Procedure Maxima,Processing,MATLAB - Robotica Industriale

Matricola: 0301285

CFU: 12

Documento di riconoscimento:

Nessuna procedura è stata svolta in collaborazione con altri studenti.
Vi è stato un confronto dei risultati ottenuti nelle procedure della cinematica inversa dei robot e del calcolo dell' energia cinetica con:

-Andrea Efficace

MATRICI DI ROTAZIONE $R_x(\theta_x), R_y(\theta_y), R_z(\theta_z)$

Notazione per esprimere le coordinate di un punto P durante una rotazione nel piano:

$P :=$ coordinate del punto dopo la rotazione

$\hat{P} :=$ coordinate del punto prima della rotazione

$R_k(\theta_k) :=$ matrice di rotazione rispetto all'asse $k \epsilon \{x, y, z\}$ di un angolo θ_k

Matrice di rotazione attorno all'asse x $R_x(\theta_x)$

Nella matrice di rotazione attorno all'asse x, i punti lungo l'asse x rimangono invariati mentre gli assi y e z rappresentano rispettivamente l'ascissa e l'ordinata di una rotazione nel piano.

La prima riga e la prima colonna sono uguali ad $\begin{pmatrix} 1 & 0 & 0 \end{pmatrix}$ e $\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$ poiché il piano che ruota è yz e la coordinata che rimane invariata è la x.
Sussistono le seguenti relazioni relazioni:

$$x = \hat{x}$$

$$\begin{pmatrix} y \\ z \end{pmatrix} = R(\theta) \begin{pmatrix} \hat{y} \\ \hat{z} \end{pmatrix} = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \begin{pmatrix} \hat{y} \\ \hat{z} \end{pmatrix}$$

Da cui:

$$\begin{aligned} & x = \hat{x}, \\ & y = c\,\mathrm{os}(\theta)\,\hat{y} - \sin(\theta)\,\hat{z} \\ & z = \sin(\theta)\,\hat{y} + \cos(\theta)\hat{z} \end{aligned}$$

```
(%i1) R[x](theta) := matrix([1,0,0],
                            [0,cos(theta),-sin(theta)],
                            [0,sin(theta), cos(theta)]);
```

(%o1) $R_x(\vartheta) := \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\vartheta) & -\sin(\vartheta) \\ 0 & \sin(\vartheta) & \cos(\vartheta) \end{pmatrix}$

```
(%i4) R[x](theta[x]);
```

(%o4) $\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\vartheta_x) & -\sin(\vartheta_x) \\ 0 & \sin(\vartheta_x) & \cos(\vartheta_x) \end{pmatrix}$

Matrice di rotazione attorno all'asse y $R_y(\theta_y)$

Nella matrice di rotazione attorno all'asse y, i punti lungo l'asse y rimangono invariati mentre gli assi z e y rappresentano rispettivamente l'ascissa e l'ordinata di una rotazione del piano.

La seconda riga e la seconda colonna sono rispettivamente uguali a $\begin{pmatrix} 0 & 1 & 0 \end{pmatrix}$ e $\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$ poiché il piano che ruota è zx e la coordinata che rimane sempre inviariata è la y.

Sussistono le seguenti relazioni:

$$y = \hat{y}$$

$$\begin{pmatrix} z \\ x \end{pmatrix} = R(\theta) \begin{pmatrix} \hat{z} \\ \hat{x} \end{pmatrix} = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \begin{pmatrix} \hat{z} \\ \hat{x} \end{pmatrix}$$

Dai cui:

$$y = \hat{y}$$
$$z = \cos(\theta)\,\hat{z} - \sin(\theta)\,\hat{x}$$
$$x = \sin(\theta)\hat{z} + \cos(\theta)\,\hat{x}$$

```
(%i1) R[y](theta) := matrix([cos(theta),0,sin(theta)],
                            [0,1,0],
                            [-sin(theta),0, cos(theta)]);
```

(%o1) $R_y(\vartheta) := \begin{pmatrix} \cos(\vartheta) & 0 & \sin(\vartheta) \\ 0 & 1 & 0 \\ -\sin(\vartheta) & 0 & \cos(\vartheta) \end{pmatrix}$

```
(%i2) R[y](theta[y])
```

(%o2) $\begin{pmatrix} \cos(\vartheta_y) & 0 & \sin(\vartheta_y) \\ 0 & 1 & 0 \\ -\sin(\vartheta_y) & 0 & \cos(\vartheta_y) \end{pmatrix}$

```
(%i3)
```

Matrice di rotazione attorno all'asse z $R_z(\theta_z)$

Nella matrice di rotazione attorno all'asse z, i punto lungo l'asse z rimangono invariati mentre gli assi x e y rappresentano rispettivamente l'ascissa e l'ordinata di una rotazione del piano.

La terza riga e la seconda colonna sono rispettivamente uguali a $\begin{pmatrix} 0 & 0 & 1 \end{pmatrix}$ e $\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$ poiché il piano che ruota è xy e la coordinata che rimane sempre inviariata è la z.

Sussistono le seguenti relazioni:

$$z = \hat{z}$$
$$\begin{pmatrix} x \\ y \end{pmatrix} = R(\theta)\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix}$$

Dai cui:

$$z = \hat{z}$$
$$x = \cos(\theta)\,\hat{x} - \sin(\theta)\,\hat{y}$$
$$y = \sin(\theta)\hat{x} + \cos(\theta)\,\hat{y}$$

```
(%i5) R[z](theta) := matrix([cos(theta),-sin(theta),0],
                            [sin(theta),cos(theta),0],
                            [0,0, 1]);
```

(%o5) $R_z(\vartheta) := \begin{pmatrix} \cos(\vartheta) & -\sin(\vartheta) & 0 \\ \sin(\vartheta) & \cos(\vartheta) & 0 \\ 0 & 0 & 1 \end{pmatrix}$

```
(%i6) R[z](theta[z])
```

(%o6) $\begin{pmatrix} \cos(\vartheta_z) & -\sin(\vartheta_z) & 0 \\ \sin(\vartheta_z) & \cos(\vartheta_z) & 0 \\ 0 & 0 & 1 \end{pmatrix}$

(%i7)

Procedura matrice di rotazione tramite asse e un angolo

Scrivere una procedura che prenda in input:

- un asse $k\epsilon\{x,y,z\}$;

-un angolo ϑ simbolico o numerico;

e in output si produca la corrispettiva matrice di rotazione.

Nel caso in cui venga immesso un asske $k\notin\{x,y,z\}$ si mostri un messaggio di errore.

Al fine di definire una funzione che può essere richiamata utilizziamo l'istruzione block composta da due campi. Il primo, all'interno di [], viene utilizzato per salvare il risultato della procedura; il secondo si scrive l'implementazione effettiva della procedura che, in questo caso, sarà composta da una combinazione di if, elseif e else per verificare la corretta immissione dell'asse k.

```
Maxima 5.44.0 http://maxima.sourceforge.net
using Lisp SBCL 2.0.0
Distributed under the GNU Public License. See the file COPYING.
Dedicated to the memory of William Schelter.
The function bug_report() provides bug reporting information.
(%i10) R(k,theta):= block([res],
                          if k = x
                              then res:matrix([1,0,0],
                             [0,cos(theta),-sin(theta)],
                             [0,sin(theta), cos(theta)])
                          elseif k = y
                              then res:matrix([cos(theta),0,sin(theta)],
                             [0,1,0],
                             [-sin(theta),0, cos(theta)])
                          elseif k = z
                              then res:matrix([cos(theta),-sin(theta),0],
                             [sin(theta),cos(theta),0],
                             [0,0, 1])
                           else
                             res:"Incorrect axis of rotation"
                           )
```

(%o10) $R(k,\vartheta):=\mathbf{block}\left([\mathrm{res}],\mathbf{if}\ k=x\ \mathbf{then}\ \mathrm{res}:\begin{pmatrix}1&0&0\\0&\cos(\vartheta)&-\sin(\vartheta)\\0&\sin(\vartheta)&\cos(\vartheta)\end{pmatrix}\mathbf{elseif}\ k=y\ \mathbf{then}\ \mathrm{res}:\begin{pmatrix}\cos(\vartheta)&0&\sin(\vartheta)\\0&1&0\\-\sin(\vartheta)&0&\cos(\vartheta)\end{pmatrix}\mathbf{elseif}\ k=z\ \mathbf{then}\ \mathrm{res}:\begin{pmatrix}\cos(\vartheta)&-\sin(\vartheta)&0\\\sin(\vartheta)&\cos(\vartheta)&0\\0&0&1\end{pmatrix}\mathbf{else}\ \mathrm{res}:\text{Incorrect axis of rotation}\right)$

Immissione in input dei tre assi ed angolo numerico:

```
(%i11) R(x, %pi/2)
```

(%o11) $\begin{pmatrix}1&0&0\\0&0&-1\\0&1&0\end{pmatrix}$

```
(%i12) R(y,%pi/2)
```

(%o12) $\begin{pmatrix}0&0&1\\0&1&0\\-1&0&0\end{pmatrix}$

```
(%i13) R(z,%pi/2)
```

$$\text{(\%o13)} \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Immissione in input dei tre assi e angolo simbolico

```
(%i15) R(x,theta[x])
```

$$\text{(\%o15)} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\vartheta_x) & -\sin(\vartheta_x) \\ 0 & \sin(\vartheta_x) & \cos(\vartheta_x) \end{pmatrix}$$

```
(%i16) R(y,theta[y])
```

$$\text{(\%o16)} \begin{pmatrix} \cos(\vartheta_y) & 0 & \sin(\vartheta_y) \\ 0 & 1 & 0 \\ -\sin(\vartheta_y) & 0 & \cos(\vartheta_y) \end{pmatrix}$$

```
(%i17) R(z,theta[z])
```

$$\text{(\%o17)} \begin{pmatrix} \cos(\vartheta_z) & -\sin(\vartheta_z) & 0 \\ \sin(\vartheta_z) & \cos(\vartheta_z) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Caso di immissione di un asse non corretto

```
(%i18) R(q,%pi/2)
```

(%o18) Incorrect axis of rotation

```
(%i19)
```

Dimostrazione proprietà commutativa matrici di rotazione

Dimostrare tramite una procedura maxima che le matrici di rotazione non commutano.In generale:

$$R_x(\alpha)\,R_y(\beta) \neq R_y(\beta)\,R_x(\alpha)$$

Quindi, verifichiamo che:

$$R_x(\alpha)\,R_y(\beta) - R_y(\beta)\,R_x(\alpha) \equiv 0$$

Se $R_x(\alpha)\,R_y(\beta) - R_y(\beta)\,R_x(\alpha) = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$ $\rightarrow$la matrici di rorazione non commutano $\rightarrow$OK

Altrimenti $\rightarrow$le matrici di rotazioni commutano $\rightarrow$ERRORE

```
Maxima 5.44.0 http://maxima.sourceforge.net
using Lisp SBCL 2.0.0
Distributed under the GNU Public License. See the file COPYING.
Dedicated to the memory of William Schelter.
The function bug_report() provides bug reporting information.
```

```
(%i1) prop(alpha,beta):=block([res],
                              R[y](beta) := matrix([cos(beta),0,sin(beta)],
                             [0,1,0],
                             [-sin(beta),0, cos(beta)]),
                             R[x](alpha) := matrix([1,0,0],
                             [0,cos(alpha),-sin(alpha)],
                             [0,sin(alpha), cos(alpha)]),
                              z:R[x](alpha).R[y](beta)-R[y](beta).R[x](alpha),
                              nullMat:matrix([0,0,0],[0,0,0],[0,0,0]),
                              if z=nullMat
                                 then res:"Ok,le matrici non commutano"
                              else
                                 res:"Errore, la proprietà commutativa fra le
      matrici di rotazione non è valida"

                                )
```

(%o1) $\mathrm{prop}(\alpha,\beta) := \mathbf{block}\left([\mathrm{res}], R_y(\beta) := \begin{pmatrix} \cos(\beta) & 0 & \sin(\beta) \\ 0 & 1 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) \end{pmatrix}, R_x(\alpha) := \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) \\ 0 & \sin(\alpha) & \cos(\alpha) \end{pmatrix}, z{:}\, R_x(\alpha) \cdot R_y(\beta) - R_y(\beta) \cdot R_x(\alpha), \mathrm{nullMat}{:} \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \mathbf{if}\ z = \mathrm{nullMat}\ \mathbf{then}\ \mathrm{res}{:}\, \text{Ok,le matrici non commutano}\ \mathbf{else}\ \mathrm{res}{:}\, \text{Errore, la proprietà commutativa fra le matrici di rotazione non è valida} \right)$

```
(%i2) prop(alpha,beta)
```

(%o2) Ok,le matrici non commutano

(%i3)

Procedura 4

Dimostrare che:

$$(1)\ R_z(\gamma) = R_x\left(\pm\frac{\pi}{2}\right) R_y(\gamma)\, R_x\left(\mp\frac{\pi}{2}\right) \quad \forall\gamma$$

Occorre dimostrare che tramite la rappresentazione delle rotazioni nello spazio tramite angoli di Eulero è possibile ottentere un qualsiasi orientamento arbitrario.

Ricordiamo, inoltre, che le rotazioni effettuate tramite angoli di Eulero prendono in considerazione solamente rotazioni attorno all'asse x e y.

Quindi il problema si riconduce a quello di dimostrare che è possibile ottenere un qualsiasi orientamento dell'asse z tramite rotazioni dei restanti due assi. A tale scopo, procediamo con il determinare la matrice di rotazione $R_z(\gamma)$, $R_x\left(+\frac{\pi}{2}\right) R_y(\gamma)\, R_x\left(-\frac{\pi}{2}\right)$,$R_x\left(-\frac{\pi}{2}\right) R_y(\gamma)\, R_x\left(+\frac{\pi}{2}\right)$. Le ultime due matrici sottratte alla matrice $R_z(\gamma)$ devono avere come risultato la matrice nulla. Data l'arbitrarietà dell'angolo γ, la (1) è dimostrata.

R(k,theta) è una funzione per il calcolo delle matrici $R_z(\gamma)$, $R_x\left(\pm\frac{\pi}{2}\right)$, $R_y(\gamma)$, $R_x\left(\mp\frac{\pi}{2}\right)$

```
(%i1) R(k,theta):= block([res],
                       if k = x
                           then res:matrix([1,0,0],
                          [0,cos(theta),-sin(theta)],
                          [0,sin(theta), cos(theta)])
                       elseif k = y
                           then res:matrix([cos(theta),0,sin(theta)],
                          [0,1,0],
                          [-sin(theta),0, cos(theta)])
                       elseif k = z
                           then res:matrix([cos(theta),-sin(theta),0],
                          [sin(theta),cos(theta),0],
                          [0,0, 1])
                        else
                          res:"Incorrect axis of rotation"
                        )
```

(%o1) $R(k,\vartheta) := \mathbf{block}\left([\mathrm{res}], \mathbf{if}\ k = x\ \mathbf{then}\ \mathrm{res}: \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\vartheta) & -\sin(\vartheta) \\ 0 & \sin(\vartheta) & \cos(\vartheta) \end{pmatrix} \mathbf{elseif}\ k = y\ \mathbf{then}\ \mathrm{res}: \begin{pmatrix} \cos(\vartheta) & 0 & \sin(\vartheta) \\ 0 & 1 & 0 \\ -\sin(\vartheta) & 0 & \cos(\vartheta) \end{pmatrix} \mathbf{elseif}\ k = z\ \mathbf{then}\ \mathrm{res}: \begin{pmatrix} \cos(\vartheta) & -\sin(\vartheta) & 0 \\ \sin(\vartheta) & \cos(\vartheta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \mathbf{else}\ \mathrm{res}: \text{Incorrect axis of rotation}\right)$

Matrice $R_z(\gamma)$

```
(%i2) R[z]:R(z, gamma);
```

(%o2) $\begin{pmatrix} \cos(\gamma) & -\sin(\gamma) & 0 \\ \sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 1 \end{pmatrix}$

Matrice $R_x\left(\frac{\pi}{2}\right)$

```
(%i3) R[x]:R(x,%pi/2);
```

(%o3) $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}$

Matrice $R_x\left(-\frac{\pi}{2}\right)$

```
(%i4) R1[x]:R(x, -(%pi)/2);
```

(%o4) $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{pmatrix}$

Matrice $R_y(\gamma)$

```
(%i5) R[y]:R(y,gamma);
```

(%o5) $\begin{pmatrix} \cos(\gamma) & 0 & \sin(\gamma) \\ 0 & 1 & 0 \\ -\sin(\gamma) & 0 & \cos(\gamma) \end{pmatrix}$

Procedura per la verifica di $R_z(\gamma) - R_x\left(\pm\frac{\pi}{2}\right) R_y(\gamma) R_x\left(\mp\frac{\pi}{2}\right) \equiv 0$

```
(%i6) proc(z,x,y,x1):=block([res],
                              m: z-x.y.x1,
                              nullMat: matrix([0,0,0],[0,0,0],[0,0,0]),
                              if(m = nullMat)
                                  then res:"La proprietà è verificata"
                              else
                                 res:"La proprietà non è verificata"
                                 )
```

(%o6) $\operatorname{proc}(z, x, y, x1) := \mathbf{block}\left([\mathrm{res}], m{:}\, z - x \cdot y \cdot x1, \mathrm{nullMat}{:} \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \mathbf{if}\ m = \mathrm{nullMat}\ \mathbf{then}\ \mathrm{res}{:}\ \text{La proprietà è verificata}\ \ \mathbf{else}\ \mathrm{res}{:}\ \text{La proprietà non è verificata} \right)$

Verifica di $R_z(\gamma) - R_x\left(+\frac{\pi}{2}\right) R_y(\gamma) R_x\left(-\frac{\pi}{2}\right) \equiv 0 \quad \forall\gamma$

```
(%i7) proc(R[z],R[x],R[y],R1[x]);
```

(%o7) La proprietà è verificata

Matrice $R_y(-\gamma)$

```
(%i8) R1[y]:R(y,-gamma);
```

(%o8) $\begin{pmatrix} \cos(\gamma) & 0 & -\sin(\gamma) \\ 0 & 1 & 0 \\ \sin(\gamma) & 0 & \cos(\gamma) \end{pmatrix}$

Verifica di $R_z(\gamma) + R_x\left(-\frac{\pi}{2}\right) R_y(-\gamma) R_x\left(+\frac{\pi}{2}\right) \equiv 0 \quad \forall\gamma$

```
(%i9) proc(R[z],R1[x],R1[y],R[x]);
```

(%o9) La proprietà è verificata

```
(%i13) R(x,alpha).R(y,beta).R(x,gamma)
```

(%o13) $(\cos(\beta), \sin(\beta)\sin(\gamma), \sin(\beta)\cos(\gamma); \sin(\alpha)\sin(\beta), \cos(\alpha)\cos(\gamma) - \sin(\alpha)\cos(\beta)\sin(\gamma), -\cos(\alpha)\sin(\gamma) - \sin(\alpha)\cos(\beta)\cos(\gamma); -\cos(\alpha)\sin(\beta), \cos(\alpha)\cos(\beta)\sin(\gamma) + \sin(\alpha)\cos(\gamma), \cos(\alpha)\cos(\beta)\cos(\gamma) - \sin(\alpha)\sin(\gamma))$

```
(%i14)
```

$$\begin{pmatrix} \cos(\beta) & \sin(\beta)\sin(\gamma) & \sin(\beta)\cos(\gamma) \\ \sin(\alpha)\sin(\beta) & \cos(\alpha)\cos(\gamma) - \sin(\alpha)\cos(\beta)\sin(\gamma) & -\cos(\alpha)\sin(\gamma) - \sin(\alpha)\cos(\beta)\cos(\gamma) \\ -\cos(\alpha)\sin(\beta) & \cos(\alpha)\cos(\beta)\sin(\gamma) + \sin(\alpha)\cos(\gamma) & \cos(\alpha)\cos(\beta)\cos(\gamma) - \sin(\alpha)\sin(\gamma) \end{pmatrix}$$

Procedura 5: matrici di rotazione tramite trasformata di Laplace

Scrivere una procedura che utilizzando la trasformata di Laplace calcoli la matrice di rotazione intorno all'asse rappresentata dal versore v, di un angolo ϑ

$$R_v(\theta) = e^{S(v)\theta}$$

Per il calcolo della matrice di rotazione tramite la trasformata di Laplace occore seguire i seguenti passi:

1) Calcolo della matrice $S(k)$ con $k \in \{e_x, e_y, e_z\}$ con e_x, e_y, e_z versori dei rispettivi assi x, y, z:

$$S(e_x) = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}; S(e_y) = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix}; S(e_z) = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

2)Calcolo della matrice $s\,I - S(k)$ di Laplace con s variabile di Laplace e I matrice identità $I \in R^{3x3}$:

$$s\,I - S(e_x) = \begin{pmatrix} s & 0 & 0 \\ 0 & s & -1 \\ 0 & 1 & s \end{pmatrix}; s\,I - S(e_y) = \begin{pmatrix} s & 0 & 1 \\ 0 & s & 0 \\ -1 & 0 & s \end{pmatrix}; s\,I - S(e_z) = \begin{pmatrix} s & -1 & 0 \\ 1 & s & 0 \\ 0 & 0 & s \end{pmatrix}.$$

3)Invertire le matrici appena ottenute:

$$(s\,I - S(e_x))^{-1} = \begin{pmatrix} \frac{1}{s} & 0 & 0 \\ 0 & \frac{s}{s^2+1} & \frac{1}{s^2+1} \\ 0 & \frac{1}{s^2+1} & \frac{s}{s^2+1} \end{pmatrix};$$

$$(s\,I - S(e_x))^{-1} = \begin{pmatrix} \frac{s^2}{s^3+s} & 0 & \frac{s}{s^3+s} \\ 0 & \frac{s^2+1}{s^3+s} & 0 \\ -\frac{s}{s^3+s} & 0 & \frac{s^2}{s^3+s} \end{pmatrix};$$

$$(s\,I - S(e_x))^{-1} = \begin{pmatrix} \frac{s^2}{s^3+s} & -\frac{s}{s^3+s} & 0 \\ \frac{s}{s^3+s} & \frac{s^2}{s^3+s} & 0 \\ 0 & 0 & \frac{s^2+1}{s^3+s} \end{pmatrix}.$$

4)Calcolare l'inversa di Laplace della matrici inverse $\mathcal{L}\{(s\,I - S(k))^{-1}\}^{-1}$:

$$R_x(\theta) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\vartheta) & -\sin(\vartheta) \\ 0 & \sin(\vartheta) & \cos(\vartheta) \end{pmatrix}; R_y(\theta) = \begin{pmatrix} \cos(\vartheta) & 0 & \sin(\vartheta) \\ 0 & 1 & 0 \\ -\sin(\vartheta) & 0 & \cos(\vartheta) \end{pmatrix}; R_z(\theta) = \begin{pmatrix} \cos(\vartheta) & -\sin(\vartheta) & 0 \\ \sin(\vartheta) & \cos(\vartheta) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

La funzione inverseLaplace(SI) calcola e restituisce in output l'antitrasformata di Laplace scorrendo tutti gli elementi della matrice data in input

```
(%i1) inverseLaplace(SI):=block([res],
                              M:SI,
                              MC:SI,
                              for i:1 thru 3 do(
                                for j:1 thru 3 do
                                   (
                                     a:M[i,j],
                                     b:ilt(a,s,theta),
                                     MC[i,j]:b
                                    )
                                ),
                              res:MC

                           )
```

(%o1) inverseLaplace(SI) := **block** ([res], M: SI, MC: SI, **for** i **thru** 3 **do for** j **thru** 3 **do** (a: $M_{i,j}$,

b: ilt(a, s, ϑ), MC$_{i,j}$: b), res: MC)

La funzione rotLaplace(k) riceve in input un vettore v e calcola 1), 2), 3). In seguito, invoca la funzione inverseLaplace per effettuare 4) e, quindi, restituire in output l'effettiva matrice di rotazione corrispondente al versoe in input.

```
(%i2) rotLaplace(k):=block([res],
                          S:ident(3),
                          I:ident(3),
                        for i:1 thru 3 do
                          (
                          for j:1 thru 3 do
                             (
                               if i=j
                                  then S[i][j]:0
                               elseif j>i
                                  then (
                                 temp:(-1)^(j-i)*k[3-remainder(i+j,3)],
                                        S[i][j]:temp,
                                        S[j][i]:-temp

                                         )
                               )
                           ),
                          res:inverseLaplace(invert(s*I-S))

                        )
```

(%o2) rotLaplace(k) := **block** ([res], S: ident(3), I: ident(3), **for** i **thru** 3 **do for** j **thru** 3 **do if** $i = j$ **then** $(S_i)_j$: 0 **elseif** $j > i$ **then** (temp: $(-1)^{j-i}\, k_{3-\mathrm{remainder}(i+j,3)}$, $(S_i)_j$: temp, $(S_j)_i$: $-$temp), res: inverseLaplace(invert$(s\, I - S)$))

Matrice di rotazione $R_x(\theta)$:

```
(%i3) R[x](theta):=rotLaplace([1,0,0]);
```

(%o3) $R_x(\vartheta)$:= rotLaplace([1, 0, 0])

```
(%i4) R[x](theta);
```

(%o4) $$\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\vartheta) & -\sin(\vartheta) \\ 0 & \sin(\vartheta) & \cos(\vartheta) \end{pmatrix}$$

Matrice di rotazione $R_y(\theta)$:

```
(%i5) R[y](theta):=rotLaplace([0,1,0]);
```

(%o5) $R_y(\vartheta)$:= rotLaplace([0, 1, 0])

```
(%i6) R[y](theta);
```

(%o6) $$\begin{pmatrix} \cos(\vartheta) & 0 & \sin(\vartheta) \\ 0 & 1 & 0 \\ -\sin(\vartheta) & 0 & \cos(\vartheta) \end{pmatrix}$$

Matrice di rotazione $R_z(\theta)$:

(%i7) `R[z](theta):=rotLaplace([0,0,1]);`

(%o7) $R_z(\vartheta) := \mathrm{rotLaplace}([0,0,1])$

(%i8) `R[z](theta);`

$$\text{(\%o8)} \quad \begin{pmatrix} \cos(\vartheta) & -\sin(\vartheta) & 0 \\ \sin(\vartheta) & \cos(\vartheta) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

(%i9)

Procedura 6: calcolo matrici di rotazioni tramite autovalori/autovettori

Scrivere una procedura che utilizzando gli autovalori/autovettori calcoli la matrice di rotazione intorno all'asse rappresentata dal versore v, di un angolo ϑ:

$$S(v) = V.\Lambda.V^{-1}$$

$$R_v(\theta) = V.e^{\Lambda\theta}.V^{-1}$$

In particolare, i passi da seguire sono i seguenti:

1) $S(v) = V.\Lambda.V^{-1}$ con V:= matrice degli auto vettori come colonne

Λ:= matrice diagonale con gli autovali nella diagonale principale

2) $e^{S(v)\theta} = Ve^{\Lambda\theta}V^{-1}$ con $e^{\Lambda\theta} :=$ matrice diagonale

3) $R_v(\theta) = V.e^{\Lambda\theta}.V^{-1}$

La funzione matRot prende in input una matrice S(v) e ne calcola autovalori, disposti nella diagonale principale della matrice Λ, e autovettori disposti come colonne nella matrice V.
In seguito, la variabile matExp salva la matrice esponenziale del punto 2).
Infine restituisce in output la matrice di rotazione del punto 3):

$$R_v(\theta) = V.e^{\Lambda\theta}.V^{-1}$$

```
(%i1) matRot(M):=block([V,matExp,eigVect,Lambda,res],
                 eigVect:eigenvectors(M),
                 V:transpose(matrix(eigVect[2][1][1],eigVect[2][2][1],
      eigVect[2][3][1])),
                 Lambda:matrix([eigVect[1][1][1],0,0],[0,eigVect[1][1][2],0],[0,
      0,eigVect[1][1][3]]),
                 matExp:matrixexp(Lambda*theta),
                 res:expand(demoivre(expand(V.matExp.invert(V))))
                 )
```

(%o1) $\mathrm{matRot}(M) := \mathbf{block}\left([V, \mathrm{matExp}, \mathrm{eigVect}, \Lambda, \mathrm{res}], \mathrm{eigVect}: \mathrm{eigenvectors}(M), V: \mathrm{transpose}\left(\left(\begin{array}{cc} (\mathrm{eigVect}_2)_1 & 1 \\ (\mathrm{eigVect}_2)_2 & 1 \\ (\mathrm{eigVect}_2)_3 & 1 \end{array}\right)\right), \Lambda: \left(\begin{array}{ccc} ((\mathrm{eigVect}_1)_1)_1 & 0 & 0 \\ 0 & ((\mathrm{eigVect}_1)_1)_2 & 0 \\ 0 & 0 & ((\mathrm{eigVect}_1)_1)_3 \end{array}\right), \mathrm{matExp}: \mathrm{matrixexp}(\Lambda\,\vartheta), \mathrm{res}: \mathrm{expand}(\mathrm{demoivre}(\mathrm{expand}(V \cdot \mathrm{matExp} \cdot \mathrm{invert}(V))))\right)$

La funzione rotEigen(k) riceve in input il versore e_x, e_y, e_z ed invoca matRot(M) per il calcolo della matrice di rotazione corrispondente. Restituisce in output l'effettiva matrice di rotazione corrispondente al versoe in input.
Altrimenti, se il versore inserito è diverso da e_x, e_y, e_z restituisce errore.

```
(%i2) rotEigen(k):=block([res],
                  S:ident(3),
               for i:1 thru 3 do
                    (
                    for j:1 thru 3 do
                       (
                        if i=j
                           then S[i][j]:0
                        elseif j>i
                           then (
                          temp:(-1)^(j-i)*k[3-remainder(i+j,3)],
                                 S[i][j]:temp,
                                 S[j][i]:-temp

                                  )
                        )
                     ),
               res:matRot(S)
               )
```

(%o2) $\mathrm{rotEigen}(k) := \mathbf{block}\ ([\mathrm{res}], S\colon \mathrm{ident}(3), \mathbf{for}\ i\ \mathbf{thru}\ 3\ \mathbf{do}\ \mathbf{for}\ j\ \mathbf{thru}\ 3\ \mathbf{do}\ \mathbf{if}\ i = j\ \mathbf{then}\ (S_i)_j\colon 0\ \mathbf{elseif}\ j > i\ \mathbf{then}\ (\mathrm{temp}\colon (-1)^{j-i} k_{3-\mathrm{remainder}(i+j,3)}, (S_i)_j\colon \mathrm{temp}, (S_j)_i\colon -\mathrm{temp}), \mathrm{res}\colon \mathrm{matRot}(S))$

Matrice di rotazione $R_x(\theta)$:

```
(%i3) R[x](theta):=rotEigen([1,0,0]);
```

(%o3) $R_x(\vartheta) := \mathrm{rotEigen}([1, 0, 0])$

```
(%i4) R[x](theta);
```

Proviso: assuming 4*theta # 0(%o4) $\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\vartheta) & -\sin(\vartheta) \\ 0 & \sin(\vartheta) & \cos(\vartheta) \end{pmatrix}$

Matice di rotazione $R_y(\theta)$:

```
(%i5) R[y](theta):=rotEigen([0,1,0]);
```

(%o5) $R_y(\vartheta) := \mathrm{rotEigen}([0, 1, 0])$

```
(%i6) R[y](theta);
```

Proviso: assuming 4*theta # 0

(%o6) $\begin{pmatrix} \cos(\vartheta) & 0 & \sin(\vartheta) \\ 0 & 1 & 0 \\ -\sin(\vartheta) & 0 & \cos(\vartheta) \end{pmatrix}$

Matrice di rotazione $R_z(\theta)$:

```
(%i7) R[z](theta):=rotEigen([0,0,1]);
```

(%o7) $R_z(\vartheta) := \mathrm{rotEigen}([0, 0, 1])$

```
(%i8) R[z](theta);
```

Proviso: assuming 4*theta # 0(%o8) $\begin{pmatrix} \cos(\vartheta) & -\sin(\vartheta) & 0 \\ \sin(\vartheta) & \cos(\vartheta) & 0 \\ 0 & 0 & 1 \end{pmatrix}$

(%i9)

Procedura 7: il prodotto vettoriale = prodotto matriciale

Scrivere una procedura che calcoli il prodotto vettoriale $v \times w$ come prodotto matriciale

$$v \times w = S(v).w$$

In particolare, vi è una corrispondenza biunivoca tra il vettore v e la matrice antisimmetrica S(v):

$$v = \begin{pmatrix} a \\ b \\ c \end{pmatrix} \Leftrightarrow S(v) = \begin{pmatrix} 0 & -c & b \\ c & 0 & -a \\ -b & a & 0 \end{pmatrix}$$

Ipotizzando che:

$$v = a\, e_x + b\, e_y + c\, e_z$$

$$w = \alpha\, e_x + \beta\, e_y + \gamma\, e_z$$

Il prodotto vettoriale $v \times w$ è pari a:

$$v \times w = v \times (\alpha\, e_x + \beta\, e_y + \gamma\, e_z) = \alpha\, v \times e_x + \beta\, v \times e_y + \gamma\, v \times e_z$$

Che in forma matriciale:

$$v \times w = \begin{pmatrix} \vdots & \vdots & \vdots \\ v \times e_x & v \times e_y & v \times e_z \\ \vdots & \vdots & \vdots \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \\ \gamma \end{pmatrix} = S(v).w$$

In cui:

$$v \times e_x = c\, e_y - b\, e_z = \begin{pmatrix} 0 \\ c \\ -b \end{pmatrix}; v \times e_y = -c\, e_x + a\, e_z = \begin{pmatrix} -c \\ 0 \\ a \end{pmatrix}; v \times e_z = b\, e_x - a\, e_y = \begin{pmatrix} b \\ -a \\ 0 \end{pmatrix};$$

Quindi:

$$S(v) = \begin{pmatrix} 0 & -c & b \\ c & 0 & -a \\ -b & a & 0 \end{pmatrix}$$

La funzione vect(v,w) prende in input due vettori $s, t \in \mathbb{R}^3$ e restituisce in output il prodotto vettoriale $v \times w$ effettuato tramite la matrice S antisimmetrica, popolata da due cicli for.
In particolare, gli indici $s_{1,3-\text{remainder}(i+j,3)}$ permetteno di scegliere ed inserire correttamente all'interno della matrice S i valori a, b, c al fine di ottenere:
$S(v) = \begin{pmatrix} 0 & -c & b \\ c & 0 & -a \\ -b & a & 0 \end{pmatrix}$
Infine, viene effettuato il prodotto punto S.t e restituito in output tramite la variabile res.

```
(%i12) vect(v,w):=block([res],

                                S:ident(3),
                                for i:1 thru 3 do
                                  (
                                  for j:1 thru 3 do
                                     (
                                       if i=j
                                          then S[i][j]:0
                                       elseif j>i
                                          then (
                                         temp:(-1)^(j-i)*v[1][3-remainder(i+j,3)],
                                                S[i][j]:temp,
                                                S[j][i]:-temp

                                                 )
                                       )
                                   ),


                                 res:S.w
                              )
```

(%o12) $\text{vect}(v, w) := \textbf{block}\ ([\text{res}], S\colon \text{ident}(3), \textbf{for}\ i\ \textbf{thru}\ 3\ \textbf{do for}\ j\ \textbf{thru}\ 3\ \textbf{do if}\ i = j\ \textbf{then}\ (S_i)_j\colon 0\ \textbf{elseif}\ j > i\ \textbf{then}\ (\text{temp}\colon (-1)^{j-i}\,(v_1)_{3-\text{remainder}(i+j,3)}, (S_i)_j\colon \text{temp}, (S_j)_i\colon -\text{temp}), \text{res}\colon S\cdot w)$

```
(%i13) v:matrix([a,b,c]);
```

(%o13) $\begin{pmatrix} a & b & c \end{pmatrix}$

```
(%i14) w:matrix([alpha,beta,gamma]);
```

(%o14) $\begin{pmatrix} \alpha & \beta & \gamma \end{pmatrix}$

```
(%i15) vectMatrix:vect(v,w);
```

(%o15) $\begin{pmatrix} b\,\gamma - \beta\,c \\ \alpha\,c - a\,\gamma \\ a\,\beta - \alpha\,b \end{pmatrix}$

Prodotto vettoriale di Maxima

```
(%i16) load(vect)
```

vect: warning: removing existing rule or rules for ".".

```
(%i16) e:[a,b,c];
```

(%o16) C:/Maxima/bin/../share/maxima/5.44.0/share/vector/vect.mac

```
(%i18) d:[alpha,beta,gamma];
```

(%o18) $[\alpha, \beta, \gamma]$

(%i19) e~d;

(%o19) $([a, b, c], [\alpha, \beta, \gamma])$

(%i20) vectMaxima:express(%)

(%o20) $[b\,\gamma - \beta\,c, \alpha\,c - a\,\gamma, a\,\beta - \alpha\,b]$

La fuznione checkVectors(x,y) verifica se due vettori sono uguali: si scorre, tramite un ciclo for, tutto il vettore x e si confrontano tutti gli elementi di x con il corrispettivo elemento di y. Nel caso in cui vemga trovato un elemento $x_i \neq y_i$ con $i \in \{1, 2, 3\}$ la funzione termina e restituisce in output un messaggio di errore.
Altrimenti, il ciclo for viene terminato ed i vettori risultano uguali.

```
(%i21) checkVectors(x,y):=block([res],
                              size:length(x),
                              tempVect: transpose(y),
                              for i:1 thru size do(
                                 if(x[i]!=y[i]) then ( res:"Vectors are not
       equals",
                                                        return)
                               ),
                              res:"Vectors are equals"
                              )
```

(%o21) checkVectors(x, y) := **block** ([res], size: length(x), tempVect: transpose(y),
for i **thru** size **do if** $x_i! = y_i$ **then** (res: Vectors are not equals , return), res: Vectors are equals)

(%i22) checkVectors(vectMatrix,vectMaxima);

(%o22) Vectors are equals

Formula di Rodriguez

La formula di Rodriguez consente di esprimere la rotazione di un sistema riferimento attorno ad un asse v attraverso rotazioni intorno ad x, y, z senza utilizzare il metodo di Laplace o il calcolo degli autovalori e autovettori.

Quindi, si vuole passare da un sistema di riferimeno R ad un sistema di riferimento $\hat{R}$ attraverso una matrice $R_v(\theta)$:

$$R \rightarrow \hat{R}$$
$$(1) \quad R_v(\theta) = R\, R_x(\theta)\, R^T$$

N.B.:La notazione c,s equivale rispettivamente a cos(ϑ), sin(ϑ).

In particolare, come prima scelta poniamo l'asse x del sistema di riferimento R concidente all'asse di rotazione v ed i restanti assi (y,z) formerando un sistema di riferimento destro. A tale scopo, ruotiamo il sistema di riferimento x di ϑ:

$$R_x(\theta) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & c & -s \\ 0 & s & c \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} (c-1) + \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix} s$$

Inoltre, notiamo che:

i. La matrice $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ è la matrice identita $I \in \mathbb{R}^{3x3}$

ii. La matrice $\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}$ è una matrice antrisimmetrica $\boldsymbol{S}(\frac{\pi}{2})$:

$$\mathbf{S} = \begin{pmatrix} 0 & 0 \\ 0 & S_2(\frac{\pi}{2}) \end{pmatrix} \text{con } S_2\left(\frac{\pi}{2}\right) \text{matrice antisimmetrica} \in \mathbb{R}^{2x2}$$

iii. La matrice $\begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ è una matrice antisimmetrica $\mathbf{S}^2(\frac{\pi}{2})$:

$$\mathbf{S} = \begin{pmatrix} 0 & 0 \\ 0 & S_2^2(\frac{\pi}{2}) \end{pmatrix} \text{con } S_2^2\left(\frac{\pi}{2}\right) \text{matrice antisimmetrica} \in \mathbb{R}^{2x2}$$

Quindi possiamo scrivere:

$$R_x(\theta) = I + S^2(1-c) + S\, s$$

Sostituendo nella (1):

$$R_v(\theta) = R\, R_x(\theta)\, R^T = I + S^2(v)\,(1-c) + S(v)\, s$$

si ottiene la formula di Rodriguez:

$$R_v(\theta) = I + S^2(v)\,(1-c) + S(v)\, s.$$

```
Maxima 5.44.0 http://maxima.sourceforge.net
using Lisp SBCL 2.0.0
Distributed under the GNU Public License. See the file COPYING.
Dedicated to the memory of William Schelter.
The function bug_report() provides bug reporting information.
```

Funzione per la costruzione di una matrice antisimmetrica $S_v(\theta)$

```
(%i1) skewMatrix(x):=block([res],
                           S:ident(3),
                           for i:1 thru 3 do
                           (
                           for j:1 thru 3 do
                              (
                                if i=j
                                   then S[i][j]:0
                                elseif j>i
                                   then (
                                  temp:(-1)^(j-i)*x[3-remainder(i+j,3)],
                                         S[i][j]:temp,
                                         S[j][i]:-temp

                                          )
                                )
                            ),
                            res:S
                            )
```

(%o1) $\mathrm{skewMatrix}(x) := \mathbf{block}\ ([\mathrm{res}], S{:}\ \mathrm{ident}(3), \mathbf{for}\ i\ \mathbf{thru}\ 3\ \mathbf{do}\ \mathbf{for}\ j\ \mathbf{thru}\ 3\ \mathbf{do}\ \mathbf{if}\ i = j\ \mathbf{then}\ (S_i)_j{:}\ 0\ \mathbf{elseif}\ j > i\ \mathbf{then}\ (\mathrm{temp}{:}\ (-1)^{j-i}\, x_{3-\mathrm{remainder}(i+j,3)}, (S_i)_j{:}\ \mathrm{temp}, (S_j)_i{:}\ -\mathrm{temp}), \mathrm{res}{:}\ S)$

Funzione per il calcolo della formual di Rodriguez: $R_v(\theta) = I + S^2(v)\,(1-c) + S(v)\,s$

```
(%i2) rodriguez(y):=block([res],
                          I:ident(3),
                          S:skewMatrix(y),
                          res:I+S.S*(1-cos(theta))+S*sin(theta)
                          )
```

(%o2) $\mathrm{rodriguez}(y) := \mathbf{block}\ ([\mathrm{res}], I{:}\ \mathrm{ident}(3), S{:}\ \mathrm{skewMatrix}(y), \mathrm{res}{:}\ I + S \cdot S\ (1 - \cos(\vartheta)) + S \sin(\vartheta))$

Verifica della matrice di rotazione rispetto all'asse x:

```
(%i3) w:[1,0,0];
```

(%o3) $[1, 0, 0]$

```
(%i4) R[x](theta):=rodriguez(w);
```

(%o4) $R_x(\vartheta) := \mathrm{rodriguez}(w)$

```
(%i5) R[x](theta);
```

(%o5) $$\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\vartheta) & -\sin(\vartheta) \\ 0 & \sin(\vartheta) & \cos(\vartheta) \end{pmatrix}$$

Verifica della matrice di rotazione rispetto all'asse x:

```
(%i6) t:[0,1,0]
```

(%o6) $[0,1,0]$

(%i7) `R[y](theta):=rodriguez(t)`

(%o7) $R_y(\vartheta):=\mathrm{rodriguez}(t)$

(%i8) `R[y](theta);`

(%o8) $$\begin{pmatrix} \cos(\vartheta) & 0 & \sin(\vartheta) \\ 0 & 1 & 0 \\ -\sin(\vartheta) & 0 & \cos(\vartheta) \end{pmatrix}$$

Verifica della matrice di rotazione rispetto all'asse z:

(%i9) `p:[0,0,1];`

(%o9) $[0,0,1]$

(%i10) `R[z](theta):=rodriguez(p);`

(%o10) $R_z(\vartheta):=\mathrm{rodriguez}(p)$

(%i36) `R[z](theta);`

(%o36) $$\begin{pmatrix} \cos(\vartheta) & -\sin(\vartheta) & 0 \\ \sin(\vartheta) & \cos(\vartheta) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

(%i37)

Calcolare asse ed angolo tramite la formula di Rodriguez

Il problema è quello di determinare l'asse e l'angolo di rotazione data una matrice di rotazione R attraverso la formula di Rodriguez.

N.B.:La notazione c,s equivale rispettivamente a cos(ϑ), sin(ϑ).

Questo problema è detto problema orientamento inverso che consiste nel determinare, data una matrice di rotazione, asse e angolo:

$$v, \theta \longleftarrow R$$

Inizialmente occorre verificare se la matrice data è una matrice di rotazione. Le condizioni che devono essere soddisfatte affinché R sia di rotazione sono:

$$R\,R^T = I$$
$$\det(R) = 1$$

O in maniera equivalente:

$$R^T R = I$$
$$\det(R) = 1$$

(1)Se la matrice R è di rotazione, l'asse di rotazione viene determinato dall'autovettore relativo all'autovalore 1. In particolare, occorre imporre:

$$(\lambda I - R)\,v = R\begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \text{con}\,\lambda = 1$$

e trovare i coefficienti a, b, c di v affinché il prodotto $(I - R)\,v = \mathbf{0}$ e $v \neq \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$.

Il vettore v normalizzato risultante è l'asse di rotazione.

(2)In aggiunta, per calcolare l'angolo di rotazione, occorre risolvere il seguente sistema:

$$\begin{cases} R = I + S^2(v)\,(1 - c) + S(v)\,s \\ R^T = I + S^2(v)\,(1 - c) - S(v)\,s \end{cases}$$

$$(a)\frac{R - R^T}{2} = S(v)\,s$$

$$(b)\frac{R + R^T}{2} - I = S^2(v)\,(1 - c)$$

L'angolo ϑ viene determinata dalla seguente formula:

$$(c)\theta = \text{atan2}(s, c)$$

In cui s viene calcolato prendendo un qualsiasi elemento $a_{i,j} \neq 0$ con $a_{i,j} \in S(v)$ con il corrispettivo elemento $b_{i,j}$ nella matrice $(a)\frac{R-R^T}{2}$ e risolvendo l'equazione:

$$b_{i,j} = a_{i,j}\sin(\theta)$$

Invece c, viene calcolato prendendo un qualsiasi elemento $c_{i,j} \neq 0$ con $c_{i,j} \in S^2(v)$ con il corrispettivo elemento $d_{i,j}$ nella matrice $(b)\frac{R+R^T}{2} - I$ e risolvendo l'equazione:

$$c_{i,j}(1-\cos(\theta)) = d_{i,j}$$

La funzione isRotation(M) prende in input una matrice M e verifica se la matrice data è di rotazione. In particolare, verifica se $MM^T = I$(i) e $\det(M) = 1$(ii).
Inoltre, se la matrice è simbolica cioè dipendente dalla variabile theta, restituisce le condizioni per cui la matrice M è di rotazione; altrimenti, la matrice non è di rotazione.

```
(%i1) isRotation(M):=block([res],
                          I:ident(3),
                          MMT:trigsimp(expand(M.transpose(M))),
                          detM:trigsimp(expand(determinant(M))),

                          if MMT=I  and detM=1
                             then(

                                  return(res:1)
                                  )

                          else(

                                res: "R is not rotation matrix"
                                )
                          )
```

(%o1) $\text{isRotation}(M) := \textbf{block}\ ([\text{res}], I\colon \text{ident}(3), \text{MMT}\colon \text{trigsimp}(\text{expand}(M \cdot \text{transpose}(M))),$ $\text{detM}\colon \text{trigsimp}(\text{expand}(\text{determinant}(M))), \textbf{if}\ \text{MMT} = I \wedge \text{detM} = 1\ \textbf{then}\ \text{return}(\text{res}\colon 1)\ \textbf{else}\ \text{res}\colon$ R is not rotation matrix)

La funzione axes(M) prende in input una matrice $M = \text{adj}(I - R)$ di rotazione e ne calcola il corrispondente asse di rotazione e,utilizzando la procedura del punto (1), selezione come asse di rotazione una colonna non nulla della matrice M.

```
(%i2) axes(M):=block([res],
                     columns:transpose(M),
                    res:zeromatrix(3,1),
                     for i:1 thru length(columns) do(
                       if(columns[i][1]# 0 or columns[i][2]#0 or
      columns[i][3]#0)
                          then ( return(m: transpose(columns[i])))

                     ),res:m

                     )
```

(%o2) $\text{axes}(M) := \textbf{block}\ ([\text{res}], \text{columns}\colon \text{transpose}(M), \text{res}\colon \text{zeromatrix}(3,1),$

for i **thru** length(columns) **do if** $(\text{columns}_i)_1 \neq 0 \vee (\text{columns}_i)_2 \neq 0 \vee (\text{columns}_i)_3 \neq 0$ **then** return(m: transpose(columns$_i$)), res: m)

La funzione skewMatrix(x) prende in input un asse x di rotazione normalizzato e ne calcola la corrispettiva matrice antisimmetrica $S_v(\theta)$.

```
(%i3) skewMatrix(x):=block([res],
                        S:ident(3),
                        for i:1 thru 3 do
                        (
                        for j:1 thru 3 do
                           (
                             if i=j
                                then S[i][j]:0
                             elseif j>i
                                then (
                               temp:(-1)^(j-i)*x[3-remainder(i+j,3)][1],
                                      S[i][j]:temp,
                                      S[j][i]:-temp

                                       )
                             )
                         ),
                         res:S
                         )
```

(%o3) skewMatrix(x) := **block** ([res], S: ident(3), **for** i **thru** 3 **do for** j **thru** 3 **do if** $i = j$ **then** $(S_i)_j$: 0 **elseif** $j > i$ **then** (temp: $(-1)^{j-i}\,(x_{3-\text{remainder}(i+j,3)})_1$, $(S_i)_j$: temp, $(S_j)_i$: $-$temp), res: S)

La funzione sinRotation(skewMat, RRT2) prende in input una matrice antisimmetrica $S_v(\theta)$ e la matrice $\frac{R-R^T}{2}$ ricavata dalla risoluzione del sistema della formula di Rodriguez e risolvere l'equazione (a).

```
(%i4) sinRotation(skewMat,RRT2):=block([res],

                           for i:1 thru 3 do(
                              for j:1 thru 3 do(
                                 a:skewMat[i][j],

                                 if a# 0
                                    then (b:RRT2[i][j],

                                          return(

                                     value:b/a
                                     ))
                               )
                             ),res:value


                           )
```

(%o4) sinRotation(skewMat, RRT2) := **block** $\Bigg($[res], **for** i **thru** 3 **do for** j **thru** 3 **do** $\Bigg($$a$:

$(\mathrm{skewMat}_i)_j$, **if** $a \neq 0$ **then** $\left(b\colon (\mathrm{RRT2}_i)_j, \mathrm{return}\left(\mathrm{value}\colon \frac{b}{a}\right)\right)\Bigg)$, res: value$\Bigg)$

La funzione corRotation(x,y) prende in input una matrice antisimmetrica $S_v(\theta)$ e la matrice $\frac{R+R^T}{2} - I$ ricavata dalla soluzione del sistema della formula di Rodriguez e risolve l'equazione (b).

```
(%i5) cosRotation(x,y):=block([res],

                                   for i:1 thru 3 do(
                                      for j:1 thru 3 do(
                                         c:x[i][j],
                                         if(c#0) then(
                                                    d:y[i][j],

                                                    return(t:(c-d)/c))
                                      )
                                   ),
                                    res:t

                                   )
```

(%o5) $\mathrm{cosRotation}(x, y) := \mathbf{block}\left([\mathrm{res}], \mathbf{for}\ i\ \mathbf{thru}\ 3\ \mathbf{do}\ \mathbf{for}\ j\ \mathbf{thru}\ 3\ \mathbf{do}\ \left(c\colon (x_i)_j, \mathbf{if}\ c \neq 0\ \mathbf{then}\ \left(d\colon (y_i)_j, \mathrm{return}\left(t\colon \frac{c-d}{c}\right)\right)\right), \mathrm{res}\colon t\right)$

La funzione degree(v,M) prende in input un asse di rotazione v e una matrice M. Invoca le funzioni sinRotation e cosRotation per salvare nelle variabili sinR e cosR i valori rispettivamente di $\sin(\theta)$ e $\cos(\theta)$. Infine applica (c) per restituire il valore di ϑ.

```
(%i6) degree(v,M):=block([sinR,cosR,res],
                             S:skewMatrix(v),
                             I:ident(3),
                             RRsin:trigsimp((M-transpose(M))*1/2),
                             RRcos:trigsimp(((M+transpose(M))*1/2)-I),
                             sinR:sinRotation(S,RRsin),

                             SS:S.S,
                             cosR:cosRotation(SS,RRcos),
                             res:atan2(expand(sinR),expand(cosR))

                               )
```

(%o6) $\mathrm{degree}(v, M) := \mathbf{block}\left([\mathrm{sinR}, \mathrm{cosR}, \mathrm{res}], S\colon \mathrm{skewMatrix}(v), I\colon \mathrm{ident}(3), \mathrm{RRsin}\colon \mathrm{trigsimp}\left(\frac{(M - \mathrm{transpose}(M))\,1}{2}\right), \mathrm{RRcos}\colon \mathrm{trigsimp}\left(\frac{(M + \mathrm{transpose}(M))\,1}{2} - I\right), \mathrm{sinR}\colon \mathrm{sinRotation}(S, \mathrm{RRsin}), \mathrm{SS}\colon S \cdot S, \mathrm{cosR}\colon \mathrm{cosRotation}(\mathrm{SS}, \mathrm{RRcos}), \mathrm{res}\colon \mathrm{atan2}(\mathrm{expand}(\mathrm{sinR}), \mathrm{expand}(\mathrm{cosR}))\right)$

La funzione axesDegree(R) prende in input una matrice R, simbolica e non, per restituire il corrispondente asse e angolo di rotazione, verificando se l'effettiva matrice in input sia di rotazione. In caso contrario, restituisce errore.

```
(%i7)  axesDegree(R):=block([v,theta,res],
                          isRot:isRotation(R),
                          if isRot=1 then (
                             I:ident(3),
                             adjR:adjoint(I-R),
                             v:axes(adjR),
                             vNorm:v/sqrt(v.v),
                             theta:degree(vNorm,R),
                             print("Axe, degree"),
                             res:[vNorm,theta]



                          )
                          else res:"R is not rotation matrix"



                         )
```

(%o7) $\mathrm{axesDegree}(R) := \mathbf{block}\left([v, \vartheta, \mathrm{res}], \mathrm{isRot}\colon \mathrm{isRotation}(R), \mathbf{if}\ \mathrm{isRot} = 1\ \mathbf{then}\left(I\colon \mathrm{ident}(3), \mathrm{adjR}\colon \mathrm{adjoint}(I - R), v\colon \mathrm{axes}(\mathrm{adjR}), \mathrm{vNorm}\colon \frac{v}{\sqrt{v\cdot v}}, \vartheta\colon \mathrm{degree}(\mathrm{vNorm}, R), \mathrm{print}(\mathrm{Axe},\ \mathrm{degree}\ ), \mathrm{res}\colon [\mathrm{vNorm}, \vartheta]\right)\ \mathbf{else}\ \mathrm{res}\colon \text{R is not rotation matrix}\right)$

Asse e angolo corrispettivi ad una rotazione lungo l'asse x:

```
(%i8) R:matrix([1,0,0],[0,-1,0],[0,0,-1])
```

(%o8) $$\begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{pmatrix}$$

```
(%i9) axesDegree(R)
```

Axe, degree

(%o9) $$\left[\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \pi\right]$$

Asse e angolo corrispettivi ad una rotazione attorno asse y:

```
(%i10) R:matrix([0,0,1],[0,1,0],[-1,0,0]);
```

(%o10) $$\begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{pmatrix}$$

```
(%i11) axesDegree(R);
```

Axe, degree

(%o11) $$\left[\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \frac{\pi}{2}\right]$$

Asse e angolo corrispettivi ad una rotazione attorno l'asse z:

```
(%i12) R:matrix([0,-1,0],[1,0,0],[0,0,1]);
```

$$\text{(\%o12)}\quad \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

```
(%i13) axesDegree(R);
```

Axe, degree

$$\text{(\%o13)}\quad \left[\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \frac{\pi}{2}\right]$$

Matrice simbolica lungo x:

```
(%i14) R[x](theta) := matrix([1,0,0],
                             [0,cos(theta),-sin(theta)],
                             [0,sin(theta), cos(theta)]);
```

$$\text{(\%o14)}\quad R_x(\vartheta) := \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\vartheta) & -\sin(\vartheta) \\ 0 & \sin(\vartheta) & \cos(\vartheta) \end{pmatrix}$$

```
(%i15) axesDegree(R[x](theta));
```

Axe, degree

$$\text{(\%o15)}\quad \left[\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \mathrm{atan2}(\sin(\vartheta), \cos(\vartheta))\right]$$

Matrice simbolica lungo y:

```
(%i16) R[y](theta) := matrix([cos(theta),0,sin(theta)],
                             [0,1,0],
                             [-sin(theta),0, cos(theta)]);
```

$$\text{(\%o16)}\quad R_y(\vartheta) := \begin{pmatrix} \cos(\vartheta) & 0 & \sin(\vartheta) \\ 0 & 1 & 0 \\ -\sin(\vartheta) & 0 & \cos(\vartheta) \end{pmatrix}$$

```
(%i17) axesDegree(R[y](theta));
```

Axe, degree

$$\text{(\%o17)}\quad \left[\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \mathrm{atan2}(\sin(\vartheta), \cos(\vartheta))\right]$$

Matrice simbolica lungo z:

```
(%i18) R[z](theta) := matrix([cos(theta),-sin(theta),0],
                             [sin(theta),cos(theta),0],
                             [0,0, 1]);
```

$$\text{(\%o18)}\quad R_z(\vartheta) := \begin{pmatrix} \cos(\vartheta) & -\sin(\vartheta) & 0 \\ \sin(\vartheta) & \cos(\vartheta) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

```
(%i19) axesDegree(R[z](theta));
```

Axe, degree

$$\text{(\%o19)}\quad \left[\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \mathrm{atan2}(\sin(\vartheta), \cos(\vartheta))\right]$$

Matrice non di rotazione:

```
(%i20) R:matrix([1,1,0],[0,1,1],[1,1,1]);
```

(%o20) $\begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$

```
(%i21) axesDegree(R);
```

(%o21) R is not rotation matrix

Matrice razionale di rotazione:

```
(%i22) R:matrix([1,0,0],[0,sqrt(3)/2,-1/2],[0,1/2,sqrt(3)/2]);
```

(%o22) $\begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{\sqrt{3}}{2} & -\frac{1}{2} \\ 0 & \frac{1}{2} & \frac{\sqrt{3}}{2} \end{pmatrix}$

```
(%i23) axesDegree(R);
```

Axe, degree

(%o23) $\left[\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \frac{\pi}{6}\right]$

```
(%i24) R:matrix([0,-1,0],[1,0,0],[0,0,1]);
```

(%o24) $\begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

```
(%i25) axesDegree(R)
```

Axe, degree

(%o25) $\left[\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \frac{\pi}{2}\right]$

Matrice di rotazione risultante dalla rotazione Ra sull'asse y e Rb sull'asse z;

$$R_a = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{pmatrix}; R_b = \begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$R_a\, R_b = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \end{pmatrix}$$

```
(%i26) R[ab]:matrix([0,1,0],[0,0,-1],[-1,0,0]);
```

(%o26) $\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & -1 \\ -1 & 0 & 0 \end{pmatrix}$

```
(%i27) axesDegree(R[ab])
```

Axe, degree

(%o27) $$\left[\begin{pmatrix}\frac{1}{\sqrt{3}}\\ \frac{1}{\sqrt{3}}\\ -\frac{1}{\sqrt{3}}\end{pmatrix}, \frac{2\,\pi}{3}\right]$$

(%i28)

Parametrizzazione di Cayley

La parametrizzazione di Cayley permette di esprimere, attraverso la matrice S antisimmetrica, la corrispettiva matrice di rotazione R e viceversa. La parametrizzazione di Cayley avviene tramite le seguenti relazioni:

$$(1) S \longrightarrow R = (I+S)\,(I-S)^{-1} \Longrightarrow R \quad \text{è di rotazione}$$

$$(2) R \longrightarrow S = (R+I)^{-1}\,(R-I) \Longrightarrow S \quad \text{è una matrice antisimmetrica}$$

In particolare, dato un asse di rotazione v occorre prima calcolare la matrice antisimmetrica S ed in seguito applicare la (1) per ottenere la corrispondente matrice di rotazione.

Alternativamente, data una matrice di rotazione R occorre applicare (2) per ottenere la matrice antisimmetrica corrispondente e, in seguito, selezionare all'interno della matrice S gli elementi dell'asse di rotazione v. Quest'ultimi corrispondono a:

$$S = \begin{pmatrix} 0 & -c & b \\ c & 0 & -a \\ -b & a & 0 \end{pmatrix} \longrightarrow v = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$

Ottenuto il vettore v è possibile calcolare il versore e l'angolo di rotazione. Infatti:

$$v = \frac{v}{\|v\|}\|v\| \quad \text{in cui} \quad \frac{v}{\|v\|} := \text{versore di rotazione} \quad \|v\| := \text{angolo di rotazione}$$

N.B.: $\|v\|$ è la norma-2 del vettore v

Per calcolare correttamente l'angolo di rotazione data la norma del vettore v, poniamo v=α. Quindi, sussistono le seguenti relazioni:

$$\alpha = \frac{\sin(\theta)}{1+\cos(\theta)} \longrightarrow \alpha^2 = \frac{1-\cos(\theta)}{1+\cos(\theta)}$$
$$(i) \qquad \cos(\theta) = \frac{1-\alpha^2}{1+\alpha^2} \equiv \frac{1-\|v\|^2}{1+\|v\|^2}$$

Data la pluralità delle soluzioni occorre calcolare anche il $\sin(\theta)$:

$$\sin(\theta)^2 = 1 - \cos(\theta)^2$$

Sostituendo (i):

$$\sin(\theta) = \pm\frac{2\,\alpha}{1+\alpha^2}$$

In cui il segno di sin(ϑ) deve soddisfare la condizione in cui $\|v\|$>0 cioé: $\frac{\sin(\theta)}{1+\cos(\theta)} > 0$.

Dopodiché l'angolo è univocamente identificato tramite la funzione atan2:

$$(\text{ii})\theta = \text{atan2}\left(\pm\frac{2\,\alpha}{1+\alpha^2}, \frac{1-\alpha^2}{1+\alpha^2}\right) = \text{atan2}(\sin(\theta), \cos(\theta))$$

```
Maxima 5.44.0 http://maxima.sourceforge.net
using Lisp SBCL 2.0.0
Distributed under the GNU Public License. See the file COPYING.
Dedicated to the memory of William Schelter.
```

```
The function bug_report() provides bug reporting information.
```

La funzione isRotation verifica se la matrice in input M è di rotazione o meno. Restituisce 1 in caso positvo, altrimenti errore.

```
(%i1) isRotation(M):=block([res],
                           I:ident(3),
                           MMT:trigsimp(expand(M.transpose(M))),
                           detM:trigsimp(expand(determinant(M))),

                           if MMT=I  and detM=1
                              then(

                                   return(res:1)
                                   )

                           else(

                                 res: "R is not rotation matrix"
                                 )
                           )
```

(%o1) isRotation(M) := **block** ([res], I: ident(3), MMT: trigsimp(expand($M \cdot$ transpose(M))), detM: trigsimp(expand(determinant(M))), **if** MMT $= I \wedge$ detM $= 1$ **then** return(res: 1) **else** res: R is not rotation matrix)

La funzione skewMatric(x) prende in input una vettore x e ne costruisce l'antisimetrica

```
(%i2) skewMatrix(x):=block([res],
                           M:ident(3),
                           for i:1 thru 3 do
                           (
                           for j:1 thru 3 do
                              (
                                if i=j
                                   then M[i][j]:0
                                elseif j>i
                                   then (
                                  temp:(-1)^(j-i)*x[3-remainder(i+j,3)],
                                         M[i][j]:temp,
                                         M[j][i]:-temp

                                          )
                                )
                            ),
                            res:M
                            )
```

(%o2) skewMatrix(x) := **block** ([res], M: ident(3), **for** i **thru** 3 **do for** j **thru** 3 **do if** $i = j$ **then** $(M_i)_j$: 0 **elseif** $j > i$ **then** (temp: $(-1)^{j-i}\, x_{3-\mathrm{remainder}(i+j,3)}$, $(M_i)_j$: temp, $(M_j)_i$: $-$temp), res: M)

La funzione degreeVector(x) prende in input un vettore x e ne calcola l'angolo come descritto nella procedura (ii) scegliendo opportunamente il segno del $\sin(\theta)$

```
(%i3) degreeVector(x):=block([res],
                              vNorm:x.x,
                                cosTheta:(1-vNorm)/(1+vNorm),
                                sinTheta:(2*sqrt(vNorm))/(1+vNorm),

                                if sinTheta/(1+cosTheta)>0
                                   then (print(atan2(sinTheta,cosTheta)),
      degree:atan2(sinTheta,cosTheta))
                                else (degree:atan2(-sinTheta,cosTheta)),
                                res:degree
                                )
```

(%o3) $\mathrm{degreeVector}(x) := \textbf{block}\left([\mathrm{res}], \mathrm{vNorm}\colon x\cdot x, \mathrm{cosTheta}\colon \frac{1-\mathrm{vNorm}}{1+\mathrm{vNorm}}, \mathrm{sinTheta}\colon \frac{2\sqrt{\mathrm{vNorm}}}{1+\mathrm{vNorm}}, \textbf{if } \frac{\mathrm{sinTheta}}{1+\mathrm{cosTheta}} > 0 \textbf{ then } (\mathrm{print}(\mathrm{atan2}(\mathrm{sinTheta}, \mathrm{cosTheta})), \mathrm{degree}\colon \mathrm{atan2}(\mathrm{sinTheta}, \mathrm{cosTheta})) \textbf{ else } \mathrm{degree}\colon \mathrm{atan2}(-\mathrm{sinTheta}, \mathrm{cosTheta}), \mathrm{res}\colon \mathrm{degree}\right)$

La funzione cayleyRotation(a) prende in input un vettore a, ne calcola l'antisimmetrica e resistuisce in output la corrispondente matrice di rotazione R attraverso la parametrizzazione di Cayley(1). In output vi sono per completezza anche la corrispettiva matrice antisimmetrica, asse di rotazione e angolo di rotazione.

```
(%i4) cayleyRotation(a):=block([res],
                                S:skewMatrix(a),
                                I:ident(3),
                                R:(I+S).invert(I-S),
                                degree:degreeVector(a),
                                print("Matrice di rotazione, Matrice
      antisimmetrica, asse, angolo"),
                                res:[facsum(expand(R)),S,a,degree])
```

(%o4) $\mathrm{cayleyRotation}(a) := \textbf{block}\ ([\mathrm{res}], S\colon \mathrm{skewMatrix}(a), I\colon \mathrm{ident}(3), R\colon (I+S)\cdot \mathrm{invert}(I-S), \mathrm{degree}\colon \mathrm{degreeVector}(a), \mathrm{print}(\text{Matrice di rotazione, Matrice antisimmetrica, asse, angolo }), \mathrm{res}\colon [\mathrm{facsum}(\mathrm{expand}(R)), S, a, \mathrm{degree}])$

La funzione cayleySkewMatrix(R) prende in input una matrice R e, tramite la parametrizzazione di Cayley, restituisce in output la corrispondente matrice antisimmetrica S.
In particolare, per completezza viene restituito un array contenente la matrice di rotazione, la matrice antisimmetrica, asse e angolo di rotazione.

```
(%i5) cayleySkewMatrix(R):=block([res],
                                  isRot:isRotation(R),
                                  if isRot=1 then(
                                     I:ident(3),
                                     S:invert(R+I).(R-I),
                                     v:[S[3][2],S[1][3],S[2][1]],
                                     degree:degreeVector(v),
                                     print("Matrice di rotazione, Matrice
      antisimmetrica, asse, angolo"),
                                     res:[R,S,v,degree]

                                  )
                                  else res:"Matrix is not rotation"
                                  )
```

(%o5) cayleySkewMatrix(R) := **block** ([res], isRot: isRotation(R), **if** isRot $=1$ **then** (I: ident(3), S: invert$(R+I)\cdot(R-I)$, v: $[(S_3)_2,\ (S_1)_3,\ (S_2)_1]$, degree: degreeVector$(v)$, print(Matrice di rotazione, Matrice antisimmetrica, asse, angolo), res: $[R, S, v, \text{degree}]$) **else** res: Matrix is not rotation)

Asse di rotazione versore lungo l'asse x

(%i6) `v:[1,0,0]`

(%o6) $[1,0,0]$

(%i7) `a:cayleyRotation(v)`

$\frac{\pi}{2}$

Matrice di rotazione, Matrice antisimmetrica, asse, angolo

(%o7) $$\left[\begin{pmatrix}1&0&0\\0&0&-1\\0&1&0\end{pmatrix},\begin{pmatrix}0&0&0\\0&0&-1\\0&1&0\end{pmatrix},[1,0,0],\frac{\pi}{2}\right]$$

La prima e la seconda parametrizzazione di Cayley restituiscono gli stessi risultati.

(%i8) `RR:a[1]`

(%o8) $$\begin{pmatrix}1&0&0\\0&0&-1\\0&1&0\end{pmatrix}$$

(%i9) `cayleySkewMatrix(a[1])`

$\frac{\pi}{2}$

Matrice di rotazione, Matrice antisimmetrica, asse, angolo

(%o9) $$\left[\begin{pmatrix}1&0&0\\0&0&-1\\0&1&0\end{pmatrix},\begin{pmatrix}0&0&0\\0&0&-1\\0&1&0\end{pmatrix},[1,0,0],\frac{\pi}{2}\right]$$

Parametrizzazione di Cayley simbolica:

(%i10) `v:[alpha,beta,%gamma];`

(%o10) $[\alpha,\beta,\gamma]$

(%i11) `a:cayleyRotation(v);`

$$\operatorname{atan2}\left(\frac{2\sqrt{\beta^2+\alpha^2+\gamma^2}}{\beta^2+\alpha^2+\gamma^2+1},\frac{-\beta^2-\alpha^2-\gamma^2+1}{\beta^2+\alpha^2+\gamma^2+1}\right)$$

Matrice di rotazione, Matrice antisimmetrica, asse, angolo

(%o11) $$\left[\begin{pmatrix}-\frac{\beta^2-\alpha^2+\gamma^2-1}{\beta^2+\alpha^2+\gamma^2+1}&\frac{2(\alpha\beta-\gamma)}{\beta^2+\alpha^2+\gamma^2+1}&\frac{2(\beta+\gamma\alpha)}{\beta^2+\alpha^2+\gamma^2+1}\\\frac{2(\alpha\beta+\gamma)}{\beta^2+\alpha^2+\gamma^2+1}&\frac{\beta^2-\alpha^2-\gamma^2+1}{\beta^2+\alpha^2+\gamma^2+1}&\frac{2(\gamma\beta-\alpha)}{\beta^2+\alpha^2+\gamma^2+1}\\-\frac{2(\beta-\gamma\alpha)}{\beta^2+\alpha^2+\gamma^2+1}&\frac{2(\gamma\beta+\alpha)}{\beta^2+\alpha^2+\gamma^2+1}&-\frac{\beta^2+\alpha^2-\gamma^2-1}{\beta^2+\alpha^2+\gamma^2+1}\end{pmatrix},\begin{pmatrix}0&-\gamma&\beta\\\gamma&0&-\alpha\\-\beta&\alpha&0\end{pmatrix},[\alpha,\beta,\gamma],\right.$$
$$\left.\operatorname{atan2}\left(\frac{2\sqrt{\beta^2+\alpha^2+\gamma^2}}{\beta^2+\alpha^2+\gamma^2+1},\frac{-\beta^2-\alpha^2-\gamma^2+1}{\beta^2+\alpha^2+\gamma^2+1}\right)\right]$$

(%i12)

Procedura 8 e 9 in cascata

```
Maxima 5.44.0 http://maxima.sourceforge.net
using Lisp SBCL 2.0.0
Distributed under the GNU Public License. See the file COPYING.
Dedicated to the memory of William Schelter.
The function bug_report() provides bug reporting information.
```

```
(%i1) skewMatrix(x):=block([res],
                           S:ident(3),
                           for i:1 thru 3 do
                           (
                           for j:1 thru 3 do
                              (
                                if i=j
                                   then S[i][j]:0
                                elseif j>i
                                   then (
                                  temp:(-1)^(j-i)*x[3-remainder(i+j,3)],
                                         S[i][j]:temp,
                                         S[j][i]:-temp

                                          )
                                )
                            ),
                            res:S
                            )
```

(%o1) skewMatrix$(x) :=$ **block** $([\mathrm{res}], S\colon \mathrm{ident}(3),$ **for** i **thru** 3 **do for** j **thru** 3 **do if** $i=j$ **then** $(S_i)_j\colon 0$ **elseif** $j>i$ **then** $(\mathrm{temp}\colon (-1)^{j-i}\, x_{3-\mathrm{remainder}(i+j,3)}, (S_i)_j\colon \mathrm{temp}, (S_j)_i\colon -\mathrm{temp}), \mathrm{res}\colon S)$

```
(%i2) rodriguez(y):=block([res],
                          I:ident(3),
                          S:skewMatrix(y),
                          res:I+S.S*(1-cos(theta))+S*sin(theta)
                          )
```

(%o2) $\mathrm{rodriguez}(y) :=$ **block** $([\mathrm{res}], I\colon \mathrm{ident}(3), S\colon \mathrm{skewMatrix}(y), \mathrm{res}\colon I + S\cdot S\,(1-\cos(\vartheta)) + S\sin(\vartheta))$

```
(%i3) isRotation(M):=block([res],
                           I:ident(3),
                           MMT:trigsimp(expand(M.transpose(M))),
                           detM:trigsimp(expand(determinant(M))),

                           if MMT=I  and detM=1
                              then(

                                   return(res:1)
                                   )

                           else(

                                 res: "R is not rotation matrix"
                                 )
                           )
```

(%o3) $\mathrm{isRotation}(M) :=$ **block** $([\mathrm{res}], I\colon \mathrm{ident}(3), \mathrm{MMT}\colon \mathrm{trigsimp}(\mathrm{expand}(M\cdot \mathrm{transpose}(M))),$

detM: trigsimp(expand(determinant(M))), **if** MMT $= I \wedge$ detM $= 1$ **then** return(res: 1) **else** res: R is not rotation matrix)

```
(%i4) axes(M):=block([res],
                      columns:transpose(M),
                     res:zeromatrix(3,1),
                      for i:1 thru length(columns) do(
                        if(columns[i][1]# 0 or columns[i][2]#0 or
      columns[i][3]#0)
                           then ( return(m: transpose(columns[i])))

                      ),res:m

                      )
```

(%o4) axes(M) := **block** ([res], columns: transpose(M), res: zeromatrix(3, 1), **for** i **thru** length(columns) **do if** $(\text{columns}_i)_1 \neq 0 \vee (\text{columns}_i)_2 \neq 0 \vee (\text{columns}_i)_3 \neq 0$ **then** return(m: transpose(columns_i)), res: m)

```
(%i5) skewMatrix(x):=block([res],
                          S:ident(3),
                          for i:1 thru 3 do
                          (
                          for j:1 thru 3 do
                             (
                               if i=j
                                  then S[i][j]:0
                               elseif j>i
                                  then (
                                 temp:(-1)^(j-i)*x[3-remainder(i+j,3)][1],
                                        S[i][j]:temp,
                                        S[j][i]:-temp

                                         )
                               )
                           ),
                           res:S
                           )
```

(%o5) skewMatrix(x) := **block** ([res], S: ident(3), **for** i **thru** 3 **do for** j **thru** 3 **do if** $i = j$ **then** $(S_i)_j$: 0 **elseif** $j > i$ **then** (temp: $(-1)^{j-i}\,(x_{3-\text{remainder}(i+j,3)})_1$, $(S_i)_j$: temp, $(S_j)_i$: $-$temp), res: S)

(%i6) sinRotation(skewMat,RRT2):=block([res],

```
                         for i:1 thru 3 do(
                            for j:1 thru 3 do(
                               a:skewMat[i][j],

                               if a# 0
                                  then (b:RRT2[i][j],

                                        return(

                                   value:b/a
                                   ))
                              )
                          ),res:value


                        )
```

(%o6) $\mathrm{sinRotation}(\mathrm{skewMat}, \mathrm{RRT2}) := \mathbf{block}\left([\mathrm{res}], \mathbf{for}\ i\ \mathbf{thru}\ 3\ \mathbf{do}\ \mathbf{for}\ j\ \mathbf{thru}\ 3\ \mathbf{do}\left(a\colon (\mathrm{skewMat}_i)_j, \mathbf{if}\ a \neq 0\ \mathbf{then}\left(b\colon (\mathrm{RRT2}_i)_j, \mathrm{return}\left(\mathrm{value}\colon \frac{b}{a}\right)\right)\right), \mathrm{res}\colon \mathrm{value}\right)$

(%i7) cosRotation(x,y):=block([res],

```
                         for i:1 thru 3 do(
                            for j:1 thru 3 do(
                               c:x[i][j],
                               if(c#0) then(
                                          d:y[i][j],

                                          return(t:(c-d)/c))
                            )
                         ),
                          res:t

                         )
```

(%o7) $\mathrm{cosRotation}(x, y) := \mathbf{block}\left([\mathrm{res}], \mathbf{for}\ i\ \mathbf{thru}\ 3\ \mathbf{do}\ \mathbf{for}\ j\ \mathbf{thru}\ 3\ \mathbf{do}\left(c\colon (x_i)_j, \mathbf{if}\ c \neq 0\ \mathbf{then}\left(d\colon (y_i)_j, \mathrm{return}\left(t\colon \frac{c-d}{c}\right)\right)\right), \mathrm{res}\colon t\right)$

(%i8) degree(v,M):=block([sinR,cosR,res],

```
                     S:skewMatrix(v),
                     I:ident(3),
                     RRsin:trigsimp((M-transpose(M))*1/2),
                     RRcos:trigsimp(((M+transpose(M))*1/2)-I),
                     sinR:sinRotation(S,RRsin),

                     SS:S.S,
                     cosR:cosRotation(SS,RRcos),
                     res:atan2(expand(sinR),expand(cosR))

                       )
```

(%o8) $\mathrm{degree}(v,M):=\mathbf{block}\left([\mathrm{sinR},\mathrm{cosR},\mathrm{res}],S\colon\mathrm{skewMatrix}(v),I\colon\mathrm{ident}(3),\mathrm{RRsin}\colon \mathrm{trigsimp}\left(\frac{(M-\mathrm{transpose}(M))\,1}{2}\right),\mathrm{RRcos}\colon\mathrm{trigsimp}\left(\frac{(M+\mathrm{transpose}(M))\,1}{2}-I\right),\mathrm{sinR}\colon \mathrm{sinRotation}(S,\mathrm{RRsin}),\mathrm{SS}\colon S\cdot S,\mathrm{cosR}\colon\mathrm{cosRotation}(\mathrm{SS},\mathrm{RRcos}),\mathrm{res}\colon\mathrm{atan2}(\mathrm{expand}(\mathrm{sinR}), \mathrm{expand}(\mathrm{cosR}))\right)$

(%i9)
```
axesDegree(R):=block([v,theta,res],
                     isRot:isRotation(R),
                     if isRot=1 then (
                        I:ident(3),
                        adjR:adjoint(I-R),
                        v:axes(adjR),
                        vNorm:v/sqrt(v.v),
                        theta:degree(vNorm,R),
                        print("Axe, degree"),
                        res:[vNorm,theta]



                     )
                     else res:"R is not rotation matrix"



                    )
```

(%o9) $\mathrm{axesDegree}(R):=\mathbf{block}\left([v,\vartheta,\mathrm{res}],\mathrm{isRot}\colon\mathrm{isRotation}(R),\mathbf{if}\ \mathrm{isRot}=1\ \mathbf{then}\left(I\colon\mathrm{ident}(3), \mathrm{adjR}\colon\mathrm{adjoint}(I-R),v\colon\mathrm{axes}(\mathrm{adjR}),\mathrm{vNorm}\colon\frac{v}{\sqrt{v\cdot v}},\vartheta\colon\mathrm{degree}(\mathrm{vNorm},R),\mathrm{print}(\text{Axe, degree }),\mathrm{res}\colon [\mathrm{vNorm},\vartheta]\right)\ \mathbf{else}\ \mathrm{res}\colon\text{R is not rotation matrix }\right)$

Ruoto attorno all'asse x di $\frac{2\pi}{3}$. La corrispettiva matrice di rotazione viene calcolata tramite la procedura di Rodriguez. (procedura 8)

(%i10) `v:1/sqrt(3)*matrix([1,-1,1]);`

(%o10) $\begin{pmatrix}\frac{1}{\sqrt{3}} & -\frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}}\end{pmatrix}$

$R_x(\theta)$ descrive una matrice di rotazione dipendente dal parametro ϑ lungo l'asse di rotazione x.

