RELAZIONE

Progetto Robotica Industriale

Giaimo Natale
Matricola 209424

Introduzione

Il progetto è stato sviluppato su MatLab, implementando i concetti e le funzioni studiate durante il corso.
Esso si suddivide in 7 file:

- **Industriale** : script principale che contiene la procedura di risoluzione dei problemi di percorso per il robot PUMA560; Esso contiene una moltitudine di funzioni che verranno descritte nel dettaglio più avanti.
- **RTsulPiano** : Funzione che prende in input tre punti e restituisce una matrice di rototraslazione RT_{Piano}^{Base} e un raggio. La matrice descrive l'operazione di trasformazione dal sistema di riferimento di base al sistema sul piano nel quale giacciono i tre punti P. Questa matrice viene utilizzata per il calcolo della circonferenza passante per i punti P;
- **lambda / lmabdad** : rappresentano le funzioni polinomiali utilizzate per il calcolo dei valori delle variabili di giunto.
- Funzioni fornite durante il corso: **Antopomorfo_Cin_Dir, Antropomorfo_Cin_Inv e Jacobiano_Antropomorfo** per la risoluzione dei problemi di cinematica diretta, inversa e differenziale per un braccio antropomorfo.

Inizializzazione

L'inizializzazione dello script principale consiste nella creazione delle variabili fondamentali:

- Tabella D-H fornita nella traccia;
- Matrice P 3x3 che descrive i punti forniti nella traccia, sulle righe abbiamo le coordinate x, y, z e ogni colonna rappresenta un punto diverso.
- Vettore T contenente gli estremi degli intervalli temporali. Essi sono stati scelti in modo arbitrario tali che $P(T_i) = P_i$.
- timeSpan è una discretizzazione dell'intervallo temporale con passo dTime.

Percorso triangolare

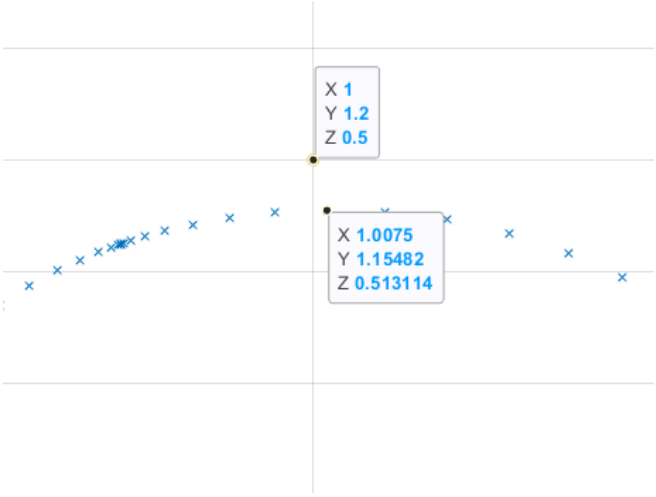
Il percorso viene suddiviso in tre segmenti rappresentanti il percorso a minima distanza tra i vertici, corrispondenti ai punti forniti dalla traccia. Questa suddivisione ci permette di risolvere il problema su ogni segmento tramite l'applicazione delle formule studiate a lezione. Una spiegazione più dettagliata viene fornita nell'analisi della funzione *pointVelocity*.
La procedura risolutiva si basa su un ciclo che itera sugli elementi del vettore timeSpan, così facendo ogni iterazione analizzerà un istante temporale. Abbiamo l'assegnazione degli output delle funzioni su quattro matrici di dimensione length(timeSpan)x3, le righe rappresentano i differenti istanti temporali e le colonne contengono i valori. Le coordinate sono espresse nei termini del sistema di riferimento di base.

- PP: posizione dell'origine di SR_3, ovvero l'end-effector;
- VV: velocità lineare di SR_3 espressa nei termini V_x, V_y, V_z;
- QQ: valori delle variabili di giunto;
- QQd: velocità angolari delle variabili di giunto.

