Corso di Laurea Magistrale in

Ingegneria Biomedica

Esame di

Robotica medica

Lezione del 05/10/23

Jacobiano geometrico

Esercitazione 2

Anno Accademico 2023/2024

1 Traccia

Costruire uno script MATLAB modulare che calcoli il Jacobiano geometrico di un manipolatore assegnata una specifica configurazione tramite tabella di Denavit-Hartenberg.

2 Background teorico

Assegnata una configurazione mediante la tabella di Denavit-Hartenberg, il Jacobiano geometrico è una matrice $6 \times N$ che lega la velocità dell'end effector, o della terna terminale, alle velocità dei giunti.
La colonna i-esima della matrice può essere vista in forma sintetica come composizione di due contributi:

$$\mathbf{J}_i = \begin{bmatrix} \mathbf{J}_{Pi} \\ \mathbf{J}_{Oi} \end{bmatrix} \in \mathbb{R}^6$$

dove $\mathbf{J}_{Pi} \in \mathbb{R}^3$ indica il contributo delle velocità lineari mentre $\mathbf{J}_{Oi} \in \mathbb{R}^3$ indica il contributo delle velocità angolari.
L'espressione della colonna i-esima in forma esplicita varia a seconda del tipo di giunto considerato:

$$\begin{bmatrix} \mathbf{J}_{Pi} \\ \mathbf{J}_{Oi} \end{bmatrix} = \begin{cases} \begin{bmatrix} \mathbf{z}_{i-i} \\ \mathbf{0} \end{bmatrix} & , \text{ Per un giunto } \textit{prismatico} \\ \begin{bmatrix} \mathbf{z}_{i-i} \times (\mathbf{p}_e - \mathbf{p}_{i-1}) \\ \mathbf{z}_{i-i} \end{bmatrix} & , \text{ Per un giunto } \textit{rotoidale} \end{cases} \tag{1}$$

i vettori $\mathbf{z}_{i-1}$, $\mathbf{p}_e$ e $\mathbf{p}_{i-1}$ sono funzione delle variabili di giunto. Definiti:

$$\mathbf{z}_0 = [0 \;\; 0 \;\; 1]^T$$

$$\mathbf{p}_0 = [0 \;\; 0 \;\; 0]^T$$

possiamo esprimere $\mathbf{z}_{i-1}$, $\mathbf{p}_e$ e $\mathbf{p}_{i-1}$ come:

$$\mathbf{z}_{i-1} = \mathbf{R}_1^0(q_1)...\mathbf{R}_{i-1}^{i-2}(q_{i-1})\mathbf{z}_0$$
$$\tilde{\mathbf{p}}_e = \mathbf{A}_1^0(q_1)...\mathbf{A}_n^{n-1}(q_n)\tilde{\mathbf{p}}_0$$
$$\tilde{\mathbf{p}}_{i-1} = \mathbf{A}_1^0(q_1)...\mathbf{A}_{i-1}^{i-2}(q_n)\tilde{\mathbf{p}}_0$$

dove $\sim$ indica la rappresentazione del vettore in coordinate omogenee. Il vettore $\mathbf{p}_e$ indica la posizione dell'origine della terna di riferimento solidale all'end effector nella terna di riferimento base.

In figura **1** sono mostrate le posizioni del vettore generico $\mathbf{p}$ e $\mathbf{z}$ nella matrice di trasformazione omogenea.

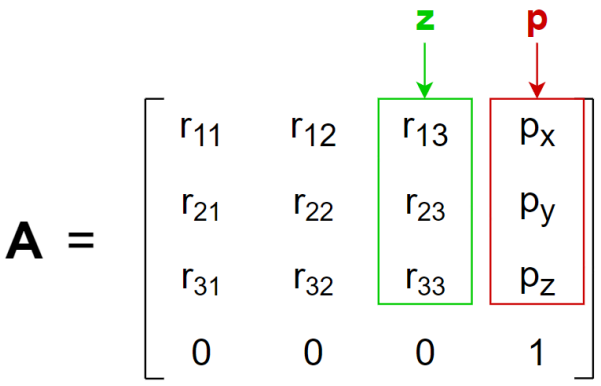

Figura 1: Posizione dei vettori $\mathbf{p}$ e $\mathbf{z}$ nella matrice di trasformazione omogenea $\mathbf{A}$.

