Kiberfizikai Gyártórendszerek

Párhuzamos Kinematika Modellezése

1. A matematikai modellezés:

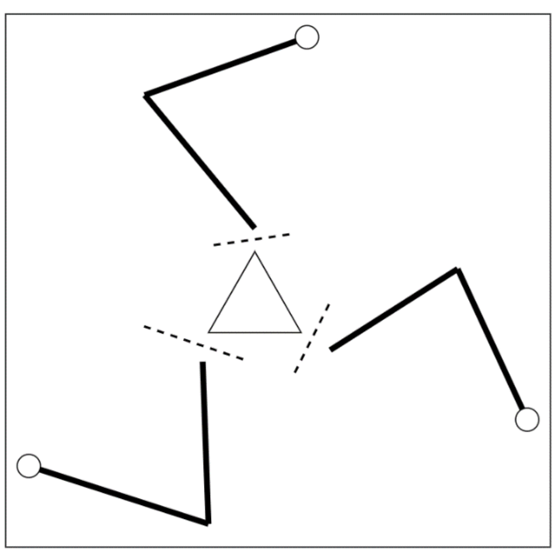
A robotot a TCP (tool center point) 3 koordinátája szerint vezéreljük a síkban. Ez a 3 koordináta leírja a központban lévő tripod helyzetét a síkban.

$$TCP = \begin{bmatrix} x \\ y \\ \varphi \end{bmatrix} = \begin{bmatrix} [mm] \\ [mm] \\ [rad] \end{bmatrix} \tag{1}$$

Ezekből a koordinátákból szükséges visszafejteni a hozzájuk tartozó motor szöghelyzeteket inverz kinematika segítségével. Ezeket a szöghelyzeteket a motorokra kivezérelve kapnánk meg a valóságban a kívánt TCP pozíciót.

$$\begin{bmatrix} x \\ y \\ \varphi \end{bmatrix} \rightarrow \begin{bmatrix} \alpha_{motor1} \\ \alpha_{motor2} \\ \alpha_{motor3} \end{bmatrix} = \begin{bmatrix} [rad] \\ [rad] \\ [rad] \end{bmatrix} \tag{2}$$

A megoldás egyik legfontosabb eleme, hogy a tripod sarkainál szét kell választanunk a robotot. A 3 kart külön kezelhetjük, és mindegyikre egyenként számíthatunk inverz kinematikát. Így a párhuzamos kinematika felbomlik 3db egymástól független soros kinematikává.



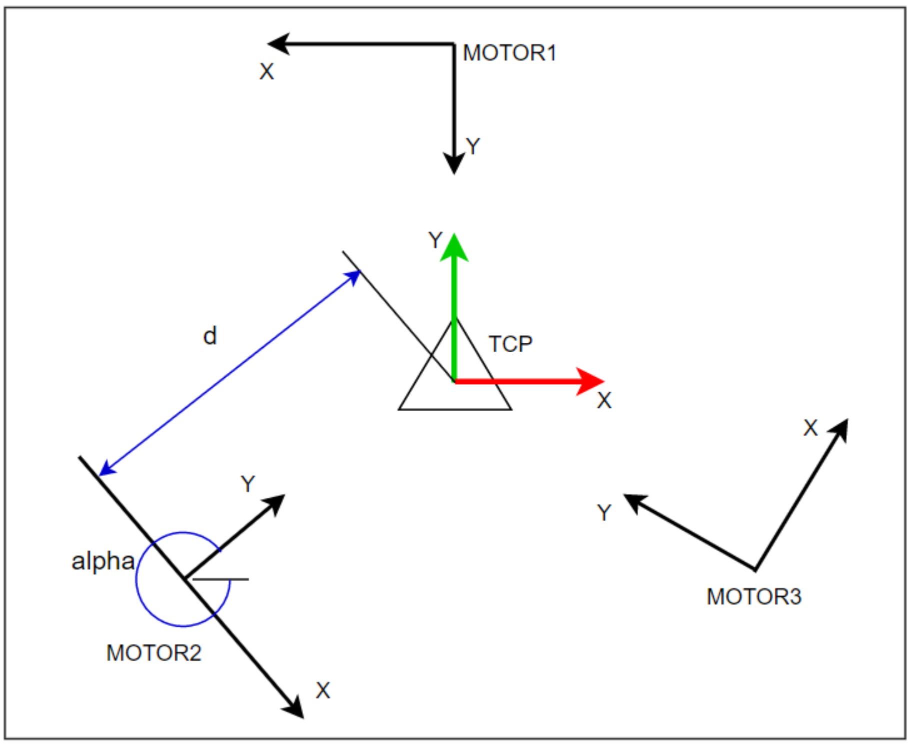
Így mindegyik karnak külön TCP-je lesz, a tripod 1-1 sarka:

