Universitatea
Transilvania
din Brașov

Facultatea de Inginerie Electrică și Știința Calculatoarelor
Departamentul de Automatică și Tehnologia Informației
Programul de studii: Robotica

Proiect la disciplina

Bazele Cinematicii Robotilor Industrialii

Brat Robotic cu Doua Cuple de Rotatie

Autor: Student Mihnea – Dimitrie DOLOIU
Programul de studii: Robotică
Grupa 4LF801A

Coordonator: Prof. univ. dr. ing. Claudiu POZNA

2022

Table of Contents

1. INTRODUCERE

1.1 DEFINITIE

Robotul este un sistem mecatronic, proiectat cu scopul de a indeplinii sarcinii complicate, cum ar fi manipularea de obiecte in spatiu, urmarirea de traiectorii (tracking), etc. Robotii vin in multe forme si varietatii, fiecare conceput pentru a indeplinii un set de sarcinii specifice domeniilor in care sunt folositi.

1.2 ABSTRACT

In aceasta lucrare se va prezenta proiectarea unui **brat robotic cu doua cuple de rotatie** si diferite metode, pentru controlul acestuia. Bratul este proiectat cu scopul de a fi capabil sa deplaseze orice obiect, cu marimi corespunzatoare, si sa urmareasca traiectorii intr-un spatiu de lucru cu o arie de 1 m^2 (un patrat cu latura de 1 m) .Proiectarea si controlul acestui brat robotic au fost realizate folosind aplicatiile CATIAv5 si Matlab.

2. MODELUL CINEMATIC AL ROBOTULUI

2.1 STRUCTURA

Robotul este alcatuit din doua cuple rotative denumite q_1 si q_2 , ambele avand aceeas axa de rotatie (in cazul nostru axa fiind considerata axa **Z**). Ambele cuple sunt capabile sa performe o rotatie completa de $360°$ (2π rad). Lungimile bratelor robotului l_1 si l_2 , au aceeas lungime de 0.5 m, in total, robotul cu bratul intins drept, avand lungimea de 1 m. Originea axelor va fi considerata ca fiind aflata la baza robotului.

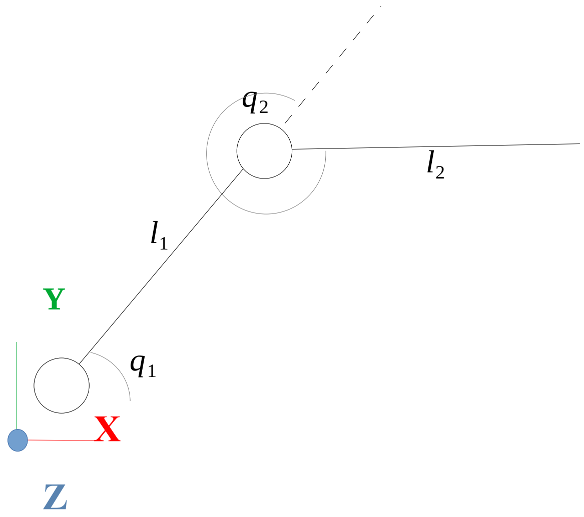


Fig 2.1 Structura bratului robotic

Fiecare segment de brat, au masele m_1 si m_2, si momentele de inertie I_1 si I_2, acestea fiind folosite in modelarea dinamica a robotolui (Cap. 3), dar ele tot facand parte din structura.

Pe baza acestei structuri se vor deriva ecuatiile care descriu comportamentul bratului robotic.

2.2 MODELUL GEOMETRIC DIRECT

MGD-ul este folosit pentru determinarea pozitiei si orientarii efectorului in spatiul cartezian, folosind pozitiile si orientarilor din spatiul articular.

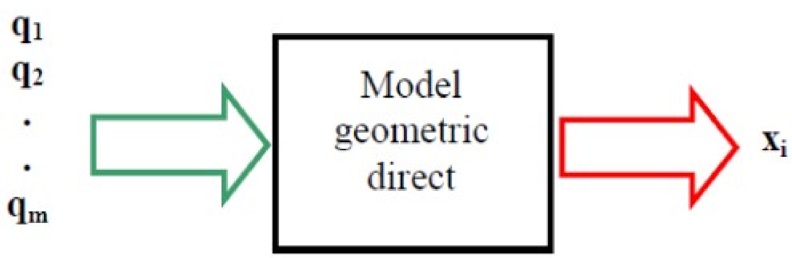


Fig 2.2.1 Modelul Geometric Direct (MGD)

Acesta este format, in cazul de fata, dintr-o singura matrice de transformare T_e^0, care permite calculul valorilor coordonatelor **X, Y, Z** in functie de valorile q_1 si q_1 .

Matricile de transformare de la un reper la urmatorul, pentru bratul robotic proiectat sunt urmatoarele:

	Transformarea	**Matricea**
$T_1^0 =$	Rotatie in jurul axei **Z** : $R_z(q_1)$ Fara translatie: $t(0,0,0)$	$\begin{pmatrix} \cos(q_1) & -\sin(q_1) & 0 & 0 \\ \sin(q_1) & \cos(q_1) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$
$T_2^1 =$	Rotatie in jurul axei **Z** : $R_z(q_2)$ Translatie: $t(l_1, 0,0)$	$\begin{pmatrix} \cos(q_2) & -\sin(q_2) & 0 & l_1 \\ \sin(q_2) & \cos(q_2) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$
$T_e^2 =$	Fara rotatie: I_3 Translatie: $t(l_2, 0,0)$	$\begin{pmatrix} 1 & 0 & 0 & l_2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$

Se va ajunge la transformarea matriciala T_e^0 prin inmultirea matricilor din tabelul precedent:

$$T_e^0 = T_0^1 \cdot T_2^1 \cdot T_e^2 \; (2.0)$$

Unde :

$$T_2^0 = T_1^0 \cdot T_2^1 = \begin{pmatrix} \cos(q_1+q_2) & -\sin(q_1+q_2) & 0 & l_1\cos(q_1) \\ \sin(q_1+q_2) & \cos(q_1+q_2) & 0 & l_1\sin(q_1) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (2.1)$$

$$T_e^0 = T_2^0 \cdot T_e^2 = \begin{pmatrix} \cos(q_1+q_2) & -\sin(q_1+q_2) & 0 & l_1\cos(q_1)+l_2\cos(q_1+q_2) \\ \sin(q_1+q_2) & \cos(q_1+q_2) & 0 & l_1\sin(q_1)+l_2\sin(q_1+q_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (2.2)$$

Unde, punctele intermediare sunt :

O = (1...3,4)T_1^0 ----- Centrul de greutate a cuplei de rotatie 1
S = (1...3,4)T_2^0 ----- Centrul de greutate a cuplei de rotatie 2
P = (1...3,4)T_e^0 ----- Centrul de greutate a efectorului

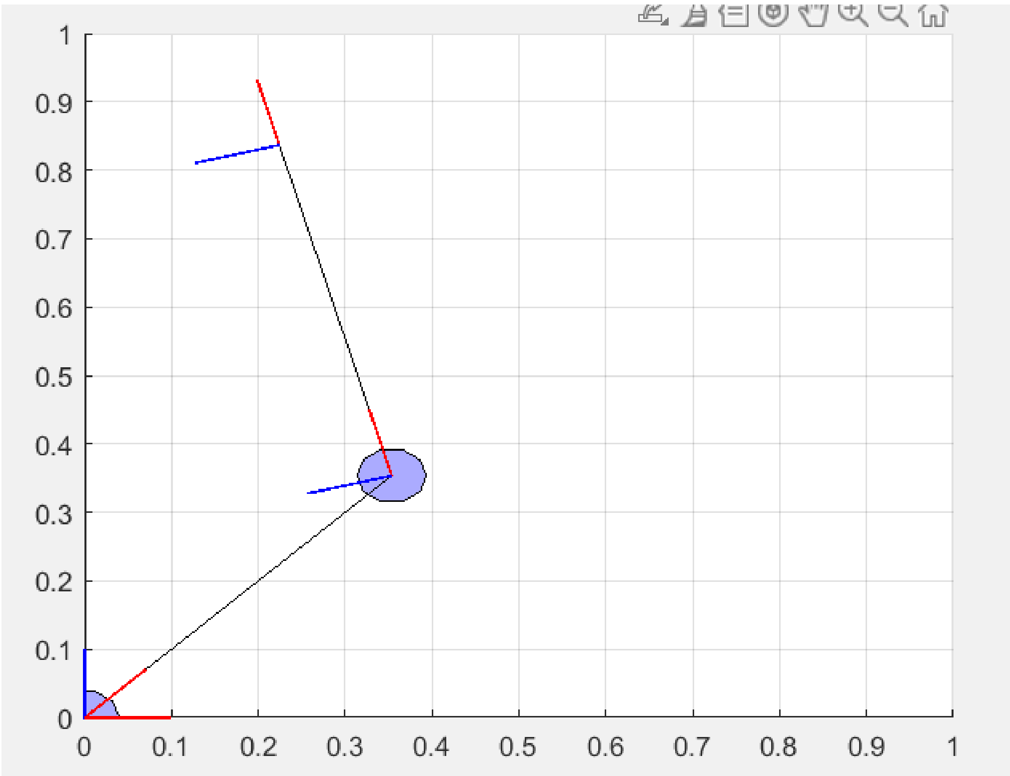


Fig 2.2.2 Pozitia si Orientarea end-efectorului pentru q(pi/4, pi/3)

Ecuatiile rezultante pentru cordoonatele carteziene vor fi :

X = $l_1 * \cos(q_1) + l_2 * \cos(q_1+q_2)$ (3.1)

Y = $l_1 * \sin(q_1) + l_2 * \sin(q_1+q_2)$ (3.2)

Z = 0 (3.3)

2.3 MODELUL CINEMATIC DIRECT

MCD permite determinarea vitezelor liniare si unghiulare a dizpozitivului efector, in functie de vitezele articulatiilor. MCD mai este folosit si pentru determinarea traiectoriei pe care bratul efector o va urma, vectorul vitezei liniare fiind folosit pentru a urma traiectoria.

Aceasta transformare se face prin intermediul unei matricii de transformare numita **Jacobian** :

$$J(q) = \begin{pmatrix} -l1\sin(q_1)-l2\sin(q_1+q_2) & -l2\sin(q_1+q_2) \\ l1\cos(q_1)+l2\cos(q_1+q_2) & l2\cos(q_1+q_2) \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 1 \end{pmatrix} \quad (4.0)$$

Jacobianul robotului proiectat

Jacobianul poate fi impartit in doua matricii de rang mai mic care descriu vitezele liniare (J_v) si cele unghiulare (J_w) :

$$J_v = \begin{pmatrix} -l1\sin(q_1)-l2\sin(q_1+q_2) & -l2\sin(q_1+q_2) \\ l1\cos(q_1)+l2\cos(q_1+q_2) & l2\cos(q_1+q_2) \\ 0 & 0 \end{pmatrix} \quad (4.1)$$

$$J_w = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 1 \end{pmatrix} \quad (4.2)$$

Jacobianurile vitezelor liniare si unghiulare in forma separata

2.4 ECUATII DE MISCARE

Traiectoria urmarita de efector este data de vectorii vitezelor liniare in punctele respective de pe traiectorie.

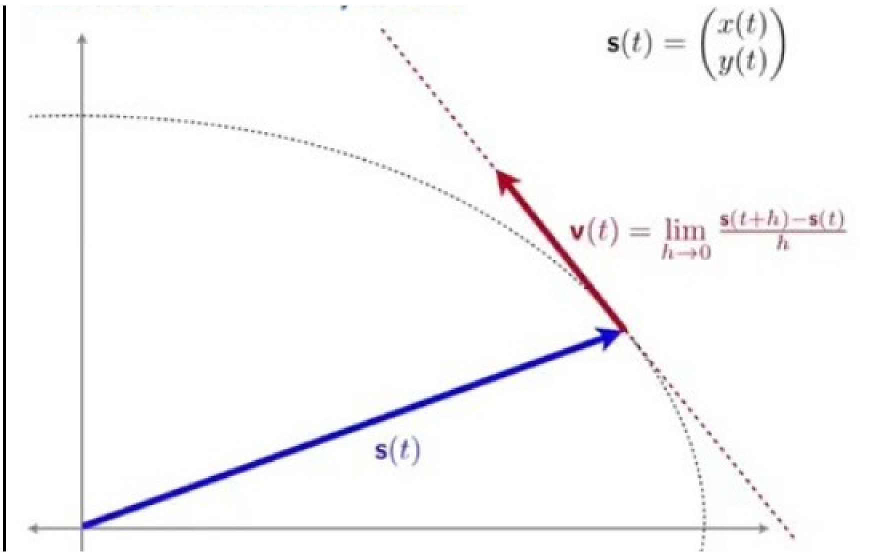


Fig 2.4 Reprezentarea vectorului vitezelor

Se doreste sa se genereze ecuatii, care sa respecte urmatoarele conditii :

- deplasarea initiala este nula : q(0)
- deplasarea finala este stiuta : q(t_f) = q
- viteza initiala este nula : $\dot{q}(0)$ = 0
- viteza finala este nula : $\dot{q}(t_f)$ = 0

Folosind conditiile de mai sus, se vor prezenta doua metode pentru deplasarea efectorului :

2.4.1 LEGE DE MISCARE SUB FORMA POLINOMIALA

$$q(t)=\frac{-2q}{t_f^3}t^3+\frac{3q}{t_f^2}t^2 \quad (5.1)$$

Legea de miscare

Legea de miscare de forma polinomiala va folosi un polinom de gradul 3, deoarece se asuma ca valoarea acceleratiei va varia liniar in timp.