(%i11) `R[x](theta):=rodriguez(transpose(v))`

(%o11) $R_x(\vartheta):=\mathrm{rodriguez}(\mathrm{transpose}(v))$

Assegno l'angolo di rotazione di $\frac{2\,\pi}{3}$ alla matrice di rotazione descritta sopra.

(%i12) `R:R[x](2*%pi/3)`

(%o12) $\begin{pmatrix}0 & -1 & 0\\ 0 & 0 & -1\\ 1 & 0 & 0\end{pmatrix}$

A questo punto si verifica che la matrice di rotazione ottenuta abbia lo stesso asse e angolo tramite la procedura 9.

(%i13) `axesDegree(R);`

Axe, degree

(%o13) $$\left[\begin{pmatrix} \frac{1}{\sqrt{3}} \\ -\frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{3}} \end{pmatrix}, \frac{2\,\pi}{3}\right]$$

(%i14)

La procedura axesDegree ritorna effettivamente l'asse di rotazione $v = \frac{1}{\sqrt{3}}\begin{pmatrix} 1 \\ -1 \\ 1 \end{pmatrix}$ e l'angolo di rotazione $\frac{2\pi}{3}$.

Cinematica nel piano

Rappresentare la cinematica diretta completa dei robot nel piano.

Calcolare la cinematica diretta significa esprimere le coordinate del punto terminale (end-effector) del robot in funzione delle variabili di giunto che sono tante quante sono le variabili di libertà. In particolare se un robot possiede n gradi di libertà, occorre definire n+1 sistemi di riferimento e la notazione utilizzata è la seguente:

$$n\,\text{DOF} \Longleftrightarrow n+1\,\text{sistemi di riferimento}\,R_0, R_1, \ldots, R_n$$

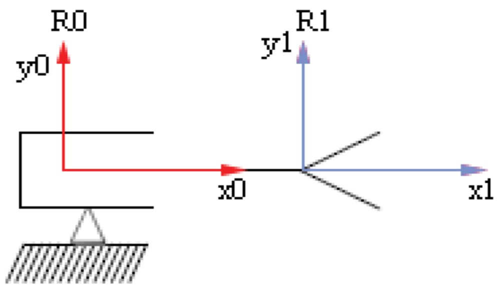
ove R_0 è il sistema di riferimento inerziale e i restanti sistemi di riferimento $R_1, \ldots, R_n$ sono sistemi di riferimento solidali al link, ma non interziali.

L'idea che occorre seguire per calcolare la cinematica diretta di un robot consiste, dopo aver fissato correttamente i sistemi di riferimento, nel sovrapporre i sistemi di riferimento tramite rotazioni e traslazioni. La matrice risultante da una rotazione e una traslazione viene detta matrice di trasformazione.

N.B.:*in caso di variabili di giunto (rotoidali(R),prismatici(P)) il simbolo utilizzato è q_i. *Inoltre, indichiamo con Q(R,0) la matrice di rotazione e R(I,d) la matrice di traslazione.

Solo nel caso planare effettuare una rotazione e poi una traslazione è equivalente ad effettuare una rotazione e poi una traslazione.

1 DOF P - 1 gradi di libertà prismatico



$$T_{01} := \text{matrice di trasformazione da}\, R_o \rightarrow R_1$$

$$T_{01}(q_1) = \begin{pmatrix} \text{rotazione} & \text{traslazione} \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} R(0) & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} I & q1 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} I & q_1\, e_x \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & q_1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

In particolare, il blocco 2x2: $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ rappresenta la cinematica di rotazione in cui gli assi sono paralleli alla base e il vettore $\begin{pmatrix} q_1 \\ 0 \end{pmatrix}$ rappresenta la cinematica di posizione con q_1, ascissa, e 0 ordinata.

La funzione rotTrasl(a,q1,k) restituisce la matrice di rotazione Q(R,0) con parametri rotTrasl(r,q1,0) e la matrice di traslazione R(I,d) con parametri rotTrasl(d,0,q1).

```
(%i1) rotTrasl(a,q1,k):=block([res],
                             zero:ident(3),
                             zero[3][1]:0,
                             zero[3][2]:0,
                             zero[3][3]:1,
                             if a= r then(
                                  zero[1][1]:cos(q1),
                                  zero[1][2]:-sin(q1),
                                  zero[2][1]:sin(q1),
                                  zero[2][2]:cos(q1),
                                  res:zero)
                             elseif a = d then(
                                zero[1][3]:k[1],
                                zero[2][3]:k[2],
                                res:zero)
                             else
                               res:"Not rotation or traslation selected"
                             )
```

(%o1) rotTrasl(a, $q1$, k) := **block** ([res], zero: ident(3), $(\mathrm{zero}_3)_1$: 0, $(\mathrm{zero}_3)_2$: 0, $(\mathrm{zero}_3)_3$: 1, **if** $a = r$ **then** ($(\mathrm{zero}_1)_1$: $\cos(q1)$, $(\mathrm{zero}_1)_2$: $-\sin(q1)$, $(\mathrm{zero}_2)_1$: $\sin(q1)$, $(\mathrm{zero}_2)_2$: $\cos(q1)$, res: zero) **elseif** $a = d$ **then** ($(\mathrm{zero}_1)_3$: k_1, $(\mathrm{zero}_2)_3$: k_2, res: zero) **else** res: Not rotation or traslation selected)

Matrice di rotazione $Q(0,0)$ in quanto non si necessità una rotazione per far coincidere gli assi del sitema di riferimento R_0 con quelli del sistema di riferimento R_1.

```
(%i2) Q:rotTrasl(r,0,[0,0])
```

(%o2) $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

Matrice di traslazione $R(I, q_1)$ in quanto occorre traslare di q1, variabile di giunto prismatico, per far coincidere le origini del sistema di riferimento R_0 con R_1.

```
(%i3) R:rotTrasl(d,0,[q[1],0])
```

(%o3) $\begin{pmatrix} 1 & 0 & q_1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

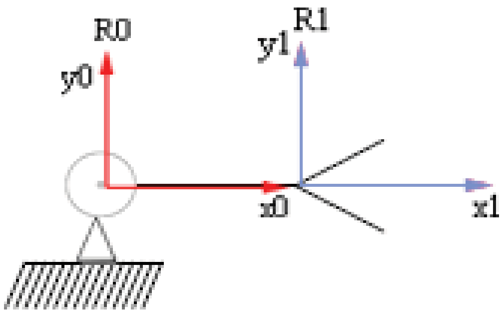
La matrice di trasformazione risultante, che rappresenta la cinematica diretta del robot 1 DOF prismatico T_{01}, risulterà uguale a: $T_{01} = Q(0,0)\, R(I, q_1)$

```
(%i4) TP[01]:Q.R
```

(%o4) $\begin{pmatrix} 1 & 0 & q_1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

(%i5)

1 DOF R - 1 grado di libertà rotoidale



$$T_{01} := \text{matrice di trasformazione da}\, R_o \rightarrow R_1$$

$$T_{01}(q_1) = \begin{pmatrix} \text{rotazione} & \text{traslazione} \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} R(q_1) & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} I & L_1 e_x \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} R(q_1) & L_1\, e_x\, R(q_1) \\ 0 & 1 \end{pmatrix} =$$

$$= \begin{pmatrix} c_1 & -s_1 & L_1 c_1 \\ s_1 & c_1 & L_1\, s_1 \\ 0 & 0 & 1 \end{pmatrix} \quad L_1 := \text{distanza che congiunge}\, R_0\, \text{e}\, R_1$$

N.B.: $\boldsymbol{c_1 \longrightarrow \cos(q_1);\, s_1 \longrightarrow \sin(q_1);}$

La matrice di rotazione $Q(q_1,\, 0)$ permette di orientare il sistema di riferimento R_0 come il sistema di riferimento $\text{dell}'\text{end} - \text{effector}\, R_1$. Poiché il giunto è rotoidale, dovremo ruotare R_0 di un angolo pari alla variabile di giunto q_1.

```
(%i5) Q:rotTrasl(r,q[1],0);
```

$$\text{(\%o5)} \begin{pmatrix} \cos(q_1) & -\sin(q_1) & 0 \\ \sin(q_1) & \cos(q_1) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

La matrice di traslazione $R(I, L_1)$ permette di far coincidere le origini del sistema di riferimento R_0 e R_1 tramite una traslazione L_1, coincidente con la distanza tra il sistema di riferimento R_0 e R_1.

```
(%i6) R:rotTrasl(d,0,[L[1],0]);
```

$$\text{(\%o6)} \begin{pmatrix} 1 & 0 & L_1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

La matrice di trasformazione risultante, che rappresenta la cinematica diretta del robot 1 DOF rotoidale T_{01}, risulterà uguale a: $T_{01} = Q(0, 0)\, R(I, q_1)$

```
(%i7) TR[01]:Q.R;
```

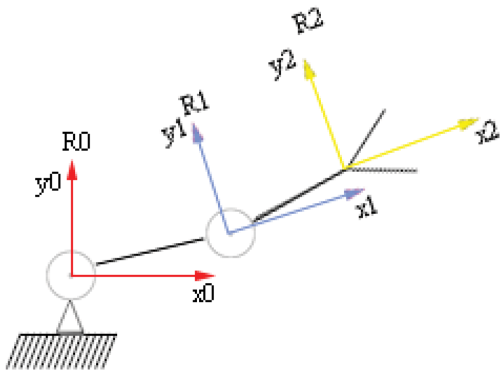
$$\text{(\%o7)} \begin{pmatrix} \cos(q_1) & -\sin(q_1) & L_1 \cos(q_1) \\ \sin(q_1) & \cos(q_1) & L_1 \sin(q_1) \\ 0 & 0 & 1 \end{pmatrix}$$

```
(%i8)
```

-I termini $\begin{pmatrix} L_1\cos(q_1) \\ L_1\sin(q_1) \end{pmatrix}$ rappresentano la cinematica di posizione in termini di coordinate dell'end-effector nel sistema di riferimento inerziale

-Il blocco 2x2 $\begin{pmatrix} \cos(q1) & -\sin(q1) \\ \sin(q1) & \cos(q1) \end{pmatrix}$ rappresenta la cinematica di orientamento del sistema di riferimento inerziale con lo stesso orientamento dell'end-effector.

2 DOF RR - 2 gradi di libertà entrambi rotoidali



$$T_{02} := \text{matrice di trasformazione da } R_o \rightarrow R_2$$

La matrice di trasformazione T_{01} corrisponde alla matrice di un robot planare 1 DOF R, mentre la matrice T_{12} corrisponde alla matrice di trasformazione tra la il sistema di riferimento R_1 e R_2:

$$T_{01}(q_1) = \begin{pmatrix} R(q_1) & L_1 R(q_1)\, e_x \\ 0 & 1 \end{pmatrix},$$

$$T_{12}(q_2) = \begin{pmatrix} R(q_2) & L_2\, R(q_2)\, e_x \\ 0 & 1 \end{pmatrix},$$

$$T_{02} = T_{01}\, T_{12} = \begin{pmatrix} R(q_1) & L_1 R(q_1)\, e_x \\ 0 & 1 \end{pmatrix} \begin{pmatrix} R(q_2) & L_2\, R(q_2)\, e_x \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} R_{12} & L_2 R_{12} e_x + L_1 R_1 e_x \\ 0 & 1 \end{pmatrix}$$

con $R_{12} = R(q_1 + q_2) = R(q_1)\, R(q_2) = R_1\, R_2$

$$T_{02} = \begin{pmatrix} c_{12} & -s_{12} & L_2\, c_{12} + L_1\, c_1 \\ s_{12} & c_{12} & L_2\, s_{12} + L_1 s_1 \\ 0 & 0 & 1 \end{pmatrix}$$

con $L_1 := \text{distanza che congiunge } R_0 \text{ e } R_1$

$L_2 := \text{distanza che congiunge } R_1 \text{ e } R_2$

N.B.: $\boldsymbol{c_{12} \longrightarrow \cos(q_1 + q_2); s_{12} \longrightarrow \sin(q_1 + q_2);}$

Matrice di trasformazione T_{01} analoga al robot planare 1 DOF R

```
(%i8) TR[01];
```

(%o8) $$\begin{pmatrix} \cos(q_1) & -\sin(q_1) & L_1 \cos(q_1) \\ \sin(q_1) & \cos(q_1) & L_1 \sin(q_1) \\ 0 & 0 & 1 \end{pmatrix}$$

La matrice di rotazione $Q2(q_2,\ 0)$ permette di orientare il sistema di riferimento R_1 come il sistema di riferimento $\text{dell'end} - \text{effector}\ R_2$. Poiché il giunto è rotoidale, dovremo ruotare R_1 di un angolo pari alla variabile di giunto q_2.

```
(%i9) Q2:rotTrasl(r,q[2],0);
```

(%o9) $$\begin{pmatrix} \cos(q_2) & -\sin(q_2) & 0 \\ \sin(q_2) & \cos(q_2) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

La matrice di traslazione $R(I, L_2)$ permette di far coincidere le origini del sistema di riferimento R_1 e R_2 tramite una traslazione L_2, coincidente con la distanza tra il sistema di riferimento R_1 e R_2.

(%i10) `R2:rotTrasl(d,0,[L[2],0]);`

(%o10) $$\begin{pmatrix} 1 & 0 & L_2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

(%i11) `T[12]:Q2.R2;`

(%o11) $$\begin{pmatrix} \cos(q_2) & -\sin(q_2) & L_2\cos(q_2) \\ \sin(q_2) & \cos(q_2) & L_2\sin(q_2) \\ 0 & 0 & 1 \end{pmatrix}$$

(%i12) `TRR[02]:trigreduce(TR[01].T[12])`

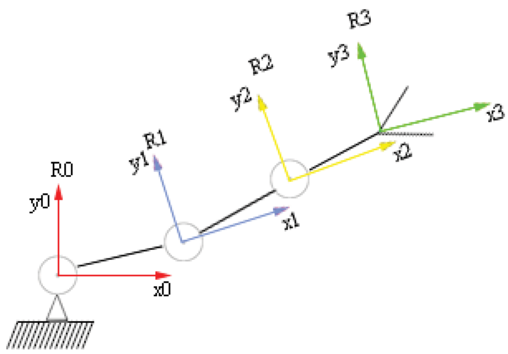
(%o12) $$\begin{pmatrix} \cos(q_2+q_1) & -\sin(q_2+q_1) & L_2\cos(q_2+q_1)+L_1\cos(q_1) \\ \sin(q_2+q_1) & \cos(q_2+q_1) & L_2\sin(q_2+q_1)+L_1\sin(q_1) \\ 0 & 0 & 1 \end{pmatrix}$$

(%i13)

-I termini $\begin{pmatrix} L_2\cos(q_2+q_1)+L_1\cos(q_1) \\ L_2\sin(q_2+q_1)+L_1\sin(q_1) \end{pmatrix}$ rappresentano la cinematica di posizione in termini di coordinate dell'end-effector nel sistema di riferimento inerziale

-Il blocco 2x2 $\begin{pmatrix} \cos(q_2+q_1) & -\sin(q_2+q_1) \\ \sin(q_2+q_1) & \cos(q_2+q_1) \end{pmatrix}$ rappresenta la cinematica di orientamento del sistema di riferimento inerziale con lo stesso orientamento dell'end-effector.

3 DOF RRR - 3 gradi di libertà rotoidali



$$T_{03} := \text{matrice di trasformazione da } R_0 \to R_3$$

$$T_{01}(q_1) = \begin{pmatrix} R(q_1) & L_1 R(q_1)\, e_x \\ 0 & 1 \end{pmatrix},$$

$$T_{12}(q_2) = \begin{pmatrix} R(q_2) & L_2\, R(q_2)\, e_x \\ 0 & 1 \end{pmatrix},$$

$$T_{23}(q_3) = \begin{pmatrix} R(q_3) & L_3\, R(q_3)\, e_x \\ 0 & 1 \end{pmatrix},$$

$$T_{03} = T_{01}\, T_{12}\, T_{23} = \begin{pmatrix} R(q_1) & L_1 R(q_1)\, e_x \\ 0 & 1 \end{pmatrix} \begin{pmatrix} R(q_2) & L_2\, R(q_2)\, e_x \\ 0 & 1 \end{pmatrix} \begin{pmatrix} R(q_3) & L_3\, R(q_3)\, e_x \\ 0 & 1 \end{pmatrix} =$$

$$= \begin{pmatrix} R_{12} & L_2 R_{12} e_x + L_1 R_1 e_x \\ 0 & 1 \end{pmatrix} \begin{pmatrix} R(q_3) & L_3\, R(q_3)\, e_x \\ 0 & 1 \end{pmatrix} =$$

$$= \begin{pmatrix} R_1 R_2 R_3 & L_3 R_1 R_2 R_3 e_x + L_2 R_{12} e_x + L_1 R_1 e_x \\ 0 & 1 \end{pmatrix}$$

con $R_{12} = R(q_1+q_2) = R(q_1)\,R(q_2) = R_1\,R_2$,

$L_1 :=$ distanza che congiunge R_0 e R_1

$L_2 :=$ distanza che congiunge R_1 e R_2

$L_3 :=$ distanza che congiunge R_2 e R_3

La matrice di trasformazione T_{02} è identica alla matrice di un robot planare 2 DOF RR

(%i13) `TRR[02];`

$$\text{(\%o13)}\quad \begin{pmatrix} \cos(q_2+q_1) & -\sin(q_2+q_1) & L_2\cos(q_2+q_1)+L_1\cos(q_1) \\ \sin(q_2+q_1) & \cos(q_2+q_1) & L_2\sin(q_2+q_1)+L_1\sin(q_1) \\ 0 & 0 & 1 \end{pmatrix}$$

La matrice di rotazione $Q_3(q_3,\, 0)$ permette di orientare il sistema di riferimento R_2 come il sistema di riferimento $\mathrm{dell'end-effector}$ R_3. Poiché il giunto è rotoidale, dovremo ruotare R_1 di un angolo pari alla variabile di giunto q_3.

(%i14) `Q3:rotTrasl(r,q[3],0);`

$$\text{(\%o14)}\quad \begin{pmatrix} \cos(q_3) & -\sin(q_3) & 0 \\ \sin(q_3) & \cos(q_3) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

La matrice di traslazione $R(I, L_3)$ permette di far coincidere le origini del sistema di riferimento R_2 e R_3 tramite una traslazione L_3, coincidente con la distanza tra il sistema di riferimento R_2 e R_3.

(%i15) `R3:rotTrasl(d,0,[L[3],0]);`

$$\text{(\%o15)}\quad \begin{pmatrix} 1 & 0 & L_3 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

(%i16) `TRR[23]:Q3.R3;`

$$\text{(\%o16)}\quad \begin{pmatrix} \cos(q_3) & -\sin(q_3) & L_3\cos(q_3) \\ \sin(q_3) & \cos(q_3) & L_3\sin(q_3) \\ 0 & 0 & 1 \end{pmatrix}$$

(%i17) `TRRR[03]:trigreduce(TRR[02].TRR[23])`

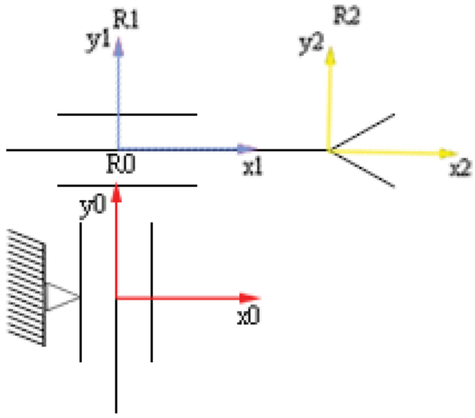
(%o17) $(\cos(q_3+q_2+q_1), -\sin(q_3+q_2+q_1), L_3\cos(q_3+q_2+q_1)+L_2\cos(q_2+q_1)+L_1\cos(q_1); \sin(q_3+q_2+q_1), \cos(q_3+q_2+q_1), L_3\sin(q_3+q_2+q_1)+L_2\sin(q_2+q_1)+L_1\sin(q_1); 0, 0, 1)$

(%i18)

-I termini $\begin{pmatrix} L_3\cos(q_3+q_2+q_1)+L_2\cos(q_2+q_1)+L_1\cos(q_1) \\ L_3\sin(q_3+q_2+q_1)+L_2\sin(q_2+q_1)+L_1\sin(q_1) \end{pmatrix}$ rappresentano la cinematica di posizione in termini di coordinate dell'end-effector nel sistema di riferimento inerziale;

-Il blocco 2x2 $\begin{pmatrix} \cos(q_3+q_2+q_1) & -\sin(q_3+q_2+q_1) \\ \sin(q_3+q_2+q_1) & \cos(q_3+q_2+q_1) \end{pmatrix}$ rappresenta la cinematica di orientamento del sistema di riferimento inerziale con lo stesso orientamento dell'end-effector.

Robot cartesiano 2 DOF PP - 2 gradi di libertà prismatici



$$T_{02} := \text{matrice di trasformazione da } R_o \rightarrow R_2$$

$$T_{01}(q_1) = \begin{pmatrix} I & q_1\, e_y \\ 0 & 1 \end{pmatrix},$$

$$T_{12}(q_1) = \begin{pmatrix} I & q_2\, e_x \\ 0 & 1 \end{pmatrix},$$

$$T_{02} = T_{01}\, T_{12} = \begin{pmatrix} I & q_1\, e_y \\ 0 & 1 \end{pmatrix} \begin{pmatrix} I & q_2\, e_x \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} I & q_2\, e_x + q_1 e_y \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & q_2 \\ 0 & 1 & q_1 \\ 0 & 0 & 1 \end{pmatrix}$$

La matrice di trasformazione T_{01} è composta dalla matrice di rotazione $Q(0,0)$ e matrice di traslazione $R(I, q_1 e_y)$ e fa coincidere il sistema di riferimento R_0 con il sistema di riferimento R_1.

```
(%i18) TPP[01]:rotTrasl(r,0,0).rotTrasl(d,0,[0,q[1]]);
```

(%o18) $\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & q_1 \\ 0 & 0 & 1 \end{pmatrix}$

La matrice di trasformazione T_{12} è composta dalla matrice di rotazione $Q(0,0)$ e matrice di traslazione $R(I, q_2 e_x)$ e fa coincidere il sistema di riferimento R_1 con il sistema di riferimento R_2.

```
(%i19) TPP[12]:rotTrasl(r,0,0).rotTrasl(d,0,[q[2],0]);
```

(%o19) $\begin{pmatrix} 1 & 0 & q_2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

```
(%i20) TPP[02]:TPP[01].TPP[12];
```

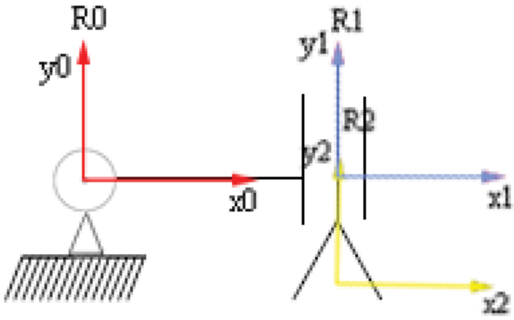
(%o20) $\begin{pmatrix} 1 & 0 & q_2 \\ 0 & 1 & q_1 \\ 0 & 0 & 1 \end{pmatrix}$

```
(%i21)
```

-I termini $\begin{pmatrix} q_2 \\ q_1 \end{pmatrix}$ rappresentano la cinematica di posizione in termini di coordinate dell'end-effector nel sistema di riferimento inerziale

-Il blocco 2x2 $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ rappresenta la cinematica di orientamento del sistema di riferimento inerziale con lo stesso orientamento dell'end-effector.

2 DOF RP - 2 gradi di libertà: il primo rotoidale, il secondo prismatico



Per questioni di leggibilità il sistema di riferimento R_2 (in giallo) è stato posto in basso e non al centro dell'end-effector.

$$T_{01} = \begin{pmatrix} R_1 & L_1 R_1 e_x \\ 0 & 1 \end{pmatrix},$$

$$T_{12} = \begin{pmatrix} I & q_2 e_y \\ 0 & 1 \end{pmatrix},$$

$$T_{02} = T_{01} T_{12} = \begin{pmatrix} R_1 & L_1 R_1 e_x \\ 0 & 1 \end{pmatrix} \begin{pmatrix} I & q_2 e_y \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} R_1 & R_1 q_2 e_y + L_1 R_1 e_x \\ 0 & 1 \end{pmatrix} =$$

$$= \begin{pmatrix} c_1 & -s_1 & -q_2 s_1 + L_1 c_1 \\ s_1 & c_1 & q_2 c_1 + L_1 s_1 \\ 0 & 0 & 1 \end{pmatrix}$$

(%i21) `TRP[01]:rotTrasl(r,q[1],[0,0]).rotTrasl(d,0,[L[1],0]);`

(%o21) $$\begin{pmatrix} \cos(q_1) & -\sin(q_1) & L_1 \cos(q_1) \\ \sin(q_1) & \cos(q_1) & L_1 \sin(q_1) \\ 0 & 0 & 1 \end{pmatrix}$$

(%i22) `TRP[12]:rotTrasl(r,0,[0,0]).rotTrasl(d,0,[0,q[2]]);`

(%o22) $$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & q_2 \\ 0 & 0 & 1 \end{pmatrix}$$

(%i23) `TRP[02](q1,q2):=TRP[01].TRP[12];`

(%o23) $\mathrm{TRP}_2(q1, q2) := \mathrm{TRP}_1 \cdot \mathrm{TRP}_{12}$

(%i24) `TRP[02](q1,q2);`

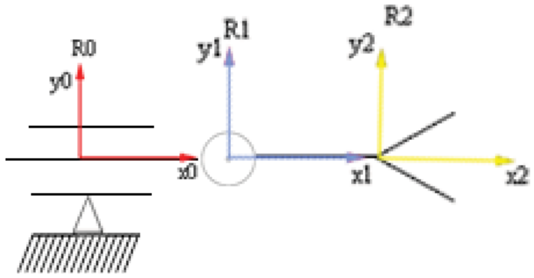
(%o24) $$\begin{pmatrix} \cos(q_1) & -\sin(q_1) & L_1 \cos(q_1) - q_2 \sin(q_1) \\ \sin(q_1) & \cos(q_1) & L_1 \sin(q_1) + q_2 \cos(q_1) \\ 0 & 0 & 1 \end{pmatrix}$$

(%i25)

-I termini $\begin{pmatrix} L_1 \cos(q_1) - q_2 \sin(q_1) \\ L_1 \sin(q_1) + q_2 \cos(q_1) \end{pmatrix}$ rappresentano la cinematica di posizione in termini di coordinate dell'end-effector nel sistema di riferimento inerziale

-Il blocco 2x2 $\begin{pmatrix} \cos(q_1) & -\sin(q_1) \\ \sin(q_1) & \cos(q_1) \end{pmatrix}$ rappresenta la cinematica di orientamento del sistema di riferimento inerziale con lo stesso orientamento dell'end-effector.

2 DOF PR - 2 gradi di libertà: prismatico e rotoidale



$$T_{02} := \text{matrice di trasformazione da } R_o \rightarrow R_2$$

$$T_{01} = \begin{pmatrix} I & q_1 e_x \\ 0 & 1 \end{pmatrix},$$

$$T_{12} = \begin{pmatrix} R_2 & L_2 R_1 e_x \\ 0 & 1 \end{pmatrix},$$

$$T_{02} = T_{01} T_{12} = \begin{pmatrix} I & q_1 e_x \\ 0 & 1 \end{pmatrix} \begin{pmatrix} R_2 & L_2 R_1 e_x \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} R_2 & q_1 e_x + L_2 R_2 e_x \\ 0 & 1 \end{pmatrix} =$$

$$= \begin{pmatrix} c_2 & -s_2 & -q_1 + L_2 c_2 \\ s_2 & c_2 & L_2 s_2 \\ 0 & 0 & 1 \end{pmatrix}$$

```
(%i25) TPR[01]:rotTrasl(r,0,[0,0]).rotTrasl(d,0,[q[1],0]);
```

(%o25) $\begin{pmatrix} 1 & 0 & q_1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

```
(%i26) TPR[12]:rotTrasl(r,q[2],[0,0]).rotTrasl(d,0,[L[2],0]);
```

(%o26) $\begin{pmatrix} \cos(q_2) & -\sin(q_2) & L_2 \cos(q_2) \\ \sin(q_2) & \cos(q_2) & L_2 \sin(q_2) \\ 0 & 0 & 1 \end{pmatrix}$

```
(%i27) TPR[02]:TPR[01].TPR[12];
```

(%o27) $\begin{pmatrix} \cos(q_2) & -\sin(q_2) & L_2 \cos(q_2) + q_1 \\ \sin(q_2) & \cos(q_2) & L_2 \sin(q_2) \\ 0 & 0 & 1 \end{pmatrix}$

(%i28)

-I termini $\begin{pmatrix} L_2 \cos(q_2) + q_1 \\ L_2 \sin(q_2) \end{pmatrix}$ rappresentano la cinematica di posizione in termini di coordinate dell'end-effector nel sistema di riferimento inerziale;

-Il blocco 2x2 $\begin{pmatrix} \cos(q_2) & -\sin(q_2) \\ \sin(q_2) & \cos(q_2) \end{pmatrix}$ rappresenta la cinematica di orientamento del sistema di riferimento inerziale con lo stesso orientamento dell'end-effector.

Asse, angolo, spostamento - Matrice di avvitamento

Scrivere una procedura che, dati asse, angolo, spostamento, generi la corrispondente matrice di avvitamento

Una matrice di avvitamento è una matrice ottenuta dalla rotazione e traslazione lungo uno stesso asse. Questa matrice ha la seguente struttura:

$$A_v(\theta, d) = \begin{pmatrix} e^{S(v)\theta} & d\,v \\ 0 & 1 \end{pmatrix}$$

In particolare:

- $\vartheta = 0 \Longrightarrow$traslazione;
- $d = 0 \Longrightarrow$ rotazione;

Poiché la traslazione e la rotazione sono effettuate lungo lo stesso asse, commutano. Quindi, una roto-traslazione è equivalente ad una trasla-rotazione.

$$T_R T_T \equiv T_T T_R$$

$$T_R = \begin{pmatrix} e^{S(v)\theta} & 0 \\ 0 & 1 \end{pmatrix}, T_T = \begin{pmatrix} I & d\,v \\ 0 & 1 \end{pmatrix},$$

$$(1) T_R T_T = \begin{pmatrix} e^{S(v)\theta} & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} I & d\,v \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} e^{S(v)\theta} & d\,e^{S(v)\theta} v \\ 0 & 1 \end{pmatrix}$$

$$(2) T_T T_R = \begin{pmatrix} I & d\,v \\ 0 & 1 \end{pmatrix} \begin{pmatrix} e^{S(v)\theta} & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} e^{S(v)\theta} & d\,v \\ 0 & 1 \end{pmatrix}$$

I prodotti matriciali (1) e (2) sono identici tenendo conto che v è l'asse di rotazione corrispondente alla rotazione effettuata:

$$e^{S(v)\theta} v = v$$

poiché v è l'asse di rotazione.

La funzione inverseLaplace(SI) calcola e restituisce in output l'antitrasformata di Laplace scorrendo tutti gli elementi della matrice data in input.

```
(%i1) inverseLaplace(SI,theta):=block([res],
                              M:SI,
                              MC:SI,
                              for i:1 thru 3 do(
                                for j:1 thru 3 do
                                  (
                                    aC:M[i,j],
                                    b:ilt(aC,s,theta),
                                    MC[i,j]:b
                                   )
                                ),
                              res:MC

                           )
```

(%o1) inverseLaplace(SI, ϑ) := **block** ([res], M: SI, MC: SI, **for** i **thru** 3 **do for** j **thru** 3 **do** (aC: $M_{i,j}$, b: ilt(aC, s, ϑ), MC$_{i,j}$: b), res: MC)

La funzione rotLaplace(k,ϑ) riceve in input un vettore v e ϑ per calcolare la matrice di rotazione relativa all'asse di rotazione v e angolo ϑ. In seguito, invoca la funzione inverseLaplace per effettuare l'inversa di laplace e, quindi, restituire in output l'effettiva matrice di rotazione.

```
(%i2) rotLaplace(k,theta):=block([res],
                         S:ident(3),
                         I:ident(3),
                       for i:1 thru 3 do
                         (
                         for j:1 thru 3 do
                            (
                              if i=j
                                 then S[i][j]:0
                              elseif j>i
                                 then (
                                temp:(-1)^(j-i)*k[3-remainder(i+j,3)],
                                       S[i][j]:temp,
                                       S[j][i]:-temp

                                        )
                              )
                          ),
                         res:inverseLaplace(invert(s*I-S),theta)


                        )
```

(%o2) $\mathrm{rotLaplace}(k,\vartheta):=\mathbf{block}\ ([\mathrm{res}], S\colon \mathrm{ident}(3), I\colon \mathrm{ident}(3),$ $\mathbf{for}\ i\ \mathbf{thru}\ 3\ \mathbf{do}\ \mathbf{for}\ j\ \mathbf{thru}\ 3\ \mathbf{do}\ \mathbf{if}\ i=j\ \mathbf{then}\ (S_i)_j\colon 0\ \mathbf{elseif}\ j>i\ \mathbf{then}\ (\mathrm{temp}\colon$ $(-1)^{j-i}\, k_{3-\mathrm{remainder}(i+j,3)}, (S_i)_j\colon \mathrm{temp}, (S_j)_i\colon -\mathrm{temp}), \mathrm{res}\colon \mathrm{inverseLaplace}(\mathrm{invert}(s\,I-S),\vartheta))$

La funzione $A_v(v,\theta,d)$ riceve in input l'asse di rotazione v, l'angolo ϑ e la traslazione d e restituisce la corrispondente matrice di avvitamento A_v:

$$A_v(\theta,d)=\begin{pmatrix} e^{S(v)\theta} & d\,v \\ 0 & 1 \end{pmatrix}$$

```
(%i3) Av(v,theta,d):=block([res],
                         Trot:rotLaplace(v,theta),
                         row:matrix([0,0,0,1]),
                         Atemp:addcol(Trot,d*transpose(v)),
                         A:addrow(Atemp,row),
                         res:trigsimp(trigrat(trigreduce(trigexpand(A))))
                         )
```

(%o3) $\mathrm{Av}(v,\vartheta,d) := \mathbf{block}\ ([\mathrm{res}], \mathrm{Trot}\colon \mathrm{rotLaplace}(v,\vartheta), \mathrm{row}\colon \begin{pmatrix} 0 & 0 & 0 & 1 \end{pmatrix}, \mathrm{Atemp}\colon \mathrm{addcol}(\mathrm{Trot},$ $d\,\mathrm{transpose}(v)), A\colon \mathrm{addrow}(\mathrm{Atemp},\mathrm{row}), \mathrm{res}\colon \mathrm{trigsimp}(\mathrm{trigrat}(\mathrm{trigreduce}(\mathrm{trigexpand}(A)))))$

```
(%i4) A[z](theta,d):=Av([0,0,1],theta,d);
```

(%o4) $A_z(\vartheta,d):=\mathrm{Av}([0,0,1],\vartheta,d)$

```
(%i5) A[z](theta,d)
```

$$\text{(\%o5)}\quad \begin{pmatrix} \cos(\vartheta) & -\sin(\vartheta) & 0 & 0 \\ \sin(\vartheta) & \cos(\vartheta) & 0 & 0 \\ 0 & 0 & 1 & d \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

(%i6) `A[x](alpha,a):=Av([1,0,0],alpha,a);`

(%o6) $A_x(\alpha,a):=\mathrm{Av}([1,0,0],\alpha,a)$

(%i7) `A[x](alpha,a)`

$$(\%o7)\quad \begin{pmatrix} 1 & 0 & 0 & a \\ 0 & \cos(\alpha) & -\sin(\alpha) & 0 \\ 0 & \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

(%i8)

Matrice di avvitamento $A_v(z,\theta,d)$, $A_v(x,\alpha,a)$

```
Maxima 5.44.0 http://maxima.sourceforge.net
using Lisp SBCL 2.0.0
Distributed under the GNU Public License. See the file COPYING.
Dedicated to the memory of William Schelter.
The function bug_report() provides bug reporting information.
```

```
(%i1) inverseLaplace(SI,theta):=block([res],
                          M:SI,
                          MC:SI,
                          for i:1 thru 3 do(
                            for j:1 thru 3 do
                               (
                                 aC:M[i,j],
                                 b:ilt(aC,s,theta),
                                 MC[i,j]:b
                                )
                             ),
                          res:MC

                        )
```

(%o1) $\mathrm{inverseLaplace}(\mathrm{SI}, \vartheta) := \textbf{block}\ ([\mathrm{res}], M{:}\ \mathrm{SI}, \mathrm{MC}{:}\ \mathrm{SI}, \textbf{for}\ i\ \textbf{thru}\ 3\ \textbf{do}\ \textbf{for}\ j\ \textbf{thru}\ 3\ \textbf{do}\ (\mathrm{aC}{:}\ M_{i,j}, b{:}\ \mathrm{ilt}(\mathrm{aC}, s, \vartheta), \mathrm{MC}_{i,j}{:}\ b), \mathrm{res}{:}\ \mathrm{MC})$

```
(%i2) rotLaplace(k,theta):=block([res],
                      S:ident(3),
                      I:ident(3),
                    for i:1 thru 3 do
                      (
                      for j:1 thru 3 do
                         (
                           if i=j
                              then S[i][j]:0
                           elseif j>i
                              then (
                             temp:(-1)^(j-i)*k[3-remainder(i+j,3)],
                                    S[i][j]:temp,
                                    S[j][i]:-temp

                                     )
                           )
                       ),
                      res:inverseLaplace(invert(s*I-S),theta)


                    )
```

(%o2) $\mathrm{rotLaplace}(k, \vartheta) := \textbf{block}\ ([\mathrm{res}], S{:}\ \mathrm{ident}(3), I{:}\ \mathrm{ident}(3), \textbf{for}\ i\ \textbf{thru}\ 3\ \textbf{do}\ \textbf{for}\ j\ \textbf{thru}\ 3\ \textbf{do}\ \textbf{if}\ i=j\ \textbf{then}\ (S_i)_j{:}\ 0\ \textbf{elseif}\ j>i\ \textbf{then}\ (\mathrm{temp}{:}\ (-1)^{j-i} k_{3-\mathrm{remainder}(i+j,3)}, (S_i)_j{:}\ \mathrm{temp}, (S_j)_i{:}\ -\mathrm{temp}), \mathrm{res}{:}\ \mathrm{inverseLaplace}(\mathrm{invert}(s\,I-S), \vartheta))$

(%i3) Av(v,theta,d):=block([res],
```
Av(v,theta,d):=block([res],
                     Trot:rotLaplace(v,theta),
                     row:matrix([0,0,0,1]),
                     Atemp:addcol(Trot,d*transpose(v)),
                     A:addrow(Atemp,row),
                     res:trigsimp(trigrat(trigreduce(trigexpand(A))))
                     )
```

(%o3) $\mathrm{Av}(v, \vartheta, d) := \mathbf{block}\ ([\mathrm{res}], \mathrm{Trot}: \mathrm{rotLaplace}(v, \vartheta), \mathrm{row}: \begin{pmatrix} 0 & 0 & 0 & 1 \end{pmatrix}, \mathrm{Atemp}: \mathrm{addcol}(\mathrm{Trot}, d\, \mathrm{transpose}(v)), A: \mathrm{addrow}(\mathrm{Atemp}, \mathrm{row}), \mathrm{res}: \mathrm{trigsimp}(\mathrm{trigrat}(\mathrm{trigreduce}(\mathrm{trigexpand}(A)))))$

(%i4) `A[z](theta,d):=Av([0,0,1],theta,d);`

(%o4) $A_z(\vartheta, d) := \mathrm{Av}([0, 0, 1], \vartheta, d)$

Matrice di avvitamento $A_v(z, \theta, d)$:

(%i5) `A[z](theta,d);`

(%o5)
$$\begin{pmatrix} \cos(\vartheta) & -\sin(\vartheta) & 0 & 0 \\ \sin(\vartheta) & \cos(\vartheta) & 0 & 0 \\ 0 & 0 & 1 & d \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Matrice di avvitamento $A_v(x, \theta, d)$

(%i6) `A[x](alpha,a):=Av([1,0,0],alpha,a);`

(%o6) $A_x(\alpha, a) := \mathrm{Av}([1, 0, 0], \alpha, a)$

(%i7) `A[x](alpha,a);`

(%o7)
$$\begin{pmatrix} 1 & 0 & 0 & a \\ 0 & \cos(\alpha) & -\sin(\alpha) & 0 \\ 0 & \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

(%i8)

Calcolare $Q_{i-1,i}$

La matrice di trasformazione omogenea $Q_{i-1,i}$ consente di sovrapporre il sistema di riferimento R_{i-1} con il sistema di riferimento R_i tramite 2 matrici di avvitamento $A_v(z,\theta,d), A_v(x,\alpha,a)$ nel seguente ordine.

Per ogni grado di libertà:

$$Q_{i-1,i} = A_v(z,\theta_i,d_i)\, A_v(x,\alpha_i,a_i)$$

```
(%i1) inverseLaplace(SI,theta):=block([res],
                                 M:SI,
                                 MC:SI,
                                 for i:1 thru 3 do(
                                   for j:1 thru 3 do
                                       (
                                         aC:M[i,j],
                                         b:ilt(aC,s,theta),
                                         MC[i,j]:b
                                        )
                                    ),
                                 res:MC

                              )
```

(%o1) inverseLaplace(SI, ϑ) := **block** ([res], M: SI, MC: SI, **for** i **thru** 3 **do for** j **thru** 3 **do** (aC: $M_{i,j}$, b: ilt(aC, s, ϑ), $\mathrm{MC}_{i,j}$: b), res: MC)

```
(%i2) rotLaplace(k,theta):=block([res],
                            S:ident(3),
                            I:ident(3),
                          for i:1 thru 3 do
                            (
                            for j:1 thru 3 do
                                (
                                  if i=j
                                     then S[i][j]:0
                                  elseif j>i
                                     then (
                                    temp:(-1)^(j-i)*k[3-remainder(i+j,3)],
                                           S[i][j]:temp,
                                           S[j][i]:-temp

                                            )
                                  )
                             ),
                            res:inverseLaplace(invert(s*I-S),theta)


                          )
```

(%o2) rotLaplace(k,ϑ) := **block** ([res], S: ident(3), I: ident(3), **for** i **thru** 3 **do for** j **thru** 3 **do if** $i=j$ **then** $(S_i)_j$: 0 **elseif** $j>i$ **then** (temp: $(-1)^{j-i}\, k_{3-\mathrm{remainder}(i+j,3)}$, $(S_i)_j$: temp, $(S_j)_i$: $-$temp), res: inverseLaplace(invert($s\,I-S$), ϑ))

```
(%i3) Av(v,theta,d):=block([res],
                            Trot:rotLaplace(v,theta),
                            row:matrix([0,0,0,1]),
                            Atemp:addcol(Trot,d*transpose(v)),
                            A:addrow(Atemp,row),
                            res:trigsimp(trigrat(trigreduce(trigexpand(A))))
                            )
```

(%o3) $\mathrm{Av}(v, \vartheta, d) := \textbf{block}\ ([\mathrm{res}], \mathrm{Trot}: \mathrm{rotLaplace}(v, \vartheta), \mathrm{row}: \begin{pmatrix} 0 & 0 & 0 & 1 \end{pmatrix}, \mathrm{Atemp}: \mathrm{addcol}(\mathrm{Trot}, d\,\mathrm{transpose}(v)), A: \mathrm{addrow}(\mathrm{Atemp}, \mathrm{row}), \mathrm{res}: \mathrm{trigsimp}(\mathrm{trigrat}(\mathrm{trigreduce}(\mathrm{trigexpand}(A)))))$

```
(%i4) Az(theta,d):=Av([0,0,1],theta,d);
```

(%o4) $\mathrm{Az}(\vartheta, d) := \mathrm{Av}([0, 0, 1], \vartheta, d)$

```
(%i5) Az(theta,d);
```

(%o5) $$\begin{pmatrix} \cos(\vartheta) & -\sin(\vartheta) & 0 & 0 \\ \sin(\vartheta) & \cos(\vartheta) & 0 & 0 \\ 0 & 0 & 1 & d \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
(%i6) Ax(alpha,a):=Av([1,0,0],alpha,a);
```

(%o6) $\mathrm{Ax}(\alpha, a) := \mathrm{Av}([1, 0, 0], \alpha, a)$

```
(%i7) Ax(alpha,a);
```

(%o7) $$\begin{pmatrix} 1 & 0 & 0 & a \\ 0 & \cos(\alpha) & -\sin(\alpha) & 0 \\ 0 & \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

La matrice Q riceve in input ϑ,d,α,a:
-ϑ,α sono angoli corrispondenti alle rotazioni necessarie a far sovrapporre il sistema di riferimento R_{i-1} con il sistema di riferimento R_i rispettivamente con la matrice di avvitamento $A_v(z, \theta, d)$ e $A_v(x, \alpha, a)$;
-d,a sono posizioni che corrispondo alle traslazioni necessarie a far sovrapporre il sistema di riferimento R_{i-1} con il sistema di riferimento R_i rispettivamente lungo l'asse z e x corrispondenti alle matrici di avvitamento $A_v(z, \theta, d)$ e $A_v(x, \alpha, a)$;

```
(%i8) Q(theta,d,alpha,a):=block([res],
      res:trigexpand(trigsimp(trigrat(trigreduce(trigexpand(Az(theta,d).Ax(alpha,
      a))))))
                                       )
```

(%o8) $Q(\vartheta, d, \alpha, a) := \textbf{block}\ ([\mathrm{res}], \mathrm{res}: \mathrm{trigexpand}(\mathrm{trigsimp}(\mathrm{trigrat}(\mathrm{trigreduce}(\mathrm{trigexpand}(\mathrm{Az}(\vartheta, d) \cdot \mathrm{Ax}(\alpha, a)))))))$

```
(%i9) Q(theta,d,alpha,a)
```

(%o9) $$\begin{pmatrix} \cos(\vartheta) & -\cos(\alpha)\sin(\vartheta) & \sin(\alpha)\sin(\vartheta) & a\cos(\vartheta) \\ \sin(\vartheta) & \cos(\alpha)\cos(\vartheta) & -\sin(\alpha)\cos(\vartheta) & a\sin(\vartheta) \\ 0 & \sin(\alpha) & \cos(\alpha) & d \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
(%i10)
(%i13)
```

Cinematica diretta 2 DOF planare

***N.B.:* le grandezze diverse da quelle di giunto q_i sono L_i, D_i. Esse sono rispettivamente la distanza tra i sistemi di riferimento R_i e R_{i+1} nelle operazioni della matrice avvitamento $A_z(\theta, d)$ e $A_x(\alpha, a)$.**

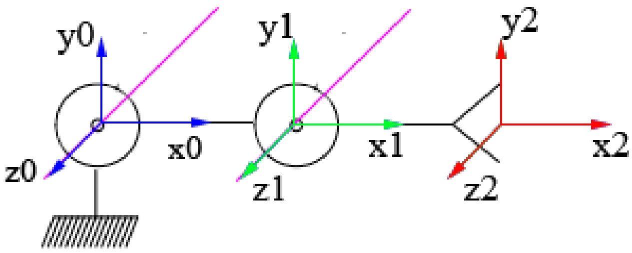


Tabella Denavit-Hartenberg 2 DOF Planare

	ϑ	d	α	a
1	q_1	0	0	L_1
2	q_2	0	0	L_2

Tabella 1.

Funzioni ausiliarie:

```
(%i1) isRotation(M):=block([MC,res,I,MMT,detM],
                        I:ident(3),
                        MC:ident(3),
                        for i:1 thru 3 do
                        (
                        for j:1 thru 3 do
                           (
                              MC[i][j]:M[i][j]
                              )
                           ),
                        MMT:trigsimp(expand(MC.transpose(MC))),
                        detM:trigsimp(expand(determinant(MC))),

                        if MMT=I  and detM=1
                           then(

                                return(res:1)
                                )

                        else(

                              res: "R is not rotation matrix"
                              )
                        )
```

(%o1) $\mathrm{isRotation}(M) := \mathbf{block}\ ([\mathrm{MC}, \mathrm{res}, I, \mathrm{MMT}, \mathrm{detM}], I\colon \mathrm{ident}(3), \mathrm{MC}\colon \mathrm{ident}(3),$
$\mathbf{for}\ i\ \mathbf{thru}\ 3\ \mathbf{do}\ \mathbf{for}\ j\ \mathbf{thru}\ 3\ \mathbf{do}\ (\mathrm{MC}_i)_j\colon (M_i)_j, \mathrm{MMT}\colon \mathrm{trigsimp}(\mathrm{expand}(\mathrm{MC}\cdot \mathrm{transpose}(\mathrm{MC}))),$

detM: trigsimp(expand(determinant(MC))), **if** MMT $= I \wedge$ detM $= 1$ **then** return(res: 1) **else** res: R is not rotation matrix)

```
(%i2) inverseLaplace(SI,theta):=block([res,M,MC,aC,b],
                              M:SI,
                              MC:SI,
                              for i:1 thru 3 do(
                                for j:1 thru 3 do
                                   (
                                     aC:M[i,j],
                                     b:ilt(aC,s,theta),
                                     MC[i,j]:b
                                    )
                                 ),
                              res:MC

                           )
```

(%o2) inverseLaplace(SI, ϑ) := **block** ([res, M, MC, aC, b], M: SI, MC: SI, **for** i **thru** 3 **do** **for** j **thru** 3 **do** (aC: $M_{i,j}$, b: ilt(aC, s, ϑ), $\mathrm{MC}_{i,j}$: b), res: MC)

```
(%i3) rotLaplace(k,theta):=block([res,S,I,temp],
                          S:ident(3),
                          I:ident(3),
                        for i:1 thru 3 do
                          (
                          for j:1 thru 3 do
                             (
                               if i=j
                                  then S[i][j]:0
                               elseif j>i
                                  then (
                                 temp:(-1)^(j-i)*k[3-remainder(i+j,3)],
                                        S[i][j]:temp,
                                        S[j][i]:-temp

                                         )
                               )
                           ),
                          res:inverseLaplace(invert(s*I-S),theta)


                        )
```

(%o3) rotLaplace(k, ϑ) := **block** ([res, S, I, temp], S: ident(3), I: ident(3), **for** i **thru** 3 **do** **for** j **thru** 3 **do** **if** $i = j$ **then** $(S_i)_j$: 0 **elseif** $j > i$ **then** (temp: $(-1)^{j-i}\, k_{3-\mathrm{remainder}(i+j,3)}$, $(S_i)_j$: temp, $(S_j)_i$: $-$temp), res: inverseLaplace(invert($s\,I - S$), ϑ))

```
(%i4) Av(v,theta,d):=block([res,Tot,row,Atemp,A],
                          Trot:rotLaplace(v,theta),
                          row:matrix([0,0,0,1]),
                          Atemp:addcol(Trot,d*transpose(v)),
                          A:addrow(Atemp,row),
                          res:A
                          )
```

(%o4) Av(v, ϑ, d) := **block** ([res, Tot, row, Atemp, A], Trot: rotLaplace(v, ϑ), row: $\begin{pmatrix} 0 & 0 & 0 & 1 \end{pmatrix}$, Atemp: addcol(Trot, d transpose(v)), A: addrow(Atemp, row), res: A)

```
(%i5) Q(theta,d,alpha,a):=block([res,tempMat,Qtrasf],
                              tempMat:Av([0,0,1],theta,d).Av([1,0,0],alpha,a),
                              Qtrasf:zeromatrix(4,4),
                              for i:1 thru 4 do
                        (
                        for j:1 thru 4 do
                            (
                              Qtrasf[i][j]:trigreduce(tempMat[i][j])
                                          )
                            ),

                              res:Qtrasf
                              )
```

(%o5) $Q(\vartheta, d, \alpha, a) := \mathbf{block}\ ([\mathrm{res}, \mathrm{tempMat}, \mathrm{Qtrasf}], \mathrm{tempMat}\colon \mathrm{Av}([0,0,1], \vartheta, d) \cdot \mathrm{Av}([1,0,0], \alpha, a),$ $\mathrm{Qtrasf}\colon \mathrm{zeromatrix}(4,4),\ \mathbf{for}\ i\ \mathbf{thru}\ 4\ \mathbf{do}\ \mathbf{for}\ j\ \mathbf{thru}\ 4\ \mathbf{do}\ (\mathrm{Qtrasf}_i)_j\colon \mathrm{trigreduce}((\mathrm{tempMat}_i)_j), \mathrm{res}\colon$ $\mathrm{Qtrasf})$

```
(%i6) let(sin(q[1]), s[1]);
```

(%o6) $\sin(q_1) \longrightarrow s_1$

```
(%i7) let(sin(q[2]), s[2]);
```

(%o7) $\sin(q_2) \longrightarrow s_2$

```
(%i8) let(cos(q[1]),c[1]);
```

(%o8) $\cos(q_1) \longrightarrow c_1$

```
(%i9) let(cos(q[2]),c[2]);
```

(%o9) $\cos(q_2) \longrightarrow c_2$

```
(%i10) let(sin(q[1]+q[2]), s[12]);
```

(%o10) $\sin(q_2 + q_1) \longrightarrow s_{12}$

```
(%i11) let(cos(q[1]+q[2]),c[12]);
```

(%o11) $\cos(q_2 + q_1) \longrightarrow c_{12}$

```
(%i12)
```

Cinematica diretta:

```
(%i12) Q[3](q1,q2,L1,L2):=trigsimp(trigrat(trigreduce(trigexpand(Q(q1,0,0,
       L1).Q(q2,0,0,L2)))));
```

(%o12) $Q_3(q1, q2, L1, L2) := \mathrm{trigsimp}(\mathrm{trigrat}(\mathrm{trigreduce}(\mathrm{trigexpand}(Q(q1, 0, 0, L1) \cdot Q(q2, 0, 0,$ $L2)))))$

```
(%i13) Qplanare:Q[3](q[1],q[2],L[1],L[2]);
```

(%o13) $$\begin{pmatrix} \cos(q_2+q_1) & -\sin(q_2+q_1) & 0 & L_2\cos(q_2+q_1) + L_1\cos(q_1) \\ \sin(q_2+q_1) & \cos(q_2+q_1) & 0 & L_2\sin(q_2+q_1) + L_1\sin(q_1) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
(%i14) letsimp(Qplanare);
```

(%o14) $$\begin{pmatrix} c_{12} & -s_{12} & 0 & L_2 c_{12} + L_1 c_1 \\ s_{12} & c_{12} & 0 & L_2 s_{12} + L_1 s_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

(%i15)

Cinematica inversa

Per effettuare la cinemetica inversa di orientamento sono necessari almeno 3 DOF (gradi di libertà), quindi si effettua solamente la cinematica inversa di posizione per questo tipo di robot. Occorre inizialmente individuare lo spazio di lavoro, le soluzioni generi e singolari ed infine le variabili di giunto q_i.

Dalla cinematica diretta del robot 2DOF sappiamo che il punto x,y viene identificato dal vettore:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} L_2 \cos(q_2 + q_1) + L_1 \cos(q_1) \\ L_2 \sin(q_2 + q_1) + L_1 \sin(q_1) \\ 0 \end{pmatrix}$$

$$\begin{pmatrix} x \\ y \end{pmatrix} = R(q_1 + q_2) \begin{pmatrix} L_2 \\ 0 \end{pmatrix} + R(q_1) \begin{pmatrix} L_1 \\ 0 \end{pmatrix}$$

Poiché $R(q_1 + q_2) = R(q_1)\, R(q_2)$, è possibile mettere in evidenza $R(q_1)$:

$$\begin{pmatrix} x \\ y \end{pmatrix} = R(q_1) \left\{ R(q_2) \begin{pmatrix} L_2 \\ 0 \end{pmatrix} + \begin{pmatrix} L_1 \\ 0 \end{pmatrix} \right\}$$

La matrice di rotazione $R(q_1)$ ha non varia la norma del vettore $\begin{pmatrix} \mathrm{x} \\ \mathrm{y} \end{pmatrix}$ e $R(q_2) \begin{pmatrix} L_2 \\ 0 \end{pmatrix} + \begin{pmatrix} L_1 \\ 0 \end{pmatrix}$. Quindi possiamo imporre che abbiano la stessa norma. In particolare:

$$\left\| \begin{pmatrix} x \\ y \end{pmatrix} \right\|_2 = \left\| R(q_2) \begin{pmatrix} L_2 \\ 0 \end{pmatrix} + \begin{pmatrix} L_1 \\ 0 \end{pmatrix} \right\|_2$$

$$\begin{pmatrix} L_2 & 0 \end{pmatrix} R(q_2)^T R(q_2) \begin{pmatrix} L_2 \\ 0 \end{pmatrix} + \begin{pmatrix} L_1 & 0 \end{pmatrix} \begin{pmatrix} L_1 \\ 0 \end{pmatrix} + 2 \begin{pmatrix} L_1 & 0 \end{pmatrix} R(q_2) \begin{pmatrix} L_2 \\ 0 \end{pmatrix}$$

$$x^2 + y^2 = L_2^2 + L_1^2 + 2\, L_1\, L_2 \cos(q_2)$$

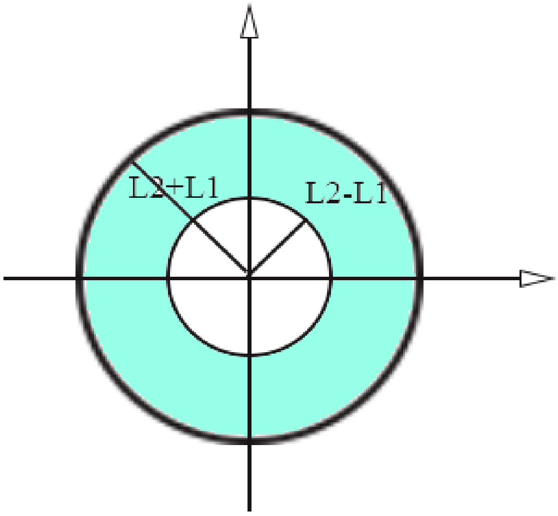
$$\cos(q_2) = \frac{x^2 + y^2 - L_2^2 - L_1^2}{2\, L_1\, L_2}$$

Poiché $-1 \leqslant \cos(q_2) \leqslant 1$, otteniamo infine lo spazio operativo del 2DOF planare:

$$-1 \leqslant \frac{x^2 + y^2 - L_2^2 - L_1^2}{2\, L_1\, L_2} \leqslant 1$$

$$L_2^2 + L_1^2 - 2\, L_1\, L_2 \leqslant x^2 + y^2 \leqslant 2\, L_1\, L_2 + L_2^2 + L_1^2$$

Le disequazioni delimitano due circonferenze di raggio $L_2 + L_1$ e $L_2 - L_1$, in cui la regione valida presa inconsiderazione è quella regione di spazio compresa tra le curve.



A questo punto è possibile determinare l'espressione della variabile di giunto q_2:

$$\cos(q_2) = \frac{x^2 + y^2 - L_2^2 - L_1^2}{2\, L_1\, L_2}$$

$$\sin(q_2) = \pm\sqrt{1 - \cos(q_2)^2} = \pm\sqrt{1 - \frac{x^2 + y^2 - L_2^2 - L_1^2}{2\, L_1\, L_2}}$$

In particolare, si hanno due soluzioni generiche se $\left|\frac{x^2 + y^2 - L_2^2 - L_1^2}{2\, L_1\, L_2}\right| \neq 1$. Nel caso in cui $\left|\frac{x^2 + y^2 - L_2^2 - L_1^2}{2\, L_1\, L_2}\right| = 1$, si ha una soluzione singolare.

A questo punto la quantità $R(q_2)\begin{pmatrix} L_2 \\ 0 \end{pmatrix} + \begin{pmatrix} L_1 \\ 0 \end{pmatrix}$ è nota:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \cos(q_1) & -\sin(q_1) \\ \sin(q_1) & \cos(q_1) \end{pmatrix} \begin{pmatrix} L_2 \cos(q_2) + L_1 \\ \sin(q_2)\, L_2 \end{pmatrix}$$

$$\begin{pmatrix} \cos(q_1) \\ \sin(q_1) \end{pmatrix} = \frac{1}{(L_2 \cos(q_2) + L_1)^2 + L_2^2 \sin(q_2)^2} \begin{pmatrix} L_2 \cos(q_2) + L_1 & \sin(q_2)\, L_2 \\ -\sin(q_2)\, L_2 & L_2 \cos(q_2) + L_1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

$$= \frac{1}{L_2^2 \cos(q_2)^2 + L_1^2 + 2\, L_2 L_1 \cos(q_2) + L_1 + L_2^2 \sin(q_2)^2} \begin{pmatrix} (L_2 \cos(q_2) + L_1)\, x + \sin(q_2)\, L_2\, y \\ -\sin(q_2)\, L_2\, x + (L_2 \cos(q_2) + L_1)\, y \end{pmatrix}$$

Da cui si ottiene la variabile di giunto q_1;

$$q_1 = \mathrm{atan2}\!\left(\frac{-\sin(q_2)\, L_2\, x + (L_2 \cos(q_2) + L_1)\, y}{L_2^2 \cos(q_2)^2 + L_1^2 + L_2^2 \sin(q_2)^2}, \frac{(L_2 \cos(q_2) + L_1)\, x + \sin(q_2)\, L_2\, y}{L_2^2 \cos(q_2)^2 + L_1^2 + L_2^2 \sin(q_2)^2} \right)$$

Poiché la quantità $L_2^2 \cos(q_2)^2 + L_1^2 + L_2^2 \sin(q_2)^2 > 0$ è possibile semplificara all'interno della funzione atan2:

$$q_1 = \mathrm{atan2}(\sin(q_2)\, L_2\, x - (L_2 \cos(q_2) + L_1)\, y, (L_2 \cos(q_2) + L_1)\, x - \sin(q_2)\, L_2\, y)$$

Il problema della cinematica inversa ed, in particolare il problema di orientamento inverso, per il robot 2DOF planare è risolto.

`Data la cinematica diretta del robot 2DOF planare, si calcola la cinematica inversa:`

(%i12)
```
Q2DOF:Q[3](%pi/3,%pi/3,10,15);
```

(%o12) $Q_3(\frac{\pi}{3},\frac{\pi}{3},10,15)$

(%i13)
```
calculate(x,y,L1,L2):=block([q2plus,q2minus,q1,res],
                                c2:(x^(2)+y^(2)-L1^(2)-L2^(2))/(2*L1*L2),
                                s2:sqrt(1-c2^2),

                                c1Num:combine(expand((L2*c2+L1)*x+s2*L2*y)),
                                s1Num:combine(expand(-s2*L2*x+(L2*c2+L1)*y)),

q1Den:L2^(2)*c2^(2)+L1^(2)+L2^(2)*s2^(2)+2*L1*L2*c2,
                         if abs(c2)=1 then print("La soluzione è singolare")
                                elseif q1Den>0 then(
                                    print("La soluzione non è singolare"),
                                    q1:atan2(s1Num,c1Num),
                                    q2alto:atan2(s2,c2),
                                    q2basso:atan2(-s2,c2),
                                    res:[[q2alto,q1],[q2basso,q1]])
                                else (
                                    q1:atan2(s1Num/q1Den,c1Num/q1Den),
                                    q2alto:atan2(s2,c2),
                                    q2basso:atan2(-s2,c2),
                                    res:[[q2alto,q1],[q2basso,q1]])



                                )
```

(%o13) $\mathrm{calculate}(x,\, y,\, L1,\, L2) := \mathbf{block}\left([q2\mathrm{plus},\, q2\mathrm{minus},\, q1,\, \mathrm{res}],\, c2\colon \frac{x^2+y^2-L1^2-L2^2}{2\, L1\, L2},\, s2\colon \sqrt{1-c2^2},\, c1\mathrm{Num}\colon \mathrm{combine}(\mathrm{expand}((L2\, c2+L1)\, x+s2\, L2\, y)),\, s1\mathrm{Num}\colon \mathrm{combine}(\mathrm{expand}((-s2)\ L2\ x+(L2\ c2+L1)\ y)),\, q1\mathrm{Den}\colon L2^2\ c2^2+L1^2+L2^2\ s2^2+2\ L1\ L2\ c2,\, \mathbf{if}\ |c2|=1\ \mathbf{then}\ \mathrm{print}(\text{La soluzione è singolare })\ \mathbf{elseif}\ q1\mathrm{Den}>0\ \mathbf{then}\ (\mathrm{print}(\text{La soluzione non è singolare }),\, q1\colon \mathrm{atan2}(s1\mathrm{Num},\, c1\mathrm{Num}),\, q2\mathrm{alto}\colon \mathrm{atan2}(s2,\, c2),\, q2\mathrm{basso}\colon \mathrm{atan2}(-s2,\, c2),\, \mathrm{res}\colon [[q2\mathrm{alto},\, q1],\, [q2\mathrm{basso},\, q1]])\ \mathbf{else}\left(q1\colon \mathrm{atan2}\left(\frac{s1\mathrm{Num}}{q1\mathrm{Den}},\, \frac{c1\mathrm{Num}}{q1\mathrm{Den}}\right),\, q2\mathrm{alto}\colon \mathrm{atan2}(s2,\, c2),\, q2\mathrm{basso}\colon \mathrm{atan2}(-s2,\, c2),\, \mathrm{res}\colon [[q2\mathrm{alto},\, q1],\, [q2\mathrm{basso},\, q1]]\right)\right)$

(%i14)
```
inv2DOF(x,y,link1,link2):=block([res],


                              circleInt:link1^2+link2^2-2*link1*link2,
                              circleEst:link1^2+link2^2+2*link1*link2,
                            if x^2+y^2>= circleInt and x^2+y^2<= circleEst then
                              (
                               print("Il punto x,y è nello spazio di lavoro"),
                               res:calculate(x,y,link1,link2)

                               )
                               else res:"Punto x,y non è ammissibile"

                             )
```

(%o14) $\mathrm{inv2DOF}(x,\, y,\, \mathrm{link1},\, \mathrm{link2}) := \mathbf{block}\ ([\mathrm{res}],\, \mathrm{circleInt}\colon \mathrm{link1}^2+\mathrm{link2}^2+(-2)\ \mathrm{link1}\ \mathrm{link2},\, \mathrm{circleEst}\colon \mathrm{link1}^2+\mathrm{link2}^2+2\ \mathrm{link1}\ \mathrm{link2},\, \mathbf{if}\ x^2+y^2\geq \mathrm{circleInt} \wedge x^2+y^2\leq \mathrm{circleEst}\ \mathbf{then}$ (print(Il punto x,y è nello spazio di lavoro), res: calculate(x, y, link1, link2)) **else** res: Punto x,y non è ammissibile)

```
(%i15) inv2DOF(-5/2,((25*sqrt(3))/2),10,15);
```

Il punto x,y è nello spazio di lavoro

La soluzione non è singolare

(%o15) $\left[\left[\frac{\pi}{3},\frac{\pi}{3}\right],\left[-\frac{\pi}{3},\frac{\pi}{3}\right]\right]$

(%i16)

Singolarità di velocità

$$\begin{cases} x = L_1 c_1 + L_2 c_{12} \\ y = L_1 s_1 + L_2 s_{12} \end{cases}$$

$$J = \frac{\delta h}{\delta q} = \begin{pmatrix} -L_1 s_1 - L_2 s_{12} & -L_2 s_{12} \\ L_1 c_1 + L_2 c_{12} & L_2 c_{12} \end{pmatrix}$$

$$\det(J) = L_1 L_2 s_1 \Longrightarrow \det(J) = 0 \rightarrow q_2 = \begin{cases} 0 \\ \pi \end{cases}$$

```
(%i7) x:L[2]*cos (q[2]+q[1])+L[1]*cos (q[1]);
```

(%o7) $L_2 \cos(q_2 + q_1) + L_1 \cos(q_1)$

```
(%i8) y:L[2]*sin (q[2]+q[1])+L[1]*sin (q[1]);
```

(%o8) $L_2 \sin(q_2 + q_1) + L_1 \sin(q_1)$

```
(%i9) J:matrix([diff(x,q[1]),diff(x,q[2])],
                [diff(y,q[1]),diff(y,q[2])]
             );
```

(%o9) $\begin{pmatrix} -L_2 \sin(q_2+q_1) - L_1 \sin(q_1) & -L_2 \sin(q_2+q_1) \\ L_2 \cos(q_2+q_1) + L_1 \cos(q_1) & L_2 \cos(q_2+q_1) \end{pmatrix}$

```
(%i12) dJ:trigsimp(trigexpand(determinant(J)));
```

(%o12) $L_1\, L_2 \sin(q_2)$

Se $q_2 = 0$:

```
(%i13) Jq2:subst(q[2]=0,J);
```

(%o13) $\begin{pmatrix} -L_2 \sin(q_1) - L_1 \sin(q_1) & -L_2 \sin(q_1) \\ L_2 \cos(q_1) + L_1 \cos(q_1) & L_2 \cos(q_1) \end{pmatrix}$

```
(%i14) nullspace(Jq2);
```

Proviso: notequal($-L_2 \sin(q_1)$, 0)

(%o14) $\mathrm{span}\left(\begin{pmatrix} -L_2 \sin(q_1) \\ (L_2 + L_1)\sin(q_1) \end{pmatrix}\right)$

Si hanno singolarità per $v \in \mathrm{Im}\left\{\begin{pmatrix} -L_2 \sin(q_1) \\ (L_2 + L_1)\sin(q_1) \end{pmatrix}\right\}$.

Se $q_2 = \pi$

```
(%i15) Jq2:subst(q[2]=%pi,J);
```

(%o15) $\begin{pmatrix} L_2 \sin(q_1) - L_1 \sin(q_1) & L_2 \sin(q_1) \\ L_1 \cos(q_1) - L_2 \cos(q_1) & -L_2 \cos(q_1) \end{pmatrix}$

```
(%i16) nullspace(Jq2);
```

Proviso: notequal($L_2 \sin(q_1), 0$)

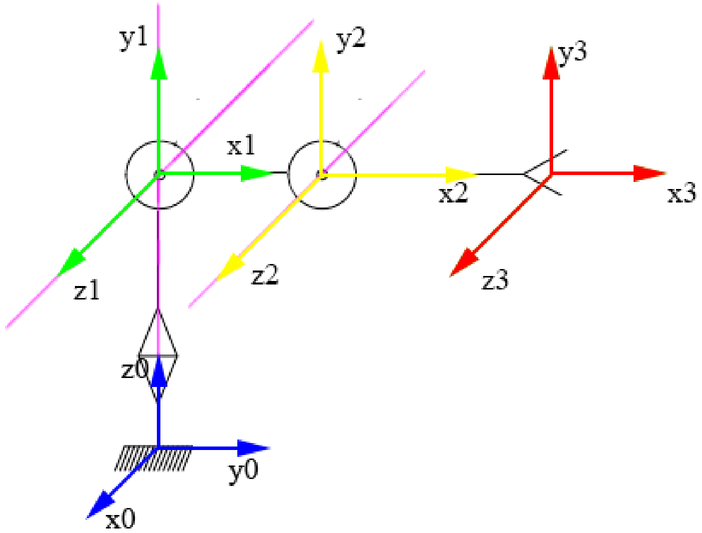
(%o16) $\operatorname{span}\left(\begin{pmatrix} L_2 \sin(q_1) \\ (L_1 - L_2)\sin(q_1) \end{pmatrix}\right)$

Si hanno singolarità per $v \in \operatorname{Im}\left\{\begin{pmatrix} L_2 \sin(q_1) \\ (L_1 - L_2)\sin(q_1) \end{pmatrix}\right\}$

(%i17)

Cinematica diretta Robot Antropomorfo

***N.B.:* le grandezze diverse da quelle di giunto q_i sono L_i, D_i. Esse sono rispettivamente la distanza tra i sistemi di riferimento R_i e R_{i+1} nelle operazioni della matrice avvitamento $A_z(\theta, d)$ e $A_x(\alpha, a)$.**



	ϑ	d	α	a
1	q_1	L_1	$\frac{\pi}{2}$	0
2	q_2	0	0	L_2
3	q_3	0	0	L_3

Tabella 1.

Funzioni ausiliarie:

```
(%i1) inverseLaplace(SI,theta):=block([res],
                        M:SI,
                        MC:SI,
                        for i:1 thru 3 do(
                          for j:1 thru 3 do
                             (
                               aC:M[i,j],
                               b:ilt(aC,s,theta),
                               MC[i,j]:b
                              )
                           ),
                        res:MC

                     )
```

(%o1) inverseLaplace(SI, ϑ) := **block** ([res], M: SI, MC: SI, **for** i **thru** 3 **do for** j **thru** 3 **do** (aC: $M_{i,j}$, b: ilt(aC, s, ϑ), MC$_{i,j}$: b), res: MC)

```
(%i2) rotLaplace(k,theta):=block([res],
                        S:ident(3),
                        I:ident(3),
                      for i:1 thru 3 do
                        (
                        for j:1 thru 3 do
                           (
                             if i=j
                                then S[i][j]:0
                             elseif j>i
                                then (
                               temp:(-1)^(j-i)*k[3-remainder(i+j,3)],
                                      S[i][j]:temp,
                                      S[j][i]:-temp

                                       )
                             )
                         ),
                        res:inverseLaplace(invert(s*I-S),theta)


                       )
```

(%o2) $\mathrm{rotLaplace}(k,\vartheta):=\mathbf{block}\ ([\mathrm{res}], S{:}\,\mathrm{ident}(3), I{:}\,\mathrm{ident}(3),$ $\mathbf{for}\ i\ \mathbf{thru}\ 3\ \mathbf{do}\ \mathbf{for}\ j\ \mathbf{thru}\ 3\ \mathbf{do}\ \mathbf{if}\ i=j\ \mathbf{then}\ (S_i)_j{:}\,0\ \mathbf{elseif}\ j>i\ \mathbf{then}\ (\mathrm{temp}{:}$ $(-1)^{j-i}\,k_{3-\mathrm{remainder}(i+j,3)}, (S_i)_j{:}\,\mathrm{temp}, (S_j)_i{:}\,-\mathrm{temp}), \mathrm{res}{:}\,\mathrm{inverseLaplace}(\mathrm{invert}(s\,I-S),\vartheta))$

```
(%i3) Av(v,theta,d):=block([res],
                        Trot:rotLaplace(v,theta),
                        row:matrix([0,0,0,1]),
                        Atemp:addcol(Trot,d*transpose(v)),
                        A:addrow(Atemp,row),
                        res:A
                        )
```

(%o3) $\mathrm{Av}(v,\vartheta,d) := \mathbf{block}\ ([\mathrm{res}], \mathrm{Trot}{:}\,\mathrm{rotLaplace}(v,\vartheta), \mathrm{row}{:}\,\begin{pmatrix} 0 & 0 & 0 & 1 \end{pmatrix}, \mathrm{Atemp}{:}\,\mathrm{addcol}(\mathrm{Trot},$ $d\,\mathrm{transpose}(v)), A{:}\,\mathrm{addrow}(\mathrm{Atemp},\mathrm{row}), \mathrm{res}{:}\,A)$

```
(%i4) Q(theta,d,alpha,a):=block([res],
                             tempMat:Av([0,0,1],theta,d).Av([1,0,0],alpha,a),
                             Qtrasf:zeromatrix(4,4),
                             for i:1 thru 4 do
                        (
                        for j:1 thru 4 do
                           (
                             Qtrasf[i][j]:trigreduce(tempMat[i][j])
                                       )
                           ),

                             res:Qtrasf
                             )
```

(%o4) $Q(\vartheta,d,\alpha,a):=\mathbf{block}\ ([\mathrm{res}], \mathrm{tempMat}{:}\,\mathrm{Av}([0,0,1],\vartheta,d)\cdot\mathrm{Av}([1,0,0],\alpha,a), \mathrm{Qtrasf}{:}$ $\mathrm{zeromatrix}(4,4), \mathbf{for}\ i\ \mathbf{thru}\ 4\ \mathbf{do}\ \mathbf{for}\ j\ \mathbf{thru}\ 4\ \mathbf{do}\ (\mathrm{Qtrasf}_i)_j{:}\,\mathrm{trigreduce}((\mathrm{tempMat}_i)_j), \mathrm{res}{:}\,\mathrm{Qtrasf})$

```
(%i5) let(sin(q[1]), s[1]);
```

(%o5) $\sin(q_1) \longrightarrow s_1$

```
(%i6) let(sin(q[2]), s[2]);
```

(%o6) $\sin(q_2) \longrightarrow s_2$

```
(%i7) let(cos(q[1]),c[1]);
```

(%o7) $\cos(q_1) \longrightarrow c_1$

```
(%i8) let(cos(q[2]),c[2]);
```

(%o8) $\cos(q_2) \longrightarrow c_2$

```
(%i9) let(sin(q[1]+q[2]), s[12]);
```

(%o9) $\sin(q_2+q_1) \longrightarrow s_{12}$

```
(%i10) let(cos(q[1]+q[2]),c[12]);
```

(%o10) $\cos(q_2+q_1) \longrightarrow c_{12}$

```
(%i11) let(sin(q[2]+q[3]),s[23]);
```

(%o11) $\sin(q_3+q_2) \longrightarrow s_{23}$

```
(%i12) let(cos(q[2]+q[3]),c[23]);
```

(%o12) $\cos(q_3+q_2) \longrightarrow c_{23}$

```
(%i13) let(sin(q[1]+q[3]),s[23]);
```

(%o13) $\sin(q_3+q_1) \longrightarrow s_{23}$

```
(%i14) let(cos(q[1]+q[3]),c[13]);
```

(%o14) $\cos(q_3+q_1) \longrightarrow c_{13}$

```
(%i15) let(sin(q[3]),s[3]);
```

(%o15) $\sin(q_3) \longrightarrow s_3$

```
(%i16) let(cos(q[3]),q[3]);
```

(%o16) $\cos(q_3) \longrightarrow q_3$

```
(%i17)
```

Cinematica diretta:

```
(%i17) Q[antropomorfo](q1,q2,q3,L1,L2,L3):=Q(q1,L1,%pi/2,0).
                                  trigreduce(trigexpand(Q(q2,0,0,L2).
                                  Q(q3,0,0,L3)));
```

(%o17) $Q_{\mathrm{antropomorfo}}(q1, q2, q3, L1, L2, L3) := Q\left(q1, L1, \frac{\pi}{2}, 0\right) \cdot \mathrm{trigreduce}(\mathrm{trigexpand}(Q(q2, 0, 0, L2) \cdot Q(q3, 0, 0, L3)))$

```
(%i18) Qantropomorfo:Q[antropomorfo](q[1],q[2],q[3],L[1],L[2],L[3]);
```

(%o18) $(\cos(q_1)\cos(q_3+q_2), -\cos(q_1)\sin(q_3+q_2), \sin(q_1), \cos(q_1)(L_3\cos(q_3+q_2)+L_2\cos(q_2)); \sin(q_1)\cos(q_3+q_2), -\sin(q_1)\sin(q_3+q_2), -\cos(q_1), \sin(q_1)(L_3\cos(q_3+q_2)+L_2\cos(q_2)); \sin(q_3+q_2), \cos(q_3+q_2), 0, L_3\sin(q_3+q_2)+L_2\sin(q_2)+L_1; 0, 0, 0, 1)$

```
(%i19) letsimp(Qantropomorfo);
```

(%o19) $$\begin{pmatrix} c_1\,c_{23} & -c_1\,s_{23} & s_1 & c_1\,L_3\,c_{23}+c_1\,L_2\,c_2 \\ s_1\,c_{23} & -s_1\,s_{23} & -c_1 & s_1\,L_3\,c_{23}+s_1\,L_2\,c_2 \\ s_{23} & c_{23} & 0 & L_3\,s_{23}+L_2\,s_2+L_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
(%i20)
```

Cinematica inversa robot antropomorfo

Al fine di risolvere il problema di cinematica inversa del robot antropomorfo occorre risolevere il problema di posizione ed orientamente inverso.Inizialmente individuare lo spazio di lavoro, le soluzioni generiche, singolari ed infine le variabili di giunto q_i ed in seguito determinare l'orientamento del robot. Dalla cinematica diretta sappiamo che:

$$Q_{\text{antropomorfo}} = \begin{pmatrix} c_1\, c_{23} & -c_1\, s_{23} & s_1 & c_1\, L_3\, c_{23} + c_1\, L_2\, c_2 \\ s_1\, c_{23} & -s_1\, s_{23} & -c_1 & s_1\, L_3\, c_{23} + s_1\, L_2\, c_2 \\ s_{23} & c_{23} & 0 & L_3\, s_{23} + L_2\, s_2 + L_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Cinematica inversa di posizione

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} c_1\, L_3\, c_{23} + c_1\, L_2\, c_2 \\ s_1\, L_3\, c_{23} + s_1\, L_2\, c_2 \\ L_3\, s_{23} + L_2\, s_2 + L_1 \end{pmatrix}$$

$$\begin{cases} x = c_1\, L_3\, c_{23} + c_1\, L_2\, c_2 \\ y = s_1\, L_3\, c_{23} + s_1\, L_2\, c_2 \end{cases} \rightarrow \begin{cases} x = c_1\,(L_3 c_{23} + L_2 c_2) \\ y = s_1 (L_3 c_{23} + L_2 c_2) \end{cases} \rightarrow \begin{cases} x^2 = c_1^2 (L_3 c_{23} + L_2 c_2)^2 \\ y^2 = s_1^2 (L_3 c_{23} + L_2 c_2)^2 \end{cases}$$

$$x^2 + y^2 = (L_3 c_{23} + L_2 c_2)^2 \rightarrow L_3 c_{23} + L_2 c_2 = \pm\sqrt{x^2 + y^2}$$

$$\begin{cases} L_3 c_{23} + L_2 c_2 = \pm\sqrt{x^2 + y^2} \\ L_3 s_{23} + L_2 s_2 = z - L_1 \end{cases}$$

è possibile riscrivere le ultime due equazioni nel seguente modo:

$$\begin{pmatrix} c_{23} & -s_{23} \\ s_{23} & c_{23} \end{pmatrix} \begin{pmatrix} L_3 \\ 0 \end{pmatrix} + \begin{pmatrix} c_2 & -s_2 \\ s_2 & c_2 \end{pmatrix} \begin{pmatrix} L_2 \\ 0 \end{pmatrix} = \begin{pmatrix} \pm\sqrt{x^2 + y^2} \\ z - L_1 \end{pmatrix}$$

In particolare $R_{23} = \begin{pmatrix} c_{23} & -s_{23} \\ s_{23} & c_{23} \end{pmatrix}$ è una matrice di rotazione nel piano quindi equivale a $R_2 R_3$:

$$R_2 \left(R_3 \begin{pmatrix} L_3 \\ 0 \end{pmatrix} + \begin{pmatrix} L_2 \\ 0 \end{pmatrix} \right) = \begin{pmatrix} \pm\sqrt{x^2 + y^2} \\ z - L_1 \end{pmatrix}$$

Poiché R_2 è una matrice di rotazione i termini $R_3 \begin{pmatrix} L_3 \\ 0 \end{pmatrix} + \begin{pmatrix} L_2 \\ 0 \end{pmatrix}, \begin{pmatrix} \pm\sqrt{x^2 + y^2} \\ z - L_1 \end{pmatrix}$ devono avere la stessa norma:

$$((\, L_3 \quad 0\,)\, R_3^T + (\, L_2 \quad 0\,)) \left(R_3 \begin{pmatrix} L_3 \\ 0 \end{pmatrix} + \begin{pmatrix} L_2 \\ 0 \end{pmatrix} \right) =$$

$$= (\, L_3 \quad 0\,)\, R_3^T R_3 \begin{pmatrix} L_3 \\ 0 \end{pmatrix} + (\, L_2 \quad 0\,) \begin{pmatrix} L_2 \\ 0 \end{pmatrix} + 2 (\, L_2 \quad 0\,) R_3 \begin{pmatrix} L_3 \\ 0 \end{pmatrix} =$$

$$= L_3^2 + L_2^2 + 2\, L_2 L_3 c_3$$

Quindi:

$$x^2 + y^2 + (z - L_1)^2 = L_3^2 + L_2^2 + 2\, L_2 L_3 c_3$$

$$c_3 = \frac{x^2 + y^2 + (z - L_1)^2 - L_3^2 - L_2^2}{2\, L_2 L_3}$$

Imponendo la condizione che $-1 \leqslant c_3 \leqslant 1$:

$$-1 \leqslant \frac{x^2 + y^2 + (z - L_1)^2 - L_3^2 - L_2^2}{2\, L_2 L_3} \leqslant 1$$

Otteniamo l'espressione dello spazio operativo:

$$(L_3 - L_2)^2 \leqslant x^2 + y^2 + (z - L_1)^2 \leqslant (L_3 + L_2)^2$$

che rappresenta una sfera cava di centro $\begin{pmatrix} 0 \\ 0 \\ L_1 \end{pmatrix}$ e raggio $|L_2 - L_3| \leqslant r \leqslant L_2 + L_3$.