$$\begin{bmatrix} x + \frac{2}{3} \cdot a \cdot \cos{(\varphi + 90°)} \\ y + \frac{2}{3} \cdot a \cdot \sin{(\varphi + 90°)} \end{bmatrix} ; \begin{bmatrix} x + \frac{2}{3} \cdot a \cdot \cos{(\varphi + 210°)} \\ y + \frac{2}{3} \cdot a \cdot \sin{(\varphi + 210°)} \end{bmatrix} ; \begin{bmatrix} x + \frac{2}{3} \cdot a \cdot \cos{(\varphi + 330°)} \\ y + \frac{2}{3} \cdot a \cdot \sin{(\varphi + 330°)} \end{bmatrix}$$

(3)(4)(5)

Ahol a a tripod oldalhossza.

Az inverz kinematikához minden motorhoz saját koordináta rendszert rendelünk, és a világ koordinátarendszerben lévő tripod vertexeket ezekbe transzformáljuk.



Ehhez 2 dimenziós homogén transzformációs mátrixra van szükség. Levezetve a fenti koordinátarendszerek alapján:

$$\boldsymbol{T} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & d \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & d \\ 0 & 0 & 1 \end{bmatrix}$$

(6)

Ahol d a motor távolsága a világ koordinátarendszertől, és α a kar koordinátarendszer világ koordinátarendszerrel bezárt szöge. Így kiszámítható a tripod vertex a kar koordinátarendszerében.

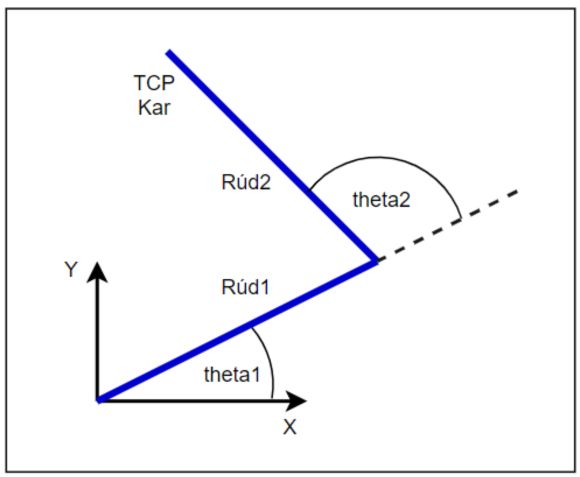
Például véve a MOTOR2-t:

$$TCP_{KAR2} = \begin{bmatrix} \cos(300°) & -\sin(300°) & 0 \\ \sin(300°) & \cos(300°) & 50 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x + \frac{2}{3} \cdot a \cdot \cos(\varphi + 210°) \\ y + \frac{2}{3} \cdot a \cdot \sin(\varphi + 210°) \\ 1 \end{bmatrix} =$$

$$= \begin{bmatrix} \cos(300°) \cdot \left(x + \frac{2}{3} \cdot a \cdot \cos(\varphi + 210°)\right) - \sin(300°) \cdot y + \frac{2}{3} \cdot a \cdot \sin(\varphi + 210°) \\ \sin(300°) \cdot x + \frac{2}{3} \cdot a \cdot \cos(\varphi + 210°) + \cos(300°) \cdot y + \frac{2}{3} \cdot a \cdot \sin(\varphi + 210°) + 50 \\ 1 \end{bmatrix}$$