Viteza si acceleratia se vor determina prin derivarea legii de miscare :

$$\dot{q}(t)=\frac{-6q}{t_f^3}t^2+\frac{6q}{t_f^2}t \quad (5.2)$$

Ecuatia vitezei

$$\ddot{q}(t)=\frac{-12q}{t_f^3}t+\frac{6q}{t_f^2} \quad (5.3)$$

Ecuatia acceleratiei

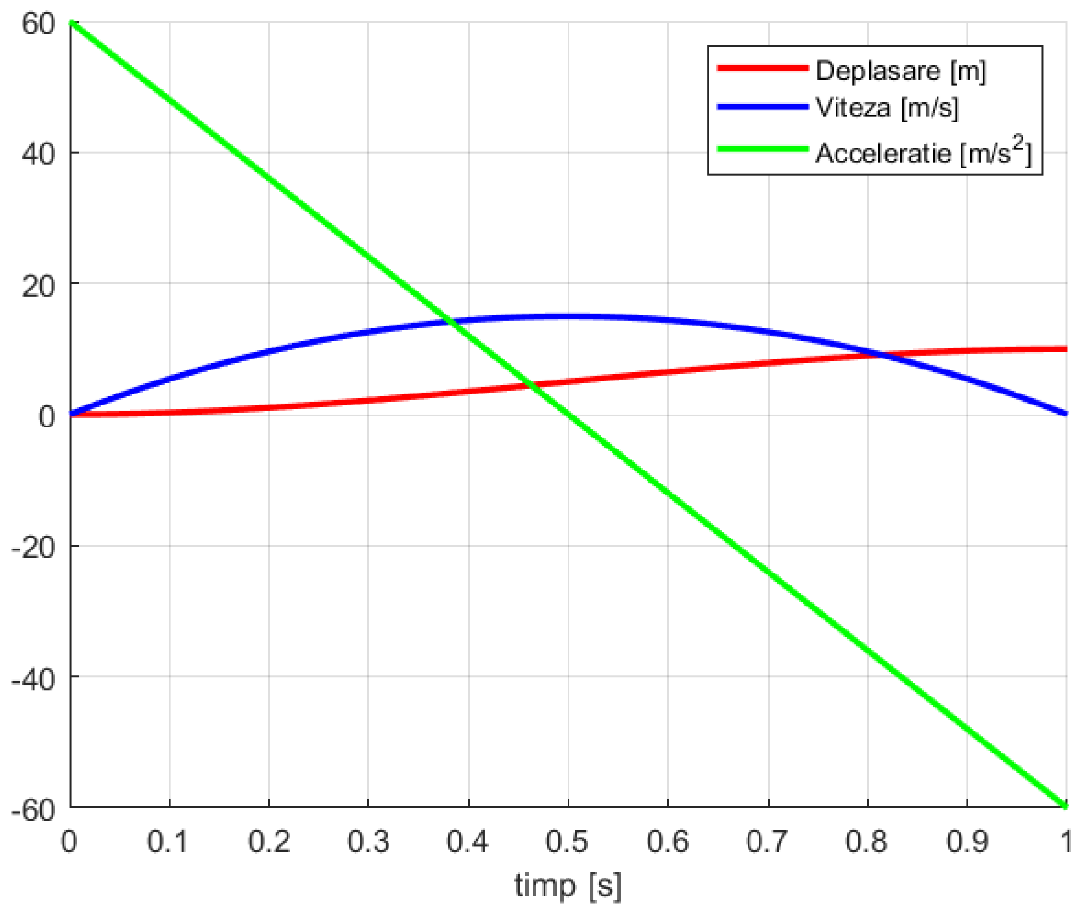


Fig 2.4.1 Graficele ecuatilor de miscare pentru q = 10

Ecuatiile pot fi scrise si sub forma matriciala :

$$\begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \\ \frac{3}{t_f^2} & \frac{-3}{t_f^2} \\ \frac{-2}{t_f^3} & \frac{2}{t_f^3} \end{pmatrix} \cdot \begin{pmatrix} q_f \\ q_i \end{pmatrix} \quad (5.4)$$

Matricia
coeficientilor

$$\begin{pmatrix} q(t) \\ \dot{q}(t) \\ \ddot{q}(t) \end{pmatrix} = \begin{pmatrix} 1 & t & t^2 & t^3 \\ 0 & 1 & 2t & 3t^2 \\ 0 & 0 & 2 & 6t \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} \quad (5.5)$$

Matrcia ecuatiilor polinomiale

Combinand cele doua ecuatii de mai sus, si inlocuind vectorul vitezelor articulare din MCD, v-om obtine urmatoarea relatie:

$$V = J(q) \cdot P(t) \cdot C(t_f) \cdot q \quad (5.6)$$

Unde :

- P este matricia ecuatiilor polinomiale (5.4)
- C este matricia coeficientilor (5.5)
- J este Jacobianul

2.4.2 LEGEA DE MISCARE FOLOSIND SEMNAL TRAPEZOID

Aceasta metoda va face controlul miscarii prin viteza. Semnalul vitezei, va lua o forma trapezoida.

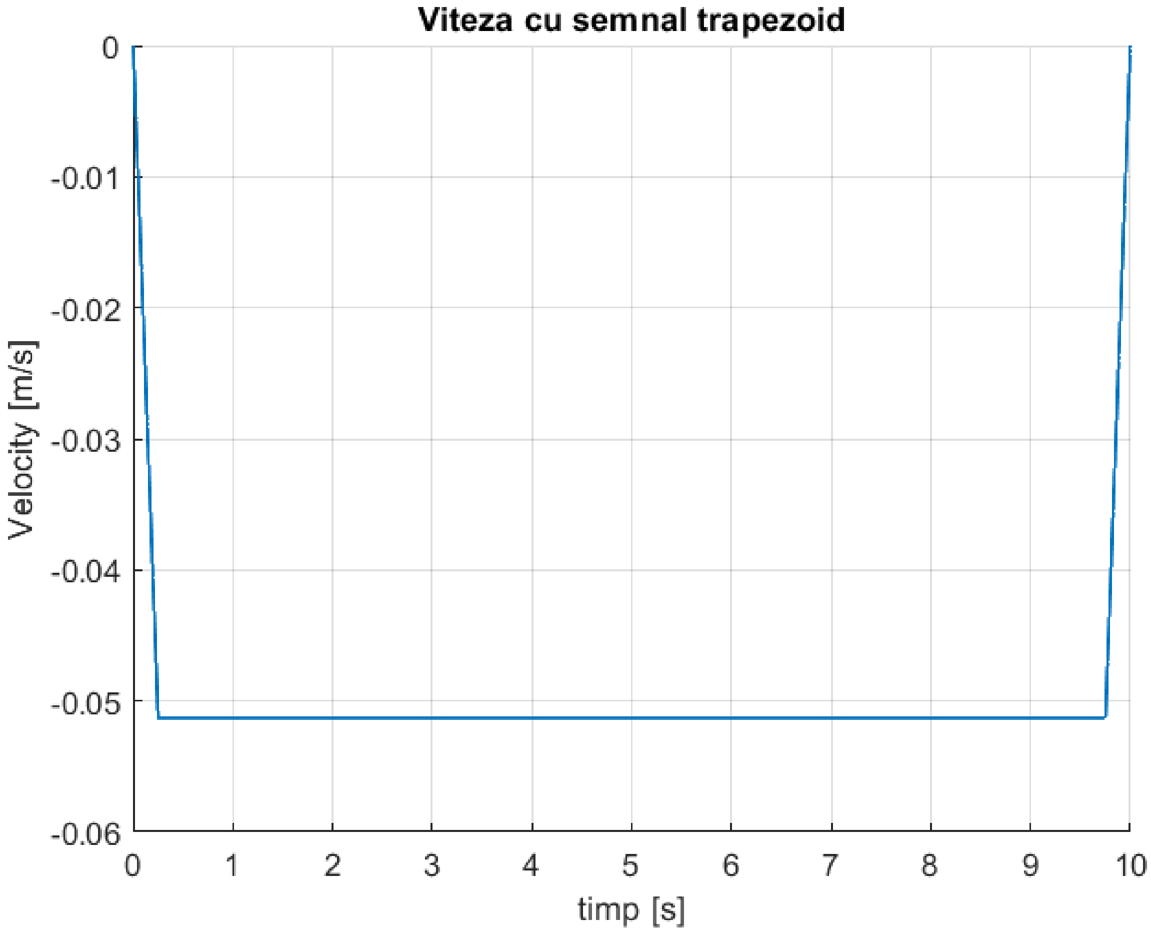


Fig 2.4.2 Viteza cu semnal trapezoid

Semnalul prezinta 3 regiunii, caracterizate prin urmatoarele ecuatii :

- Regiunea de crestere a vitezei pana la valoarea sa maxima:

$$V = \frac{V_{max}}{t_r} \cdot t \quad (6.1)$$

- Mentinerea vitezei la valoarea sa maxima:

$$V = V_{max} \quad (6.2)$$

- Descreserea vitezei pana la valoarea nula:

$$V = \frac{V_{max}}{t_r} \cdot (t_f - t) \quad (6.3)$$

Dupa cum se observa in figura 2.4.2 si in ecuatiile care descriu figura, efectorul accelereaza pe o perioada t_r, numita **timp de crestere** pana se ajunga la viteza sa maxima. Aceasta viteza va fi mentinuta pe majoritatea cursei, pana cand se ajunge la perioada t_d, numita **timp de descrestere** si fiind egala cu $t_f - t_r$, unde t_f este **timpul final**. Viteza efectorului v-a scadea la valoarea 0, atunci cand se atinge t_f.

Similar cu metoda folosind polinoame, trebuie sa se stie pozitia finala la care trebuie sa ajunga efectorul, Viteza maxima, fiind dependenta de acest parametru, cat si de t_r si t_f.

$$V_{max} = \frac{X}{(t_f - t_r)} \quad (6.4)$$

Valoarea vitezei maxime

2.5 MODELUL GEOMETRIC INVERS

MGI permite determinarea configurației în care trebuie sa ajunga structura mecanica a robotului (a vectorului coordonatelor articulare q_j) astfel încât dispozitivul efector să fie poziționat în poziția dorită x_i.

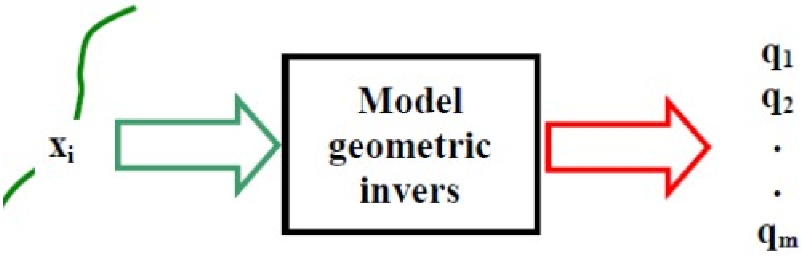


Fig 2.5 Modelul Geometric Invers

Exista mai multe metode de determinare a MGI, in aceasta lucrare se prezenta metoda numerica. Aceasta metoda este iterativa, deci rezultatul trebuie sa convearga la o valoare.

$$q_{k+1} = q_k + \alpha \cdot J^{-1}(q) \cdot \Delta x \quad (7.1)$$

Algoritmul de aproximare
numerica Newton

Unde :

- q_{k+1} este iteratia curenta
- q_k este iteratia precedenta
- α factorul de invatare
- Δx eroarea de pozitie fata de pozitia dorita.