Per determinare la variabile di giunto q_3:

$$c_3 = \frac{x^2 + y^2 + (z - L_1)^2 - L_3^2 - L_2^2}{2\, L_2 L_3}$$

$$s_3 = \pm\sqrt{1 - c_3}$$

$$q_3 = \mathrm{atan2}(\pm s_3, c_3)$$

A questo punto il termine $R_3\begin{pmatrix} L_3 \\ 0 \end{pmatrix} + \begin{pmatrix} L_2 \\ 0 \end{pmatrix}$ è una quantità nota detta A_1, A_2. In aggiunta, i termni $\begin{pmatrix} \pm\sqrt{x^2 + y^2} \\ z - L_1 \end{pmatrix}$ li definiamo come B_1, B_2 ottendendo:

$$\begin{pmatrix} c_2 & -s_2 \\ s_2 & c_2 \end{pmatrix} \begin{pmatrix} A_1 \\ A_2 \end{pmatrix} = \begin{pmatrix} \pm B_1 \\ B_2 \end{pmatrix}$$

$$\begin{pmatrix} A_1 & -A_2 \\ A_2 & A_1 \end{pmatrix} \begin{pmatrix} c_2 \\ s_2 \end{pmatrix} = \begin{pmatrix} \pm B_1 \\ B_2 \end{pmatrix}$$

$$\det\left(\begin{pmatrix} A_1 & -A_2 \\ A_2 & A_1 \end{pmatrix}\right) = A_1^2 + A_2^2 \neq 0 \rightarrow \text{è possibile effettuare} \quad l'\text{inversa}$$

$$\begin{pmatrix} c_2 \\ s_2 \end{pmatrix} = \frac{1}{A_1^2 + A_2^2} \begin{pmatrix} A_1 & A_2 \\ -A_2 & A_1 \end{pmatrix} \begin{pmatrix} \pm B_1 \\ B_2 \end{pmatrix}$$

$$\begin{pmatrix} c_2 \\ s_2 \end{pmatrix} = \begin{pmatrix} \frac{\pm A_1 B_1 + A_2 B_2}{A_1^2 + A_2^2} \\ \frac{A_1 B_2 \mp A_2 B_1}{A_1^2 + A_2^2} \end{pmatrix}$$

$$q_2 = \mathrm{atan2}(s_2, c_2) = \mathrm{atan2}\left(\frac{A_1 B_2 \mp A_2 B_1}{A_1^2 + A_2^2}, \frac{\pm A_1 B_1 + A_2 B_2}{A_1^2 + A_2^2}\right) =$$

$$= \mathrm{atan2}(A_1 B_2 \mp A_2 B_1, \pm A_1 B_1 + A_2 B_2)$$

Per la variabile di giunto q_1, si ricorda che:

$$\begin{cases} x = c_1\,(L_3\, c_{23} + L_2\, c_2) \\ y = s_1\,(L_3\, c_{23} + L_2\, c_2) \end{cases}$$

Poiché $(L_3\, c_{23} + L_2\, c_2)$ è una quantità nota:

$$\begin{cases} c_1 = \frac{x}{(L_3\, c_{23} + L_2\, c_2)} \\ s_1 = \frac{y}{(L_3\, c_{23} + L_2\, c_2)} \end{cases}$$

In conclusione:

$$q_1 = \mathrm{atan2}(s_1, c_1) = \mathrm{atan2}\left(\frac{y}{(L_3\, c_{23} + L_2\, c_2)}, \frac{x}{(L_3\, c_{23} + L_2\, c_2)}\right)$$

Orientamento inverso

La risoluzione del problema di orientamento inverso si basa sulla scelta di una terna di Eulero o di una terna nautica in condizioni non singolari se possibile.

$$R_{\text{antorpomorfo}} = \begin{pmatrix} c_1\, c_{23} & -c_1\, s_{23} & s_1 \\ s_1\, c_{23} & -s_1\, s_{23} & -c_1 \\ s_{23} & c_{23} & 0 \end{pmatrix}$$

$$R_{\text{zyx}} = \begin{pmatrix} c_y c_z & \cdots\cdots\cdots & \\ c_y s_z & \cdots\cdots\cdots & \\ -s_y & s_x c_y & c_x c_y \end{pmatrix}$$

$$-s_y = s_{23} \neq \pm 1 \rightarrow q_2 + q_3 = \begin{cases} \pm\frac{\pi}{2} \rightarrow \text{soluzione singolare} \\ \text{altrimenti} \rightarrow \text{soluzine regolare} \end{cases}$$

$$\begin{cases} s_y = -s_{23} \\ c_y = \pm\sqrt{1 - s_y} = \pm c_{23} \end{cases} \rightarrow \phi_y = \text{atan2}(-s_{23}, c_{23}) = \begin{cases} -(q_2 + q_3) \\ \pi + (q_2 + q_3) \end{cases}$$

$$\begin{cases} s_x c_y = c_{23} \\ c_x c_y = 0 \end{cases} \rightarrow \begin{cases} \pm s_x c_{23} = c_{23} \\ \pm c_x c_{23} = 0 \end{cases} \rightarrow \begin{cases} s_x = \pm 1 \\ c_x = 0 \end{cases} \rightarrow \phi_x = \text{atan2}(\pm 1, 0) = \begin{cases} \frac{\pi}{2} \\ -\frac{\pi}{2} \end{cases}$$

$$\begin{cases} c_y c_z = c_1 c_{23} \\ c_y s_z = s_1 c_{23} \end{cases} \rightarrow \begin{cases} c_z = \pm c_1 \\ s_z = \pm s_1 \end{cases} \rightarrow \phi_z = \text{atan2}(\pm s_1, \pm c_1) = \begin{cases} q_1 \\ q_1 + \pi \end{cases}$$

Riassumendo:

$$\begin{pmatrix} \frac{\pi}{2} \\ -(q_2 + q_3) \\ q_1 \end{pmatrix}, \begin{pmatrix} -\frac{\pi}{2} \\ \pi + q_2 + q_3 \\ q_1 + \pi \end{pmatrix}$$

In alternativa, tramite la scelta di una terna di Eulero:

$$R_{\text{zyz}} = \begin{pmatrix} \cdots & \cdots & \cos(\alpha)\sin(\beta) \\ \cdots & \cdots & \sin(\alpha)\sin(\beta) \\ -\sin(\beta)\cos(\gamma) & \sin(\beta)\sin(\gamma) & \cos(\beta) \end{pmatrix}$$

$$\cos(\beta) = 0 \rightarrow \sin(\beta) = \pm 1 \rightarrow \beta = \text{atan2}(\pm 1, 0) = \begin{cases} \frac{\pi}{2} \\ -\frac{\pi}{2} \end{cases}$$

$$\begin{cases} \sin(\beta)\sin(\gamma) = c_{23} \\ -\sin(\beta)\cos(\gamma) = s_{23} \end{cases} \rightarrow \begin{cases} \sin(\gamma) = \pm c_{23} \\ \cos(\gamma) = \mp s_{23} \end{cases} \rightarrow \gamma = \text{atan2}(\pm c_{23}, \mp s_{23})$$

$$\gamma = \text{atan2}(\pm c_{23}, \mp s_{23}) = \begin{cases} q_2 + q_3 + \frac{\pi}{2} \\ q_2 + q_3 - \frac{\pi}{2} \end{cases}$$

$$\begin{cases} \cos(\alpha)\sin(\beta) = s_1 \\ \sin(\alpha)\sin(\beta) = -c_1 \end{cases} \rightarrow \begin{cases} \cos(\alpha) = \pm s_1 \\ \sin(\alpha) \mp c_1 \end{cases} \rightarrow \alpha = \text{atan2}(\mp c_1, \pm s_1) = \begin{cases} q_1 - \frac{\pi}{2} \\ q_1 + \frac{\pi}{2} \end{cases}$$

Riassumendo:

$$\begin{pmatrix} q_1 - \frac{\pi}{2} \\ \frac{\pi}{2} \\ q_2 + q_3 + \frac{\pi}{2} \end{pmatrix}, \begin{pmatrix} q_1 + \frac{\pi}{2} \\ -\frac{\pi}{2} \\ q_2 + q_3 - \frac{\pi}{2} \end{pmatrix}$$

```
(%i17) isRotation(M):=block([MC,res],
                           I:ident(3),
                           MC:ident(3),
                           for i:1 thru 3 do
                           (
                           for j:1 thru 3 do
                              (
                                 MC[i][j]:M[i][j]
                                 )
                              ),
                           MMT:trigsimp(expand(MC.transpose(MC))),
                           detM:trigsimp(expand(determinant(MC))),

                           if MMT=I  and detM=1
                              then(

                                   return(res:1)
                                   )

                           else(

                                 res: "R is not rotation matrix"
                                 )
                           )
```

(%o17) isRotation(M) := **block** ([MC, res], I: ident(3), MC: ident(3), **for** i **thru** 3 **do for** j **thru** 3 **do** $(\mathrm{MC}_i)_j$: $(M_i)_j$, MMT: trigsimp(expand(MC · transpose(MC))), detM: trigsimp(expand(determinant(MC))), **if** MMT $= I \wedge$ detM $= 1$ **then** return(res: 1) **else** res: R is not rotation matrix)

```
(%i18) skewMatrix(x):=block([res],
                           S:ident(3),
                           for i:1 thru 3 do
                           (
                           for j:1 thru 3 do
                              (
                                if i=j
                                   then S[i][j]:0
                                elseif j>i
                                   then (
                                  temp:(-1)^(j-i)*x[3-remainder(i+j,3)],
                                         S[i][j]:temp,
                                         S[j][i]:-temp

                                          )
                                )
                            ),
                            res:S
                            )
```

(%o18) skewMatrix(x) := **block** ([res], S: ident(3), **for** i **thru** 3 **do for** j **thru** 3 **do if** $i =$ j **then** $(S_i)_j$: 0 **elseif** $j > i$ **then** (temp: $(-1)^{j-i}\, x_{3-\mathrm{remainder}(i+j,3)}$, $(S_i)_j$: temp, $(S_j)_i$: $-$temp), res: S)

```
(%i19) rodriguez(y,arg):=block([res],
                         I:ident(3),
                         S:skewMatrix(y),
                         res:I+S.S*(1-cos(arg))+S*sin(arg)
                         )
```

(%o19) $\mathrm{rodriguez}(y,\mathrm{arg}) := \mathbf{block}\ ([\mathrm{res}], I{:}\ \mathrm{ident}(3), S{:}\ \mathrm{skewMatrix}(y), \mathrm{res}{:}\ I + S\cdot S\,(1 - \cos(\mathrm{arg})) + S\sin(\mathrm{arg}))$

```
(%i20) calculate(x,y,z,L1,L2,L3):=block(
       [c1,s1,c2,s2,c3,s3,res,A,B,q1,q2,q3],
       condition: x^(2)+y^(2)+(z-L1)^2,
       if(condition>(L2+L3)^2 or condition<(L2-L3)^2 or (x=0 and y=0)) then
                    (error("La soluzione è singolare")),
             c3:trigsimp(ratsimp((condition-L3^(2)-L2^(2))/(2*L2*L3))),
             s3:trigsimp(ratsimp(sqrt(1-c3^2))),
             q3:atan2(s3,c3),
             B:[ratsimp(sqrt(x^(2)+y^(2))),ratsimp(z-L1)],
             A:[trigsimp(ratsimp(cos(q3)*L3+L2)),
                abs(trigsimp(ratsimp(sin(q3)*L3)))],
             c2:[A[1]*B[1]+A[2]*B[2],
                -A[1]*B[1]+A[2]*B[2],
                A[1]*B[1]-A[2]*B[2],
                -A[1]*B[1]-A[2]*B[2]],
             s2:[-A[2]*B[1]+A[1]*B[2],
                 A[2]*B[1]+A[1]*B[2],
                 A[2]*B[1]+A[1]*B[2],
                 -A[2]*B[1]+A[1]*B[2]],
             q2:[atan2(ratsimp(s2[1]),ratsimp(c2[1])),
                 atan2(ratsimp(s2[2]),ratsimp(c2[2])),
                 atan2(ratsimp(s2[3]),ratsimp(c2[3])),
                 atan2(ratsimp(s2[4]),ratsimp(c2[4]))],
              den:[L3*cos(q3+q2[1])+L2*cos(q2[1]),
                L3*cos(q3+q2[2])+L2*cos(q2[2]),
                L3*cos(-q3+q2[3])+L2*cos(q2[3]),
                L3*cos(-q3+q2[4])+L2*cos(q2[4])],
              c1:[ratsimp(x/den[1]),
                  ratsimp(x/den[2]),
                  ratsimp(x/den[3]),
                  ratsimp(x/den[4])],
              s1:[ratsimp(y/den[1]),
                  ratsimp(y/den[2]),
                  ratsimp(y/den[3]),
                  ratsimp(y/den[4])],
              q1:[atan2(s1[1],c1[1]),
                 atan2(s1[2],c1[2]),
                 atan2(s1[3],c1[3]),
                 atan2(s1[4],c1[4])],

              res:[[q1[1],q2[1],q3],
                   [q1[2],q2[2],q3],
                   [q1[3],q2[3],-q3],
                   [q1[4],q2[4],-q3]]
       )
```

(%o20) $\mathrm{calculate}(x,y,z,L1,L2,L3) := \mathbf{block}\ \Big([c1, s1, c2, s2, c3, s3, \mathrm{res}, A, B, q1, q2, q3],$

condition: $x^2+y^2+(z-L1)^2$, **if** condition $>(L2+L3)^2 \vee$ condition $<(L2-L3)^2 \vee x=0 \wedge y=0$ **then** error(La soluzione è singolare), $c3$: trigsimp$\left(\text{ratsimp}\left(\frac{\text{condition}-L3^2-L2^2}{2\,L2\,L3}\right)\right)$, $s3$: trigsimp$\left(\text{ratsimp}\left(\sqrt{1-c3^2}\right)\right)$, $q3$: atan2$(s3, c3)$, B: $\left[\text{ratsimp}\left(\sqrt{x^2+y^2}\right), \text{ratsimp}(z-L1)\right]$, A: [trigsimp(ratsimp($\cos(q3)\,L3+L2$)), |trigsimp(ratsimp($\sin(q3)\,L3$))|], $c2$: $[A_1\,B_1+A_2\,B_2, (-A_1)\,B_1+A_2\,B_2, A_1\,B_1-A_2\,B_2, (-A_1)\,B_1-A_2\,B_2]$, $s2$: $[(-A_2)\,B_1+A_1\,B_2, A_2\,B_1+A_1\,B_2, A_2\,B_1+A_1\,B_2, (-A_2)\,B_1+A_1\,B_2]$, $q2$: [atan2(ratsimp($s2_1$), ratsimp($c2_1$)), atan2(ratsimp($s2_2$), ratsimp($c2_2$)), atan2(ratsimp($s2_3$), ratsimp($c2_3$)), atan2(ratsimp($s2_4$), ratsimp($c2_4$))], den: $[L3\cos(q3+q2_1)+L2\cos(q2_1), L3\cos(q3+q2_2)+L2\cos(q2_2), L3\cos(-q3+q2_3)+L2\cos(q2_3), L3\cos(-q3+q2_4)+L2\cos(q2_4)]$, $c1$: $\left[\text{ratsimp}\left(\frac{x}{\text{den}_1}\right), \text{ratsimp}\left(\frac{x}{\text{den}_2}\right), \text{ratsimp}\left(\frac{x}{\text{den}_3}\right), \text{ratsimp}\left(\frac{x}{\text{den}_4}\right)\right]$, $s1$: $\left[\text{ratsimp}\left(\frac{y}{\text{den}_1}\right), \text{ratsimp}\left(\frac{y}{\text{den}_2}\right), \text{ratsimp}\left(\frac{y}{\text{den}_3}\right), \text{ratsimp}\left(\frac{y}{\text{den}_4}\right)\right]$, $q1$: [atan2($s1_1, c1_1$), atan2($s1_2, c1_2$), atan2($s1_3, c1_3$), atan2($s1_4, c1_4$)], res: $[[q1_1, q2_1, q3], [q1_2, q2_2, q3], [q1_3, q2_3, -q3], [q1_4, q2_4, -q3]]$)

```
(%i21) orientation(Qdiretta):=block([sx,cx,sy,cy,phiy1,phiy2,phiz1,phiz2,phix1,
       phix2,sz,sxfirst,second,res],

                               rotation:isRotation(Qdiretta),
                               if rotation=1 then(
                               sy:Qdiretta[3][1],

                               if sy=1 or sy=-1 then print("soluzione
       singolare")
                               else(
                               cy:sqrt(1-sy^2),
                               phiy1:atan2(-sy,cy),
                               phiy2:atan2(-sy,-cy),

                               sx:Qdiretta[3][2]/cy,
                               cx:Qdiretta[3][3]/cy,
                               phix1:atan2(sx,cx),
                               phix2:atan2(-sx,cx),
                               cz:Qdiretta[1][1]/cy,
                               sz:Qdiretta[2][1]/cy,
                               phiz1:atan2(+sz,cz),
                               phiz2:atan2(-sz,cz),
                               first:[phix1,phiy1,phiz1],
                               second:[phix2,phiy2,phiz2],

                               res:[first,second])
                               )


       );
```

(%o21) orientation(Qdiretta) := **block** ([sx, cx, sy, cy, phiy1, phiy2, phiz1, phiz2, phix1, phix2, sz, sxfirst, second, res], rotation: isRotation(Qdiretta), **if** rotation $=1$ **then** (sy: $(\text{Qdiretta}_3)_1$, **if** sy $=1 \vee$ sy $=-1$ **then** print(soluzione singolare) **else** (cy: $\sqrt{1-\text{sy}^2}$, phiy1: atan2($-$sy, cy), phiy2:

atan2(−sy, −cy), sx: $\frac{(\mathrm{Qdiretta}_3)_2}{\mathrm{cy}}$, cx: $\frac{(\mathrm{Qdiretta}_3)_3}{\mathrm{cy}}$, phix1: atan2(sx, cx), phix2: atan2(−sx, cx), cz: $\frac{(\mathrm{Qdiretta}_1)_1}{\mathrm{cy}}$, sz: $\frac{(\mathrm{Qdiretta}_2)_1}{\mathrm{cy}}$, phiz1: atan2(+sz, cz), phiz2: atan2(−sz, cz), first: [phix1, phiy1, phiz1], second: [phix2, phiy2, phiz2], res: [first, second])))

```
(%i22) invAntropomorfo(x,y,z,L1,L2,L3,alpha,beta,gamma):=block(
        [pos,R,orient],
         pos:calculate(x,y,z,L1,L2,L3),
         R:rodriguez([0,0,1],alpha).
           rodriguez([0,1,0],beta).
           rodriguez([1,0,0],gamma),
         print("Rzyx=",R),
         orient:orientation(R),
         print("Position=",pos),
         print("Orientation=",orient)
         )
```

(%o22) $\mathrm{invAntropomorfo}(x, y, z, L1, L2, L3, \alpha, \beta, \gamma) := \mathbf{block}\ ([\mathrm{pos}, R, \mathrm{orient}], \mathrm{pos}: \mathrm{calculate}(x, y, z, L1, L2, L3), R: \mathrm{rodriguez}([0,0,1], \alpha) \cdot \mathrm{rodriguez}([0,1,0], \beta) \cdot \mathrm{rodriguez}([1,0,0], \gamma), \mathrm{print}(\mathrm{Rzyx}=, R), \mathrm{orient}: \mathrm{orientation}(R), \mathrm{print}(\mathrm{Position}=, \mathrm{pos}), \mathrm{print}(\mathrm{Orientation}=, \mathrm{orient}))$

```
(%i23) invAntropomorfo(1,1,1,1,1,2,%pi/2,3*%pi/4,%pi/4);
```

$$\mathrm{Rzyx}= \begin{pmatrix} 0 & -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{2} & \frac{1}{2} \\ -\frac{1}{\sqrt{2}} & -\frac{1}{2} & -\frac{1}{2} \end{pmatrix}$$

$$\mathrm{Position}= \left[\left[\frac{\pi}{4}, \arctan\left(\sqrt{7}\right) - \pi, \pi - \arctan\left(\frac{\sqrt{7}}{3}\right)\right], \left[-\frac{3\,\pi}{4}, \arctan\left(\sqrt{7}\right), \pi - \arctan\left(\frac{\sqrt{7}}{3}\right)\right], \left[\frac{\pi}{4}, \pi - \arctan\left(\sqrt{7}\right), \arctan\left(\frac{\sqrt{7}}{3}\right) - \pi\right], \left[-\frac{3\,\pi}{4}, -\arctan\left(\sqrt{7}\right), \arctan\left(\frac{\sqrt{7}}{3}\right) - \pi\right]\right]$$

$$\mathrm{Orientation}= \left[\left[-\frac{3\,\pi}{4}, \frac{\pi}{4}, -\frac{\pi}{2}\right], \left[\frac{3\,\pi}{4}, \frac{3\,\pi}{4}, \frac{\pi}{2}\right]\right]$$

(%o23) $\left[\left[-\frac{3\,\pi}{4}, \frac{\pi}{4}, -\frac{\pi}{2}\right], \left[\frac{3\,\pi}{4}, \frac{3\,\pi}{4}, \frac{\pi}{2}\right]\right]$

Singolarità

```
Maxima 5.44.0 http://maxima.sourceforge.net
using Lisp SBCL 2.0.0
Distributed under the GNU Public License. See the file COPYING.
Dedicated to the memory of William Schelter.
The function bug_report() provides bug reporting information.
(%i1) x:cos (q[1])*(L[3]*cos (q[3]+q[2])+L[2]*cos (q[2]));
```

(%o1) $\cos(q_1)\,(L_3 \cos(q_3 + q_2) + L_2 \cos(q_2))$

```
(%i2) y:sin (q[1])*(L[3]*cos (q[3]+q[2])+L[2]*cos (q[2]));
```

(%o2) $\sin(q_1)\,(L_3 \cos(q_3 + q_2) + L_2 \cos(q_2))$

```
(%i3) z:L[3]*sin (q[3]+q[2])+L[2]*sin (q[2])+L[1];
```

(%o3) $L_3 \sin(q_3 + q_2) + L_2 \sin(q_2) + L_1$

(%i4) J:J:matrix([diff(x,q[1]),diff(x,q[2]),diff(x,q[3])],
[diff(y,q[1]),diff(y,q[2]),diff(y,q[3])],
[diff(z,q[1]),diff(z,q[2]),diff(z,q[3])]);

(%o4) $(-\sin(q_1)(L_3\cos(q_3+q_2)+L_2\cos(q_2)), \cos(q_1)(-L_3\sin(q_3+q_2)-L_2\sin(q_2)), -L_3\cos(q_1)\sin(q_3+q_2); \cos(q_1)(L_3\cos(q_3+q_2)+L_2\cos(q_2)), \sin(q_1)(-L_3\sin(q_3+q_2)-L_2\sin(q_2)), -L_3\sin(q_1)\sin(q_3+q_2); 0, L_3\cos(q_3+q_2)+L_2\cos(q_2), L_3\cos(q_3+q_2))$

(%i5) dJ:factor(trigsimp(determinant(J)));

(%o5) $-L_2 L_3 (L_3\cos(q_3+q_2)+L_2\cos(q_2))(\cos(q_2)\sin(q_3+q_2)-\sin(q_2)\cos(q_3+q_2))$

$$L_3\sin(q_2)\sin(q_3)-L_3\cos(q_2)\cos(q_3)=L_2\cos(q_2)$$

$$\begin{cases} b=L_3\sin(q_2) \\ a=L_3\cos(q_2) \end{cases}$$

$$b\sin(q_3)-a\cos(q_3)=\frac{L_2}{L_3}a$$

$$b\sin(q_3)=a\left(\cos(q_3)+\frac{L_2}{L_3}\right)$$

$$\mathbf{q_3\neq 0}\rightarrow b=a\left(\frac{L_3\cos(q_3)+L_2}{L_3\sin(q_3)}\right)$$

$$a^2+a^2\left(\frac{L_3\cos(q_3)+L_2}{L_3\sin(q_3)}\right)^2=L_3^2$$

$$a^2\left(\frac{L_3^2\sin(q_3^2)+(L_3\cos(q_3)+L_2)^2}{L_3^2\sin(q_3^2)}\right)=L_3^2$$

$$a^2\left(\frac{L_3^2+L_2^2+2L_3L_2\cos(q_3)}{L_3^2\sin(q_3)^2}\right)=L_3^2$$

$$a^2=\frac{L_3^4\sin(q_3^2)}{L_3^2+L_2^2+2L_3L_2\cos(q_3)}\rightarrow a=\pm\frac{L_3^2\sin(q_3)}{\sqrt{L_3^2+L_2^2+2L_3L_2\cos(q_3)}}$$

$$b=\frac{\pm L_3^2\sin(q_3)}{\sqrt{L_3^2+L_2^2+2L_3L_2\cos(q_3)}}\left(\frac{L_3\cos(q_3)+L_2}{L_3\sin(q_3)}\right)=\frac{\pm L_3(L_3\cos(q_3)+L_2)}{\sqrt{2L_3^2+L_2^2+2L_3L_2\cos(q_3)}}$$

$$\begin{cases} \pm\frac{L_3(L_3\cos(q_3)+L_2)}{\sqrt{2\,L_2\,L_3^3\cos(q_3)+L_3^4+L_2^2\,L_3^2}}=\sin(q_2) \\ \pm\frac{L_3^2\sin(q_3)}{\sqrt{2\,L_2\,L_3^3\cos(q_3)+L_3^4+L_2^2\,L_3^2}}=\cos(q_2) \end{cases}$$

$$q_2=\mathrm{atan2}(\pm(L_3\cos(q_3)+L_2),\pm\sin(q_3))$$

$$\mathbf{q_3=0}\rightarrow a\left(1+\frac{L_2}{L_3}\right)=0\rightarrow a=0$$

$$\begin{cases} L_3\sin(q_2)=b \\ L_3\cos(q_2)=0 \end{cases}\rightarrow\begin{cases} \cos(q_2)=0 \\ \sin(q_2)=\frac{b}{L_3} \end{cases}\rightarrow q_2=\frac{\pi}{2}, b=L_3$$

Riassumendo:

$$q_3\neq 0\wedge q_2=\mathrm{atan2}(\pm L_3(L_3\cos(q_3)+L_2),\pm L_3^2\sin(q_3))\vee q_3=0\wedge q_2=\frac{\pi}{2}$$

Verifica che le soluzioni trovate annullino il determinante:

```
(%i6) subst([q[3]=0,q[2]=%pi/2],dJ);
```

(%o6) 0

```
(%i7) trigsimp(trigexpand(subst([q[2]=atan2((L[3]*(L[3]*cos(q[3])+L[2])),
      L[3]^2*sin(q[3]))],dJ)));
```

(%o7) 0

Caso $q_3 = 0 \wedge q_2 = \frac{\pi}{2}$:

```
(%i8) Jq32:subst([q[3]=0,q[2]=%pi/2],J);
```

(%o8) $$\begin{pmatrix} 0 & (-L_3 - L_2)\cos(q_1) & -L_3\cos(q_1) \\ 0 & (-L_3 - L_2)\sin(q_1) & -L_3\sin(q_1) \\ 0 & 0 & 0 \end{pmatrix}$$

```
(%i9) nullspace(Jq32);
```

Proviso: notequal$(-L_3\cos(q_1), 0)$

(%o9) $$\operatorname{span}\left(\begin{pmatrix} 0 \\ -L_3\cos(q_1) \\ (L_3 + L_2)\cos(q_1) \end{pmatrix}, \begin{pmatrix} -L_3\cos(q_1) \\ 0 \\ 0 \end{pmatrix}\right)$$

Se $q_1 \neq 0$, le singolarità di velocità si hanno per $v \in \mathrm{Im}\left\{\begin{pmatrix} 0 \\ -L_3\cos(q_1) \\ (L_3 + L_2)\cos(q_1) \end{pmatrix}, \begin{pmatrix} -L_3\cos(q_1) \\ 0 \\ 0 \end{pmatrix}\right\}$.

Se $q_1 = 0$:

```
(%i11) Jq321:subst(q[1]=0,Jq32);
```

(%o11) $$\begin{pmatrix} 0 & -L_3 - L_2 & -L_3 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

```
(%i12) nullspace(Jq321);
```

Proviso: notequal$(-L_3, 0)$

(%o12) $$\operatorname{span}\left(\begin{pmatrix} 0 \\ -L_3 \\ L_3 + L_2 \end{pmatrix}, \begin{pmatrix} -L_3 \\ 0 \\ 0 \end{pmatrix}\right)$$

le singolarità di velocità si hanno per $v \in \mathrm{Im}\left\{\begin{pmatrix} 0 \\ -L_3 \\ L_3 + L_2 \end{pmatrix}, \begin{pmatrix} -L_3 \\ 0 \\ 0 \end{pmatrix}\right\}$

Caso $q_3 \neq 0 \wedge q_2 = \mathrm{atan2}(\pm L_3(L_3\cos(q_3) + L_2), \pm L_3^2\sin(q_3))$:

```
(%i14) Jq32:trigsimp(trigexpand(subst([q[2]=atan2((L[3]*(L[3]*cos(q[3])+L[2])),
       L[3]^2*sin(q[3]))],J)));
```

(%o14) $$\left(0, -\frac{2\,L_2\,L_3^2\cos(q_1)\cos(q_3) + (L_3^3 + L_2^2\,L_3)\cos(q_1)}{\sqrt{2\,L_2\,L_3^3\cos(q_3) + L_3^4 + L_2^2\,L_3^2}}, -\frac{\sqrt{2\,L_2\,L_3^3\cos(q_3) + L_3^4 + L_2^2\,L_3^2}\,(L_2\cos(q_1)\cos(q_3) + L_3\cos(q_1))}{2\,L_2\,L_3\cos(q_3) + L_3^2 + L_2^2}; 0, -\frac{2\,L_2\,L_3^2\sin(q_1)\cos(q_3) + (L_3^3 + L_2^2\,L_3)\sin(q_1)}{\sqrt{2\,L_2\,L_3^3\cos(q_3) + L_3^4 + L_2^2\,L_3^2}}, -\frac{\sqrt{2\,L_2\,L_3^3\cos(q_3) + L_3^4 + L_2^2\,L_3^2}\,(L_2\sin(q_1)\cos(q_3) + L_3\sin(q_1))}{2\,L_2\,L_3\cos(q_3) + L_3^2 + L_2^2}; 0, 0, -\frac{L_2\sqrt{2\,L_2\,L_3^3\cos(q_3) + L_3^4 + L_2^2\,L_3^2}\sin(q_3)}{2\,L_2\,L_3\cos(q_3) + L_3^2 + L_2^2}\right)$$

```
(%i15) nullspace(Jq32);
```

Proviso: notequal$\left(-\cos(q_1)\sqrt{2\,L_2\,L_3^3\cos(q_3) + L_3^4 + L_2^2\,L_3^2}, 0\right) \wedge$ notequal$((2\,L_2^2\,L_3^3\cos(q_1)\cos(q_3) + (L_2\,L_3^4 + L_2^3\,L_3^2)\cos(q_1))\sin(q_3), 0)$

(%o15) $\mathrm{span}\left(\begin{pmatrix} (2\,L_2^2\,L_3^3\cos(q_1)\cos(q_3)+(L_2\,L_3^4+L_2^3\,L_3^2)\cos(q_1))\sin(q_3)\\ 0\\ 0\end{pmatrix}\right)$

Se $q_1 \neq 0$, si hanno singolarità di velocita per $v \in \mathrm{Im}\left\{\begin{pmatrix} (2\,L_2^2\,L_3^3\cos(q_1)\cos(q_3)+(L_2\,L_3^4+L_2^3\,L_3^2)\cos(q_1))\sin(q_3)\\ 0\\ 0\end{pmatrix}\right\}$.

Se $q_1 = 0$:

(%i16) `Jq321:subst(q[1]=0,Jq32);`

$$\text{(\%o16)}\quad \begin{pmatrix} 0 & -\frac{2\,L_2\,L_3^2\cos(q_3)+L_3^3+L_2^2\,L_3}{\sqrt{2\,L_2\,L_3^3\cos(q_3)+L_3^4+L_2^2\,L_3^2}} & -\frac{(L_2\cos(q_3)+L_3)\sqrt{2\,L_2\,L_3^3\cos(q_3)+L_3^4+L_2^2\,L_3^2}}{2\,L_2\,L_3\cos(q_3)+L_3^2+L_2^2}\\ 0 & 0 & 0\\ 0 & 0 & -\frac{L_2\sqrt{2\,L_2\,L_3^3\cos(q_3)+L_3^4+L_2^2\,L_3^2}\sin(q_3)}{2\,L_2\,L_3\cos(q_3)+L_3^2+L_2^2}\end{pmatrix}$$

(%i17) `nullspace(Jq321);`

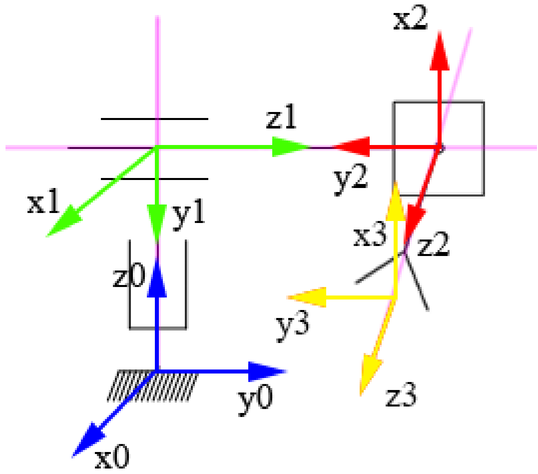
Proviso: $\mathrm{notequal}\left(-\sqrt{2\,L_2\,L_3^3\cos(q_3)+L_3^4+L_2^2\,L_3^2},0\right)\wedge \mathrm{notequal}((2\,L_2^2\,L_3^3\cos(q_3)+L_2\,L_3^4+L_2^3\,L_3^2)\sin(q_3),0)$

(%o17) $\mathrm{span}\left(\begin{pmatrix} (2\,L_2^2\,L_3^3\cos(q_3)+L_2\,L_3^4+L_2^3\,L_3^2)\sin(q_3)\\ 0\\ 0\end{pmatrix}\right)$

(%i18)

SCinematica diretta Robot Cartesiano

***N.B.:* le grandezze diverse da quelle di giunto q_i sono L_i, D_i. Esse sono rispettivamente la distanza tra i sistemi di riferimento R_i e R_{i+1} nelle operazioni della matrice avvitamento $A_z(\theta, d)$ e $A_x(\alpha, a)$.**



	ϑ	d	α	a
1	0	q_1	$-\frac{\pi}{2}$	0
2	$-\frac{\pi}{2}$	q_2	$-\frac{\pi}{2}$	0
3	0	q_3	0	0

Tabella 1.

Funzioni ausiliarie:

```
(%i1) inverseLaplace(SI,theta):=block([res],
                                M:SI,
                                MC:SI,
                                for i:1 thru 3 do(
                                  for j:1 thru 3 do
                                     (
                                       aC:M[i,j],
                                       b:ilt(aC,s,theta),
                                       MC[i,j]:b
                                     )
                                   ),
                                res:MC

                            )
```

(%o1) inverseLaplace(SI, ϑ) := **block** ([res], M: SI, MC: SI, **for** i **thru** 3 **do for** j **thru** 3 **do** (aC: $M_{i,j}$, b: ilt(aC, s, ϑ), MC$_{i,j}$: b), res: MC)

(%i2)
```
rotLaplace(k,theta):=block([res],
                         S:ident(3),
                         I:ident(3),
                       for i:1 thru 3 do
                         (
                         for j:1 thru 3 do
                            (
                              if i=j
                                 then S[i][j]:0
                              elseif j>i
                                 then (
                               temp:(-1)^(j-i)*k[3-remainder(i+j,3)],
                                      S[i][j]:temp,
                                      S[j][i]:-temp

                                       )
                              )
                          ),
                         res:inverseLaplace(invert(s*I-S),theta)


                        )
```

(%o2) $\mathrm{rotLaplace}(k,\vartheta) := \mathbf{block}\ ([\mathrm{res}], S\colon \mathrm{ident}(3), I\colon \mathrm{ident}(3),$
$\mathbf{for}\ i\ \mathbf{thru}\ 3\ \mathbf{do}\ \mathbf{for}\ j\ \mathbf{thru}\ 3\ \mathbf{do}\ \mathbf{if}\ i=j\ \mathbf{then}\ (S_i)_j\colon 0\ \mathbf{elseif}\ j>i\ \mathbf{then}\ (\mathrm{temp}\colon$
$(-1)^{j-i}\,k_{3-\mathrm{remainder}(i+j,3)}, (S_i)_j\colon \mathrm{temp}, (S_j)_i\colon -\mathrm{temp}), \mathrm{res}\colon \mathrm{inverseLaplace}(\mathrm{invert}(s\,I-S),\vartheta))$

(%i3)
```
Av(v,theta,d):=block([res],
                         Trot:rotLaplace(v,theta),
                         row:matrix([0,0,0,1]),
                         Atemp:addcol(Trot,d*transpose(v)),
                         A:addrow(Atemp,row),
                         res:A
                         )
```

(%o3) $\mathrm{Av}(v,\vartheta,d) := \mathbf{block}\ ([\mathrm{res}], \mathrm{Trot}\colon \mathrm{rotLaplace}(v,\vartheta), \mathrm{row}\colon \begin{pmatrix} 0 & 0 & 0 & 1 \end{pmatrix}, \mathrm{Atemp}\colon \mathrm{addcol}(\mathrm{Trot},$
$d\,\mathrm{transpose}(v)), A\colon \mathrm{addrow}(\mathrm{Atemp},\mathrm{row}), \mathrm{res}\colon A)$

(%i4)
```
Q(theta,d,alpha,a):=block([res],
                              tempMat:Av([0,0,1],theta,d).Av([1,0,0],alpha,a),
                              Qtrasf:zeromatrix(4,4),
                              for i:1 thru 4 do
                         (
                         for j:1 thru 4 do
                            (
                              Qtrasf[i][j]:trigreduce(tempMat[i][j])
                                       )
                            ),

                              res:Qtrasf
                              )
```

(%o4) $Q(\vartheta,d,\alpha,a) := \mathbf{block}\ ([\mathrm{res}], \mathrm{tempMat}\colon \mathrm{Av}([0,0,1],\vartheta,d)\cdot \mathrm{Av}([1,0,0],\alpha,a), \mathrm{Qtrasf}\colon$
$\mathrm{zeromatrix}(4,4), \mathbf{for}\ i\ \mathbf{thru}\ 4\ \mathbf{do}\ \mathbf{for}\ j\ \mathbf{thru}\ 4\ \mathbf{do}\ (\mathrm{Qtrasf}_i)_j\colon \mathrm{trigreduce}((\mathrm{tempMat}_i)_j), \mathrm{res}\colon \mathrm{Qtrasf})$

(%i5)
```
let(sin(q[1]), s[1]);
```

(%o5) $\sin(q_1) \longrightarrow s_1$

```
(%i6) let(sin(q[2]), s[2]);
```

(%o6) $\sin(q_2) \longrightarrow s_2$

```
(%i7) let(cos(q[1]),c[1]);
```

(%o7) $\cos(q_1) \longrightarrow c_1$

```
(%i8) let(cos(q[2]),c[2]);
```

(%o8) $\cos(q_2) \longrightarrow c_2$

```
(%i9) let(sin(q[1]+q[2]), s[12]);
```

(%o9) $\sin(q_2+q_1) \longrightarrow s_{12}$

```
(%i10) let(cos(q[1]+q[2]),c[12]);
```

(%o10) $\cos(q_2+q_1) \longrightarrow c_{12}$

```
(%i11)
```

Cinematica diretta:

```
(%i11) Q[rc](q1,q2,q3):=trigsimp(trigrat(trigreduce(trigexpand(
                        Q(0,q1,-(%pi/2),0).
                         Q(-(%pi/2),q2,-(%pi/2),0).
                         Q(0,q3,0,0)
                         ))));
```

(%o11) $Q_{\mathrm{rc}}(q1,\, q2,\, q3) := \mathrm{trigsimp}\Big(\mathrm{trigrat}\Big(\mathrm{trigreduce}\Big(\mathrm{trigexpand}\Big(Q\Big(0,\, q1,\, -\frac{\pi}{2},\, 0\Big)\cdot Q\Big(-\frac{\pi}{2},\, q2,\, -\frac{\pi}{2},0\Big)\cdot Q(0,\, q3, 0, 0)\Big)\Big)\Big)\Big)$

```
(%i12) Q[rc](q[1],q[2],q[3]);
```

(%o12) $\begin{pmatrix} 0 & 0 & 1 & q_3 \\ 0 & -1 & 0 & q_2 \\ 1 & 0 & 0 & q_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$

```
(%i13)
```

Cinematica inversa

Dato che il robot cartesiano è un robot con 3 gradi di libertà (3DOF) è possibile effettuare l'analisi della cinematica inversa di posizione e di orientamento inverso.

Occorre inizialmente individuare lo spazio di lavoro, le soluzioni generiche, singolari ed infine le variabili di giunto q_i.

Dalla cinematica diretta sappiamo che:

$$Q_{\text{cartesiano}} = \begin{pmatrix} 0 & 0 & 1 & q_3 \\ 0 & -1 & 0 & q_2 \\ 1 & 0 & 0 & q_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Cinematica inversa di posizione

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} q_3 \\ q_2 \\ q_1 \end{pmatrix}$$

Da cui lo spazio di lavoro, idealmente, è $\mathbb{R}^3$, non vi sono singolarità ma solamente una soluzione. Inoltre, possiamo semplicemente risolvere il problema di cinematica inversa di posizione ponendo:

$$q_1 = z$$
$$q_2 = y$$
$$q_3 = x$$

Orientamento inverso

La risoluzione del problema di orientamento inverso si basa sulla scelta di una terna di Eulero o nautica in condizione non singolari. In particolare, l'elemento più semplice deve risultare diverso da ± 1. In particolare:

$$R_{\text{cartesiano}} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & -1 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

$$\text{Terna nautica: } R_{\text{yzx}} = \begin{pmatrix} c_y c_z & \cdots\cdots\cdots \\ s_z \quad c_z c_x & -c_z s_x \\ s_y c_z & \cdots\cdots\cdots \end{pmatrix}$$

Poiché l'elemento $s_z = 0 \neq \pm 1$ è possibile risolvere il problema di orientamento inverso con la terna nautica yzx. Infatti:

$$s_z = 0 \neq \pm 1 \longrightarrow c_z = \pm\sqrt{1 - s_z} = \pm 1$$

$$\phi_z = \text{atan2}(s_z, c_z)$$

$$\phi_z = \{0, \pi\}$$

$$\begin{matrix} c_y c_z = 0 & & c_y = 0 \\ & \xrightarrow{c_z = \pm 1} & \\ s_y c_z = 1 & & s_y = \pm 1 \end{matrix}$$

$$\phi_y = \text{atan2}(\pm s_y, c_y)$$

$$\phi_y = \left\{\frac{\pi}{2}, -\frac{\pi}{2}\right\}$$

$$\begin{matrix} c_z c_x = -1 & & c_x = \mp 1 \\ & \xrightarrow{c_z = \pm 1} & \\ -c_z s_x = 0 & & s_x = 0 \end{matrix}$$

$$\phi_x = \text{atan2}(s_x, \mp c_x)$$

$$\phi_x = \{\pi, 0\}$$

Riassumendo, le soluzioni sono:

$$\begin{pmatrix} \pi \\ \frac{\pi}{2} \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ -\frac{\pi}{2} \\ \pi \end{pmatrix}$$

In alternativa utilizzando una terna di Eulero:

$$R_{\text{zyz}} = \begin{pmatrix} \cdots\cdots\cdots\cdots & -c_\alpha s_\beta \\ \cdots\cdots\cdots\cdots & -s_\alpha s_\beta \\ s_\beta c_\gamma \quad - s_\beta s_\gamma & c_\beta \end{pmatrix}$$

$$c_\beta = 0 \longrightarrow s_\beta = \pm 1$$

$$\beta = \text{atan2}(\pm s_\beta, c_\beta)$$

$$\beta = \left\{\frac{\pi}{2}, -\frac{\pi}{2}\right\}$$

$$\begin{matrix} c_\alpha s_\beta = 1 & & c_\alpha = \mp 1 \\ & -s_\beta = \pm 1 \rightarrow & \\ s_\alpha s_\beta = 0 & & s_\alpha = 0 \end{matrix}$$

$$\alpha = \text{atan2}(s_\alpha, \mp c_\alpha)$$

$$\alpha = \{\pi, 0\}$$

$$\begin{matrix} c_\gamma s_\beta = 1 & & c_\gamma = \pm 1 \\ & -s_\beta = \pm 1 \rightarrow & \\ s_\gamma s_\beta = 0 & & s_\gamma = 0 \end{matrix}$$

$$\gamma = \text{atan2}(s_\gamma, \mp c_\gamma)$$

$$\gamma = \{0, \pi\}$$

Riassumendo, le soluzioni sono:

$$\begin{pmatrix} \pi \\ \frac{\pi}{2} \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ -\frac{\pi}{2} \\ \pi \end{pmatrix}$$

```
(%i13) R(k,theta):= block([res],
                         if k = x
                             then res:matrix([1,0,0],
                            [0,cos(theta),-sin(theta)],
                            [0,sin(theta), cos(theta)])
                         elseif k = y
                             then res:matrix([cos(theta),0,sin(theta)],
                            [0,1,0],
                            [-sin(theta),0, cos(theta)])
                         elseif k = z
                             then res:matrix([cos(theta),-sin(theta),0],
                            [sin(theta),cos(theta),0],
                            [0,0, 1])
                          else
                            res:"Incorrect axis of rotation"
                          )
```

(%o13) $R(k, \vartheta) := \mathbf{block}\left([\text{res}], \mathbf{if}\ k = x\ \mathbf{then}\ \text{res:} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\vartheta) & -\sin(\vartheta) \\ 0 & \sin(\vartheta) & \cos(\vartheta) \end{pmatrix} \mathbf{elseif}\ k = y\ \mathbf{then}\ \text{res:} \right.$

$\begin{pmatrix} \cos(\vartheta) & 0 & \sin(\vartheta) \\ 0 & 1 & 0 \\ -\sin(\vartheta) & 0 & \cos(\vartheta) \end{pmatrix}$ **elseif** $k = z$ **then** res: $\begin{pmatrix} \cos(\vartheta) & -\sin(\vartheta) & 0 \\ \sin(\vartheta) & \cos(\vartheta) & 0 \\ 0 & 0 & 1 \end{pmatrix}$ **else** res: Incorrect axis of rotation)

```
(%i15) Reulero:R(z,gamma).R(y,beta).R(z,alpha);
```

(%o15) $(\cos(\alpha)\cos(\beta)\cos(\gamma) - \sin(\alpha)\sin(\gamma), -\cos(\alpha)\sin(\gamma) - \sin(\alpha)\cos(\beta)\cos(\gamma), \sin(\beta)\cos(\gamma); \cos(\alpha)\cos(\beta)\sin(\gamma) + \sin(\alpha)\cos(\gamma), \cos(\alpha)\cos(\gamma) - \sin(\alpha)\cos(\beta)\sin(\gamma), \sin(\beta)\sin(\gamma); -\cos(\alpha)\sin(\beta), \sin(\alpha)\sin(\beta), \cos(\beta))$

```
(%i17) Rnautico:R(y,phi[y]).R(z,phi[z]).R(x,phi[x]);
```

(%o17) $(\cos(\varphi_y)\cos(\varphi_z), \sin(\varphi_x)\sin(\varphi_y) - \cos(\varphi_x)\cos(\varphi_y)\sin(\varphi_z), \sin(\varphi_x)\cos(\varphi_y)\sin(\varphi_z) + \cos(\varphi_x)\sin(\varphi_y); \sin(\varphi_z), \cos(\varphi_x)\cos(\varphi_z), -\sin(\varphi_x)\cos(\varphi_z); -\sin(\varphi_y)\cos(\varphi_z), \cos(\varphi_x)\sin(\varphi_y)\sin(\varphi_z) + \sin(\varphi_x)\cos(\varphi_y), \cos(\varphi_x)\cos(\varphi_y) - \sin(\varphi_x)\sin(\varphi_y)\sin(\varphi_z))$

```
(%i18) isRotation(M):=block([MC,res],
                          I:ident(3),
                          MC:ident(3),
                          for i:1 thru 3 do
                          (
                          for j:1 thru 3 do
                             (
                                MC[i][j]:M[i][j]
                                )
                             ),
                          MMT:trigsimp(expand(MC.transpose(MC))),
                          detM:trigsimp(expand(determinant(MC))),

                          if MMT=I  and detM=1
                             then(

                                  return(res:1)
                                  )

                          else(

                                res: "R is not rotation matrix"
                                )
                          )
```

(%o18) isRotation(M) := **block** ([MC, res], I: ident(3), MC: ident(3), **for** i **thru** 3 **do for** j **thru** 3 **do** $(\mathrm{MC}_i)_j$: $(M_i)_j$, MMT: trigsimp(expand(MC · transpose(MC))), detM: trigsimp(expand(determinant(MC))), **if** MMT $= I \wedge$ detM $= 1$ **then** return(res: 1) **else** res: R is not rotation matrix)

```
(%i21) invCartesiano(Qdiretta):=block([pos,orien1,orien2,res],
                                        rotation:isRotation(Qdiretta),
                                        if rotation=1 then(
                                            pos:transpose(
                                                  [
                                                  Qdiretta[1][4],
                                                  Qdiretta[2][4],
                                                  Qdiretta[3][4]
                                                  ]),
                                            sz:Qdiretta[2][1],
                                            cz:sqrt(1-sz^(2)),
                                            phiz1:atan2(sz,cz),
                                            phiz2:atan2(sz,-cz),
                                            cy:Qdiretta[1][1]/cz,
                                            sy:Qdiretta[3][1]/cz,
                                            phiy1:atan2(sy,cy),
                                            phiy2:atan2(-sy,cy),
                                            cx:Qdiretta[2][2]/cz,
                                            sx:Qdiretta[2][3]/cz,
                                            phix1:atan2(sx,cx),
                                            phix2:atan2(sx,-cx),
                                            orien1:transpose([phix1,phiy1,phiz1]),
                                            orien2:transpose([phix2,phiy2,phiz2]),
                                            res:[pos,orien1,orien2]

                                        )
                                        else res:rotation
                                        );
```

(%o21) $\mathrm{invCartesiano}(\mathrm{Qdiretta}) := \mathbf{block}\Big([\mathrm{pos}, \mathrm{orien1}, \mathrm{orien2}, \mathrm{res}], \mathrm{rotation}\colon$
$\mathrm{isRotation}(\mathrm{Qdiretta}), \mathbf{if}\ \mathrm{rotation} = 1\ \mathbf{then}\ \Big(\mathrm{pos}\colon \mathrm{transpose}([(\mathrm{Qdiretta}_1)_4, (\mathrm{Qdiretta}_2)_4,$
$(\mathrm{Qdiretta}_3)_4]), \mathrm{sz}\colon (\mathrm{Qdiretta}_2)_1, \mathrm{cz}\colon \sqrt{1-\mathrm{sz}^2}, \mathrm{phiz1}\colon \mathrm{atan2}(\mathrm{sz}, \mathrm{cz}), \mathrm{phiz2}\colon \mathrm{atan2}(\mathrm{sz}, -\mathrm{cz}), \mathrm{cy}\colon$
$\frac{(\mathrm{Qdiretta}_1)_1}{\mathrm{cz}}, \mathrm{sy}\colon \frac{(\mathrm{Qdiretta}_3)_1}{\mathrm{cz}}, \mathrm{phiy1}\colon \mathrm{atan2}(\mathrm{sy}, \mathrm{cy}), \mathrm{phiy2}\colon \mathrm{atan2}(-\mathrm{sy}, \mathrm{cy}), \mathrm{cx}\colon \frac{(\mathrm{Qdiretta}_2)_2}{\mathrm{cz}}, \mathrm{sx}\colon$
$\frac{(\mathrm{Qdiretta}_2)_3}{\mathrm{cz}}, \mathrm{phix1}\colon \mathrm{atan2}(\mathrm{sx}, \mathrm{cx}), \mathrm{phix2}\colon \mathrm{atan2}(\mathrm{sx}, -\mathrm{cx}), \mathrm{orien1}\colon \mathrm{transpose}([\mathrm{phix1}, \mathrm{phiy1}, \mathrm{phiz1}]),$
$\mathrm{orien2}\colon \mathrm{transpose}([\mathrm{phix2}, \mathrm{phiy2}, \mathrm{phiz2}]), \mathrm{res}\colon [\mathrm{pos}, \mathrm{orien1}, \mathrm{orien2}]\Big)\ \mathbf{else}\ \mathrm{res}\colon \mathrm{rotation}\Big)$

```
(%i22) invCartesiano(Q[rc](q[1],q[2],q[3]));
```

(%o22) $$\left[\begin{pmatrix} q_3 \\ q_2 \\ q_1 \end{pmatrix}, \begin{pmatrix} \pi \\ \frac{\pi}{2} \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ -\frac{\pi}{2} \\ \pi \end{pmatrix}\right]$$

```
(%i24) invCartesiano(Q[rc](10,15,20));
```

(%o24) $$\left[\begin{pmatrix} 20 \\ 15 \\ 10 \end{pmatrix}, \begin{pmatrix} \pi \\ \frac{\pi}{2} \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ -\frac{\pi}{2} \\ \pi \end{pmatrix}\right]$$

(%i25)

Singolarità di velocità

```
(%i11) x:q3;
```
(%o11) $q3$

```
(%i12) y:q2
```
(%o12) $q2$

```
(%i13) z:q1
```
(%o13) $q1$

```
(%i31) J:matrix([diff(x,q1),diff(x,q2),diff(x,q3)],
                [diff(y,q1),diff(y,q2),diff(y,q3)],
                [diff(z,q1),diff(z,q2),diff(z,q3)]);
```
(%o31) $\begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}$

```
(%i32) dJ:determinant(J);
```
(%o32) -1

```
(%i33)
```

Poiché $\det(J) \neq 0$ $\forall$q, non vi sono singolarità cinematiche di velocità. Infatti è possibile ottenere tutte le velocita che $\in \mathrm{Im}\{J\} = \left[\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}\right]$.

In aggiunta le velocità che risultano singolare sono date da $\ker\{J\} = \left\{\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}\right\}$.

```
(%i33) Jtr:-transpose(J);
```
(%o33) $\begin{pmatrix} 0 & 0 & -1 \\ 0 & -1 & 0 \\ -1 & 0 & 0 \end{pmatrix}$

```
(%i34) dJtr:determinant(Jtr);
```
(%o34) 1

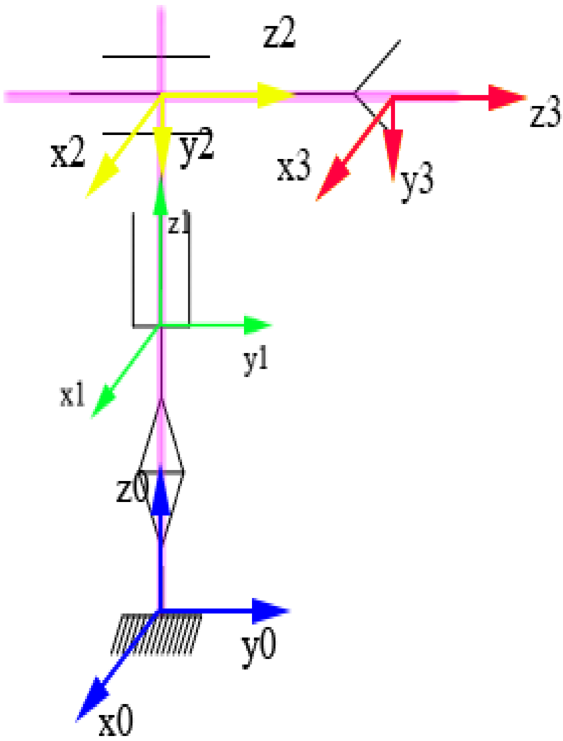
```
(%i35)
```

Poiché $\det(J) \neq 0$ $\forall$q, non vi sono singolarità cinematiche di forza. Infatti è possibile ottenere tutte le forze che $\in \mathrm{Im}\{J\} = \left[\begin{pmatrix} -1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ -1 \end{pmatrix}, \begin{pmatrix} 0 \\ -1 \\ 0 \end{pmatrix}\right]$.

In aggiunta le forze che risultano singolare sono date da $\ker\{J\} = \left\{\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}\right\}$.

Cinematica diretta Robot Cilindrico

***N.B.:* le grandezze diverse da quelle di giunto q_i sono L_i, D_i. Esse sono rispettivamente la distanza tra i sistemi di riferimento R_i e R_{i+1} nelle operazioni della matrice avvitamento $A_z(\theta, d)$ e $A_x(\alpha, a)$.**



	ϑ	d	α	a
1	q_1	L_1	0	0
2	0	q_2	$-\frac{\pi}{2}$	0
3	0	q_3	0	0

Tabella 1.

Funzioni ausiliarie:

```
(%i1) inverseLaplace(SI,theta):=block([res],
                              M:SI,
                              MC:SI,
                              for i:1 thru 3 do(
                                for j:1 thru 3 do
                                   (
                                     aC:M[i,j],
                                     b:ilt(aC,s,theta),
                                     MC[i,j]:b
                                    )
                                 ),
                              res:MC

                            )
```

(%o1) inverseLaplace(SI, ϑ) := **block** ([res], M: SI, MC: SI, **for** i **thru** 3 **do for** j **thru** 3 **do** (aC: $M_{i,j}$, b: ilt(aC, s, ϑ), MC$_{i,j}$: b), res: MC)

(%i2)
```
rotLaplace(k,theta):=block([res],
                        S:ident(3),
                        I:ident(3),
                       for i:1 thru 3 do
                        (
                        for j:1 thru 3 do
                           (
                             if i=j
                                then S[i][j]:0
                             elseif j>i
                                then (
                              temp:(-1)^(j-i)*k[3-remainder(i+j,3)],
                                     S[i][j]:temp,
                                     S[j][i]:-temp

                                      )
                             )
                         ),
                        res:inverseLaplace(invert(s*I-S),theta)


                       )
```

(%o2) $\mathrm{rotLaplace}(k,\vartheta):=\mathbf{block}\ ([\mathrm{res}], S: \mathrm{ident}(3), I: \mathrm{ident}(3),$ $\mathbf{for}\ i\ \mathbf{thru}\ 3\ \mathbf{do}\ \mathbf{for}\ j\ \mathbf{thru}\ 3\ \mathbf{do}\ \mathbf{if}\ i=j\ \mathbf{then}\ (S_i)_j: 0\ \mathbf{elseif}\ j>i\ \mathbf{then}\ (\mathrm{temp}:$ $(-1)^{j-i}\, k_{3-\mathrm{remainder}(i+j,3)}, (S_i)_j: \mathrm{temp}, (S_j)_i: -\mathrm{temp}), \mathrm{res}: \mathrm{inverseLaplace}(\mathrm{invert}(s\,I-S),\vartheta))$

(%i3)
```
Av(v,theta,d):=block([res],
                        Trot:rotLaplace(v,theta),
                        row:matrix([0,0,0,1]),
                        Atemp:addcol(Trot,d*transpose(v)),
                        A:addrow(Atemp,row),
                        res:A
                        )
```

(%o3) $\mathrm{Av}(v,\vartheta,d):=\mathbf{block}\ ([\mathrm{res}], \mathrm{Trot}: \mathrm{rotLaplace}(v,\vartheta), \mathrm{row}: \begin{pmatrix} 0 & 0 & 0 & 1 \end{pmatrix}, \mathrm{Atemp}: \mathrm{addcol}(\mathrm{Trot},$ $d\,\mathrm{transpose}(v)), A: \mathrm{addrow}(\mathrm{Atemp},\mathrm{row}), \mathrm{res}: A)$

(%i4)
```
Q(theta,d,alpha,a):=block([res],
                             tempMat:Av([0,0,1],theta,d).Av([1,0,0],alpha,a),
                             Qtrasf:zeromatrix(4,4),
                             for i:1 thru 4 do
                        (
                        for j:1 thru 4 do
                           (
                             Qtrasf[i][j]:trigreduce(tempMat[i][j])
                                       )
                           ),

                             res:Qtrasf
                             )
```

(%o4) $Q(\vartheta,d,\alpha,a):=\mathbf{block}\ ([\mathrm{res}], \mathrm{tempMat}: \mathrm{Av}([0,0,1],\vartheta,d)\cdot \mathrm{Av}([1,0,0],\alpha,a), \mathrm{Qtrasf}:$ $\mathrm{zeromatrix}(4,4), \mathbf{for}\ i\ \mathbf{thru}\ 4\ \mathbf{do}\ \mathbf{for}\ j\ \mathbf{thru}\ 4\ \mathbf{do}\ (\mathrm{Qtrasf}_i)_j: \mathrm{trigreduce}((\mathrm{tempMat}_i)_j), \mathrm{res}: \mathrm{Qtrasf})$

(%i5)
```
let(sin(q[1]), s[1]);
```

(%o5) $\sin(q_1) \longrightarrow s_1$

```
(%i6) let(sin(q[2]), s[2]);
```

(%o6) $\sin(q_2) \longrightarrow s_2$

```
(%i7) let(cos(q[1]),c[1]);
```

(%o7) $\cos(q_1) \longrightarrow c_1$

```
(%i8) let(cos(q[2]),c[2]);
```

(%o8) $\cos(q_2) \longrightarrow c_2$

```
(%i9) let(sin(q[1]+q[2]), s[12]);
```

(%o9) $\sin(q_2 + q_1) \longrightarrow s_{12}$

```
(%i10) let(cos(q[1]+q[2]),c[12]);
```

(%o10) $\cos(q_2 + q_1) \longrightarrow c_{12}$

```
(%i11)

(%i11) Q[cilindrico](q1,q2,q3,L1):=trigsimp(trigrat(trigreduce(trigexpand(
                         Q(q1,L1,0,0).
                          Q(0,q2,-(%pi/2),0).
                          Q(0,q3,0,0)
                          ))));
```

(%o11) $Q_{\text{cilindrico}}(q1, q2, q3, L1) := \text{trigsimp}\Big(\text{trigrat}\Big(\text{trigreduce}\Big(\text{trigexpand}\Big(Q(q1, L1, 0, 0) \cdot Q\Big(0, q2, -\frac{\pi}{2}, 0\Big) \cdot Q(0, q3, 0, 0)\Big)\Big)\Big)\Big)$

```
(%i12) Qcilindrico:Q[cilindrico](q[1],q[2],q[3],L1);
```

(%o12) $\begin{pmatrix} \cos(q_1) & 0 & -\sin(q_1) & -q_3 \sin(q_1) \\ \sin(q_1) & 0 & \cos(q_1) & q_3 \cos(q_1) \\ 0 & -1 & 0 & L1 + q_2 \\ 0 & 0 & 0 & 1 \end{pmatrix}$

```
(%i13) letsimp(Qcilindrico);
```

(%o13) $\begin{pmatrix} c_1 & 0 & -s_1 & -s_1\, q_3 \\ s_1 & 0 & c_1 & c_1\, q_3 \\ 0 & -1 & 0 & L1 + q_2 \\ 0 & 0 & 0 & 1 \end{pmatrix}$

```
(%i14)
```

Cinematica Inversa Robot Cilindrico

Al fine di risolvere il problema di cinematica inversa del robot cilindrico occorre risolevere il problema di posizione ed orientamente inverso.Inizialmente individuare lo spazio di lavoro, le soluzioni generiche, singolari ed infine le variabili di giunto q_i ed in seguito determinare l'orientamento del robot.