Circonferenza

Data la differente natura del problema è indispensabile effettuare delle decisioni di progettazione per facilitare l'implementazione della soluzione.
La circonferenza viene calcolata in un sistema di riferimento con origine al centro della circonferenza stessa e congruente al piano sul quale giacciono i punti. Questo ci permette di poter costruire un sistema di riferimento sul piano e calcolare la matrice di rototraslazione dal sistema di base al piano. Importante sottolineare che, essendo tutti i punti sul piano, la coordinata z sullo stesso sarà sempre pari a 0 per tutti i punti appartenenti alla circonferenza. Si noterà, nei dati finali, che l'utilizzo di questa procedura porterà a piccole deviazioni ($\leq 1cm$) dei punti effettivi e la loro posizione prevista; personalmente li considero accettabili in questo contesto data l'unità di misura utilizzata (*metri*) e le funzioni

implementate e richiamate. Nell'immagine notiamo l'errore sopra discusso.



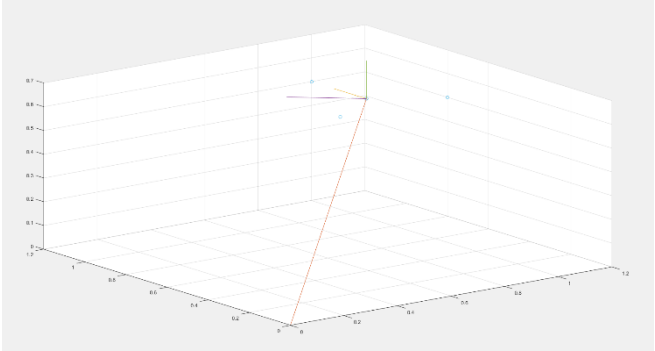
Concettualmente la procedura risolutiva è analoga a quella adottata per il percorso triangolare: si analizzano semi-circonferenze descritte dagli angoli $\theta_i = \tan_2^{-1}\left(\frac{y_{P_{i_{Piano}}}}{x_{P_{i_{Piano}}}}\right)$, corrispondenti ai punti forniti nella traccia nel sistema di riferimento del piano dove essi giacciono.

RTsulPiano

Questa funzione serve per ricavare la matrice di rototraslazione che esprime la trasformazione dal sistema di coordinate di base ad un sistema sul piano dei tre punti forniti. La nuova origine sarà posta al centro della circonferenza descritta dai tre punti.

Iniziamo calcolando la normale del piano, essa sarà fornita dal prodotto vettoriale di due vettori non paralleli appartenenti al piano stesso. I due vettori sono forniti dai punti stessi. Calcoliamo, quindi, le coordinate del centro della circonferenza (punto mediano tra i tre punti) e ricaviamo il raggio.

Passiamo al definire i nuovi assi. Partendo dalla normale e dalla descrizione dell'asse x per il sistema di riferimento di base, ottengo un vettore $u_1 = n\ x\ [1,0,0]$ ad essi ortogonale.
Ripeto l'operazione tra la normale e u1 per ottenere $u_2 = n\ x\ u1$. Poiché l'asse z è la normale stessa, siamo adesso in possesso del nuovo sistema di riferimento descritto nelle coordinate del sistema di base. Trasliamo i vettori u sommando il vettore che descrive le coordinate del centro della circonferenza per traslare l'origine dei vettori all'origine del piano.
Il grafico seguente corrisponde alla figura *'sistema di riferimento sul piano'* generato dalla funzione *plotThis* di *RTsulPiano*.



Essendo adesso in possesso dei nuovi assi non resta che ricavare la matrice di rototraslazione. Otteniamo gli angoli compresi tra i corrispondenti assi che rappresentano gli angoli di Eulero:

$$\theta = \frac{u_1 * [1,0,0]}{|u_1|}$$
$$\gamma = \frac{u_2 * [0,1,0]}{|u_2|},$$
$$\omega = \frac{u_3 * [0,0,1]}{|u_3|}$$

Ricaviamo la matrice di rotazione con la funzione *eul2r* del RoboticsToolbox di *Peter Corke*.

$$eul2r(\theta, \gamma, \omega) = R_z(\theta) * R_y(\gamma) * R_x(\omega)$$

Poiché siamo in possesso delle coordinate dell'origine del sistema p nei termini del sistema b, possiamo ricavare immediatamente il vettore di traslazione. Sia d il vettore di traslazione dal sitema di base al sistema del piano $d = O_b - O_p$ con $O_b = [0,0,0]$, $O_p = centro \rightarrow d = -centro$
Ricaviamo così la matrice di rototraslazione RT_{Piano}^{Base} che descrive la trasformazione di coordinate dal sistema di base al sistema sul piano con origine il centro della circonferenza. Per ottenere la matrice che descrive la trasformazione dal sistema del piano al sistema di base basta invertire la matrice ottenuta:

$$RT_{Base}^{Piano} = \left(RT_{Piano}^{Base}\right)^{-1}$$

Siamo adesso in possesso della matrice RT dal sistema di base al piano, dobbiamo adesso descrivere i punti nei termini dell'angolo θ sulla circonferenza per avere dei riferimenti congruenti durante le operazioni. Di questo si occupa la funzione *pointToTheta*.

