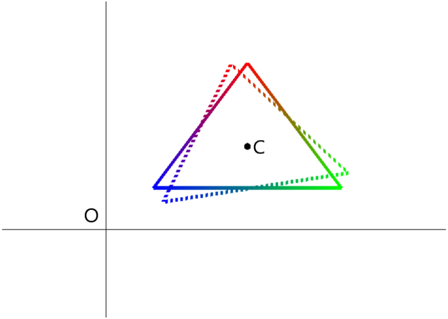
- La apelarea primitivei, v este transformat după regula
$$\mathrm{v} \mapsto (M_{T_1} \cdot M_{T_2} \cdot \ldots \cdot M_{T_n}) \cdot \mathrm{v}.$$

Exemplu - codul sursă 04_C_1_animatie.cpp

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity(); // in varful stivei este incarcata matricea identica
glTranslated(i, 200.0, 0.0); // generata o matrice 4x4 T, corespunzatoare translatiei, inm. cu id
glPushMatrix(); // in stiva: 2 matrice, ambele T
glRotated(j, 0.0, 0.0, 1.0); // generata o matrice 4x4 R, corespunzatoare rotatiei in stiva: varf: T*R, pozitia 2: T
glColor3f(1.0, 0.0, 0.0);
glRecti(-5, 30, 5, 40); // dreptunghiul rosu (asupra sa actioneaza rotatie, apoi translatie)
glPopMatrix(); // in varful stivei: T
glColor3f(0.0, 0.0, 1.0);
glRecti(-20, -12, 20, 12); // dreptunghiul albastru, asupra sa actioneaza translatie
glPopMatrix();
glutSwapBuffers();
```

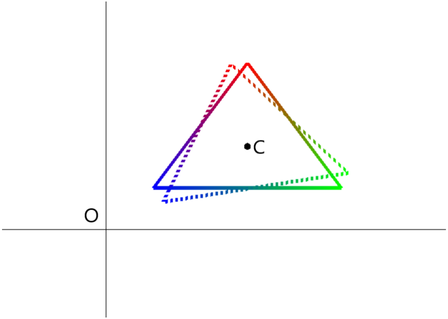
Rotații și scalări cu centrul diferit de originea O. Codul sursă `05_C_4_centru_rotatii_scalari.cpp`

Dacă dorim să efectuăm o rotație / scalare având centrul C diferit de O:



Rotații și scalări cu centrul diferit de originea O. Codul sursă `05_C_4_centru_rotatii_scalari.cpp`

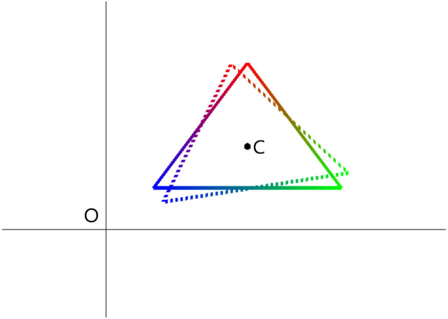
Dacă dorim să efectuăm o rotație / scalare având centrul C diferit de O:



- translatăm centrul C în O cu o translație de vector $-\overrightarrow{OC}$,

Rotații și scalări cu centrul diferit de originea O. Codul sursă `05_C_4_centru_rotatii_scalari.cpp`

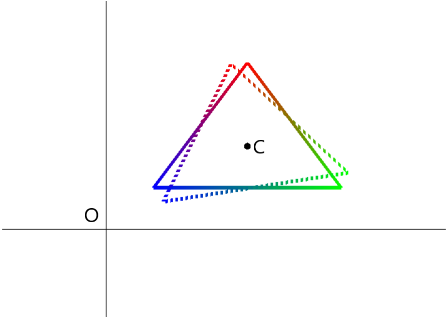
Dacă dorim să efectuăm o rotație / scalare având centrul C diferit de O:



- translatăm centrul C în O cu o translație de vector $-\overrightarrow{OC}$,
- aplicăm rotația / scalarea,

Rotații și scalări cu centrul diferit de originea O. Codul sursă `05_C_4_centru_rotatii_scalari.cpp`

Dacă dorim să efectuăm o rotație / scalare având centrul C diferit de O:



- translatăm centrul C în O cu o translație de vector $-\overrightarrow{OC}$,
- aplicăm rotația / scalarea,
- aplicăm translația de vector $\overrightarrow{OC}$, așa încât centrul (situat în O) să revină în poziția inițială.

Concluzii / De reținut!

- În OpenGL există funcții standard pentru o serie de transformări (translație, rotație, scalare).

Concluzii / De reținut!

- În OpenGL există funcții standard pentru o serie de transformări (translație, rotație, scalare).
- Pentru modelarea / implementarea transformărilor sunt utilizate 4 coordonate. Unei transformări i se asociază o matrice 4×4 care acționează prin înmulțire.

Concluzii / De reținut!

- În OpenGL există funcții standard pentru o serie de transformări (translație, rotație, scalare).
- Pentru modelarea / implementarea transformărilor sunt utilizate 4 coordonate. Unei transformări i se asociază o matrice 4×4 care acționează prin înmulțire.
- Compunerea transformărilor corespunde înmulțirii matricelor. Atenție la ordinea în care sunt indicate transformările în codul sursă!

Concluzii / De reținut!

- În OpenGL există funcții standard pentru o serie de transformări (translație, rotație, scalare).
- Pentru modelarea / implementarea transformărilor sunt utilizate 4 coordonate. Unei transformări i se asociază o matrice 4×4 care acționează prin înmulțire.
- Compunerea transformărilor corespunde înmulțirii matricelor. Atenție la ordinea în care sunt indicate transformările în codul sursă!
- În OpenGL sunt utilizate stive de matrice pentru a manevra în mod convenabil compunerea transformărilor.