

0.1 Izračun kotov rotacije izometrične projekcije

Izometrična projekcija je vrsta **pravokotne projekcije**, kjer so osnovni trije enotski vektorji tega prostora:

- $\mathbf{i} = [1, 0, 0]^T$,
- $\mathbf{j} = [0, 1, 0]^T$,
- $\mathbf{k} = [0, 0, 1]^T$,

najprej zarotirani in na to projicirani v **ravnino XZ** tako, da so njihove projekcije **enako dolgu**. Od tod izvira tudi izraz, saj uporabljamo isto **metriko** za vse tri osi. To je poseben primer **aksonometrične projekcije** in jo poimenujemo **pravokotna aksonometrična projekcija**.

0.1.1 Rotacija vektorjev in projekcija

Najprej zarotiramo osnovne vektorje tako, da bo pogled na vektorje iz ustreznega kota. Uporabimo **dve rotaciji**:

1. **rotacijo okoli osi Z** za kot α in
2. **rotacijo okoli osi X** za kot β .

Po rotaciji nato **projiciramo vektorje v ravnino XZ**.

Rotacija okoli osi Z za kot α :

$$\mathbf{R}_Z(\alpha) = \begin{bmatrix} \cos \alpha & -\sin \alpha & 0 \\ \sin \alpha & \cos \alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Rotacija okoli osi X za kot β :

$$\mathbf{R}_X(\beta) = \begin{bmatrix} 0 & \cos \beta & -\sin \beta \\ 0 & \sin \beta & \cos \beta \end{bmatrix}$$

Tako lahko združimo obe rotaciji v:

$$\mathbf{R}_{XZ} = \mathbf{R}_X(\beta) \cdot \mathbf{R}_Z(\alpha) = \begin{bmatrix} \cos \alpha & -\sin \alpha & 0 \\ \cos \beta \cdot \sin \alpha & \cos \beta \cdot \cos \alpha & -\sin \beta \\ \sin \beta \cdot \sin \alpha & \sin \beta \cdot \cos \alpha & \cos \beta \end{bmatrix}$$

Nato izvedemo še projekcijo na ravnino XZ. To storimo z množenjem s projekcijsko matriko:

$$\mathbf{P}_{XZ} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Torej skupna transformacija je:

$$\mathbf{T} = \mathbf{P}_{XZ} \cdot \mathbf{R}_{XZ}$$

in dobimo končno transformacijsko matriko:

$$\mathbf{T} = \begin{bmatrix} \cos \alpha & -\sin \alpha & 0 \\ 0 & 0 & 0 \\ \sin \beta \cdot \sin \alpha & \sin \beta \cdot \cos \alpha & \cos \beta \end{bmatrix}$$

Pogoj za izometrično projekcijo je, da so projekcije prvotnih osnovnih enotskih vektorjev enako dolge. Zato najprej izračunamo projekcije enotskih vektorjev:

- $\mathbf{i}' = \mathbf{T} \cdot \mathbf{i} \rightarrow$ prva stolpčna vektorja matrike \mathbf{T}
- $\mathbf{j}' = \mathbf{T} \cdot \mathbf{j} \rightarrow$ drugi stolpec
- $\mathbf{k}' = \mathbf{T} \cdot \mathbf{k} \rightarrow$ tretji stolpec

Te vektorje označimo kot:

$$\mathbf{i}' = \begin{bmatrix} \cos \alpha \\ 0 \\ \sin \beta \cdot \sin \alpha \end{bmatrix}, \quad \mathbf{j}' = \begin{bmatrix} -\sin \alpha \\ 0 \\ \sin \beta \cdot \cos \alpha \end{bmatrix}, \quad \mathbf{k}' = \begin{bmatrix} 0 \\ 0 \\ \cos \beta \end{bmatrix}$$

Njihove dolžine morajo biti enako dolge:

$$\|\mathbf{i}'\| = \|\mathbf{j}'\| = \|\mathbf{k}'\|$$

Izračun dolžin:

- $\|\mathbf{i}'\|^2 = \cos^2 \alpha + \sin^2 \alpha \cdot \sin^2 \beta$
- $\|\mathbf{j}'\|^2 = \sin^2 \alpha + \cos^2 \alpha \cdot \sin^2 \beta$
- $\|\mathbf{k}'\|^2 = \cos^2 \beta$