pointToTheta

Questa funzione si occupa di applicare la matrice di rototraslazione sui punti e ottenere la loro descrizione nei termini dell'angolo theta che essi descrivono sulla circonferenza. Iniziamo con l'applicazione della matrice di rototraslazione:

$$P_{i_{Piano}} = RT_{Piano}^{Base} * P_{i_{Base}}$$

Ricaviamo gli angoli applicando la funzione *atan2* di MatLab. Poiché la circonferenza è centrata nell'origine del sistema di riferimento corrente, otteniamo direttamente l'angolo al quale siamo interessati. Siamo adesso in possesso della notazione $P_p(i) = \theta(i)$; definendo, per ogni intervallo temporale $\tau = [T_i, T_{i+1})$, gli angoli $\theta_i = \theta(T_i)$ *e* $\theta_2 = \theta(T_{i+1})$ possiamo analizzare la circonferenza come la concatenazione di tre semi-circonferenze. Salviamo, quindi, gli angoli in un vettore Θ.

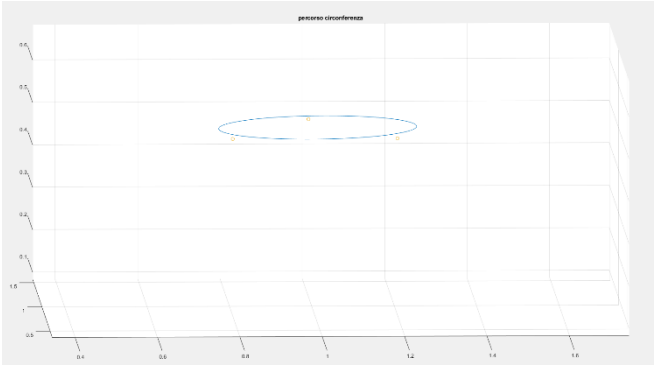
Importante tener conto che, essendoci una rotazione completa, potrebbero risultare problemi con il senso di percorrenza.

Ad esempio, se $\theta = \left[\frac{\pi}{2}, \pi, \frac{3}{2}\pi\right]$ abbiamo percorrenza in senso orario in $\theta_1 \to \theta_2 \to \theta_3$; il programma tratterà il percorso $\theta_3 \to \theta_1$ in senso anti-orario.

Per evitare questo il primo elemento del vettore viene copiato come ultimo elemento e si analizzano i primi due elementi per capire il senso di percorrenza:

$$\begin{cases} anti\ orario\ se\ \theta_1 > \theta_2 \\ orario\ altrimenti \end{cases}$$

Per concludere si effettua una iterazione sugli elementi del vettore Θ e si correggono eventuali problemi sommando (o sottraendo) 2π a tutti gli elementi successivi all'estremo finale della semi-circonferenza nella quale viene compiuta la rivoluzione.



Percorso descritto dall'end effector rispetto ai punti forniti, notiamo la presenza dell'errore precedentemente descritto.

Abbiamo così ricavato tutti i dati necessari per la risoluzione del problema del percorso circolare. Il procedimento è analogo a quello adottato per il percorso triangolare.

Posizione e velocità variabili di giunto

L'idea alla base del procedimento è quella di ricavare un punto in base al valore di σ applicando le funzioni:

- Percorso a minima distanza descritto dagli estremi $[P_1, P_2)$:

$$P(\sigma) = P_1 + \lambda(\sigma)(P_2 - P_1)$$
$$V(\sigma) = \dot{\lambda}(\sigma)(P_2 - P_1)$$