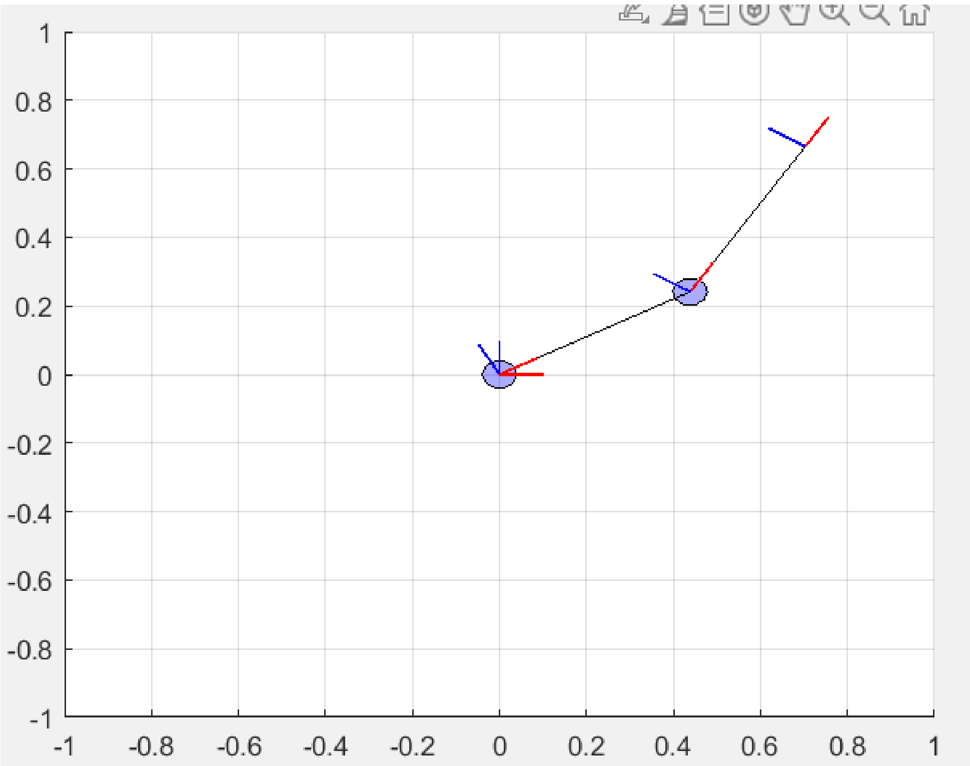


Fig 2.5 Brat Robotic la coordonatele x = 0.7071 si y = 0.7071

O problema care poate fi intampinata ar fi inversarea Jacobianului. Este posibil ca Jacobianul sa nu fie o matrice de forma patratica, ceea ce nu permite inversarea acesteia prin metoda conventionala.

Exista 2 moduri prin care se poate afla inversa matricii J :

- Pseudo-inversare

$$J^{-1} = \left(J^T \cdot J\right)^{-1} J^T \quad (7.2)$$

- Folosirea doar a unui minor caracteristic

$$J = \begin{pmatrix} -l1\sin(q_1) - l2\sin(q_1+q_2) & -l2\sin(q_1+q_2) \\ l1\cos(q_1) + l2\cos(q_1+q_2) & l2\cos(q_1+q_2) \end{pmatrix} (7.3)$$

(Minorul caracteristic al robotul proiectat)

Metoda folosita nu este ceea mai optima, existand pozitii care nu pot fi atinse. In capitolele urmatoare, se vor folosi metode mai sofisticate, care au la baza metoda prezentata in acest subcapitol.

3. MODELUL DINAMIC AL ROBOTULUI

Modelul Dinamic este o aproximare a realitatii care permite stabilirea unei relatii cauzale conditiile in care starea sistemului este cunoscuta, acesta fiind capabil sa faca predictii asupra viitorului doar daca, pe langa starea curenta, sunt cunoscute si starile anterioare. Este nevoie deci de cunoașterea unei istorii a ceea ce s-a întâmplat deja.

Formula generalizata, pentru determinarea Fortelor/Momentelor generate de un sistem este :

$$M(q)[\ddot{q}]+C(q)[\dot{q}^2]+B(q)[\dot{q}\dot{q}]+G(q)=T \quad (8.0)$$

Unde :

- M reprezinta matricia de inertie a sistemului
- C si B, reprezinta matricea fortelor centrifuge, respectiv Coriolis
- G matricea fortelor gravitationale

Determinarea acestor matricii se poate face folosind o varietate de metode. In lucrarea aceasta se vor prezenta 2 dintre aceste metode.

3.1 METODA NEWTON-EULER

Determinarea Fortelor/Momentelor din arcticulatii, se va face in doua etape :

- Etapa Cinematica
- Etapa Dinamica

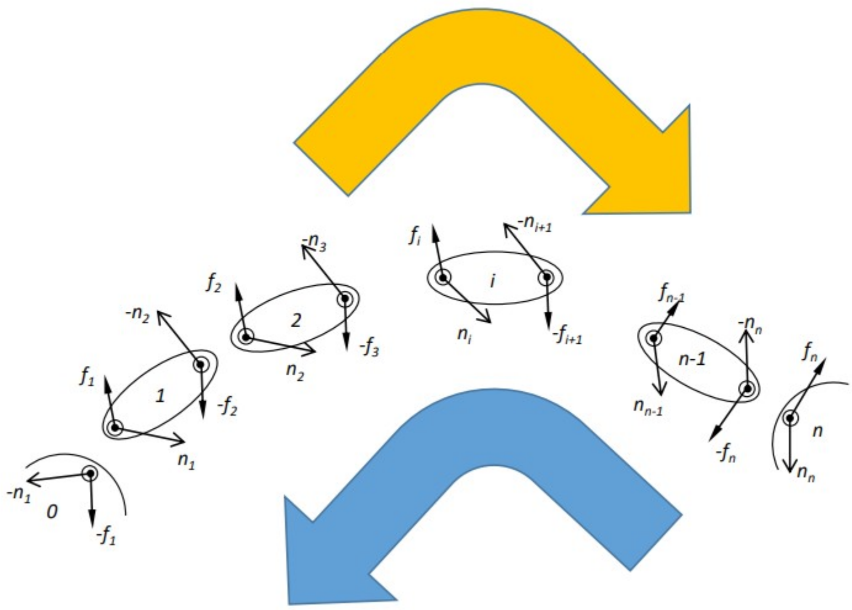


Fig. 3.1.1 Ordinea de parcurgere a metodei Newton-Euler

Etapa Cinematica consta in parcurgerea lantului cinematic de la baza robotului, la efector (sageata galbena). Se vor cauta sa se determine acceleratiile unghiulare $\dot{\omega}$ si liniare $\dot{v}$ ale fiecarui link al robotului.

Formulele pentru determinarea acestor doua variabile sunt :

$$^{i}a_j = R_0^i \, {}^{0}\ddot{p}_j \quad (9.1)$$

$$^{j}\varepsilon_i = {}^{i}\dot{\omega}_j \quad (9.2)$$

Unde :

- $^{i}a_j$ este acceleratia liniara j in reperul i
- $^{0}\ddot{p}_j$ este acceleratia liniara masurata in centrul de masa al segmentului j in reperul global

- R_0^i este matricea de rotatie de la reperul global, la reperul i
- ${}^j\varepsilon_i = {}^i\dot{\omega}_j$ acceleratia unghiulara i in reperul j

La fiecare link, al robotului, se vor calcula Fortele F si Momentele N ale fiecarui link :

$$F_i = m_i\,{}^i a_i \quad (9.3)$$

$$N_i = I_i\,{}^i\varepsilon_i + ({}^i\omega_i)^T I_i\,{}^i\omega_i \quad (9.4)$$

Unde :

- m_i masa link-ului i
- I_i torsorul de inertie al link-ului i

Dupa ce s-a trecut prin intregul lant cinematic, si s-au calculat toate fortele/momentele din fiecare segment, se va trece in **etapa Dinamica**. In etapa dinamica se vor calcula fortele si momentele din articulatii. Aceasta se va parcurge de capatul efectorului, la baza (sageata albastra).

$${}^i f_i = R_{i+1}^i\,{}^{i+1}f_{i+1} + F_i \quad (9.5)$$

$${}^i n_i = R_{i+1}^i\,{}^{i+1}n_{i+1} + N_i + {}^i p_{i+1} R_{i+1}^i\,{}^{i+1}f_{i+1} - {}^i p_i\,{}^i f_i \quad (9.6)$$

3.2 METODA LAGRANGE-EULER

Determinarea Fortelor/Momentelor se va face folosind Lagrangeanul. Pentru a folosi Lagrangeanul este mai intai nevoie sa se determine energia cinetica **K** si potentiala **U**.

$$\mathrm{K} = \frac{1}{2} \cdot (\dot{q})^T M(q)\,\dot{q} \quad (10.1)$$

$$\mathrm{U} = G(q)\,q \quad (10.2)$$

Lagrangeanul este diferenta dintre energia cinetica si energia potentiala :

L = K – U (10.3)

Unde cuplul este dat de urmatoarea formula :

$$\frac{d}{dt}\frac{\partial K}{\partial \dot{q}} - \frac{\partial K}{\partial q} + \frac{\partial U}{\partial q} = \tau \ (10.4)$$

$$\frac{d}{dt}\frac{\partial K}{\partial \dot{q}} - \frac{\partial K}{\partial q} = M(q)[\ddot{q}] + B(q)[\dot{q}^2] + C(q)[\dot{q}\dot{q}] \ (10.4.1)$$

$$\frac{\partial U}{\partial q} = G(q) \ (10.4.2)$$

Unde :

$$M(q) = \Sigma\left(m_i J_{vi}^T J_{vi} + J_{wi}^T I_{ci} J_{wi}\right) \ (10.5)$$

$$J_{v1} = \begin{pmatrix} -c_{11}\sin(q_1) & 0 \\ c_{11}\cos(q_1) & 0 \\ 0 & 0 \end{pmatrix} \ (10.6)$$

$$J_{v2} = \begin{pmatrix} -l_1\sin(q_1) - c_{11}\sin(q_1+q_2) & -c_{11}\sin(q_1+q_2) \\ l_1\cos(q_1) + c_{11}\cos(q_1+q_2) & c_{11}\cos(q_1+q_2) \\ 0 & 0 \end{pmatrix} \ (10.7)$$

$$J_{w1} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \end{pmatrix} \ (10.8)$$

$$J_{w2} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 1 \end{pmatrix} \ (10.9)$$

Comparativ cu metoda Newton-Euler prezentata la subcapitolul anterior, Lagrange-Euler este mult mai rapida la calcule, dar nu se vor da toate Fortele/Momentele din cuple, ci doar cele specifice tipului de articulatie care este (articulatie R – moment de rotatie, articulatie T – forta).

3.3 DETERMINAREA ECUATIEI MOMENTELOR

Indiferent de metoda folosita, in final se vor obtine urmatoarele matricii :

$$M(q) = \begin{pmatrix} m_1 (c_{11})^2 + m_2 (l_1^2 + c_{22}^2) + 2 m_2 l_1 c_{22} \cos(q_2) + I_{zz,1} + I_{zz,2} & m_2 c_{22}^2 + m_2 c_{22} l_1 \cos(q_2) + I_{zz,2} \\ m_2 c_{22}^2 + m_2 c_{22} l_1 \cos(q_2) + I_{zz,2} & m_2 c_{22}^2 + I_{zz,2} \end{pmatrix} \quad (11.1)$$

$$C(q) = \begin{pmatrix} 0 & -m_2 l_1 c_{22} \sin(q_2) \\ -m_2 l_1 c_{22} \sin(q_2) & 0 \end{pmatrix} \quad (11.2)$$

$$B(q) = \begin{pmatrix} -2 m_2 l_1 c_{22} \sin(q_2) \\ 0 \end{pmatrix} \quad (11.3)$$

$G(q) = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ (11.4) , deoarece vectorul acceleratiei gravitationale este : $g = \begin{pmatrix} 0 \\ 0 \\ g \end{pmatrix}$

4. SIMULAREA VIRTUALA A BRATULUI ROBOTIC

4.1 PROTOTIPUL VIRTUAL

Modelul virtual al robotului a fost creat, folosind aplicatia de proiectare grafica, CATIAv5. Bratul Robotic este alcatuit din 3 partii, o baza si 2 link-uri, unul dintre acestea fiind link-ul efector.