Dalla cinematica diretta del robot cilindrico sappiamo che:

$$Q_{\text{cilindrico}} = \begin{pmatrix} c_1 & 0 & -s_1 & -s_1\, q_3 \\ s_1 & 0 & c_1 & c_1\, q_3 \\ 0 & -1 & 0 & q_2 + L_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Cinematica inversa di posizione

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} -\sin(q_1)\, q_3 \\ \cos(q_1)\, q_3 \\ q_2 + L_1 \end{pmatrix}$$

La variabile di igunto q_2, poiché L_1 e z sono noti:

$$q_2 = z - L_1$$

Quindi occorre risolvere:

$$\begin{cases} x = -\sin(q_1)\, q_3 \\ y = \cos(q_1)\, q_3 \end{cases} \longrightarrow \begin{cases} x^2 = \sin(q_1)^2\, q_3^2 \\ y^2 = \cos(q_1)^2\, q_3^2 \end{cases}$$

$$x^2 + y^2 = q_3^2\, (\sin(q_1)^2 + \cos(q_1)^2)$$

Determinando di conseguenza lo spazio operativo $:= x^2 + y^2 = q_3^2$

Rappresenta un cilindro il cui asse di rotazione è una soluzione singolare ottenuta da:

$$q_3 = \pm\sqrt{x^2 + y^2} \quad 2\,\text{soluzioni}\,\text{generiche}$$

$$q_3 = 0 \longrightarrow \sqrt{x^2 + y^2} = 0 \longrightarrow \text{soluzione}\,\text{singolare}$$

Per determinare q_1 occorre supporre che $x^2 + y^2 \neq 0 \longrightarrow q_3 \neq 0$:

$$\begin{cases} x = -\sin(q_1)\, q_3 \\ y = \cos(q_1)\, q_3 \end{cases} \longrightarrow \begin{cases} \sin(q_1) = -\frac{x}{q_3} \\ \cos(q_1) = \frac{y}{q_3} \end{cases}$$

Infine:

$$q_1 = \text{atan2}(\sin(q_1), \cos(q_1))$$

Orientamento inverso

La risoluzione del problema di orientamento inverso si basa sulla scelta di una terna di Eulero o nautica in condizione non singolari.

$$R_{\text{cilindrico}} = \begin{pmatrix} c_1 & 0 & -s_1 \\ s_1 & 0 & c_1 \\ 0 & -1 & 0 \end{pmatrix}$$

$$R_{\text{zyx}} = \begin{pmatrix} c_y c_z & \cdots\cdots\cdots \\ c_y c_z & \cdots\cdots\cdots \\ -s_y & s_x c_y & c_x c_y \end{pmatrix}$$

Poiché l'elemento $-s_y = 0 \neq \pm 1$ è possibile risolvere il problema di orientamento inverso con la terna nautica zyx. In particolare:

$$s_y = 0 \longrightarrow c_y = \pm 1 \longrightarrow \phi_y = \text{atan2}(s_y, c_y) \quad\longrightarrow\quad \phi_y = \begin{cases} 0 \\ \pi \end{cases}$$

$$\begin{cases} c_y s_x = -1 \\ c_y c_x = 0 \end{cases} \quad\longrightarrow\quad \phi_x = \text{atan2}(\mp s_x, c_x) \longrightarrow \phi_x = \begin{cases} -\frac{\pi}{2} \\ \frac{\pi}{2} \end{cases}$$

$$\begin{cases} c_y c_z = c_1 \\ c_y s_z = s_1 \end{cases} \longrightarrow \phi_z = \operatorname{atan2}(\pm s_1, \pm c_1) \longrightarrow \phi_z = \begin{cases} q_1 \\ q_1 + \pi \end{cases}$$

Riassumendo, le soluzioni sono:

$$\begin{pmatrix} -\frac{\pi}{2} \\ 0 \\ q_1 \end{pmatrix}, \begin{pmatrix} \frac{\pi}{2} \\ \pi \\ q_1 + \pi \end{pmatrix}$$

In alternativa, utilizzando una terna di Eulero:

$$R_{\text{yxy}} = \begin{pmatrix} \cdots & \sin(\alpha)\sin(\beta) & \cdots \\ \sin(\beta)\sin(\gamma) & \cos(\beta) & -\sin(\beta)\cos(\gamma) \\ \cdots & \cos(\alpha)\sin(\beta) & \cdots \end{pmatrix}$$

$$\cos(\beta) = 0 \longrightarrow \sin(\beta) = \pm\sqrt{1 - \cos(\beta)^2} = \pm 1$$

$$\beta = \operatorname{atan2}(\pm\sin(\beta), \cos(\beta))$$

$$\beta = \begin{cases} \frac{\pi}{2} \\ -\frac{\pi}{2} \end{cases}$$

$$\begin{cases} \sin(\alpha)\sin(\beta) = 0 \\ \cos(\alpha)\sin(\beta) = -1 \end{cases} \longrightarrow \begin{cases} \sin(\alpha) = 0 \\ \cos(\alpha) = \mp 1 \end{cases}$$

$$\alpha = \operatorname{atan2}(\sin(\alpha), \mp\cos(\alpha))$$

$$\alpha = \begin{cases} \pi \\ 0 \end{cases}$$

$$\begin{cases} \sin(\beta)\sin(\gamma) = s_1 \\ -\sin(\beta)\cos(\gamma) = c_1 \end{cases} \longrightarrow \begin{cases} \sin(\gamma) = \pm s_1 \\ \cos(\gamma) = \mp c_1 \end{cases}$$

$$\gamma = \operatorname{atan2}(\pm s_1, \mp c_1)$$

$$\gamma = \begin{cases} q_1 \\ q_1 + \pi \end{cases}$$

Riassumendo, si hanno 2 soluzioni:

$$\begin{pmatrix} \frac{\pi}{2} \\ \pi \\ q_1 + \pi \end{pmatrix}, \begin{pmatrix} -\frac{\pi}{2} \\ 0 \\ q_1 \end{pmatrix}$$

(%i16)
```
R(k,theta):= block([res],
                    if k = x
                        then res:matrix([1,0,0],
                       [0,cos(theta),-sin(theta)],
                       [0,sin(theta), cos(theta)])
                    elseif k = y
                        then res:matrix([cos(theta),0,sin(theta)],
                       [0,1,0],
                       [-sin(theta),0, cos(theta)])
                    elseif k = z
                        then res:matrix([cos(theta),-sin(theta),0],
                       [sin(theta),cos(theta),0],
                       [0,0, 1])
                     else
                       res:"Incorrect axis of rotation"
                     )
```

(%o16) $R(k,\vartheta):=\mathbf{block}\left([\mathrm{res}], \mathbf{if}\ k=x\ \mathbf{then}\ \mathrm{res}: \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\vartheta) & -\sin(\vartheta) \\ 0 & \sin(\vartheta) & \cos(\vartheta) \end{pmatrix} \mathbf{elseif}\ k=y\ \mathbf{then}\ \mathrm{res}: \begin{pmatrix} \cos(\vartheta) & 0 & \sin(\vartheta) \\ 0 & 1 & 0 \\ -\sin(\vartheta) & 0 & \cos(\vartheta) \end{pmatrix} \mathbf{elseif}\ k=z\ \mathbf{then}\ \mathrm{res}: \begin{pmatrix} \cos(\vartheta) & -\sin(\vartheta) & 0 \\ \sin(\vartheta) & \cos(\vartheta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \mathbf{else}\ \mathrm{res}: \text{Incorrect axis of rotation}\right)$

(%i17)
```
isRotation(M):=block([MC,res],
                      I:ident(3),
                      MC:ident(3),
                      for i:1 thru 3 do
                      (
                      for j:1 thru 3 do
                         (
                            MC[i][j]:M[i][j]
                            )
                         ),
                      MMT:trigsimp(expand(MC.transpose(MC))),
                      detM:trigsimp(expand(determinant(MC))),

                      if MMT=I  and detM=1
                         then(

                             return(res:1)
                             )

                      else(

                            res: "R is not rotation matrix"
                            )
                      )
```

(%o17) $\mathrm{isRotation}(M):=\mathbf{block}\ ([\mathrm{MC},\mathrm{res}], I{:}\,\mathrm{ident}(3), \mathrm{MC}{:}\,\mathrm{ident}(3),$
$\mathbf{for}\ i\ \mathbf{thru}\ 3\ \mathbf{do}\ \mathbf{for}\ j\ \mathbf{thru}\ 3\ \mathbf{do}\ (\mathrm{MC}_i)_j{:}\,(M_i)_j, \mathrm{MMT}{:}\,\mathrm{trigsimp}(\mathrm{expand}(\mathrm{MC}\cdot\mathrm{transpose}(\mathrm{MC}))),$
$\mathrm{detM}{:}\,\mathrm{trigsimp}(\mathrm{expand}(\mathrm{determinant}(\mathrm{MC}))), \mathbf{if}\ \mathrm{MMT}=I\wedge\mathrm{detM}=1\ \mathbf{then}\ \mathrm{return}(\mathrm{res}{:}\,1)\ \mathbf{else}\ \mathrm{res}{:}\,$R is not rotation matrix)

(%i28)
```
invCilindrico(x,y,z,phi,L1):=block([R,pos1,pos2,orien1,orien2,res],

                                   if x^2+y^2#0 then(
                                     R:matrix([cos(phi),0,sin(phi)],
                                              [sin(phi),0,cos(phi)],
                                              [0,1,0]),
                                     q3:cabs(trigsimp(sqrt(x^2+y^2))),
                                     q1alto:atan2(-x/q3,y/q3),
                                     q1basso:atan2(x/q3,-y/q3),
                                     q2:z-L1,
                                     pos1:[q1alto,q2,q3],
                                     pos2:[q1basso,q2,-q3],
                                     sy:R[3][1],
                                     cy:sqrt(1-sy^2),
                                     phiy2:atan2(sy,cy),
                                     phiy1:atan2(sy,-cy),
                                     sx:R[3][2],
                                     cx:R[3][3],
                                     phix1:atan2(-sx,cx),
                                     phix2:atan2(sx,cx),
                                     cz:R[1][1],
                                     sz:R[2][1],
                                     phiz1:atan2(sz,cz),
                                     phiz2:atan2(-sz,-cz),
                                     orien1:[phix1,phiy1,phiz1],
                                     orien2:[phix2,phiy2,phiz2],
                                     res:[pos1,pos2,orien1,orien2]
                                   )
                                   else res:"La configurazione è singolare"

                                 );
```

(%o28) $\mathrm{invCilindrico}(x,\ y,\ z,\ \varphi,\ L1) := \mathbf{block}\left([R,\ \mathrm{pos1},\ \mathrm{pos2},\ \mathrm{orien1},\ \mathrm{orien2},\ \mathrm{res}],\ \mathbf{if}\ x^2+y^2 \neq 0\ \mathbf{then}\left(R{:}\begin{pmatrix}\cos(\varphi) & 0 & \sin(\varphi)\\ \sin(\varphi) & 0 & \cos(\varphi)\\ 0 & 1 & 0\end{pmatrix},\ q3{:}\ \mathrm{cabs}\left(\mathrm{trigsimp}\left(\sqrt{x^2+y^2}\right)\right),\ q1\mathrm{alto}{:}\ \mathrm{atan2}\left(\frac{-x}{q3},\frac{y}{q3}\right),\ q1\mathrm{basso}{:}\ \mathrm{atan2}\left(\frac{x}{q3},\frac{-y}{q3}\right),\ q2{:}\ z-L1,\ \mathrm{pos1}{:}\ [q1\mathrm{alto},\ q2,\ q3],\ \mathrm{pos2}{:}\ [q1\mathrm{basso},\ q2,\ -q3],\ \mathrm{sy}{:}\ (R_3)_1,\ \mathrm{cy}{:}\ \sqrt{1-\mathrm{sy}^2},\ \mathrm{phiy2}{:}\ \mathrm{atan2}(\mathrm{sy},\ \mathrm{cy}),\ \mathrm{phiy1}{:}\ \mathrm{atan2}(\mathrm{sy},\ -\mathrm{cy}),\ \mathrm{sx}{:}\ (R_3)_2,\ \mathrm{cx}{:}\ (R_3)_3,\ \mathrm{phix1}{:}\ \mathrm{atan2}(-\mathrm{sx},\ \mathrm{cx}),\ \mathrm{phix2}{:}\ \mathrm{atan2}(\mathrm{sx},\ \mathrm{cx}),\ \mathrm{cz}{:}\ (R_1)_1,\ \mathrm{sz}{:}\ (R_2)_1,\ \mathrm{phiz1}{:}\ \mathrm{atan2}(\mathrm{sz},\ \mathrm{cz}),\ \mathrm{phiz2}{:}\ \mathrm{atan2}(-\mathrm{sz},\ -\mathrm{cz}),\ \mathrm{orien1}{:}\ [\mathrm{phix1},\ \mathrm{phiy1},\ \mathrm{phiz1}],\ \mathrm{orien2}{:}\ [\mathrm{phix2},\ \mathrm{phiy2},\ \mathrm{phiz2}],\ \mathrm{res}{:}\ [\mathrm{pos1},\ \mathrm{pos2},\ \mathrm{orien1},\ \mathrm{orien2}]\right)\ \mathbf{else}\ \mathrm{res}{:}\ \text{La configurazione è singolare}\right)$

(%i29)
```
QdirettaC: Q[cilindrico](q[1],q[2],q[3],15);
```

(%o29)
$$\begin{pmatrix}\cos(q_1) & 0 & -\sin(q_1) & -q_3\sin(q_1)\\ \sin(q_1) & 0 & \cos(q_1) & q_3\cos(q_1)\\ 0 & -1 & 0 & q_2+15\\ 0 & 0 & 0 & 1\end{pmatrix}$$

(%i42) `invCilindrico(10,10,10,%pi/2,15);`

(%o42) $\left[\left[-\frac{\pi}{4},-5,5\,2^{\frac{3}{2}}\right],\left[\frac{3\,\pi}{4},-5,-5\,2^{\frac{3}{2}}\right],\left[-\frac{\pi}{2},\pi,\frac{\pi}{2}\right],\left[\frac{\pi}{2},0,-\frac{\pi}{2}\right]\right]$

(%i43) `QdirettaC: Q[cilindrico](-(%pi/4),-5,5*2^((3/2)),15);`

(%o43) $\begin{pmatrix} \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} & 10 \\ -\frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} & 10 \\ 0 & -1 & 0 & 10 \\ 0 & 0 & 0 & 1 \end{pmatrix}$

(%i44)

Singolarità di velocità

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} -\sin(q_1)\,q_3 \\ \cos(q_1)\,q_3 \\ q_2+L_1 \end{pmatrix}$$

$$J=\frac{\delta h}{\delta q}=\begin{pmatrix} -q_3c_1 & 0 & -s_1 \\ -q_3s_1 & 0 & c_1 \\ 0 & 1 & 0 \end{pmatrix}$$

$$\det(J)=0 \Leftrightarrow q_3=0$$

$$J(q_3=0)=\begin{pmatrix} 0 & 0 & -s_1 \\ 0 & 0 & c_1 \\ 0 & 1 & 0 \end{pmatrix}$$

$$a\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}+b\begin{pmatrix} -s_1 \\ c_1 \\ 0 \end{pmatrix}=\Im m \to \mathrm{Ker}\{J\}=\Im m\left\{\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}\right\}$$

In singolarità con $v=\begin{pmatrix} V \\ 0 \\ 0 \end{pmatrix} \forall v \Rightarrow w=\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$

(%i1) `x:-sin(q[1])*q[3]`

(%o1) $-q_3\sin(q_1)$

(%i2) `y:cos(q[1])*q[3]`

(%o2) $q_3\cos(q_1)$

(%i3) `z:q[2]+L[1]`

(%o3) q_2+L_1

(%i4)
```
J(q1,q2,q3):=matrix([diff(x,q1),diff(x,q2),diff(x,q3)],
              [diff(y,q1),diff(y,q2),diff(y,q3)],
              [diff(z,q1),diff(z,q2),diff(z,q3)]);
```

(%o4) $J(q1,q2,q3):=\begin{pmatrix} \mathrm{diff}(x,q1) & \mathrm{diff}(x,q2) & \mathrm{diff}(x,q3) \\ \mathrm{diff}(y,q1) & \mathrm{diff}(y,q2) & \mathrm{diff}(y,q3) \\ \mathrm{diff}(z,q1) & \mathrm{diff}(z,q2) & \mathrm{diff}(z,q3) \end{pmatrix}$

(%i5) `J:J(q[1],q[2],q[3]);`

(%o5) $\begin{pmatrix} -q_3\cos(q_1) & 0 & -\sin(q_1) \\ -q_3\sin(q_1) & 0 & \cos(q_1) \\ 0 & 1 & 0 \end{pmatrix}$

```
(%i6) dJ:trigsimp(determinant(J));
```

(%o6) q_3

```
(%i9) Jq3:subst(q[3]=0,J);
```

(%o9) $\begin{pmatrix} 0 & 0 & -\sin(q_1) \\ 0 & 0 & \cos(q_1) \\ 0 & 1 & 0 \end{pmatrix}$

```
(%i10) nullspace(Jq3);
```

Proviso: notequal$(-\sin(q_1),0) \wedge$ notequal$(-\sin(q_1),0)$

(%o10) $\mathrm{span}\left(\begin{pmatrix} -\sin(q_1) \\ 0 \\ 0 \end{pmatrix}\right)$

Se $q_1 \neq 0$, le singolarità di velocità si hanno per $v \in \mathrm{Im}\left\{\begin{pmatrix} -\sin(q_1) \\ 0 \\ 0 \end{pmatrix}\right\}$.

Se $q_1 = 0$:

```
(%i11) Jq31:subst(q[1]=0,Jq3);
```

(%o11) $\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$

```
(%i12) nullspace(Jq31);
```

(%o12) $\mathrm{span}\left(\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}\right)$

$q_1 \neq 0$, le singolarità di velocità si hanno per $v \in \mathrm{Im}\left\{\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}\right\}$.

Singolarità di forza

```
(%i13) Jtr(q1,q2,q3):=-transpose(J(q1,q2,q3));
```

(%o13) $\mathrm{Jtr}(q1, q2, q3) := -\mathrm{transpose}(J(q1, q2, q3))$

```
(%i14) Jtrn:Jtr(q[1],q[2],q[3]);
```

(%o14) $\begin{pmatrix} q_3\cos(q_1) & q_3\sin(q_1) & 0 \\ 0 & 0 & -1 \\ \sin(q_1) & -\cos(q_1) & 0 \end{pmatrix}$

```
(%i15) dJtr:trigsimp(determinant(Jtrn));
```

(%o15) $-q_3$

```
(%i16) Jq3:subst(q[3]=0,Jtrn);
```

(%o16) $\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ \sin(q_1) & -\cos(q_1) & 0 \end{pmatrix}$

```
(%i18) nullspace(Jq3);
```

Proviso: notequal$(\cos(q_1), 0)$

(%o18) $\mathrm{span}\left(\begin{pmatrix} \cos(q_1) \\ \sin(q_1) \\ 0 \end{pmatrix}\right)$

(%i19)

$\mathrm{Ker}\{\mathrm{J}\} = \begin{pmatrix} \text{-cos } (q_1) \\ \text{sin } (q_1) \\ 0 \end{pmatrix} \rightarrow \forall q_1 \neq 0$

Non si possono applicare forze t.c. $\tau = \mathrm{Im}\{\begin{pmatrix} \text{-cos } (q_1) \\ \text{sin } (q_1) \\ 0 \end{pmatrix}\}$

Se q_1=0:

(%i19) `Jq1:subst(q[1]=0,Jq3);`

(%o19) $\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & -1 & 0 \end{pmatrix}$

(%i20) `nullspace(Jq1);`

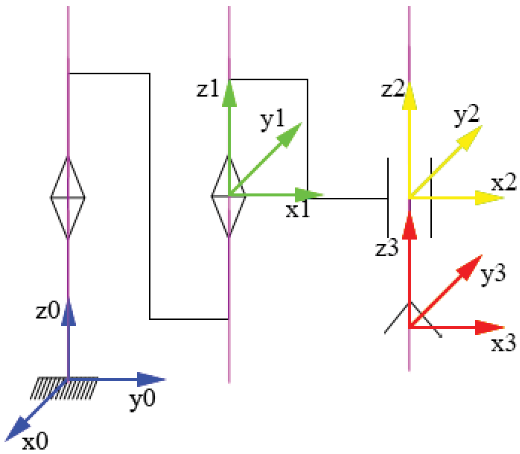
(%o20) $\mathrm{span}\left(\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}\right)$

(%i21)

Non si possono applicare forze t.c. $\tau = \mathrm{Im}\{\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}\}$

Cinematica diretta SCARA

***N.B.:* le grandezze diverse da quelle di giunto q_i sono L_i, D_i. Esse sono rispettivamente la distanza tra i sistemi di riferimento R_i e R_{i+1} nelle operazioni della matrice avvitamento $A_z(\theta, d)$ e $A_x(\alpha, a)$.**



	ϑ	d	α	a
1	q_1	L_1	0	D_1
2	q_2	0	0	D_2
3	0	q_3	0	0

Tabella 1.

Funzioni ausiliarie:

```
(%i1) inverseLaplace(SI,theta):=block([res],
                                 M:SI,
                                 MC:SI,
                                 for i:1 thru 3 do(
                                   for j:1 thru 3 do
                                      (
                                        aC:M[i,j],
                                        b:ilt(aC,s,theta),
                                        MC[i,j]:b
                                       )
                                    ),
                                 res:MC

                              )
```

(%o1) inverseLaplace(SI, ϑ) := **block** ([res], M: SI, MC: SI, **for** i **thru** 3 **do for** j **thru** 3 **do** (aC: $M_{i,j}$, b: ilt(aC, s, ϑ), MC$_{i,j}$: b), res: MC)

```
(%i2) rotLaplace(k,theta):=block([res],
                        S:ident(3),
                        I:ident(3),
                       for i:1 thru 3 do
                        (
                        for j:1 thru 3 do
                           (
                            if i=j
                               then S[i][j]:0
                            elseif j>i
                               then (
                             temp:(-1)^(j-i)*k[3-remainder(i+j,3)],
                                    S[i][j]:temp,
                                    S[j][i]:-temp

                                     )
                            )
                         ),
                        res:inverseLaplace(invert(s*I-S),theta)


                      )
```

(%o2) $\mathrm{rotLaplace}(k,\vartheta) := \textbf{block}\ ([\mathrm{res}], S\colon \mathrm{ident}(3), I\colon \mathrm{ident}(3),$ $\textbf{for}\ i\ \textbf{thru}\ 3\ \textbf{do}\ \textbf{for}\ j\ \textbf{thru}\ 3\ \textbf{do}\ \textbf{if}\ i=j\ \textbf{then}\ (S_i)_j\colon 0\ \textbf{elseif}\ j>i\ \textbf{then}\ (\mathrm{temp}\colon$ $(-1)^{j-i}\,k_{3-\mathrm{remainder}(i+j,3)}, (S_i)_j\colon \mathrm{temp}, (S_j)_i\colon -\mathrm{temp}), \mathrm{res}\colon \mathrm{inverseLaplace}(\mathrm{invert}(s\,I-S),\vartheta))$

```
(%i3) Av(v,theta,d):=block([res],
                        Trot:rotLaplace(v,theta),
                        row:matrix([0,0,0,1]),
                        Atemp:addcol(Trot,d*transpose(v)),
                        A:addrow(Atemp,row),
                        res:A
                        )
```

(%o3) $\mathrm{Av}(v,\vartheta,d) := \textbf{block}\ ([\mathrm{res}], \mathrm{Trot}\colon \mathrm{rotLaplace}(v,\vartheta), \mathrm{row}\colon \begin{pmatrix} 0 & 0 & 0 & 1 \end{pmatrix}, \mathrm{Atemp}\colon \mathrm{addcol}(\mathrm{Trot},$ $d\,\mathrm{transpose}(v)), A\colon \mathrm{addrow}(\mathrm{Atemp},\mathrm{row}), \mathrm{res}\colon A)$

```
(%i4) Q(theta,d,alpha,a):=block([res],
                            tempMat:Av([0,0,1],theta,d).Av([1,0,0],alpha,a),
                            Qtrasf:zeromatrix(4,4),
                            for i:1 thru 4 do
                        (
                        for j:1 thru 4 do
                           (
                            Qtrasf[i][j]:trigreduce(tempMat[i][j])
                                     )
                          ),

                            res:Qtrasf
                            )
```

(%o4) $Q(\vartheta,d,\alpha,a) := \textbf{block}\ ([\mathrm{res}], \mathrm{tempMat}\colon \mathrm{Av}([0,0,1],\vartheta,d)\cdot\mathrm{Av}([1,0,0],\alpha,a), \mathrm{Qtrasf}\colon$ $\mathrm{zeromatrix}(4,4), \textbf{for}\ i\ \textbf{thru}\ 4\ \textbf{do}\ \textbf{for}\ j\ \textbf{thru}\ 4\ \textbf{do}\ (\mathrm{Qtrasf}_i)_j\colon \mathrm{trigreduce}((\mathrm{tempMat}_i)_j), \mathrm{res}\colon \mathrm{Qtrasf})$

```
(%i5) let(sin(q[1]), s[1]);
```

(%o5) $\sin(q_1) \longrightarrow s_1$

```
(%i6) let(sin(q[2]), s[2]);
```

(%o6) $\sin\left(q_2\right) \longrightarrow s_2$

```
(%i7) let(cos(q[1]),c[1]);
```

(%o7) $\cos\left(q_1\right) \longrightarrow c_1$

```
(%i8) let(cos(q[2]),c[2]);
```

(%o8) $\cos\left(q_2\right) \longrightarrow c_2$

```
(%i9) let(sin(q[1]+q[2]), s[12]);
```

(%o9) $\sin\left(q_2+q_1\right) \longrightarrow s_{12}$

```
(%i10) let(cos(q[1]+q[2]),c[12]);
```

(%o10) $\cos\left(q_2+q_1\right) \longrightarrow c_{12}$

```
(%i11)
(%i11)
```

Cinematica diretta:

```
(%i11) Q[scara](q1,q2,q3,L1,D1,D2):=trigsimp(trigrat(trigreduce(trigexpand(
                         Q(q1,L1,0,D1).
                          Q(q2,0,0,D2).
                          Q(0,q3,0,0)
                          ))));
```

(%o11) $Q_{\mathrm{scara}}(q1, q2, q3, L1, D1, D2) := \mathrm{trigsimp}(\mathrm{trigrat}(\mathrm{trigreduce}(\mathrm{trigexpand}(Q(q1, L1, 0, D1) \cdot Q(q2, 0, 0, D2) \cdot Q(0, q3, 0, 0)))))$

```
(%i12) Qscara:Q[scara](q[1],q[2],q[3],L[1],D[1],D[2]);
```

(%o12) $$\begin{pmatrix} \cos\left(q_2+q_1\right) & -\sin\left(q_2+q_1\right) & 0 & D_2\cos\left(q_2+q_1\right)+D_1\cos\left(q_1\right) \\ \sin\left(q_2+q_1\right) & \cos\left(q_2+q_1\right) & 0 & D_2\sin\left(q_2+q_1\right)+D_1\sin\left(q_1\right) \\ 0 & 0 & 1 & q_3+L_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
(%i13) letsimp(Qscara);
```

(%o13) $$\begin{pmatrix} c_{12} & -s_{12} & 0 & D_2\, c_{12}+D_1\, c_1 \\ s_{12} & c_{12} & 0 & D_2\, s_{12}+D_1\, s_1 \\ 0 & 0 & 1 & q_3+L_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
(%i14)
```

Cinematica inversa SCARA

Al fine di risolvere il problema di cinematica inversa del robot scara occorre risolevere il problema di posizione ed orientamente inverso.Inizialmente individuare lo spazio di lavoro, le soluzioni generiche, singolari ed infine le variabili di giunto q_i ed in seguito determinare l'orientamento del robot.

Dalla cinematica diretta del robot SCARA sappiamo che:

$$Q_{\mathrm{SCARA}} = \begin{pmatrix} c_{12} & -s_{12} & 0 & D_2\, c_{12}+D_1\, c_1 \\ s_{12} & c_{12} & 0 & D_2\, s_{12}+D_1\, s_1 \\ 0 & 0 & 1 & q_3+L_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Cinematica inversa di posizione

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} D_2\, c_{12} + D_1\, c_1 \\ D_2\, s_{12} + D_1\, s_1 \\ q_3 + L_1 \end{pmatrix}$$

Dato chew L_1 è nota è possibile ottenere la varibiale di giungo q_3:

$$q_3 = z$$

e, riscrivendo le equazioni rimanenti, si ottiene il seguente sistema:

$$\begin{cases} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} D_2 \cos(q_2 + q_1) + D_1 \cos(q_1) \\ D_2 \sin(q_2 + q_1) + D_1 \sin(q_1) \\ 0 \end{pmatrix} \\ q_3 = z \end{cases}$$

Occorre ora calcolare le variabili di giunto q_1, q_2. Il problema è equivalente al problema di cinematica di posizione inversa del 2DOF planare. Infatti:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} D_2 \cos(q_2 + q_1) + D_1 \cos(q_1) \\ D_2 \sin(q_2 + q_1) + D_1 \sin(q_1) \\ 0 \end{pmatrix}$$

$$\begin{pmatrix} x \\ y \end{pmatrix} = R(q_1 + q_2) \begin{pmatrix} D_2 \\ 0 \end{pmatrix} + R(q_1) \begin{pmatrix} D_1 \\ 0 \end{pmatrix}$$

Poiché $R(q_1 + q_2) = R(q_1)\, R(q_2)$, è possibile mettere in evidenza $R(q_1)$:

$$\begin{pmatrix} x \\ y \end{pmatrix} = R(q_1) \left\{ R(q_2) \begin{pmatrix} D_2 \\ 0 \end{pmatrix} + \begin{pmatrix} D_1 \\ 0 \end{pmatrix} \right\}$$

La matrice di rotazione $R(q_1)$ ha non varia la norma del vettore $\begin{pmatrix} \mathrm{x} \\ \mathrm{y} \end{pmatrix}$ e $R(q_2) \begin{pmatrix} D_2 \\ 0 \end{pmatrix} + \begin{pmatrix} D_1 \\ 0 \end{pmatrix}$. Quindi possiamo imporre che abbiano la stessa norma. In particolare:

$$\left|\left| \begin{pmatrix} x \\ y \end{pmatrix} \right|\right|_2 = \left|\left| R(q_2) \begin{pmatrix} D_2 \\ 0 \end{pmatrix} + \begin{pmatrix} D_1 \\ 0 \end{pmatrix} \right|\right|_2$$

$$\begin{pmatrix} D_2 & 0 \end{pmatrix} R(q_2)^T R(q_2) \begin{pmatrix} D_2 \\ 0 \end{pmatrix} + \begin{pmatrix} D_1 & 0 \end{pmatrix} \begin{pmatrix} D_1 \\ 0 \end{pmatrix} + 2 \begin{pmatrix} D_1 & 0 \end{pmatrix} R(q_2) \begin{pmatrix} D_2 \\ 0 \end{pmatrix}$$

$$x^2 + y^2 = D_2^2 + D_1^2 + 2\, D_1\, D_2 \cos(q_2)$$

$$\cos(q_2) = \frac{x^2 + y^2 - D_2^2 - D_1^2}{2\, D_1\, D_2}$$

Poiché $-1 \leqslant \cos(q_2) \leqslant 1$, otteniamo infine lo spazio operativo del robot SCARA:

$$-1 \leqslant \frac{x^2 + y^2 - D_2^2 - D_1^2}{2\, D_1\, D_2} \leqslant 1$$

$$D_2^2 + D_1^2 - 2\, D_1\, D_2 \leqslant x^2 + y^2 \leqslant 2\, D_1\, D_2 + D_2^2 + D_1^2$$

Le disequazioni delimitano due circonferenze di raggio $D_2 + D_1$ e $D_2 - D_1$, in cui la regione valida presa inconsiderazione è quella regione di spazio compresa tra le curve. Infine, considerando, a differenza del robot 2DOF planare, anche la variabile q_3 si delimita un cilindro cavo di altezza q_3 e di raggio interno $D_2 - D_1$ e di raggio esterno $D_2 + D_1$.

A questo punto è possibile determinare l'espressione della variabile di giunto q_2:

$$\cos(q_2) = \frac{x^2 + y^2 - D_2^2 - D_1^2}{2\, D_1\, D_2}$$

$$\sin(q_2) = \pm\sqrt{1 - \cos(q_2)^2} = \pm\sqrt{1 - \frac{x^2 + y^2 - D_2^2 - D_1^2}{2\, D_1\, D_2}}$$

In particolare, si hanno due soluzioni generiche se $\left|\frac{x^2 + y^2 - D_2^2 - D_1^2}{2\, D_1\, D_2}\right| \neq 1$. Nel caso in cui $\left|\frac{x^2 + y^2 - D_2^2 - D_1^2}{2\, D_1\, D_2}\right| = 1$, si ha una soluzione singolare.

A questo punto la quantità $R(q_2)\begin{pmatrix} D_2 \\ 0 \end{pmatrix} + \begin{pmatrix} D_1 \\ 0 \end{pmatrix}$ è nota:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \cos(q_1) & -\sin(q_1) \\ \sin(q_1) & \cos(q_1) \end{pmatrix} \begin{pmatrix} D_2 \cos(q_2) + D_1 \\ \sin(q_2)\, D_2 \end{pmatrix}$$

$$\begin{pmatrix} \cos(q_1) \\ \sin(q_1) \end{pmatrix} = \frac{1}{(D_2 \cos(q_2) + D_1)^2 + D_2^2 \sin(q_2)^2} \begin{pmatrix} D_2 \cos(q_2) + D_1 & \sin(q_2)\, D_2 \\ -\sin(q_2)\, D_2 & D_2 \cos(q_2) + D_1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

$$= \frac{1}{D_2^2 \cos(q_2)^2 + D_1^2 + 2\, D_2 D_1 \cos(q_2) + D_1 + D_2^2 \sin(q_2)^2} \begin{pmatrix} (D_2 \cos(q_2) + D_1)\, x + \sin(q_2)\, D_2\, y \\ -\sin(q_2)\, D_2\, x + (D_2 \cos(q_2) + D_1)\, y \end{pmatrix}$$

Da cui si ottiene la variabile di giunto q_1;

$$q_1 = \operatorname{atan2}\left(\frac{-\sin(q_2)\, D_2\, x + (D_2 \cos(q_2) + D_1)\, y}{D_2^2 \cos(q_2)^2 + D_1^2 + D_2^2 \sin(q_2)^2}, \frac{(D_2 \cos(q_2) + D_1)\, x + \sin(q_2)\, D_2\, y}{D_2^2 \cos(q_2)^2 + D_1^2 + D_2^2 \sin(q_2)^2} \right)$$

Poiché la quantità $D_2^2 \cos(q_2)^2 + D_1^2 + D_2^2 \sin(q_2)^2 > 0$ è possibile semplificara all'interno della funzione atan2:

$$q_1 = \operatorname{atan2}(\sin(q_2)\, D_2\, x - (D_2 \cos(q_2) + D_1)\, y, (D_2 \cos(q_2) + D_1)\, x - \sin(q_2)\, D_2\, y)$$

Il problema della cinematica inversa ed, in particolare il problema di posizione inverso, per il robot SCARA è risolto.

Orientamento inverso

La risoluzione del problema di orientamento inverso si basa sulla scelta di una terna di Eulero o nautica in condizione non singolari, se possibile.

$$R_{\text{SCARA}} = \begin{pmatrix} \cos(q_2+q_1) & -\sin(q_2+q_1) & 0 \\ \sin(q_2+q_1) & \cos(q_2+q_1) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$R_{\text{zyx}} = \begin{pmatrix} c_y c_z & \cdots\cdots\cdots & \\ c_y s_z & \cdots\cdots\cdots & \\ -s_y & s_x c_y & c_x c_y \end{pmatrix}$$

Poiché l'elemento $-s_y = 0 \neq \pm 1$ è possibile risolvere il problema di orientamento inverso con la terna nautica zyx. In particolare:

$$s_y = 0 \longrightarrow c_y = \pm 1 \longrightarrow \phi_y = \operatorname{atan2}(s_y, c_y) \longrightarrow \phi_y = \begin{cases} 0 \\ \pi \end{cases}$$

$$\begin{cases} s_x c_y = 0 \\ c_x c_y = 1 \end{cases} \longrightarrow \begin{cases} s_x = 0 \\ c_x = \pm 1 \end{cases} \longrightarrow \phi_x = \operatorname{atan2}(s_x, \pm c_x) \longrightarrow \phi_x = \begin{cases} 0 \\ \pi \end{cases}$$

$$\begin{cases} c_y c_z = \cos(q_2+q_1) \\ c_y s_z = \sin(q_2+q_1) \end{cases} \longrightarrow \begin{cases} c_z = \pm\cos(q_1+q_2) \\ s_z = \pm\sin(q_1+q_2) \end{cases} \longrightarrow \phi_z = \operatorname{atan2}(\pm\sin(q_1+q_2), \pm\cos(q_1+q_2))$$

$$\phi_z = \begin{cases} q_1+q_2 \\ q_1+q_2+\pi \end{cases}$$

Riassumendo:

$$\begin{pmatrix} 0 \\ 0 \\ q_1+q_2 \end{pmatrix}, \begin{pmatrix} \pi \\ \pi \\ q_1+q_2+\pi \end{pmatrix}$$

In alternativa, tramite una scelta di una terna di Eulero:

$$R_{\text{xzx}} = \begin{pmatrix} \cos(\beta) & -\sin(\beta)\cos(\gamma) & \sin(\beta)\sin(\gamma) \\ \cos(\alpha)\sin(\beta) & \cdots & \cdots \\ \sin(\alpha)\sin(\beta) & \cdots & \cdots \end{pmatrix}$$

$$R_{\text{zyx}} = \begin{pmatrix} c_y c_z & \cdots\cdots\cdots & \\ c_y s_z & \cdots\cdots\cdots & \\ -s_y & s_x c_y & c_x c_y \end{pmatrix}$$

$$\cos(\beta) = \cos(q_1+q_2) \;\; \rightarrow \;\; \sin(\beta) = \pm\sqrt{1-\cos(q_1+q_2)^2} = \pm\sin(q_1+q_2)$$

La configurazione scelta è singolare se:

$$1-\cos(q_1+q_2)^2 = 0 \rightarrow \cos(q_1+q_2) = \pm 1 \rightarrow q_1+q_2 = \begin{cases} 0 \\ \pi \end{cases}$$

Supponiamo quindi che $q_1+q_2 \neq \left\{ \begin{smallmatrix} 0 \\ \pi \end{smallmatrix} \right.$:

$$\beta = \mathrm{atan2}\left(\sqrt{1-\cos(q_1+q_2)^2}, \cos(q_1+q_2) \right)$$

$$\beta = \left\{ \begin{array}{l} q_1+q_2 \\ q_1+q_2+\pi \end{array} \right.$$

$$\left\{ \begin{array}{l} \cos(\alpha)\sin(\beta) = \sin(q_2+q_1) \\ \sin(\alpha)\sin(\beta) = 0 \end{array} \right. \rightarrow \left\{ \begin{array}{l} \cos(\alpha) = \pm 1 \\ \sin(\alpha) = 0 \end{array} \right.$$

$$\alpha = \mathrm{atan2}(0, \pm 1) = \left\{ \begin{array}{l} 0 \\ \pi \end{array} \right.$$

$$\left\{ \begin{array}{l} -\sin(\beta)\cos(\gamma) = -\sin(q_2+q_1) \\ \sin(\beta)\sin(\gamma) = 0 \end{array} \right. \rightarrow \left\{ \begin{array}{l} \cos(\gamma) = \pm 1 \\ \sin(\gamma) = 0 \end{array} \right.$$

$$\gamma = \mathrm{atan2}(0, \pm 1) = \left\{ \begin{array}{l} 0 \\ \pi \end{array} \right.$$

Riassumendo, sotto l'ipotesi che $q_1+q_2 \neq \left\{ \begin{smallmatrix} 0 \\ \pi \end{smallmatrix} \right.$:

$$\begin{pmatrix} 0 \\ 0 \\ q_1+q_2 \end{pmatrix}, \begin{pmatrix} \pi \\ \pi \\ q_1+q_2+\pi \end{pmatrix}$$

```
(%i14) isRotation(M):=block([MC,res],
                            I:ident(3),
                            MC:ident(3),
                            for i:1 thru 3 do
                            (
                            for j:1 thru 3 do
                               (
                                  MC[i][j]:M[i][j]
                                  )
                               ),
                            MMT:trigsimp(expand(MC.transpose(MC))),
                            detM:trigsimp(expand(determinant(MC))),

                            if MMT=I  and detM=1
                               then(

                                    return(res:1)
                                    )

                            else(

                                  res: "R is not rotation matrix"
                                  )
                            )
```

(%o14) isRotation(M) := **block** ([MC, res], I: ident(3), MC: ident(3),

for i **thru** 3 **do for** j **thru** 3 **do** $(\mathrm{MC}_i)_j$: $(M_i)_j$, MMT: trigsimp(expand(MC · transpose(MC))), detM: trigsimp(expand(determinant(MC))), **if** MMT $= I \wedge$ detM $= 1$ **then** return(res: 1) **else** res: R is not rotation matrix)

(%i80)
```
calculate(x,y,L1,L2,z):=block([q2plus,q2minus,q1,res],
                               c2:(x^(2)+y^(2)-L1^(2)-L2^(2))/(2*L1*L2),
                               s2:sqrt(1-c2^2),

                               c1Num:combine(expand((L2*c2+L1)*x+s2*L2*y)),
                               s1Num:combine(expand(-s2*L2*x+(L2*c2+L1)*y)),

    q1Den:L2^(2)*c2^(2)+L1^(2)+L2^(2)*s2^(2)+2*L1*L2*c2,
                        if abs(c2)=1 then print("La soluzione è singolare")
                               elseif q1Den>0 then(
                                   print("La soluzione non è singolare"),
                                   q1:atan2(ratsimp(s1Num),ratsimp(c1Num)),
                                   q2alto:atan2(ratsimp(s2),ratsimp(c2)),
                                   q2basso:atan2(ratsimp(-s2),ratsimp(c2)),
                                   res:[[q2alto,q1,z],[q2basso,q1,z]])
                               else (
                                   q1:atan2(s1Num/q1Den,c1Num/q1Den),
                                   q2alto:atan2(ratsimp(s2),ratsimp(c2)),
                                   q2basso:atan2(ratsimp(-s2),ratsimp(c2)),
                                   res:[[q2alto,q1],[q2basso,q1]]
                                   )



                               )
```

(%o80) calculate$(x, y, L1, L2, z) :=$ **block** $\Big($[q2plus, q2minus, q1, res], $c2$: $\dfrac{x^2+y^2-L1^2-L2^2}{2\,L1\,L2}$, $s2$: $\sqrt{1-c2^2}$, c1Num: combine(expand($(L2\,c2+L1)\,x+s2\,L2\,y$)), s1Num: combine(expand($(-s2)\ L2\ x + (L2\ c2 + L1)\ y$)), q1Den: $L2^2\ c2^2 + L1^2 + L2^2\ s2^2 + 2\ L1\ L2\ c2$, **if** $|c2| = 1$ **then** print(La soluzione è singolare) **elseif** q1Den > 0 **then** (print(La soluzione non è singolare), q1: atan2(ratsimp(s1Num), ratsimp(c1Num)), q2alto: atan2(ratsimp($s2$), ratsimp($c2$)), q2basso: atan2(ratsimp($-s2$), ratsimp($c2$)), res: [[q2alto, $q1$, z], [q2basso, $q1$, z]]) **else** $\Big(q1$: atan2$\left(\dfrac{s1\mathrm{Num}}{q1\mathrm{Den}}, \dfrac{c1\mathrm{Num}}{q1\mathrm{Den}}\right)$, q2alto: atan2(ratsimp($s2$), ratsimp($c2$)), q2basso: atan2(ratsimp($-s2$), ratsimp($c2$)), res: [[q2alto, $q1$], [q2basso, $q1$]]$\Big)\Big)$

```
(%i96) orientation(Qdiretta):=block([sx,cx,sy,cy,phiy1,phiy2,phiz1,phiz2,phix1,
       phix2,sz,sxfirst,second,res],


                                        sy:Qdiretta[3][1],
                                        cy:sqrt(1-sy^2),
                                        phiy1:atan2(sy,cy),
                                        phiy2:atan2(sy,-cy),
                                        sx:Qdiretta[3][2],
                                        cx:Qdiretta[3][3],
                                        phix1:atan2(sx,cx),
                                        phix2:atan2(sx,-cx),
                                        cz:Qdiretta[1][1],
                                        sz:Qdiretta[2][1],
                                        phiz1:atan2(sz,cz),
                                        phiz2:atan2(-sz,-cz),
                                        first:[phix1,phiy1,phiz1],
                                        second:[phix2,phiy2,phiz2],

                                        res:[first,second]

       );
```

(%o96) orientation(Qdiretta) := **block** ([sx, cx, sy, cy, phiy1, phiy2, phiz1, phiz2, phix1, phix2, sz, sxfirst, second, res], sy: $(\mathrm{Qdiretta}_3)_1$, cy: $\sqrt{1-\mathrm{sy}^2}$, phiy1: atan2(sy, cy), phiy2: atan2(sy, −cy), sx: $(\mathrm{Qdiretta}_3)_2$, cx: $(\mathrm{Qdiretta}_3)_3$, phix1: atan2(sx, cx), phix2: atan2(sx, −cx), cz: $(\mathrm{Qdiretta}_1)_1$, sz: $(\mathrm{Qdiretta}_2)_1$, phiz1: atan2(sz, cz), phiz2: atan2(−sz, −cz), first: [phix1, phiy1, phiz1], second: [phix2, phiy2, phiz2], res: [first, second])

```
(%i97) invScara(Qdiretta,D1,D2):=block([pos1,pos2,orien1,orien2,res],
                                       rotation:isRotation(Qdiretta),
                                       if rotation=1 then(
                                          x:Qdiretta[1][4],
                                          y:Qdiretta[2][4],
                                          z:Qdiretta[3][4],
                                          circleInt:D1^(2)+D2^(2)-2*D1*D2,
                                          circleEst:D1^(2)+D2^(2)+2*D1*D2,
                                        print("Il punto x,y è nello spazio di lavoro"),
                                        pos:calculate(x,y,D1,D2,z),
                                        pos1:pos[1],
                                        pos2:pos[2],
                                        orien:orientation(Qdiretta),
                                        orien1:orien[1],
                                        orien2:orien[2],
                                        res:[pos1,pos2,orien1,orien2]

                                       )
                                       else res:rotation
                                       );
```

(%o97) invScara(Qdiretta, $D1$, $D2$) := **block** ([pos1, pos2, orien1, orien2, res], rotation: isRotation(Qdiretta), **if** rotation = 1 **then** (x: $(\mathrm{Qdiretta}_1)_4$, y: $(\mathrm{Qdiretta}_2)_4$, z: $(\mathrm{Qdiretta}_3)_4$, circleInt: $D1^2 + D2^2 + (-2)\, D1\, D2$, circleEst: $D1^2 + D2^2 + 2\, D1\, D2$, print(Il punto x,y è nello spazio di lavoro), pos: calculate($x, y, D1, D2, z$), pos1: pos_1, pos2: pos_2, orien: orientation(Qdiretta), orien1: orien_1, orien2: orien_2, res: [pos1, pos2, orien1, orien2]) **else** res: rotation)

(%i98) `Qdiretta:Q[scara](%pi/3,0,10,5,15,5)`

(%o98) $$\begin{pmatrix} \frac{1}{2} & -\frac{\sqrt{3}}{2} & 0 & 10 \\ \frac{\sqrt{3}}{2} & \frac{1}{2} & 0 & \frac{5\,3^{\frac{3}{2}}}{2}+\frac{5\sqrt{3}}{2} \\ 0 & 0 & 1 & 15 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

(%i99) `invScara(Qdiretta,15,5);`

Il punto x,y è nello spazio di lavoro

La soluzione non è singolare

(%o99) $$\left[\left[0,\frac{\pi}{3},15\right],\left[0,\frac{\pi}{3},15\right],\left[0,0,\frac{\pi}{3}\right],\left[\pi,\pi,-\frac{2\pi}{3}\right]\right]$$

(%i100)

Singolarià

(%i1) `x:D[2]*cos (q[2]+q[1])+D[1]*cos (q[1]);`

(%o1) $D_2\cos(q_2+q_1)+D_1\cos(q_1)$

(%i2) `y:D[2]*sin (q[2]+q[1])+D[1]*sin (q[1]);`

(%o2) $D_2\sin(q_2+q_1)+D_1\sin(q_1)$

(%i3) `z:q[3]+L[1];`

(%o3) q_3+L_1

(%i4)
```
J:matrix([diff(x,q[1]),diff(x,q[2]),diff(x,q[3])],
         [diff(y,q[1]),diff(y,q[2]),diff(y,q[3])],
         [diff(z,q[1]),diff(z,q[2]),diff(z,q[3])]);
```

(%o4) $$\begin{pmatrix} -D_2\sin(q_2+q_1)-D_1\sin(q_1) & -D_2\sin(q_2+q_1) & 0 \\ D_2\cos(q_2+q_1)+D_1\cos(q_1) & D_2\cos(q_2+q_1) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

(%i5) `dJ:trigreduce(expand(determinant(J)));`

(%o5) $D_1\,D_2\sin(q_2)$

Singolarità se $\det(J)=0 \rightarrow \sin(q_2)=0 \longrightarrow q_2=\begin{cases} 0 \\ \pi \end{cases}$

(%i6) `Jq:trigreduce(subst(q[2]=0,J));`

(%o6) $$\begin{pmatrix} -D_2\sin(q_1)-D_1\sin(q_1) & -D_2\sin(q_1) & 0 \\ D_2\cos(q_1)+D_1\cos(q_1) & D_2\cos(q_1) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

(%i7) `nullspace(Jq);`

Proviso: notequal$(-D_2\sin(q_1),0)\wedge$notequal$(-D_2\sin(q_1),0)$

(%o7) $$\mathrm{span}\left(\begin{pmatrix} -D_2\sin(q_1) \\ (D_2+D_1)\sin(q_1) \\ 0 \end{pmatrix}\right)$$

Se $q_1\neq 0$, le singolarità di velocità si hanno per $v\in \mathrm{Im}\left\{\begin{pmatrix} -D_2\sin(q_1) \\ (D_2+D_1)\sin(q_1) \\ 0 \end{pmatrix}\right\}$.

Se $q_1=0$:

(%i8) `Jq1:subst(q[1]=0,Jq);`

(%o8) $$\begin{pmatrix} 0 & 0 & 0 \\ D_2+D_1 & D_2 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

(%i9) `nullspace(Jq1);`

Proviso: notequal$(D_2,0) \wedge$ notequal$(D_2,0)$

(%o9) $$\mathrm{span}\left(\begin{pmatrix} D_2 \\ -D_2-D_1 \\ 0 \end{pmatrix}\right)$$

Se $q_1 \neq 0$ le singolarità di velocità per $v \in \Im m\left\{\begin{pmatrix} D_2 \\ -D_2-D_1 \\ 0 \end{pmatrix}\right\}$

(%i10) `Jq1:subst(q[1]=0,Jq);`

(%o10) $$\begin{pmatrix} 0 & 0 & 0 \\ D_2+D_1 & D_2 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

(%i11) `nullspace(Jq1);`

Proviso: notequal$(D_2,0) \wedge$ notequal$(D_2,0)$

(%o11) $$\mathrm{span}\left(\begin{pmatrix} D_2 \\ -D_2-D_1 \\ 0 \end{pmatrix}\right)$$

Se $q_1=0$ le singolarità di velocita si hanno per $v \in \mathrm{Im}\left\{\begin{pmatrix} -D_2 \\ D_2-D_1 \\ 0 \end{pmatrix}\right\}$.

Se $q_2=\pi$:

(%i12) `Jq2:subst(q[2]=%pi,Jq);`

(%o12) $$\begin{pmatrix} -D_2\sin(q_1)-D_1\sin(q_1) & -D_2\sin(q_1) & 0 \\ D_2\cos(q_1)+D_1\cos(q_1) & D_2\cos(q_1) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

(%i13) `nullspace(Jq2);`

Proviso: notequal$(-D_2\sin(q_1),0) \wedge$ notequal$(-D_2\sin(q_1),0)$

(%o13) $$\mathrm{span}\left(\begin{pmatrix} -D_2\sin(q_1) \\ (D_2+D_1)\sin(q_1) \\ 0 \end{pmatrix}\right)$$

(%i14)

Singolarità di Forza

(%i12) `J:-transpose(J);`

(%o12) $$\begin{pmatrix} D_2\sin(q_2+q_1)+D_1\sin(q_1) & -D_2\cos(q_2+q_1)-D_1\cos(q_1) & 0 \\ D_2\sin(q_2+q_1) & -D_2\cos(q_2+q_1) & 0 \\ 0 & 0 & -1 \end{pmatrix}$$

(%i13) `dJ:trigreduce(expand(determinant(J)));`

(%o13) $-D_1\,D_2\sin(q_2)$

```
(%i14) Jq:subst(q[2]=0,J);
```

$$\text{(\%o14)}\quad \begin{pmatrix} D_2\sin(q_1)+D_1\sin(q_1) & -D_2\cos(q_1)-D_1\cos(q_1) & 0 \\ D_2\sin(q_1) & -D_2\cos(q_1) & 0 \\ 0 & 0 & -1 \end{pmatrix}$$

```
(%i15) nullspace(Jq);
```

Proviso: notequal$((D_2+D_1)\sin(q_1),0)\wedge$notequal$((-D_2-D_1)\sin(q_1),0)$

$$\text{(\%o15)}\quad \mathrm{span}\left(\begin{pmatrix} (-D_2-D_1)\cos(q_1) \\ (-D_2-D_1)\sin(q_1) \\ 0 \end{pmatrix}\right)$$

Se $q_1 \neq 0$ le singolarità di forza per $\tau \in \Im m\left\{\begin{pmatrix} (-D_2-D_1)\cos(q_1) \\ (-D_2-D_1)\sin(q_1) \\ 0 \end{pmatrix}\right\}$

Se $q_1 = 0$:

```
(%i16) Jq1:subst(q[1]=0,Jq);
```

$$\text{(\%o16)}\quad \begin{pmatrix} 0 & -D_2-D_1 & 0 \\ 0 & -D_2 & 0 \\ 0 & 0 & -1 \end{pmatrix}$$

```
(%i17) nullspace(Jq1);
```

Proviso: notequal$(-D_2-D_1,0)\wedge$notequal$(D_2+D_1,0)$

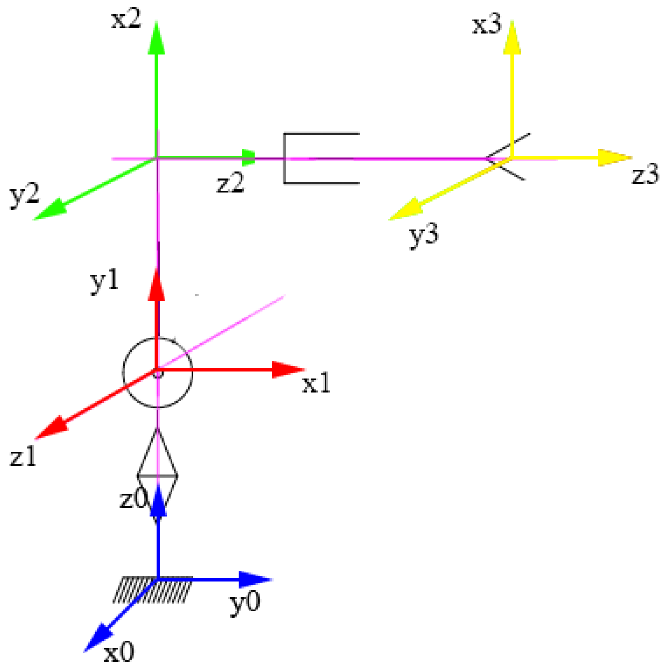
$$\text{(\%o17)}\quad \mathrm{span}\left(\begin{pmatrix} D_2+D_1 \\ 0 \\ 0 \end{pmatrix}\right)$$

```
(%i18)
```

le singolarità di forza per $\tau \in \Im m\left\{\begin{pmatrix} D_2+D_1 \\ 0 \\ 0 \end{pmatrix}\right\}$

Cinematica diretta Robot sferico I tipo

N.B.: **le grandezze diverse da quelle di giunto q_i sono L_i, D_i. Esse sono rispettivamente la distanza tra i sistemi di riferimento R_i e R_{i+1} nelle operazioni della matrice avvitamento $A_z(\theta, d)$ e $A_x(\alpha, a)$.**



	ϑ	d	α	a
1	q_1	L_1	$\frac{\pi}{2}$	0
2	q_2	0	$\frac{\pi}{2}$	L_2
3	0	q_3	0	0

Tabella 1.

Funzioni ausiliarie:

```
(%i1) inverseLaplace(SI,theta):=block([res],
                            M:SI,
                            MC:SI,
                            for i:1 thru 3 do(
                              for j:1 thru 3 do
                                (
                                  aC:M[i,j],
                                  b:ilt(aC,s,theta),
                                  MC[i,j]:b
                                 )
                              ),
                            res:MC

                          )
```

(%o1) inverseLaplace(SI, ϑ) := **block** ([res], M: SI, MC: SI, **for** i **thru** 3 **do for** j **thru** 3 **do** (aC:

$M_{i,j}, b$: ilt(aC, s, ϑ), MC$_{i,j}$: b), res: MC)

(%i2)
```
rotLaplace(k,theta):=block([res],
                        S:ident(3),
                        I:ident(3),
                      for i:1 thru 3 do
                        (
                        for j:1 thru 3 do
                           (
                             if i=j
                                then S[i][j]:0
                             elseif j>i
                                then (
                               temp:(-1)^(j-i)*k[3-remainder(i+j,3)],
                                      S[i][j]:temp,
                                      S[j][i]:-temp

                                       )
                             )
                         ),
                        res:inverseLaplace(invert(s*I-S),theta)


                       )
```

(%o2) $\mathrm{rotLaplace}(k,\vartheta) :=$ **block** ([res], S: ident(3), I: ident(3), **for** i **thru** 3 **do for** j **thru** 3 **do if** $i=j$ **then** $(S_i)_j$: 0 **elseif** $j>i$ **then** (temp: $(-1)^{j-i}\,k_{3-\mathrm{remainder}(i+j,3)}$, $(S_i)_j$: temp, $(S_j)_i$: $-$temp), res: inverseLaplace(invert($s\,I-S$), ϑ))

(%i3)
```
Av(v,theta,d):=block([res],
                        Trot:rotLaplace(v,theta),
                        row:matrix([0,0,0,1]),
                        Atemp:addcol(Trot,d*transpose(v)),
                        A:addrow(Atemp,row),
                        res:A
                        )
```

(%o3) Av(v, ϑ, d) := **block** ([res], Trot: rotLaplace(v, ϑ), row: $\begin{pmatrix} 0 & 0 & 0 & 1 \end{pmatrix}$, Atemp: addcol(Trot, d transpose(v)), A: addrow(Atemp, row), res: A)

(%i4)
```
Q(theta,d,alpha,a):=block([res],
                             tempMat:Av([0,0,1],theta,d).Av([1,0,0],alpha,a),
                             Qtrasf:zeromatrix(4,4),
                             for i:1 thru 4 do
                        (
                        for j:1 thru 4 do
                           (
                             Qtrasf[i][j]:trigreduce(tempMat[i][j])
                                       )
                           ),

                             res:Qtrasf
                             )
```

(%o4) $Q(\vartheta,d,\alpha,a) :=$ **block** ([res], tempMat: Av([0, 0, 1], ϑ, d) $\cdot$ Av([1, 0, 0], α, a), Qtrasf:

zeromatrix$(4, 4)$, **for** i **thru** 4 **do for** j **thru** 4 **do** $(\text{Qtrasf}_i)_j$: trigreduce$((\text{tempMat}_i)_j)$, res: Qtrasf)

```
(%i5) let(sin(q[1]), s[1]);
```

(%o5) $\sin(q_1) \longrightarrow s_1$

```
(%i6) let(sin(q[2]), s[2]);
```

(%o6) $\sin(q_2) \longrightarrow s_2$

```
(%i7) let(cos(q[1]),c[1]);
```

(%o7) $\cos(q_1) \longrightarrow c_1$

```
(%i8) let(cos(q[2]),c[2]);
```

(%o8) $\cos(q_2) \longrightarrow c_2$

```
(%i9) let(sin(q[1]+q[2]), s[12]);
```

(%o9) $\sin(q_2 + q_1) \longrightarrow s_{12}$

```
(%i10) let(cos(q[1]+q[2]),c[12]);
```

(%o10) $\cos(q_2 + q_1) \longrightarrow c_{12}$

(%i11)

Cinematica diretta:

```
(%i11) Q[sfericoRRP](q1,q2,q3,L1,L2):=
                    Q(q1,L1,%pi/2,0).
                    Q(q2,0,%pi/2,L2).
                    Q(0,q3,0,0)
                    ;
```

(%o11) $Q_{\text{sfericoRRP}}(q1, q2, q3, L1, L2) := Q\left(q1, L1, \frac{\pi}{2}, 0\right) \cdot Q\left(q2, 0, \frac{\pi}{2}, L2\right) \cdot Q(0, q3, 0, 0)$

```
(%i12) QsfericoRRP:Q[sfericoRRP](q[1],q[2],q[3],L[1],L[2]);
```

(%o12) $$\begin{pmatrix} \cos(q_1)\cos(q_2) & \sin(q_1) & \cos(q_1)\sin(q_2) & \cos(q_1)\,(q_3\sin(q_2) + L_2\cos(q_2)) \\ \sin(q_1)\cos(q_2) & -\cos(q_1) & \sin(q_1)\sin(q_2) & \sin(q_1)\,(q_3\sin(q_2) + L_2\cos(q_2)) \\ \sin(q_2) & 0 & -\cos(q_2) & L_2\sin(q_2) - q_3\cos(q_2) + L_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
(%i13) letsimp(QsfericoRRP);
```

(%o13) $$\begin{pmatrix} c_1\,c_2 & s_1 & c_1\,s_2 & c_1\,s_2\,q_3 + c_1\,L_2\,c_2 \\ s_1\,c_2 & -c_1 & s_1\,s_2 & s_1\,s_2\,q_3 + s_1\,L_2\,c_2 \\ s_2 & 0 & -c_2 & -c_2\,q_3 + L_2\,s_2 + L_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

(%i14)

Cinematica inversa robot sferico I tipo

Al fine di risolvere il problema di cinematica inversa del robot sferico di I tipo occorre risolevere il problema di posizione ed orientamente inverso.Inizialmente individuare lo spazio di lavoro, le soluzioni generiche, singolari ed infine le variabili di giunto q_i ed in seguito determinare l'orientamento del robot.

Dalla cinematica diretta del robot sferico di I tipo, sappiamo che:

$$Q_{\text{sferico}} = \begin{pmatrix} c_1\,c_2 & s_1 & c_1\,s_2 & c_1\,s_2\,q_3 + c_1\,L_2\,c_2 \\ s_1\,c_2 & -c_1 & s_1\,s_2 & s_1\,s_2\,q_3 + s_1\,L_2\,c_2 \\ s_2 & 0 & -c_2 & -c_2\,q_3 + L_2\,s_2 + L_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Cinematica inversa di posizione

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} c_1\,s_2\,q_3 + c_1\,L_2\,c_2 \\ s_1\,s_2\,q_3 + s_1\,L_2\,c_2 \\ -c_2\,q_3 + L_2\,s_2 + L_1 \end{pmatrix}$$

$$\begin{cases} x = c_1\,s_2\,q_3 + c_1\,L_2\,c_2 \\ y = s_1\,s_2\,q_3 + s_1\,L_2\,c_2 \end{cases} \longrightarrow \begin{cases} x^2 = c_1^2\,(s_2\,q_3 + L_2\,c_2)^2 \\ y = s_1^2(s_2\,q_3 + L_2\,c_2)^2 \end{cases}$$

$$x^2 + y^2 = (c_1^2 + s_1^2)\,(s_2\,q_3 + L_2\,c_2)^2 \rightarrow \begin{cases} x^2 + y^2 = (c_1^2 + s_1^2)\,(s_2\,q_3 + L_2\,c_2)^2 \\ z = -c_2\,q_3 + L_2\,s_2 + L_1 \end{cases}$$

$$\begin{cases} L_1 - z = -c_2\,q_3 + L_2\,s_2 \\ q_3 s_2 + L_2 c_2 = \pm\sqrt{x^2 + y^2} \end{cases}$$

A questo punto è possibile riscrivere l'ultima espressione come:

$$\begin{pmatrix} c_2 & -s_2 \\ s_2 & c_2 \end{pmatrix} \begin{pmatrix} q_3 \\ L_2 \end{pmatrix} = \begin{pmatrix} L_1 - z \\ \pm\sqrt{x^2 + y^2} \end{pmatrix}$$

Poiché $\begin{pmatrix} c_2 & -s_2 \\ s_2 & c_2 \end{pmatrix}$ è una matrice di rotazione, *i* vettori $\begin{pmatrix} q_3 \\ L_2 \end{pmatrix}$ e $\begin{pmatrix} L_1 - z \\ \pm\sqrt{x^2 + y^2} \end{pmatrix}$ devono avere la stessa norma.Imponendo questa condizione, si ottiene l'espressione dello spazio operativo:

$$q_3^2 + L_2^2 = (L_1 - z)^2 + x^2 + y^2$$

Nell'ultima equazione è presente una variabile di giunto che, una volta fissata, permette la definizione del luogo geometrico che identifica lo spazio operativo. In definitiva, il luogo geometrico descritto è una sfera cava di centro $\begin{pmatrix} 0 \\ 0 \\ L_1 \end{pmatrix}$ e raggio $r_{q_3} = \sqrt{q_3^2 + L_2^2}$ di cui si ottiene il raggio interno imponendo $q_3 = 0 \rightarrow r = L_2$ e il raggio esterno per $q_3 \rightarrow \pm\infty \rightarrow r = +\infty$.

Una volta determinato lo spazio operativo, occorre determinare i punti di singolarità e le soluzioni generiche:

$$q_3 = \pm\sqrt{x^2 + y^2 + (z - L_1)^2 - L_2^2}$$

In particolare, si hanno due soluzioni generiche ed una sincolarita se:

$$x^2 + y^2 + (z - L_1)^2 = L_2^2$$

che coincide con la sfera di raggio più piccolo.

Inoltre, definendo l'espressione di q_2, L_2 e di $L_1 - z, \pm\sqrt{x^2+y^2}$ come rispettivamente A_1, A_2, B_1, B_2, possiamo calcolare le altre 2 variabili di giunto:

$$\begin{cases} A_1 c_2 - A_2 s_2 = B_1 \\ A_1 s_2 + A_2 c_2 = B_2 \end{cases} \longrightarrow \begin{pmatrix} A_1 & -A_2 \\ A_2 & A_1 \end{pmatrix} \begin{pmatrix} c_2 \\ s_2 \end{pmatrix} = \begin{pmatrix} B_1 \\ B_2 \end{pmatrix}$$

Il determinante della matrice $\det\begin{pmatrix} A_1 & -A_2 \\ A_2 & A_1 \end{pmatrix} \neq 0$ nei punti non singolari e, sotto questa ipotesi è possibile effetuare l'inversa:

$$\begin{pmatrix} c_2 \\ s_2 \end{pmatrix} = \frac{1}{A_1^2 + A_2^2} \begin{pmatrix} A_1 & A_2 \\ -A_2 & A_1 \end{pmatrix} \begin{pmatrix} B_1 \\ B_2 \end{pmatrix}$$

Quindi la variabile di giunto q_2 ha la seguente espressione:

$$q_2 = \operatorname{atan2}(-A_2 B_1 + A_1 B_2, A_1 B_1 + A_2 B_2)$$

A questo punto la quantità $(s_2\, q_3 + L_2\, c_2)$ è nota e supponiamo che sia $\neq 0$:

$$c_1 = \frac{x}{s_2\, q_3 + L_2\, c_2} = \frac{x}{\sqrt{x^2+y^2}}$$
$$s_1 = \frac{y}{s_2\, q_3 + L_2\, c_2} = \frac{y}{\sqrt{x^2+y^2}}$$

Quindi:

$$q_1 = \operatorname{atan2}\left(\frac{y}{\sqrt{x^2+y^2}}, \frac{x}{\sqrt{x^2+y^2}}\right)$$

Dato che $\sqrt{x^2+y^2} > 0$:

$$q_1 = \operatorname{atan2}(y, x)$$

Orientamento inverso

La risoluzione del problema di orientamento inverso si basa sulla scelta di una terna di Eulero o nautica in condizione non singolari, se possibile.

$$R_{\text{sfericoI}} = \begin{pmatrix} \cos(q_1)\cos(q_2) & \sin(q_1) & \cos(q_1)\sin(q_2) \\ \sin(q_1)\cos(q_2) & -\cos(q_1) & \sin(q_1)\sin(q_2) \\ \sin(q_2) & 0 & -\cos(q_2) \end{pmatrix}$$

$$R_{\text{zyx}} = \begin{pmatrix} c_y c_z & \cdots\cdots\cdots & \\ c_y s_z & \cdots\cdots\cdots & \\ -s_y & s_x c_y & c_x c_y \end{pmatrix}$$

Uguagliando i termini $R_{3,1}$ della matrice di rotazione R_{sfericoI} con quella della matrice di rotazione con angoli nautivi R_{zyx}:

$$-s_y = \sin(q_2)\,\text{singolare se} -s_y = \sin(q_2) = \pm 1 \to q_2 = \pm\frac{\pi}{2}$$

Supponsiamo che $q_2 \neq \pm\frac{\pi}{2}$:

$$\begin{cases} s_y = -\sin(q_2) \\ c_y = \pm\sqrt{1-s_2^2} = \pm c_2 \end{cases} \longrightarrow \phi_y = \text{atan2}(-s_2 \pm c_2) = \begin{cases} -q_2 \\ q_2 + \pi \end{cases}$$

Inoltre:

$$\begin{cases} s_x c_y = 0 \\ c_x c_y = \mp 1 \end{cases} \longrightarrow \begin{cases} \pm s_x c_2 = 0 \\ \pm c_x c_2 = -c_2 \end{cases} \longrightarrow \begin{cases} s_x = 0 \\ c_x = \mp 1 \end{cases} \to \phi_x = \text{atan2}(0, \mp 1) = \begin{cases} \pi \\ 0 \end{cases}$$

$$\begin{cases} c_y c_z = c_1 c_2 \\ c_y c_z = s_1 c_2 \end{cases} \to \begin{cases} \pm c_2 c_z = c_1 c_2 \\ \pm c_2 c_z = s_1 c_2 \end{cases} \to \begin{cases} c_z = \pm c_1 \\ s_z = \pm s_1 \end{cases} \to \phi_z = \text{atan2}(\pm s_1, c_1) = \begin{cases} q_1 \\ q_1 + \pi \end{cases}$$

Riassumendo:

$$\begin{pmatrix} \pi \\ -q_2 \\ q_1 \end{pmatrix}, \begin{pmatrix} 0 \\ q_2 + \pi \\ q_1 + \pi \end{pmatrix}$$

In alternativa tramite una scelta di una terna di eulero:

$$R_{\text{zyz}} = \begin{pmatrix} \cdots & \cdots & \cos(\alpha)\sin(\beta) \\ \cdots & \cdots & \sin(\alpha)\sin(\beta) \\ -\sin(\beta)\cos(\gamma) & \sin(\beta)\sin(\gamma) & \cos(\beta) \end{pmatrix}$$

$$\cos(\beta) = -\cos(q_2) \neq \pm 1 \to q_2 \neq \begin{cases} \pi \\ 0 \end{cases}$$

Supponiamo che $q_2 \neq \begin{cases} \pi \\ 0 \end{cases}$ per non avere una soluzione singolare:

$$\sin(\beta) = \pm\sqrt{1-\cos(\beta)^2} = \pm\sqrt{1-\cos(q_2)^2} = \pm\sin(q_2)$$

$$\beta = \text{atan2}(\pm\sin(q_2), \cos(q_2)) = \begin{cases} q_2 \\ -q_2 \end{cases}$$

$$\begin{cases} \cos(\alpha)\sin(\beta) = c_1 s_2 \\ \sin(\alpha)\sin(\beta) = s_1 s_2 \end{cases} \to \begin{cases} \pm\cos(\alpha)\sin(q_2) = \cos(q_1)\sin(q_2) \\ \pm\sin(\alpha)\sin(q_2) = \sin(q_1)\sin(q_2) \end{cases} \to \begin{cases} \cos(\alpha) = \pm\cos(q_1) \\ \sin(\alpha) = \pm\sin(q_1) \end{cases}$$

$$\alpha = \text{atan2}(\pm\sin(q_1), \pm\cos(q_1)) = \begin{cases} q_1 \\ q_1 + \pi \end{cases}$$

$$\begin{cases} -\sin(\beta)\cos(\gamma) = s_2 \\ \sin(\beta)\sin(\gamma) = 0 \end{cases} \to \begin{cases} \mp\cos(\gamma) = + \\ \sin(\gamma) = 0 \end{cases} \longrightarrow \gamma = \text{atan2}(0, \mp 1) = \begin{cases} \pi \\ 0 \end{cases}$$

Riassumendo:

$$\begin{pmatrix} q_1 \\ q_2 \\ \pi \end{pmatrix}, \begin{pmatrix} q_1 + \pi \\ -q_2 \\ 0 \end{pmatrix}$$

```
(%i14) isRotation(M):=block([MC,res],
                          I:ident(3),
                          MC:ident(3),
                          for i:1 thru 3 do
                          (
                          for j:1 thru 3 do
                             (
                                MC[i][j]:M[i][j]
                                )
                             ),
                          MMT:trigsimp(expand(MC.transpose(MC))),
                          detM:trigsimp(expand(determinant(MC))),

                          if MMT=I  and detM=1
                             then(

                                  return(res:1)
                                  )

                          else(

                                res: "R is not rotation matrix"
                                )
                          )
```

(%o14) isRotation(M) := **block** ([MC, res], I: ident(3), MC: ident(3), **for** i **thru** 3 **do for** j **thru** 3 **do** $(\mathrm{MC}_i)_j$: $(M_i)_j$, MMT: trigsimp(expand(MC · transpose(MC))), detM: trigsimp(expand(determinant(MC))), **if** MMT $= I \wedge$ detM $= 1$ **then** return(res: 1) **else** res: R is not rotation matrix)

```
(%i15) calculate(x,y,L1,L2,z):=block([q3plus,q3minus,q2plus,q2minus,q1,res],

             if x^(2)+y^(2)+(z-L1)^2=L2^(2) then print("La soluzione è singolare")
                                         else
                                         (
       q3value:sqrt(trigreduce(trigexpand(ratsimp(x^(2)+y^(2)+(z-L1)^(2)-
       L2^(2))))),
       q3plus:trigreduce(trigexpand(ratsimp(q3value))),
       q3minus:-trigreduce(trigexpand(ratsimp(q3value))),
       twoNorm:trigreduce(trigexpand(sqrt(x^(2)+y^(2)))),
       s2plus:trigreduce(trigexpand(ratsimp(-L2*(L1-z)+q3plus*twoNorm))),
       c2plus:trigreduce(trigexpand(ratsimp(q3plus*(L1-z)+L2*twoNorm))),
       s2minus:trigreduce(trigexpand(ratsimp(-L2*(L1-z)-q3minus*twoNorm))),
       c2minus:trigreduce(trigexpand(ratsimp(q3minus*(L1-z)-L2*twoNorm))),
                                          q2plus:atan2(s2plus,c2plus),
                                          q2minus:atan2(s2minus,c2minus),
                                          q1:atan2(y,x),
                                          res:[[q1,q2plus,q3plus],[q1,q2minus,q3minus]]
                                         )
                                         )
```

(%o15) $\text{calculate}(x,\ y,\ L1,\ L2,\ z) := \textbf{block}\left([q3\text{plus},\ q3\text{minus},\ q2\text{plus},\ q2\text{minus},\ q1,\ \text{res}],\ \textbf{if}\ x^2 + y^2 + (z-L1)^2 = L2^2\ \textbf{then}\ \text{print}(\text{La soluzione è singolare })\ \textbf{else}\left(q3\text{value}\colon \sqrt{\text{trigreduce}(\text{trigexpand}(\text{ratsimp}(x^2+y^2+(z-L1)^2-L2^2)))},\ q3\text{plus}\colon \text{trigreduce}(\text{trigexpand}(\text{ratsimp}(q3\text{value}))),\ q3\text{minus}\colon -\text{trigreduce}(\text{trigexpand}(\text{ratsimp}(q3\text{value}))),\ \text{twoNorm}\colon \text{trigreduce}\left(\text{trigexpand}\left(\sqrt{x^2+y^2}\right)\right),\ s2\text{plus}\colon \text{trigreduce}(\text{trigexpand}(\text{ratsimp}((-L2)\,(L1-z)+q3\text{plus}\,\text{twoNorm}))),\ c2\text{plus}\colon \text{trigreduce}(\text{trigexpand}(\text{ratsimp}(q3\text{plus}\,(L1-z)+L2\,\text{twoNorm}))),\ s2\text{minus}\colon \text{trigreduce}(\text{trigexpand}(\text{ratsimp}((-L2)\,(L1-z)-q3\text{minus}\,\text{twoNorm}))),\ c2\text{minus}\colon \text{trigreduce}(\text{trigexpand}(\text{ratsimp}(q3\text{minus}\,(L1-z)-L2\,\text{twoNorm}))),\ q2\text{plus}\colon \text{atan2}(s2\text{plus}, c2\text{plus}),\ q2\text{minus}\colon \text{atan2}(s2\text{minus}, c2\text{minus}),\ q1\colon \text{atan2}(y,x),\ \text{res}\colon [[q1,\ q2\text{plus},\ q3\text{plus}],\ [q1,\ q2\text{minus},\ q3\text{minus}]]\right)\right)$

(%i28)
```
orientation(Qdiretta):=block([sx,cx,sy,cy,phiy1,phiy2,phiz1,phiz2,phix1,
phix2,sz,sxfirst,second,res],

                                rotation:isRotation(Qdiretta),
                                if rotation=1 then(
                                sy:Qdiretta[3][1],

                                if sy=1 or sy=-1 then "soluzione singolare"
                                else(
                                cy:sqrt(1-sy^2),
                                phiy1:atan2(-sy,cy),
                                phiy2:atan2(-sy,-cy),

                                sx:Qdiretta[3][2]/cy,
                                cx:Qdiretta[3][3]/cy,
                                phix1:atan2(sx,cx),
                                phix2:atan2(sx,-cx),
                                cz:Qdiretta[1][1]/cy,
                                sz:Qdiretta[2][1]/cy,
                                phiz1:atan2(sz,cz),
                                phiz2:atan2(-sz,-cz),
                                first:[phix1,phiy1,phiz1],
                                second:[phix2,phiy2,phiz2],

                                res:[first,second])
                                )
                                else res:rotation

);
```

(%o28) $\text{orientation}(\text{Qdiretta}) := \textbf{block}\left([\text{sx, cx, sy, cy, phiy1, phiy2, phiz1, phiz2, phix1, phix2, sz, sxfirst, second, res}],\ \text{rotation: isRotation(Qdiretta)},\ \textbf{if}\ \text{rotation} = 1\ \textbf{then}\left(\text{sy}\colon (\text{Qdiretta}_3)_1,\ \textbf{if}\ \text{sy} = 1 \vee \text{sy} = -1\ \textbf{then}\ \text{soluzione singolare }\ \textbf{else}\left(\text{cy}\colon \sqrt{1-\text{sy}^2},\ \text{phiy1}\colon \text{atan2}(-\text{sy}, \text{cy}),\ \text{phiy2}\colon \text{atan2}(-\text{sy}, -\text{cy}),\ \text{sx}\colon \frac{(\text{Qdiretta}_3)_2}{\text{cy}},\ \text{cx}\colon \frac{(\text{Qdiretta}_3)_3}{\text{cy}},\ \text{phix1}\colon \text{atan2}(\text{sx}, \text{cx}),\ \text{phix2}\colon \text{atan2}(\text{sx}, -\text{cx}),\ \text{cz}\colon \frac{(\text{Qdiretta}_1)_1}{\text{cy}},\ \text{sz}\colon \frac{(\text{Qdiretta}_2)_1}{\text{cy}},\ \text{phiz1}\colon \text{atan2}(\text{sz}, \text{cz}),\ \text{phiz2}\colon \text{atan2}(-\text{sz}, -\text{cz}),\ \text{first}\colon [\text{phix1}, \text{phiy1}, \text{phiz1}],\ \text{second}\colon [\text{phix2}, \text{phiy2}, \text{phiz2}],\ \text{res}\colon [\text{first}, \text{second}]\right)\right)\ \textbf{else}\ \text{res}\colon \text{rotation}\right)$