- Semi-circonferenza descritta dagli angoli $[\theta_1, \theta_2)$:

$$P(\sigma) = P_{centro} + raggio * |c\theta(\sigma)\ \ s\theta(\sigma)|^T$$
$$V(\sigma) = raggio * |-s\theta(\sigma)\ c\theta(\sigma)|^T * \dot{\theta}(\sigma)$$
$$\theta(\sigma) = \theta_1 + \lambda(\sigma) * (\theta_2 - \theta_1)$$
$$\dot{\theta}(\sigma) = \dot{\lambda}(\sigma) * (\theta_2 - \theta_1)$$

Con:

$$\sigma = \frac{t - T_1}{T_2 - T_1}$$
$$\lambda(\sigma) = 6\sigma^5 - 15\sigma^4 + 10\sigma^3$$
$$\dot{\lambda}(\sigma) = 30\sigma^4 - 60\sigma^3 + 30\sigma^2$$

La scelta di un polinomio di quinto grado deriva dal fatto che vogliamo imporre l'accelerazione a 0 nei punti estremi del percorso in analisi. Ottenute le posizioni dell'end-effector agli istanti temporali, possiamo applicare cinematica inversa e differenziale per ottenere i valori delle variabili di giunto.

Poiché analizziamo i percorsi per ogni istante temporale ed è diretta la trasformazione in termini di σ, possiamo implementare le formule sopra riportate.

Notiamo che, all'istante temporale $t_j \in [T_i, T_{i+1})$, il programma deve essere in grado di riconoscere in quale intervallo ricade l'istante che si sta analizzando. Una volta ricavato l'intervallo temporale è diretto ottenere gli estremi (punti o angoli) nell'intervallo corrente; di questo si occupa la funzione *index*.

Index

Funzione che prende in input due valori, l'istante temporale *time* e un intero $type = \begin{cases} 0\ minima\ distanza \\ 1\ semi\ circonferenza \end{cases}$; poiché, per costruzione, l'insieme di punti ha dimensione minore dell'insieme degli angoli, è indispensabile gestire l'overflow dell'indice se vogliamo le coordinate dei punti, questo avviene nell'ultimo if presente in questa funzione.
La funzione scorre gli estremi degli intervalli temporali e si ferma quando trova l'appartenenza di *time*. Per costruzione $t_j \in [T_i, T_{i+1})$, per l'istante temporale finale si verificherà che $T_{finale} \notin [T_{finale-1}, T_{finale})$, preveniamo l'evenienza forzando l'ultimo intervallo come $[T_{finale-1}, T_{finale}]$.

In possesso degli indici che rappresentano gli estremi dell'intervallo corrente, possiamo applicare le formule sopra descritte per ricavare i valori delle variabili di giunto. Specifichiamo due funzioni diverse per le due traiettorie da seguire: *pointVelocity* per la triangolare e $pointVelocityCircum$ per la circolare.
Esse implementano le formule usufruendo di due funzioni: *sigma* e *pTob*.
Come suggerisce il nome, *sigma* è l'implementazione di $\sigma = \frac{t-T_1}{T_2-T_1}$.
pTob prende in input un punto nel sistema di riferimento del piano $P_{i_{Piano}}$ e lo trasforma nei termini del sistema di riferimento di base applicando $P_{i_{Base}} = RM_{Base}^{Piano} * P_{i_{Piano}}$.

Cinematica

La cinematica del robot (diretta, inversa e differenziale) viene calcolata all'interno delle tre funzioni fornite durante il corso (AA 2021/2022).

Cinematica Diretta

La funzione **Antropomorfo_Cin_Dir** prende in input due vettori L e Q rappresentanti la lunghezza dei bracci del manipolatore e il valore delle variabili di giunto. Fornisce, in output, il punto P (sistema di riferimento di base) nel quale è posizionato l'end effector con le variabili di giunto prese in input.
Si ricavano le equazioni delle coordinate x, y, z partendo dai dati forniti nella tabella *DH*.
Siano:
c_i e s_i il seno e il coseno applicati sulla variabile di giunto Q_i;

$$D = L_2c_2 + L_3c_{13}$$

Allora le equazioni saranno:

$$\begin{cases} x = Dc_1 \\ y = Ds_1 \\ z = Ls_2 + L_3s_{23} \end{cases}$$

Cinematica Inversa

La funzione **Antropomorfo_Cin_Inv** prende in input le lunghezze L dei bracci e le coordinate del punto P espresse nel sistema di base; fornisce in output i valori delle variabili di giunto che descrivono la posa del robot per far si che l'end effector sia posizionato sul punto P.
Analogamente al passo precedente, le equazioni sono determinate partendo dalla tabella DH.
Siano:
c_i e s_i il coseno e il seno per la variabile di giunto Q_i;