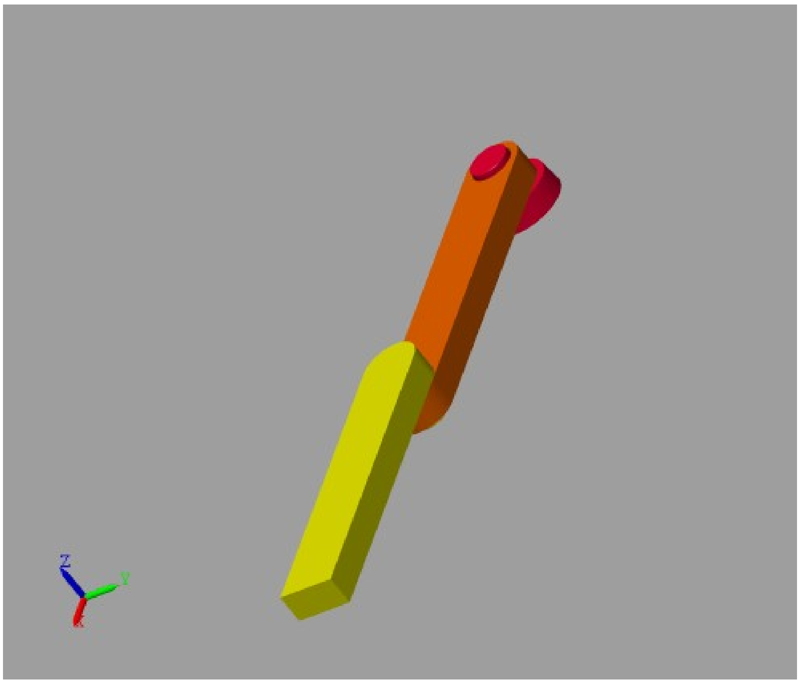


Fig 4.1.1 Modelul Virtual al Bratului Robotic

Pentru fiecare cupla, se vor alege aceelas motor electric, pentru punerea in miscare a bratului. Motorul se va alege in functie de momentele maxime, care pot aparea la aceste cuple. Nu exista o problema daca sunt situatii in care cuplul sa depaseasca valoarea nominala, cat timp aceasta depasire nu are loc pe durata lungii de timp.

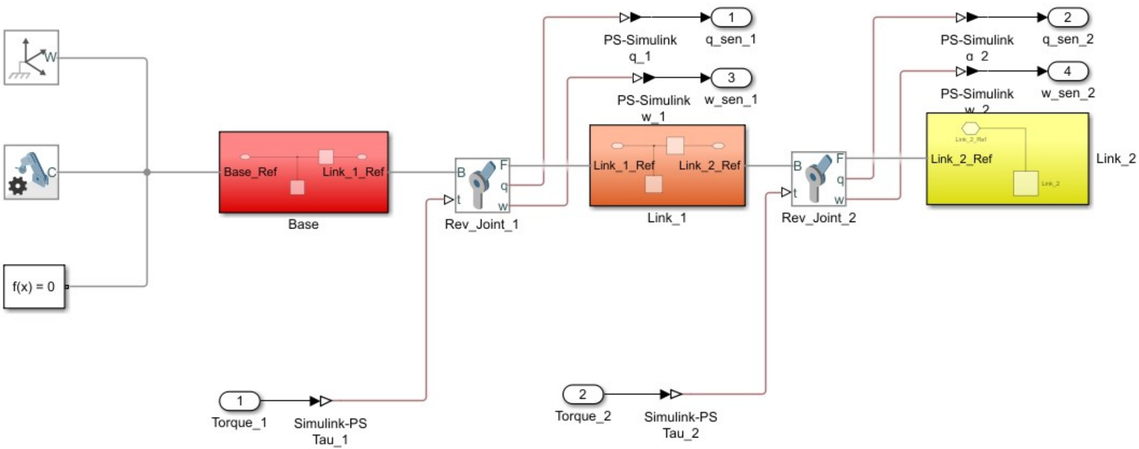
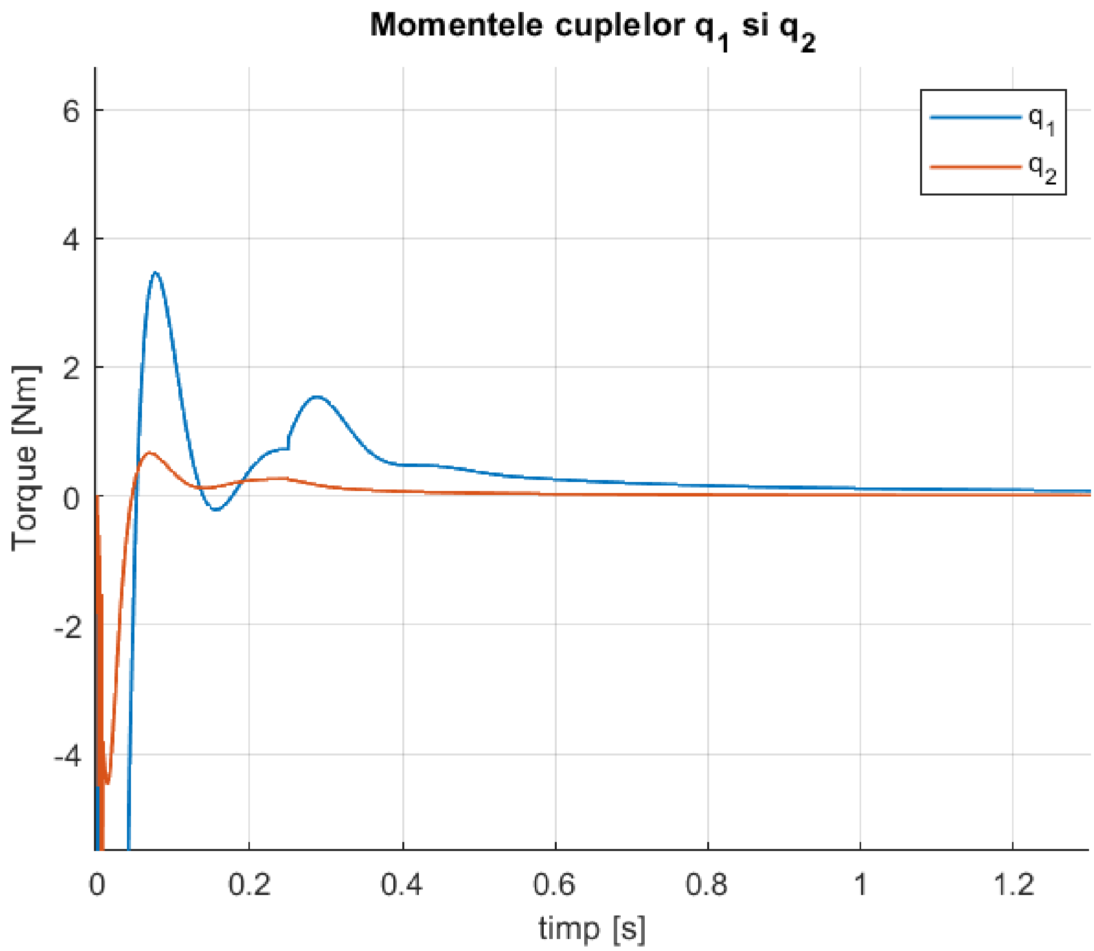


Fig 4.1.2 Schema bloc al Modelului Virtual al Bratului robotic



.

Fig 4.1.3 Graficele momentelor in cuple

Se observa in graficul din figura 4.1.2, ca exista o crestere brusca a momentului (4Nm - 10Nm), timp de o fractiune de secunda, cand estepus in miscare robotul. In rest, valoarea momentelor se afla in domeniul mNm – zecilor de mNm.

Motorul ales este un motor de curent continuu, cu perii, **DCX 35 L,** din gama maxon.

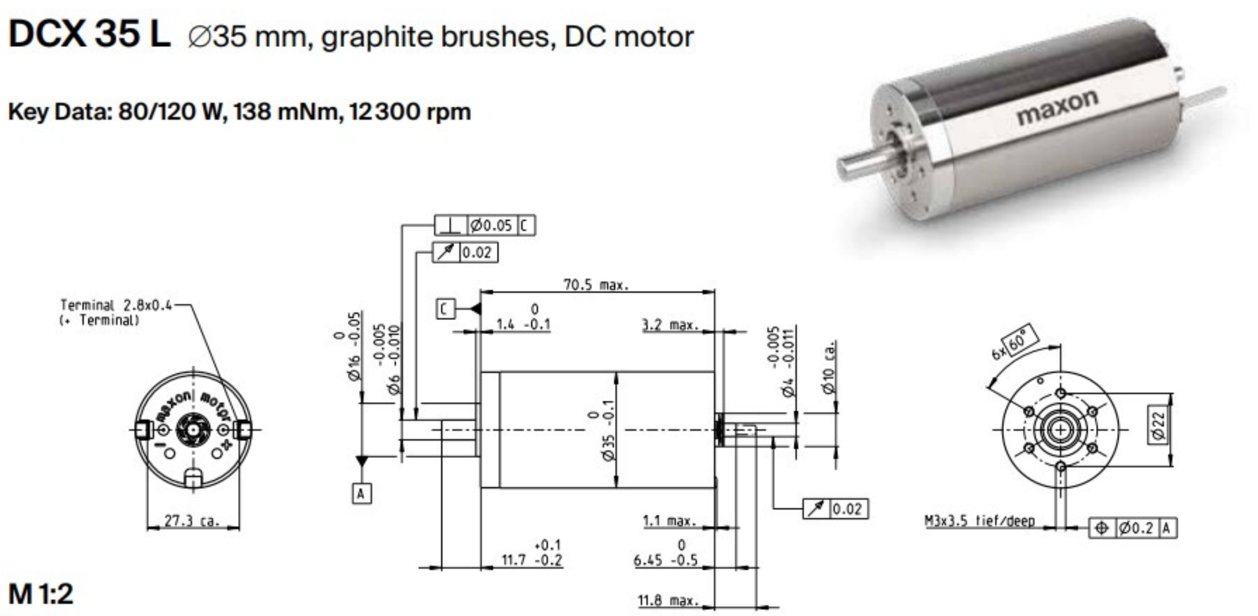


Fig 4.1.4 Desenul tehnic al motorului

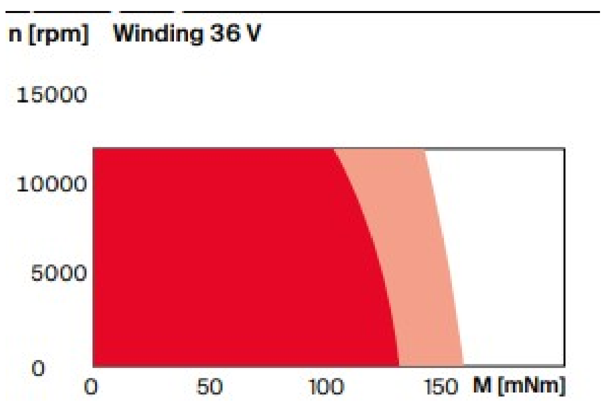


Fig 4.1.5 Caracteristica mecaninca de functionare a motorului

In figura 4.1.4 este prezentata caracteristica mecanica a motorului. Zona de functionare in continuu a motorului este colorata cu rosu, adica motorul poate functiona in zona data, indiferent de perioada de functionare. Zona alba in schimb, permite o functionalitate pe o perioada scurta de timp.

Pentru a creste valoarea momentului, peste ceea nominala, fara a strica motorul, se va folosi un reductor.

1_	Nominal voltage	V	18
2_	No load speed	rpm	7200
3_	No load current	mA	177
4_	Nominal speed	rpm	6640
5_	Nominal torque	mNm	120
6_	Nominal current (max. continuous current)	A	5.32
7_	Stall torque	mNm	1980
8_	Stall current	A	84.8
9_	Max. efficiency	%	88
10_	Terminal resistance	Ω	0.212
11_	Terminal inductance	mH	0.077
12_	Torque constant	mNm/A	23.4
13_	Speed constant	rpm/V	408
14_	Speed/torque gradient	rpm/mNm	3.70
15_	Mechanical time constant	ms	3.97
16_	Rotor inertia	gcm^2	102

Fig 4.1.6 Parametrii Motorului

Reductorul ales, este reductorul planetar **GPX 42,** din gama maxon.