```
(%i29) invSfericoI(Qdiretta,L1,L2):=block([x,y,z,pos1,pos2,orien1,orien2,res],
                                          x:Qdiretta[1][4],
                                          y:Qdiretta[2][4],
                                          z:Qdiretta[3][4],
                                          pos:calculate(x,y,L1,L2,z),
                                          orien:orientation(Qdiretta),
                                          orien1:orien[1],
                                          orien2:orien[2],
                                          pos1:pos[1],
                                          pos2:pos[2],
                                          res:[pos1,pos2,orien1,orien2]
                                          );
```

(%o29) invSfericoI(Qdiretta, $L1$, $L2$) := **block** ([x, y, z, pos1, pos2, orien1, orien2, res], x: $(\mathrm{Qdiretta}_1)_4$, y: $(\mathrm{Qdiretta}_2)_4$, z: $(\mathrm{Qdiretta}_3)_4$, pos: calculate(x, y, $L1$, $L2$, z), orien: orientation(Qdiretta), orien1: orien_1, orien2: orien_2, pos1: pos_1, pos2: pos_2, res: [pos1, pos2, orien1, orien2])

```
(%i30) QsfericoRRP:Q[sferico](%pi/3,0,10,5,10);
```

(%o30)
$$\begin{pmatrix} \frac{1}{2} & \frac{\sqrt{3}}{2} & 0 & 5 \\ \frac{\sqrt{3}}{2} & -\frac{1}{2} & 0 & 5\sqrt{3} \\ 0 & 0 & -1 & -5 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
(%i31) invSfericoI(QsfericoRRP,5,10);
```

(%o31)
$$\left[\left[\frac{\pi}{3}, 0, 10\right], \left[\frac{\pi}{3}, \pi, -10\right], \left[\pi, 0, \frac{\pi}{3}\right], \left[0, \pi, -\frac{2\pi}{3}\right]\right]$$

```
(%i32) Qsferico11:Q[sferico](q1,q2,q3,L1,L2)
```

(%o32) $(\cos(q1)\cos(q2), \sin(q1), \cos(q1)\sin(q2), \cos(q1)(\sin(q2)\,q3 + L2\cos(q2));$ $\sin(q1)\cos(q2), -\cos(q1), \sin(q1)\sin(q2), \sin(q1)(\sin(q2)\,q3 + L2\cos(q2)); \sin(q2), 0, -\cos(q2),$ $-\cos(q2)\,q3 + L2\sin(q2) + L1; 0, 0, 0, 1)$

```
(%i33) invSfericoI(Qsferico11,L1,L2);
```

(%o33)
$$\Bigg[\Bigg[\mathrm{atan2}(\sin(q1)(\sin(q2)\,q3 + L2\cos(q2)), \cos(q1)(\sin(q2)\,q3 + L2\cos(q2))),$$
$$\mathrm{atan2}\Bigg(\frac{\sqrt{-\cos(2\,q2)\,q3^2 + q3^2 + 2\,L2\sin(2\,q2)\,q3 + L2^2\cos(2\,q2) + L2^2}\,|q3|}{\sqrt{2}} - L2\cos(q2)\,q3 +$$
$$L2^2\sin(q2), \cos(q2)\,q3\,|q3| - L2\sin(q2)\,|q3| +$$
$$\frac{L2\sqrt{-\cos(2\,q2)\,q3^2 + q3^2 + 2\,L2\sin(2\,q2)\,q3 + L2^2\cos(2\,q2) + L2^2}}{\sqrt{2}}\Bigg), |q3|\Bigg],$$
$$\Bigg[\mathrm{atan2}(\sin(q1)(\sin(q2)\,q3 + L2\cos(q2)), \cos(q1)(\sin(q2)\,q3 + L2\cos(q2))),$$
$$\mathrm{atan2}\Bigg(\frac{\sqrt{-\cos(2\,q2)\,q3^2 + q3^2 + 2\,L2\sin(2\,q2)\,q3 + L2^2\cos(2\,q2) + L2^2}\,|q3|}{\sqrt{2}} - L2\cos(q2)\,q3 +$$
$$L2^2\sin(q2), -\cos(q2)\,q3\,|q3| + L2\sin(q2)\,|q3| -$$
$$\frac{L2\sqrt{-\cos(2\,q2)\,q3^2 + q3^2 + 2\,L2\sin(2\,q2)\,q3 + L2^2\cos(2\,q2) + L2^2}}{\sqrt{2}}\Bigg), -|q3|\Bigg], \Bigg[\mathrm{atan2}\Bigg(0,$$
$$-\frac{\cos(q2)}{\sqrt{1-\sin(q2)^2}}\Bigg), -\mathrm{atan2}\left(\sin(q2), \sqrt{1-\sin(q2)^2}\right), \mathrm{atan2}\Bigg(\frac{\sin(q1)\cos(q2)}{\sqrt{1-\sin(q2)^2}}, \frac{\cos(q1)\cos(q2)}{\sqrt{1-\sin(q2)^2}}\Bigg)\Bigg],$$

$$\left[\operatorname{atan2}\left(0, \frac{\cos(q2)}{\sqrt{1-\sin(q2)^2}}\right), -\operatorname{atan2}\left(\sin(q2), -\sqrt{1-\sin(q2)^2}\right), -\operatorname{atan2}\left(\frac{\sin(q1)\cos(q2)}{\sqrt{1-\sin(q2)^2}},\right.\right.$$
$$\left.\left.-\frac{\cos(q1)\cos(q2)}{\sqrt{1-\sin(q2)^2}}\right)\right]\Bigg]$$

(%i34)

Singolarità

```
Maxima 5.44.0 http://maxima.sourceforge.net
using Lisp SBCL 2.0.0
Distributed under the GNU Public License. See the file COPYING.
Dedicated to the memory of William Schelter.
The function bug_report() provides bug reporting information.
```

```
(%i1) x:cos (q[1])*(q[3]*sin (q[2])+L[2]*cos (q[2]));
```

(%o1) $\cos(q_1)(q_3 \sin(q_2) + L_2 \cos(q_2))$

```
(%i2) y:sin (q[1])*(q[3]*sin (q[2])+L[2]*cos (q[2]));
```

(%o2) $\sin(q_1)(q_3 \sin(q_2) + L_2 \cos(q_2))$

```
(%i3) z:L[2]*sin (q[2])-q[3]*cos (q[2])+L[1];
```

(%o3) $L_2 \sin(q_2) - q_3 \cos(q_2) + L_1$

```
(%i29) J:matrix([diff(x,q[1]),diff(x,q[2]),diff(x,q[3])],
                [diff(y,q[1]),diff(y,q[2]),diff(y,q[3])],
                [diff(z,q[1]),diff(z,q[2]),diff(z,q[3])]);
```

(%o29) $(-\sin(q_1)(q_3 \sin(q_2) + L_2 \cos(q_2)), \cos(q_1)(q_3 \cos(q_2) - L_2 \sin(q_2)), \cos(q_1)\sin(q_2);$ $\cos(q_1)(q_3 \sin(q_2) + L_2 \cos(q_2)), \sin(q_1)(q_3 \cos(q_2) - L_2 \sin(q_2)), \sin(q_1)\sin(q_2); 0, q_3 \sin(q_2) +$ $L_2 \cos(q_2), -\cos(q_2))$

```
(%i5) dJ:trigsimp(expand(determinant(J)));
```

(%o5) $q_3^2 \sin(q_2) + L_2\, q_3 \cos(q_2)$

$$q_3^2 \sin(q_2) + L_2\, q_3 \cos(q_2) \neq 0$$

Supponendo che $q_3 \neq 0$:

$$\sin(q_2) = -\frac{L_2 \cos(q_2)}{q_3}$$

$$\frac{L_2^2 \cos^2(q_2)}{q_3^2} + \cos^2(q_2) = 1$$

$$\cos^2(q_2)\left(\frac{q_3^2 + L_2^2}{q_3^2}\right) = \frac{q_3^2}{q_3^2 + L_2^2} \to \cos(q_2) = \pm\frac{q_3}{\sqrt{q_3^2 + L_2^2}}$$

$$\sin(q_2) = \mp\frac{L_2}{\sqrt{q_3^2 + L_2^2}}$$

$$q_2 = \operatorname{atan2}(\mp L_2, \pm q_3)$$

Le singolarità di velocità di hanno per:

$$q_3 = 0 \wedge q_2 = \begin{cases} \operatorname{atan2}(-L_2, q_3) \\ \operatorname{atan2}(+L_2, -q_3) \end{cases}$$

```
(%i6) Jq3:subst(q[3]=0,J);
```

(%o6) $\begin{pmatrix} -L_2\sin(q_1)\cos(q_2) & -L_2\cos(q_1)\sin(q_2) & \cos(q_1)\sin(q_2) \\ L_2\cos(q_1)\cos(q_2) & -L_2\sin(q_1)\sin(q_2) & \sin(q_1)\sin(q_2) \\ 0 & L_2\cos(q_2) & -\cos(q_2) \end{pmatrix}$

```
(%i7) trigsimp(nullspace(Jq3));
```

Proviso: $\mathrm{notequal}(\cos(q_1)\sin(q_2),0) \wedge \mathrm{notequal}((L_2\sin(q_1)^2 + L_2\cos(q_1)^2)\cos(q_2)\sin(q_2),0)$

(%o7) $\mathrm{span}\left(\begin{pmatrix} 0 \\ L_2\cos(q_2)\sin(q_2) \\ L_2^2\cos(q_2)\sin(q_2) \end{pmatrix}\right)$

Supponendo che $q_2 \neq 0$, le singolarità di velocità di hanno per $v \in \mathrm{Im}\left\{\begin{pmatrix} 0 \\ L_2\cos(q_2)\sin(q_2) \\ L_2^2\cos(q_2)\sin(q_2) \end{pmatrix}\right\}$

Se $q_2 = 0$:

```
(%i8) Jq32:subst(q[2]=0,Jq3);
```

(%o8) $\begin{pmatrix} -L_2\sin(q_1) & 0 & 0 \\ L_2\cos(q_1) & 0 & 0 \\ 0 & L_2 & -1 \end{pmatrix}$

```
(%i9) nullspace(Jq32);
```

Proviso: $\mathrm{notequal}(-L_2\sin(q_1),0) \wedge \mathrm{notequal}(-L_2^2\sin(q_1),0)$

(%o9) $\mathrm{span}\left(\begin{pmatrix} 0 \\ -L_2\sin(q_1) \\ -L_2^2\sin(q_1) \end{pmatrix}\right)$

Se $q_3 = 0$, $q_2 = 0$, $q_1 \neq 0$, le singolarità di velocità sono per $v \in \Im m\left\{\begin{pmatrix} 0 \\ -L_2\sin(q_1) \\ -L_2^2\sin(q_1) \end{pmatrix}\right\}$

Se $q_3 = 0$, $q_2 = 0$, $q_1 = 0$:

```
(%i10) Jq321:subst(q[1]=0,Jq32);
```

(%o10) $\begin{pmatrix} 0 & 0 & 0 \\ L_2 & 0 & 0 \\ 0 & L_2 & -1 \end{pmatrix}$

```
(%i11) nullspace(Jq321);
```

Proviso: $\mathrm{notequal}(L_2,0) \wedge \mathrm{notequal}(L_2^2,0)$

(%o11) $\mathrm{span}\left(\begin{pmatrix} 0 \\ L_2 \\ L_2^2 \end{pmatrix}\right)$

(%i12)

le singolarità di velocità sono per $v \in \Im m\left\{\begin{pmatrix} 0 \\ L_2 \\ L_2^2 \end{pmatrix}\right\}$

Inoltre se $q_2 = \mathrm{atan2}(-L_2, q_3)$:

```
(%i14) J:ratsimp(subst(q[2]=atan2(-L[2]/sqrt(q[3]^2+L[2]^2),q[3]/
       sqrt(q[3]^2+L[2]^2)),J));
```

(%o14) $\begin{pmatrix} 0 & \sqrt{q_3^2+L_2^2}\cos(q_1) & -\frac{L_2\cos(q_1)}{\sqrt{q_3^2+L_2^2}} \\ 0 & \sqrt{q_3^2+L_2^2}\sin(q_1) & -\frac{L_2\sin(q_1)}{\sqrt{q_3^2+L_2^2}} \\ 0 & 0 & -\frac{q_3}{\sqrt{q_3^2+L_2^2}} \end{pmatrix}$

```
(%i15) nullspace(J);
```

Proviso: $\mathrm{notequal}\left(\sqrt{q_3^2+L_2^2}\cos(q_1),0\right) \wedge \mathrm{notequal}((-q_3^3-L_2^2\,q_3)\cos(q_1),0)$

(%o15) $\mathrm{span}\left(\begin{pmatrix} (-q_3^3-L_2^2\,q_3)\cos(q_1) \\ 0 \\ 0 \end{pmatrix}\right)$

Se $q_1 \neq \frac{\pi}{2}$, le singolarità di velocità sono per $v \in \Im m\left\{\begin{pmatrix} (-q_3^3-L_2^2\,q_3)\cos(q_1) \\ 0 \\ 0 \end{pmatrix}\right\}$.

Se $q_1 = \frac{\pi}{2}$:

```
(%i20)  Jq1:subst(q[1]=%pi/2,J);
```

(%o20) $\begin{pmatrix} 0 & 0 & 0 \\ 0 & \sqrt{q_3^2+L_2^2} & -\frac{L_2}{\sqrt{q_3^2+L_2^2}} \\ 0 & 0 & -\frac{q_3}{\sqrt{q_3^2+L_2^2}} \end{pmatrix}$

```
(%i21) nullspace(Jq1);
```

Proviso: $\mathrm{notequal}\left(\sqrt{q_3^2+L_2^2},0\right) \wedge \mathrm{notequal}(-q_3^3-L_2^2\,q_3,0)$

(%o21) $\mathrm{span}\left(\begin{pmatrix} -q_3^3-L_2^2\,q_3 \\ 0 \\ 0 \end{pmatrix}\right)$

le singolarità di velocità sono per $v \in \Im m\left\{\begin{pmatrix} -q_3^3-L_2^2\,q_3 \\ 0 \\ 0 \end{pmatrix}\right\}$

Infine, se $q_2 = \mathrm{atan2}(L_2, -q_3)$:

```
(%i22) J:ratsimp(subst(q[2]=atan2(L[2]/sqrt(q[3]^2+L[2]^2),-q[3]/
       sqrt(q[3]^2+L[2]^2)),J));
```

(%o22) $\begin{pmatrix} 0 & \sqrt{q_3^2+L_2^2}\cos(q_1) & -\frac{L_2\cos(q_1)}{\sqrt{q_3^2+L_2^2}} \\ 0 & \sqrt{q_3^2+L_2^2}\sin(q_1) & -\frac{L_2\sin(q_1)}{\sqrt{q_3^2+L_2^2}} \\ 0 & 0 & -\frac{q_3}{\sqrt{q_3^2+L_2^2}} \end{pmatrix}$

```
(%i23) nullspace(J);
```

Proviso: $\mathrm{notequal}\left(\sqrt{q_3^2+L_2^2}\cos(q_1),0\right) \wedge \mathrm{notequal}((-q_3^3-L_2^2\,q_3)\cos(q_1),0)$

(%o23) $\mathrm{span}\left(\begin{pmatrix} (-q_3^3-L_2^2\,q_3)\cos(q_1) \\ 0 \\ 0 \end{pmatrix}\right)$

Se $q_1 \neq \frac{\pi}{2}$, le singolarità di velocità sono per $v \in \Im m\left\{\begin{pmatrix} (-q_3^3-L_2^2\,q_3)\cos(q_1) \\ 0 \\ 0 \end{pmatrix}\right\}$.

Se $q_1 = \frac{\pi}{2}$:

```
(%i24) J:subst(q[1]=%pi/2,J);
```

(%o24) $\begin{pmatrix} 0 & 0 & 0 \\ 0 & \sqrt{q_3^2+L_2^2} & -\frac{L_2}{\sqrt{q_3^2+L_2^2}} \\ 0 & 0 & -\frac{q_3}{\sqrt{q_3^2+L_2^2}} \end{pmatrix}$

```
(%i25) nullspace(J);
```

Proviso: notequal$\left(\sqrt{q_3^2+L_2^2},0\right) \wedge$ notequal$(-q_3^3-L_2^2\,q_3,0)$

(%o25) $\operatorname{span}\left(\begin{pmatrix} -q_3^3-L_2^2\,q_3 \\ 0 \\ 0 \end{pmatrix}\right)$

le singolarità di velocità sono per $v \in \Im m\left\{\begin{pmatrix} (-q_3^3-L_2^2\,q_3)\cos(q_1) \\ 0 \\ 0 \end{pmatrix}\right\}$.

(%i26)

Singolairtà di Forza

(%i40) `J:-transpose(J);`

(%o40) $(-\sin(q_1)\,(q_3\sin(q_2)+L_2\cos(q_2)),\cos(q_1)\,(q_3\cos(q_2)-L_2\sin(q_2)),\cos(q_1)\sin(q_2);$ $\cos(q_1)\,(q_3\sin(q_2)+L_2\cos(q_2)),\sin(q_1)\,(q_3\cos(q_2)-L_2\sin(q_2)),\sin(q_1)\sin(q_2);0,q_3\sin(q_2)+$ $L_2\cos(q_2),-\cos(q_2))$

(%i31) `dJ:trigsimp(expand(determinant(J)));`

(%o31) $-q_3^2\sin(q_2)-L_2\,q_3\cos(q_2)$

$$-q_3^2\sin(q_2)-L_2\,q_3\cos(q_2)\neq 0$$

Supponendo che $q_3\neq 0$:

$$\sin(q_2)=-\frac{L_2\cos(q_2)}{q_3}$$

$$\frac{L_2^2\cos^2(q_2)}{q_3^2}+\cos^2(q_2)=1$$

$$\cos^2(q_2)\left(\frac{q_3^2+L_2^2}{q_3^2}\right)=\frac{q_3^2}{q_3^2+L_2^2}\rightarrow\cos(q_2)=\pm\frac{q_3}{\sqrt{q_3^2+L_2^2}}$$

$$\sin(q_2)=\mp\frac{L_2}{\sqrt{q_3^2+L_2^2}}$$

$$q_2=\operatorname{atan2}(\mp L_2,\pm q_3)$$

Le singolarità di forza si hanno per:

$$q_3=0\wedge q_2=\begin{cases}\operatorname{atan2}(-L_2,q_3)\\ \operatorname{atan2}(+L_2,-q_3)\end{cases}$$

$$-q_3^2\sin(q_2)-L_2\,q_3\cos(q_2)=0$$

Se $q_3\neq 0$:

$$-q_3\sin(q_2)-L_2\cos(q_2)=0$$

$$\sin(q_2)=-\frac{L_2}{q_3}\cos(q_2)$$

$$\sin(q_2)^2+\cos(q_2)^2=1\rightarrow-\frac{L_2^2}{q_3^2}\cos(q_2)^2+\cos(q_2)^2=1$$

(%i32) `Jq3:subst(q[3]=0,J);`

(%o32) $\begin{pmatrix} L_2\sin(q_1)\cos(q_2) & -L_2\cos(q_1)\cos(q_2) & 0 \\ L_2\cos(q_1)\sin(q_2) & L_2\sin(q_1)\sin(q_2) & -L_2\cos(q_2) \\ -\cos(q_1)\sin(q_2) & -\sin(q_1)\sin(q_2) & \cos(q_2) \end{pmatrix}$

(%i33) `trigsimp(nullspace(Jq3));`

Proviso: notequal$(L_2\sin(q_1)\cos(q_2),0)\wedge$ notequal$(-L_2^2\sin(q_1)\cos(q_2)^2,0)$

(%o33) $\mathrm{span}\left(\begin{pmatrix} -L_2^2\cos(q_1)\cos(q_2)^2 \\ -L_2^2\sin(q_1)\cos(q_2)^2 \\ -L_2^2\cos(q_2)\sin(q_2) \end{pmatrix}\right)$

(%i34)

$$L_2\sin(q_1)\cos(q_2)\neq 0 \rightarrow q_1\neq 0,\, q_2\neq\frac{\pi}{2}$$

Se $q_1\neq 0, q_2\neq\frac{\pi}{2}$, si hanno le singolarità di forza se $\tau\in\mathrm{Im}\left\{\begin{pmatrix} -L_2^2\cos(q_1)\cos(q_2)^2 \\ -L_2^2\sin(q_1)\cos(q_2)^2 \\ -L_2^2\cos(q_2)\sin(q_2) \end{pmatrix}\right\}$

Se $q_1=0\wedge q_2\neq\frac{\pi}{2}$:

```
(%i34) Jq31:subst(q[1]=0,Jq3);
```

(%o34) $\begin{pmatrix} 0 & -L_2\cos(q_2) & 0 \\ L_2\sin(q_2) & 0 & -L_2\cos(q_2) \\ -\sin(q_2) & 0 & \cos(q_2) \end{pmatrix}$

```
(%i35) nullspace(Jq31);
```

Proviso: $\mathrm{notequal}(-L_2\cos(q_2),0)\wedge\mathrm{notequal}(L_2^2\cos(q_2)^2,0)$

(%o35) $\mathrm{span}\left(\begin{pmatrix} L_2^2\cos(q_2)^2 \\ 0 \\ L_2^2\cos(q_2)\sin(q_2) \end{pmatrix}\right)$

(%i36)

Se $q_1=0, q_2\neq\frac{\pi}{2}$, si hanno le singolarità di forza se $\tau\in\mathrm{Im}\left\{\begin{pmatrix} L_2^2\cos(q_2)^2 \\ 0 \\ L_2^2\cos(q_2)\sin(q_2) \end{pmatrix}\right\}$

Se $q_1\neq 0, q_2=\frac{\pi}{2}$:

```
(%i36) Jq32:subst(q[2]=%pi/2,Jq3);
```

(%o36) $\begin{pmatrix} 0 & 0 & 0 \\ L_2\cos(q_1) & L_2\sin(q_1) & 0 \\ -\cos(q_1) & -\sin(q_1) & 0 \end{pmatrix}$

```
(%i37) nullspace(Jq32);
```

Proviso: $\mathrm{notequal}(L_2\cos(q_1),0)$

(%o37) $\mathrm{span}\left(\begin{pmatrix} 0 \\ 0 \\ -L_2\sin(q_1) \end{pmatrix}, \begin{pmatrix} -L_2\sin(q_1) \\ L_2\cos(q_1) \\ 0 \end{pmatrix}\right)$

si hanno le singolarità di forza se $\tau\in\mathrm{Im}\left\{\begin{pmatrix} 0 \\ 0 \\ -L_2\sin(q_1) \end{pmatrix}, \begin{pmatrix} -L_2\sin(q_1) \\ L_2\cos(q_1) \\ 0 \end{pmatrix}\right\}$

Se $q_1=0, q_2=\frac{\pi}{2}$:

```
(%i38) Jq321:subst([q[2]=%pi/2,q[1]=0],Jq3);
```

(%o38) $\begin{pmatrix} 0 & 0 & 0 \\ L_2 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix}$

```
(%i39) nullspace(Jq321);
```

Proviso: $\mathrm{notequal}(L_2,0)$

(%o39) $\operatorname{span}\left(\begin{pmatrix}0\\0\\L_2\end{pmatrix},\begin{pmatrix}0\\L_2\\0\end{pmatrix}\right)$

si hanno le singolarità di forza se $\tau\in\operatorname{Im}\left\{\begin{pmatrix}0\\0\\L_2\end{pmatrix},\begin{pmatrix}0\\L_2\\0\end{pmatrix}\right\}$

Se $q_2=\operatorname{atan2}(-L_2,q_3)$:

```
(%i43) Jq2:subst(q[2]=atan2(-L[2],q[3]),J);
```

(%o43) $$\begin{pmatrix}0 & \left(\frac{q_3^2}{\sqrt{q_3^2+L_2^2}}+\frac{L_2^2}{\sqrt{q_3^2+L_2^2}}\right)\cos(q_1) & -\frac{L_2\cos(q_1)}{\sqrt{q_3^2+L_2^2}}\\ 0 & \left(\frac{q_3^2}{\sqrt{q_3^2+L_2^2}}+\frac{L_2^2}{\sqrt{q_3^2+L_2^2}}\right)\sin(q_1) & -\frac{L_2\sin(q_1)}{\sqrt{q_3^2+L_2^2}}\\ 0 & 0 & -\frac{q_3}{\sqrt{q_3^2+L_2^2}}\end{pmatrix}$$

```
(%i44) nullspace(Jq2)
```

Proviso: $\operatorname{notequal}\left(\sqrt{q_3^2+L_2^2}\cos(q_1),0\right)\wedge\operatorname{notequal}((-q_3^3-L_2^2\,q_3)\cos(q_1),0)$

(%o44) $\operatorname{span}\left(\begin{pmatrix}(-q_3^3-L_2^2\,q_3)\cos(q_1)\\0\\0\end{pmatrix}\right)$

Se $q_1\neq 0$ si hanno le singolarità di forza se $\tau\in\operatorname{Im}\left\{\begin{pmatrix}(-q_3^3-L_2^2\,q_3)\cos(q_1)\\0\\0\end{pmatrix}\right\}$.

Se $q_1=0$:

```
(%i45) Jq21:subst(q[1]=0,Jq2);
```

(%o45) $$\begin{pmatrix}0 & \frac{q_3^2}{\sqrt{q_3^2+L_2^2}}+\frac{L_2^2}{\sqrt{q_3^2+L_2^2}} & -\frac{L_2}{\sqrt{q_3^2+L_2^2}}\\ 0 & 0 & 0\\ 0 & 0 & -\frac{q_3}{\sqrt{q_3^2+L_2^2}}\end{pmatrix}$$

```
(%i46) nullspace(Jq21);
```

Proviso: $\operatorname{notequal}\left(\sqrt{q_3^2+L_2^2},0\right)\wedge\operatorname{notequal}(-q_3^3-L_2^2\,q_3,0)$

(%o46) $\operatorname{span}\left(\begin{pmatrix}-q_3^3-L_2^2\,q_3\\0\\0\end{pmatrix}\right)$

si hanno le singolarità di forza se $\tau\in\operatorname{Im}\left\{\begin{pmatrix}-q_3^3-L_2^2\,q_3\\0\\0\end{pmatrix}\right\}$

Se $q_2=\operatorname{atan2}(L_2,-q_3)$:

```
(%i47) Jq2:subst(q[2]=atan2(L[2],-q[3]),J);
```

(%o47) $$\begin{pmatrix}0 & \left(-\frac{q_3^2}{\sqrt{q_3^2+L_2^2}}-\frac{L_2^2}{\sqrt{q_3^2+L_2^2}}\right)\cos(q_1) & \frac{L_2\cos(q_1)}{\sqrt{q_3^2+L_2^2}}\\ 0 & \left(-\frac{q_3^2}{\sqrt{q_3^2+L_2^2}}-\frac{L_2^2}{\sqrt{q_3^2+L_2^2}}\right)\sin(q_1) & \frac{L_2\sin(q_1)}{\sqrt{q_3^2+L_2^2}}\\ 0 & 0 & \frac{q_3}{\sqrt{q_3^2+L_2^2}}\end{pmatrix}$$

```
(%i48) nullspace(Jq2);
```

Proviso: $\operatorname{notequal}\left(-\sqrt{q_3^2+L_2^2}\cos(q_1),0\right)\wedge\operatorname{notequal}((-q_3^3-L_2^2\,q_3)\cos(q_1),0)$

(%o48) $\mathrm{span}\left(\begin{pmatrix} (-q_3^3 - L_2^2\, q_3)\cos(q_1) \\ 0 \\ 0 \end{pmatrix}\right)$

Se $q_1 \neq 0$ si hanno le singolarità di forza se $\tau \in \mathrm{Im}\left\{\begin{pmatrix} (-q_3^3 - L_2^2\, q_3)\cos(q_1) \\ 0 \\ 0 \end{pmatrix}\right\}$.

Se $q_1 = 0$:

```
(%i49) Jq21:subst(q[1]=0,Jq2);
```

(%o49) $\begin{pmatrix} 0 & -\frac{q_3^2}{\sqrt{q_3^2+L_2^2}} - \frac{L_2^2}{\sqrt{q_3^2+L_2^2}} & \frac{L_2}{\sqrt{q_3^2+L_2^2}} \\ 0 & 0 & 0 \\ 0 & 0 & \frac{q_3}{\sqrt{q_3^2+L_2^2}} \end{pmatrix}$

```
(%i50) nullspace(Jq21);
```

Proviso: $\mathrm{notequal}\left(-\sqrt{q_3^2+L_2^2}, 0\right) \wedge \mathrm{notequal}(-q_3^3 - L_2^2\, q_3, 0)$

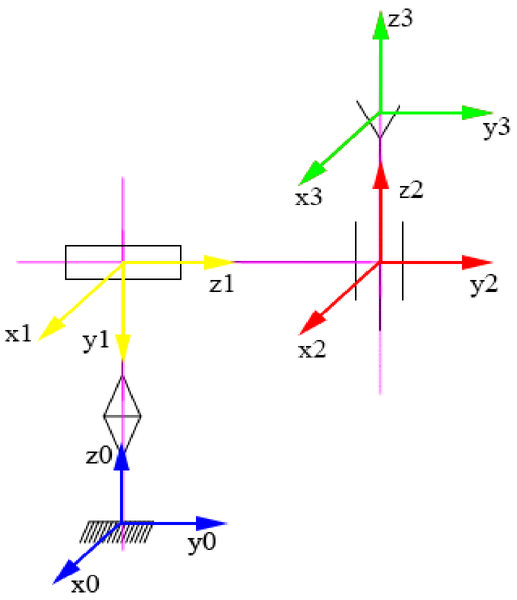
(%o50) $\mathrm{span}\left(\begin{pmatrix} -q_3^3 - L_2^2\, q_3 \\ 0 \\ 0 \end{pmatrix}\right)$

si hanno le singolarità di forza se $\tau \in \mathrm{Im}\left\{\begin{pmatrix} -q_3^3 - L_2^2\, q_3 \\ 0 \\ 0 \end{pmatrix}\right\}$.

```
(%i51)
```

Cinematica diretta Robot Stanford

***N.B.:* le grandezze diverse da quelle di giunto q_i sono L_i, D_i. Esse sono rispettivamente la distanza tra i sistemi di riferimento R_i e R_{i+1} nelle operazioni della matrice avvitamento $A_z(\theta, d)$ e $A_x(\alpha, a)$.**



	ϑ	d	α	a
1	q_1	L_1	$-\frac{\pi}{2}$	0
2	q_2	L_2	$\frac{\pi}{2}$	0
3	$-\frac{\pi}{2}$	q_3	0	0

Tabella 1.

Funzioni ausiliarie:

```
(%i1) inverseLaplace(SI,theta):=block([res],
                             M:SI,
                             MC:SI,
                             for i:1 thru 3 do(
                               for j:1 thru 3 do
                                  (
                                    aC:M[i,j],
                                    b:ilt(aC,s,theta),
                                    MC[i,j]:b
                                   )
                                ),
                             res:MC

                          )
```

(%o1) inverseLaplace(SI, ϑ) := **block** ([res], M: SI, MC: SI, **for** i **thru** 3 **do for** j **thru** 3 **do** (aC: $M_{i,j}$, b: ilt(aC, s, ϑ), MC$_{i,j}$: b), res: MC)

(%i2)
```
rotLaplace(k,theta):=block([res],
                         S:ident(3),
                         I:ident(3),
                       for i:1 thru 3 do
                         (
                         for j:1 thru 3 do
                            (
                              if i=j
                                 then S[i][j]:0
                              elseif j>i
                                 then (
                                temp:(-1)^(j-i)*k[3-remainder(i+j,3)],
                                       S[i][j]:temp,
                                       S[j][i]:-temp

                                        )
                              )
                          ),
                         res:inverseLaplace(invert(s*I-S),theta)


                        )
```

(%o2) $\text{rotLaplace}(k,\vartheta) := \mathbf{block}\ ([\text{res}], S\text{: ident}(3), I\text{: ident}(3),$ $\mathbf{for}\ i\ \mathbf{thru}\ 3\ \mathbf{do}\ \mathbf{for}\ j\ \mathbf{thru}\ 3\ \mathbf{do}\ \mathbf{if}\ i=j\ \mathbf{then}\ (S_i)_j\text{: }0\ \mathbf{elseif}\ j>i\ \mathbf{then}\ (\text{temp:}$ $(-1)^{j-i}\,k_{3-\text{remainder}(i+j,3)}, (S_i)_j\text{: temp}, (S_j)_i\text{: }-\text{temp}), \text{res: inverseLaplace}(\text{invert}(s\,I-S),\vartheta))$

(%i3)
```
Av(v,theta,d):=block([res],
                         Trot:rotLaplace(v,theta),
                         row:matrix([0,0,0,1]),
                         Atemp:addcol(Trot,d*transpose(v)),
                         A:addrow(Atemp,row),
                         res:A
                         )
```

(%o3) $\text{Av}(v,\vartheta,d) := \mathbf{block}\ ([\text{res}], \text{Trot: rotLaplace}(v,\vartheta), \text{row: } \begin{pmatrix} 0 & 0 & 0 & 1 \end{pmatrix}, \text{Atemp: addcol}(\text{Trot},$ $d\,\text{transpose}(v)), A\text{: addrow}(\text{Atemp},\text{row}), \text{res: } A)$

(%i4)
```
Q(theta,d,alpha,a):=block([res],
                              tempMat:Av([0,0,1],theta,d).Av([1,0,0],alpha,a),
                              Qtrasf:zeromatrix(4,4),
                              for i:1 thru 4 do
                         (
                         for j:1 thru 4 do
                            (
                              Qtrasf[i][j]:trigreduce(tempMat[i][j])
                                        )
                            ),

                              res:Qtrasf
                              )
```

(%o4) $Q(\vartheta,d,\alpha,a) := \mathbf{block}\ ([\text{res}], \text{tempMat: Av}([0,0,1],\vartheta,d)\cdot\text{Av}([1,0,0],\alpha,a), \text{Qtrasf:}$ $\text{zeromatrix}(4,4), \mathbf{for}\ i\ \mathbf{thru}\ 4\ \mathbf{do}\ \mathbf{for}\ j\ \mathbf{thru}\ 4\ \mathbf{do}\ (\text{Qtrasf}_i)_j\text{: trigreduce}((\text{tempMat}_i)_j), \text{res: Qtrasf})$

(%i5)
```
let(sin(q[1]), s[1]);
```

(%o5) $\sin(q_1) \longrightarrow s_1$

```
(%i6) let(sin(q[2]), s[2]);
```

(%o6) $\sin(q_2) \longrightarrow s_2$

```
(%i7) let(cos(q[1]),c[1]);
```

(%o7) $\cos(q_1) \longrightarrow c_1$

```
(%i8) let(cos(q[2]),c[2]);
```

(%o8) $\cos(q_2) \longrightarrow c_2$

```
(%i9) let(sin(q[1]+q[2]), s[12]);
```

(%o9) $\sin(q_2 + q_1) \longrightarrow s_{12}$

```
(%i10) let(cos(q[1]+q[2]),c[12]);
```

(%o10) $\cos(q_2 + q_1) \longrightarrow c_{12}$

```
(%i11) let(sin(q[2]+q[3]),s[23]);
```

(%o11) $\sin(q_3 + q_2) \longrightarrow s_{23}$

```
(%i12) let(cos(q[2]+q[3]),c[23]);
```

(%o12) $\cos(q_3 + q_2) \longrightarrow c_{23}$

```
(%i13) let(sin(q[1]+q[3]),s[23]);
```

(%o13) $\sin(q_3 + q_1) \longrightarrow s_{23}$

```
(%i14) let(cos(q[1]+q[3]),c[13]);
```

(%o14) $\cos(q_3 + q_1) \longrightarrow c_{13}$

```
(%i15) let(sin(q[3]),s[3]);
```

(%o15) $\sin(q_3) \longrightarrow s_3$

```
(%i16) let(cos(q[3]),q[3]);
```

(%o16) $\cos(q_3) \longrightarrow q_3$

```
(%i17)
```

Cinematica diretta:

```
(%i17) Q[stanford](q1,q2,q3,L1,L2):=
                            Q(q1,L1,-%pi/2,0).
                            Q(q2,L2,%pi/2,0).
                            Q(-%pi/2,q3,0,0)
                            ;
```

(%o17) $Q_{\text{stanford}}(q1, q2, q3, L1, L2) := Q\left(q1, L1, \frac{-\pi}{2}, 0\right) \cdot Q\left(q2, L2, \frac{\pi}{2}, 0\right) \cdot Q\left(\frac{-\pi}{2}, q3, 0, 0\right)$

```
(%i18) Qstanford:Q[stanford](q[1],q[2],q[3],L[1],L[2]);
```

(%o18) $$\begin{pmatrix} \sin(q_1) & \cos(q_1)\cos(q_2) & \cos(q_1)\sin(q_2) & q_3\cos(q_1)\sin(q_2) - L_2\sin(q_1) \\ -\cos(q_1) & \sin(q_1)\cos(q_2) & \sin(q_1)\sin(q_2) & q_3\sin(q_1)\sin(q_2) + L_2\cos(q_1) \\ 0 & -\sin(q_2) & \cos(q_2) & q_3\cos(q_2) + L_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
(%i19) letsimp(Qstanford);
```

$$\text{(\%o19)} \begin{pmatrix} s_1 & c_1\,c_2 & c_1\,s_2 & c_1\,s_2\,q_3 - s_1\,L_2 \\ -c_1 & s_1\,c_2 & s_1\,s_2 & s_1\,s_2\,q_3 + c_1\,L_2 \\ 0 & -s_2 & c_2 & c_2\,q_3 + L_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
(%i20)
```

Cinematica inversa robot Stanford

Al fine di risolvere il problema di cinematica inversa del robot sferico di II tipo (Stanford) occorre risolevere il problema di posizione ed orientamente inverso.Inizialmente individuare lo spazio di lavoro, le soluzioni generiche, singolari ed infine le variabili di giunto q_i ed in seguito determinare l'orientamento del robot.

Dalla cinematica diretta sappiamo che:

$$Q_{\text{Stanford}} = \begin{pmatrix} s_1 & c_1\,c_2 & c_1\,s_2 & c_1\,s_2\,q_3 - s_1\,L_2 \\ -c_1 & s_1\,c_2 & s_1\,s_2 & s_1\,s_2\,q_3 + c_1\,L_2 \\ 0 & -s_2 & c_2 & c_2\,q_3 + L_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Cinematica inversa di posizione

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} c_1\,s_2\,q_3 - s_1\,L_2 \\ s_1\,s_2\,q_3 + c_1\,L_2 \\ c_2\,q_3 + L_1 \end{pmatrix}$$

$$\begin{cases} x = c_1\,s_2\,q_3 - s_1\,L_2 \\ y = s_1\,s_2\,q_3 + c_1\,L_2 \end{cases}$$

$$x^2 + y^2 = c_1^2 s_2^2 q_3^2 + s_1^2 L_2^2 - 2 c_1 s_2 q_3 s_1 c_1 + s_1^2 s_2^2 q_3^2 + c_1^2 L_2^2 + 2 s_1\,s_2\,q_3 c_1\,L_2$$

$$\begin{cases} x^2 + y^2 = s_2^2 q_3^2 + L_2^2 \\ (z - L_1)^2 = c_2^2 q_3^2 \end{cases} \rightarrow \quad x^2 + y^2 + (z - L_1)^2 = q_3^2 + L_2^2$$

L'equazione $x^2 + y^2 + (z - L_1)^2 - L_2^2 = q_3^2$ descrive lo spazio operativo del robot Stanford. In particolare, rappresenta una sfera cava di centro $\begin{pmatrix} 0 \\ 0 \\ L_1 \end{pmatrix}$ e raggio r$=\sqrt{L_2^2 + q_3^3}$. Il raggio interno si ottiene con $q_3 = 0 \rightarrow r = L_2$ e il raggio esterno con $q_3 = \pm\infty \rightarrow r = +\infty$.

Per determinare lo spazio operativo, occorre determinare i punti di singolarità e le soluzioni generiche:

$$q_3 = \pm\sqrt{x^2 + y^2 + (z - L_1)^2 - L_2^2}$$

Si ha una singolarità se: $x^2 + y^2 + (z - L_1)^2 - L_2^2 = 0$.

Inoltre, la variabile di giunto $q_3 \neq 0$. Quindi:

$$c_2 = \frac{z - L_1}{q_3}$$

$$s_2 = \pm\sqrt{\frac{x^2 + y^2 - L_2^2}{q_3^2}}$$

Affinche s_2 sia definito deve essere valida la relazione $x^2 + y^2 \geqslant L_2^2$. Quindi si hanno due soluzioni generiche ed una singolarità per $x^2 + y^2 = L_2^2$ che ci permette di definire lo spazio operativo finale. Esso è una sfera attraversata da un cilindro cavo di raggio L_2.

In aggiunta:

$$q_2 = \text{atan2}\left(\pm\sqrt{\frac{x^2 + y^2 - L_2^2}{q_3^2}}, \frac{z - L_1}{q_3}\right)$$

Al fine di determinare l'espressione della variabile di giunto q_1, si impone:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} c_1 & -s_1 \\ s_1 & c_1 \end{pmatrix} \begin{pmatrix} q_3 s_2 \\ L_2 \end{pmatrix}$$

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} q_3 s_2 & -L_2 \\ L_2 & q_3 s_2 \end{pmatrix} \begin{pmatrix} c_1 \\ s_1 \end{pmatrix}$$

Poiché $\det\begin{pmatrix} q_3 s_2 & -L_2 \\ L_2 & q_3 s_2 \end{pmatrix} = q_3^2 s_2^2 + L_2^2 \neq 0$, è possibile effettuare l'inversa ed ottenere:

$$\begin{pmatrix} c_1 \\ s_1 \end{pmatrix} = \frac{1}{q_3^2 s_2^2 + L_2^2} \begin{pmatrix} q_3 s_2 & L_2 \\ -L_2 & q_3 s_2 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

$$\begin{cases} c_1 = \frac{q_3 s_2 x + L_2 y}{q_3^2 s_2^2 + L_2^2} \\ s_1 = \frac{-L_2 x + q_3 s_2 y}{q_3^2 s_2^2 + L_2^2} \end{cases}$$

$$q_1 = \text{atan2}\left(\frac{-L_2 x + q_3 s_2 y}{q_3^2 s_2^2 + L_2^2}, \frac{q_3 s_2 x + L_2 y}{q_3^2 s_2^2 + L_2^2}\right) = \text{atan2}(-L_2 x + q_3 s_2 y, q_3 s_2 x + L_2 y)$$

Orientamento inverso

La risoluzione del problema di orientamento inverso si basa sulla scelta di una terna di Eulero o nautica in condizione non singolari, se possibile.

$$R_{\text{Stanford}} = \begin{pmatrix} s_1 & c_1\,c_2 & c_1\,s_2 \\ -c_1 & s_1\,c_2 & s_1\,s_2 \\ 0 & -s_2 & c_2 \end{pmatrix}$$

$$R_{\text{zyx}} = \begin{pmatrix} c_y c_z & \cdots\cdots\cdots & \\ c_y s_z & \cdots\cdots\cdots & \\ -s_y & s_x c_y & c_x c_y \end{pmatrix}$$

$$s_y = 0 \rightarrow c_y = \pm 1 \qquad\qquad \rightarrow \phi_y = \text{atan2}(0, \pm 1) = \begin{cases} 0 \\ \pi \end{cases}$$

$$\begin{cases} s_x\,c_y = -s_2 \\ c_x\,c_y = c_2 \end{cases} \rightarrow \begin{cases} \pm s_x = s_2 \\ \pm c_{\text{yx}} = c_2 \end{cases} \rightarrow \begin{cases} s_{\text{yx}} = \mp s_2 \\ c_x = \pm c_2 \end{cases} \rightarrow \phi_y = \text{atan2}(\mp s_2, \pm c_2) = \begin{cases} -q_2 \\ -q_2 + \pi \end{cases}$$

$$\begin{cases} c_y\,c_z = s_1 \\ c_y\,s_z = -c_1 \end{cases} \rightarrow \begin{cases} \pm c_z = s_1 \\ \pm s_z = -c_1 \end{cases} \rightarrow \begin{cases} c_z = \pm s_1 \\ s_z = \mp c_1 \end{cases} \rightarrow \phi_z = \text{atan2}(\mp c_1, \pm c_1) = \begin{cases} q_1 - \frac{\pi}{2} \\ q_1 + \frac{\pi}{2} \end{cases}$$

Riassumendo:

$$\begin{pmatrix} -q_2 \\ 0 \\ q_1 - \frac{\pi}{2} \end{pmatrix}, \begin{pmatrix} -q_2 + \pi \\ \pi \\ q_1 + \frac{\pi}{2} \end{pmatrix}$$

In alternativa, tramite una scelta di una terna di Eulero:

$$R_{\text{zyz}} = \begin{pmatrix} \cdots & \cdots & \cos(\alpha)\sin(\beta) \\ \cdots & \cdots & \sin(\alpha)\sin(\beta) \\ -\sin(\beta)\cos(\gamma) & \sin(\beta)\sin(\gamma) & \cos(\beta) \end{pmatrix}$$

$$\cos(\beta) = c_2 \neq \pm 1 \rightarrow q_2 \neq \begin{cases} 0 \\ \pi \end{cases}$$

Supponiamo che $q_2 \neq \begin{cases} 0 \\ \pi \end{cases} \rightarrow \cos(\beta) \neq \pm 1 \rightarrow \sin(\beta) \neq 0$:

$$\sin(\beta) = \pm\sqrt{1 - \cos(\beta)^2} = \pm\sqrt{1 - \cos(q_2)^2} = \pm\sin(q_2)$$

$$\beta = \text{atan2}(\pm\sin(q_2), \cos(q_2)) = \begin{cases} q_2 \\ -q_2 \end{cases}$$

$$\begin{cases} \sin(\beta)\sin(\gamma) = 0 \\ -\sin(\beta)\cos(\gamma) = -s_2 \end{cases} \rightarrow \begin{cases} \sin(\gamma) = 0 \\ \cos(\gamma) = \pm 1 \end{cases} \rightarrow \gamma = \text{atan2}(0, \pm 1) = \begin{cases} 0 \\ \pi \end{cases}$$

$$\begin{cases} \cos(\alpha)\sin(\beta) = c_1 s_2 \\ \sin(\alpha)\sin(\beta) = s_1 s_2 \end{cases} \rightarrow \begin{cases} \cos(\alpha) = \pm c_1 \\ \sin(\alpha) = \pm s_1 \end{cases} \rightarrow \alpha = \text{atan2}(\pm s_1, \pm c_1) = \begin{cases} q_1 \\ q_1 + \pi \end{cases}$$

Riassumendo:

$$\begin{pmatrix} q_1 \\ q_2 \\ 0 \end{pmatrix}, \begin{pmatrix} q_1 + \pi \\ -q_2 \\ \pi \end{pmatrix}$$

```
(%i20) isRotation(M):=block([MC,res],
                          I:ident(3),
                          MC:ident(3),
                          for i:1 thru 3 do
                          (
                          for j:1 thru 3 do
                             (
                                MC[i][j]:M[i][j]
                                )
                             ),
                          MMT:trigsimp(expand(MC.transpose(MC))),
                          detM:trigsimp(expand(determinant(MC))),

                          if MMT=I  and detM=1
                             then(

                                  return(res:1)
                                  )

                          else(

                                res: "R is not rotation matrix"
                                )
                          )
```

(%o20) isRotation$(M) :=$ **block** ([MC, res], I: ident(3), MC: ident(3), **for** i **thru** 3 **do for** j **thru** 3 **do** $(\mathrm{MC}_i)_j$: $(M_i)_j$, MMT: trigsimp(expand(MC · transpose(MC))), detM: trigsimp(expand(determinant(MC))), **if** MMT $= I \wedge$ detM $= 1$ **then** return(res: 1) **else** res: R is not rotation matrix)

$-L_2x + q_3s_2y,\ q_3s_2x + L_2y$

```
(%i27) calculate(x,y,L1,L2,z):=block(
       [q3plus,q3minus,q2plus,q2minus,q1plus,q1minus,res],
       if x^(2)+y^(2)+(z-L1)^2=L2^(2) then print("La soluzione è singolare")
       else
       (
       q3value:sqrt(trigreduce(trigexpand(ratsimp(x^(2)+y^(2)+(z-L1)^(2)-
       L2^(2))))),
       q3plus:trigreduce(trigexpand(ratsimp(q3value))),
       q3minus:-trigreduce(trigexpand(ratsimp(q3value))),
       s2plus:trigreduce(trigexpand(ratsimp(sqrt((x^(2)+y^(2)-L2^(2))/
       q3value^2)))),
       s2minus:-s2plus,
       c2plus:trigreduce(trigexpand(ratsimp((z-L1)/q3plus))),
       c2minus:trigreduce(trigexpand(ratsimp((z-L1)/q3minus))),
       q2plus:atan2(s2plus,c2plus),
       q2minus:atan2(s2minus,c2minus),
       s1plus:trigreduce(trigexpand(ratsimp(-L2*x+q3plus*s2plus*y))),
       s1minus:trigreduce(trigexpand(ratsimp(-L2*x+q3minus*s2minus*y))),
       c1plus:trigreduce(trigexpand(ratsimp(q3plus*s2plus*x+L2*y))),
       c1minus:trigreduce(trigexpand(ratsimp(q3minus*s2minus*x+L2*y))),
       q1plus:atan2(s1plus,c1plus),
       q1minus:atan2(s1minus,c1minus),
       res:[[q1plus,q2plus,q3plus],[q1minus,q2minus,q3minus]]
       )
       )
```

(%o27) calculate$(x, y, L1, L2, z)$:= **block** $\Bigg($ [$q3$plus, $q3$minus, $q2$plus, $q2$minus, $q1$plus, $q1$minus, res], **if** $x^2+y^2+(z-L1)^2 = L2^2$ **then** print(La soluzione è singolare) **else** $\Bigg($ $q3$value: $\sqrt{\text{trigreduce}(\text{trigexpand}(\text{ratsimp}(x^2+y^2+(z-L1)^2-L2^2)))}$, $q3$plus: trigreduce(trigexpand(ratsimp($q3$value))), $q3$minus: $-$trigreduce(trigexpand(ratsimp($q3$value))), $s2$plus: $\text{trigreduce}\left(\text{trigexpand}\left(\text{ratsimp}\left(\sqrt{\frac{x^2+y^2-L2^2}{q3\text{value}^2}}\right)\right)\right)$, $s2$minus: $-s2$plus, $c2$plus: $\text{trigreduce}\left(\text{trigexpand}\left(\text{ratsimp}\left(\frac{z-L1}{q3\text{plus}}\right)\right)\right)$, $c2$minus: $\text{trigreduce}\left(\text{trigexpand}\left(\text{ratsimp}\left(\frac{z-L1}{q3\text{minus}}\right)\right)\right)$, $q2$plus: atan2($s2$plus, $c2$plus), $q2$minus: atan2($s2$minus, $c2$minus), $s1$plus: trigreduce(trigexpand(ratsimp($(-L2)\,x + q3\text{plus}\,s2\text{plus}\,y$))), $s1$minus: trigreduce(trigexpand(ratsimp($(-L2)\,x + q3\text{minus}\,s2\text{minus}\,y$))), $c1$plus: trigreduce(trigexpand(ratsimp($q3\text{plus}\,s2\text{plus}\,x + L2\,y$))), $c1$minus: trigreduce(trigexpand(ratsimp($q3\text{minus}\,s2\text{minus}\,x + L2\,y$))), $q1$plus: atan2($s1$plus, $c1$plus), $q1$minus: atan2($s1$minus, $c1$minus), res: [[$q1$plus, $q2$plus, $q3$plus], [$q1$minus, $q2$minus, $q3$minus]] $\Bigg)\Bigg)$

(%i77)
```
orientation(Qdiretta):=block([sx,cx,sy,cy,phiy1,phiy2,phiz1,phiz2,phix1,
phix2,sz,sxfirst,second,res],

                                   rotation:isRotation(Qdiretta),
                                   if rotation=1 then(
                                   sy:Qdiretta[3][1],

                                   if sy=1 or sy=-1 then "soluzione singolare"
                                   else(
                                   cy:sqrt(1-sy^2),
                                   phiy1:atan2(-sy,cy),
                                   phiy2:atan2(-sy,-cy),

                                   sx:Qdiretta[3][2]/cy,
                                   cx:Qdiretta[3][3]/cy,
                                   phix1:atan2(sx,cx),
                                   phix2:phix1+%pi,
                                   cz:Qdiretta[1][1]/cy,
                                   sz:Qdiretta[2][1]/cy,
                                   phiz1:atan2(sz,cz)-%pi/2,
                                   print(phiz1),
                                   phiz2:phiz1+(%pi/2),
                                   first:[phix1,phiy1,phiz1],
                                   second:[phix2,phiy2,phiz2],

                                   res:[first,second])
                                   )
                                   else res:rotation

);
```

(%o77) orientation(Qdiretta) := **block** $\Big($ [sx, cx, sy, cy, phiy1, phiy2, phiz1, phiz2, phix1, phix2, sz, sxfirst, second, res], rotation: isRotation(Qdiretta), **if** rotation $=1$ **then** $\Big($ sy: $(\text{Qdiretta}_3)_1$, **if** sy $=$

$1 \vee \mathrm{sy} = -1$ **then** soluzione singolare **else** $\Big($cy: $\sqrt{1-\mathrm{sy}^2}$, phiy1: atan2$(-\mathrm{sy}, \mathrm{cy})$, phiy2: atan2$(-\mathrm{sy}, -\mathrm{cy})$, sx: $\frac{(\mathrm{Qdiretta}_3)_2}{\mathrm{cy}}$, cx: $\frac{(\mathrm{Qdiretta}_3)_3}{\mathrm{cy}}$, phix1: atan2(sx, cx), phix2: phix1 $+ \pi$, cz: $\frac{(\mathrm{Qdiretta}_1)_1}{\mathrm{cy}}$, sz: $\frac{(\mathrm{Qdiretta}_2)_1}{\mathrm{cy}}$, phiz1: atan2(sz, cz) $- \frac{\pi}{2}$, print(phiz1), phiz2: phiz1 $+ \frac{\pi}{2}$, first: [phix1, phiy1, phiz1], second: [phix2, phiy2, phiz2], res: [first, second]$\Big)\Big)$ **else** res: rotation$\Big)$

(%i78)
```
invStanford(Qdiretta,L1,L2):=block(
                    [x,y,z,pos1,pos2,orien1,orien2,res],
                    x:Qdiretta[1][4],
                        y:Qdiretta[2][4],
                        z:Qdiretta[3][4],
                        pos:calculate(x,y,L1,L2,z),
                        orien:orientation(Qdiretta),
                        orien1:orien[1],
                        orien2:orien[2],
                        pos1:pos[1],
                        pos2:pos[2],
                        res:[pos1,pos2,orien1,orien2]
                        );
```

(%o78) invStanford(Qdiretta, $L1$, $L2$) := **block** ([x, y, z, pos1, pos2, orien1, orien2, res], x: $(\mathrm{Qdiretta}_1)_4$, y: $(\mathrm{Qdiretta}_2)_4$, z: $(\mathrm{Qdiretta}_3)_4$, pos: calculate(x, y, $L1$, $L2$, z), orien: orientation(Qdiretta), orien1: orien_1, orien2: orien_2, pos1: pos_1, pos2: pos_2, res: [pos1, pos2, orien1, orien2])

(%i79)
```
Qstanford:Q[stanford](%pi/3,%pi/3,5,10,10);
```

$$\text{(\%o79)}\quad \begin{pmatrix} \frac{\sqrt{3}}{2} & \frac{1}{4} & \frac{\sqrt{3}}{4} & -\frac{5\,3^{\frac{3}{2}}}{4} \\ -\frac{1}{2} & \frac{\sqrt{3}}{4} & \frac{3}{4} & \frac{35}{4} \\ 0 & -\frac{\sqrt{3}}{2} & \frac{1}{2} & \frac{25}{2} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

(%i80)
```
invStanford(Qstanford,10,10);
```

$$-\frac{2\,\pi}{3}$$

$$\text{(\%o80)}\quad \left[\left[\frac{\pi}{3}, \frac{\pi}{3}, 5\right], \left[\frac{\pi}{3}, -\frac{2\,\pi}{3}, -5\right], \left[-\frac{\pi}{3}, 0, -\frac{2\,\pi}{3}\right], \left[\frac{2\,\pi}{3}, \pi, -\frac{\pi}{6}\right]\right]$$

(%i81)
```
Qstanford:Q[stanford](q[1],q[2],q[3],L[1],L[2]);
```

$$\text{(\%o81)}\quad \begin{pmatrix} \sin(q_1) & \cos(q_1)\cos(q_2) & \cos(q_1)\sin(q_2) & q_3\cos(q_1)\sin(q_2) - L_2\sin(q_1) \\ -\cos(q_1) & \sin(q_1)\cos(q_2) & \sin(q_1)\sin(q_2) & q_3\sin(q_1)\sin(q_2) + L_2\cos(q_1) \\ 0 & -\sin(q_2) & \cos(q_2) & q_3\cos(q_2) + L_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

(%i39)
```
invStanford(Qstanford,L[1],L[2]);
```

$$\text{(\%o39)}\quad \left[\left[-\mathrm{atan2}\left(\frac{L_2\,q_3\sin(q_2+q_1)}{2} + \frac{q_3\sqrt{q_3^2 - q_3^2\cos(2\,q_2)}\cos(q_2+q_1)}{2^{\frac{3}{2}}} + \frac{L_2\,q_3\sin(q_2-q_1)}{2} - \right.\right.\right.$$
$$\frac{q_3\sqrt{q_3^2 - q_3^2\cos(2\,q_2)}\cos(q_2-q_1)}{2^{\frac{3}{2}}} - \frac{L_2\cos(q_1)\sqrt{q_3^2 - q_3^2\cos(2\,q_2)}}{\sqrt{2}} - L_2^2\sin(q_1),$$

$$\frac{q_3\sqrt{q_3^2-q_3^2\cos(2\,q_2)}\sin(q_2+q_1)}{2^{\frac{3}{2}}}-\frac{L_2\,q_3\cos(q_2+q_1)}{2}+\frac{q_3\sqrt{q_3^2-q_3^2\cos(2\,q_2)}\sin(q_2-q_1)}{2^{\frac{3}{2}}}+$$

$$\left.\frac{L_2\,q_3\cos(q_2-q_1)}{2}-\frac{L_2\sin(q_1)\sqrt{q_3^2-q_3^2\cos(2\,q_2)}}{\sqrt{2}}+L_2^2\cos(q_1)\right),\mathrm{atan2}\left(\frac{\sqrt{q_3^2-q_3^2\cos(2\,q_2)}}{\sqrt{2}\,|q_3|},\right.$$

$$\left.\left.\frac{q_3\cos(q_2)}{|q_3|}\right),|q_3|\right],\left[-\mathrm{atan2}\left(\frac{L_2\,q_3\sin(q_2+q_1)}{2}+\frac{q_3\sqrt{q_3^2-q_3^2\cos(2\,q_2)}\cos(q_2+q_1)}{2^{\frac{3}{2}}}+\right.\right.$$

$$\frac{L_2\,q_3\sin(q_2-q_1)}{2}-\frac{q_3\sqrt{q_3^2-q_3^2\cos(2\,q_2)}\cos(q_2-q_1)}{2^{\frac{3}{2}}}-\frac{L_2\cos(q_1)\sqrt{q_3^2-q_3^2\cos(2\,q_2)}}{\sqrt{2}}-$$

$$L_2^2\sin(q_1),\frac{q_3\sqrt{q_3^2-q_3^2\cos(2\,q_2)}\sin(q_2+q_1)}{2^{\frac{3}{2}}}-\frac{L_2\,q_3\cos(q_2+q_1)}{2}+$$

$$\frac{q_3\sqrt{q_3^2-q_3^2\cos(2\,q_2)}\sin(q_2-q_1)}{2^{\frac{3}{2}}}+\frac{L_2\,q_3\cos(q_2-q_1)}{2}-\frac{L_2\sin(q_1)\sqrt{q_3^2-q_3^2\cos(2\,q_2)}}{\sqrt{2}}+$$

$$\left.\left.L_2^2\cos(q_1)\right),-\mathrm{atan2}\left(\frac{\sqrt{q_3^2-q_3^2\cos(2\,q_2)}}{\sqrt{2}\,|q_3|},-\frac{q_3\cos(q_2)}{|q_3|}\right),-|q_3|\right],[\mathrm{atan2}(\sin(q_2),\cos(q_2)),0,$$

$$\left.-\mathrm{atan2}(\sin(q_1),-\cos(q_1))],[-\mathrm{atan2}(\sin(q_2),-\cos(q_2)),\pi,\mathrm{atan2}(\sin(q_1),\cos(q_1))]\right]$$

(%i40)

Singolarità di velocità

```
Maxima 5.44.0 http://maxima.sourceforge.net
using Lisp SBCL 2.0.0
Distributed under the GNU Public License. See the file COPYING.
Dedicated to the memory of William Schelter.
The function bug_report() provides bug reporting information.
```

(%i1)

(%i1) `x:q[3]*cos (q[1])*sin (q[2])-L[2]*sin (q[1]);`

(%o1) $q_3\cos(q_1)\sin(q_2)-L_2\sin(q_1)$

(%i2) `y:q[3]*sin (q[1])*sin (q[2])+L[2]*cos (q[1]);`

(%o2) $q_3\sin(q_1)\sin(q_2)+L_2\cos(q_1)$

(%i3) `z:q[3]*cos (q[2])+L[1];`

(%o3) $q_3\cos(q_2)+L_1$

(%i23)
```
J:J:matrix([diff(x,q[1]),diff(x,q[2]),diff(x,q[3])],
           [diff(y,q[1]),diff(y,q[2]),diff(y,q[3])],
           [diff(z,q[1]),diff(z,q[2]),diff(z,q[3])]);
```

(%o23) $$\begin{pmatrix} -q_3\sin(q_1)\sin(q_2)-L_2\cos(q_1) & q_3\cos(q_1)\cos(q_2) & \cos(q_1)\sin(q_2) \\ q_3\cos(q_1)\sin(q_2)-L_2\sin(q_1) & q_3\sin(q_1)\cos(q_2) & \sin(q_1)\sin(q_2) \\ 0 & -q_3\sin(q_2) & \cos(q_2) \end{pmatrix}$$

(%i5) `dJ:trigsimp(expand(determinant(J)));`

(%o5) $-q_3^2 \sin(q_2)$

Si hanno singolarità per:

$$q_3 = 0 \vee q_2 = 0$$

Caso $q_3 = 0$:

(%i6) `Jq3:subst(q[3]=0,J);`

(%o6) $\begin{pmatrix} -L_2\cos(q_1) & 0 & \cos(q_1)\sin(q_2) \\ -L_2\sin(q_1) & 0 & \sin(q_1)\sin(q_2) \\ 0 & 0 & \cos(q_2) \end{pmatrix}$

(%i7) `nullspace(Jq3)`

Proviso: notequal$(-L_2\cos(q_1), 0) \wedge$ notequal$(-L_2\cos(q_1)\cos(q_2), 0)$

(%o7) $\operatorname{span}\left(\begin{pmatrix} 0 \\ -L_2\cos(q_1)\cos(q_2) \\ 0 \end{pmatrix}\right)$

$\ker(J(q_3=0)) = \Im m\left\{\begin{pmatrix} 0 \\ -L_2\cos(q_1)\cos(q_2) \\ 0 \end{pmatrix}\right\} \to v = k\begin{pmatrix} 0 \\ -L_2\cos(q_1)\cos(q_2) \\ 0 \end{pmatrix} \forall k,\ q_1,\ q2 \neq \begin{cases} \frac{\pi}{2} \\ \frac{3\pi}{2} \end{cases} \to w = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$

Se $q_1 = \frac{\pi}{2} \wedge q_2 \neq \frac{\pi}{2}$:

(%i8) `Jq31:subst(q[1]=%pi/2,Jq3);`

(%o8) $\begin{pmatrix} 0 & 0 & 0 \\ -L_2 & 0 & \sin(q_2) \\ 0 & 0 & \cos(q_2) \end{pmatrix}$

(%i9) `nullspace(Jq31);`

Proviso: notequal$(-L_2, 0) \wedge$ notequal$(-L_2\cos(q_2), 0)$

(%o9) $\operatorname{span}\left(\begin{pmatrix} 0 \\ -L_2\cos(q_2) \\ 0 \end{pmatrix}\right)$

Si hanno singolarità di velocità se $v \in \operatorname{Im}\left\{\begin{pmatrix} 0 \\ -L_2\cos(q_2) \\ 0 \end{pmatrix}\right\}$.

Se $q_1 \neq \frac{\pi}{2} \wedge q_2 = \frac{\pi}{2}$:

(%i10) `Jq32:subst(q[2]=%pi/2,Jq3);`

(%o10) $\begin{pmatrix} -L_2\cos(q_1) & 0 & \cos(q_1) \\ -L_2\sin(q_1) & 0 & \sin(q_1) \\ 0 & 0 & 0 \end{pmatrix}$

(%i11) `nullspace(Jq32);`

Proviso: notequal$(\cos(q_1), 0)$

(%o11) $\operatorname{span}\left(\begin{pmatrix} 0 \\ \cos(q_1) \\ 0 \end{pmatrix}, \begin{pmatrix} \cos(q_1) \\ 0 \\ L_2\cos(q_1) \end{pmatrix}\right)$

Si hanno singolarità di velocità se $v \in \operatorname{Im}\left\{\begin{pmatrix} 0 \\ \cos(q_1) \\ 0 \end{pmatrix}, \begin{pmatrix} \cos(q_1) \\ 0 \\ L_2\cos(q_1) \end{pmatrix}\right\}$.

Se $q_1 = \frac{\pi}{2} \wedge q_2 = \frac{\pi}{2}$:

(%i12) `Jq32:subst([q[1]=%pi/2,q[2]=%pi/2],Jq3);`

$$\text{(\%o12)}\quad \begin{pmatrix} 0 & 0 & 0 \\ -L_2 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$$

(%i13) `nullspace(Jq32);`

$$\text{(\%o13)}\quad \operatorname{span}\left(\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ L_2 \end{pmatrix}\right)$$

Si hanno singolarità di velocità se $v \in \operatorname{Im}\left\{\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ L_2 \end{pmatrix}\right\}$.

Inoltre, se $q_2 = 0$:

(%i19) `Jq2:subst(q[2]=0,J);`

$$\text{(\%o19)}\quad \begin{pmatrix} -L_2 \cos(q_1) & q_3 \cos(q_1) & 0 \\ -L_2 \sin(q_1) & q_3 \sin(q_1) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

(%i20) `nullspace(Jq2);`

Proviso: notequal$(-L_2 \cos(q_1), 0) \wedge$ notequal$(-L_2 \cos(q_1), 0)$

$$\text{(\%o20)}\quad \operatorname{span}\left(\begin{pmatrix} -q_3 \cos(q_1) \\ -L_2 \cos(q_1) \\ 0 \end{pmatrix}\right)$$

$$\ker(J(q_2=0)) = \Im m\left\{\begin{pmatrix} -q_3 \cos(q_1) \\ -L_2 \cos(q_1) \\ 0 \end{pmatrix}\right\} \to v = k\begin{pmatrix} -q_3 \cos(q_1) \\ -L_2 \cos(q_1) \\ 0 \end{pmatrix} \forall k, q_{3;} \quad q_1 \neq \frac{\pi}{2} \to w = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

Se $q_1 = \frac{\pi}{2}$:

(%i21) `Jq21:subst(q[1]=%pi/2,Jq2);`

$$\text{(\%o21)}\quad \begin{pmatrix} 0 & 0 & 0 \\ -L_2 & q_3 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

(%i22) `nullspace(Jq21);`

Proviso: notequal$(-L_2, 0) \wedge$ notequal$(-L_2, 0)$

$$\text{(\%o22)}\quad \operatorname{span}\left(\begin{pmatrix} -q_3 \\ -L_2 \\ 0 \end{pmatrix}\right)$$

Si hanno singolarità di velocità se $v \in \operatorname{Im}\left\{\begin{pmatrix} -q_3 \\ -L_2 \\ 0 \end{pmatrix}\right\}$.

Singolarità di forza

(%i24) `J:-transpose(J)`

$$\text{(\%o24)}\quad \begin{pmatrix} q_3 \sin(q_1) \sin(q_2) + L_2 \cos(q_1) & L_2 \sin(q_1) - q_3 \cos(q_1) \sin(q_2) & 0 \\ -q_3 \cos(q_1) \cos(q_2) & -q_3 \sin(q_1) \cos(q_2) & q_3 \sin(q_2) \\ -\cos(q_1) \sin(q_2) & -\sin(q_1) \sin(q_2) & -\cos(q_2) \end{pmatrix}$$

(%i26) `dJ:trigsimp(expand(determinant(J)));`

$$\text{(\%o26)}\quad q_3^2 \sin(q_2)$$

Si hanno singolarità di forza per $q_3 = 0 \vee q_2 = 0$.

Se $q_3 = 0$:

```
(%i27) Jq3:subst(q[3]=0,J);
```

$$(\%o27)\quad \begin{pmatrix} L_2\cos(q_1) & L_2\sin(q_1) & 0 \\ 0 & 0 & 0 \\ -\cos(q_1)\sin(q_2) & -\sin(q_1)\sin(q_2) & -\cos(q_2) \end{pmatrix}$$

```
(%i28) nullspace(Jq3);
```

Proviso: $\mathrm{notequal}(L_2\cos(q_1),0) \wedge \mathrm{notequal}(-L_2\cos(q_1)\cos(q_2),0)$

$$(\%o28)\quad \mathrm{span}\left(\begin{pmatrix} L_2\sin(q_1)\cos(q_2) \\ -L_2\cos(q_1)\cos(q_2) \\ 0 \end{pmatrix}\right)$$

Se $q_1=\frac{\pi}{2} \wedge q_2 \neq \frac{\pi}{2}$:

```
(%i29) Jq31:subst(q[1]=%pi/2,Jq3);
```

$$(\%o29)\quad \begin{pmatrix} 0 & L_2 & 0 \\ 0 & 0 & 0 \\ 0 & -\sin(q_2) & -\cos(q_2) \end{pmatrix}$$

```
(%i30) nullspace(Jq31);
```

Proviso: $\mathrm{notequal}(L_2,0) \wedge \mathrm{notequal}(-L_2\cos(q_2),0)$

$$(\%o30)\quad \mathrm{span}\left(\begin{pmatrix} -L_2\cos(q_2) \\ 0 \\ 0 \end{pmatrix}\right)$$

Si hanno singolarità di forza se $\tau \in \mathrm{Im}\left\{\begin{pmatrix} -L_2\cos(q_2) \\ 0 \\ 0 \end{pmatrix}\right\}$.

Se $q_2=\frac{\pi}{2} \wedge q_1 \neq \frac{\pi}{2}$:

```
(%i31) Jq32:subst(q[2]=%pi/2,Jq3);
```

$$(\%o31)\quad \begin{pmatrix} L_2\cos(q_1) & L_2\sin(q_1) & 0 \\ 0 & 0 & 0 \\ -\cos(q_1) & -\sin(q_1) & 0 \end{pmatrix}$$

```
(%i32) nullspace(Jq32);
```

Proviso: $\mathrm{notequal}(L_2\cos(q_1),0)$

$$(\%o32)\quad \mathrm{span}\left(\begin{pmatrix} 0 \\ 0 \\ -L_2\sin(q_1) \end{pmatrix}, \begin{pmatrix} -L_2\sin(q_1) \\ L_2\cos(q_1) \\ 0 \end{pmatrix}\right)$$

Si hanno singolarità di forza se $\tau \in \mathrm{Im}\left\{\begin{pmatrix} 0 \\ 0 \\ -L_2\sin(q_1) \end{pmatrix}, \begin{pmatrix} -L_2\sin(q_1) \\ L_2\cos(q_1) \\ 0 \end{pmatrix}\right\}$.

Se $q_2=\frac{\pi}{2} \wedge q_1 = \frac{\pi}{2}$:

```
(%i33) Jq32:subst([q[2]=%pi/2,q[1]=%pi/2],Jq3);
```

$$(\%o33)\quad \begin{pmatrix} 0 & L_2 & 0 \\ 0 & 0 & 0 \\ 0 & -1 & 0 \end{pmatrix}$$

```
(%i34) nullspace(Jq32);
```

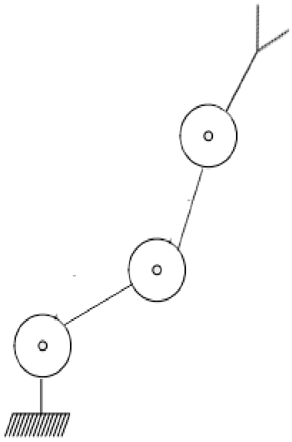
Proviso: $\mathrm{notequal}(L_2,0)$

$$(\%o34)\quad \mathrm{span}\left(\begin{pmatrix} 0 \\ 0 \\ L_2 \end{pmatrix}, \begin{pmatrix} L_2 \\ 0 \\ 0 \end{pmatrix}\right)$$

Si hanno singolarità di forza se $\tau \in \mathrm{Im}\left\{\begin{pmatrix} 0 \\ 0 \\ L_2 \end{pmatrix}, \begin{pmatrix} L_2 \\ 0 \\ 0 \end{pmatrix}\right\}$

(%i35)

Cinematica inversa 3 DOF Planare RRR



$$P = \begin{pmatrix} x \\ y \end{pmatrix} := \text{coordinate end} - \text{effector}; \phi := \text{orientamento}$$

$$R_{03} = R(q_1 + q_2 + q_3)$$

$$d_{03} = R_{03} \begin{pmatrix} L_3 \\ 0 \end{pmatrix} + R(q_1 + q_2) \begin{pmatrix} L_2 \\ 0 \end{pmatrix} + R(q_1) \begin{pmatrix} L_1 \\ 0 \end{pmatrix}$$

Dato $\begin{pmatrix} \mathrm{x} \\ \mathrm{y} \end{pmatrix}$ si determina q_1, q_2, q_3:

$$\begin{cases} \begin{pmatrix} x \\ y \end{pmatrix} = R_{123} \begin{pmatrix} L_3 \\ 0 \end{pmatrix} + R_{12} \begin{pmatrix} L_2 \\ 0 \end{pmatrix} + R_1 \begin{pmatrix} L_1 \\ 0 \end{pmatrix} \\ \phi = q_1 + q_2 + q_3 \end{cases}$$

$$\begin{pmatrix} x \\ y \end{pmatrix} = R(\phi) \begin{pmatrix} L_3 \\ 0 \end{pmatrix} + R(q_1 + q_2) \begin{pmatrix} L_2 \\ 0 \end{pmatrix} + R(q_1) \begin{pmatrix} L_1 \\ 0 \end{pmatrix}$$

$$\begin{pmatrix} x \\ y \end{pmatrix} - R(\phi) \begin{pmatrix} L_3 \\ 0 \end{pmatrix} = R(q_1 + q_2) \begin{pmatrix} L_2 \\ 0 \end{pmatrix} + R(q_1) \begin{pmatrix} L_1 \\ 0 \end{pmatrix}$$

$$\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = R(q_1 + q_2) \begin{pmatrix} L_2 \\ 0 \end{pmatrix} + R(q_1) \begin{pmatrix} L_1 \\ 0 \end{pmatrix}$$

$$\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = R(q_1 + q_2) \begin{pmatrix} L_2 \\ 0 \end{pmatrix} + R(q_1) \begin{pmatrix} L_1 \\ 0 \end{pmatrix}$$

Poiché $R(q_1 + q_2) = R(q_1)\, R(q_2)$, è possibile mettere in evidenza $R(q_1)$:

$$\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = R(q_1) \left\{ R(q_2) \begin{pmatrix} L_2 \\ 0 \end{pmatrix} + \begin{pmatrix} L_1 \\ 0 \end{pmatrix} \right\}$$

La matrice di rotazione $R(q_1)$ ha non varia la norma del vettore $\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix}$ e $R(q_2)\begin{pmatrix} L_2 \\ 0 \end{pmatrix} + \begin{pmatrix} L_1 \\ 0 \end{pmatrix}$. Quindi possiamo imporre che abbiano la stessa norma. In particolare:

$$\left\| \begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} \right\|_2 = \left\| R(q_2)\begin{pmatrix} L_2 \\ 0 \end{pmatrix} + \begin{pmatrix} L_1 \\ 0 \end{pmatrix} \right\|_2$$

$$\begin{pmatrix} L_2 & 0 \end{pmatrix} R(q_2)^T R(q_2) \begin{pmatrix} L_2 \\ 0 \end{pmatrix} + \begin{pmatrix} L_1 & 0 \end{pmatrix} \begin{pmatrix} L_1 \\ 0 \end{pmatrix} + 2 \begin{pmatrix} L_1 & 0 \end{pmatrix} R(q_2) \begin{pmatrix} L_2 \\ 0 \end{pmatrix}$$

$$\hat{x}^2 + \hat{y}^2 = L_2^2 + L_1^2 + 2\, L_1\, L_2 \cos(q_2)$$

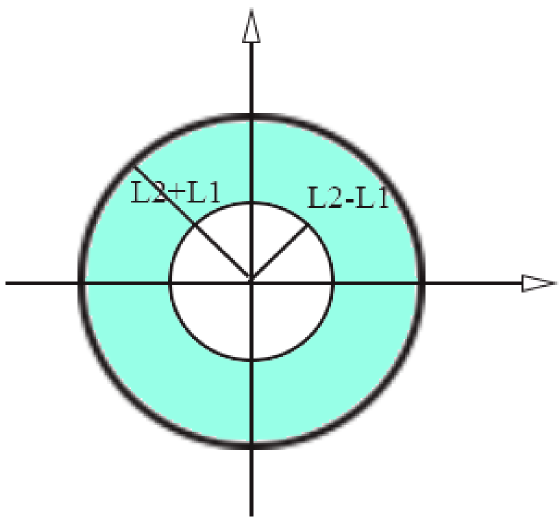
$$\cos(q_2) = \frac{\hat{x}^2 + \hat{y}^2 - L_2^2 - L_1^2}{2\, L_1\, L_2}$$

Poiché $-1 \leqslant \cos(q_2) \leqslant 1$, otteniamo infine lo spazio operativo del 2DOF planare:

$$-1 \leqslant \frac{\hat{x}^2 + \hat{y}^2 - L_2^2 - L_1^2}{2\, L_1\, L_2} \leqslant 1$$

$$L_2^2 + L_1^2 - 2\, L_1\, L_2 \leqslant \hat{x}^2 + \hat{y}^2 \leqslant 2\, L_1\, L_2 + L_2^2 + L_1^2$$

Le disequazioni delimitano due circonferenze di raggio $L_2 + L_1$ e $L_2 - L_1$, in cui la regione valida presa inconsiderazione è quella regione di spazio compresa tra le curve.