(7)

A kar koordinátarendszerében így már megoldható az inverz kinematika:



A rudak hosszainak és a forgástengelyek ismeretében a *tinyik* Python könyvtár függvényei visszaadnak egy megoldást $\begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix}$-re. Könnyen belátható, hogy 2 megoldása van a feladatnak, úgy kell definiálni a geometriai kényszereket, hogy mindig megfelelő legyen a visszaadott első (pozitív) megoldás, ne legyen ugrás 2 pozíció között. Ezeket a θ_1 szögeket kellene a motorokra kivezérelni.

A motorok és a tripod vertexek koordinátái ismertek a világ koordinátarendszerben, viszont a passzív csuklók koordinátái nem. Először meg kell határozni a kar koordinátarendszerében, aztán át kell transzformálni.

$$passzív\ csukló_{KAR} = \begin{bmatrix} Rúd1 \cdot \cos(\theta_1) \\ Rúd1 \cdot \sin(\theta_1) \end{bmatrix}$$

(8)

Homogén transzformációs mátrix-al vissza transzformálhatóak ezek a koordináták is, példának véve a MOTOR2-t:

$$passzív\ csukló_{VILÁG} = \begin{bmatrix} \cos(60°) & -\sin(60°) & 0 \\ \sin(60°) & \cos(60°) & -50 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} Rúd1 \cdot \cos(\theta_1) \\ Rúd1 \cdot \sin(\theta_1) \\ 1 \end{bmatrix} =$$

$$= \begin{bmatrix} \cos(60°) \cdot Rúd1 \cdot \cos(\theta_1) - \sin(60°) \cdot Rúd1 \cdot \sin(\theta_1) \\ \sin(60°) \cdot Rúd1 \cdot \sin(\theta_1) + \cos(60°) \cdot Rúd1 \cdot \sin(\theta_1) - 50 \\ 1 \end{bmatrix} \tag{9}$$

Mindhárom karra kiszámolva az összes lényes pont elérhető a világ koordinátarendszerben, és ismert minden kivezérelendő szög.