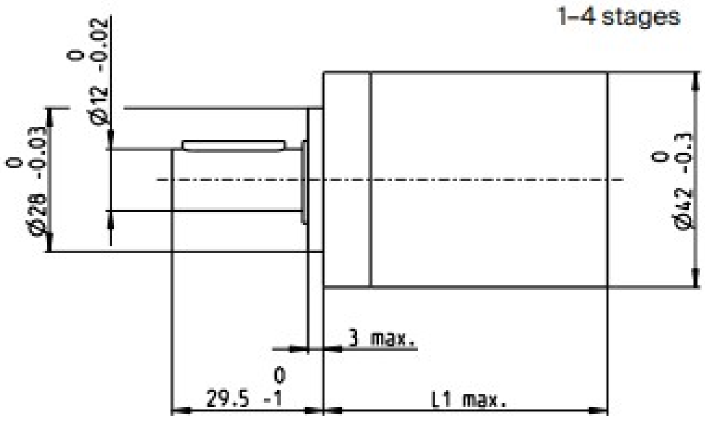


Fig 4.1.7 Desenul tehnic al reductorului planetar

Reduction	150 : 1
Absolute reduction	2401/16
Number of stages	4
Max. continuous torque	15 Nm
Max. intermittent torque	22.5 Nm
Direction of rotation, drive to output	Equal
Max. efficiency	64 %
Average backlash no load	1 °
Mass inertia	5 gcm²
Gearhead length (L1)	84.5 mm
Max. transmittable power (continuous)	20 W
Max. transmittable power (intermittent)	25 W

Fig 4.1.8 Parametrii reductorului

$$M_n = 150 * 120\,mNm * 0.88 * 0.64 = 10{,}137.6\,mNm$$

Valoarea momentului nominal dupa aplicarea reductorului

4.2 CONTROLUL ROBOTULUI DIRECT PRIN CUPLE

Prima schema de control al robotului va controla bratul robotic, folosind valoarea pozitie in spatiul articular a celor 2 cuple.

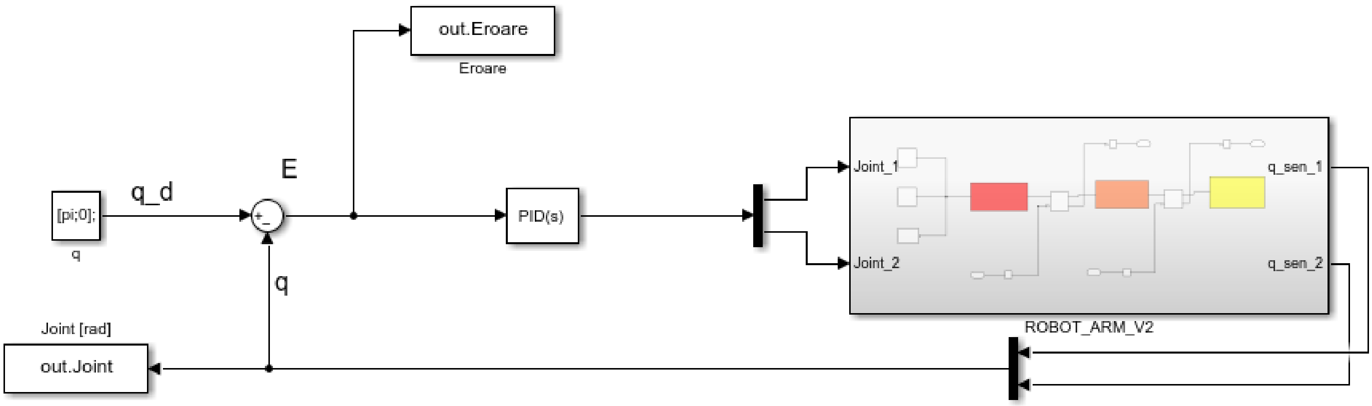


Fig 4.2.1 Schema bloc

Intrarea schemei este valoarile dorite la care vrem sa ajunga articulatiile. Se i-a eroarea deplasarii **E**, aceasta fiind $q_d - q$. Eroarea E arata, cat de mult difera valoarea dorita, de referinta q_d fata de valoarea returnata de la senzorii q. PID-ul are rolul de regulator, asigurand ca eroarea sa fie cat mai mica, pentru cazul primei schemei, semnalul de eroare va fi amplificat doar cu proportionalul P, derivatorul D si integratorul I avand amandoi coeficientii nulii. Cu cat valoarea erorii este mai mare, cu atat corectia facuta de sistem va fi mai semnificativa.

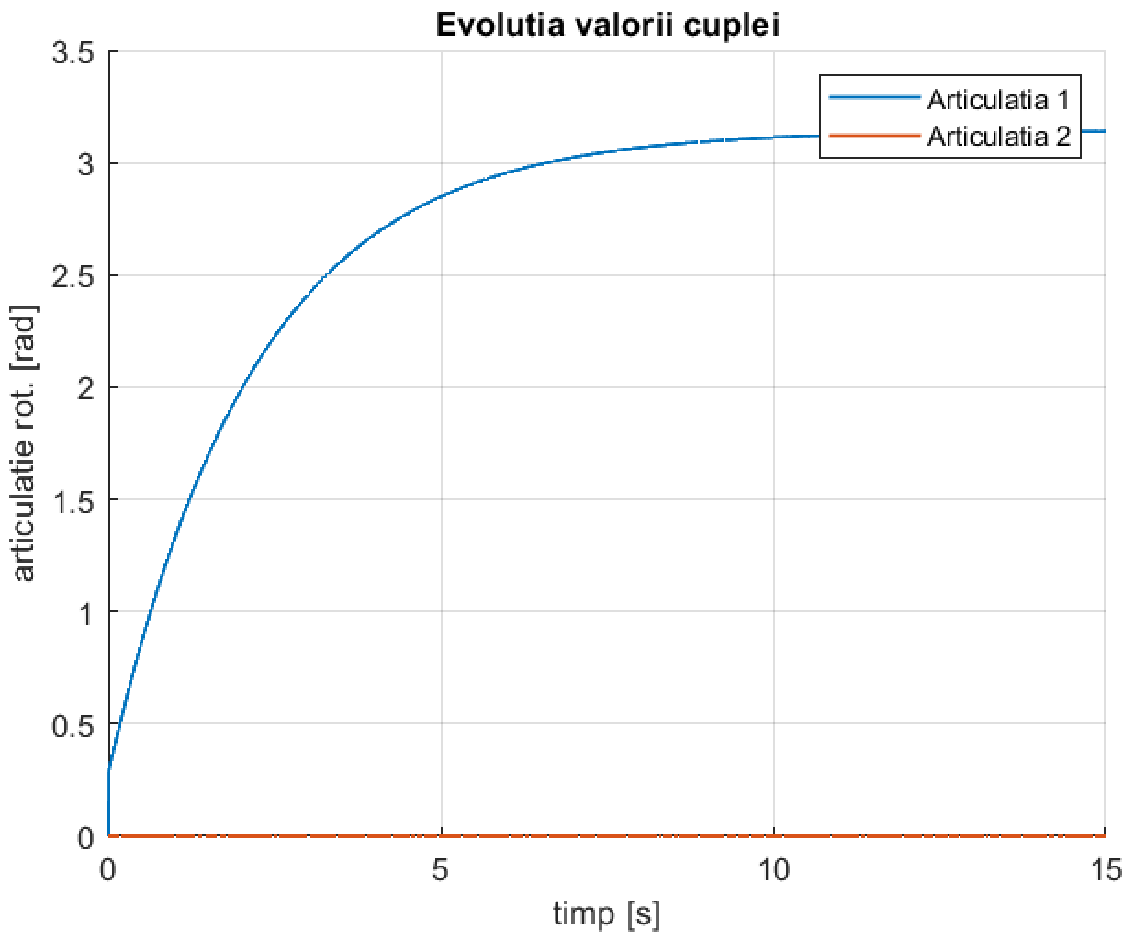


Fig 4.2.2 Graful evolutiei valorii cuplelor in timp

Nu este necesar, adaugarea termenirol D si I la regulator, deoarece ,dupa cum se observa in figura 4.2.2, nu exista suprareglaj si valoarea se stabilizeaza la valoarea dorita.

4.3 CONTROLUL ROBOTULUI PRIN MOTOARE

Asemanator cu schema precedenta, controlul va fi facut tot folosind pozitiile dorite in spatiul articular. Diferenta este ca valorile dorite vor fi date de motoare, acestea fiind controlate prin voltajul dat la intrare.

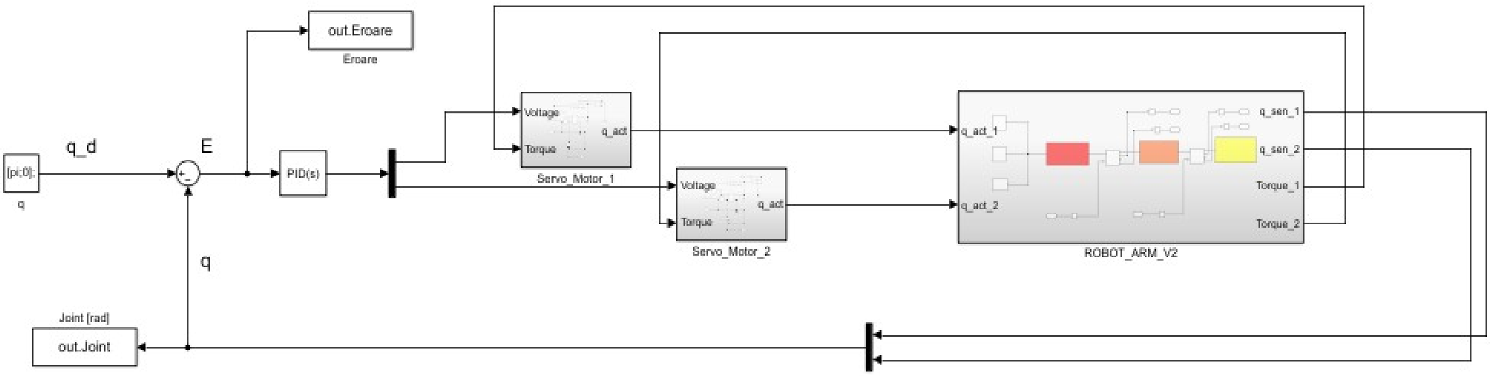


Fig 4.3.1 Schema bloc

Eroarea este calculata si amplificata, prin regulatorul PID, si aceasta este trimisa la motoare ca voltaj. Motoarele convertesc tensiunea primita, in putere mecanica (viteza unghiulara ω si momentul de rotatie T).

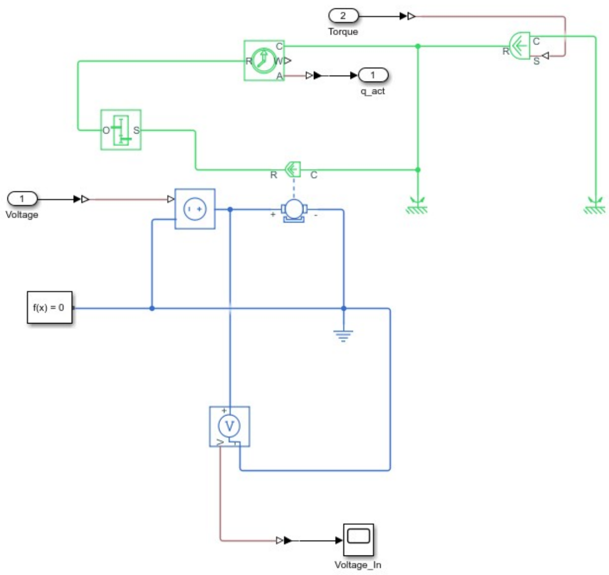


Fig 4.3.2 Schema bloc a servo-motorului

Ecuatia fundamentala a miscarii este :

$$T - T_s = I \frac{d\omega}{dt} \quad (12.0)$$

Legea ne zice ca, aparitia unui moment motor T, diferit de un moment de sarcina T_s ($T - T_s$ nu este null), duce la variatia impulsului, si vice versa.

In cazul dat momentul motor este generat de servo-motor, iar momentul rezistent este momentul bratului.

Diferenta acestor dpua momente, vor duce la variatia vitezei unghiulare ω a motorului, care va deveni constanta in momentul cand cele doua momente T si T_s egale.