A questo punto è possibile determinare l'espressione della variabile di giunto q_2:

$$\cos(q_2) = \frac{\hat{x}^2 + \hat{y}^2 - L_2^2 - L_1^2}{2\, L_1\, L_2}$$

$$\sin(q_2) = \pm\sqrt{1 - \cos(q_2)^2} = \pm\sqrt{1 - \frac{\hat{x}^2 + \hat{y}^2 - L_2^2 - L_1^2}{2\, L_1\, L_2}}$$

In particolare, si hanno due soluzioni generiche se $\left| \frac{\hat{x}^2 + \hat{y}^2 - L_2^2 - L_1^2}{2\, L_1\, L_2} \right| \neq 1$. Nel caso in cui $\left| \frac{\hat{x}^2 + \hat{y}^2 - L_2^2 - L_1^2}{2\, L_1\, L_2} \right| = 1$, si ha una soluzione singolare.

A questo punto la quantità $R(q_2)\begin{pmatrix} L_2 \\ 0 \end{pmatrix}+\begin{pmatrix} L_1 \\ 0 \end{pmatrix}$ è nota:

$$\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix}=\begin{pmatrix} \cos(q_1) & -\sin(q_1) \\ \sin(q_1) & \cos(q_1) \end{pmatrix}\begin{pmatrix} L_2\cos(q_2)+L_1 \\ \sin(q_2)\,L_2 \end{pmatrix}$$

$$\begin{pmatrix} \cos(q_1) \\ \sin(q_1) \end{pmatrix}=\frac{1}{(L_2\cos(q_2)+L_1)^2+L_2^2\sin(q_2)^2}\begin{pmatrix} L_2\cos(q_2)+L_1 & \sin(q_2)\,L_2 \\ -\sin(q_2)\,L_2 & L_2\cos(q_2)+L_1 \end{pmatrix}\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix}$$

$$=\frac{1}{L_2^2\cos(q_2)^2+L_1^2+2\,L_2L_1\cos(q_2)+L_1+L_2^2\sin(q_2)^2}\begin{pmatrix} (L_2\cos(q_2)+L_1)\,\hat{x}+\sin(q_2)\,L_2\,\hat{y} \\ -\sin(q_2)\,L_2\,\hat{x}+(L_2\cos(q_2)+L_1)\,\hat{y} \end{pmatrix}$$

Da cui si ottiene la variabile di giunto q_1;

$$q_1=\operatorname{atan2}\!\left(\frac{-\sin(q_2)\,L_2\,\hat{x}+(L_2\cos(q_2)+L_1)\,\hat{y}}{L_2^2\cos(q_2)^2+L_1^2+L_2^2\sin(q_2)^2},\frac{(L_2\cos(q_2)+L_1)\,\hat{x}+\sin(q_2)\,L_2\,\hat{y}}{L_2^2\cos(q_2)^2+L_1^2+L_2^2\sin(q_2)^2}\right)$$

Poiché la quantità $L_2^2\cos(q_2)^2+L_1^2+L_2^2\sin(q_2)^2>0$ è possibile semplificara all'interno della funzione atan2:

$$q_1=\operatorname{atan2}(\sin(q_2)\,L_2\,\hat{x}-(L_2\cos(q_2)+L_1)\,\hat{y},(L_2\cos(q_2)+L_1)\,\hat{x}+\sin(q_2)\,L_2\,\hat{y})$$

Infine:

$$q_3=\phi-(q_1+q_2)$$

```
Maxima 5.44.0 http://maxima.sourceforge.net
using Lisp SBCL 2.0.0
Distributed under the GNU Public License. See the file COPYING.
Dedicated to the memory of William Schelter.
The function bug_report() provides bug reporting information.
(%i1) calculate(x,y,L1,L2):=block([q2plus,q2minus,q1,res],
                                 c2:(x^(2)+y^(2)-L1^(2)-L2^(2))/(2*L1*L2),
                                 s2:sqrt(1-c2^2),

                                 c1Num:combine(expand((L2*c2+L1)*x+s2*L2*y)),
                                 s1Num:combine(expand(-s2*L2*x+(L2*c2+L1)*y)),

      q1Den:L2^(2)*c2^(2)+L1^(2)+L2^(2)*s2^(2)+2*L1*L2*c2,
                            if abs(c2)=1 then print("La soluzione è singolare")
                                 elseif q1Den>0 then(
                                     print("La soluzione non è singolare"),
                                     q1:atan2(s1Num,c1Num),
                                     q2alto:atan2(s2,c2),
                                     q2basso:atan2(-s2,c2),
                                     res:[[q2alto,q1],[q2basso,q1]])
                                 else (
                                     q1:atan2(s1Num/q1Den,c1Num/q1Den),
                                     q2alto:atan2(s2,c2),
                                     q2basso:atan2(-s2,c2),
                                     res:[[q2alto,q1],[q2basso,q1]])



                                 )
```

(%o1) $\operatorname{calculate}(x,y,L1,L2):=\mathbf{block}\left([q2plus,q2minus,q1,\mathrm{res}],c2:\frac{x^2+y^2-L1^2-L2^2}{2\,L1\,L2},s2:\right.$

$\sqrt{1-c2^2}$, $c1$Num: combine(expand($(L2\,c2+L1)\,x+s2\,L2\,y$)), $s1$Num: combine(expand($((-s2)\;L2\;x+(L2\;c2+L1)\;y)$)), $q1$Den: $L2^2\;c2^2+L1^2+L2^2\;s2^2+2\;L1\;L2\;c2$, **if** $|c2|=1$ **then** print(La soluzione è singolare) **elseif** $q1$Den >0 **then** (print(La soluzione non è singolare), $q1$: atan2($s1$Num, $c1$Num), $q2$alto: atan2($s2$, $c2$), $q2$basso: atan2($-s2$, $c2$), res: [[$q2$alto, $q1$], [$q2$basso, $q1$]]) **else** $\left(q1\text{: atan2}\left(\frac{s1\text{Num}}{q1\text{Den}}, \frac{c1\text{Num}}{q1\text{Den}}\right), q2\text{alto: atan2}(s2, c2), q2\text{basso: atan2}(-s2, c2), \text{res: }[[q2\text{alto}, q1], [q2\text{basso}, q1]]\right)\Big)$

(%i56)
```
inv2DOF(x,y,phi,link1,link2,link3):=block([R,circeInt,circleEst,res,q3,
pos],
                             R:matrix([cos(phi),-sin(phi)],
                                      [sin(phi), cos(phi)]),
                             coord:R.matrix([link3],[0]),
                             xcap:x-coord[1],
                             ycap:y-coord[2],

                             circleInt:link1^2+link2^2-2*link1*link2,
                             circleEst:link1^2+link2^2+2*link1*link2,
                           if (xcap[1]^2+ycap[1]^2>= circleInt and
xcap[1]^2+ycap[1]^2<= circleEst )then
                             (
                              print("Il punto x,y è nello spazio di lavoro"),
                              pos:calculate(xcap[1],ycap[1],link1,link2),
                              res:[[pos[1][2],pos[1][1],phi-
(pos[1][2]+pos[1][1])],[pos[2][2],pos[2][1],phi-(pos[2][1]+pos[2][2]) ]]

                              )
                              else res:"Punto x,y non è ammissibile"
                            )
```

(%o56) inv2DOF(x, y, φ, link1, link2, link3) := **block** $\Bigg($ [R, circeInt, circleEst, res, $q3$, pos], R: $\begin{pmatrix} \cos(\varphi) & -\sin(\varphi) \\ \sin(\varphi) & \cos(\varphi) \end{pmatrix}$, coord: $R \cdot \begin{pmatrix} \text{link3} \\ 0 \end{pmatrix}$, xcap: $x-\text{coord}_1$, ycap: $y-\text{coord}_2$, circleInt: $\text{link1}^2+\text{link2}^2+(-2)\,\text{link1 link2}$, circleEst: $\text{link1}^2+\text{link2}^2+2\,\text{link1 link2}$, **if** $\text{xcap}_1^2+\text{ycap}_1^2 \geq \text{circleInt} \wedge \text{xcap}_1^2+\text{ycap}_1^2 \leq \text{circleEst}$ **then** (print(Il punto x,y è nello spazio di lavoro), pos: calculate(xcap_1, ycap_1, link1, link2), print(pos1 , $(\text{pos}_1)_2$, pos2: , $(\text{pos}_1)_1$), print(POS= , $(\text{pos}_1)_2+(\text{pos}_1)_2$), res: $[[(\text{pos}_1)_2, (\text{pos}_1)_1, \varphi-((\text{pos}_1)_2+(\text{pos}_1)_1)], [(\text{pos}_2)_2, (\text{pos}_2)_1, \varphi-((\text{pos}_2)_1+(\text{pos}_2)_2)]]$) **else** res: Punto x,y non è ammissibile $\Bigg)$

(%i57)
```
inv2DOF(1,2,%pi/2,1,1,1);
```

Il punto x,y è nello spazio di lavoro

La soluzione non è singolare
pos1 0 pos2: $\frac{\pi}{2}$

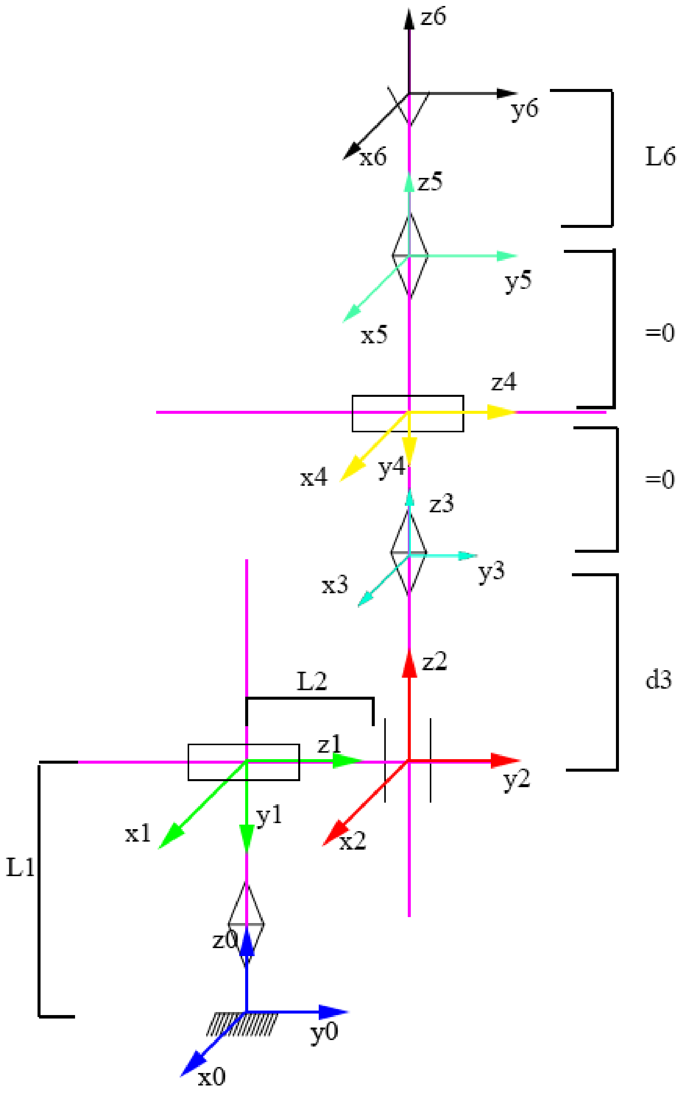
POS= 0

(%o57) $\left[\left[0, \frac{\pi}{2}, 0\right], \left[0, -\frac{\pi}{2}, \pi\right]\right]$

(%i58)

Stanford completo (robot sferico II tipo (stanford) + polso sferico)

N.B.: **le grandezze diverse da quelle di giunto q_i sono L_i, D_i. Esse sono rispettivamente la distanza tra i sistemi di riferimento R_i e R_{i+1} nelle operazioni della matrice avvitamento $A_z(\theta, d)$ e $A_x(\alpha, a)$.**



	ϑ	d	α	a
1	q_1	L_1	$-\frac{\pi}{2}$	0
2	q_2	L_2	$\frac{\pi}{2}$	0
3	0	q_3	0	0
4	q_4	0	$-\frac{\pi}{2}$	0
5	q_5	0	$\frac{\pi}{2}$	0
6	q_6	L_6	0	0

Tabella 1.

Funzioni ausiliarie:

```
(%i1) inverseLaplace(SI,theta):=block([res],
                              M:SI,
                              MC:SI,
                              for i:1 thru 3 do(
                                for j:1 thru 3 do
                                   (
                                    aC:M[i,j],
                                    b:ilt(aC,s,theta),
                                    MC[i,j]:b
                                   )
                                 ),
                              res:MC

                           )
```

(%o1) inverseLaplace(SI, ϑ) := **block** ([res], M: SI, MC: SI, **for** i **thru** 3 **do for** j **thru** 3 **do** (aC: $M_{i,j}$, b: ilt(aC, s, ϑ), $\mathrm{MC}_{i,j}$: b), res: MC)

```
(%i2) rotLaplace(k,theta):=block([res],
                          S:ident(3),
                          I:ident(3),
                        for i:1 thru 3 do
                          (
                          for j:1 thru 3 do
                             (
                              if i=j
                                 then S[i][j]:0
                              elseif j>i
                                 then (
                                temp:(-1)^(j-i)*k[3-remainder(i+j,3)],
                                       S[i][j]:temp,
                                       S[j][i]:-temp

                                        )
                              )
                           ),
                          res:inverseLaplace(invert(s*I-S),theta)


                        )
```

(%o2) rotLaplace(k, ϑ) := **block** ([res], S: ident(3), I: ident(3), **for** i **thru** 3 **do for** j **thru** 3 **do if** $i = j$ **then** $(S_i)_j$: 0 **elseif** $j > i$ **then** (temp: $(-1)^{j-i}\, k_{3-\mathrm{remainder}(i+j,3)}$, $(S_i)_j$: temp, $(S_j)_i$: $-$temp), res: inverseLaplace(invert($s\,I - S$), ϑ))

```
(%i3) Av(v,theta,d):=block([res],
                          Trot:rotLaplace(v,theta),
                          row:matrix([0,0,0,1]),
                          Atemp:addcol(Trot,d*transpose(v)),
                          A:addrow(Atemp,row),
                          res:A
                          )
```

(%o3) Av(v, ϑ, d) := **block** ([res], Trot: rotLaplace(v, ϑ), row: $\begin{pmatrix} 0 & 0 & 0 & 1 \end{pmatrix}$, Atemp: addcol(Trot, d transpose(v)), A: addrow(Atemp, row), res: A)

```
(%i4) Q(theta,d,alpha,a):=block([res],
                              tempMat:Av([0,0,1],theta,d).Av([1,0,0],alpha,a),
                              Qtrasf:zeromatrix(4,4),
                              for i:1 thru 4 do
                         (
                         for j:1 thru 4 do
                            (
                              Qtrasf[i][j]:trigreduce(tempMat[i][j])
                                        )
                             ),

                              res:Qtrasf
                              )
```

(%o4) $Q(\vartheta, d, \alpha, a) := \textbf{block}\ ([\mathrm{res}], \mathrm{tempMat}\colon \mathrm{Av}([0,0,1], \vartheta, d) \cdot \mathrm{Av}([1,0,0], \alpha, a), \mathrm{Qtrasf}\colon \mathrm{zeromatrix}(4,4), \textbf{for } i \textbf{ thru } 4 \textbf{ do for } j \textbf{ thru } 4 \textbf{ do } (\mathrm{Qtrasf}_i)_j\colon \mathrm{trigreduce}((\mathrm{tempMat}_i)_j), \mathrm{res}\colon \mathrm{Qtrasf})$

```
(%i5) let(sin(q[1]), s[1]);
```

(%o5) $\sin(q_1) \longrightarrow s_1$

```
(%i6) let(sin(q[2]), s[2]);
```

(%o6) $\sin(q_2) \longrightarrow s_2$

```
(%i7) let(cos(q[1]),c[1]);
```

(%o7) $\cos(q_1) \longrightarrow c_1$

```
(%i8) let(cos(q[2]),c[2]);
```

(%o8) $\cos(q_2) \longrightarrow c_2$

```
(%i9) let(sin(q[1]+q[2]), s[12]);
```

(%o9) $\sin(q_2 + q_1) \longrightarrow s_{12}$

```
(%i10) let(cos(q[1]+q[2]),c[12]);
```

(%o10) $\cos(q_2 + q_1) \longrightarrow c_{12}$

```
(%i11) let(sin(q[2]+q[3]),s[23]);
```

(%o11) $\sin(q_3 + q_2) \longrightarrow s_{23}$

```
(%i12) let(cos(q[2]+q[3]),c[23]);
```

(%o12) $\cos(q_3 + q_2) \longrightarrow c_{23}$

```
(%i13) let(sin(q[1]+q[3]),s[23]);
```

(%o13) $\sin(q_3 + q_1) \longrightarrow s_{23}$

```
(%i14) let(cos(q[1]+q[3]),c[13]);
```

(%o14) $\cos(q_3 + q_1) \longrightarrow c_{13}$

```
(%i15) let(sin(q[3]),s[3]);
```

(%o15) $\sin(q_3) \longrightarrow s_3$

```
(%i16) let(cos(q[3]),c[3]);
```

(%o16) $\cos(q_3) \longrightarrow c_3$

```
(%i17) let(sin(q[4]),s[4]);
```

(%o17) $\sin(q_4) \longrightarrow s_4$

(%i18) `let(cos(q[4]),c[4]);`

(%o18) $\cos(q_4) \longrightarrow c_4$

(%i19) `let(sin(q[5]),s[5]);`

(%o19) $\sin(q_5) \longrightarrow s_5$

(%i20) `let(cos(q[5]),c[5]);`

(%o20) $\cos(q_5) \longrightarrow c_5$

(%i21) `let(sin(q[6]),s[6]);`

(%o21) $\sin(q_6) \longrightarrow s_6$

(%i22) `let(cos(q[6]),c[6]);`

(%o22) $\cos(q_6) \longrightarrow c_6$

(%i23)

Cinematica diretta:

(%i26)
```
Q[stanford6DOF](q1,q2,q3,q4,q5,q6,L1,L2,L6):=
                     Q(q1,L1,-%pi/2,0).
                     Q(q2,L2,%pi/2,0).
                     Q(0,q3,0,0).
                     Q(q4,0,-%pi/2,0).
                     Q(q5,0,%pi/2,0).
                     Q(q6,L6,0,0);
```

(%o26) $Q_{\text{stanford6DOF}}(q1, q2, q3, q4, q5, q6, L1, L2, L6) := Q\left(q1, L1, \frac{-\pi}{2}, 0\right) \cdot Q\left(q2, L2, \frac{\pi}{2}, 0\right) \cdot Q(0, q3, 0, 0) \cdot Q\left(q4, 0, \frac{-\pi}{2}, 0\right) \cdot Q\left(q5, 0, \frac{\pi}{2}, 0\right) \cdot Q(q6, L6, 0, 0)$

(%i27)
```
Qstanford6DOF:Q[standford6DOF](q[1],q[2],q[3],q[4],q[5],q[6],L[1],L[2],
L[6]);
```

(%o27) $(\cos(q_1)(\cos(q_2)(\cos(q_4)\cos(q_5)\cos(q_6) - \sin(q_4)\sin(q_6)) - \sin(q_2)\sin(q_5)\cos(q_6)) - \sin(q_1)(\cos(q_4)\sin(q_6) + \sin(q_4)\cos(q_5)\cos(q_6)), \cos(q_1)(\cos(q_2)(-\cos(q_4)\cos(q_5)\sin(q_6) - \sin(q_4)\cos(q_6)) + \sin(q_2)\sin(q_5)\sin(q_6)) - \sin(q_1)(\cos(q_4)\cos(q_6) - \sin(q_4)\cos(q_5)\sin(q_6)), \cos(q_1)(\cos(q_2)\cos(q_4)\sin(q_5) + \sin(q_2)\cos(q_5)) - \sin(q_1)\sin(q_4)\sin(q_5), \cos(q_1)(L_6\cos(q_2)\cos(q_4)\sin(q_5) + \sin(q_2)(L_6\cos(q_5) + q_3)) - \sin(q_1)(L_6\sin(q_4)\sin(q_5) + L_2); \sin(q_1)(\cos(q_2)(\cos(q_4)\cos(q_5)\cos(q_6) - \sin(q_4)\sin(q_6)) - \sin(q_2)\sin(q_5)\cos(q_6)) + \cos(q_1)(\cos(q_4)\sin(q_6) + \sin(q_4)\cos(q_5)\cos(q_6)), \sin(q_1)(\cos(q_2)(-\cos(q_4)\cos(q_5)\sin(q_6) - \sin(q_4)\cos(q_6)) + \sin(q_2)\sin(q_5)\sin(q_6)) + \cos(q_1)(\cos(q_4)\cos(q_6) - \sin(q_4)\cos(q_5)\sin(q_6)), \sin(q_1)(\cos(q_2)\cos(q_4)\sin(q_5) + \sin(q_2)\cos(q_5)) + \cos(q_1)\sin(q_4)\sin(q_5), \cos(q_1)(L_6\sin(q_4)\sin(q_5) + L_2) + \sin(q_1)(L_6\cos(q_2)\cos(q_4)\sin(q_5) + \sin(q_2)(L_6\cos(q_5) + q_3)); -\sin(q_2)(\cos(q_4)\cos(q_5)\cos(q_6) - \sin(q_4)\sin(q_6)) - \cos(q_2)\sin(q_5)\cos(q_6), \cos(q_2)\sin(q_5)\sin(q_6) - \sin(q_2)(-\cos(q_4)\cos(q_5)\sin(q_6) - \sin(q_4)\cos(q_6)), \cos(q_2)\cos(q_5) - \sin(q_2)\cos(q_4)\sin(q_5), -L_6\sin(q_2)\cos(q_4)\sin(q_5) + \cos(q_2)(L_6\cos(q_5) + q_3) + L_1; 0, 0, 0, 1)$

(%i28) `letsimp(Qstanford6DOF);`

(%o28) $(-c_1 c_2 s_4 s_6 - s_1 c_4 s_6 - c_1 s_2 s_5 c_6 - s_1 s_4 c_5 c_6 + c_1 c_2 c_4 c_5 c_6, c_1 s_2 s_5 s_6 + s_1 s_4 c_5 s_6 - c_1 c_2 c_4 c_5 s_6 - c_1 c_2 s_4 c_6 - s_1 c_4 c_6, -s_1 s_4 s_5 + c_1 c_2 c_4 s_5 + c_1 s_2 c_5, -s_1 s_4 s_5 L_6 + c_1 c_2 c_4 s_5 L_6 + c_1 s_2 c_5 L_6 + c_1 s_2 q_3 - s_1 L_2; -s_1 c_2 s_4 s_6 + c_1 c_4 s_6 - s_1 s_2 s_5 c_6 + c_1 s_4 c_5 c_6 + s_1 c_2 c_4 c_5 c_6, s_1 s_2 s_5 s_6 - c_1 s_4 c_5 s_6 - s_1 c_2 c_4 c_5 s_6 - s_1 c_2 s_4 c_6 + c_1 c_4 c_6, c_1 s_4 s_5 + s_1 c_2 c_4 s_5 + s_1 s_2 c_5, c_1 s_4 s_5 L_6 + s_1 c_2 c_4 s_5 L_6 + s_1 s_2 c_5 L_6 + s_1 s_2 q_3 + c_1 L_2; s_2 s_4 s_6 - c_2 s_5 c_6 - s_2 c_4 c_5 c_6, c_2 s_5 s_6 + s_2 c_4 c_5 s_6 + s_2 s_4 c_6, c_2 c_5 - s_2 c_4 s_5, -s_2 c_4 s_5 L_6 + c_2 c_5 L_6 + c_2 q_3 + L_1; 0, 0, 0, 1)$

Cinematica Inversa Robot Stanford

Al fine di risolvere il problema di cinematica inversa del robot Stanford occorre risolvere il problema di posizione ed orientamente inverso.Inizialmente occorre verificare la condizione di disaccoppiamento, individuare lo spazio di lavoro, le soluzioni generiche, singolari ed infine le variabili di giunto q_i ed in seguito determinare l'orientamento del robot.

Poiché il robot Stanford è un robot 6 DOF (6 gradi di libertà), occorre disaccoppiare la struttura portante dal suo polso. Ciò è possibile se:

$$d_{36}(q_b) = R_{36}(q_6)\, d_1 + d_0$$

In particolare:

$$Q_{03} = \begin{pmatrix} s_1 & c_1c_2 & c_1s_2 & q_3c_1s_2 - L_2s_1 \\ -c_1 & s_1c_2 & s_1s_2 & q_3s_1s_2 + L_2\, c_1 \\ 0 & -s_2 & c_2 & L_1 + q_3c_2 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$Q_{36} = \begin{pmatrix} c_4\, c_5\, c_6 - s_4\, s_6 & -c_4\, c_5\, s_6 - s_4\, c_6 & c_4\, s_5 & c_4\, s_5\, L_6 \\ c_4\, s_6 + s_4\, c_5\, c_6 & c_4\, c_6 - s_4\, c_5\, s_6 & s_4\, s_5 & s_4\, s_5\, L_6 \\ -s_5\, c_6 & s_5\, s_6 & c_5 & c_5\, L_6 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\begin{pmatrix} c_4\, s_5\, L_6 \\ s_4\, s_5\, L_6 \\ c_5\, L_6 \end{pmatrix} = \begin{pmatrix} c_4\, c_5\, c_6 - s_4\, s_6 & -c_4\, c_5\, s_6 - s_4\, c_6 & c_4\, s_5 \\ c_4\, s_6 + s_4\, c_5\, c_6 & c_4\, c_6 - s_4\, c_5\, s_6 & s_4\, s_5 \\ -s_5\, c_6 & s_5\, s_6 & c_5 \end{pmatrix} d_1 + d_0$$

Ottenendo:

$$d_1 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} L_6 \quad d_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

Quindi:

$$\begin{cases} R = R_{03}\, R_{36} \\ P = R_{03}\, d_{36} + d_{03} = R_{03}(R_{36}d_1 + d_0) + d_{03} = R\, d_1 + d_{03} \end{cases}$$

$$P - R\, d_1 = \hat{P} = d_{03} \longrightarrow \begin{pmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \end{pmatrix} = \begin{pmatrix} q_3c_1s_2 - L_2s_1 \\ q_3s_1s_2 + L_2\, c_1 \\ L_1 + q_3c_2 \end{pmatrix}$$

$$\begin{cases} \hat{x} = q_3c_1s_2 - L_2s_1 \\ \hat{y} = q_3s_1s_2 + L_2\, c_1 \\ \hat{z} = L_1 + q_3c_2 \end{cases} \rightarrow \begin{cases} \begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = \begin{pmatrix} c_1 & -s_1 \\ s_1 & c_1 \end{pmatrix} \begin{pmatrix} q_3s_2 \\ L_2 \end{pmatrix} \\ (\hat{z} - L_1)^2 = q_3^2c_2^2 \end{cases}$$

Poiché $\begin{pmatrix} c_1 & -s_1 \\ s_1 & c_1 \end{pmatrix}$ è una matrice di rotazione,$\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix}$ e $\begin{pmatrix} q_3s_2 \\ L_2 \end{pmatrix}$ devono avere stessa norma:

$$\begin{cases} \hat{x}^2 + \hat{y}^2 = q_3^2s_2^2 + L_2^2 \\ (\hat{z} - L_1)^2 = q_3^2c_2^2 \end{cases} \rightarrow \hat{x}^2 + \hat{y}^2 + (\hat{z} - L_1)^2 - L_2^2 = q_3^2(s_2^2 + c_2^2)$$

Ottenendo:

$$\hat{x}^2 + \hat{y}^2 + (\hat{z} - L_1)^2 - L_2^2 = q_3^2$$

$$q_3 = \pm\sqrt{\hat{x}^2 + \hat{y}^2 + (\hat{z} - L_1)^2 - L_2^2}$$

Poich $q_3 \neq 0$, è possibile ottente c_2:

$$c_2 = \frac{(\hat{z} - L_1)}{q_3}, s_2 = \pm\sqrt{1 - c_2^2} \longrightarrow q_2 = \text{atan2}\left(\pm\sqrt{1 - \frac{(\hat{z} - L_1)^2}{q_3^2}}, \frac{(\hat{z} - L_1)^2}{q_3^2}\right)$$

A questo punto, la quantità $\left(\begin{smallmatrix} q_3 s_2 \\ L_2 \end{smallmatrix}\right)$ è nota, quindi è possibile determinare la variabile di giunto q_1:

$$\left(\begin{array}{c} \hat{x} \\ \hat{y} \end{array}\right) = \left(\begin{array}{cc} c_1 & -s_1 \\ s_1 & c_1 \end{array}\right)\left(\begin{array}{c} q_3 s_2 \\ L_2 \end{array}\right) \rightarrow \left(\begin{array}{c} \hat{x} \\ \hat{y} \end{array}\right) = \left(\begin{array}{cc} q_3 s_2 & -L_2 \\ L_2 & q_3 s_2 \end{array}\right)\left(\begin{array}{c} c_1 \\ s_1 \end{array}\right)$$

Poiché $\det\left(\begin{array}{cc} q_3 s_2 & -L_2 \\ L_2 & q_3 s_2 \end{array}\right) = q_3^2 s_2^2 + L_2^2 \neq 0$, è possibile effettuare l'inversa:

$$\left(\begin{array}{c} c_1 \\ s_1 \end{array}\right) = \frac{1}{q_3^2 s_2^2 + L_2^2}\left(\begin{array}{cc} q_3 s_2 & L_2 \\ -L_2 & q_3 s_2 \end{array}\right)\left(\begin{array}{c} \hat{x} \\ \hat{y} \end{array}\right) = \left(\begin{array}{c} q_3 s_2 \hat{x} + L_2 \hat{y} \\ -L_2 \hat{x} + q_3 s_2 \hat{y} \end{array}\right)$$

$$q_1 = \text{atan2}(-L_2 \hat{x} + q_3 s_2 \hat{y}, q_3 s_2 \hat{x} + L_2 \hat{y})$$

Riassumendo:

$$\left(\begin{array}{c} q_{3+} \\ q_{2+} \\ q_1 \end{array}\right), \left(\begin{array}{c} q_{3+} \\ q_{2-} \\ q_1 \end{array}\right), \left(\begin{array}{c} q_{3-} \\ q_{2+} \\ q_1 \end{array}\right), \left(\begin{array}{c} q_{3-} \\ q_{2-} \\ q_1 \end{array}\right)$$

Orientamento inverso

$$\hat{R} = R_{03}^T R$$

In cui:

$$R_{03} = \left(\begin{array}{ccc} s_1 & c_1 c_2 & c_1 s_2 \\ -c_1 & s_1 c_2 & s_1 s_2 \\ 0 & -s_2 & c_2 \end{array}\right), R = R_{\text{zyz}} = \left(\begin{array}{ccc} c_\alpha c_\beta c_\gamma - s_\alpha s_\gamma & -c_\alpha c_\beta s_\gamma - s_\alpha c_\gamma & c_\alpha s_\beta \\ c_\alpha s_\gamma + s_\alpha c_\beta c_\gamma & c_\alpha c_\gamma - s_\alpha c_\beta s_\gamma & s_\alpha s_\beta \\ -s_\beta c_\gamma & s_\beta s_\gamma & c_\beta \end{array}\right)$$

$$\hat{R} = \left(\begin{array}{ccc} \hat{r}_{1,1} & \hat{r}_{1,2} & \hat{r}_{1,3} \\ \hat{r}_{2,1} & \hat{r}_{2,2} & \hat{r}_{2,3} \\ \hat{r}_{3,1} & \hat{r}_{3,2} & \hat{r}_{3,3} \end{array}\right)$$

$$c_5 = \hat{r}_{3,3}$$

$$s_5 = \pm\sqrt{1 - \hat{r}_{3,3}^2}$$

$$q_5 = \text{atan}\left(\pm\sqrt{1 - \hat{r}_{3,3}^2}, \hat{r}_{3,3}\right)$$

$$\left\{\begin{array}{l} s_5 s_6 = \hat{r}_{3,2} \\ -s_5 c_6 = \hat{r}_{3,1} \end{array}\right. \rightarrow \left\{\begin{array}{l} s_6 = \pm\frac{\hat{r}_{3,2}}{s_5} \\ c_6 = \mp\frac{\hat{r}_{3,1}}{s_5} \end{array}\right. \rightarrow q_6 = \text{atan2}\left(\pm\frac{\hat{r}_{3,2}}{s_5}, \mp\frac{\hat{r}_{3,1}}{s_5}\right)$$

$$\left\{\begin{array}{l} c_4 s_5 = \hat{r}_{1,3} \\ s_4 s_5 = \hat{r}_{2,3} \end{array}\right. \rightarrow \left\{\begin{array}{l} c_4 = \pm\frac{\hat{r}_{1,3}}{s_5} \\ s_4 = \pm\frac{\hat{r}_{2,3}}{s_5} \end{array}\right. \rightarrow q_4 = \text{atan2}\left(\pm, \frac{\hat{r}_{2,3}}{s_5}, \pm\frac{\hat{r}_{1,3}}{s_5}\right)$$

Riassumendo:

$$\begin{pmatrix} q_{3+} \\ q_{2+} \\ q_1 \end{pmatrix} \rightarrow R_{03,1} = \begin{pmatrix} s_1 & c_1\,c_2 & c_1\,s_2 \\ -c_1 & s_1\,c_2 & s_1\,s_2 \\ 0 & -s_2 & c_2 \end{pmatrix} \rightarrow \begin{pmatrix} q_4 \\ q_5 \\ q_6 \end{pmatrix}, \begin{pmatrix} q_4 \\ q_5 \\ q_6 \end{pmatrix}$$

$$\begin{pmatrix} q_{3-} \\ q_{2+} \\ q_1 \end{pmatrix} \rightarrow R_{03,3} = \begin{pmatrix} s_1 & c_1\,c_2 & c_1\,s_2 \\ -c_1 & s_1\,c_2 & s_1\,s_2 \\ 0 & -s_2 & c_2 \end{pmatrix} \rightarrow \begin{pmatrix} q_4 \\ q_5 \\ q_6 \end{pmatrix}, \begin{pmatrix} q_4 \\ q_5 \\ q_6 \end{pmatrix}$$

$$\begin{pmatrix} q_{3+} \\ q_{2-} \\ q_1 \end{pmatrix} \rightarrow R_{03,2} = \begin{pmatrix} s_1 & c_1\,c_2 & c_1\,s_2 \\ -c_1 & s_1\,c_2 & s_1\,s_2 \\ 0 & -s_2 & c_2 \end{pmatrix} \rightarrow \begin{pmatrix} q_4 \\ q_5 \\ q_6 \end{pmatrix}, \begin{pmatrix} q_4 \\ q_5 \\ q_6 \end{pmatrix}$$

$$\begin{pmatrix} q_{3-} \\ q_{2-} \\ q_1 \end{pmatrix} \rightarrow R_{03,4} = \begin{pmatrix} s_1 & c_1\,c_2 & c_1\,s_2 \\ -c_1 & s_1\,c_2 & s_1\,s_2 \\ 0 & -s_2 & c_2 \end{pmatrix} \rightarrow \begin{pmatrix} q_4 \\ q_5 \\ q_6 \end{pmatrix}, \begin{pmatrix} q_4 \\ q_5 \\ q_6 \end{pmatrix}$$

```
(%i23) skewMatrix(x):=block([res],
                            S:ident(3),
                            for i:1 thru 3 do
                            (
                            for j:1 thru 3 do
                               (
                                 if i=j
                                    then S[i][j]:0
                                 elseif j>i
                                    then (
                                   temp:(-1)^(j-i)*x[3-remainder(i+j,3)],
                                          S[i][j]:temp,
                                          S[j][i]:-temp

                                           )
                                 )
                             ),
                             res:S
                             )
```

(%o23) $\mathrm{skewMatrix}(x) := \mathbf{block}\ ([\mathrm{res}], S\colon \mathrm{ident}(3), \mathbf{for}\ i\ \mathbf{thru}\ 3\ \mathbf{do}\ \mathbf{for}\ j\ \mathbf{thru}\ 3\ \mathbf{do}\ \mathbf{if}\ i = j\ \mathbf{then}\ (S_i)_j\colon 0\ \mathbf{elseif}\ j > i\ \mathbf{then}\ (\mathrm{temp}\colon (-1)^{j-i}\, x_{3-\mathrm{remainder}(i+j,3)}, (S_i)_j\colon \mathrm{temp}, (S_j)_i\colon -\mathrm{temp}), \mathrm{res}\colon$

$S)$

```
(%i24) rodriguez(y,arg):=block([res],
                          I:ident(3),
                          S:skewMatrix(y),
                          res:I+S.S*(1-cos(arg))+S*sin(arg)
                          )
```

(%o24) $\mathrm{rodriguez}(y, \mathrm{arg}) := \mathbf{block}\ ([\mathrm{res}], I\colon \mathrm{ident}(3), S\colon \mathrm{skewMatrix}(y), \mathrm{res}\colon I + S \cdot S\,(1 - \cos(\mathrm{arg})) + S \sin(\mathrm{arg}))$

```
(%i39) posInversa(x,y,z,L1,L2):=block(
                   [valueq3,valueq2,q3,q2,s1,c1,q1,res],
                   if x^(2)+y^(2)+(z-L1)^2-L2^(2)<=0 then error("singolare!"),
                    valueq3:sqrt(x^(2)+y^(2)+(z-L1)^2-L2^(2)),

                    q3:[valueq3,-valueq3],
                    valueq2:((z-L1)^2)/q3^(2),
                    q2:[atan2(sqrt(1-valueq2),valueq2),
                        atan2(-sqrt(1-valueq2),valueq2)],

                    s1:[-L2*x+q3[1]*sin(q2[1][1][1])*y,
                        -L2*x+q3[1]*sin(q2[1][1][2])*y,
                        -L2*x+q3[2]*sin(q2[1][1][1])*y,
                        -L2*x+q3[2]*sin(q2[1][1][2])*y],

                    c1:[q3[1]*sin(q2[1][1][1])*x+L2*y,
                        q3[1]*sin(q2[1][1][2])*x+L2*y,
                        q3[2]*sin(q2[1][1][1])*x+L2*y,
                        q3[2]*sin(q2[1][1][2])*x+L2*y],
                    q1:[atan2(s1[1],c1[1]),
                        atan2(s1[2],c1[2]),
                        atan2(s1[3],c1[3]),
                        atan2(s1[4],c1[4])],
                     res:[[q1[1],q2[1][1][1],q3[1]],
                          [q1[2],q2[1][1][2],q3[1]],
                          [q1[3],q2[1][1][1],q3[2]],
                          [q1[4],q2[1][1][2],q3[2]]]
                    )
```

(%o39) $\mathrm{posInversa}(x, y, z, L1, L2) := \mathbf{block}\Big([\mathrm{valueq3}, \mathrm{valueq2}, q3, q2, s1, c1, q1, \mathrm{res}], \mathbf{if}\ x^2 + y^2 + (z - L1)^2 - L2^2 \le 0\ \mathbf{then}\ \mathrm{error}(\mathrm{singolare!}\ ), \mathrm{valueq3}\colon \sqrt{x^2 + y^2 + (z - L1)^2 - L2^2}, q3\colon [\mathrm{valueq3}, -\mathrm{valueq3}], \mathrm{valueq2}\colon \frac{(z - L1)^2}{q3^2}, q2\colon \big[\mathrm{atan2}\big(\sqrt{1 - \mathrm{valueq2}}, \mathrm{valueq2}\big), \mathrm{atan2}\big(-\sqrt{1 - \mathrm{valueq2}}, \mathrm{valueq2}\big)\big], s1\colon [(-L2)\,x + q3_1 \sin(((q2_1)_1)_1)\,y, (-L2)\,x + q3_1 \sin(((q2_1)_1)_2)\,y, (-L2)\,x + q3_2 \sin(((q2_1)_1)_1)\,y, (-L2)\,x + q3_2 \sin(((q2_1)_1)_2)\,y], c1\colon [q3_1 \sin(((q2_1)_1)_1)\,x + L2\,y, q3_1 \sin(((q2_1)_1)_2)\,x + L2\,y, q3_2 \sin(((q2_1)_1)_1)\,x + L2\,y, q3_2 \sin(((q2_1)_1)_2)\,x + L2\,y], q1\colon [\mathrm{atan2}(s1_1, c1_1), \mathrm{atan2}(s1_2, c1_2), \mathrm{atan2}(s1_3, c1_3), \mathrm{atan2}(s1_4, c1_4)], \mathrm{res}\colon [[q1_1, ((q2_1)_1)_1, q3_1], [q1_2, ((q2_1)_1)_2, q3_1],$

$[q1_3, ((q2_1)_1)_1, q3_2], [q1_4, ((q2_1)_1)_2, q3_2]]\Big)$

(%i58)
```
orienInversa(R,sol):=block([R03,Rcap,q4,q5,q6,res],
          Rcap:[0,0,0,0],

          for i:1 thru 4 do (

            R03:matrix([sin(sol[i][1]),cos(sol[i][1])*cos(sol[i][2]),
cos(sol[i][1])*cos(sol[i][2])],
                      [-cos(sol[i][1]),sin(sol[i][1])*cos(sol[i][2]),
sin(sol[i][1])*sin(sol[i][2])],
                                     [0,-sin(sol[i][2]),
cos(sol[i][2])]),


           Rcap[i]:transpose(R03).R

           ),

           for i:1 thru 4 do (

                           c5:Rcap[i][3][3],
                           s5:sqrt(1-c5^2),

                           q5plus:atan2(s5,c5),
                           q5minus:atan2(-s5,c5),

                           s6:abs(Rcap[i][3][2]/s5),
                           c6:abs(Rcap[i][3][1]/s5),

                           q6plus:atan2(s6,-c6),
                           q6minus:atan2(-s6,c6),
                           c4:Rcap[i][1][3]/s5,
                           s4:Rcap[i][2][3]/s5,

                           q4plus:atan2(s4,c4),
                           q4minus:atan2(-s4,-c4),
                           first:[q4plus,q5plus,q6plus],
                           second:[q4minus,q5minus,q6minus],
                           res[i]:[first,second]
                           ),
                            res)
```

(%o58) $\operatorname{orienInversa}(R, \mathrm{sol}) := \mathbf{block}\Bigg([R03, \mathrm{Rcap}, q4, q5, q6, \mathrm{res}], \mathrm{Rcap}\colon [0,0,0,0],$

$$\mathbf{for}\ i\ \mathbf{thru}\ 4\ \mathbf{do}\left(R03\colon \begin{pmatrix} \sin((\mathrm{sol}_i)_1) & \cos((\mathrm{sol}_i)_1)\cos((\mathrm{sol}_i)_2) & \cos((\mathrm{sol}_i)_1)\cos((\mathrm{sol}_i)_2) \\ -\cos((\mathrm{sol}_i)_1) & \sin((\mathrm{sol}_i)_1)\cos((\mathrm{sol}_i)_2) & \sin((\mathrm{sol}_i)_1)\sin((\mathrm{sol}_i)_2) \\ 0 & -\sin((\mathrm{sol}_i)_2) & \cos((\mathrm{sol}_i)_2) \end{pmatrix},\right.$$

$\mathrm{Rcap}_i\text{: transpose}(R03)\cdot R\Big)$, **for** i **thru** 4 **do** $\Big(c5\text{: }((\mathrm{Rcap}_i)_3)_3,\ s5\text{: }\sqrt{1-c5^2},\ q5\text{plus: atan2}(s5,c5),$
$q5\text{minus: atan2}(-s5,c5),\ s6\text{: }\left|\frac{((\mathrm{Rcap}_i)_3)_2}{s5}\right|,\ c6\text{: }\left|\frac{((\mathrm{Rcap}_i)_3)_1}{s5}\right|,\ q6\text{plus: atan2}(s6,-c6),\ q6\text{minus:}$
$\text{atan2}(-s6,c6),\ c4\text{: }\frac{((\mathrm{Rcap}_i)_1)_3}{s5},\ s4\text{: }\frac{((\mathrm{Rcap}_i)_2)_3}{s5},\ q4\text{plus: atan2}(s4,c4),\ q4\text{minus: atan2}(-s4,-c4),$
$\text{first: }[q4\text{plus},q5\text{plus},q6\text{plus}],\ \text{second: }[q4\text{minus},q5\text{minus},q6\text{minus}],\ \mathrm{res}_i\text{: [first, second]}\Big),\ \mathrm{res}\Big)$

```
(%i67) invSTANFORD(x,y,z,L1,L2,L6,alpha,beta,gamma):=block(
                [R,pos,orien,res],
                 R:rodriguez([0,0,1],alpha).
                            rodriguez([0,1,0],beta).
                            rodriguez([0,0,1],gamma),
                          coordPolso:R.matrix([0],[0],[L6]),
                          xCap:x-coordPolso[1],
                          yCap:y-coordPolso[2],
                          zCap:z-coordPolso[3],
                          pos:posInversa(xCap[1],yCap[1],zCap[1],L1,L2),

                          orien:orienInversa(R,pos),

                          res:[
                            [pos[1][1],pos[1][2],pos[1][3],
       orien[1][1][1],orien[1][1][2],orien[1][1][3]],
                            [pos[1][1],pos[1][2],pos[1][3],
       orien[1][2][1],orien[1][2][2],orien[1][2][3]],
                            [pos[2][1],pos[2][2],pos[2][3],
       orien[2][1][1],orien[2][1][2],orien[2][1][3]],
                            [pos[2][1],pos[2][2],pos[2][3],
       orien[2][2][1],orien[2][2][2],orien[2][2][3]],
                            [pos[3][1],pos[3][2],pos[3][3],
       orien[3][1][1],orien[3][1][2],orien[3][1][3]],
                            [pos[3][1],pos[3][2],pos[3][3],
       orien[3][2][1],orien[3][2][2],orien[3][2][3]],
                            [pos[4][1],pos[4][2],pos[4][3],
       orien[4][1][1],orien[4][1][2],orien[4][1][3]],
                            [pos[4][1],pos[4][2],pos[4][3],
       orien[4][2][1],orien[4][2][2],orien[4][2][3]]
                          ])
```

(%o67) $\text{invSTANFORD}(x,y,z,L1,L2,L6,\alpha,\beta,\gamma):=\textbf{block}\Big([R,\text{pos},\text{orien},\text{res}],\ R\text{: rodriguez}([0,$ $0,1],\alpha)\cdot\text{rodriguez}([0,1,0],\beta)\cdot\text{rodriguez}([0,0,1],\gamma),\ \text{coordPolso: }R\cdot\begin{pmatrix}0\\0\\L6\end{pmatrix},\ \text{xCap: }x-$ $\text{coordPolso}_1,\ \text{yCap: }y-\text{coordPolso}_2,\ \text{zCap: }z-\text{coordPolso}_3,\ \text{pos: posInversa}(\text{xCap}_1,\text{yCap}_1,\text{zCap}_1,$ $L1,L2),\ \text{orien: orienInversa}(R,\text{pos}),\ \text{res: }[[(\text{pos}_1)_1,(\text{pos}_1)_2,(\text{pos}_1)_3,((\text{orien}_1)_1)_1,((\text{orien}_1)_1)_2,$ $((\text{orien}_1)_1)_3],[(\text{pos}_1)_1,(\text{pos}_1)_2,(\text{pos}_1)_3,((\text{orien}_1)_2)_1,((\text{orien}_1)_2)_2,((\text{orien}_1)_2)_3],[(\text{pos}_2)_1,(\text{pos}_2)_2,$ $(\text{pos}_2)_3,((\text{orien}_2)_1)_1,((\text{orien}_2)_1)_2,((\text{orien}_2)_1)_3],[(\text{pos}_2)_1,(\text{pos}_2)_2,(\text{pos}_2)_3,((\text{orien}_2)_2)_1,((\text{orien}_2)_2)_2,$ $((\text{orien}_2)_2)_3],[(\text{pos}_3)_1,(\text{pos}_3)_2,(\text{pos}_3)_3,((\text{orien}_3)_1)_1,((\text{orien}_3)_1)_2,((\text{orien}_3)_1)_3],[(\text{pos}_3)_1,(\text{pos}_3)_2,$ $(\text{pos}_3)_3,((\text{orien}_3)_2)_1,((\text{orien}_3)_2)_2,((\text{orien}_3)_2)_3],[(\text{pos}_4)_1,(\text{pos}_4)_2,(\text{pos}_4)_3,((\text{orien}_4)_1)_1,((\text{orien}_4)_1)_2,$

$$((\mathrm{orien}_4)_1)_3],[(\mathrm{pos}_4)_1,(\mathrm{pos}_4)_2,(\mathrm{pos}_4)_3,((\mathrm{orien}_4)_2)_1,((\mathrm{orien}_4)_2)_2,((\mathrm{orien}_4)_2)_3]]\Bigg)$$

(%i74) `invSTANFORD(10,1,1,0.412,0.154,0.263,%pi,%pi/2,%pi/4);`

(%o74) $[[0.08221974213827725, 1.567549250542036, 10.32720664071364, -3.102209469265483,$
$1.57403243393346, \pi - 0.7458814168204394], [0.08221974213827725, 1.567549250542036, 10.32720\backslash$
$664071364, 0.03938318432431065, -1.57403243393346, -0.7458814168204394], [0.0822197421382\backslash$
$7725, 1.567549250542036, 10.32720664071364, -3.102209469265483, 1.57403243393346, \pi - 0.745\backslash$
$8814168204394], [0.08221974213827725, 1.567549250542036, 10.32720664071364, 0.0393831843243\backslash$
$1065, -1.57403243393346, -0.7458814168204394], [-3.029550830113841, 1.567549250542036,$
$-10.32720664071364, 0.02885150023036301, 1.567569610176263, \pi - 0.814431758491189], [-3.029\backslash$
$550830113841, 1.567549250542036, -10.32720664071364, -3.11274115335943, -1.56756961017\backslash$
$6263, -0.814431758491189], [-3.029550830113841, 1.567549250542036, -10.32720664071364, 0.02\backslash$
$885150023036301, 1.567569610176263, \pi - 0.814431758491189], [-3.029550830113841, 1.56754925\backslash$
$0542036, -10.32720664071364, -3.11274115335943, -1.567569610176263, -0.814431758491189]]$

(%i75)

sketch_22_3DOFPlanare

```
float q1=0;
float q2=0.0;
float q3=0.0;
float q1g=0.0;
float q2g=0.0;
float q3g=0.0;
float angolo1=0.0;
float angolo2=0.0;
float angolo3=0.0;
int link1=100;
int link2=100;
int link3=100;
float angolo1v=0.0;
float angolo2v=0.0;
float angolo3v=0.0;
float phi=0.0;

float x=0;
float y=0;
int target_r=255;
int target_g=255;
int target_b=255;
float K =0.1;
float gomito = 1.0;
float gomitoN=0.0;
float gomitoG=0.0;
float kk=0.00001;

void setup()
{
 size(1000, 1000);
 background(#45E6EA);
}

void draw()
```

```
{
 newton();
 gradiente();

 //Leggi di controllo
 angolo1v=angolo1v+K*(angolo1-angolo1v);
 angolo2v=angolo2v+K*(angolo2-angolo2v);
 angolo3v=angolo3v+K*(angolo3-angolo3v);

 background(#45E6EA);
 details();

 pushMatrix();
 translate(500, 500);
 fill(255);
 circle(0, 0, 2*(link1+link2+link3));

 pushMatrix();
 translate(x, y);
 //Punto da raggiungere
 target(50);
 popMatrix();

 treDOF(angolo1, angolo2, angolo3, 255, 183, 77, link1, link2, link3);
 popMatrix();
//Robot vero controllo proporzionale
 pushMatrix();
 translate(500, 500);
 treDOF(angolo1v, angolo2v, angolo3v, 0, 255, 0, link1, link2, link3);
 popMatrix();
//Robot che implementa l&apos;algoritmo di Newton
 pushMatrix();
 translate(500, 500);
 treDOF(q1-PI/2, q2, q3, 132, 255, 255, link1, link2, link3);
 popMatrix();
//Robot che implementa l&apos;algoritmo del gradiente
```

```
 pushMatrix();
 translate(500, 500);
 treDOF(q1g-PI/2, q2g, q3g, 88, 24, 69, link1, link2, link3);
 popMatrix();
}

void treDOF(float q1, float q2, float q3, int r, int g, int b, int l1, int l2, int l3)
{
 rotate(q1);
 link(50, 50, 100, r, g, b, l1);
 translate(0, 100);
 rotate(q2);
 link(50, 50, 100, r, g, b, l2);
 translate(0, 100);
 rotate(q3);
 link(50, 50, 100, r, g, b, l3);
}

void target(int R) {
 fill(target_r, target_g, target_b);
 circle(0, 0, R);
}

void link(int R, int larg, int lung, int r, int g, int b, int t)
{
 fill(r, g, b, t);
 circle(0, 0, R);
 circle(0, lung, R);
 rect(-R/2, 0, larg, lung);
}


void keyPressed()
{
 if (keyCode == &apos;1&apos;) angolo1 = angolo1+0.5;
 if (keyCode == &apos;2&apos;) angolo2 = angolo2+0.5;
```

```
 if (keyCode == &apos;3&apos;) angolo3 = angolo3+0.5;
 if (keyCode == &apos;7&apos;) angolo3 = angolo3-0.5;
 if (keyCode == &apos;8&apos;) angolo1 = angolo1-0.5;
 if (keyCode == &apos;9&apos;) angolo2 = angolo2-0.5;
 if (keyCode == ENTER) {
   angolo1=0;
   angolo2=0;
   angolo3=0;
 }
 if (keyCode == &apos;G&apos;) {
   gomito=-gomito;
   gomitoN++;
   gomitoG++;
   mousePressed();
 }
 if (keyCode == &apos;O&apos;) {
   phi+=0.523599;
   angolo1=0;
   angolo2=0;
   angolo3=0;
   mousePressed();
 }
 if (keyCode == &apos;P&apos;) {
   phi-=0.523599;
   angolo1=0;
   angolo2=0;
   angolo3=0;
   mousePressed();
 }
}

/*
details() è una funzione che permette la stampa di tutte le informazioni nella
finestra di esecuzione
*/
void details() {
```

```
 String line="q1="+angolo1+"\nq2=+"+angolo2+"\nq3="+angolo3;
 String lineV="q1v="+angolo1v+"\nq2v=+"+angolo2v+"\nq3v="+angolo3v;
 String phiText="phi="+phi+"="+(angolo1+angolo2+angolo3);
 String lineNewton="q1="+q1+"\nq2="+q2+"\nq3="+q3;
 String lineGradiente="q1="+q1g+"\nq2="+q2g+"\nq3="+q3g;

 String phiTextNewton="phi="+phi+"="+(q1+q2+q3);
 String phiTextGradiente="phi="+phi+"="+(q1g+q2g+q3g);

 textSize(20);

 fill(0);
 textLeading(20);
 text(line, 5, 100);
 textLeading(20);
 fill(0);
 text(lineV, 200, 100);
 textLeading(20);
 text(phiText, 5, 20);
 text("Newton:"+phiTextNewton, 5, 40);
 text("Gradiente:"+phiTextGradiente, 5, 60);

 text("Newton", 400, 80);
 text("Gradiente", 600, 80);
 textLeading(20);

 text(lineNewton, 400, 100);
 textLeading(20);

 text(lineGradiente, 600, 100);
}

/*
 La funzione newton() implementa l&apos;algoritmo di Newton che permette
di stimare in maniera asintotica i parametri
 di giunto e di conseguenza risolvendo il problema di cinematica
```

```
diretta/inversa di un qualsiasi robot. Si basa
 sulla conoscenza della matrice Jacobiana J(q) ottenuto dalle variabili x,y,phi.
 VANTAGGI:
   - Convergenza esponenziale -> convergenza molto veloce;
   - Poco affetto da rumori ed incertezze;
 SVANTAGGI:
   - In vicinaza alle singolarità, il problema diventa malcondizionato poiché
questo algoritmo necessita la conscenza
     dell&apos;inversa di J e quindi il suo determinante è prossimo a 0.
   - In punti non appartenenti allo spazio di lavoro, il robot procederà a scatti
non raggiungendo la soluzione.
 Al fine di ridurre questa problematica si corregge il det(J) per valori prossimi
a 0.

 OSS.:
   Nel caso del robot 3DOF al fine di ottenere il cambio gomito del robor si
devono rispettare due specifiche:
     - mantenere il robot lontano dalle singolairtà;
     - portare il robot verso la seconda soluzione;
   Quindi, per portare il robot verso la zona "attrattiva" della seconda soluzione
evitando la singolarità, al cambio gomito
   sin(q2)->-sin(q2)=sin(q2+PI).
   A questo punto l&apos;algoritmo di Newton convergerà normalmente alla
soluzione richiesta,ottenendo così il risultato richiesto.
*/

void newton() {



 float kN=0.1;
 if (gomitoN%2!=0) {
   q2=q2+gomito*PI;

   gomitoN++;
 }
 float s_2=sin(q2);
```

```
 if (abs(s_2)<0.01) {
  println("Correzione Newton");
  s_2=gomito*0.01;
 }
 float s_1=sin(q1);
 float c_2=cos(q2);
 float c_1=cos(q1);
 float c_3=cos(q3);
 float s_3=sin(q3);
 float s_12=c_1*s_2 + c_2*s_1;
 float c_12=c_1*c_2-s_1*s_2;
 float s_23=c_2*s_3+s_2*c_3;
 float c_23=c_2*c_3-s_2*s_3;
 float
Q1=(s_12*y+c_12*x+link3*s_3*phi+((-q2-q1)*link3-link3*q3)*s_3-link3*c_
3-link1*c_2-link2)/(link1*s_2);
 float
Q2=-((link2*s_12+link1*s_1)*y+(link2*c_12+link1*c_1)*x+(link1*link3*s_2
3+link2*link3*s_3)*phi+((-link1*q2-link1*q1)*link3-link1*link3*q3)*s_23-li
nk1*link3*c_23+((-link2*q2-q1*link2)*link3-link2*link3*q3)*s_3-link2*link
3*c_3-2*link1*link2*c_2-link2*link2-link1*link1)/(link1*link2*s_2);
 float
Q3=(s_1*y+c_1*x+(link3*s_23+link2*s_2)*phi+((-q2-q1)*link3-link3*q3)*s
_23-link3*c_23+(-link2*q3-link2*q2-q1*link2)*s_2-link2*c_2-link1)/(link2*s
_2);
 q1=q1+kN*Q1;

 q2=q2+kN*Q2;

 q3=q3+kN*Q3;
}

/*
 La funzione gradiente() implementa l&apos;algoritmo del gradiente. Questo
algoritmo si basa, come l&apos;algoritmo di Newton
 sulla conoscenza della matrice Jacobiana J(q), ma per effettuare la stima dei
```

```
parametri si utilizza la sua trasposta
 negata.
 L&apos;idea è quella di percorrere la direzione del gradiente in cerca di
minimi locali. Vi sono numerose implementazioni di
 questo algoritmo (ADAM,RMSProp,Antigradiente,AdaGrad etc), in questo
robot è stato implementato l&apos;algoritmo della discesa
 del gradiente in cui si sceglie la direzione massima di discesa del gradiente per
ottenere il minimo locale della funzione
 del parametro.
 VANTAGGI:
   - In vicinanza delle singolarità non occorre correggere la stima;
   - Convergenza asintotica al valore vero;
 SVANTAGGI:
   -  Forte dipendenza con il learning rate kk: per valori troppo alti, la stima dei
parametri non converge; per valori
                  troppo bassi, la convergenza diventa molto lenta.
  -  Affetto da rumori;
  -  La stima può essere costante nel caso in qui la derivata di q è nulla
*/
void gradiente() {
 if (gomitoG%2!=0) {
   q2g=q2g+gomito*PI;
   gomitoG++;
 }
 float s_2=sin(q2g);
 if(s_2>-0.9) s_2+=-0.01;
float phig =phi;
 float s_1=sin(q1g);
 float c_2=cos(q2g);
 float c_1=cos(q1g);
 float c_3=cos(q3g);
 float s_3=sin(q3g);
 float c_123=c_1*c_2*c_3-s_1*s_2*c_3-s_1*c_2*s_3-c_1*s_2*s_3;
 float s_123=-s_1*s_2*s_3+s_1* c_2*c_3+c_1*s_2*c_3+c_1*c_2*s_3;
 float s_12=c_1*s_2 + c_2*s_1;
 float c_12=c_1*c_2-s_1*s_2;
```

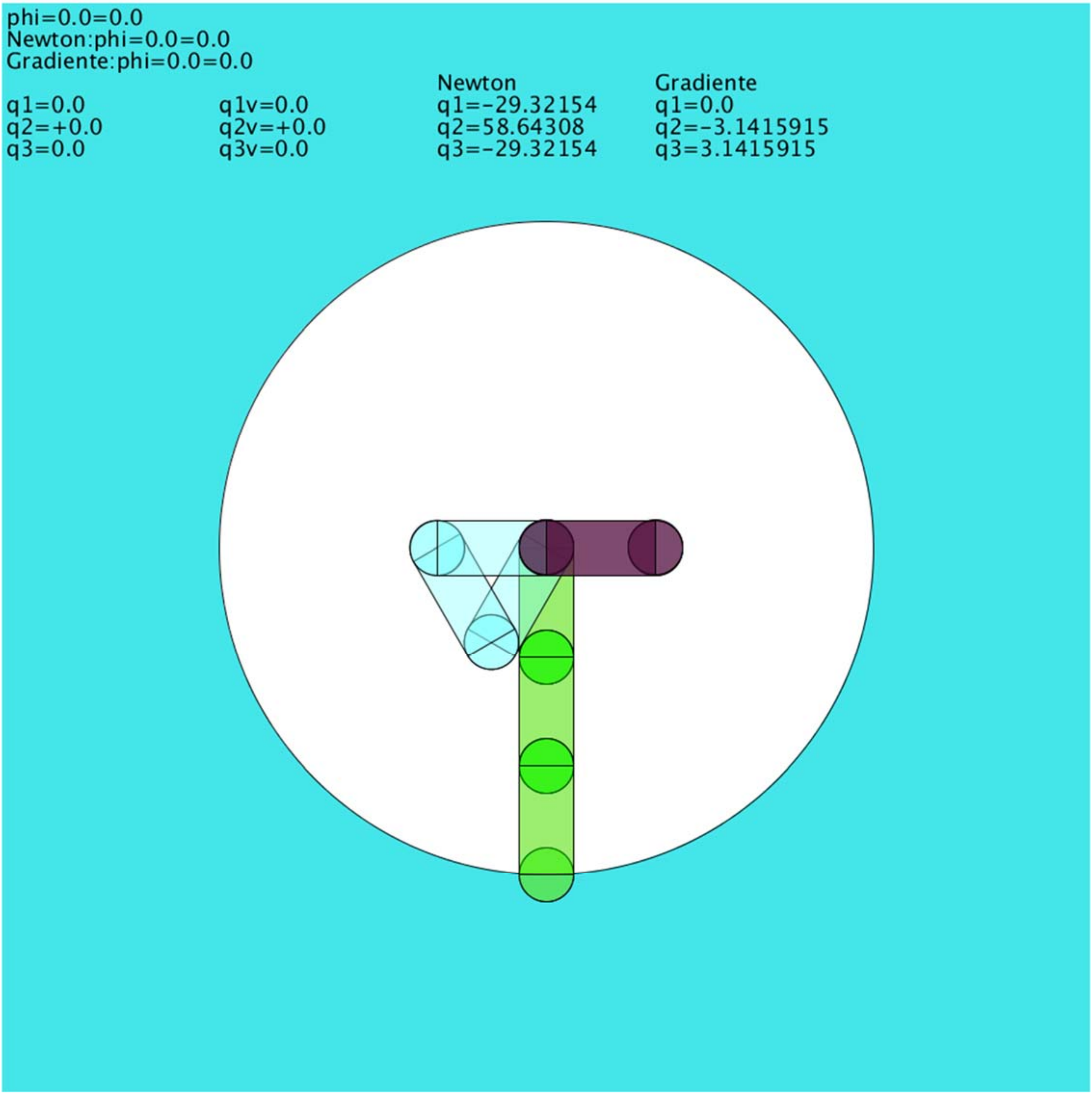
```
 float s_23=c_2*s_3+s_2*c_3;
 float L_3=link3;
 float L_2=link2;
 float L_1=link1;
 float
Q1g=((L_3*c_123+L_2*c_12+L_1*c_1)*y+(-L_3*s_123-L_2*s_12-L_1*s_1
)*x+phi-q3g-q2g-q1g);
 float
Q2g=((L_3*c_123+L_2*c_12)*y+(-L_3*s_123-L_2*s_12)*x+L_1*L_3*s_23
+L_1*L_2*s_2+phi-q3g-q2g-q1g);
 float
Q3g=(L_3*c_123*y-L_3*s_123*x+phi+L_1*L_3*s_23+L_2*L_3*s_3-q3g-q
2g-q1g);

   q1g=q1g+kk*Q1g;

   q2g=q2g+kk*Q2g;
   phig=phig+0.00000000000000001*Q3g;
   q3g=phig-q1g-q2g;

}

/*
 Implementazione della cinematica del 3DOF
*/
void mousePressed() {
 x=mouseX-500;
 y=mouseY-500;
 println(x, y);

 float xpolso= x-cos(phi)*link3;
 float ypolso= y-sin(phi)*link3;
 println("XPOLSO="+xpolso+"YPOLSO="+ypolso);
 if (sqrt(abs(pow(xpolso, 2)+pow(ypolso, 2)))<(link2+link1) &&
sqrt(abs(pow(xpolso, 2)+pow(ypolso, 2)))>abs(link2-link1)) {
   target_b=0;
```

```
  target_r=0;
  target_g=255;
  println("Punto raggiungibile");
 } else {
  target_b=0;
  target_r=255;
  target_g=0;
  println("Punto non raggiungibile");
 }

 float
a=(xpolso*xpolso+ypolso*ypolso-link1*link1-link2*link2)/(2*link1*link2);
 float c2=a;
 float s2=gomito*sqrt(abs(1-a*a));
 float b1=link1+c2*link2;
 float b2=link2*s2;
 float c1=b1*xpolso+b2*ypolso;
 float s1=-b2*xpolso+b1*ypolso;
 angolo1 = atan2(s1, c1)-PI/2;
 angolo2 = atan2(s2, c2);
 angolo3=phi-(angolo2+angolo1)-PI/2;
 println("Angolo1="+angolo1+"\tAngolo2:"+angolo2+"\tAngolo3:"+angolo3);
}
```

Risultati 3 DOF

N.B.:

- in viola 3DOF con algoritmo del gradiente;
- in celeste 3DOF con algoritmo di newton;
- in verde robot vero;
- in arancione robot phantom;




phi=0.523599=-1.0471973
Newton:phi=0.523599=0.5236037
Gradiente:phi=0.523599=0.5235987
q1=-3.349103
q2=+1.7768532
q3=0.5250524
q1v=-3.349104
q2v=+1.7768526
q3v=0.52505267
Newton
q1=-12.567828
q2=17.07271
q3=-3.9812782
Gradiente
q1=-1.7586368
q2=-4.5207477
q3=6.8029833

sketch_23_DHstrutture

```
import java.awt.event.KeyEvent;
int depth=0;
float q1=0;
float q2=0;
float q3=0;
float q4=0;
float q5=0;
float q6=0;
String input="";
String robotName="";
float q1v=0;
float q2v=0;
float q3v=0;
float q4v=0;
float q5v=0;
float q6v=0;
float Kp=0.01;
float Kchar=1.0;

float angoloX=0;
float angoloY=0;
float angoloXp=0;
float angoloYp=0;

float L1=80;
float D1=50;
float D2=50;
float D4=50;
float D6=50;
float L2=80;
float L3=80;
float L6 = 80;
void setup()
{
 size(1000, 1000, P3D);
 background(#607d8b);
}

void draw()
{
 background(#607d8b);
 details(q1, q2, q3,q4, q5, q6, q1v, q2v, q3v,q4v, q5v, q6v);
 translate(width/2, height/2, depth);
```

```
 q1v=q1v-Kp*(q1v-q1);
 q2v=q2v-Kp*(q2v-q2);
 q3v=q3v-Kp*(q3v-q3);
 q4v=q4v-Kp*(q4v-q4);
 q5v=q5v-Kp*(q5v-q5);
 q6v=q6v-Kp*(q6v-q6);

 rotateY(-angoloY);
 rotateX(angoloX);
 rotateY(PI/2.0);
 rotateX(PI/2.0);
 rotateZ(PI/2.0);

 directionalLight(126, 126, 126, 0, 0, 0.7);
 ambientLight(200, 200, 200);
 fill(#f4511e);
 base();
 fill(#F0D01D, 100);
 noStroke();
 pushMatrix();
 robot(q1, q2, q3,q4,q5,q6);
 popMatrix();
 fill(0, 255, 0);

 robot(q1v, q2v, q3v,q4v,q5v,q6v);
}
/*
 Implementazione di tutti i robot utilizzando la tabella di Denavit-Hartenberg
*/
void robot(float a1, float a2, float a3,float a4,float a5,float a6) {
 switch(robotName) {
 case "cartesiano":
   link(0, a1, -PI/2, 0);
   link(-PI/2, a2, -PI/2, 0);
   link(0, a3, 0, 0);
   break;
 case "cilindrico":
link(a1,L1, 0, 0);
   link(0, a2, -PI/2, 0);
   link(0, a3, 0, 0);
   break;
 case "scara":
   link(a1,L1, 0, D1);
```

```
  link(a2,0,0, D2);
  link(0, a3, 0, 0);

  break;
case "sfericoI":
  link(a1,L1, PI/2, 0);
  link(a2,0,PI/2, L2);
  link(0, a3, 0, 0);
  break;
case "sfericoII":
  link(a1,L1, -PI/2, 0);
  link(a2,L2,PI/2,0);
  link(-PI/2, a3, 0, 0);

  break;
case "antropomorfo":
      link(a1,L1, PI/2, 0);
  link(a2,0,0,L2);
  link( a3,0, 0, L3);

  break;
case "puma":
      link(a1,D1, -PI/2, 0);
  link(a2,0,0,L2);
  link( a3,0, PI/2, 0);
      link(a4,D4, -PI/2, 0);
  link(a5,0,PI/2,0);
  link(a6, D6, 0, 0);

  break;
case "stanford":
 link(a1,L1, -PI/2, 0);
  link(a2,L2,PI/2,0);
  link(-PI/2, a3, 0, 0);
      link(a4,0, -PI/2, 0);
  link(a5,0,PI/2,0);
  link(a6, L6, 0, 0);

  break;

default:
  robotName="";
  break;
}
```

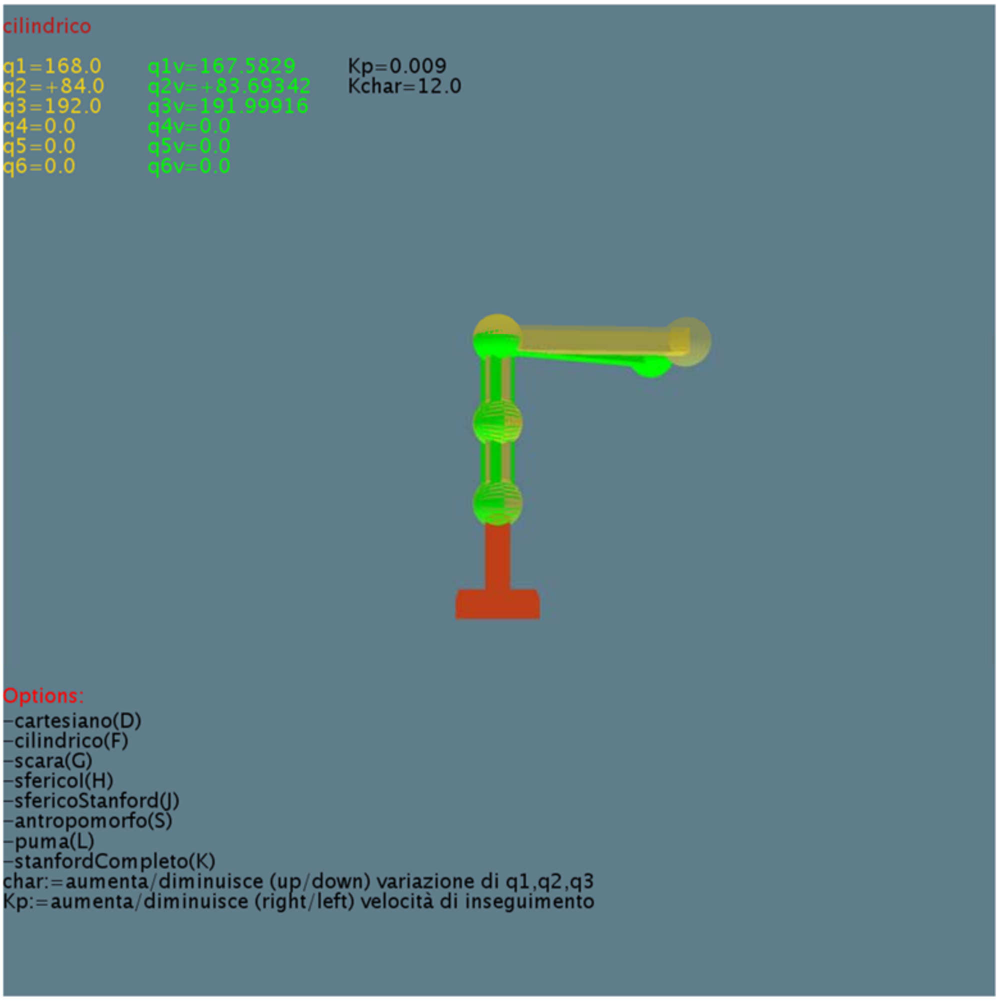
```
}
/*
La funzione details() contiene tutte le informazioni visualizzate a schermo
*/
void details(float a1, float a2, float a3, float a4,float a5,float a6,float a1v, float a2v, float a3v,float
a4v,float a5v,float a6v) {
 String line="q1="+a1+"\nq2=+"+a2+"\nq3="+a3+"\nq4="+a4+"\nq5="+a5+"\nq6="+a6;
 String
lineV="q1v="+a1v+"\nq2v=+"+a2v+"\nq3v="+a3v+"\nq4v="+a4v+"\nq5v="+a5v+"\nq6v="+a6v;
 String car="Kp="+Kp+"\nKchar="+Kchar+"\n";
 String
options="-cartesiano(D)\n-cilindrico(F)\n-scara(G)\n-sfericoI(H)\n-sfericoStanford(J)\n-antropomorfo(
S)\n-puma(L)\n-stanfordCompleto(K)\nchar:=aumenta/diminuisce (up/down) variazione di
q1,q2,q3\nKp:=aumenta/diminuisce (right/left) velocità di inseguimento";
 textSize(20);
 textLeading(20);
 fill(#b71c1c);
 text(robotName, 5, 30);
 fill(#F0D01D);
 textLeading(20);
 text(line, 5, 70);
 textLeading(20);
 fill(0, 255, 0);
 text(lineV, 150, 70);
 fill(0);
 textLeading(20);
 text(car, 350, 70);
 textLeading(20);
 fill(255, 0, 0);
 text("Options:\n", 5, 700);
 textLeading(20);
 fill(0);
 textLeading(20);
 text(options, 5, 725);
}

void base() {
 pushMatrix();
 translate(0, 0, -100);
 box(80, 80, 20);
 popMatrix();
 pushMatrix();
 translate(0, 0, -50);
```

