Vizualizácia priamej kinematickej úlohy

zadanie č. 1

Imrich Dzvoňár
2. Bc

12.4. 2025 RK

Zadanie:

- Zariadenie tohto typu slúži na evakuáciu osôb z výškových budov. Je preto nesmierne dôležité pomôcť požiarnikom pri realizácii výpočtov pri polohovaní takéhoto zariadenia.

- Majme svetový súradnicový systém $x_0y_0z_0$ podľa obrázka (Obr. 1). V koncovom súradnicovom systéme $x_ky_kz_k$ sa nachádza bod P, ktorý predstavuje aj koncový bod plošiny resp. kontaktný bod plošiny s budovou. Štyri hydraulické pohony zabezpečujú: rotáciu základne (ϕ_1), nakláňanie rebríka (ϕ_2), vysúvanie rebríka (L_5) a nakláňanie plošiny (ϕ_3). Konfigurácia takéhoto automobilovej plošiny pri prevoze nasledovná:
  $L_5 = L_5min$, $\phi_1 = 0$, $\phi_2 = 90°$ a $\phi_3 = -180°$.

<u>Úlohy:</u>

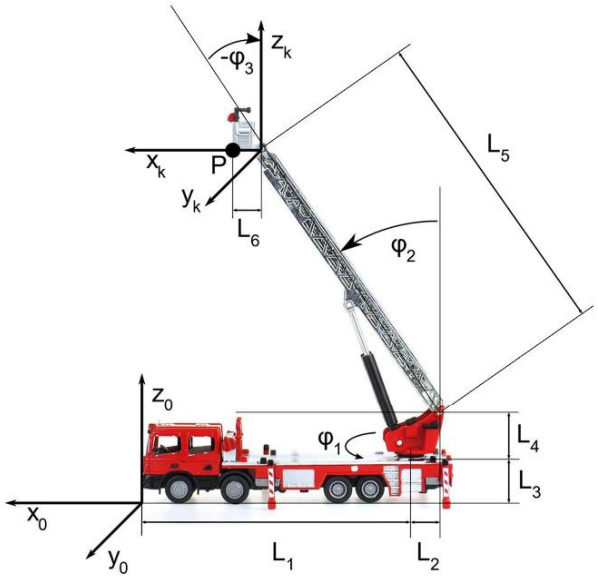
1. Pomocou homogénnej transformácie odvoďte vzťah pre výpočet transformačnej matice T0k medzi svetovým súradnicovým systémom x0y0z0 a súradnicovým systémom $x_ky_kz_k$.

2. Pomocou odvodenej transformačnej matice T0k z úlohy 1.) vyriešte priamu kinematickú úlohu pre koncový bod P požiarnického rebríka s plošinou a uveďte vzťah pre výpočet polohy koncového bodu P vo svetovom súradnicovom systéme, ak sú dané: L1, L2, L3, L4, L5, L6, ϕ1, ϕ2 a ϕ3. Jednotlivé homogénne transformačné matice vypíšte a dosaďte do nich správne argumenty (dĺžky, uhly).

3. Vykreslite zjednodušený mechanizmus a vykreslite aj jednotlivé pomocné súradnicové systémy od nultého až po k-ty. (x-červenou farbou, y-zelenou, z-modrou) (obr. 2).

4. Vykreslite obálky pracovného priestoru v bázovej rovine mechanizmu X_0Z_0, tiež aj v rovine X_0Y_0, ak platí :
   L1 = 10 [m], L2 = 1 [m], L3 = 1 [m], L4 = 1 [m], L5 = <10, 40> [m], L6 = 1[m],
   $\phi1 \in <-\infty, \infty>$, $\phi2 \in <0^o, 90^o>$, $\phi3 \in <0^o, -180^o>$.

Vypracovanie:



Obr. 1 Požiarna výšková automobilová plošina

Na odvodenie transformačnej matice T_0 použijeme postupné násobenie homogénnych transformačných matíc, ktoré opisujú jednotlivé pohyby (rotácie a translácie) medzi súradnicovými systémami jednotlivých článkov.
Začíname od svetového súradnicového systému a každú transformáciu reprezentujeme pomocou homogénnej matice – najprv aplikujeme rotáciu, potom transláciu – a tento proces opakujeme pre všetky kĺby až po koncový člen.

Transformačná matica medzi svetovým a koncovým systémom má tvar:

$$T_0 = T_{01} * T_{12} * T_{23} * T_{34} \ldots\ldots T_{(k-1)k}$$

Výsledná matica T_0 potom slúži na vyjadrenie polohy a orientácie koncového bodu plošiny, reprezentovaného homogénnym vektorom P_0, voči svetovému súradnicovému systému ako:

$$P_0 = T_0 * P_k$$

kde P_k je poloha bodu P v koncovom súradnicovom systéme a P_0 je jeho poloha vo svetovom systéme.