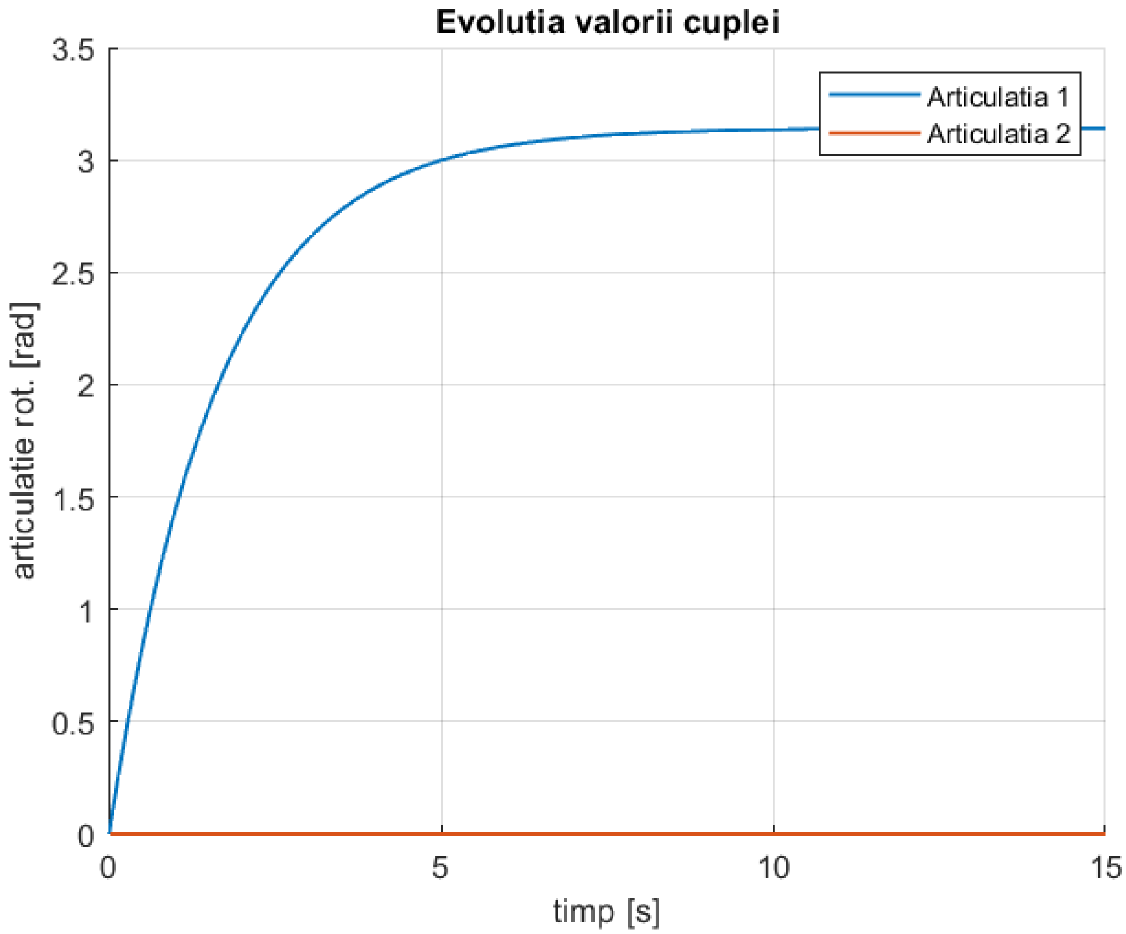


Fig 4.3.3 Graful evolutiei valorii cuplelor in timp

4.4 CONTROLUL ROBOTULUI PRIN FORTE/MOMENTE

Controlul bratului robotic se efectueaza acum, folosind pozitiile, vitezele, si acceleratiile dorite. Pentru ca algoritmul sa functioneze, va fi nevoit sa se faca o interpolare de la pozitia initiala, pana la pozitia finala dorite, trecanduse prin mai multe puncte intermediare. Acest lucru duce ca efectorul sa urmareasca o traiectorie. Generarea traiectoriilor se pot efectua folosind una din legile discutate in subcapitolele 2.4.1 si 2.4.2.

Pentru exemplul dat, generarea traiectoriei va folosi legea de miscare discutata in capitolul 2.4.2, **Legea de miscare folosind semnal trapezoid**.

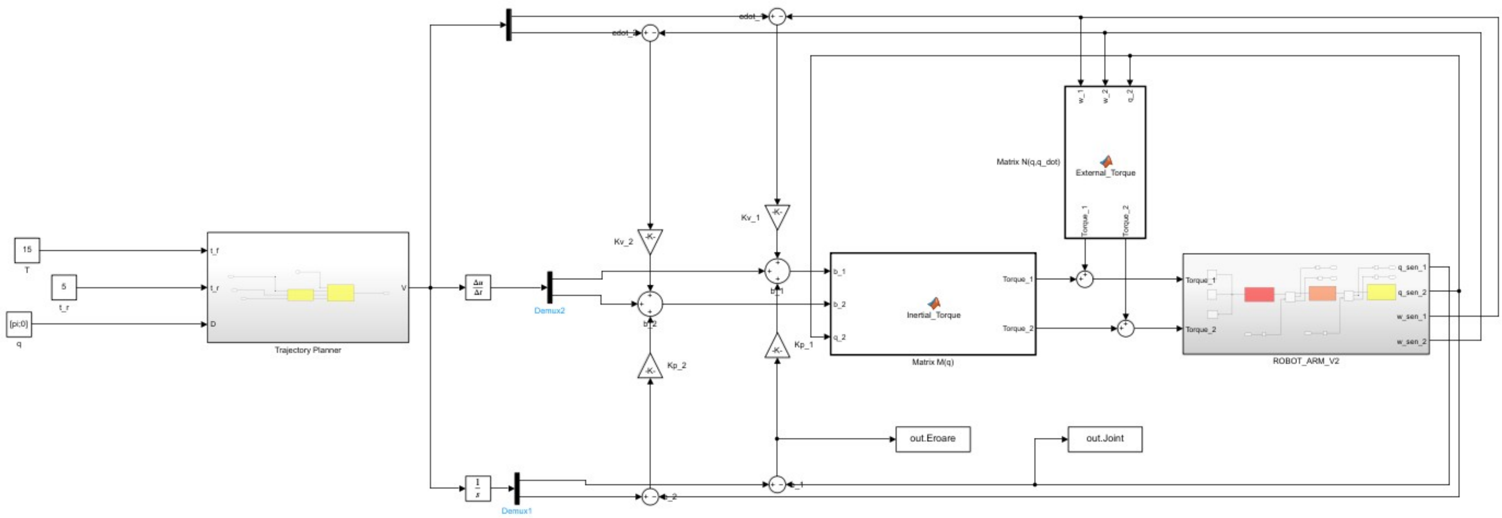


Fig 4.4.1 Schema bloc

Schema prezentata in figura 4.4.1 este format din 2 blocuri importante: blocul de generare a traiectoriei, si schema de control in sine.

Ecuatia care caracterizeaza sistemul este :

$$\ddot{e}+K_v\dot{e}+K_p e=0 \quad (13.1)$$

Ecuatia diferentiala a sistemului

Unde :

- K_v este coeficientul erori vitezelor

- K_p este coeficientul erori pozitilor

Prin modificarea valorilor K_v si K_p, se poate controla forma semnalului de iesire, ecuatia diferentiala find de tip PT2.

$$\frac{1}{s^2+2\omega_n \zeta s+\omega_n} \quad (13.2)$$

Ecuatia caracteristica a unui sistem PT2

Pentru ca sistemul PT2 sa fie stabil, trebuie ca polii sistemului sa fie mai mici de zero ($s_{1,2}<0$). Valoarea polilor depinde in intregime de valorile lui ω si ζ, unde ω reprezinta pulsatia naturala a sistemului si ζ coeficientul de amortizare.

In sistemul nostru coeficientii care afecteaza stabilitatea sistemului, pe cat si altii parametrii cum ar suprareglajul , banda de stabilitate etc. vor fi coeficientii K_v si K_p .

Valoriile alese pentru ceii doi coeficientii sunt : $K_p=100^2$ si $K_v=200$

Ambii coeficientii au fost alesi ca fiind coeficientii unui binomi, deoarece in acel punct semnalul are calitatile cele mai bune (suprareglaj zero, nu este supra amortizat, timpul traznitoriul cel mic etc.)

Semnalul de intrare va avea forma urmatoare:

$$\ddot{q}-u \quad (13.3)$$

$$u=-K_v\dot{e}-K_p e \quad (13.4)$$

Legea de comanda

In schema de control legea de comanda (13.3) va fi folosite in Modelul Dinamic Invers (MDI), pentru a determina cuplele din articulatii, robotul fiind controlat prin cuple.

$$T = M(q)[\ddot{q} - u] + N(q, \dot{q}) \quad (13.5)$$

Unde :

- $N(q, \dot{q})$ este $B(q)[\dot{q}^2] + C(q)[\dot{q}\dot{q}] + G(q)$

Momentele T apoi sunt transmise in cuple iar bratul robotic se misca in pozitia dorita.

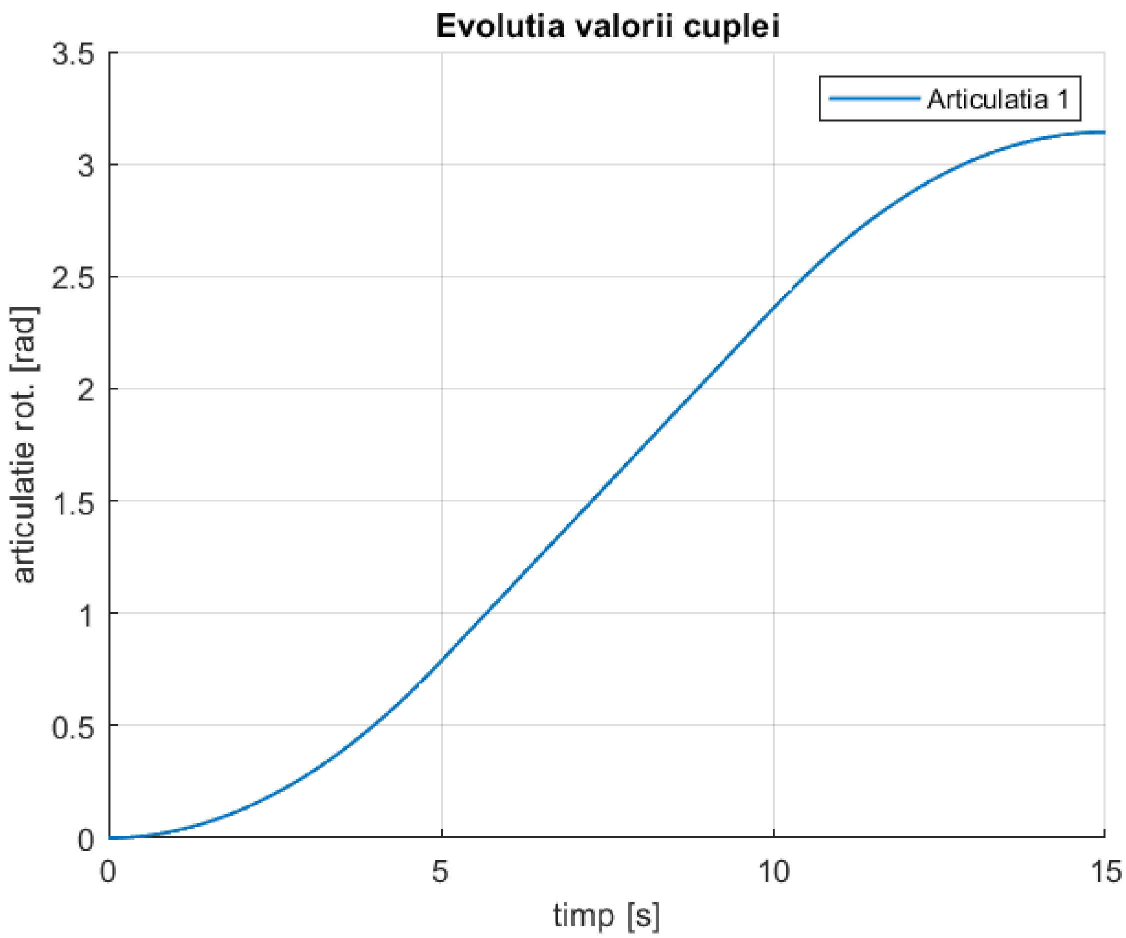


Fig 4.4.2 Graful evolutiei valorii cuplei in timp

Din traductoare se scot valorile pozitie, vitezei, la momentul curent, pentru a da “update” la erorile de poitie si viteza, iar procesul se repeta pana cand valorile erorilor sunt sub o valoare deja prestabilita de “treshold”.