3 Progettazione dello script

Al fine di implementare correttamente una funzione in grado di calcolare lo Jacobiano geometrico di un manipolatore a partire dalla tabella di Denavit-Hartenberg associata, è possibile utilizzare lo schema funzionale mostrato in figura **2**

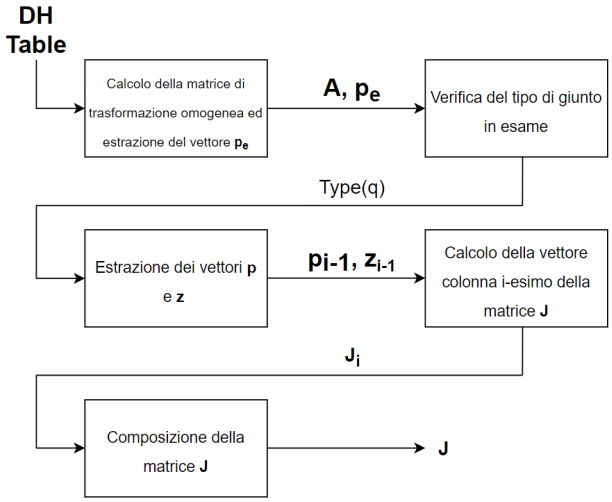

Figura 2: Workflow descrittivo della funzione *Jacobian*.

4 Implementazione

La funzione implementata (i.e. *Jacobian*) è disponibile al link GitHub a piè pagina[1].

La funzione prende in input due parametri:

- **DH**; matrice Nx4 rappresentativa della tabella di Denavit-Hartenberg.
- **jointConfiguration** (opzionale); array 1xN rappresentativo della sequenza di giunti. L'array deve essere composto da una sequenza di 0 e 1 dove 0 indica un giunto di tipo prismatico mentre 1 indica un giunto di tipo rotoidale (un manipolatore sferico è rappresentato dall'array [1 1 0]).
  Se il parametro non viene fornito viene assunto che il manipolatore sia composto unicamente da giunti rotoidali.

Fornito in input almeno il parametro *DH*, la funzione estrae il numero di link dalla dimensione della matrice DH ed effettua un check sull'esistenza della seconda variabile, creandola ed inizializzandola ad un array unitario qualora non dovesse esistere. Se il parametro *jointConfiguration* viene inserito dall'utente, la funzione effettua un check per verificare se la lunghezza del vettore è congrua al numero di righe della matrice DH (Figura **3**).

```
% Check the number of links
nLinks = length(DH(:,1));

% if 'jointConfiguration' variable is not given assumes all spherical
% joint
if ~exist('jointConfiguration', 'var')
    jointConfiguration = ones(1,nLinks);
end

% Check if the length of 'jointConfiguration' variable is equal to the
% number of links in the DH table
if length(jointConfiguration) ~= length(DH(:,1))
    disp(">>> Invalid joint configuration...")
```

Figura 3: Verifica delle variabili e check sull'integrità dei dati forniti

Effettuata questa operazione preliminare, lo script richiama la funzione *DirectKinematics* descritta nell'esercitazione 1[2] per calcolare l'insieme delle matrici di trasformazione omogenea.
Sfruttando l'ultima matrice si va ad estrarre il vettore $\mathbf{p_e}$ e si va a calcolare la prima colonna della matrice **J**, controllando anche il tipo di giunto (Figura **4**).
Infine, il processo viene ripetuto iterativamente tramite un ciclo *for* fino a coprire il numero di giunti totale ed effettuando, per ogni giunto, un check sulla tipologia (Figura **5**).

[1] https://github.com/reFraw/robotica-medica/blob/main/Esercitazioni/Esercitazione_2/Script/Jacobian.m
[2] https://github.com/reFraw/robotica-medica/blob/main/Esercitazioni/Esercitazione_1/Script/DirectKinematics.m

```
% Calculate the homogeneous transformation matrix from base
% triplets to i-th triplets
T0 = DirectKinematics(DH);

% extract the (A^0_e) matrix and calculate the position of (O^0_e)
A0_N = T0(:,:,nLinks);
pe = A0_N(1:3, 4);

% Set value for p0 and z0 and initialize the Jacobian matrix
p0 = [0 0 0]';
z0 = [0 0 1]';

J = zeros(6, nLinks);

% Calculate the first column of the Jacobian matrix
if jointConfiguration(1) == 1
    firstColumn = [cross(z0, (pe-p0)); z0];
    J(:,1) = firstColumn;

else
    J(:,1) = [z0' 0 0 0]';

end
```

Figura 4: Calcolo della prima colonna della matrice **J**.

```
% Calculate the remaining columns of the Jacobian matrix
for i = 2 : nLinks
    A = T0(:,:,i-1);

    p = A(1:3, 4);
    z = A(1:3, 3);

    if jointConfiguration(i) == 1
        JPi = cross(z, (pe-p));
        JOi = z;

        Ji = [JPi; JOi];

    else
        Ji = [z; 0; 0; 0];

    end

    J(:,i) = Ji;

end
```

Figura 5: Calcolo delle rimanenti colonne della matrice **J**.