$$c_3 = \frac{\left(P_x^2 + P_y^2 - L_2^2 - L_3^2\right)}{2L_2L_3}$$

$$s_3 = \sqrt{1 - c_3^2}$$

$$A = \begin{bmatrix} L_2 + L_3c_3 & L_3s_3 \\ -L_3s_3 & L_2 + L_3c_3 \end{bmatrix}$$

$$\begin{bmatrix} B_1 \\ B_2 \end{bmatrix} = A^{-1} \begin{bmatrix} P_z \\ \sqrt{P_x^2 + P_y^2} \end{bmatrix}$$

Allora otteniamo le variabili di giunto come:

$$\begin{cases} q_1 = \tan_2^{-1}(P_y, P_x) \\ q_2 = \tan_2^{-1}(s_3, c_3) \\ q_3 = \tan_2^{-1}(B_1, B_2) \end{cases}$$

Cinematica Differenziale

La funzione **Jacobiano_Antropomorfo** prende in input le lunghezze L dei bracci e le variabili di giunto Q; restituisce, in output, lo Jacobiano Analitico per il manipolatore rispetto al valore di Q.
Lo jacobiano analitico è ottenuto differenziando, rispetto al tempo, le equazioni della cinematica diretta.
Con $P(t)$ posizione all'istante t descritta da un set

di equazioni $P(t) = \begin{cases} x(t) = D(t)c_1(t) \\ y(t) = D(t)s_1(t) \\ z = Ls_2(t) + L_3s_{23}(t) \end{cases}$ e

$$D(t) = L_2c_2(t) + L_3c_{13}(t)$$

Differenziando rispetto al tempo otteniamo:

$$\dot{P} = J(Q)\dot{Q}$$

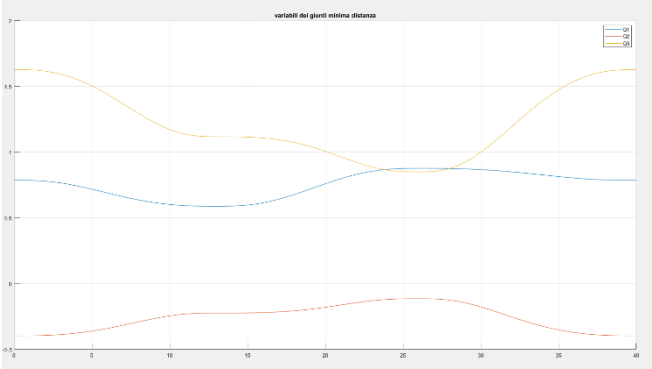
con $J(Q)$ pari alla matrice

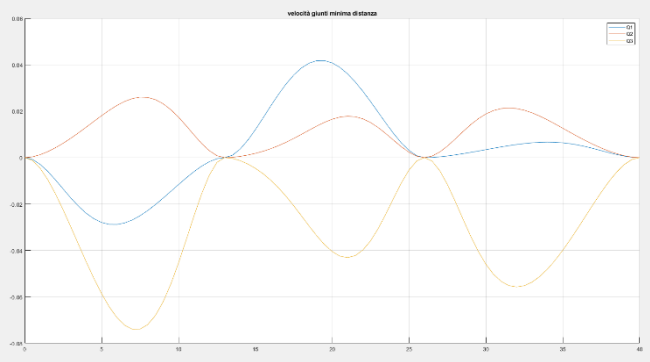
$$\begin{bmatrix} -s_1(L_2c_2 + L_3c_{23}) & -c_1(L_2s_2 + L_3s_{23}) & -L_3s_{23}c_1 \\ c_1(L_2c_2 + L_3c_{23}) & -s_1(L_2s_2 + L_3s_{23}) & -L_3s_{23}s_1 \\ 0 & L_2c_2 + L_3c_{23} & L_3c_{23} \end{bmatrix}$$

L'operazione per calcolare $\dot{P}$ viene svolta all'esterno della funzione.

Questo conclude la spiegazione per il funzionamento del codice. Di seguito riportati gli andamenti temporali delle variabili di giunto. Questi grafici vengono forniti dallo script a fine esecuzione.

Percorso triangolare





Percorso circolare