Rešimo sistem enačb:

$$\begin{aligned}\cos^2 \alpha + \sin^2 \alpha \cdot \sin^2 \beta &= \cos^2 \beta \\ \sin^2 \alpha + \cos^2 \alpha \cdot \sin^2 \beta &= \cos^2 \beta\end{aligned}$$

sledi:

$$\cos^2 \alpha = \sin^2 \alpha = \frac{1}{2} \Rightarrow \alpha = 45^\circ$$

in

$$\cos^2 \beta = \frac{1}{2} + \frac{1}{2} \cdot \sin^2 \beta = \frac{1}{2}(1 + \sin^2 \beta) \Rightarrow \cos^2 \beta = \frac{2}{3} \Rightarrow \cos \beta = \sqrt{\frac{2}{3}} \approx 0.816 \Rightarrow \beta \approx \arccos(\sqrt{2/3}) \approx 35.26^\circ$$

Ker gre za reševanje periodičnih funkcij je rešitev več. Za naš primer bomo izbrali rešitev izometrične projekcije če uporabimo:

- **rotacijo okoli Z:** $\alpha = 45^\circ$,
- **rotacijo okoli X:** $\beta \approx -35.26^\circ$ (rotacija prostora "naprej" - stran od projekcijske ravnine) in
- **projekcijo:** na ravnino **XZ**.

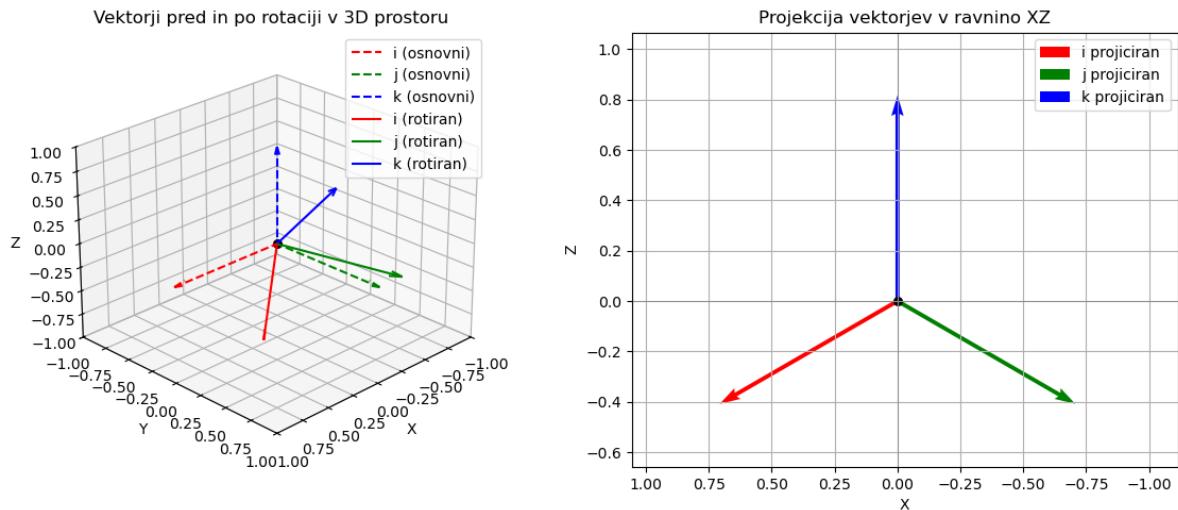
Končna transformacijska matrika **T**:

$$\mathbf{T} \approx \begin{bmatrix} \cos 45^\circ & -\sin 45^\circ & 0 \\ 0 & 0 & 0 \\ \sin 35.26^\circ \cdot \sin 45^\circ & \sin 35.26^\circ \cdot \cos 45^\circ & \cos 35.26^\circ \end{bmatrix} = \begin{bmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & 0 \\ 0 & 0 & 0 \\ \frac{1}{\sqrt{3}} \cdot \frac{\sqrt{2}}{2} & \frac{1}{\sqrt{3}} \cdot \frac{\sqrt{2}}{2} & \sqrt{\frac{2}{3}} \end{bmatrix}$$

Situacijo lahko razumemo tudi iz drugega zornega kota, na primer kam moramo premakniti trenutno projekcijo ravnino Π_{XY} z normalnim vektorjem $\mathbf{e}_y = [0, 0, 1]^T$, da dobimo enak rezultat. Zanimajo nas koordinate novega normalnega vektorja \mathbf{n}_{POV} te projekcijske ravnine Π_{POV} . Ker imamo izračunano rotacijsko matriko **R** je to enostavno inverzna rotacija:

$$\mathbf{n}_{POV} = \mathbf{R}^{-1} \mathbf{e}_y = \frac{1}{\sqrt{3}} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \approx \begin{bmatrix} 0.5774 \\ 0.5774 \\ 0.5774 \end{bmatrix} \quad (1)$$

S tem smo dobili rezultat, ki je prikazan na sl. 1.



Slika 1: Rotacija prostora tako, da dobimo izometrično projekcijo.