Pomocou odvodeného vzťahu pre transformačnú maticu T_0 som vyriešil priamu kinematickú úlohu pre koncový bod P požiarnického rebríka. Jednotlivé pohyby medzi článkami systému som vyjadril ako homogénne transformačné matice – rotácie a translácie – a postupne som ich medzi sebou násobil podľa poradia pohybov.

Zoznam transformačných matíc (v tvare 4×4):

1. Posun základne rebríka smerom hore o výšku L4:

$$\mathbf{Tz = trans\_z(L4)}$$

2. Rotácia základne rebríka o uhol φ_1 okolo osi Z:

$$\mathbf{Trz = rot\_z\_deg(\varphi 1)}$$

3. Rotácia rebríka o uhol φ_2 okolo osi Y a následný výsuv o dĺžku L5:

$$\mathbf{Tryx = rot\_y\_deg(\varphi 2) \cdot trans\_x(}L2 + L3 + L4 + \mathbf{L5)}$$

4. Naklonenie plošiny o uhol φ_3 okolo osi Y a posun o dĺžku L6:

$$\mathbf{Tryx = rot\_y\_deg(\varphi 3) \cdot trans\_x(L6)}$$

Výsledná transformačná matica:

$$\mathbf{T_0 = Tz \cdot Trz \cdot Tryx \cdot Tryx}$$

Alebo zápis použitý na cvičení:

$$\mathbf{T_0} = \mathrm{Tz}\,(L1)\cdot \mathrm{Rz}\,(\varphi 1)\cdot \mathrm{Ry}\,(\varphi 2)\cdot \mathrm{Tz}\,(L2 + L3 + L4 + L5)\cdot \mathrm{Ry}\,(\varphi 3)\cdot \mathrm{Tz}\,(L6)$$

Po dosadení $\mathbf{L_1, L_2, L_3,}$ L_4, L_5, L_6, $\boldsymbol{\varphi_1, \varphi_2, \varphi_3}$ do matíc **trans_z, trans_y, trans_x, rot_z_deg, rot_y_deg**

$$T_z = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad T_x = \begin{pmatrix} 1 & 0 & 0 & dx \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad R_z = \begin{pmatrix} \sin(\varphi) & -\cos(\varphi) & 0 & 0 \\ \cos(\varphi) & \sin(\varphi) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad R_y = \begin{pmatrix} \cos(\varphi) & 0 & \sin(\varphi) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\varphi) & 0 & \cos(\varphi) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Obr. 2 Translačných a rotačných matíc zo zdrojového kodú

$$T_{0k} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & L4 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \sin(\phi 1) & -\cos(\phi 1) & 0 & 0 \\ \cos(\phi 1) & \sin(\phi 1) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \cos(\phi 2) & 0 & \sin(\phi 2) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\phi 2) & 0 & \cos(\phi 2) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & L2+L3+L4+L5 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\cdot \begin{pmatrix} \cos(\phi 3) & 0 & \sin(\phi 3) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\phi 3) & 0 & \cos(\phi 3) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & L6 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Obr. 3 výpočet $\mathbf{T_0}$, a dosadenie do matíc

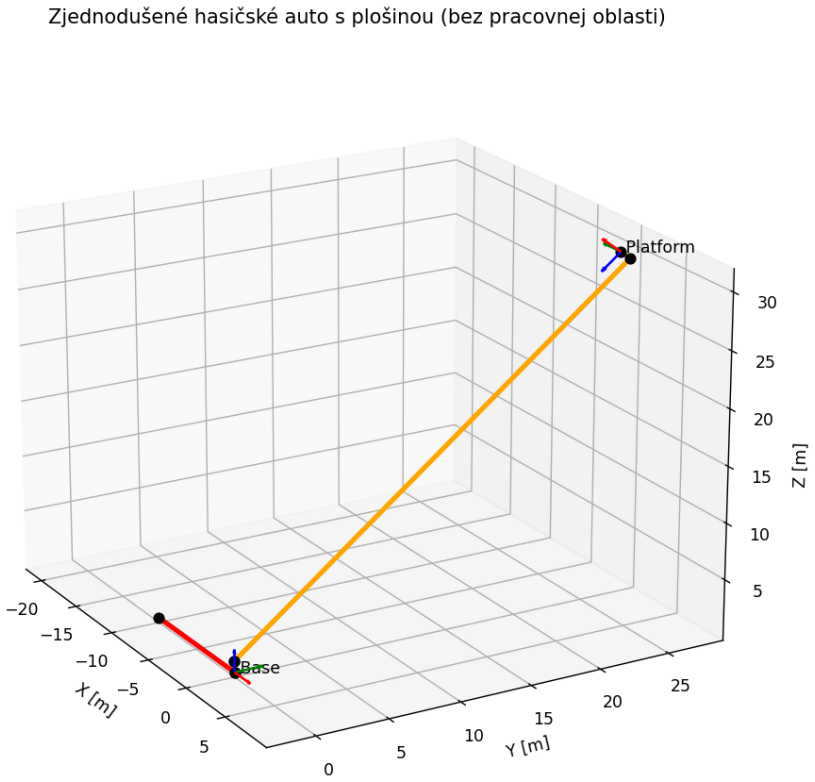
Matica T_0 vyjadruje polohu a orientáciu koncového súradnicového systému plošiny v svetovom súradnicovom systéme.