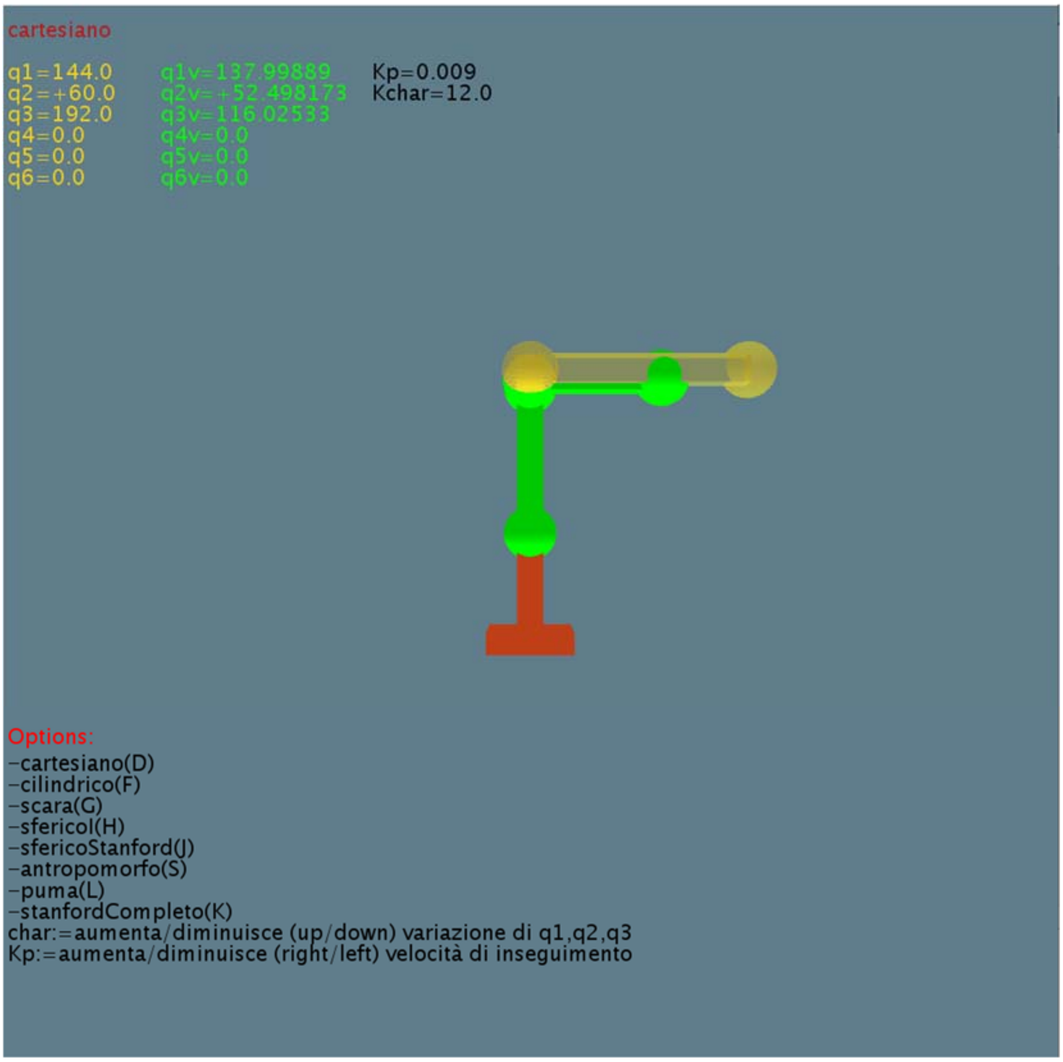
```
box(25, 25, 80);
popMatrix();
}



void link(float theta, float d, float alpha, float a) {
 rotateZ(theta);
 sphere(25);
 translate(0.0, 0.0, d/2);
 box(25, 25, d);
 translate(0.0, 0.0, d/2);
 sphere(25);
 rotateX(alpha);
 translate(a/2, 0.0, 0.0);
 box(a, 25, 25);
 translate(a/2, 0.0, 0.0);
}

void mousePressed() {
 angoloYp=angoloY+PI*mouseX/100000.0;
 angoloXp=angoloX+PI*mouseY/100000.0;
}

void mouseDragged() {
 angoloY=angoloY+PI*mouseX/100000.0;
 angoloX=angoloX+PI*mouseY/100000.0;
}

void keyPressed() {
 if (keyCode==&apos;R&apos;) {
   q1=0.0;
   q2=0.0;
   q3=0.0;
   angoloX=0;
   angoloY=0;
   Kp=0.02;
   Kchar=1;
 }
 if (keyCode==&apos;1&apos;) {
   q1+=Kchar*1;
 }
 if (keyCode==&apos;2&apos;) {
   q2+=Kchar*1;
 }
```

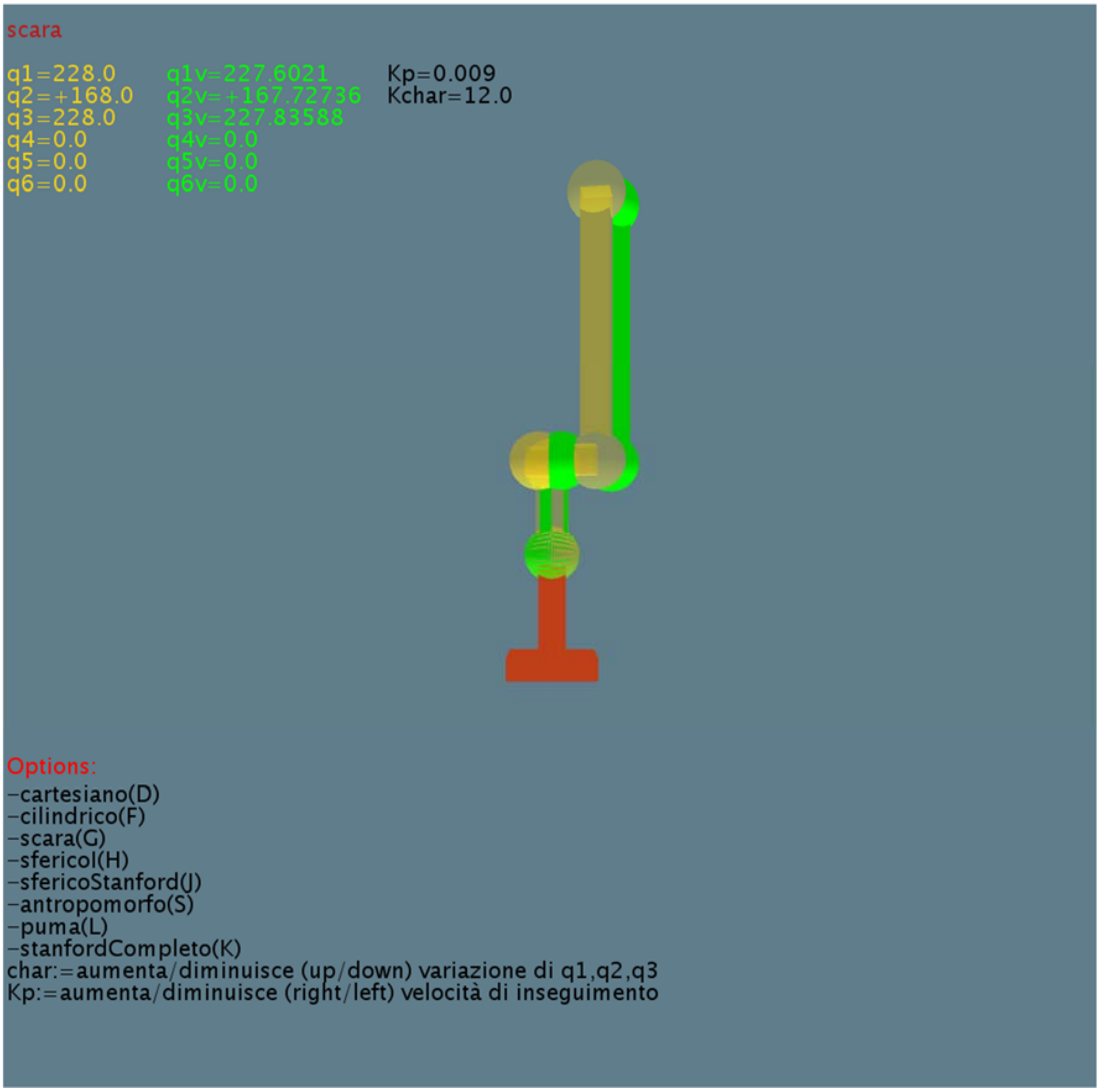
```
if (keyCode==&apos;3&apos;) {
  q3+=Kchar*1;
}
 if (keyCode==&apos;4&apos;) {
  q4+=Kchar*1;
}
if (keyCode==&apos;5&apos;) {
  q5+=Kchar*1;
}
if (keyCode==&apos;6&apos;) {
  q6+=Kchar*1;
}
if (keyCode==&apos;9&apos;) {
  q1-=Kchar*1;
}
if (keyCode==&apos;8&apos;) {
  q2-=Kchar*1;
}
if (keyCode==&apos;7&apos;) {
  q3-=Kchar*1;
}
 if (keyCode==&apos;Z&apos;) {
  q4-=Kchar*1;
}
if (keyCode==&apos;X&apos;) {
  q5-=Kchar*1;
}
if (keyCode==&apos;Y&apos;) {
  q6-=Kchar*1;
}
if (keyCode==LEFT) {
  Kp+=0.001;
}
if (keyCode==RIGHT) {
  Kp-=0.001;
}
if (keyCode==UP) {
  Kchar+=1;
}
if (keyCode==DOWN) {
  Kchar-=1;
}

if (keyCode==&apos;D&apos;) {
```

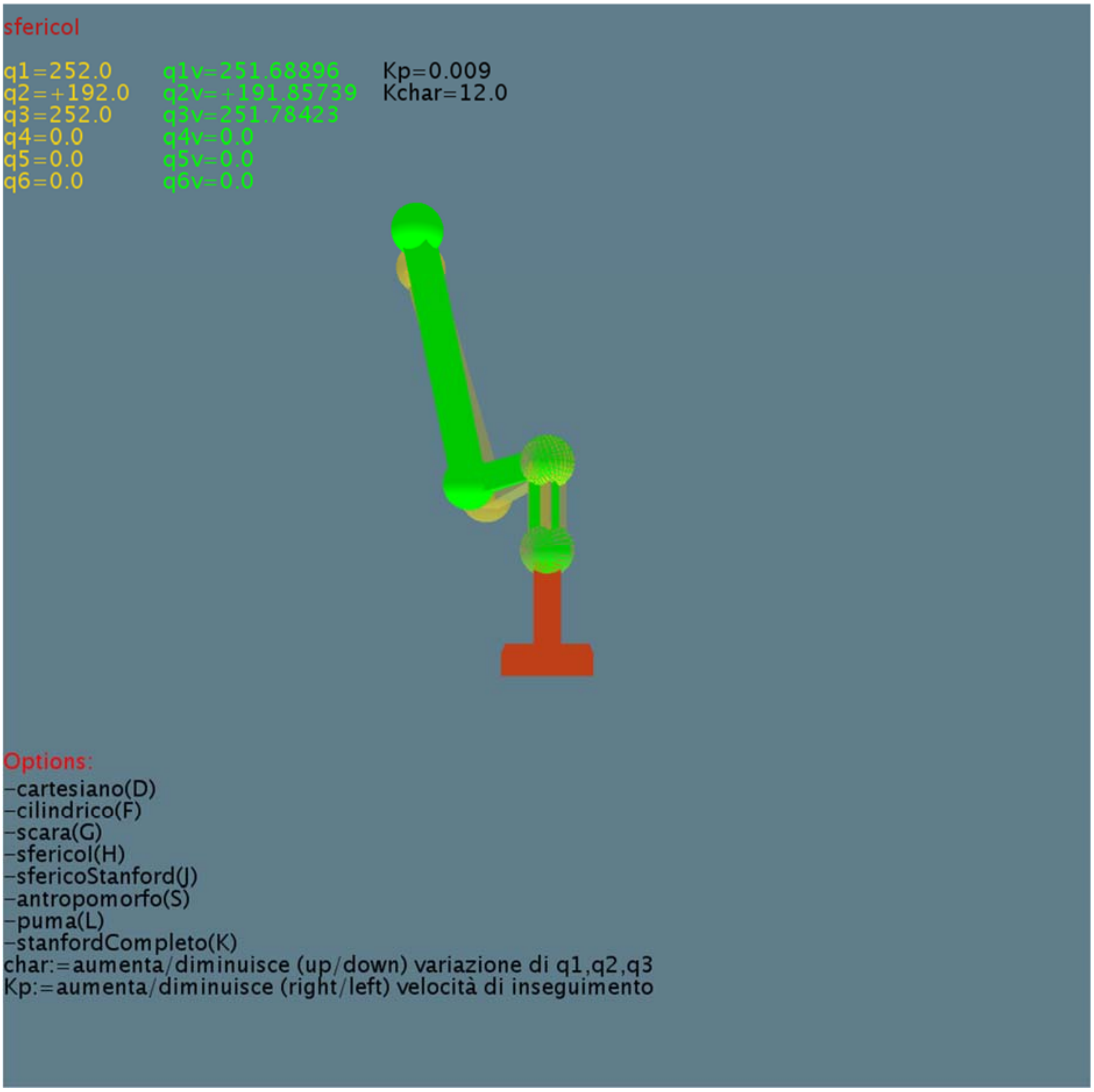
```
  robotName="cartesiano";
}
if (keyCode==&apos;F&apos;) {
  robotName="cilindrico";
}
if (keyCode==&apos;G&apos;) {
  robotName="scara";
}
if (keyCode==&apos;H&apos;) {
  robotName="sfericoI";
}
if (keyCode==&apos;J&apos;) {
  robotName="sfericoII";
}
if (keyCode==&apos;K&apos;) {
  robotName="stanford";
}
if (keyCode==&apos;L&apos;) {
  robotName="puma";
}
if (keyCode==&apos;S&apos;) {
  robotName="antropomorfo";
}


}
```

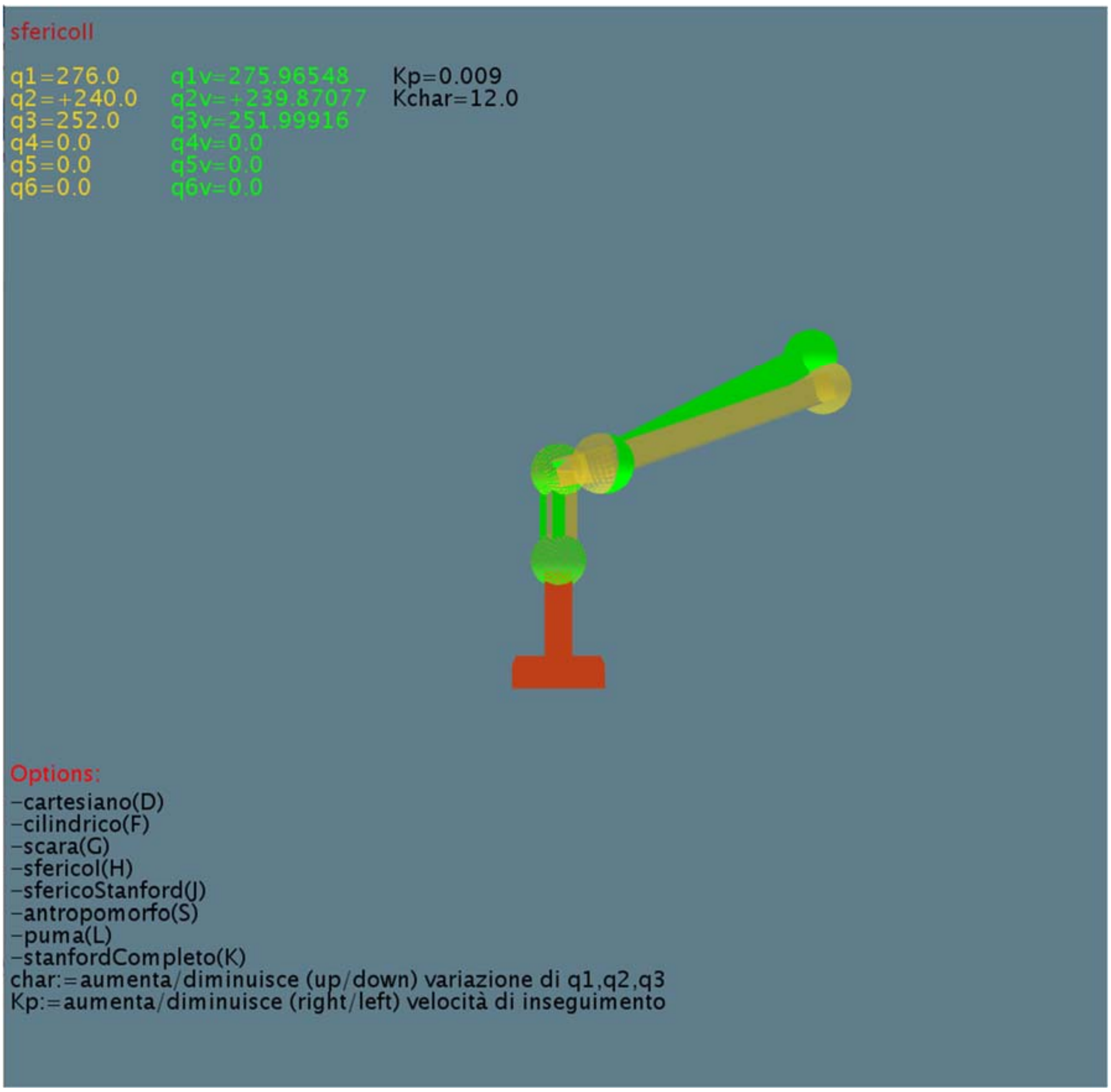
Risulati strutture Denavit

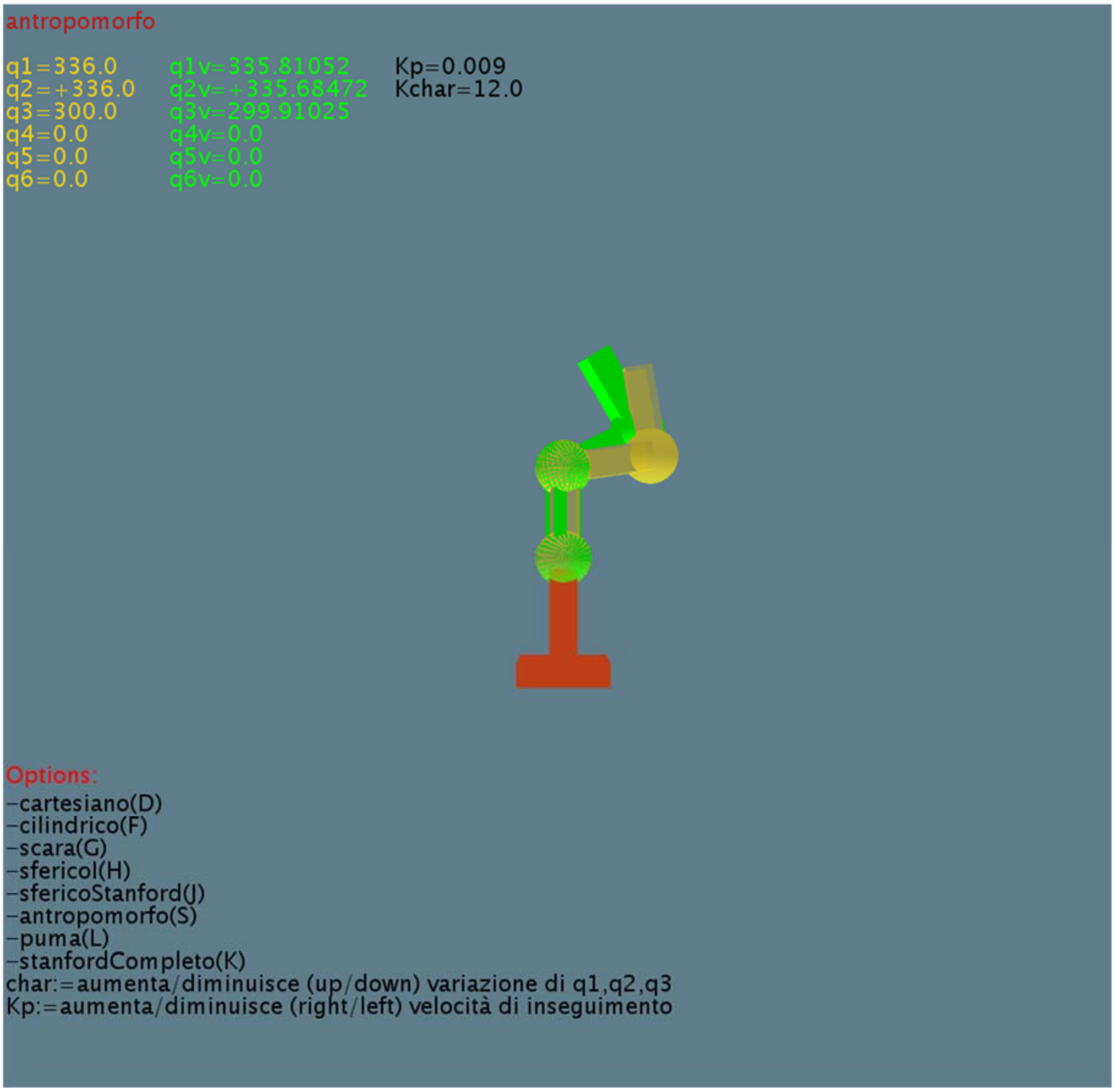


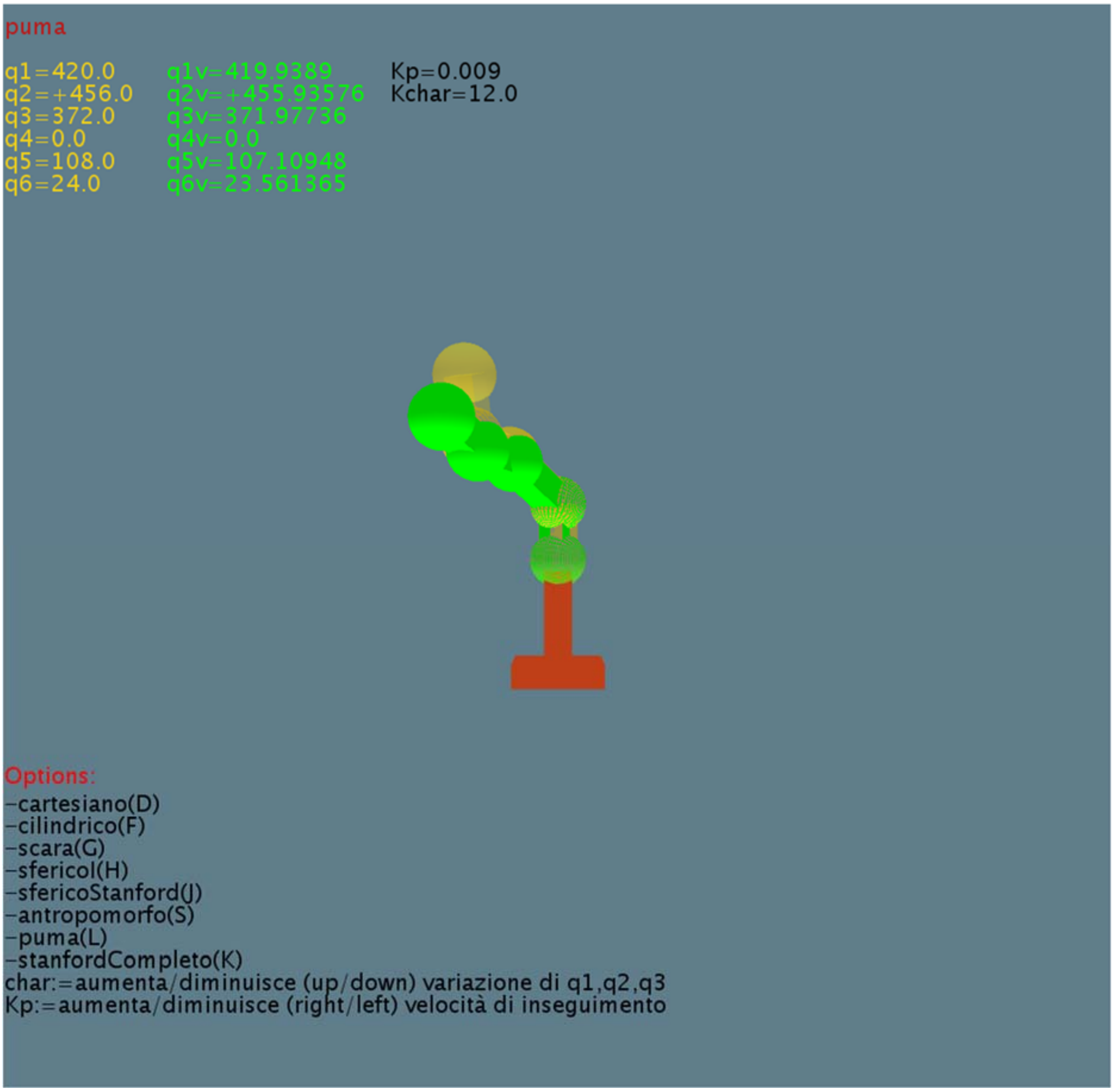

cartesiano
q1=144.0
q2=+60.0
q3=192.0
q4=0.0
q5=0.0
q6=0.0
q1v=137.99889
q2v=+52.498173
q3v=116.02533
q4v=0.0
q5v=0.0
q6v=0.0
Kp=0.009
Kchar=12.0
Options:
-cartesiano(D)
-cilindrico(F)
-scara(G)
-sfericoI(H)
-sfericoStanford(J)
-antropomorfo(S)
-puma(L)
-stanfordCompleto(K)
char:=aumenta/diminuisce (up/down) variazione di q1,q2,q3
Kp:=aumenta/diminuisce (right/left) velocità di inseguimento

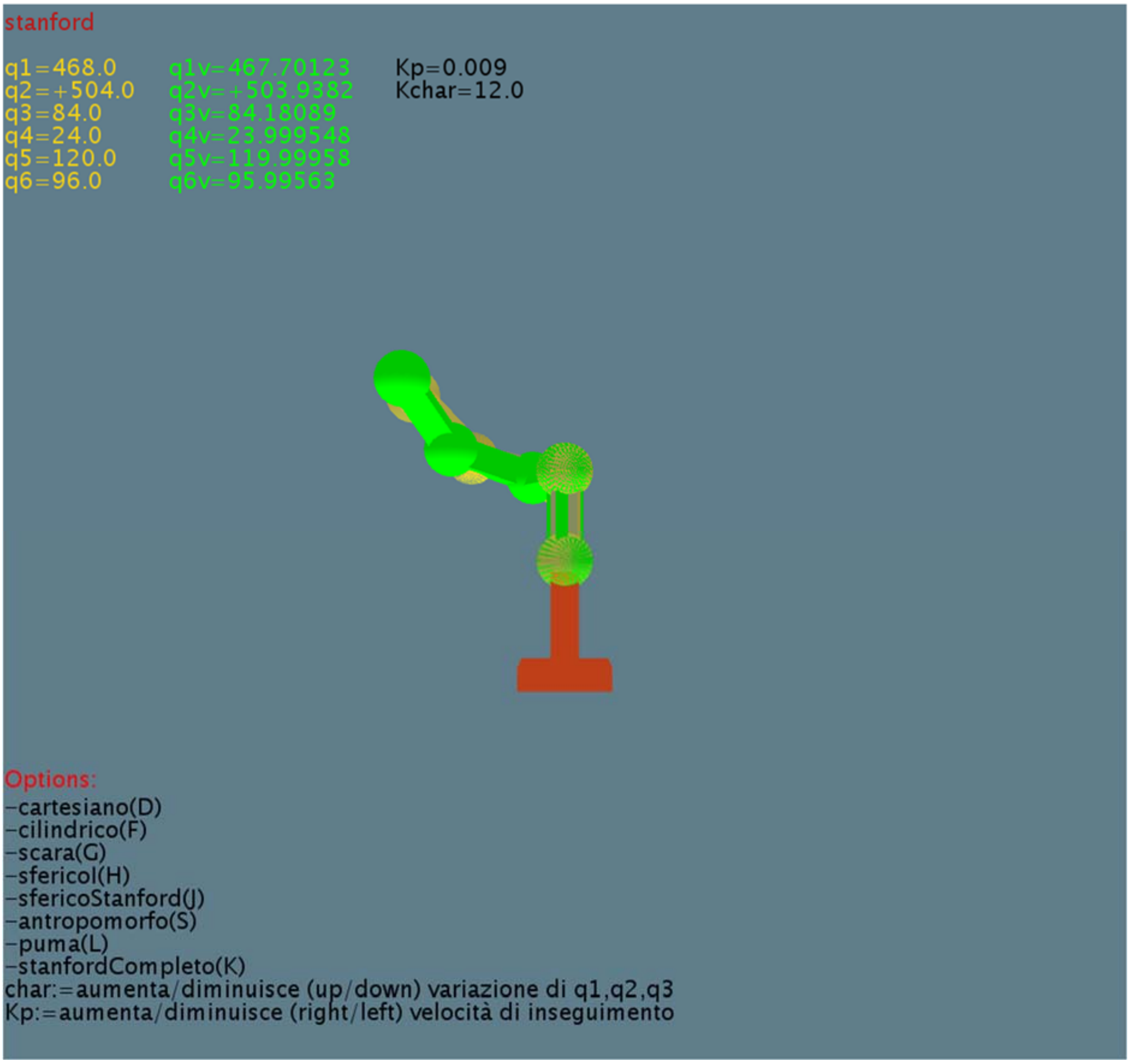

scara
q1=228.0
q2=+168.0
q3=228.0
q4=0.0
q5=0.0
q6=0.0
q1v=227.6021
q2v=+167.72736
q3v=227.83588
q4v=0.0
q5v=0.0
q6v=0.0
Kp=0.009
Kchar=12.0
Options:
-cartesiano(D)
-cilindrico(F)
-scara(G)
-sfericoI(H)
-sfericoStanford(J)
-antropomorfo(S)
-puma(L)
-stanfordCompleto(K)
char:=aumenta/diminuisce (up/down) variazione di q1,q2,q3
Kp:=aumenta/diminuisce (right/left) velocità di inseguimento


sfericol
q1=252.0
q2=+192.0
q3=252.0
q4=0.0
q5=0.0
q6=0.0
q1v=251.68896
q2v=+191.85739
q3v=251.78423
q4v=0.0
q5v=0.0
q6v=0.0
Kp=0.009
Kchar=12.0
Options:
-cartesiano(D)
-cilindrico(F)
-scara(G)
-sfericol(H)
-sfericoStanford(J)
-antropomorfo(S)
-puma(L)
-stanfordCompleto(K)
char:=aumenta/diminuisce (up/down) variazione di q1,q2,q3
Kp:=aumenta/diminuisce (right/left) velocità di inseguimento


sfericoII
q1=276.0
q2=+240.0
q3=252.0
q4=0.0
q5=0.0
q6=0.0
q1v=275.96548
q2v=+239.87077
q3v=251.99916
q4v=0.0
q5v=0.0
q6v=0.0
Kp=0.009
Kchar=12.0
Options:
-cartesiano(D)
-cilindrico(F)
-scara(G)
-sfericoI(H)
-sfericoStanford(J)
-antropomorfo(S)
-puma(L)
-stanfordCompleto(K)
char:=aumenta/diminuisce (up/down) variazione di q1,q2,q3
Kp:=aumenta/diminuisce (right/left) velocità di inseguimento


antropomorfo
q1=336.0
q2=+336.0
q3=300.0
q4=0.0
q5=0.0
q6=0.0
q1v=335.81052
q2v=+335.68472
q3v=299.91025
q4v=0.0
q5v=0.0
q6v=0.0
Kp=0.009
Kchar=12.0
Options:
-cartesiano(D)
-cilindrico(F)
-scara(G)
-sfericoI(H)
-sfericoStanford(J)
-antropomorfo(S)
-puma(L)
-stanfordCompleto(K)
char:=aumenta/diminuisce (up/down) variazione di q1,q2,q3
Kp:=aumenta/diminuisce (right/left) velocità di inseguimento


puma
q1=420.0
q2=+456.0
q3=372.0
q4=0.0
q5=108.0
q6=24.0
q1v=419.9389
q2v=+455.93576
q3v=371.97736
q4v=0.0
q5v=107.10948
q6v=23.561365
Kp=0.009
Kchar=12.0
Options:
-cartesiano(D)
-cilindrico(F)
-scara(G)
-sfericoI(H)
-sfericoStanford(J)
-antropomorfo(S)
-puma(L)
-stanfordCompleto(K)
char:=aumenta/diminuisce (up/down) variazione di q1,q2,q3
Kp:=aumenta/diminuisce (right/left) velocità di inseguimento


stanford
q1=468.0
q2=+504.0
q3=84.0
q4=24.0
q5=120.0
q6=96.0
q1v=467.70123
q2v=+503.9382
q3v=84.18089
q4v=23.999548
q5v=119.99958
q6v=95.99563
Kp=0.009
Kchar=12.0
Options:
-cartesiano(D)
-cilindrico(F)
-scara(G)
-sfericoI(H)
-sfericoStanford(J)
-antropomorfo(S)
-puma(L)
-stanfordCompleto(K)
char:=aumenta/diminuisce (up/down) variazione di q1,q2,q3
Kp:=aumenta/diminuisce (right/left) velocità di inseguimento

Procedura 26

Scrivere una procedura Maxima che, prendendo in ingresso la tabella di Denavit-Hartenberg e le informazioni necessarie per individuare i baricentri dei link, restituisca: energia cinetica dovuta alla rotazione ed energia cinetica dovuta alla traslazione per ogni link e per l'intero robot; la matrice delle inerzie generalizzate per ogni link e per l'intero robot.

```
(%i1) kill(all);
```

(%o0) **done**

Procedure ausiliarie per il calcolo della cinemarica diretta in accordo a Denavit-Hartenberg

```
(%i1) inverseLaplace(SI,theta):=block([res,M,MC,aC,b],
                              M:SI,
                              MC:SI,
                              for i:1 thru 3 do(
                                for j:1 thru 3 do
                                   (
                                     aC:M[i,j],
                                     b:ilt(aC,s,theta),
                                     MC[i,j]:b
                                    )
                                 ),
                              res:MC

                           )
```

(%o1) $\mathrm{inverseLaplace}(\mathrm{SI},\vartheta):=\mathbf{block}\ ([\mathrm{res},M,\mathrm{MC},\mathrm{aC},b],M\colon\mathrm{SI},\mathrm{MC}\colon\mathrm{SI},$
$\mathbf{for}\ i\ \mathbf{thru}\ 3\ \mathbf{do}\ \mathbf{for}\ j\ \mathbf{thru}\ 3\ \mathbf{do}\ (\mathrm{aC}\colon M_{i,j},b\colon\mathrm{ilt}(\mathrm{aC},s,\vartheta),\mathrm{MC}_{i,j}\colon b),\mathrm{res}\colon\mathrm{MC})$

```
(%i2) rotLaplace(k,theta):=block([res,S,I,temp],
                          S:ident(3),
                          I:ident(3),
                        for i:1 thru 3 do
                          (
                          for j:1 thru 3 do
                              (
                                if i=j
                                   then S[i][j]:0
                                elseif j>i
                                   then (
                                  temp:(-1)^(j-i)*k[3-remainder(i+j,3)],
                                         S[i][j]:temp,
                                         S[j][i]:-temp

                                          )
                                )
                           ),
                          res:inverseLaplace(invert(s*I-S),theta)

                        )
```

(%o2) $\mathrm{rotLaplace}(k,\vartheta):=\mathbf{block}\ ([\mathrm{res},S,I,\mathrm{temp}],S\colon\mathrm{ident}(3),I\colon\mathrm{ident}(3),$
$\mathbf{for}\ i\ \mathbf{thru}\ 3\ \mathbf{do}\ \mathbf{for}\ j\ \mathbf{thru}\ 3\ \mathbf{do}\ \mathbf{if}\ i=j\ \mathbf{then}\ (S_i)_j\colon 0\ \mathbf{elseif}\ j>i\ \mathbf{then}\ (\mathrm{temp}\colon$

$(-1)^{j-i}\, k_{3-\text{remainder}(i+j,3)}, (S_i)_j\text{: temp}, (S_j)_i\text{: } -\text{temp}), \text{res: inverseLaplace(invert}(s\, I - S), \vartheta))$

```
(%i3) Av(v,theta,d):=block([res,Trot,row,Atemp,A],
                          Trot:rotLaplace(v,theta),
                          row:matrix([0,0,0,1]),
                          Atemp:addcol(Trot,d*transpose(v)),
                          A:addrow(Atemp,row),
                          res:A
                          )$
(%i4) Q(theta,d,alpha,a):=block([res,tempMat,Qtrasf],
                               tempMat:Av([0,0,1],theta,d).Av([1,0,0],alpha,a),
                               Qtrasf:zeromatrix(4,4),
                               for i:1 thru 4 do
                          (
                          for j:1 thru 4 do
                             (
                               Qtrasf[i][j]:trigreduce(tempMat[i][j])
                                          )
                             ),

                               res:Qtrasf
                               )
```

(%o4) $Q(\vartheta, d, \alpha, a) := \textbf{block}\ ([\text{res}, \text{tempMat}, \text{Qtrasf}], \text{tempMat: Av}([0,0,1], \vartheta, d) \cdot \text{Av}([1,0,0], \alpha, a), \text{Qtrasf: zeromatrix}(4,4), \textbf{for}\ i\ \textbf{thru}\ 4\ \textbf{do for}\ j\ \textbf{thru}\ 4\ \textbf{do}\ (\text{Qtrasf}_i)_j\text{: trigreduce}((\text{tempMat}_i)_j), \text{res: Qtrasf})$

```
(%i5) Qdirect(DH):=block([res,Q,Qtemp],
                        Q:[Q(DH[1][1],DH[1][2],DH[1][3],DH[1][4])],
                        for i:2 thru length(DH) do(

                             Qtemp:Q(DH[i][1],DH[i][2],DH[i][3],DH[i][4]),

                             Q:append(Q,[trigsimp(trigreduce(trigexpand(Q[i-
      1].Qtemp)))])


                            ),
                         res:Q)
```

(%o5) $\text{Qdirect(DH)} := \textbf{block}\ ([\text{res}, Q, \text{Qtemp}], Q\text{: }[Q((\text{DH}_1)_1, (\text{DH}_1)_2, (\text{DH}_1)_3, (\text{DH}_1)_4)], \textbf{for}\ i\ \textbf{from}\ 2\ \textbf{thru}\ \text{length(DH)}\ \textbf{do}\ (\text{Qtemp: } Q((\text{DH}_i)_1, (\text{DH}_i)_2, (\text{DH}_i)_3, (\text{DH}_i)_4), Q\text{: append}(Q, [\text{trigsimp(trigreduce(trigexpand}(Q_{i-1} \cdot \text{Qtemp}))))]), \text{res: } Q)$

Qbc(Q,bc):=prende in ingresso la matrice Q della cinematica diretta ed applica la traslazione necessaria a portare il sistema di riferimento nel baricentro.

```
(%i6) Qbc(Q,bc,dist):=block([traslBC,Qcap],
                    Qcap:[],
                    ex:matrix([1],[0],[0]), ez:matrix([0],[0],[1]),
                    for j:1 thru length(Q) do(
                            traslBC:addrow(addcol(ident(3),dist[j]),[0,0,0,1]),
                            Qcap:append(Qcap,[trigsimp(Q[j].traslBC)])

                            ),
                    Qcap
                    )
```

(%o6) $\mathrm{Qbc}(Q,\mathrm{bc},\mathrm{dist}) := \mathbf{block}\left([\mathrm{traslBC},\mathrm{Qcap}],\mathrm{Qcap}:[\,],\mathrm{ex}:\begin{pmatrix}1\\0\\0\end{pmatrix},\mathrm{ez}:\begin{pmatrix}0\\0\\1\end{pmatrix},\right.$
$\mathbf{for}\ j\ \mathbf{thru}\ \mathrm{length}(Q)\ \mathbf{do}\ (\mathrm{traslBC}:\mathrm{addrow}(\mathrm{addcol}(\mathrm{ident}(3),\mathrm{dist}_j),[0,0,0,1]),\mathrm{Qcap}:\mathrm{append}(\mathrm{Qcap},$
$\left.[\mathrm{trigsimp}(Q_j\cdot\mathrm{traslBC})])),\mathrm{Qcap}\right)$

inerzia(j):= funzione che associa al link j-esimo la corrispettiva matrice di interzia.

```
(%i7) inerzia(j):=block(
      [II],
      II:matrix([alpha[xx][i],alpha[xy][i],alpha[xz][i]],
                [alpha[xy][i],alpha[yy][i],alpha[yz][i]],
                [alpha[xz][i],alpha[yz][i],alpha[zz][i]]),
      II:subst(j, i, II),
      return(II)
      )
```

(%o7) $\mathrm{inerzia}(j) := \mathbf{block}\left([\mathrm{II}],\mathrm{II}:\begin{pmatrix}(\alpha_{\mathrm{xx}})_i & (\alpha_{\mathrm{xy}})_i & (\alpha_{\mathrm{xz}})_i\\(\alpha_{\mathrm{xy}})_i & (\alpha_{\mathrm{yy}})_i & (\alpha_{\mathrm{yz}})_i\\(\alpha_{\mathrm{xz}})_i & (\alpha_{\mathrm{yz}})_i & (\alpha_{\mathrm{zz}})_i\end{pmatrix},\mathrm{II}:\mathrm{subst}(j,i,\mathrm{II}),\mathrm{return}(\mathrm{II})\right)$

```
(%i8) massa(k):=M[k];
```

(%o8) $\mathrm{massa}(k) := M_k$

ek(DH):=funzione responsabile del calcolo dell'energia cinetica dell'intero robot.

$$\mathrm{DH} = \begin{pmatrix}\theta & d & \alpha & a\\ \vdots & \vdots & \vdots & \vdots\end{pmatrix}$$

Calcolo cinematica diretta in accordo all'algoritmo di Denavit-Hartenberg:

$$Q_{0,n} = Q_{01}Q_{12}\ldots Q_{n-1,n} \quad \text{con } n \equiv \#\mathrm{DOF}$$

Applico traslazioni necessarie per portare il sistema il $\mathrm{SR}_{i-1,i}$ con l'origine coincidente con il baricentro del link:

$$\hat{Q}_{01} = Q_{01}\begin{pmatrix}I & d\\0 & 1\end{pmatrix}, \hat{Q}_{12}\ldots\hat{Q}_{n-1,n} \quad \text{in cui } d \text{ sono le coordinate del baricentro del link} - \frac{L_i}{2}$$

A questo punto, l'energia cinetica del link i-esimo:

$$T_i = T_{i_a} + T_{i_b} = \frac{1}{2}\omega_i^T R_i \mathbb{I}_i R_i^T \omega_I + \frac{1}{2}M_i \dot{d}_i^T \dot{d}_i$$

In cui:

$$\omega_i \equiv \dot{q}_i\, e_k \qquad \text{con}\, k \in \{x, z\}\, \text{in base all'asse su cui avviene la rotazione}$$

In particolare:

$$\omega_i = \omega = \begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix} \text{ottenuto da}\, S(\omega) = \begin{pmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ \omega_y & \omega_x & 0 \end{pmatrix} = \dot{R}R^T$$

$$R := \text{matrice di rotazione associata}\, a\, \hat{Q}_{i-1,i}$$

$$\mathbb{I} := \text{matrice di inerzia del link}\, i - \text{esimo}$$

$$M_i := \text{massa dell'}i - \text{esimo link}$$

$$d_i := \text{coordinate di posizione associata}\, a\, \hat{Q}_{i-1,i}$$

$$T_a := \text{energia cinetica associata alla rotazione}$$

$$T_b := \text{energia cinetica associata alla traslazione}$$

Inoltre si definisce, la matrice delle inerzie generalizzate B_i, nel seguente modo:

$$T_i = T_{i_a} + T_{i_b} = \frac{1}{2}(\ \dot{q}_1 \ldots \dot{q}_n\ ) B_i \begin{pmatrix} \dot{q}_1 \\ \vdots \\ \dot{q}_n \end{pmatrix}$$

In altri termini è la forma quadratica corrispondende all'energia totale del link i-esimo. In particolare, per l'intero robot la matrice delle interzie generalizzate:

$$B = B_1 + \cdots + B_n$$

(%i9)
```
ek(DH,dist):=block([Q,Qcap,I,wtemp,w,Si,Tatemp,Ta,Tbtemp,Tb,d,dd,Qend,B,
Btemp,T,Tot,Btot,res],
                    I:[],w:[],Ta:[],Tb:[],B:[],T:[],Ttot:0,
                    depends([q,omega],t),
                    Q:Qdirect(DH),

                    Qcap:Qbc(Q,DH,dist),

                   for i:1 thru length(Qcap) do( I:append(I,[inerzia(i)]),
                        R:matrix([Qcap[i][1][1],Qcap[i][1][2],
Qcap[i][1][3]],[Qcap[i][2][1],Qcap[i][2][2],Qcap[i][2][3]],[Qcap[i][3][1],
Qcap[i][3][2],Qcap[i][3][3]]),
                       dR:diff(R,t),
                       for j:1 thru length(DH) do(
                           dR:subst('diff(q[j],t)=omega[j],dR)),
                       Sw:dR.transpose(R),
                       wtemp:matrix([Sw[3][2]],[Sw[1][3]],[Sw[2][1]]),
                       w:append(w,[trigreduce(expand(wtemp))]),
                  Tatemp:(1/2)*transpose(wtemp).R.I[i].transpose(R).wtemp,
                  Tatemp:trigsimp(trigreduce(trigexpand(Tatemp))),
                   Ta:append(Ta,[Tatemp]),
                   d:matrix([Qcap[i][1][4]],[Qcap[i][2][4]],[Q[i][3][4]]),
                   dd:diff(d,t),
                   for j:1 thru length(DH)
do(dd:subst('diff(q[j],t)=omega[j],dd)),Tbtemp:(massa(i)/
2)*trigsimp(trigreduce(trigexpand(transpose(dd).dd))),
                   Tb:append(Tb,[Tbtemp]),
                   T:append(T,[trigreduce(Tatemp+Tbtemp)])),
                   for i:1 thru length(DH) do(
                     Ttot:T[i]+Ttot
                   ),
                   B:zeromatrix(length(DH),length(DH)),
                   for i:1 thru length(DH) do(
                     B[i][i]:coeff(collectterms(expand(2*Ttot),omega[i]^2),
omega[i],2),
                     for j:1 thru length(DH) do(
                     if i#j then
                        (
                          B[i][j]:ratsimp(coeff(coeff(expand(Ttot*(1/2)),
omega[i],1),omega[j],1))
                         )
                   )),
                   if length(DH)#2 then(
                   res:[[Ta[1],Tb[1],T[1]],
                        [Ta[2],Tb[2],T[2]],
                        [Ta[3],Tb[3],T[3]],[B]])
                   else (res:[[Ta[1],Tb[1],T[1]],
                        [Ta[2],Tb[2],T[2]],[B]])
                    )
```

(%o9) $\mathrm{ek}(\mathrm{DH},\mathrm{dist}) := \mathbf{block}\Big([Q,\mathrm{Qcap},I,\mathrm{wtemp},w,\mathrm{Si},\mathrm{Tatemp},\mathrm{Ta},\mathrm{Tbtemp},\mathrm{Tb},d,\mathrm{dd},\mathrm{Qend},B,$ $\mathrm{Btemp},T,\mathrm{Tot},\mathrm{Btot},\mathrm{res}],I\colon[],w\colon[],\mathrm{Ta}\colon[],\mathrm{Tb}\colon[],B\colon[],T\colon[],\mathrm{Ttot}\colon0,\mathrm{depends}([q,\omega],t),Q\colon$

Qdirect(DH), Qcap: Qbc(Q, DH, dist), **for** i **thru** length(Qcap) **do** $\Big($ I: append(I, [inerzia(i)]), R: $\begin{pmatrix} ((\mathrm{Qcap}_i)_1)_1 & ((\mathrm{Qcap}_i)_1)_2 & ((\mathrm{Qcap}_i)_1)_3 \\ ((\mathrm{Qcap}_i)_2)_1 & ((\mathrm{Qcap}_i)_2)_2 & ((\mathrm{Qcap}_i)_2)_3 \\ ((\mathrm{Qcap}_i)_3)_1 & ((\mathrm{Qcap}_i)_3)_2 & ((\mathrm{Qcap}_i)_3)_3 \end{pmatrix}$, dR: diff($R, t$), **for** j **thru** length(DH) **do** dR: subst$\left(\frac{1}{\mathrm{mtimes()}}\, q_j = \omega_j, \mathrm{dR}\right)$, Sw: dR $\cdot$ transpose(R), wtemp: $\begin{pmatrix} (\mathrm{Sw}_3)_2 \\ (\mathrm{Sw}_1)_3 \\ (\mathrm{Sw}_2)_1 \end{pmatrix}$, w: append(w, [trigreduce(expand(wtemp))]), Tatemp: $\frac{1}{2}$ transpose(wtemp) $\cdot R \cdot I_i \cdot$ transpose(R) $\cdot$ wtemp, Tatemp: trigsimp(trigreduce(trigexpand(Tatemp))), Ta: append(Ta, [Tatemp]), d: $\begin{pmatrix} ((\mathrm{Qcap}_i)_1)_4 \\ ((\mathrm{Qcap}_i)_2)_4 \\ ((Q_i)_3)_4 \end{pmatrix}$, dd: diff($d, t$), **for** j **thru** length(DH) **do** dd: subst$\left(\frac{1}{\mathrm{mtimes()}}\, q_j = \omega_j, \mathrm{dd}\right)$, Tbtemp: $\frac{\mathrm{massa}(i)}{2}$ trigsimp(trigreduce(trigexpand(transpose(dd) $\cdot$ dd))), Tb: append(Tb, [Tbtemp]), T: append(T, [trigreduce(Tatemp + Tbtemp)]) $\Big)$, **for** i **thru** length(DH) **do** Ttot: T_i + Ttot, B: zeromatrix(length(DH), length(DH)), **for** i **thru** length(DH) **do** $\Big($ $(B_i)_i$: coeff(collectterms(expand(2 Ttot), ω_i^2), ω_i, 2), **for** j **thru** length(DH) **do if** $i \neq j$ **then** $(B_i)_j$: ratsimp$\left(\mathrm{coeff}\left(\mathrm{coeff}\left(\mathrm{expand}\left(\mathrm{Ttot}\ \left(\frac{1}{2}\right)\right), \omega_i, 1\right), \omega_j, 1\right)\right)$, **if** length(DH) $\neq 2$ **then** res: [[Ta$_1$, Tb$_1$, T_1], [Ta$_2$, Tb$_2$, T_2], [Ta$_3$, Tb$_3$, T_3], [B]] **else** res: [[Ta$_1$, Tb$_1$, T_1], [Ta$_2$, Tb$_2$, T_2], [B]] $\Big)$

ep(DH):=funzione responsabile del calcolo dell'energia potenziale a cui è soggetto il robot.

$$\mathrm{DH} = \begin{pmatrix} \theta & d & \alpha & a \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix}$$

Calcolo cinematica diretta in accordo all'algoritmo di Denavit-Hartenberg:

$$Q_{0,n} = Q_{01} Q_{12} \ldots Q_{n-1,n} \quad \text{con}\, n \equiv \#\mathrm{DOF}$$

Applico traslazioni necessarie per portare il sistema il $\mathrm{SR}_{i-1,i}$ con l'origine coincidente con il baricentro del link:

$\hat{Q}_{01} = Q_{01} \begin{pmatrix} I & d \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} \hat{R} & \hat{d} \\ 0 & 1 \end{pmatrix}, \hat{Q}_{12} \ldots \hat{Q}_{n-1,n}$ in cui d sono le coordinate del baricentro del link $- \frac{L_i}{2}$

L'energia potenziale per il link i-esimo ha la seguente forma:

$$U_i = -M\, g^T d \quad \text{con}\ g = 9,8\, e_z \simeq 10\, e_z, d = \hat{d} := \text{coordinate nel baricentro}$$

Per l'intero robot:

$$U = \sum_{i=1}^{n} U_i = -\sum_{i=1}^{n} M_i\, g^T d_i$$

```
(%i10) ep(DH,dist):=block([Q,Qcap,g,U,Utemp,dTemp,prev,Utot],
                         Q:[], Qcap:[],U:[],Utot:zeromatrix(3,3),Utot:0,
                          depends([q,omega],t),
                         g:10*matrix([0],[0],[1]),
                         prev:ident(4),
                         Q:Qdirect(DH),
                          Qcap:Qbc(Q,DH,dist),

                            for i:1 thru length(Qcap) do(
                             print("Energia gravitazionale link",i),
                             dTemp:matrix([Qcap[i][1][4]],[Qcap[i][2][4]],
                                           [Qcap[i][3][4]]),

                             Utemp:M[i]*transpose(g).dTemp,
                             U:append(U,[Utemp]),
                             print("U[",i,"]=",Utemp)
                           ),
                            for i:1 thru length(U) do(
                                Utot:Utot+U[i]
                            ),
                            print("Energia gravitazionale totale=",
       ratsimp(trigsimp(trigreduce(trigexpand(Utot))))
                            );
```

(%o10) $\mathrm{ep}(\mathrm{DH}, \mathrm{dist}) := \textbf{block}\left([Q, \mathrm{Qcap}, g, U, \mathrm{Utemp}, \mathrm{dTemp}, \mathrm{prev}, \mathrm{Utot}], Q\colon [], \mathrm{Qcap}\colon [], U\colon [],\right.$ $\mathrm{Utot}\colon \mathrm{zeromatrix}(3,3), \mathrm{Utot}\colon 0, \mathrm{depends}([q,\omega],t), g\colon 10\begin{pmatrix}0\\0\\1\end{pmatrix}, \mathrm{prev}\colon \mathrm{ident}(4), Q\colon \mathrm{Qdirect}(\mathrm{DH}),$

$\mathrm{Qcap}\colon \mathrm{Qbc}(Q,\mathrm{DH},\mathrm{dist}), \textbf{for } i \textbf{ thru } \mathrm{length}(\mathrm{Qcap}) \textbf{ do } \left(\mathrm{print}(\text{Energia gravitazionale link }, i),\right.$ $\mathrm{dTemp}\colon \begin{pmatrix}((\mathrm{Qcap}_i)_1)_4\\((\mathrm{Qcap}_i)_2)_4\\((\mathrm{Qcap}_i)_3)_4\end{pmatrix}, \mathrm{Utemp}\colon M_i\,\mathrm{transpose}(g)\cdot \mathrm{dTemp}, U\colon \mathrm{append}(U,[\mathrm{Utemp}]), \mathrm{print}(\mathrm{U}[\ , i,]=$

$\left., \mathrm{Utemp})\right), \textbf{for } i \textbf{ thru } \mathrm{length}(U) \textbf{ do } \mathrm{Utot}\colon \mathrm{Utot}+U_i, \mathrm{print}(\text{Energia gravitazionale totale= },$

$\left.\mathrm{ratsimp}(\mathrm{trigsimp}(\mathrm{trigreduce}(\mathrm{trigexpand}(\mathrm{Utot}))))\right)$

```
(%i11) dinamica(DH,dist):=block([T],
                              T:ek(DH,dist),
                              for i:1 thru length(T)-1 do(
                                 print("Energia cinetica link", i),
                                  print("Energia cinetica rotazione Ta=",
       T[i][1]),
                                  print("Energia cinetica traslazione Tb=",
       T[i][2]),print("Energia cinetica totale T=",T[i][3])

                                ),print("Matrice inerzie generalizzate B=",
       T[length(T)]),ep(DH,dist));
```

(%o11) $\mathrm{dinamica}(\mathrm{DH}, \mathrm{dist}) := \textbf{block}\ ([T], T\colon \mathrm{ek}(\mathrm{DH}, \mathrm{dist}), \textbf{for } i \textbf{ thru } \mathrm{length}(T) - 1 \textbf{ do } (\mathrm{print}(\text{Energia cinetica link }, i), \mathrm{print}(\text{Energia cinetica rotazione Ta= }, (T_i)_1), \mathrm{print}(\text{Energia cinetica traslazione Tb= }, (T_i)_2), \mathrm{print}(\text{Energia cinetica totale T= }, (T_i)_3)), \mathrm{print}(\text{Matrice inerzie}$

generalizzate B= $, T_{\mathrm{length}(T)}), \mathrm{ep}(\mathrm{DH}, \mathrm{dist}))$

2DOF PLANARE

```
(%i12) DH:[[q[1],0,0,L[1]],[q[2],0,0,L[2]]];
```

(%o12) $[[q_1, 0, 0, L_1], [q_2, 0, 0, L_2]]$

```
(%i13) distance:[matrix([-L[1]/2],[0],[0]),matrix([-L[2]/2],[0],[0])];
```

$$\text{(\%o13)}\quad \left[\begin{pmatrix} -\frac{L_1}{2} \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} -\frac{L_2}{2} \\ 0 \\ 0 \end{pmatrix}\right]$$

```
(%i14) dinamica(DH,distance)
```

Energia cinetica link 1

Energia cinetica rotazione Ta= $\dfrac{\omega_1^2\,(\alpha_{zz})_1}{2}$

Energia cinetica traslazione Tb= $\dfrac{L_1^2\,M_1\,\omega_1^2}{8}$

Energia cinetica totale T= $\dfrac{\omega_1^2\,(\alpha_{zz})_1}{2} + \dfrac{L_1^2\,M_1\,\omega_1^2}{8}$

Energia cinetica link 2

Energia cinetica rotazione Ta= $\dfrac{(\omega_2^2 + 2\,\omega_1\,\omega_2 + \omega_1^2)\,(\alpha_{zz})_2}{2}$

Energia cinetica traslazione Tb=
$$\frac{M_2\,((4\,L_1\,\omega_1\,L_2\,\omega_2 + 4\,L_1\,\omega_1^2\,L_2)\cos(q_2) + L_2^2\,\omega_2^2 + 2\,\omega_1\,L_2^2\,\omega_2 + \omega_1^2\,L_2^2 + 4\,L_1^2\,\omega_1^2)}{8}$$

Energia cinetica totale T= $(4\,L_1\,\omega_1\,L_2\,M_2\,\omega_2\cos(q_2) + 4\,L_1\,\omega_1^2\,L_2\,M_2\cos(q_2) + L_2^2\,M_2\,\omega_2^2 + 2\,\omega_1\,L_2^2\,M_2\,\omega_2 + \omega_1^2\,L_2^2\,M_2 + 4\,L_1^2\,\omega_1^2\,M_2)/8 + \dfrac{\omega_2^2\,(\alpha_{zz})_2 + 2\,\omega_1\,\omega_2\,(\alpha_{zz})_2 + \omega_1^2\,(\alpha_{zz})_1}{2}$

Matrice inerzie generalizzate B= $\left[\left(L_1\,L_2\,M_2\cos(q_2) + (\alpha_{zz})_2 + \frac{L_2^2\,M_2}{4} + L_1^2\,M_2 + (\alpha_{zz})_1 + \frac{L_1^2\,M_1}{4}, \frac{2\,L_1\,L_2\,M_2\cos(q_2) + 4\,(\alpha_{zz})_2 + L_2^2\,M_2}{8}; \frac{2\,L_1\,L_2\,M_2\cos(q_2) + 4\,(\alpha_{zz})_2 + L_2^2\,M_2}{8}, (\alpha_{zz})_2 + \frac{L_2^2\,M_2}{4}\right)\right]$

Energia gravitazionale link 1

U[1]= 0

Energia gravitazionale link 2

U[2]= 0

Energia gravitazionale totale= 0

(%o14) 0

Robot Cartesiano

```
(%i15) DH:[[0,q[1],-%pi/2,0],[-%pi/2,q[2],-%pi/2,0],[0,q[3],0,0]];
```

$$\text{(\%o15)}\quad \left[\left[0, q_1, -\frac{\pi}{2}, 0\right], \left[-\frac{\pi}{2}, q_2, -\frac{\pi}{2}, 0\right], [0, q_3, 0, 0]\right]$$

```
(%i16) distance:[matrix([0],[-L[1]/2],[0]),matrix([0],[-L[2]/2],[0]),matrix([0],
       [0],[-L[3]/2])];
```

$$\text{(\%o16)}\quad \left[\begin{pmatrix} 0 \\ -\frac{L_1}{2} \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ -\frac{L_2}{2} \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ -\frac{L_3}{2} \end{pmatrix}\right]$$

(%i17) `dinamica(DH,distance);`

Energia cinetica link 1
Energia cinetica rotazione Ta= 0

Energia cinetica traslazione Tb= $\frac{M_1\,\omega_1^2}{2}$

Energia cinetica totale T= $\frac{M_1\,\omega_1^2}{2}$
Energia cinetica link 2

Energia cinetica rotazione Ta= 0
Energia cinetica traslazione Tb= $\frac{M_2\,(\omega_2^2+\omega_1^2)}{2}$

Energia cinetica totale T= $\frac{M_2\,\omega_2^2+\omega_1^2\,M_2}{2}$

Energia cinetica link 3
Energia cinetica rotazione Ta= 0

Energia cinetica traslazione Tb= $\frac{M_3\,(\omega_3^2+\omega_2^2+\omega_1^2)}{2}$
Energia cinetica totale T= $\frac{M_3\,\omega_3^2+\omega_2^2\,M_3+\omega_1^2\,M_3}{2}$

Matrice inerzie generalizzate B= $\left[\begin{pmatrix} M_3+M_2+M_1 & 0 & 0 \\ 0 & M_3+M_2 & 0 \\ 0 & 0 & M_3 \end{pmatrix}\right]$
Energia gravitazionale link 1

U[1]= $5\,M_1\,(2\,q_1+L_1)$

Energia gravitazionale link 2
U[2]= $10\,q_1\,M_2$

Energia gravitazionale link 3
U[3]= $10\,q_1\,M_3$

Energia gravitazionale totale= $10\,q_1\,M_3+10\,q_1\,M_2+10\,M_1\,q_1+5\,L_1\,M_1$
(%o17) $10\,q_1\,M_3+10\,q_1\,M_2+10\,M_1\,q_1+5\,L_1\,M_1$

Robot Cilindrico
(%i18) `DH:[[q[1],L[1],0,0],[0,q[2],-%pi/2,0],[0,q[3],0,0]];`

(%o18) $\left[[q_1, L_1, 0, 0], \left[0, q_2, -\frac{\pi}{2}, 0\right], [0, q_3, 0, 0]\right]$

(%i19) `distance:[matrix([0],[0],[-L[1]/2]),matrix([0],[-L[2]/2],[0]),matrix([0],[0],[-L[3]/2])];`

(%o19) $\left[\begin{pmatrix} 0 \\ 0 \\ -\frac{L_1}{2} \end{pmatrix}, \begin{pmatrix} 0 \\ -\frac{L_2}{2} \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ -\frac{L_3}{2} \end{pmatrix}\right]$

(%i20) `dinamica(DH,distance);`

Energia cinetica link 1
Energia cinetica rotazione Ta= $\frac{\omega_1^2\,(\alpha_{zz})_1}{2}$

Energia cinetica traslazione Tb= 0

Energia cinetica totale T= $\frac{\omega_1^2\,(\alpha_{zz})_1}{2}$

Energia cinetica link 2

Energia cinetica rotazione Ta= $\frac{\omega_1^2\,(\alpha_{yy})_2}{2}$

Energia cinetica traslazione Tb= $\frac{M_2\,\omega_2^2}{2}$

Energia cinetica totale T= $\frac{\omega_1^2\,(\alpha_{yy})_2}{2}+\frac{M_2\,\omega_2^2}{2}$

Energia cinetica link 3

Energia cinetica rotazione Ta= $\frac{\omega_1^2\,(\alpha_{yy})_3}{2}$

Energia cinetica traslazione Tb= $\frac{M_3\,(4\,\omega_1^2\,q_3^2-4\,\omega_1^2\,L_3\,q_3+4\,\omega_3^2+\omega_1^2\,L_3^2+4\,\omega_2^2)}{8}$

Energia cinetica totale T= $\frac{\omega_1^2\,(\alpha_{yy})_3}{2}+\frac{4\,\omega_1^2\,M_3\,q_3^2-4\,\omega_1^2\,L_3\,M_3\,q_3+4\,M_3\,\omega_3^2+\omega_1^2\,L_3^2\,M_3+4\,\omega_2^2\,M_3}{8}$

Matrice inerzie generalizzate B=

$$\left[\begin{pmatrix} (\alpha_{yy})_3+M_3\,q_3^2-L_3\,M_3\,q_3+\frac{L_3^2\,M_3}{4}+(\alpha_{yy})_2+(\alpha_{zz})_1 & 0 & 0\\ 0 & M_3+M_2 & 0\\ 0 & 0 & M_3\end{pmatrix}\right]$$

Energia gravitazionale link 1

U[1]= $5\,L_1\,M_1$

Energia gravitazionale link 2

U[2]= $5\,M_2\,(2\,q_2+L_2+2\,L_1)$

Energia gravitazionale link 3

U[3]= $10\,(q_2+L_1)\,M_3$

Energia gravitazionale totale= $(10\,q_2+10\,L_1)\,M_3+10\,M_2\,q_2+(5\,L_2+10\,L_1)\,M_2+5\,L_1\,M_1$

(%o20) $(10\,q_2+10\,L_1)\,M_3+10\,M_2\,q_2+(5\,L_2+10\,L_1)\,M_2+5\,L_1\,M_1$

Robot SCARA

```
(%i21) DH:[[q[1],L[1],0,D[1]],[q[2],0,0,0],[0,q[3],0,0]];
```

(%o21) $[[q_1,L_1,0,D_1],[q_2,0,0,0],[0,q_3,0,0]]$

```
(%i22) distance:[matrix([-D[1]/2],[0],[0]),matrix([-D[2]/2],[0],[0]),matrix([0],
       [0],[-L[3]/2])];
```

(%o22) $$\left[\begin{pmatrix}-\frac{D_1}{2}\\0\\0\end{pmatrix},\begin{pmatrix}-\frac{D_2}{2}\\0\\0\end{pmatrix},\begin{pmatrix}0\\0\\-\frac{L_3}{2}\end{pmatrix}\right]$$

```
(%i23) dinamica(DH,distance);
```

Energia cinetica link 1

Energia cinetica rotazione Ta= $\frac{\omega_1^2\,(\alpha_{zz})_1}{2}$

Energia cinetica traslazione Tb= $\frac{D_1^2\,M_1\,\omega_1^2}{8}$

Energia cinetica totale T= $\frac{\omega_1^2\,(\alpha_{zz})_1}{2}+\frac{D_1^2\,M_1\,\omega_1^2}{8}$

Energia cinetica link 2

Energia cinetica rotazione Ta= $\dfrac{(\omega_2^2+2\,\omega_1\,\omega_2+\omega_1^2)\,(\alpha_{zz})_2}{2}$

Energia cinetica traslazione Tb= $-M_2\,((4\,D_1\,\omega_1\,D_2\,\omega_2+4\,D_1\,\omega_1^2\,D_2)\cos(q_2)-D_2^2\,\omega_2^2-2\,\omega_1\,D_2^2\,\omega_2-\omega_1^2\,D_2^2-4\,D_1^2\,\omega_1^2)/8$

Energia cinetica totale T= $\dfrac{\omega_2^2\,(\alpha_{zz})_2+2\,\omega_1\,\omega_2\,(\alpha_{zz})_2+\omega_1^2\,(\alpha_{zz})_2}{2}-(4\,D_1\,\omega_1\,D_2\,M_2\,\omega_2\cos(q_2)+4\,D_1\,\omega_1^2\,D_2\,M_2\cos(q_2)-D_2^2\,M_2\,\omega_2^2-2\,\omega_1\,D_2^2\,M_2\,\omega_2-\omega_1^2\,D_2^2\,M_2-4\,D_1^2\,\omega_1^2\,M_2)/8$

Energia cinetica link 3

Energia cinetica rotazione Ta= $\dfrac{(\omega_2^2+2\,\omega_1\,\omega_2+\omega_1^2)\,(\alpha_{zz})_3}{2}$

Energia cinetica traslazione Tb= $\dfrac{M_3\,(\omega_3^2+D_1^2\,\omega_1^2)}{2}$

Energia cinetica totale T= $\dfrac{\omega_2^2\,(\alpha_{zz})_3+2\,\omega_1\,\omega_2\,(\alpha_{zz})_3+\omega_1^2\,(\alpha_{zz})_3}{2}+\dfrac{M_3\,\omega_3^2+D_1^2\,\omega_1^2\,M_3}{2}$

Matrice inerzie generalizzate B= $\left[\left(-D_1\,D_2\,M_2\cos(q_2)+(\alpha_{zz})_3+D_1^2\,M_3+(\alpha_{zz})_2+\frac{D_2^2\,M_2}{4}+D_1^2\,M_2+(\alpha_{zz})_1+\frac{D_1^2\,M_1}{4},-\frac{2\,D_1\,D_2\,M_2\cos(q_2)-4\,(\alpha_{zz})_3-4\,(\alpha_{zz})_2-D_2^2\,M_2}{8},0;-\frac{2\,D_1\,D_2\,M_2\cos(q_2)-4\,(\alpha_{zz})_3-4\,(\alpha_{zz})_2-D_2^2\,M_2}{8},(\alpha_{zz})_3+(\alpha_{zz})_2+\frac{D_2^2\,M_2}{4},0;0,0,M_3\right)\right]$

Energia gravitazionale link 1

U[1]= $10\,L_1\,M_1$

Energia gravitazionale link 2

U[2]= $10\,L_1\,M_2$

Energia gravitazionale link 3

U[3]= $5\,M_3\,(2\,q_3-L_3+2\,L_1)$

Energia gravitazionale totale= $10\,M_3\,q_3+(10\,L_1-5\,L_3)\,M_3+10\,L_1\,M_2+10\,L_1\,M_1$

(%o23) $10\,M_3\,q_3+(10\,L_1-5\,L_3)\,M_3+10\,L_1\,M_2+10\,L_1\,M_1$

Robot Sferico Tipo 1

```
(%i24) DH:[[q[1],L[1],%pi/2,0],[q[2],0,%pi/2,L[2]],[0,q[3],0,0]];
```

(%o24) $\left[\left[q_1,L_1,\frac{\pi}{2},0\right],\left[q_2,0,\frac{\pi}{2},L_2\right],[0,q_3,0,0]\right]$

```
(%i25) distance:[matrix([0],[-L[1]/2],[0]),matrix([-L[2]/2],[0],[0]),matrix([0],
       [0],[-L[3]/2])];
```

(%o25) $\left[\begin{pmatrix}0\\-\frac{L_1}{2}\\0\end{pmatrix},\begin{pmatrix}-\frac{L_2}{2}\\0\\0\end{pmatrix},\begin{pmatrix}0\\0\\-\frac{L_3}{2}\end{pmatrix}\right]$

```
(%i26) sfericoI:dinamica(DH,distance);
```

Energia cinetica link 1

Energia cinetica rotazione Ta= $\dfrac{\omega_1^2\,(\alpha_{yy})_1}{2}$

Energia cinetica traslazione Tb= 0

Energia cinetica totale T= $\dfrac{\omega_1^2\,(\alpha_{yy})_1}{2}$

Energia cinetica link 2

Energia cinetica rotazione Ta= $-(2\,\omega_1^2\,(\alpha_{xz})_2\sin(2\,q_2)+(\omega_1^2\,(\alpha_{xx})_2-\omega_1^2\,(\alpha_{zz})_2)\cos(2\,q_2)-4\,\omega_1\,\omega_2\,(\alpha_{xy})_2\sin(q_2)+4\,\omega_1\,\omega_2\,(\alpha_{yz})_2\cos(q_2)-\omega_1^2\,(\alpha_{zz})_2-2\,\omega_2^2\,(\alpha_{yy})_2-\omega_1^2\,(\alpha_{xx})_2)/4$

Energia cinetica traslazione Tb= $\frac{M_2\left(\left(3\,L_2^2\,\omega_2^2+\omega_1^2\,L_2^2\right)\cos\left(2\,q_2\right)+5\,L_2^2\,\omega_2^2+\omega_1^2\,L_2^2\right)}{16}$

Energia cinetica totale T= $\frac{3\,L_2^2\,M_2\,\omega_2^2\cos\left(2\,q_2\right)+\omega_1^2\,L_2^2\,M_2\cos\left(2\,q_2\right)+5\,L_2^2\,M_2\,\omega_2^2+\omega_1^2\,L_2^2\,M_2}{16}-$ $\left(2\,\omega_1^2\left(\alpha_{xz}\right)_2\sin\left(2\,q_2\right)-\omega_1^2\left(\alpha_{zz}\right)_2\cos\left(2\,q_2\right)+\omega_1^2\left(\alpha_{xx}\right)_2\cos\left(2\,q_2\right)-4\,\omega_1\,\omega_2\left(\alpha_{xy}\right)_2\sin\left(q_2\right)+4\,\omega_1\,\omega_2\left(\alpha_{yz}\right)_2\cos\left(q_2\right)-\omega_1^2\left(\alpha_{zz}\right)_2-2\,\omega_2^2\left(\alpha_{yy}\right)_2-\omega_1^2\left(\alpha_{xx}\right)_2\right)/4$

Energia cinetica link 3

Energia cinetica rotazione Ta= $-\left(2\,\omega_1^2\left(\alpha_{xz}\right)_3\sin\left(2\,q_2\right)+\left(\omega_1^2\left(\alpha_{xx}\right)_3-\omega_1^2\left(\alpha_{zz}\right)_3\right)\cos\left(2\,q_2\right)-4\,\omega_1\,\omega_2\left(\alpha_{xy}\right)_3\sin\left(q_2\right)+4\,\omega_1\,\omega_2\left(\alpha_{yz}\right)_3\cos\left(q_2\right)-\omega_1^2\left(\alpha_{zz}\right)_3-2\,\omega_2^2\left(\alpha_{yy}\right)_3-\omega_1^2\left(\alpha_{xx}\right)_3\right)/4$

Energia cinetica traslazione Tb= $M_3\left(\left(\left(\left(8\,\omega_2\,\omega_3-8\,L_2\,\omega_2^2+8\,\omega_1^2\,L_2\right)q_3-4\,\omega_2\,L_3\,\omega_3+\left(4\,L_2\,\omega_2^2-4\,\omega_1^2\,L_2\right)L_3\right)\sin\left(2\,q_2\right)+\left(\left(4\,\omega_2^2-4\,\omega_1^2\right)q_3^2+\left(4\,\omega_1^2-4\,\omega_2^2\right)L_3\,q_3-4\,\omega_3^2+8\,L_2\,\omega_2\,\omega_3+\left(\omega_2^2-\omega_1^2\right)L_3^2-4\,L_2^2\,\omega_2^2+4\,\omega_1^2\,L_2^2\right)\cos\left(2\,q_2\right)+\left(16\,L_2\,\omega_2^2-16\,\omega_2\,\omega_3\right)q_3\cos\left(q_2\right)\sin\left(q_2\right)+\left(-8\,\omega_2^2\,q_3^2+8\,\omega_3^2-16\,L_2\,\omega_2\,\omega_3+8\,L_2^2\,\omega_2^2\right)\cos\left(q_2\right)^2+\left(12\,\omega_2^2+4\,\omega_1^2\right)q_3^2+\left(-4\,\omega_2^2-4\,\omega_1^2\right)L_3\,q_3+4\,\omega_3^2-8\,L_2\,\omega_2\,\omega_3+\left(\omega_2^2+\omega_1^2\right)L_3^2+4\,L_2^2\,\omega_2^2+4\,\omega_1^2\,L_2^2\right)/16$

Energia cinetica totale T= $\left(8\,\omega_1^2\,L_2\,M_3\,q_3\sin\left(2\,q_2\right)-4\,\omega_2\,L_3\,M_3\,\omega_3\sin\left(2\,q_2\right)+4\,L_2\,\omega_2^2\,L_3\,M_3\sin\left(2\,q_2\right)-4\,\omega_1^2\,L_2\,L_3\,M_3\sin\left(2\,q_2\right)-4\,\omega_1^2\,M_3\,q_3^2\cos\left(2\,q_2\right)-4\,\omega_2^2\,L_3\,M_3\,q_3\cos\left(2\,q_2\right)+4\,\omega_1^2\,L_3\,M_3\,q_3\cos\left(2\,q_2\right)+\omega_2^2\,L_3^2\,M_3\cos\left(2\,q_2\right)-\omega_1^2\,L_3^2\,M_3\cos\left(2\,q_2\right)+4\,\omega_1^2\,L_2^2\,M_3\cos\left(2\,q_2\right)+8\,\omega_2^2\,M_3\,q_3^2+4\,\omega_1^2\,M_3\,q_3^2-4\,\omega_2^2\,L_3\,M_3\,q_3-4\,\omega_1^2\,L_3\,M_3\,q_3+8\,M_3\,\omega_3^2-16\,L_2\,\omega_2\,M_3\,\omega_3+\omega_2^2\,L_3^2\,M_3+\omega_1^2\,L_3^2\,M_3+8\,L_2^2\,\omega_2^2\,M_3+4\,\omega_1^2\,L_2^2\,M_3\right)/16-\left(2\,\omega_1^2\left(\alpha_{xz}\right)_3\sin\left(2\,q_2\right)-\omega_1^2\left(\alpha_{zz}\right)_3\cos\left(2\,q_2\right)+\omega_1^2\left(\alpha_{xx}\right)_3\cos\left(2\,q_2\right)-4\,\omega_1\,\omega_2\left(\alpha_{xy}\right)_3\sin\left(q_2\right)+4\,\omega_1\,\omega_2\left(\alpha_{yz}\right)_3\cos\left(q_2\right)-\omega_1^2\left(\alpha_{zz}\right)_3-2\,\omega_2^2\left(\alpha_{yy}\right)_3-\omega_1^2\left(\alpha_{xx}\right)_3\right)/4$