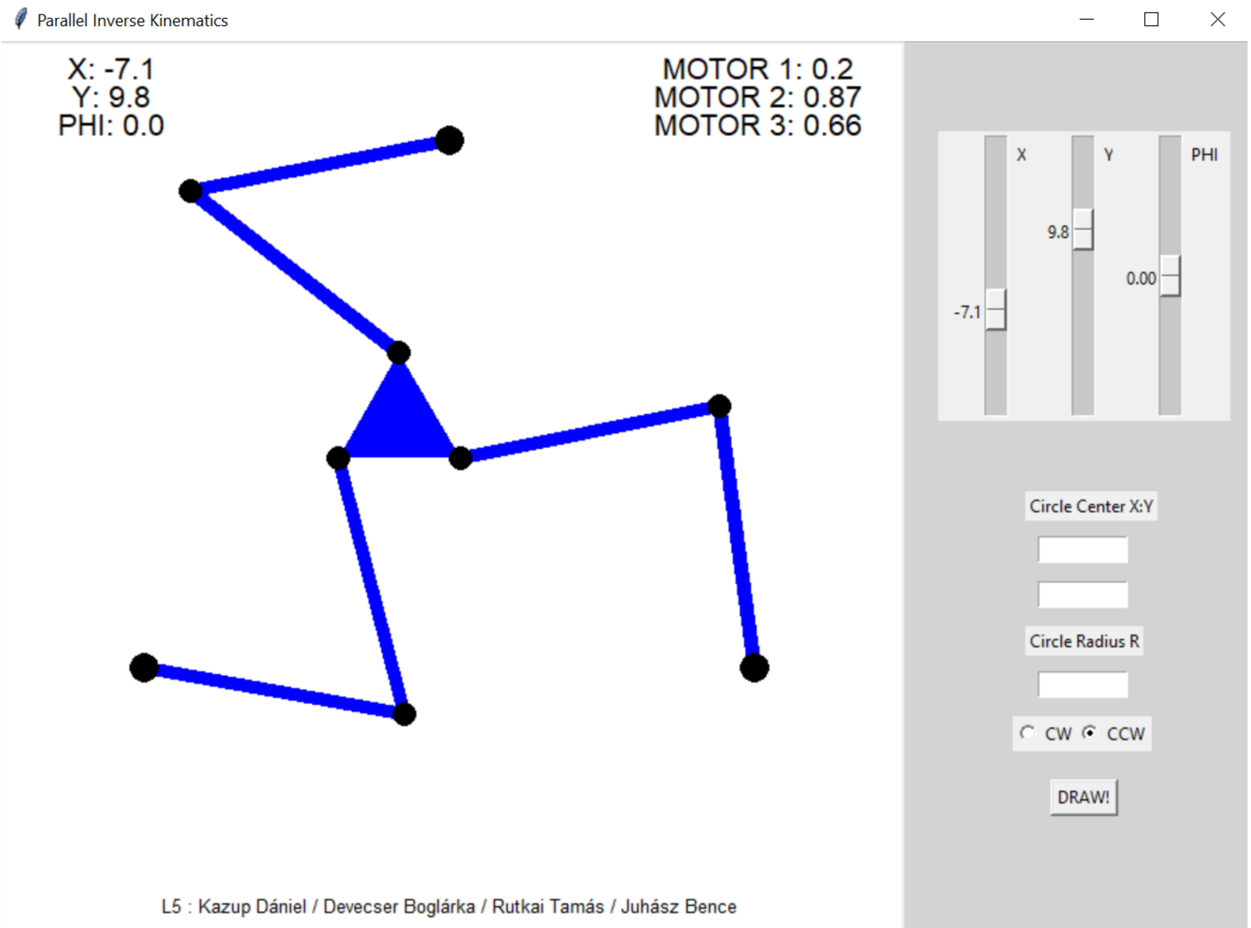
2. Megoldás Pythonban:

A grafikus felülethez és a számításokhoz az alábbi könyvtárak kerültek felhasználásra:

- Tkinter - Grafikus megjelenítéshez és vezérlő widget-ekhez
- Pygame - Külső vezérléshez, például játékkonzollal
- Numpy - Mátrix műveletekhez
- Tinyik - Inverz kinematika számításhoz

A feladat objektum-orientáltan lett megvalósítva, az alábbi osztályokkal:

- Robot kar osztály - szükséges paraméterek saját frame-ben, homogén transzformációs mátrixok, konverziós függvények
- Manipulátor osztály - 3 példány a robot karból, szükséges geometriai kényszerek, TCP vezérlő függvények, paraméterek világ frame-ben
- Grafika osztály - A GUI elkészítése, widgetek és alakzatok létrehozása
- Smartpad osztály - Játékkonzoltól jövő bemenetek feldolgozása, és visszaadása végrehajtásra. (opcionális)
- Kör osztály - A program képes kört interpolálni, a megadott paraméterekkel kirajzolásra küldött kör egy ilyen objektumként jön létre az összes paraméterével.



A felület megjeleníti a motor szöghelyzeteit, a TCP koordinátáit. A jobb oldalon lévő panelon a csúszkákkal állíthatjuk a TCP-t az adott limitek között.

Kontrolleres vezérléssel joystick-el állíthatjuk a tripod helyzetét és szögelfordulást, gombokkal pedig lépésekben mozgathatunk.

Kör interpoláció is lehetséges, meg kell adnunk a kör középpont koordinátáit és a kör sugarát, valamint ki kell választani a körüljárás irányát (clockwise, counter clockwise). Ez létrehoz egy kör objektumot, ami adott felbontással diszkrét pontokra bontja a kört. Ezeken a pontokon van keresztülvezérelve a robot.