Bod P, ktorý sa nachádza na konci plošiny, som v jej lokálnom súradnicovom systéme definoval ako homogénny vektor:

$$p_k = [0, 0, 0, 1]^T$$

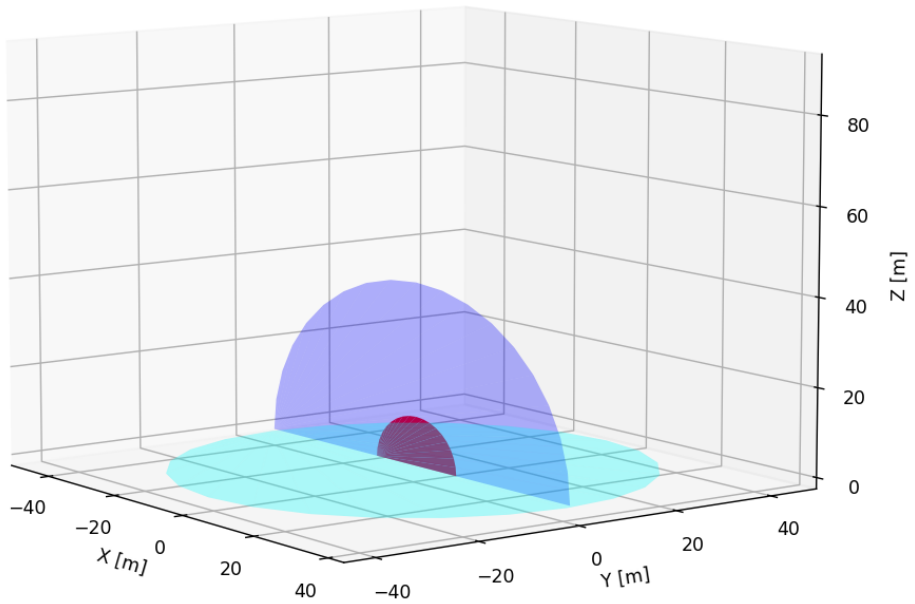
Jeho poloha vo svetovom súradnicovom systéme je daná vzťahom:

$$p_0 = T_0 \cdot p_k$$

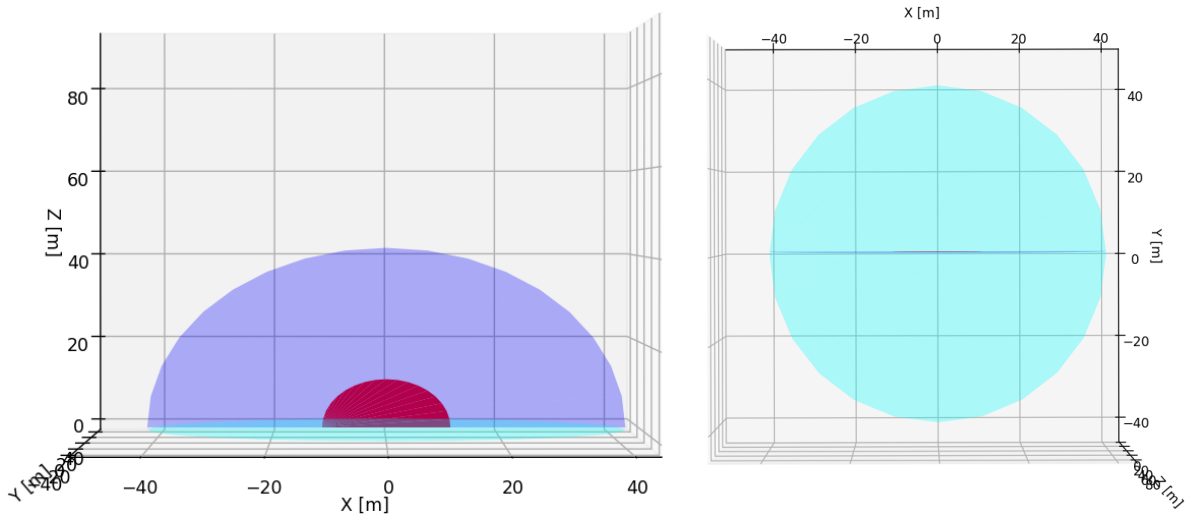


Obr. 4 Vykreslenie zjednodušeného mechanizmu

Pracovným priestor (XY a XZ) požiarnej výškovej plošiny



Obr. 5 Vykreslenie pracovného priestoru (XY a XZ) požiarnej výškovej plošiny (červený úsek samozrejme do XZ pracovnej obálky NEPATRÍ)



Obr. 6 Vykreslenie pracovného priestoru XZ (fialová zóna) a XY(modrá zóna)

Toto zadanie som vypracoval v Pythone použil som knižnicu matplotlib na vykreslovanie. Ďalšiu knižnicu ktorú som použil je NumPy, ktorá mi uľahčila prácu s maticami na úroveň MatLabu, avšak dala mi výhodu zjednodušiť vykreslované objekty vďaka čomu som mohol obrázky zobrazovať v reálnom čase a nemusel som si ich ukladať ako súbory. Toto mi urýchlilo prácu na zadaní.

```python
def trans_z(d):
    # 4x4 posun v Z o d
    return [
        [1, 0, 0, 0],
        [0, 1, 0, 0],
        [0, 0, 1, d],
        [0, 0, 0, 1]
    ]
```

```python
def trans_x(dx):
    # 4x4 posun v X o dx
    return [
        [1, 0, 0, dx],
        [0, 1, 0, 0],
        [0, 0, 1, 0],
        [0, 0, 0, 1]
    ]
```

```python
def rot_z_deg(fi):
    sin_val = math.sin(math.radians(fi))
    cos_val = math.cos(math.radians(fi))
    return [
        [sin_val, -cos_val, 0, 0],
        [cos_val,  sin_val, 0, 0],
        [0,        0,       1, 0],
        [0,        0,       0, 1]
    ]
```

```python
def rot_y_deg(fi):
    sin_val = math.sin(math.radians(fi))
    cos_val = math.cos(math.radians(fi))
    return [
        [cos_val,  0, sin_val, 0],
        [0,        1, 0,       0],
        [-sin_val, 0, cos_val, 0],
        [0,        0, 0,       1]
    ]
```

Vytvoril som si funkcie transformačných a rotačných matíc ktoré som volal pri výpočte T_0,

Ďalej som si vytvoril funkciu na násobenie matíc. Síce mal som importovanú knižnicu ktorá na to mala príkaz, ale keďže našou úlohou bolo naučiť sa pracovať s maticami (poliami) tak som si vytvoril vlastný.

```python
def Nasobenie_matic(A, B):
    # Maticové násobenie (A a B sú zoznamy zoznamov)
    m, n, p = len(A), len(B), len(B[0])
    C = [[0 for _ in range(p)] for _ in range(m)]
    for i in range(m):
        for j in range(p):
            for k in range(n):
                C[i][j] += A[i][k] * B[k][j]
    return C
```

Nasledujúci kód vytvára reťaz homogénnych transformačných matíc: T0 (identita) sa posunie (T0_1), potom sa otočí o Z (T1_2) a následne sa otočí a posunie (T2_3, T3_4). Kumulatívne sa získavajú body P0, p1, p2, p3, p4, ktoré sa vykreslia v 3D ako spojitá čiara.

```
#Výpočet mechanizmu rebríka (podľa MATLAB-logiky) ----------
T0 = eye4()
T0_1 = trans_z(L1)
T1_2 = rot_z_deg(phi1_deg)
T2_3 = Nasobenie_matic(rot_y_deg(phi2_deg), trans_x(L2))
T3_4 = Nasobenie_matic(rot_y_deg(phi3_deg), trans_x(L3))

T0_1_cum = T0_1
T0_2_cum = Nasobenie_matic(T0_1_cum, T1_2)
T0_3_cum = Nasobenie_matic(T0_2_cum, T2_3)
T0_4_cum = Nasobenie_matic(T0_3_cum, T3_4)

P0 = [0, 0, 0]  # Zem
p1 = get_xyz(T0_1_cum)
p2 = get_xyz(T0_2_cum)
p3 = get_xyz(T0_3_cum)
p4 = get_xyz(T0_4_cum)

# ---------- Vizuálne vykreslenie mechanizmu rebríka ----------
fig = plt.figure()
ax = fig.add_subplot(projection='3d')

# Spojíme body do jednej čiary (mechanizmus rebríka)
x_vals = [P0[0], p1[0], p2[0], p3[0], p4[0]]
y_vals = [P0[1], p1[1], p2[1], p3[1], p4[1]]
z_vals = [P0[2], p1[2], p2[2], p3[2], p4[2]]
ax.plot(x_vals, y_vals, z_vals,
        c='orange', linewidth=3, marker='o',
        markerfacecolor='k', markeredgecolor='k')
```