Če želite nekoliko eksperimentirati z rotacijami in projekcijami lahko preskusite :

Listing 1: izracun izometricne projekcije.

```

1 import numpy as np
2 import matplotlib.pyplot as plt
3 from mpl_toolkits.mplot3d import Axes3D
4
5 # --- 1. Definicija kotov ---
6 alpha_deg = 45
7 beta_deg = -35.264 # žprabilno arccos(sqrt(2/3))
8 alpha = np.radians(alpha_deg)
9 beta = np.radians(beta_deg)
10
11 # --- 2. Rotacijski matriki ---
12 Rz = np.array([
13     [np.cos(alpha), -np.sin(alpha), 0],
14     [np.sin(alpha), np.cos(alpha), 0],
15     [0, 0, 1]
16 ])
17
18 Rx = np.array([
19     [1, 0, 0],
20     [0, np.cos(beta), -np.sin(beta)],
21     [0, np.sin(beta), np.cos(beta)]
22 ])

```

```

1   # --- 3. Projekcijska matrika (XZ projekcija) ---
2   P_XZ = np.array([
3       [1, 0, 0],
4       [0, 0, 0],
5       [0, 0, 1]
6   ])
7
8   # --- 4. Skupna transformacija ---
9   T = P_XZ @ Rx @ Rz
10
11  # --- 5. Osnovni vektorji ---
12  i = np.array([1, 0, 0])
13  j = np.array([0, 1, 0])
14  k = np.array([0, 0, 1])
15
16  # --- 6. Rotirani vektorji ---
17  i_rot = Rx @ Rz @ i
18  j_rot = Rx @ Rz @ j
19  k_rot = Rx @ Rz @ k
20
21  # --- 7. Projekcije rotiranih vektorjev ---
22  i_proj = T @ i
23  j_proj = T @ j
24  k_proj = T @ k
25
26  # --- 8. Prikaz ---
27  fig = plt.figure(figsize=(12,6))
28
29  # Levi graf: 3D pogled pred in po rotaciji
30  ax3d = fig.add_subplot(1, 2, 1, projection='3d')
31  ax3d.set_title("Vektorji pred in po rotaciji v 3D prostoru")
32
33  # Črtkani - osnovni
34  ax3d.quiver(0, 0, 0, *i, color='r', linestyle='dashed', label='i (osnovni)', arrow_length_ratio=0.1)
35  ax3d.quiver(0, 0, 0, *j, color='g', linestyle='dashed', label='j (osnovni)', arrow_length_ratio=0.1)
36  ax3d.quiver(0, 0, 0, *k, color='b', linestyle='dashed', label='k (osnovni)', arrow_length_ratio=0.1)

```

```

1 # Polne - rotirani
2 ax3d.quiver(0, 0, 0, *i_rot, color='r', label='i (rotiran)',
3             arrow_length_ratio=0.1)
4 ax3d.quiver(0, 0, 0, *j_rot, color='g', label='j (rotiran)',
5             arrow_length_ratio=0.1)
6 ax3d.quiver(0, 0, 0, *k_rot, color='b', label='k (rotiran)',
7             arrow_length_ratio=0.1)
8
9 ax3d.scatter(0, 0, 0, color='black', s=30)
10 ax3d.set_xlim([-1, 1])
11 ax3d.set_ylim([-1, 1])
12 ax3d.set_zlim([-1, 1])
13 ax3d.set_xlabel('X')
14 ax3d.set_ylabel('Y')
15 ax3d.set_zlabel('Z')
16 ax3d.view_init(elev=25, azim=45)
17 ax3d.legend()
18
19 # Desni graf: projekcija v XZ ravnino
20 ax2d = fig.add_subplot(1, 2, 2)
21 ax2d.set_title("Projekcija vektorjev v ravnino XZ")
22
23 # Projekcije
24 ax2d.quiver(0, 0, i_proj[0], i_proj[2], angles='xy', scale_units='xy',
25             scale=1, color='r', label='i projiciran')
26 ax2d.quiver(0, 0, j_proj[0], j_proj[2], angles='xy', scale_units='xy',
27             scale=1, color='g', label='j projiciran')
28 ax2d.quiver(0, 0, k_proj[0], k_proj[2], angles='xy', scale_units='xy',
29             scale=1, color='b', label='k projiciran')
30
31 # Osi in žmreja
32 ax2d.axhline(0, color='gray', lw=0.5)
33 ax2d.axvline(0, color='gray', lw=0.5)
34 ax2d.scatter(0, 0, color='black', s=30)
35 ax2d.set_xlabel('X')
36 ax2d.set_ylabel('Z')
37 ax2d.axis('equal')
38 ax2d.grid(True)
39 ax2d.set_xlim(-1, 1.5)
40 ax2d.set_ylim(-0.5, 1.5)
41 ax2d.legend()
42
43 plt.tight_layout()
44 plt.show()

```