4.5 CONTROLUL ROBOTULUI IN SPATIUL CARTEZIAN

Asemanator capitolului trecut, controlul se face identic, diferenta fiind ca se vor folosi pozitiile in spatiul cartezian si nu cele din spatiul articular.

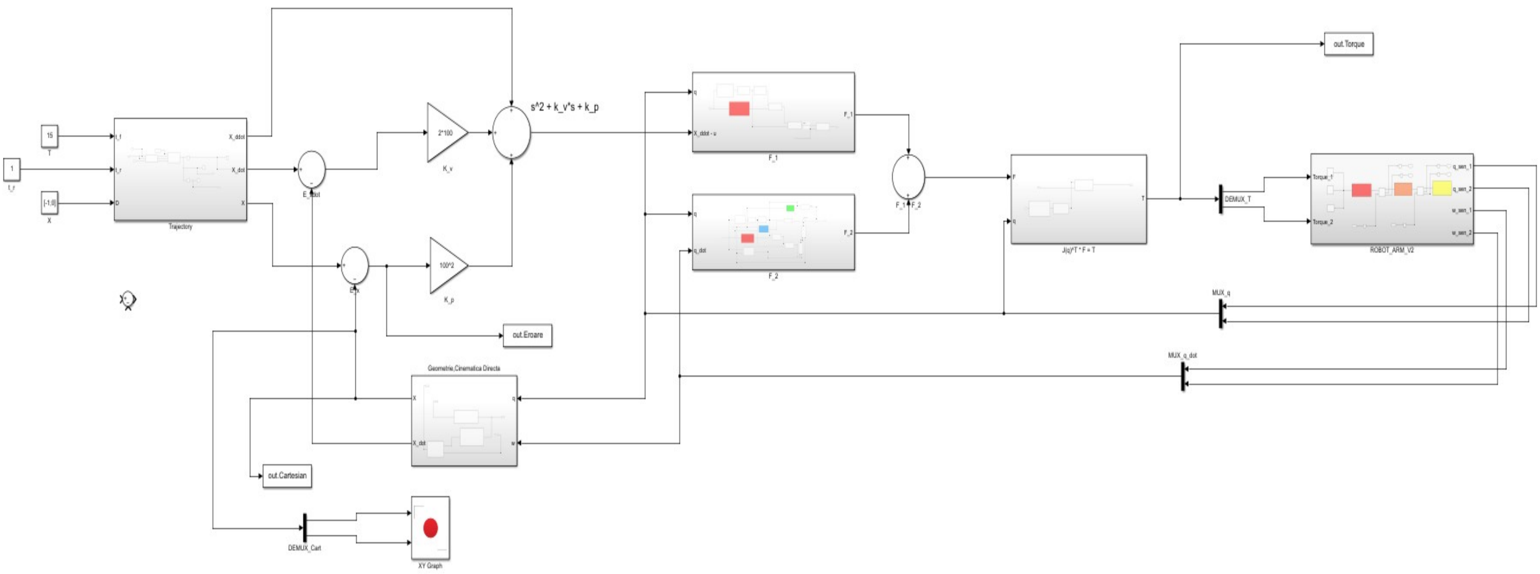


Fig 4.5.1 Schema bloc

Acum legea de comanda devine :

$$\ddot{x} - u \quad (14.1)$$

$$u = -K_v \dot{e} - K_p e \quad (13.4)$$

Si relatia (14.1) va fi folosita in determinarea fortelor in spatiul cartezian :

$$F = F_1 + F_2 \quad (14.2)$$

$$F_1 = M(x)[\ddot{x} - u] \quad (14.3)$$

$$F_2 = V(\dot{x}, x) + G(x) \quad (14.4)$$

Unde :

- M(x) este $J(q)^{-T} M(q) J(q)^{-1}$
- $V(x, \dot{x})$ este $J(q)^{-T} V(q, \dot{q}) - M(q) \dot{J}(q) \dot{q}$, unde $V(q, \dot{q}) = C(q)[\dot{q}^2] + B(q)[\dot{q}\dot{q}]$
- G(x) este $J(q)^{-T} G(q)$

Forta aflata in relatia (14.2) va fi convertita din spatiul cartezian, in spatiul articular prin relatia (14.5) :

$$J(q)^T F = T \quad (14.5)$$

Momentul T, fiind apoi introdus in cuple pentru a deplasa robotul, asemanator subcapitolului 4.4 .

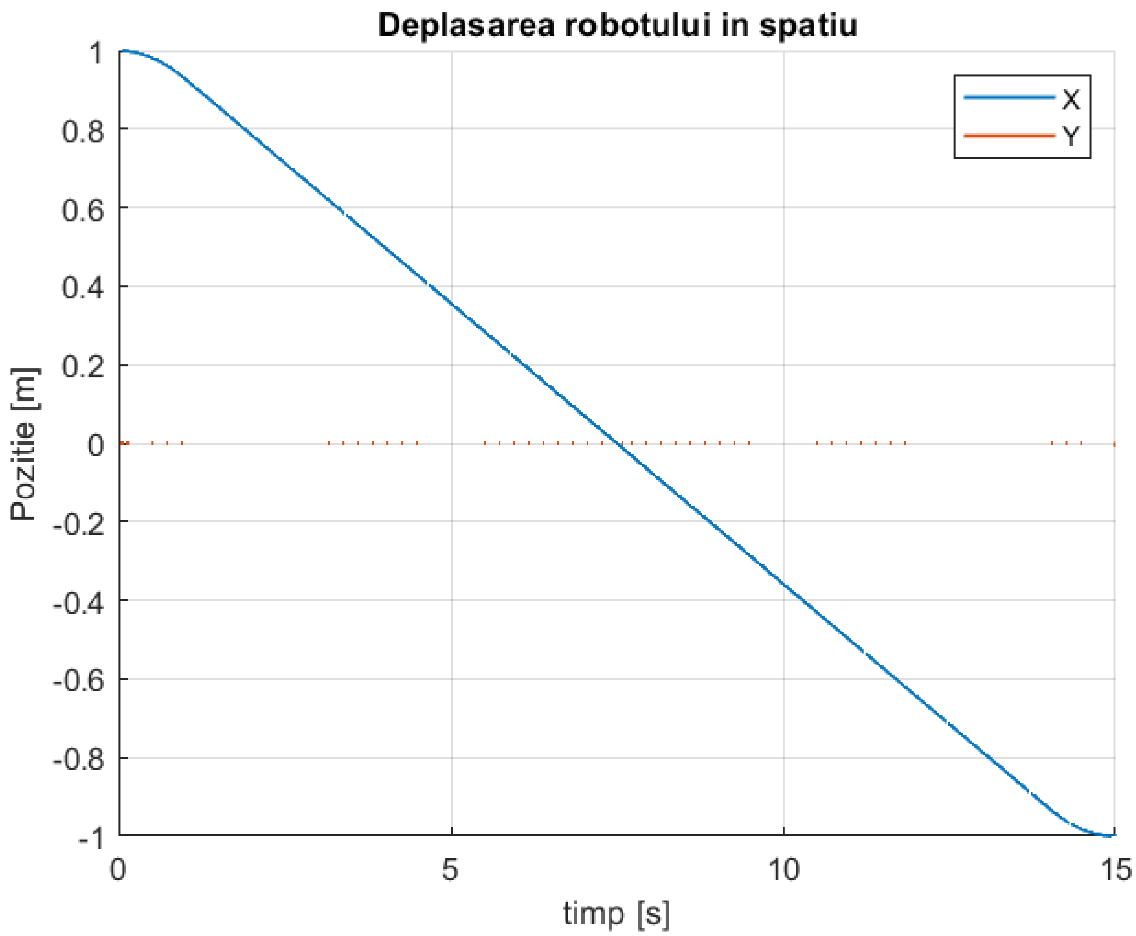


Fig 4.5.2 Evolutia deplasarii efectorului in spatiu

4.6 EVOLUTIA SI COMPARAREA ERORILOR

In acest capitol se vor discuta si afisa evolutia valoarilor erorilor, fiecarei metode discutate in subcapitolele anterioare.

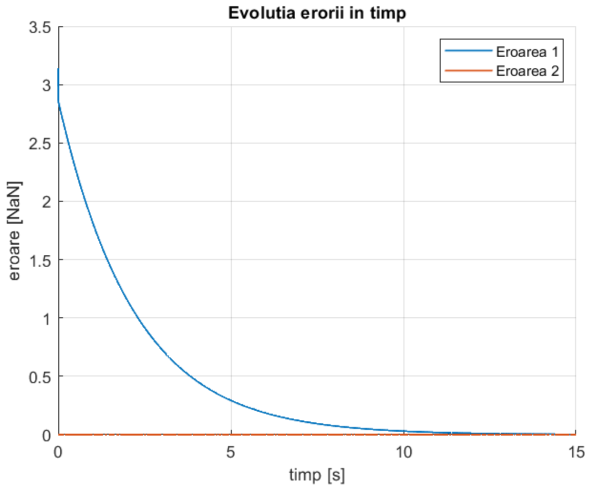


Fig 4.6.1 Evolutia erorii subcapitolul 4.2

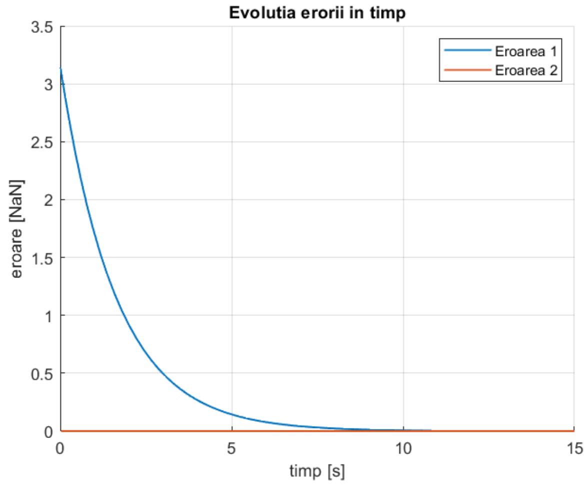


Fig 4.6.2 Evolutia erorii subcapitolul 4.3

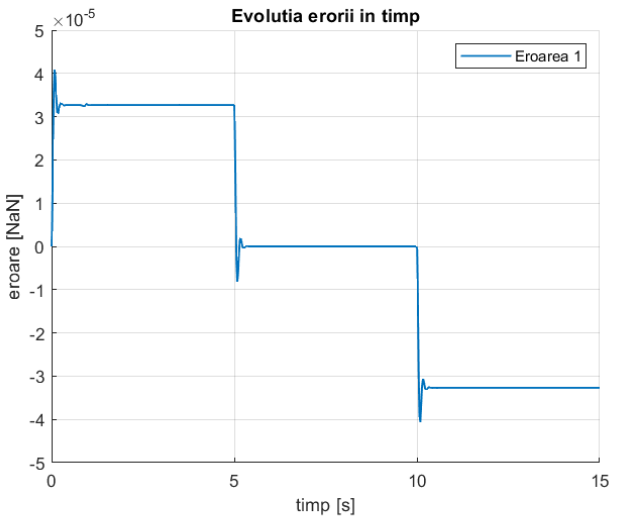


Fig 4.6.3 Evolutia erorii subcapitol 4.4

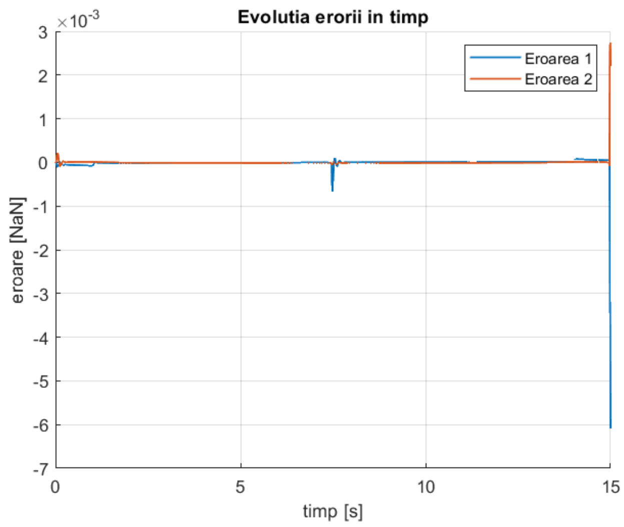


Fig 4.6.4 Evolutia erorii subcapitol 4.5

Dupa cum se observa, eroare primelor doua scheme, 4.2 si 4.3, au o evolutie a erorii foarte asemanatoare, ambele fiind o scadere exponentiala a valorii erorii spre 0.