V riadkoch

```
fig = plt.figure()
ax = fig.add_subplot(projection='3d')
```

Vykresľujem 3D priestor, je to z knižnice Matplotlib.

Kód generuje obálku pracovného priestoru tak, že cyklom prechádza všetky povolené hodnoty uhlov a dĺžok a pre každý výpočet pomocou násobenia transformačných matíc získa koncovú pozíciu (x, y, z). Následne sa tieto body premietnu do určenej roviny (napr. do roviny XY) a algoritmom (konvexným obalom) určíme najvzdialenejších bodov uzavretý polygón. Tento polygón potom predstavuje hranicu pracovného priestoru, ktorá je vykreslená ako spojitá čiara alebo vyplnená plocha.

```
ws_points = []
for phi1 in phi1_vals:
    for phi2 in phi2_vals:
        for phi3 in phi3_vals:
            for L2_val in L2_vals:
                point = compute_platform_position(phi1, phi2, phi3, L1, L2_val, L3)
                ws_points.append(point)
ws_points = np.array(ws_points)

# Výpočet konvexnej obálky v XY rovine
proj_xy = ws_points[:, :2]
hull_xy = convex_hull_2d(proj_xy.tolist())
hull_xy = np.array(hull_xy)
z_const = np.mean(ws_points[:, 1])
triangles_xy = triangulate_polygon(hull_xy.tolist())
triangles_xy = np.array(triangles_xy)
```

```
ax.plot_trisurf(hull_xy[:, 0], hull_xy[:, 1],
                np.full_like(hull_xy[:, 0], z_const),
                triangles=triangles_xy, color='cyan', alpha=0.3, shade=False)
```

Návod na kompiláciu môjho kódu:

PC musí mať nainštalovaný Python tak isto aj spomínané knižnice (NumPy, Matplotlib).
V terminály stačí skompilovať môj súbor (DzvonarRobZadanie1.py), potom ho spustiť. To je všetko súbor by sa mal zapnúť a fungovať.

Záver(zhodnotenie):

V tomto zadaní sme riešili kinematickú úlohu, kde sme mali vypočítať koncový bod plošiny, resp. kontaktný bod plošiny s budovou. Kde sme museli násobiť translačné a aj rotačné homogénne matice v správnom poradí. Od prvého bodu postupne cez každý kĺb až po posledný koncový bod.

Pomocou homogénnej transformácie som odvodil vzťah pre výpočet transformačnej matice, ktorú som nazval možno trochu netradične T_0. Zoskupenie matíc, ktoré som násobil, som spravil tiež nezvyčajne (tým chcem povedať - trochu inak, ako sme robili na cvičeniach) ale správne poradie som dodržal.

Priamu kinematickú úlohu som vyriešil správne, keďže sa systém, alebo teda výšková plošina, správa predvídateľne, tak ako som chcel.

Vykreslil som zjednodušený mechanizmus, aj keď nie je najprehľadnejší a to z dôvodu pomeru medzi L5 a napríklad L6, kde som mal rozdiel dĺžok o jeden rád. Toto ale neovplyvňuje funkčnosť simulácie.

Vykreslenie obálky pracovného priestoru som zvládol, aj napriek výzvam pri obálke XZ, kedy sa mi nedarilo vykresliť miesto, kam sa rameno nevie dostať (viď. Obrázok 6 červená zóna). A tak, čo som nevedel nenakresliť, som nakreslil inak.

Pri tomto zadaní som sa naučil lepšie pracovať s maticami v oblasti programovania (teda poľami), porozumel som lepšie tomuto spôsobu zisťovania konečného bodu manipulátorov. Získal som skúsenosti z novými knižnicami v Pythone.

Zoznam použitej literatúry:
https://numpy.org/doc/

https://matplotlib.org/stable/index.html

Zadanie som vypracoval sám. Čestne prehlasujem, že som ho neskopíroval a nikomu inému neposkytol. Nech mi je Isaac Asimov svedkom.

Imrich Dzvoňár