Celalte doua grafice, 4.4 si 4.5, au valorii ale erorii care oscileaza in jurul valorilor de 10^{-5} , respectiv 10^{-3} , acest lucru fiind datorat faptului ca valorile $q_d, \dot{q}_d, \ddot{q}_d$, respectiv $x_d, \dot{x}_d, \ddot{x}_d$ nu raman constante in timp ,ci isi schimba vallorile, acestea luand valorile punctelor interpolate pe traiectorie.

Toate erorile, la $t \rightarrow \infty$ iau valorii de zero, ceea ce inseamna ca acestea ajung in pozitia specificata.

5. CONCLUZIE

Am proiectat un brat robotic, cu doua cuple rotative (RR), care este capabil sa indeplineasca sarciniile de “pick and place” si “tracking.

6. ANEXE

```
function RR_Robot(q,l,dim)
%UNTITLED Summary of this function goes here
%   Detailed explanation goes here

for t  = 0:0.01:1

view(dim)
hold on;
grid on;
pause(0.0001)
cla


axis([0 1 0 1 -0.25 0.25])

q_1 = -2*q(1,:)*t.^3 + 3*q(1,:)*t.^2; %Legea de miscare cupla_1
q_2 = -2*q(2,:)*t.^3 + 3*q(2,:)*t.^2; %Legea de miscare cupla_2

%-----------MGD----------------

x = [0.1;0;0];
y = [0;0.1;0];   %%Reperul {0} --> Baza
z = [0;0;0.1];
o = [0;0;0];

```

```matlab


T_01 = [cos(q_1) , -sin(q_1) , 0 , 0; ...
        sin(q_1) ,  cos(q_1) , 0 , 0; ...   %Transformare de la {0} la {1}
            0    ,      0     , 1 , 0; ...
            0    ,      0     , 0 , 1];

T_12 = [cos(q_2) , -sin(q_2) , 0 , l(1,:); ...
        sin(q_2) ,  cos(q_2) , 0 ,    0  ; ...  %Transformare de la {1} la {2}
            0    ,      0     , 1 ,    0  ; ...
            0    ,      0     , 0 ,    1  ];

T_2e = [    1    ,      0     , 0 , l(2,:); ...
            0    ,      1     , 0 ,    0  ; ...  %Transformare de la {2} la {e}
            0    ,      0     , 1 ,    0  ; ...
            0    ,      0     , 0 ,    1  ];

T_02 = T_01 * T_12 ;%Transformare de la {0} la {2}
T_0e = T_02 * T_2e ;%Transformare de la {0} la {e}

Ox = T_01*[x;1];
Oy = T_01*[y;1]; %Reperul {1} --> Cupla_Rotatie_1
Oz = T_01*[z;1];
Oo = T_01*[o;1];

Sx = T_02*[x;1];
Sy = T_02*[y;1]; %Reperul {2} --> Cupla_Rotatie_2
Sz = T_02*[z;1];
So = T_02*[o;1];


Px = T_0e*[x;1];
Py = T_0e*[y;1]; %Reperul {e} --> Efectorul
Pz = T_0e*[z;1];
Po = T_0e*[o;1];


R_matrix = [Oo,So,Po]; %Robotul

cupla_1 = nsidedpoly(10, 'Center', [Oo(1,:) Oo(2,:)], 'Radius', 0.04);
plot(cupla_1, 'FaceColor', 'b')

```

```
cupla_2 = nsidedpoly(10, 'Center', [So(1,:) So(2,:)], 'Radius', 0.04);
plot(cupla_2, 'FaceColor', 'b')

plot(R_matrix(1,:),R_matrix(2,:),'black','LineWidth',0.5)

plot3([o(1,:),x(1,:)],[o(2,:),x(2,:)],[o(3,:),x(3,:)],'r','LineWidth',1)
plot3([o(1,:),y(1,:)],[o(2,:),y(2,:)],[o(3,:),y(3,:)],'b','LineWidth',1)
plot3([o(1,:),z(1,:)],[o(2,:),z(2,:)],[o(3,:),y(3,:)],'g','LineWidth',1)

plot3([Oo(1,:),Ox(1,:)],[Oo(2,:),Ox(2,:)],[Oo(3,:),Ox(3,:)],'r','LineWidth',1)
plot3([Oo(1,:),Oy(1,:)],[Oo(2,:),Oy(2,:)],[Oo(3,:),Oy(3,:)],'b','LineWidth',1)
plot3([Oo(1,:),Oz(1,:)],[Oo(2,:),Oz(2,:)],[Oo(3,:),Oz(3,:)],'g','LineWidth',1)

plot3([So(1,:),Sx(1,:)],[So(2,:),Sx(2,:)],[So(3,:),Sx(3,:)],'r','LineWidth',1)
plot3([So(1,:),Sy(1,:)],[So(2,:),Sy(2,:)],[So(3,:),Sy(3,:)],'b','LineWidth',1)
plot3([So(1,:),Sz(1,:)],[So(2,:),Sz(2,:)],[So(3,:),Sz(3,:)],'g','LineWidth',1)

plot3([Po(1,:),Px(1,:)],[Po(2,:),Px(2,:)],[Po(3,:),Px(3,:)],'r','LineWidth',1)
plot3([Po(1,:),Py(1,:)],[Po(2,:),Py(2,:)],[Po(3,:),Py(3,:)],'b','LineWidth',1)
plot3([Po(1,:),Pz(1,:)],[Po(2,:),Pz(2,:)],[Po(3,:),Pz(3,:)],'g','LineWidth',1)

end
end
```

Fig 6.1 Functie Matlab Robot_RR

```
function Robot_Graph_V2(q)
%UNTITLED2 Summary of this function goes here
%   Detailed explanation goes here
T = 1;
hold
grid
t = 0:0.01:1;

P = -2*q*(t.^3)/T^3 + 3*q*(t.^2)/T^2;
V = -6*q*(t.^2)/T^3 + 6*q*(t)/T^2;
A = -12*q*(t)/T^3 + 6*q/T^2;

plot(t,P,'Color','r','LineWidth',2);
plot(t,V,'Color','b','LineWidth',2);
plot(t,A,'Color','g','LineWidth',2);
legend('Deplasare [m]','Viteza [m/s]','Acceleratie [m/s^2]')
xlabel('timp [s]')




end
```

Fig 6.2 Generare Lege de miscare polinomiala

```
function V= Velocity(t,t_f,t_r,V_max)

V = [0;0];
if t <= t_r
    V = V_max*t/t_r;
end

if (t > t_r) && (t < t_f - t_r)
    V = V_max;
end

if (t >= t_f - t_r) && (t <= t_f)
    V = V_max*(t_f - t)/t_r;
end
```

Fig 6.3 Generare Lege de miscare semnal Trapezoid

```
function Robot_Simuly(p_d,l,K)
%UNTITLED Summary of this function goes here
%   Detailed explanation goes here
hold on;
grid on;

q_1 = 10.^-9; %10.^-9
q_2 = 10.^-9;

eps =  0.05; % precizia 0.0001

error = p_d ;
sum_error = [0;0];

axis([-1 1 -1 1 ])

while (sqrt(error(1,:).^2 + error(2,:).^2)) > eps

pause(0.1)
cla

error_pred = error;


x = [0.1;0;0];
y = [0;0.1;0];   %%Reperul {0} --> Baza
z = [0;0;0.1];
o = [0;0;0];


```

```
T_01 = [cos(q_1) , -sin(q_1) , 0 , 0; ...
        sin(q_1) ,  cos(q_1) , 0 , 0; ...   %Transformare de la {0} la {1}
            0     ,     0      , 1 , 0; ...
            0     ,     0      , 0 , 1];

T_12 = [cos(q_2) , -sin(q_2) , 0 , l(1,:); ...
        sin(q_2) ,  cos(q_2) , 0 ,    0  ; ...  %Transformare de la {1} la {2}
            0     ,     0      , 1 ,    0  ; ...
            0     ,     0      , 0 ,    1  ];

T_2e = [    1     ,     0      , 0 , l(2,:); ...
            0     ,     1      , 0 ,    0  ; ...  %Transformare de la {2} la {e}
            0     ,     0      , 1 ,    0  ; ...
            0     ,     0      , 0 ,    1  ];

T_02 = T_01 * T_12 ;%Transformare de la {0} la {2}
T_0e = T_02 * T_2e ;%Transformare de la {0} la {e}

Ox = T_01*[x;1];
Oy = T_01*[y;1]; %Reperul {1} --> Cupla_Rotatie_1
Oz = T_01*[z;1];
Oo = T_01*[o;1];

Sx = T_02*[x;1];
Sy = T_02*[y;1]; %Reperul {2} --> Cupla_Rotatie_2
Sz = T_02*[z;1];
So = T_02*[o;1];


Px = T_0e*[x;1];
Py = T_0e*[y;1]; %Reperul {e} --> Efectorul
Pz = T_0e*[z;1];
Po = T_0e*[o;1];


R_matrix = [Oo,So,Po]; %Robotul

cupla_1 = nsidedpoly(10, 'Center', [Oo(1,:) Oo(2,:)], 'Radius', 0.04);
plot(cupla_1, 'FaceColor', 'b')
```

```
cupla_2 = nsidedpoly(10, 'Center', [So(1,:) So(2,:)], 'Radius', 0.04);
plot(cupla_2, 'FaceColor', 'b')

plot(R_matrix(1,:),R_matrix(2,:),'black','LineWidth',0.5)

plot3([o(1,:),x(1,:)],[o(2,:),x(2,:)],[o(3,:),x(3,:)],'r','LineWidth',1)
plot3([o(1,:),y(1,:)],[o(2,:),y(2,:)],[o(3,:),y(3,:)],'b','LineWidth',1)
plot3([o(1,:),z(1,:)],[o(2,:),z(2,:)],[o(3,:),y(3,:)],'g','LineWidth',1)

plot3([Oo(1,:),Ox(1,:)],[Oo(2,:),Ox(2,:)],[Oo(3,:),Ox(3,:)],'r','LineWidth',1)
plot3([Oo(1,:),Oy(1,:)],[Oo(2,:),Oy(2,:)],[Oo(3,:),Oy(3,:)],'b','LineWidth',1)
plot3([Oo(1,:),Oz(1,:)],[Oo(2,:),Oz(2,:)],[Oo(3,:),Oz(3,:)],'g','LineWidth',1)

plot3([So(1,:),Sx(1,:)],[So(2,:),Sx(2,:)],[So(3,:),Sx(3,:)],'r','LineWidth',1)
plot3([So(1,:),Sy(1,:)],[So(2,:),Sy(2,:)],[So(3,:),Sy(3,:)],'b','LineWidth',1)
plot3([So(1,:),Sz(1,:)],[So(2,:),Sz(2,:)],[So(3,:),Sz(3,:)],'g','LineWidth',1)

plot3([Po(1,:),Px(1,:)],[Po(2,:),Px(2,:)],[Po(3,:),Px(3,:)],'r','LineWidth',1)
plot3([Po(1,:),Py(1,:)],[Po(2,:),Py(2,:)],[Po(3,:),Py(3,:)],'b','LineWidth',1)
plot3([Po(1,:),Pz(1,:)],[Po(2,:),Pz(2,:)],[Po(3,:),Pz(3,:)],'g','LineWidth',1)

x_k = l(1,:)*cos(q_1) + l(2,:)*cos(q_1 + q_2)
y_k = l(1,:)*sin(q_1) + l(2,:)*sin(q_1 + q_2)

Jv =  [-l(1,:)*sin(q_1)-l(2,:)*sin(q_1+q_2),-l(2,:)*sin(q_1+q_2);
        l(1,:)*cos(q_1)+l(2,:)*cos(q_1+q_2), l(2,:)*cos(q_1+q_2)];

error = p_d - [x_k;y_k]
delta_error = error - error_pred;
sum_error = sum_error + error;

delta_q = inv(Jv)*(K(1,:)*error + K(2,:)*delta_error + K(3,:)*sum_error); %PID

q_1 = q_1 + delta_q(1,:);
q_2 = q_2 + delta_q(2,:);


end
```

Fig 6.4 Geometrie inversa Robot RR

7. BIBLIOGRAFIE

- [1] Pozna, C. (2021). Bazele Cinematicii Robotilor Industriali.
- [2] Pozna, C. (2021). Dinamica Robotilor
- [3] Pozna, C. (2021). Sisteme de Conducere in Robotica.
- [4] https://www.maxongroup.com/maxon/view/content/index