Matrice inerzie generalizzate B= $\left[\left(-\left(\alpha_{xz}\right)_3\sin\left(2\,q_2\right)+L_2\,M_3\,q_3\sin\left(2\,q_2\right)-\frac{L_2\,L_3\,M_3\sin\left(2\,q_2\right)}{2}-\left(\alpha_{xz}\right)_2\sin\left(2\,q_2\right)+\frac{\left(\alpha_{zz}\right)_3\cos\left(2\,q_2\right)}{2}-\frac{\left(\alpha_{xx}\right)_3\cos\left(2\,q_2\right)}{2}-\frac{M_3\,q_3^2\cos\left(2\,q_2\right)}{2}+\frac{L_3\,M_3\,q_3\cos\left(2\,q_2\right)}{2}-\frac{L_3^2\,M_3\cos\left(2\,q_2\right)}{8}+\frac{L_2^2\,M_3\cos\left(2\,q_2\right)}{2}+\frac{\left(\alpha_{zz}\right)_2\cos\left(2\,q_2\right)}{2}-\frac{\left(\alpha_{xx}\right)_2\cos\left(2\,q_2\right)}{2}+\frac{L_2^2\,M_2\cos\left(2\,q_2\right)}{8}+\frac{\left(\alpha_{zz}\right)_3}{2}+\frac{\left(\alpha_{xx}\right)_3}{2}+\frac{M_3\,q_3^2}{2}-\frac{L_3\,M_3\,q_3}{2}+\frac{L_3^2\,M_3}{8}+\frac{L_2^2\,M_3}{2}+\frac{\left(\alpha_{zz}\right)_2}{2}+\frac{\left(\alpha_{xx}\right)_2}{2}+\frac{L_2^2\,M_2}{8}+\left(\alpha_{yy}\right)_1,\frac{\left(\left(\alpha_{xy}\right)_3+\left(\alpha_{xy}\right)_2\right)\sin\left(q_2\right)+\left(-\left(\alpha_{yz}\right)_3-\left(\alpha_{yz}\right)_2\right)\cos\left(q_2\right)}{2},0;\frac{\left(\left(\alpha_{xy}\right)_3+\left(\alpha_{xy}\right)_2\right)\sin\left(q_2\right)+\left(-\left(\alpha_{yz}\right)_3-\left(\alpha_{yz}\right)_2\right)\cos\left(q_2\right)}{2},\frac{L_2\,L_3\,M_3\sin\left(2\,q_2\right)}{2}-\frac{L_3\,M_3\,q_3\cos\left(2\,q_2\right)}{2}+\frac{L_3^2\,M_3\cos\left(2\,q_2\right)}{8}+\frac{3\,L_2^2\,M_2\cos\left(2\,q_2\right)}{8}+\left(\alpha_{yy}\right)_3+M_3\,q_3^2-\frac{L_3\,M_3\,q_3}{2}+\frac{L_3^2\,M_3}{8}+L_2^2\,M_3+\left(\alpha_{yy}\right)_2+\frac{5\,L_2^2\,M_2}{8},-\frac{L_3\,M_3\sin\left(2\,q_2\right)+4\,L_2\,M_3}{8};0,-\frac{L_3\,M_3\sin\left(2\,q_2\right)+4\,L_2\,M_3}{8},M_3\right)\right]$

Energia gravitazionale link 1

U[1]= $5\,L_1\,M_1$

Energia gravitazionale link 2

U[2]= $5\,M_2\left(L_2\sin\left(q_2\right)+2\,L_1\right)$

Energia gravitazionale link 3

U[3]= $5\,M_3\left(2\,L_2\sin\left(q_2\right)+\left(L_3-2\,q_3\right)\cos\left(q_2\right)+2\,L_1\right)$

Energia gravitazionale totale= $\left(10\,L_2\,M_3+5\,L_2\,M_2\right)\sin\left(q_2\right)+\left(5\,L_3\,M_3-10\,M_3\,q_3\right)\cos\left(q_2\right)+10\,L_1\,M_3+10\,L_1\,M_2+5\,L_1\,M_1$

(%o26) $\left(10\,L_2\,M_3+5\,L_2\,M_2\right)\sin\left(q_2\right)+\left(5\,L_3\,M_3-10\,M_3\,q_3\right)\cos\left(q_2\right)+10\,L_1\,M_3+10\,L_1\,M_2+5\,L_1\,M_1$

Robot Sferico II

```
(%i27) DH:[[q[1],L[1],-%pi/2,0],[q[2],L[2],%pi/2,0],[0,q[3],0,0]];
```

(%o27) $\left[\left[q_1,L_1,-\frac{\pi}{2},0\right],\left[q_2,L_2,\frac{\pi}{2},0\right],\left[0,q_3,0,0\right]\right]$

```
(%i28) distance:[matrix([0],[-L[1]/2],[0]),matrix([0],[-L[2]/2],[0]),matrix([0],
[0],[-L[3]/2])];
```

(%o28) $$\left[\begin{pmatrix} 0 \\ -\frac{L_1}{2} \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ -\frac{L_2}{2} \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ -\frac{L_3}{2} \end{pmatrix}\right]$$

```
(%i29) dinamica(DH,distance);
```

Energia cinetica link 1

Energia cinetica rotazione Ta= $\frac{\omega_1^2 (\alpha_{yy})_1}{2}$

Energia cinetica traslazione Tb= 0

Energia cinetica totale T= $\frac{\omega_1^2 (\alpha_{yy})_1}{2}$

Energia cinetica link 2

Energia cinetica rotazione Ta= $-(2\,\omega_1^2\,(\alpha_{xz})_2 \sin(2\,q_2) + (\omega_1^2\,(\alpha_{xx})_2 - \omega_1^2\,(\alpha_{zz})_2)\cos(2\,q_2) + 4\,\omega_1\,\omega_2\,(\alpha_{xy})_2 \sin(q_2) - 4\,\omega_1\,\omega_2\,(\alpha_{yz})_2 \cos(q_2) - \omega_1^2\,(\alpha_{zz})_2 - 2\,\omega_2^2\,(\alpha_{yy})_2 - \omega_1^2\,(\alpha_{xx})_2)/4$

Energia cinetica traslazione Tb= $\frac{\omega_1^2\,L_2^2\,M_2}{8}$

Energia cinetica totale T= $\frac{\omega_1^2\,L_2^2\,M_2}{8} - (2\,\omega_1^2\,(\alpha_{xz})_2 \sin(2\,q_2) - \omega_1^2\,(\alpha_{zz})_2 \cos(2\,q_2) + \omega_1^2\,(\alpha_{xx})_2 \cos(2\,q_2) + 4\,\omega_1\,\omega_2\,(\alpha_{xy})_2 \sin(q_2) - 4\,\omega_1\,\omega_2\,(\alpha_{yz})_2 \cos(q_2) - \omega_1^2\,(\alpha_{zz})_2 - 2\,\omega_2^2\,(\alpha_{yy})_2 - \omega_1^2\,(\alpha_{xx})_2)/4$

Energia cinetica link 3

Energia cinetica rotazione Ta= $-(2\,\omega_1^2\,(\alpha_{xz})_3 \sin(2\,q_2) + (\omega_1^2\,(\alpha_{xx})_3 - \omega_1^2\,(\alpha_{zz})_3)\cos(2\,q_2) + 4\,\omega_1\,\omega_2\,(\alpha_{xy})_3 \sin(q_2) - 4\,\omega_1\,\omega_2\,(\alpha_{yz})_3 \cos(q_2) - \omega_1^2\,(\alpha_{zz})_3 - 2\,\omega_2^2\,(\alpha_{yy})_3 - \omega_1^2\,(\alpha_{xx})_3)/4$

Energia cinetica traslazione Tb= $M_3\,((8\,\omega_2\,\omega_3\,q_3 - 4\,\omega_2\,L_3\,\omega_3)\sin(2\,q_2) + ((4\,\omega_2^2 - 4\,\omega_1^2)\,q_3^2 + (4\,\omega_1^2 - 4\,\omega_2^2)\,L_3\,q_3 - 4\,\omega_3^2 + (\omega_2^2 - \omega_1^2)\,L_3^2)\cos(2\,q_2) + (-16\,\omega_2\,\omega_3\,q_3 \cos(q_2) - 16\,\omega_1\,L_2\,\omega_3)\sin(q_2) + (8\,\omega_3^2 - 8\,\omega_2^2\,q_3^2)\cos(q_2)^2 + (8\,\omega_1\,L_2\,\omega_2\,L_3 - 16\,\omega_1\,L_2\,\omega_2\,q_3)\cos(q_2) + (12\,\omega_2^2 + 4\,\omega_1^2)\,q_3^2 + (-4\,\omega_2^2 - 4\,\omega_1^2)\,L_3\,q_3 + 4\,\omega_3^2 + (\omega_2^2 + \omega_1^2)\,L_3^2 + 8\,\omega_1^2\,L_2^2)/16$

Energia cinetica totale T= $(-4\,\omega_2\,L_3\,M_3\,\omega_3 \sin(2\,q_2) - 4\,\omega_1^2\,M_3\,q_3^2 \cos(2\,q_2) - 4\,\omega_2^2\,L_3\,M_3\,q_3 \cos(2\,q_2) + 4\,\omega_1^2\,L_3\,M_3\,q_3 \cos(2\,q_2) + \omega_2^2\,L_3^2\,M_3 \cos(2\,q_2) - \omega_1^2\,L_3^2\,M_3 \cos(2\,q_2) - 16\,\omega_1\,L_2\,M_3\,\omega_3 \sin(q_2) - 16\,\omega_1\,L_2\,\omega_2\,M_3\,q_3 \cos(q_2) + 8\,\omega_1\,L_2\,\omega_2\,L_3\,M_3 \cos(q_2) + 8\,\omega_2^2\,M_3\,q_3^2 + 4\,\omega_1^2\,M_3\,q_3^2 - 4\,\omega_2^2\,L_3\,M_3\,q_3 - 4\,\omega_1^2\,L_3\,M_3\,q_3 + 8\,M_3\,\omega_3^2 + \omega_2^2\,L_3^2\,M_3 + \omega_1^2\,L_3^2\,M_3 + 8\,\omega_1^2\,L_2^2\,M_3)/16 - (2\,\omega_1^2\,(\alpha_{xz})_3 \sin(2\,q_2) - \omega_1^2\,(\alpha_{zz})_3 \cos(2\,q_2) + \omega_1^2\,(\alpha_{xx})_3 \cos(2\,q_2) + 4\,\omega_1\,\omega_2\,(\alpha_{xy})_3 \sin(q_2) - 4\,\omega_1\,\omega_2\,(\alpha_{yz})_3 \cos(q_2) - \omega_1^2\,(\alpha_{zz})_3 - 2\,\omega_2^2\,(\alpha_{yy})_3 - \omega_1^2\,(\alpha_{xx})_3)/4$

Matrice inerzie generalizzate B= $\Big[\Big(-(\alpha_{xz})_3 \sin(2\,q_2) - (\alpha_{xz})_2 \sin(2\,q_2) + \frac{(\alpha_{zz})_3 \cos(2\,q_2)}{2} - \frac{(\alpha_{xx})_3 \cos(2\,q_2)}{2} - \frac{M_3\,q_3^2 \cos(2\,q_2)}{2} + \frac{L_3\,M_3\,q_3 \cos(2\,q_2)}{2} - \frac{L_3^2\,M_3 \cos(2\,q_2)}{8} + \frac{(\alpha_{zz})_2 \cos(2\,q_2)}{2} - \frac{(\alpha_{xx})_2 \cos(2\,q_2)}{2} + \frac{(\alpha_{zz})_3}{2} + \frac{(\alpha_{xx})_3}{2} + \frac{M_3\,q_3^2}{2} - \frac{L_3\,M_3\,q_3}{2} + \frac{L_3^2\,M_3}{8} + L_2^2\,M_3 + \frac{(\alpha_{zz})_2}{2} + \frac{(\alpha_{xx})_2}{2} + \frac{L_2^2\,M_2}{4} + (\alpha_{yy})_1, -\frac{(2\,(\alpha_{xy})_3 + 2\,(\alpha_{xy})_2)\sin(q_2) + (-2\,(\alpha_{yz})_3 + 2\,L_2\,M_3\,q_3 - L_2\,L_3\,M_3 - 2\,(\alpha_{yz})_2)\cos(q_2)}{4}, -\frac{L_2\,M_3 \sin(q_2)}{2}; -\frac{(2\,(\alpha_{xy})_3 + 2\,(\alpha_{xy})_2)\sin(q_2) + (-2\,(\alpha_{yz})_3 + 2\,L_2\,M_3\,q_3 - L_2\,L_3\,M_3 - 2\,(\alpha_{yz})_2)\cos(q_2)}{4}, -\frac{L_3\,M_3\,q_3 \cos(2\,q_2)}{2} + \frac{L_3^2\,M_3 \cos(2\,q_2)}{8} + (\alpha_{yy})_3 + M_3\,q_3^2 - \frac{L_3\,M_3\,q_3}{2} + \frac{L_3^2\,M_3}{8} + (\alpha_{yy})_2, -\frac{L_3\,M_3 \sin(2\,q_2)}{8}; -\frac{L_2\,M_3 \sin(q_2)}{2}, -\frac{L_3\,M_3 \sin(2\,q_2)}{8}, M_3\Big)\Big]$

Energia gravitazionale link 1

U[1]= $15\,L_1\,M_1$

Energia gravitazionale link 2

U[2]= $10\, L_1\, M_2$
Energia gravitazionale link 3

U[3]= $5\, M_3\, ((2\, q_3 - L_3) \cos(q_2) + 2\, L_1)$
Energia gravitazionale totale= $(10\, M_3\, q_3 - 5\, L_3\, M_3) \cos(q_2) + 10\, L_1\, M_3 + 10\, L_1\, M_2 + 15\, L_1\, M_1$

(%o29) $(10\, M_3\, q_3 - 5\, L_3\, M_3) \cos(q_2) + 10\, L_1\, M_3 + 10\, L_1\, M_2 + 15\, L_1\, M_1$

Robot Antropomorfo

```
(%i30) DH:[[q[1],L[1],%pi/2,0],[q[2],0,0,D[3]],[q[3],0,0,D[3]]];
```

(%o30) $\left[\left[q_1, L_1, \frac{\pi}{2}, 0\right], [q_2, 0, 0, D_3], [q_3, 0, 0, D_3]\right]$

```
(%i31) distance:[matrix([0],[-L[1]/2],[0]),matrix([-L[2]/2],[0],[0]),matrix([-
L[3]/2],[0],[0])];
```

(%o31) $\left[\begin{pmatrix} 0 \\ -\frac{L_1}{2} \\ 0 \end{pmatrix}, \begin{pmatrix} -\frac{L_2}{2} \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} -\frac{L_3}{2} \\ 0 \\ 0 \end{pmatrix}\right]$

```
(%i32) dinamica(DH,distance);
```

Energia cinetica link 1
Energia cinetica rotazione Ta= $\frac{\omega_1^2\, (\alpha_{yy})_1}{2}$

Energia cinetica traslazione Tb= 0
Energia cinetica totale T= $\frac{\omega_1^2\, (\alpha_{yy})_1}{2}$

Energia cinetica link 2
Energia cinetica rotazione Ta= $(2\, \omega_1^2\, (\alpha_{xy})_2 \sin(2\, q_2) + (\omega_1^2\, (\alpha_{yy})_2 - \omega_1^2\, (\alpha_{xx})_2) \cos(2\, q_2) + 4\, \omega_1\, \omega_2\, (\alpha_{xz})_2 \sin(q_2) + 4\, \omega_1\, \omega_2\, (\alpha_{yz})_2 \cos(q_2) + 2\, \omega_2^2\, (\alpha_{zz})_2 + \omega_1^2\, (\alpha_{yy})_2 + \omega_1^2\, (\alpha_{xx})_2)/4$
Energia cinetica traslazione Tb= $M_2\, ((4\, \omega_1^2\, D_3^2 + (4\, L_2\, \omega_2^2 - 4\, \omega_1^2\, L_2)\, D_3 - L_2^2\, \omega_2^2 + \omega_1^2\, L_2^2) \cos(2\, q_2) + (8\, \omega_2^2 + 4\, \omega_1^2)\, D_3^2 + (-4\, L_2\, \omega_2^2 - 4\, \omega_1^2\, L_2)\, D_3 + L_2^2\, \omega_2^2 + \omega_1^2\, L_2^2)/16$
Energia cinetica totale T= $(2\, \omega_1^2\, (\alpha_{xy})_2 \sin(2\, q_2) + \omega_1^2\, (\alpha_{yy})_2 \cos(2\, q_2) - \omega_1^2\, (\alpha_{xx})_2 \cos(2\, q_2) + 4\, \omega_1\, \omega_2\, (\alpha_{xz})_2 \sin(q_2) + 4\, \omega_1\, \omega_2\, (\alpha_{yz})_2 \cos(q_2) + 2\, \omega_2^2\, (\alpha_{zz})_2 + \omega_1^2\, (\alpha_{yy})_2 + \omega_1^2\, (\alpha_{xx})_2)/4 + (4\, \omega_1^2\, M_2\, D_3^2 \cos(2\, q_2) + 4\, L_2\, M_2\, \omega_2^2\, D_3 \cos(2\, q_2) - 4\, \omega_1^2\, L_2\, M_2\, D_3 \cos(2\, q_2) - L_2^2\, M_2\, \omega_2^2 \cos(2\, q_2) + \omega_1^2\, L_2^2\, M_2 \cos(2\, q_2) + 8\, M_2\, \omega_2^2\, D_3^2 + 4\, \omega_1^2\, M_2\, D_3^2 - 4\, L_2\, M_2\, \omega_2^2\, D_3 - 4\, \omega_1^2\, L_2\, M_2\, D_3 + L_2^2\, M_2\, \omega_2^2 + \omega_1^2\, L_2^2\, M_2)/16$
Energia cinetica link 3

Energia cinetica rotazione Ta= $(2\, \omega_1^2\, (\alpha_{xy})_3 \sin(2\, q_3 + 2\, q_2) + (\omega_1^2\, (\alpha_{yy})_3 - \omega_1^2\, (\alpha_{xx})_3) \cos(2\, q_3 + 2\, q_2) + (4\, \omega_1\, \omega_3 + 4\, \omega_1\, \omega_2)\, (\alpha_{xz})_3 \sin(q_3 + q_2) + (4\, \omega_1\, \omega_3 + 4\, \omega_1\, \omega_2)\, (\alpha_{yz})_3 \cos(q_3 + q_2) + (2\, \omega_3^2 + 4\, \omega_2\, \omega_3 + 2\, \omega_2^2)\, (\alpha_{zz})_3 + \omega_1^2\, (\alpha_{yy})_3 + \omega_1^2\, (\alpha_{xx})_3)/4$
Energia cinetica traslazione Tb= $-M_3\, (((L_3^2 - 4\, D_3\, L_3 + 4\, D_3^2)\, \omega_3^2 + (2\, \omega_2\, L_3^2 - 8\, \omega_2\, D_3\, L_3 + 8\, \omega_2\, D_3^2)\, \omega_3 + (\omega_2^2 - \omega_1^2)\, L_3^2 + (4\, \omega_1^2 - 4\, \omega_2^2)\, D_3\, L_3 + (4\, \omega_2^2 - 4\, \omega_1^2)\, D_3^2) \cos(2\, q_3 + 2\, q_2) + ((8\, \omega_2\, D_3^2 - 4\, \omega_2\, D_3\, L_3)\, \omega_3 + (4\, \omega_1^2 - 4\, \omega_2^2)\, D_3\, L_3 + (8\, \omega_2^2 - 8\, \omega_1^2)\, D_3^2) \cos(q_3 + 2\, q_2) + (-8\, D_3^2\, \omega_3^2 - 16\, \omega_2\, D_3^2\, \omega_3 - 8\, \omega_2^2\, D_3^2) \cos(q_3 + q_2)^2 + (-16\, \omega_2\, D_3^2\, \omega_3 - 16\, \omega_2^2\, D_3^2) \cos(q_2) \cos(q_3 + q_2) + ((4\, \omega_2\, D_3\, L_3 - 8\, \omega_2\, D_3^2)\, \omega_3 + (4\, \omega_2^2 + 4\, \omega_1^2)\, D_3\, L_3 + (-8\, \omega_2^2 - 8\, \omega_1^2)\, D_3^2) \cos(q_3) + (4\, \omega_2^2 - 4\, \omega_1^2)\, D_3^2 \cos(2\, q_2) + 8\, \omega_2^2\, D_3^2 \sin(q_2)^2 + (-L_3^2 + 4\, D_3\, L_3 - 4\, D_3^2)\, \omega_3^2 + (-2\, \omega_2\, L_3^2 + 8\, \omega_2\, D_3\, L_3 - 8\, \omega_2\, D_3^2)\, \omega_3 + (-\omega_2^2 - \omega_1^2)\, L_3^2 + (4\, \omega_2^2 + 4\, \omega_1^2)\, D_3\, L_3 + (-16\, \omega_2^2 - 8\, \omega_1^2)\, D_3^2)/16$

Energia cinetica totale T= $(2\,\omega_1^2\,(\alpha_{xy})_3 \sin(2\,q_3+2\,q_2) + \omega_1^2\,(\alpha_{yy})_3 \cos(2\,q_3+2\,q_2) - \omega_1^2\,(\alpha_{xx})_3 \cos(2\,q_3+2\,q_2) + 4\,\omega_1\,\omega_3\,(\alpha_{xz})_3 \sin(q_3+q_2) + 4\,\omega_1\,\omega_2\,(\alpha_{xz})_3 \sin(q_3+q_2) + 4\,\omega_1\,\omega_3\,(\alpha_{yz})_3 \cos(q_3+q_2) + 4\,\omega_1\,\omega_2\,(\alpha_{yz})_3 \cos(q_3+q_2) + 2\,\omega_3^2\,(\alpha_{zz})_3 + 4\,\omega_2\,\omega_3\,(\alpha_{zz})_3 + 2\,\omega_2^2\,(\alpha_{zz})_3 + \omega_1^2\,(\alpha_{yy})_3 + \omega_1^2\,(\alpha_{xx})_3)/4 - (L_3^2\,M_3\,\omega_3^2 \cos(2\,q_3+2\,q_2) - 4\,D_3\,L_3\,M_3\,\omega_3^2 \cos(2\,q_3+2\,q_2) + 2\,\omega_2\,L_3^2\,M_3\,\omega_3 \cos(2\,q_3+2\,q_2) - 8\,\omega_2\,D_3\,L_3\,M_3\,\omega_3 \cos(2\,q_3+2\,q_2) + \omega_2^2\,L_3^2\,M_3 \cos(2\,q_3+2\,q_2) - \omega_1^2\,L_3^2\,M_3 \cos(2\,q_3+2\,q_2) - 4\,\omega_2^2\,D_3\,L_3\,M_3 \cos(2\,q_3+2\,q_2) + 4\,\omega_1^2\,D_3\,L_3\,M_3 \cos(2\,q_3+2\,q_2) - 4\,\omega_1^2\,D_3^2\,M_3 \cos(2\,q_3+2\,q_2) - 4\,\omega_2\,D_3\,L_3\,M_3\,\omega_3 \cos(q_3+2\,q_2) - 4\,\omega_2^2\,D_3\,L_3\,M_3 \cos(q_3+2\,q_2) + 4\,\omega_1^2\,D_3\,L_3\,M_3 \cos(q_3+2\,q_2) - 8\,\omega_1^2\,D_3^2\,M_3 \cos(q_3+2\,q_2) + 4\,\omega_2\,D_3\,L_3\,M_3\,\omega_3 \cos(q_3) - 16\,\omega_2\,D_3^2\,M_3\,\omega_3 \cos(q_3) + 4\,\omega_2^2\,D_3\,L_3\,M_3 \cos(q_3) + 4\,\omega_1^2\,D_3\,L_3\,M_3 \cos(q_3) - 16\,\omega_2^2\,D_3^2\,M_3 \cos(q_3) - 8\,\omega_1^2\,D_3^2\,M_3 \cos(q_3) - 4\,\omega_1^2\,D_3^2\,M_3 \cos(2\,q_2) - L_3^2\,M_3\,\omega_3^2 + 4\,D_3\,L_3\,M_3\,\omega_3^2 - 8\,D_3^2\,M_3\,\omega_3^2 - 2\,\omega_2\,L_3^2\,M_3\,\omega_3 + 8\,\omega_2\,D_3\,L_3\,M_3\,\omega_3 - 16\,\omega_2\,D_3^2\,M_3\,\omega_3 - \omega_2^2\,L_3^2\,M_3 - \omega_1^2\,L_3^2\,M_3 + 4\,\omega_2^2\,D_3\,L_3\,M_3 + 4\,\omega_1^2\,D_3\,L_3\,M_3 - 16\,\omega_2^2\,D_3^2\,M_3 - 8\,\omega_1^2\,D_3^2\,M_3)/16$

Matrice inerzie generalizzate B= $\Big[\Big((\alpha_{xy})_3 \sin(2\,q_3+2\,q_2) + \frac{(\alpha_{yy})_3 \cos(2\,q_3+2\,q_2)}{2} - \frac{(\alpha_{xx})_3 \cos(2\,q_3+2\,q_2)}{2} + \frac{L_3^2\,M_3 \cos(2\,q_3+2\,q_2)}{8} - \frac{D_3\,L_3\,M_3 \cos(2\,q_3+2\,q_2)}{2} + \frac{D_3^2\,M_3 \cos(2\,q_3+2\,q_2)}{2} - \frac{D_3\,L_3\,M_3 \cos(q_3+2\,q_2)}{2} + D_3^2\,M_3 \cos(q_3+2\,q_2) - \frac{D_3\,L_3\,M_3 \cos(q_3)}{2} + D_3^2\,M_3 \cos(q_3) + (\alpha_{xy})_2 \sin(2\,q_2) + \frac{D_3^2\,M_3 \cos(2\,q_2)}{2} + \frac{M_2\,D_3^2 \cos(2\,q_2)}{2} - \frac{L_2\,M_2\,D_3 \cos(2\,q_2)}{2} + \frac{(\alpha_{yy})_2 \cos(2\,q_2)}{2} - \frac{(\alpha_{xx})_2 \cos(2\,q_2)}{2} + \frac{L_2^2\,M_2 \cos(2\,q_2)}{8} + \frac{(\alpha_{yy})_3}{2} + \frac{(\alpha_{xx})_3}{2} + \frac{L_3^2\,M_3}{8} - \frac{D_3\,L_3\,M_3}{2} + D_3^2\,M_3 + \frac{M_2\,D_3^2}{2} - \frac{L_2\,M_2\,D_3}{2} + \frac{(\alpha_{yy})_2}{2} + \frac{(\alpha_{xx})_2}{2} + \frac{L_2^2\,M_2}{8} + (\alpha_{yy})_1, \frac{(\alpha_{xz})_3 \sin(q_3+q_2) + (\alpha_{yz})_3 \cos(q_3+q_2) + (\alpha_{xz})_2 \sin(q_2) + (\alpha_{yz})_2 \cos(q_2)}{2}, \frac{(\alpha_{xz})_3 \sin(q_3+q_2) + (\alpha_{yz})_3 \cos(q_3+q_2)}{2}; \frac{(\alpha_{xz})_3 \sin(q_3+q_2) + (\alpha_{yz})_3 \cos(q_3+q_2) + (\alpha_{xz})_2 \sin(q_2) + (\alpha_{yz})_2 \cos(q_2)}{2}, -\frac{L_3^2\,M_3 \cos(2\,q_3+2\,q_2)}{8} + \frac{D_3\,L_3\,M_3 \cos(2\,q_3+2\,q_2)}{2} + \frac{D_3\,L_3\,M_3 \cos(q_3+2\,q_2)}{2} - \frac{D_3\,L_3\,M_3 \cos(q_3)}{2} + 2\,D_3^2\,M_3 \cos(q_3) + \frac{L_2\,M_2\,D_3 \cos(2\,q_2)}{2} - \frac{L_2^2\,M_2 \cos(2\,q_2)}{8} + (\alpha_{zz})_3 + \frac{L_3^2\,M_3}{8} - \frac{D_3\,L_3\,M_3}{2} + 2\,D_3^2\,M_3 + M_2\,D_3^2 - \frac{L_2\,M_2\,D_3}{2} + (\alpha_{zz})_2 + \frac{L_2^2\,M_2}{8}, -((L_3^2 - 4\,D_3\,L_3)\,M_3 \cos(2\,q_3+2\,q_2) - 2\,D_3\,L_3\,M_3 \cos(q_3+2\,q_2) + (2\,D_3\,L_3 - 8\,D_3^2)\,M_3 \cos(q_3) - 8\,(\alpha_{zz})_3 + (-L_3^2 + 4\,D_3\,L_3 - 8\,D_3^2)\,M_3)/16; \frac{(\alpha_{xz})_3 \sin(q_3+q_2) + (\alpha_{yz})_3 \cos(q_3+q_2)}{2}, -((L_3^2 - 4\,D_3\,L_3)\,M_3 \cos(2\,q_3+2\,q_2) - 2\,D_3\,L_3\,M_3 \cos(q_3+2\,q_2) + (2\,D_3\,L_3 - 8\,D_3^2)\,M_3 \cos(q_3) - 8\,(\alpha_{zz})_3 + (-L_3^2 + 4\,D_3\,L_3 - 8\,D_3^2)\,M_3)/16, -\frac{L_3^2\,M_3 \cos(2\,q_3+2\,q_2)}{8} + \frac{D_3\,L_3\,M_3 \cos(2\,q_3+2\,q_2)}{2} + (\alpha_{zz})_3 + \frac{L_3^2\,M_3}{8} - \frac{D_3\,L_3\,M_3}{2} + D_3^2\,M_3\Big)\Big]$

Energia gravitazionale link 1

U[1]= $5\,L_1\,M_1$

Energia gravitazionale link 2

U[2]= $5\,M_2\,((2\,D_3 - L_2) \sin(q_2) + 2\,L_1)$

Energia gravitazionale link 3

U[3]= $-5\,M_3\,((L_3 - 2\,D_3) \sin(q_3+q_2) - 2\,D_3 \sin(q_2) - 2\,L_1)$

Energia gravitazionale totale= $(10\,D_3 - 5\,L_3)\,M_3 \sin(q_3+q_2) + (10\,D_3\,M_3 + 10\,M_2\,D_3 - 5\,L_2\,M_2) \sin(q_2) + 10\,L_1\,M_3 + 10\,L_1\,M_2 + 5\,L_1\,M_1$

(%o32) $(10\,D_3 - 5\,L_3)\,M_3 \sin(q_3+q_2) + (10\,D_3\,M_3 + 10\,M_2\,D_3 - 5\,L_2\,M_2) \sin(q_2) + 10\,L_1\,M_3 + 10\,L_1\,M_2 + 5\,L_1\,M_1$

(%i33)

Matrice di Inerzia

Definiamo P:= coordinate di un ponto generico solidale al copro ($\hat{P}$):

$$\mathbb{I} = \rho \int_V S^T(P) S(P) \partial V \quad \text{con } V = \int_V \partial V \, , \, \rho = \frac{M}{V}$$

Ipotizziamo che il punto $P = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$ abbia x,y,z sui piani di simmetria del corpo. Allora:

$$S^T S = \begin{pmatrix} y^2 + z^2 & -xy & -xz \\ -xy & x^2 + z^2 & -yz \\ -xz & -yz & x^2 + y^2 \end{pmatrix}$$

Se la densità del corpo ρ=cost:

$$\mathbb{I} = \frac{M}{V} \iiint \begin{pmatrix} y^2 + z^2 & -xy & -xz \\ -xy & x^2 + z^2 & -yz \\ -xz & -yz & x^2 + y^2 \end{pmatrix} \partial x \partial y \partial z$$

In alternativa:

$$\mathbb{I} = \iiint \rho(x, y, z) \begin{pmatrix} y^2 + z^2 & -xy & -xz \\ -xy & x^2 + z^2 & -yz \\ -xz & -yz & x^2 + y^2 \end{pmatrix} \partial x \partial y \partial z$$

Parallelepipedo di lati A,B,C senza tappi

$$x \in \left[-\frac{A}{2}, \frac{A}{2} \right], y \in \left[-\frac{B}{2}, \frac{B}{2} \right], z \in \left[-\frac{C}{2}, \frac{C}{2} \right]$$

$$\mathbb{I} = \frac{M}{V} \iiint \begin{pmatrix} y^2 + z^2 & -xy & -xz \\ -xy & x^2 + z^2 & -yz \\ -xz & -yz & x^2 + y^2 \end{pmatrix} \partial x \partial y \partial z$$

$$x^2 \longrightarrow \int_{-\frac{C}{2}}^{\frac{C}{2}} \int_{-\frac{B}{2}}^{\frac{B}{2}} \int_{-\frac{A}{2}}^{\frac{A}{2}} x^2 \partial x \partial y \partial z = \int_{-\frac{C}{2}}^{\frac{C}{2}} \int_{-\frac{B}{2}}^{\frac{B}{2}} \left[\frac{x^3}{3} \right]_{-\frac{A}{2}}^{\frac{A}{2}} = \int_{-\frac{C}{2}}^{\frac{C}{2}} \int_{-\frac{B}{2}}^{\frac{B}{2}} \frac{A^3}{12} \partial y \partial z = \frac{A^3}{12} \mathrm{BC}$$

$$\rho x^2 \Rightarrow \frac{M}{\mathrm{ABC}} \frac{A^3}{12} \mathrm{BC} = \frac{\mathrm{MA}^2}{12}$$

$$y^2 \longrightarrow \frac{\mathrm{MB}^2}{12}; z^2 \longrightarrow \frac{\mathrm{MC}^2}{12}$$

$$\mathrm{xy} \longrightarrow \int_{-\frac{C}{2}}^{\frac{C}{2}} \int_{-\frac{B}{2}}^{\frac{B}{2}} \int_{-\frac{A}{2}}^{\frac{A}{2}} \mathrm{xy} \partial x \partial y \partial z = \int_{-\frac{C}{2}}^{\frac{C}{2}} \int_{-\frac{B}{2}}^{\frac{B}{2}} \left[\frac{x^2}{2} y \right]_{-\frac{A}{2}}^{\frac{A}{2}} \partial y \partial z = 0$$

$$\mathrm{xz} \longrightarrow 0; \mathrm{yz} \longrightarrow 0$$

Quindi:

$$\mathbb{I} = \frac{M}{12}\begin{pmatrix} B^2+C^2 & 0 & 0 \\ 0 & A^2+C^2 & 0 \\ 0 & 0 & A^2+B^2 \end{pmatrix}$$

```
Maxima 5.44.0 http://maxima.sourceforge.net
using Lisp SBCL 2.0.0
Distributed under the GNU Public License. See the file COPYING.
Dedicated to the memory of William Schelter.
The function bug_report() provides bug reporting information.
(%i1) SSt:matrix([y^2+z^2,-y*x,-x*z],[-y*z,x^2+z^2,-y*z],[-x*z,-y*z,x^2+y^2]);
```

(%o1) $$\begin{pmatrix} z^2+y^2 & -x\,y & -x\,z \\ -y\,z & z^2+x^2 & -y\,z \\ -x\,z & -y\,z & y^2+x^2 \end{pmatrix}$$

```
(%i1) paral(A,B,C,M):=block([x2,y2,z2,xy,xz,yz,rho,I],
          rho:M/(A*B*C),
          x2:integrate(integrate(integrate(x^2,x,-A/2,A/2),y,-B/2,B/2),z,-C/
      2,C/2),
          y2:integrate(integrate(integrate(y^2,x,-A/2,A/2),y,-B/2,B/2),z,-C/
      2,C/2),
          z2:integrate(integrate(integrate(z^2,x,-A/2,A/2),y,-B/2,B/2),z,-C/
      2,C/2),
          xy:integrate(integrate(integrate(x*y,x,-A/2,A/2),y,-B/2,B/2),z,-C/
      2,C/2),
          xz:integrate(integrate(integrate(x*z,x,-A/2,A/2),y,-B/2,B/2),z,-C/
      2,C/2),
          yz:integrate(integrate(integrate(y*z,x,-A/2,A/2),y,-B/2,B/2),z,-C/
      2,C/2),
          I:zeromatrix(3,3),
          I[1][1]:y2+z2,I[1][2]:-xy,I[1][3]:-xz,
          I[2][1]:I[1][2],I[2][2]:x2+z2,I[2][3]:-yz,
          I[3][1]:I[1][3],I[3][2]:I[2][3],I[3][3]:x2+y2,
          I:ratsimp(rho*I),
          print("Matrice di Ineriza"),
          print(I)



                    );
```

(%o2) $\operatorname{paral}(A,B,C,M):=\mathbf{block}\left([x2,y2,z2,\mathrm{xy},\mathrm{xz},\mathrm{yz},\rho,I],\rho:\frac{M}{A\,B\,C},x2:\right.$
$\operatorname{integrate}\left(\operatorname{integrate}\left(\operatorname{integrate}\left(x^2,x,\frac{-A}{2},\frac{A}{2}\right),y,\frac{-B}{2},\frac{B}{2}\right),z,\frac{-C}{2},\frac{C}{2}\right),y2:$
$\operatorname{integrate}\left(\operatorname{integrate}\left(\operatorname{integrate}\left(y^2,x,\frac{-A}{2},\frac{A}{2}\right),y,\frac{-B}{2},\frac{B}{2}\right),z,\frac{-C}{2},\frac{C}{2}\right),z2:$
$\operatorname{integrate}\left(\operatorname{integrate}\left(\operatorname{integrate}\left(z^2,x,\frac{-A}{2},\frac{A}{2}\right),y,\frac{-B}{2},\frac{B}{2}\right),z,\frac{-C}{2},\frac{C}{2}\right),\mathrm{xy}:$
$\operatorname{integrate}\left(\operatorname{integrate}\left(\operatorname{integrate}\left(x\,y,x,\frac{-A}{2},\frac{A}{2}\right),y,\frac{-B}{2},\frac{B}{2}\right),z,\frac{-C}{2},\frac{C}{2}\right),\mathrm{xz}:$

$\text{integrate}\left(\text{integrate}\left(\text{integrate}\left(x\,z, x, \frac{-A}{2}, \frac{A}{2}\right), y, \frac{-B}{2}, \frac{B}{2}\right), z, \frac{-C}{2}, \frac{C}{2}\right)$, yz: $\text{integrate}\left(\text{integrate}\left(\text{integrate}\left(y\,z, x, \frac{-A}{2}, \frac{A}{2}\right), y, \frac{-B}{2}, \frac{B}{2}\right), z, \frac{-C}{2}, \frac{C}{2}\right)$, I: zeromatrix(3, 3), $(I_1)_1$: $y2 + z2$, $(I_1)_2$: $-\text{xy}$, $(I_1)_3$: $-\text{xz}$, $(I_2)_1$: $(I_1)_2$, $(I_2)_2$: $x2 + z2$, $(I_2)_3$: $-\text{yz}$, $(I_3)_1$: $(I_1)_3$, $(I_3)_2$: $(I_2)_3$, $(I_3)_3$: $x2 + y2$, I: ratsimp($\rho\,I$), print(Matrice di Ineriza), print(I))

```
(%i3) paral(A,B,C,M);
```

Matrice di Ineriza

$$\begin{pmatrix} \frac{(C^2+B^2)\,M}{12} & 0 & 0 \\ 0 & \frac{(C^2+A^2)\,M}{12} & 0 \\ 0 & 0 & \frac{(B^2+A^2)\,M}{12} \end{pmatrix}$$

(%o3) $$\begin{pmatrix} \frac{(C^2+B^2)\,M}{12} & 0 & 0 \\ 0 & \frac{(C^2+A^2)\,M}{12} & 0 \\ 0 & 0 & \frac{(B^2+A^2)\,M}{12} \end{pmatrix}$$

(%i4)

Parallelepipedo di lato A,B,C con tappi A',B',C'

```
(%i1) paral(A,B,C,M):=block([x2,y2,z2,xy,xz,yz,rho,I],
                rho:M/(A*B*C),
                x2:integrate(integrate(integrate(x^2,x,-A/2,A/2),y,-B/2,B/2),z,-C/
      2,C/2),
                y2:integrate(integrate(integrate(y^2,x,-A/2,A/2),y,-B/2,B/2),z,-C/
      2,C/2),
                z2:integrate(integrate(integrate(z^2,x,-A/2,A/2),y,-B/2,B/2),z,-C/
      2,C/2),
                xy:integrate(integrate(integrate(x*y,x,-A/2,A/2),y,-B/2,B/2),z,-C/
      2,C/2),
                xz:integrate(integrate(integrate(x*z,x,-A/2,A/2),y,-B/2,B/2),z,-C/
      2,C/2),
                yz:integrate(integrate(integrate(y*z,x,-A/2,A/2),y,-B/2,B/2),z,-C/
      2,C/2),
                I:zeromatrix(3,3),
                I[1][1]:y2+z2,I[1][2]:-xy,I[1][3]:-xz,
                I[2][1]:I[1][2],I[2][2]:x2+z2,I[2][3]:-yz,
                I[3][1]:I[1][3],I[3][2]:I[2][3],I[3][3]:x2+y2,
                I:ratsimp(rho*I),
                print("Matrice di Ineriza"),
                print(I)



                     );
```

(%o1) paral(A, B, C, M) := **block** $\left([x2, y2, z2, \text{xy}, \text{xz}, \text{yz}, \rho, I], \rho: \frac{M}{A\,B\,C}, x2:\right.$ $\text{integrate}\left(\text{integrate}\left(\text{integrate}\left(x^2, x, \frac{-A}{2}, \frac{A}{2}\right), y, \frac{-B}{2}, \frac{B}{2}\right), z, \frac{-C}{2}, \frac{C}{2}\right)$, $y2$:

$\text{integrate}\left(\text{integrate}\left(\text{integrate}\left(y^2, x, \frac{-A}{2}, \frac{A}{2}\right), y, \frac{-B}{2}, \frac{B}{2}\right), z, \frac{-C}{2}, \frac{C}{2}\right), z2\text{:}$
$\text{integrate}\left(\text{integrate}\left(\text{integrate}\left(z^2, x, \frac{-A}{2}, \frac{A}{2}\right), y, \frac{-B}{2}, \frac{B}{2}\right), z, \frac{-C}{2}, \frac{C}{2}\right), \text{xy:}$
$\text{integrate}\left(\text{integrate}\left(\text{integrate}\left(x\,y, x, \frac{-A}{2}, \frac{A}{2}\right), y, \frac{-B}{2}, \frac{B}{2}\right), z, \frac{-C}{2}, \frac{C}{2}\right), \text{xz:}$
$\text{integrate}\left(\text{integrate}\left(\text{integrate}\left(x\,z, x, \frac{-A}{2}, \frac{A}{2}\right), y, \frac{-B}{2}, \frac{B}{2}\right), z, \frac{-C}{2}, \frac{C}{2}\right), \text{yz:}$
$\text{integrate}\left(\text{integrate}\left(\text{integrate}\left(y\,z, x, \frac{-A}{2}, \frac{A}{2}\right), y, \frac{-B}{2}, \frac{B}{2}\right), z, \frac{-C}{2}, \frac{C}{2}\right), I\text{: zeromatrix}(3,3), (I_1)_1\text{:}$
$y2 + z2, (I_1)_2\text{: } -\text{xy}, (I_1)_3\text{: } -\text{xz}, (I_2)_1\text{: } (I_1)_2, (I_2)_2\text{: } x2 + z2, (I_2)_3\text{: } -\text{yz}, (I_3)_1\text{: } (I_1)_3, (I_3)_2\text{: } (I_2)_3, (I_3)_3\text{:}$
$x2 + y2, I\text{: ratsimp}(\rho\, I), \text{print}(\text{Matrice di Ineriza }), \text{print}(I)\Big)$

```
(%i2) paralTappi(A,B,C,A1,B1,C1,M):=block(
                V:A*B*C-A1*B1*C1,
                rho:M/V,
                x2:integrate(integrate(integrate(x^2,x,-A/2,A/2),y,-B/2,B/2),z,-C/
      2,C/2),
                y2:integrate(integrate(integrate(y^2,x,-A/2,A/2),y,-B/2,B/2),z,-C/
      2,C/2),
                z2:integrate(integrate(integrate(z^2,x,-A/2,A/2),y,-B/2,B/2),z,-C/
      2,C/2),
                xy:integrate(integrate(integrate(x*y,x,-A/2,A/2),y,-B/2,B/2),z,-C/
      2,C/2),
                xz:integrate(integrate(integrate(x*z,x,-A/2,A/2),y,-B/2,B/2),z,-C/
      2,C/2),
                yz:integrate(integrate(integrate(y*z,x,-A/2,A/2),y,-B/2,B/2),z,-C/
      2,C/2),
                x21:integrate(integrate(integrate(x^2,x,-A1/2,A1/2),y,-B1/2,B1/2),
      z,-C1/2,C1/2),
                y21:integrate(integrate(integrate(y^2,x,-A1/2,A1/2),y,-B1/2,B1/2),
      z,-C1/2,C1/2),
                z21:integrate(integrate(integrate(z^2,x,-A1/2,A1/2),y,-B1/2,B1/2),
      z,-C1/2,C1/2),
                xy1:integrate(integrate(integrate(x*y,x,-A1/2,A1/2),y,-B1/2,B1/2),
      z,-C1/2,C1/2),
                xz1:integrate(integrate(integrate(x*z,x,-A1/2,A1/2),y,-B1/2,B1/2),
      z,-C1/2,C1/2),
                yz1:integrate(integrate(integrate(y*z,x,-A1/2,A1/2),y,-B1/2,B1/2),
      z,-C1/2,C1/2),
                x2:x2-x21,y2:y2-y21,z2:z2-z21,
                xy:xy-xy1,xz:xz-xz1,yz:yz-yz1,
                I:zeromatrix(3,3),
                I[1][1]:y2+z2,I[1][2]:-xy,I[1][3]:-xz,
                I[2][1]:I[1][2],I[2][2]:x2+z2,I[2][3]:-yz,
                I[3][1]:I[1][3],I[3][2]:I[2][3],I[3][3]:x2+y2,
                I:ratsimp(expand(rho*I)),
                print("Matrice di Ineriza"),
                print(I));
```

(%o2) $\text{paralTappi}(A, B, C, A1, B1, C1, M) := \textbf{block}\Big(V\text{: } A\,B\,C - A1\,B1\,C1, \rho\text{: } \frac{M}{V}, x2\text{:}$

$\text{integrate}\left(\text{integrate}\left(\text{integrate}\left(x^2, x, \frac{-A}{2}, \frac{A}{2}\right), y, \frac{-B}{2}, \frac{B}{2}\right), z, \frac{-C}{2}, \frac{C}{2}\right)$, $y2$:
$\text{integrate}\left(\text{integrate}\left(\text{integrate}\left(y^2, x, \frac{-A}{2}, \frac{A}{2}\right), y, \frac{-B}{2}, \frac{B}{2}\right), z, \frac{-C}{2}, \frac{C}{2}\right)$, $z2$:
$\text{integrate}\left(\text{integrate}\left(\text{integrate}\left(z^2, x, \frac{-A}{2}, \frac{A}{2}\right), y, \frac{-B}{2}, \frac{B}{2}\right), z, \frac{-C}{2}, \frac{C}{2}\right)$, xy:
$\text{integrate}\left(\text{integrate}\left(\text{integrate}\left(x\,y, x, \frac{-A}{2}, \frac{A}{2}\right), y, \frac{-B}{2}, \frac{B}{2}\right), z, \frac{-C}{2}, \frac{C}{2}\right)$, xz:
$\text{integrate}\left(\text{integrate}\left(\text{integrate}\left(x\,z, x, \frac{-A}{2}, \frac{A}{2}\right), y, \frac{-B}{2}, \frac{B}{2}\right), z, \frac{-C}{2}, \frac{C}{2}\right)$, yz:
$\text{integrate}\left(\text{integrate}\left(\text{integrate}\left(y\,z, x, \frac{-A}{2}, \frac{A}{2}\right), y, \frac{-B}{2}, \frac{B}{2}\right), z, \frac{-C}{2}, \frac{C}{2}\right)$, $x21$:
$\text{integrate}\left(\text{integrate}\left(\text{integrate}\left(x^2, x, \frac{-A1}{2}, \frac{A1}{2}\right), y, \frac{-B1}{2}, \frac{B1}{2}\right), z, \frac{-C1}{2}, \frac{C1}{2}\right)$, $y21$:
$\text{integrate}\left(\text{integrate}\left(\text{integrate}\left(y^2, x, \frac{-A1}{2}, \frac{A1}{2}\right), y, \frac{-B1}{2}, \frac{B1}{2}\right), z, \frac{-C1}{2}, \frac{C1}{2}\right)$, $z21$:
$\text{integrate}\left(\text{integrate}\left(\text{integrate}\left(z^2, x, \frac{-A1}{2}, \frac{A1}{2}\right), y, \frac{-B1}{2}, \frac{B1}{2}\right), z, \frac{-C1}{2}, \frac{C1}{2}\right)$, xy1:
$\text{integrate}\left(\text{integrate}\left(\text{integrate}\left(x\,y, x, \frac{-A1}{2}, \frac{A1}{2}\right), y, \frac{-B1}{2}, \frac{B1}{2}\right), z, \frac{-C1}{2}, \frac{C1}{2}\right)$, xz1:
$\text{integrate}\left(\text{integrate}\left(\text{integrate}\left(x\,z, x, \frac{-A1}{2}, \frac{A1}{2}\right), y, \frac{-B1}{2}, \frac{B1}{2}\right), z, \frac{-C1}{2}, \frac{C1}{2}\right)$, yz1:
$\text{integrate}\left(\text{integrate}\left(\text{integrate}\left(y\,z, x, \frac{-A1}{2}, \frac{A1}{2}\right), y, \frac{-B1}{2}, \frac{B1}{2}\right), z, \frac{-C1}{2}, \frac{C1}{2}\right)$, $x2$: $x2 - x21$, $y2$: $y2 - y21$, $z2$: $z2 - z21$, xy: xy − xy1, xz: xz − xz1, yz: yz − yz1, I: zeromatrix(3, 3), $(I_1)_1$: $y2 + z2$, $(I_1)_2$: −xy, $(I_1)_3$: −xz, $(I_2)_1$: $(I_1)_2$, $(I_2)_2$: $x2 + z2$, $(I_2)_3$: −yz, $(I_3)_1$: $(I_1)_3$, $(I_3)_2$: $(I_2)_3$, $(I_3)_3$: $x2 + y2$, I: ratsimp(expand($\rho\, I$)), print(Matrice di Ineriza), print(I))

Parallelepipedo con tappi

```
(%i3) paralTappi(A,B,C,A1,B1,C1,M);
```

Matrice di Ineriza

$$\left(\frac{(A1\,B1\,C1^3 + A1\,B1^3\,C1 - A\,B\,C^3 - A\,B^3\,C)\,M}{12\,A1\,B1\,C1 - 12\,A\,B\,C}, 0, 0; 0, \frac{(A1\,B1\,C1^3 + A1^3\,B1\,C1 - A\,B\,C^3 - A^3\,B\,C)\,M}{12\,A1\,B1\,C1 - 12\,A\,B\,C}, 0; 0, 0, \frac{((A1\,B1^3 + A1^3\,B1)\,C1 + (-A\,B^3 - A^3\,B)\,C)\,M}{12\,A1\,B1\,C1 - 12\,A\,B\,C}\right)$$

(%o3) $$\left(\frac{(A1\,B1\,C1^3 + A1\,B1^3\,C1 - A\,B\,C^3 - A\,B^3\,C)\,M}{12\,A1\,B1\,C1 - 12\,A\,B\,C}, 0, 0; 0, \frac{(A1\,B1\,C1^3 + A1^3\,B1\,C1 - A\,B\,C^3 - A^3\,B\,C)\,M}{12\,A1\,B1\,C1 - 12\,A\,B\,C}, 0; 0, 0, \frac{((A1\,B1^3 + A1^3\,B1)\,C1 + (-A\,B^3 - A^3\,B)\,C)\,M}{12\,A1\,B1\,C1 - 12\,A\,B\,C}\right)$$

Parallelepipedo senza tappi

```
(%i4) paralTappi(A,B,C,A,B1,C1,M);
```

Matrice di Ineriza

$$\left(\frac{(B1\,C1^3+B1^3\,C1-B\,C^3-B^3\,C)\,M}{12\,B1\,C1-12\,B\,C},0,0;0,\frac{(B1\,C1^3+A^2\,B1\,C1-B\,C^3-A^2\,B\,C)\,M}{12\,B1\,C1-12\,B\,C},0;0,0,\frac{((B1^3+A^2\,B1)\,C1+(-B^3-A^2\,B)\,C)\,M}{12\,B1\,C1-12\,B\,C}\right)$$

(%o4) $$\left(\frac{(B1\,C1^3+B1^3\,C1-B\,C^3-B^3\,C)\,M}{12\,B1\,C1-12\,B\,C},0,0;0,\frac{(B1\,C1^3+A^2\,B1\,C1-B\,C^3-A^2\,B\,C)\,M}{12\,B1\,C1-12\,B\,C},0;0,0,\frac{((B1^3+A^2\,B1)\,C1+(-B^3-A^2\,B)\,C)\,M}{12\,B1\,C1-12\,B\,C}\right)$$

```
(%i5)
```

Cilindro di raggio R ed altezza H

$$\mathbb{I}=\frac{M}{V}\iiint\begin{pmatrix} y^2+z^2 & -xy & -xz \\ -xy & x^2+z^2 & -yz \\ -xz & -yz & x^2+y^2 \end{pmatrix}\partial x\partial y\partial z$$

$$V=A_b\,H=\pi R^2H$$

$$\rho=\frac{M}{\pi R^2H}$$

Al fine di svolgere l'integrale, una delle scelte è quella di ricorrere alle coordinate cilindriche. In particolare:

$$\begin{cases} x=\rho\cos(\theta) \\ y=\rho\sin(\theta) \\ z=z \end{cases}\longrightarrow\theta\in[0,2\pi],\,\rho\in[0,R],\,z\in\left[-\frac{H}{2},\frac{H}{2}\right]$$

```
(%i23) cilindro(R,H):=block([x,y,z,a,b,c,d],
                    x:rho*cos(theta),
                    y:rho*sin(theta),
                    z:z,
                     V:%pi*R^2*H,
          d:M/V,
          x2:integrate(integrate(integrate(rho*x^2,theta,0,2*%pi),rho,0,R),
z,-H/2,H/2),
          y2:integrate(integrate(integrate(rho*y^2,theta,0,2*%pi),rho,0,R),
z,-H/2,H/2),
          z2:integrate(integrate(integrate(rho*z^2,theta,0,2*%pi),rho,0,R),
z,-H/2,H/2),
          xy:integrate(integrate(integrate(rho*x*y,theta,0,2*%pi),rho,0,R),
z,-H/2,H/2),
          xz:integrate(integrate(integrate(rho*x*z,theta,0,2*%pi),rho,0,R),
z,-H/2,H/2),
          yz:integrate(integrate(integrate(rho*y*z,theta,0,2*%pi),rho,0,R),
z,-H/2,H/2),

          I:zeromatrix(3,3),
          I[1][1]:y2+z2,I[1][2]:-xy,I[1][3]:-xz,
          I[2][1]:I[1][2],I[2][2]:x2+z2,I[2][3]:-yz,
          I[3][1]:I[1][3],I[3][2]:I[2][3],I[3][3]:x2+y2,
          I:ratsimp((d*I)),
          print("Matrice di Ineriza"),
          print(I))
```

(%o23) $\mathrm{cilindro}(R,H) := \textbf{block}\Bigg([x,y,z,a,b,c,d], x\colon \rho\cos(\vartheta), y\colon \rho\sin(\vartheta), z\colon z, V\colon \pi\, R^2\, H, d\colon \frac{M}{V},$
$x2\colon \mathrm{integrate}\Big(\mathrm{integrate}(\mathrm{integrate}(\rho\, x^2, \vartheta, 0, 2\,\pi), \rho, 0, R), z, \frac{-H}{2}, \frac{H}{2}\Big), y2\colon$
$\mathrm{integrate}\Big(\mathrm{integrate}(\mathrm{integrate}(\rho\, y^2, \vartheta, 0, 2\,\pi), \rho, 0, R), z, \frac{-H}{2}, \frac{H}{2}\Big), z2\colon$
$\mathrm{integrate}\Big(\mathrm{integrate}(\mathrm{integrate}(\rho\, z^2, \vartheta, 0, 2\,\pi), \rho, 0, R), z, \frac{-H}{2}, \frac{H}{2}\Big), \mathrm{xy}\colon$
$\mathrm{integrate}\Big(\mathrm{integrate}(\mathrm{integrate}(\rho\, x\, y, \vartheta, 0, 2\,\pi), \rho, 0, R), z, \frac{-H}{2}, \frac{H}{2}\Big), \mathrm{xz}\colon$
$\mathrm{integrate}\Big(\mathrm{integrate}(\mathrm{integrate}(\rho\, x\, z, \vartheta, 0, 2\,\pi), \rho, 0, R), z, \frac{-H}{2}, \frac{H}{2}\Big), \mathrm{yz}\colon$
$\mathrm{integrate}\Big(\mathrm{integrate}(\mathrm{integrate}(\rho\, y\, z, \vartheta, 0, 2\,\pi), \rho, 0, R), z, \frac{-H}{2}, \frac{H}{2}\Big), I\colon \mathrm{zeromatrix}(3,3), (I_1)_1\colon y2 +$
$z2, (I_1)_2\colon -\mathrm{xy}, (I_1)_3\colon -\mathrm{xz}, (I_2)_1\colon (I_1)_2, (I_2)_2\colon x2 + z2, (I_2)_3\colon -\mathrm{yz}, (I_3)_1\colon (I_1)_3, (I_3)_2\colon (I_2)_3, (I_3)_3\colon x2 + y2,$
$I\colon \mathrm{ratsimp}(d\, I), \mathrm{print}(\text{Matrice di Ineriza }), \mathrm{print}(I)\Bigg)$

```
(%i24) cilindro(R,H);
```

Matrice di Ineriza

$$\begin{pmatrix} \frac{3\,M\,R^2+H^2\,M}{12} & 0 & 0 \\ 0 & \frac{3\,M\,R^2+H^2\,M}{12} & 0 \\ 0 & 0 & \frac{M\,R^2}{2} \end{pmatrix}$$

(%o24) $$\begin{pmatrix} \frac{3\,M\,R^2+H^2\,M}{12} & 0 & 0 \\ 0 & \frac{3\,M\,R^2+H^2\,M}{12} & 0 \\ 0 & 0 & \frac{M\,R^2}{2} \end{pmatrix}$$

(%i25)

Cilindro di raggio R, altezza H cavo con cilindro R',H senza tappi

```
(%i40) cilindro(R,H,R1):=block([x,y,z,a,b,c,d],
                            x:rho*cos(theta),
                            y:rho*sin(theta),
                            z:z,
                             V:%pi*(R)^2*H-%pi*(R1)^2*H,
                 d:M/V,
                 x2:integrate(integrate(integrate(rho*x^2,theta,0,2*%pi),rho,R1,
       R),z,0,H),
                 y2:integrate(integrate(integrate(rho*y^2,theta,0,2*%pi),rho,R1,
       R),z,0,H),
                 z2:integrate(integrate(integrate(rho*z^2,theta,0,2*%pi),rho,R1,
       R),z,0,H),
                 xy:integrate(integrate(integrate(rho*x*y,theta,0,2*%pi),rho,R1,
       R),z,0,H),
                 xz:integrate(integrate(integrate(rho*x*z,theta,0,2*%pi),rho,R1,
       R),z,0,H),
                 yz:integrate(integrate(integrate(rho*y*z,theta,0,2*%pi),rho,R1,
       R),z,0,H),

                 I:zeromatrix(3,3),
                 I[1][1]:y2+z2,I[1][2]:-xy,I[1][3]:-xz,
                 I[2][1]:I[1][2],I[2][2]:x2+z2,I[2][3]:-yz,
                 I[3][1]:I[1][3],I[3][2]:I[2][3],I[3][3]:x2+y2,
                 I:ratsimp(expand(d*I)),
                 print("Matrice di Ineriza"),
                 print(I))
```

(%o40) $\mathrm{cilindro}(R,H,R1):=\mathbf{block}\Big([x,y,z,a,b,c,d],x\colon\rho\cos(\vartheta),y\colon\rho\sin(\vartheta),z\colon z,V\colon\pi\,R^2\,H-\pi\,R1^2\,H,d\colon\frac{M}{V},x2\colon\mathrm{integrate}(\mathrm{integrate}(\mathrm{integrate}(\rho\,x^2,\vartheta,0,2\,\pi),\rho,R1,R),z,0,H),y2\colon\mathrm{integrate}(\mathrm{integrate}(\mathrm{integrate}(\rho\,y^2,\vartheta,0,2\,\pi),\rho,R1,R),z,0,H),z2\colon\mathrm{integrate}(\mathrm{integrate}(\mathrm{integrate}(\rho\,z^2,\vartheta,0,2\,\pi),\rho,R1,R),z,0,H),\mathrm{xy}\colon\mathrm{integrate}(\mathrm{integrate}(\mathrm{integrate}(\rho\,x\,y,\vartheta,0,2\,\pi),\rho,R1,R),z,0,H),\mathrm{xz}\colon\mathrm{integrate}(\mathrm{integrate}(\mathrm{integrate}(\rho\,x\,z,\vartheta,0,2\,\pi),\rho,R1,R),z,0,H),\mathrm{yz}\colon\mathrm{integrate}(\mathrm{integrate}(\mathrm{integrate}(\rho\,y\,z,\vartheta,0,2\,\pi),\rho,R1,R),z,0,H),I\colon\mathrm{zeromatrix}(3,3),(I_1)_1\colon y2+z2,(I_1)_2\colon-\mathrm{xy},(I_1)_3\colon-\mathrm{xz},(I_2)_1\colon(I_1)_2,(I_2)_2\colon x2+z2,(I_2)_3\colon-\mathrm{yz},(I_3)_1\colon(I_1)_3,(I_3)_2\colon(I_2)_3,(I_3)_3\colon x2+y2,I\colon\mathrm{ratsimp}(\mathrm{expand}(d\,I)),\mathrm{print}(\text{Matrice di Ineriza }),\mathrm{print}(I)\Big)$

```
(%i41) cilindro(R,H,R[1]);
```

Matrice di Ineriza

$$\begin{pmatrix} \frac{3\,M\,R^2+(4\,H^2+3\,R_1^2)\,M}{12} & 0 & 0 \\ 0 & \frac{3\,M\,R^2+(4\,H^2+3\,R_1^2)\,M}{12} & 0 \\ 0 & 0 & \frac{M\,R^2+R_1^2\,M}{2} \end{pmatrix}$$

(%o41) $$\begin{pmatrix} \frac{3\,M\,R^2+(4\,H^2+3\,R_1^2)\,M}{12} & 0 & 0 \\ 0 & \frac{3\,M\,R^2+(4\,H^2+3\,R_1^2)\,M}{12} & 0 \\ 0 & 0 & \frac{M\,R^2+R_1^2\,M}{2} \end{pmatrix}$$

Cilindro di raggio R, altezza H cavo con cilindro R',H' con tappi

(%i52)
```
cilindro(R,H):=block([x,y,z,a,b,c,d],
                     x:rho*cos(theta),
                     y:rho*sin(theta),
                     z:z,

          x2:integrate(integrate(integrate(rho*x^2,theta,0,2*%pi),rho,0,R),
z,-H/2,H/2),
          y2:integrate(integrate(integrate(rho*y^2,theta,0,2*%pi),rho,0,R),
z,-H/2,H/2),
          z2:integrate(integrate(integrate(rho*z^2,theta,0,2*%pi),rho,0,R),
z,-H/2,H/2),
          xy:integrate(integrate(integrate(rho*x*y,theta,0,2*%pi),rho,0,R),
z,-H/2,H/2),
          xz:integrate(integrate(integrate(rho*x*z,theta,0,2*%pi),rho,0,R),
z,-H/2,H/2),
          yz:integrate(integrate(integrate(rho*y*z,theta,0,2*%pi),rho,0,R),
z,-H/2,H/2),

          I:zeromatrix(3,3),
          I[1][1]:y2+z2,I[1][2]:-xy,I[1][3]:-xz,
          I[2][1]:I[1][2],I[2][2]:x2+z2,I[2][3]:-yz,
          I[3][1]:I[1][3],I[3][2]:I[2][3],I[3][3]:x2+y2,
          I:ratsimp(I)
          )
```

(%o52) $\mathrm{cilindro}(R,H) := \mathbf{block}\Big([x,y,z,a,b,c,d], x\colon \rho\cos(\vartheta), y\colon \rho\sin(\vartheta), z\colon z, x2\colon$
$\mathrm{integrate}\left(\mathrm{integrate}(\mathrm{integrate}(\rho\, x^2,\vartheta,0,2\,\pi),\rho,0,R),z,\frac{-H}{2},\frac{H}{2}\right), y2\colon$
$\mathrm{integrate}\left(\mathrm{integrate}(\mathrm{integrate}(\rho\, y^2,\vartheta,0,2\,\pi),\rho,0,R),z,\frac{-H}{2},\frac{H}{2}\right), z2\colon$
$\mathrm{integrate}\left(\mathrm{integrate}(\mathrm{integrate}(\rho\, z^2,\vartheta,0,2\,\pi),\rho,0,R),z,\frac{-H}{2},\frac{H}{2}\right), \mathrm{xy}\colon$
$\mathrm{integrate}\left(\mathrm{integrate}(\mathrm{integrate}(\rho\, x\, y,\vartheta,0,2\,\pi),\rho,0,R),z,\frac{-H}{2},\frac{H}{2}\right), \mathrm{xz}\colon$
$\mathrm{integrate}\left(\mathrm{integrate}(\mathrm{integrate}(\rho\, x\, z,\vartheta,0,2\,\pi),\rho,0,R),z,\frac{-H}{2},\frac{H}{2}\right), \mathrm{yz}\colon$
$\mathrm{integrate}\left(\mathrm{integrate}(\mathrm{integrate}(\rho\, y\, z,\vartheta,0,2\,\pi),\rho,0,R),z,\frac{-H}{2},\frac{H}{2}\right), I\colon \mathrm{zeromatrix}(3,3), (I_1)_1\colon y2+$

$z2$, $(I_1)_2$: $-$xy, $(I_1)_3$: $-$xz, $(I_2)_1$: $(I_1)_2$, $(I_2)_2$: $x2+z2$, $(I_2)_3$: $-$yz, $(I_3)_1$: $(I_1)_3$, $(I_3)_2$: $(I_2)_3$, $(I_3)_3$: $x2+y2$, I: ratsimp$(I)\Big)$

```
(%i53) cilindroSV(R,H,H1,R1):=block([x,y,z,a,b,c,d],
                        V:%pi*R^2*H-%pi*R1^2*H1,
                      d:M/V,
                      I:ratsimp(expand(cilindro(R,H))),
                      Icavo:ratsimp(expand(cilindro(R1,H1))),
                      print("Matrice di Ineriza"),
                      print(ratsimp(d*I)))
```

(%o53) cilindroSV$(R, H, H1, R1)$:= **block** $\Big($ $[x, y, z, a, b, c, d]$, V: $\pi\, R^2\, H - \pi\, R1^2\, H1$, d: $\frac{M}{V}$, I: ratsimp(expand(cilindro(R, H))), Icavo: ratsimp(expand(cilindro$(R1, H1)$)), print(Matrice di Ineriza), print(ratsimp$(d\, I)$)$\Big)$

```
(%i54) cilindroSV(R,H,H[1],R[1]);
```

Matrice di Ineriza

$$\begin{pmatrix} \frac{(3\,H_1\,R_1^4+H_1^3\,R_1^2)\,M}{12\,H\,R^2-12\,H_1\,R_1^2} & 0 & 0 \\ 0 & \frac{(3\,H_1\,R_1^4+H_1^3\,R_1^2)\,M}{12\,H\,R^2-12\,H_1\,R_1^2} & 0 \\ 0 & 0 & \frac{H_1\,R_1^4\,M}{2\,H\,R^2-2\,H_1\,R_1^2} \end{pmatrix}$$

(%o54)
$$\begin{pmatrix} \frac{(3\,H_1\,R_1^4+H_1^3\,R_1^2)\,M}{12\,H\,R^2-12\,H_1\,R_1^2} & 0 & 0 \\ 0 & \frac{(3\,H_1\,R_1^4+H_1^3\,R_1^2)\,M}{12\,H\,R^2-12\,H_1\,R_1^2} & 0 \\ 0 & 0 & \frac{H_1\,R_1^4\,M}{2\,H\,R^2-2\,H_1\,R_1^2} \end{pmatrix}$$

(%i55)

Sfera di raggio R

$$\mathbb{I} = \frac{M}{V}\iiint \begin{pmatrix} y^2+z^2 & -xy & -xz \\ -xy & x^2+z^2 & -yz \\ -xz & -yz & x^2+y^2 \end{pmatrix} \partial x \partial y \partial z$$

$$V = \frac{4}{3}\pi R^3$$

$$\rho = \frac{M}{\frac{4}{3}\pi R^3}$$

Al fine di svolgere l'integrale, una delle scelte è quella di ricorrere alle coordinate sferiche. In particolare:

$$\begin{cases} x = \rho \sin(\phi)\cos(\theta) \\ y = \rho \sin(\phi)\sin(\theta) \\ z = \rho \cos(\phi) \end{cases} \longrightarrow \theta \in [0; 2\pi], \rho \in \left[-\frac{R}{2}, \frac{R}{2}\right], \phi \in [0; 2\pi]$$

```
(%i57) sfera(R):=block([x,y,z,a,b,c,d],
                     x:rho*cos(theta)*sin(phi),
                     y:rho*sin(theta)*sin(phi),
                     z:rho*cos(phi),
                      V:(4/3)*%pi*(R)^3,
          d:M/V,
          x2:integrate(integrate(integrate(rho^2*sin(phi)*x^2,theta,0,
2*%pi),rho,0,R),phi,0,%pi),
          y2:integrate(integrate(integrate(rho^2*sin(phi)*y^2,theta,0,
2*%pi),rho,0,R),phi,0,%pi),
          z2:integrate(integrate(integrate(rho^2*sin(phi)*z^2,theta,0,
2*%pi),rho,0,R),phi,0,%pi),
          xy:integrate(integrate(integrate(rho^2*sin(phi)*x*y,theta,0,
2*%pi),rho,0,R),phi,0,%pi),
          xz:integrate(integrate(integrate(rho^2*sin(phi)*x*z,theta,0,
2*%pi),rho,0,R),phi,0,%pi),
          yz:integrate(integrate(integrate(rho^2*sin(phi)*y*z,theta,0,
2*%pi),rho,0,R),phi,0,%pi),

          I:zeromatrix(3,3),
          I[1][1]:y2+z2,I[1][2]:-xy,I[1][3]:-xz,
          I[2][1]:I[1][2],I[2][2]:x2+z2,I[2][3]:-yz,
          I[3][1]:I[1][3],I[3][2]:I[2][3],I[3][3]:x2+y2,
          I:ratsimp(expand(d*I)),
          print("Matrice di Ineriza"),
          print(I))
```

(%o57) $\mathrm{sfera}(R) := \textbf{block}\Big([x, y, z, a, b, c, d], x\colon \rho\cos(\vartheta)\sin(\varphi), y\colon \rho\sin(\vartheta)\sin(\varphi), z\colon \rho\cos(\varphi), V\colon \frac{4}{3}\pi R^3, d\colon \frac{M}{V}, x2\colon \mathrm{integrate}(\mathrm{integrate}(\mathrm{integrate}(\rho^2\sin(\varphi)\,x^2, \vartheta, 0, 2\pi), \rho, 0, R), \varphi, 0, \pi), y2\colon \mathrm{integrate}(\mathrm{integrate}(\mathrm{integrate}(\rho^2\sin(\varphi)\,y^2, \vartheta, 0, 2\pi), \rho, 0, R), \varphi, 0, \pi), z2\colon \mathrm{integrate}(\mathrm{integrate}(\mathrm{integrate}(\rho^2\sin(\varphi)\,z^2, \vartheta, 0, 2\pi), \rho, 0, R), \varphi, 0, \pi), \mathrm{xy}\colon \mathrm{integrate}(\mathrm{integrate}(\mathrm{integrate}(\rho^2\sin(\varphi)\,x\,y, \vartheta, 0, 2\pi), \rho, 0, R), \varphi, 0, \pi), \mathrm{xz}\colon \mathrm{integrate}(\mathrm{integrate}(\mathrm{integrate}(\rho^2\sin(\varphi)\,x\,z, \vartheta, 0, 2\pi), \rho, 0, R), \varphi, 0, \pi), \mathrm{yz}\colon \mathrm{integrate}(\mathrm{integrate}(\mathrm{integrate}(\rho^2\sin(\varphi)\,y\,z, \vartheta, 0, 2\pi), \rho, 0, R), \varphi, 0, \pi), I\colon \mathrm{zeromatrix}(3, 3), (I_1)_1\colon y2 + z2, (I_1)_2\colon -\mathrm{xy}, (I_1)_3\colon -\mathrm{xz}, (I_2)_1\colon (I_1)_2, (I_2)_2\colon x2 + z2, (I_2)_3\colon -\mathrm{yz}, (I_3)_1\colon (I_1)_3, (I_3)_2\colon (I_2)_3, (I_3)_3\colon x2 + y2, I\colon \mathrm{ratsimp}(\mathrm{expand}(d\,I)), \mathrm{print}(\text{Matrice di Ineriza }), \mathrm{print}(I)\Big)$

```
(%i58) sfera(R);
```

Matrice di Ineriza

$$\begin{pmatrix} \frac{2MR^2}{5} & 0 & 0 \\ 0 & \frac{2MR^2}{5} & 0 \\ 0 & 0 & \frac{2MR^2}{5} \end{pmatrix}$$

(%o58) $$\begin{pmatrix} \frac{2MR^2}{5} & 0 & 0 \\ 0 & \frac{2MR^2}{5} & 0 \\ 0 & 0 & \frac{2MR^2}{5} \end{pmatrix}$$

(%i59)

Sfera cava di raggio R e R'

```
Maxima 5.44.0 http://maxima.sourceforge.net
using Lisp SBCL 2.0.0
Distributed under the GNU Public License. See the file COPYING.
Dedicated to the memory of William Schelter.
The function bug_report() provides bug reporting information.
```

```
(%i61) sfera(R,R1):=block([x,y,z,a,b,c,d],
                          x:rho*cos(theta)*sin(phi),
                          y:rho*sin(theta)*sin(phi),
                          z:rho*cos(phi),
                           V:(4/3)*%pi*(R)^3-(4/3)*%pi*(R1)^3,
              d:M/V,
              x2:integrate(integrate(integrate(rho^2*sin(phi)*x^2,theta,0,
       2*%pi),rho,R1,R),phi,0,%pi),
              y2:integrate(integrate(integrate(rho^2*sin(phi)*y^2,theta,0,
       2*%pi),rho,R1,R),phi,0,%pi),
              z2:integrate(integrate(integrate(rho^2*sin(phi)*z^2,theta,0,
       2*%pi),rho,R1,R),phi,0,%pi),
              xy:integrate(integrate(integrate(rho^2*sin(phi)*x*y,theta,0,
       2*%pi),rho,R1,R),phi,0,%pi),
              xz:integrate(integrate(integrate(rho^2*sin(phi)*x*z,theta,0,
       2*%pi),rho,R1,R),phi,0,%pi),
              yz:integrate(integrate(integrate(rho^2*sin(phi)*y*z,theta,0,
       2*%pi),rho,R1,R),phi,0,%pi),

              I:zeromatrix(3,3),
              I[1][1]:y2+z2,I[1][2]:-xy,I[1][3]:-xz,
              I[2][1]:I[1][2],I[2][2]:x2+z2,I[2][3]:-yz,
              I[3][1]:I[1][3],I[3][2]:I[2][3],I[3][3]:x2+y2,
              I:ratsimp(expand(d*I)),
              print("Matrice di Ineriza"),
              print(I))
```

(%o61) $\mathrm{sfera}(R, R1) := \mathbf{block}\Big([x, y, z, a, b, c, d], x\colon \rho\cos(\vartheta)\sin(\varphi), y\colon \rho\sin(\vartheta)\sin(\varphi), z\colon \rho\cos(\varphi), V\colon \frac{4}{3}\pi R^3 - \frac{4}{3}\pi R1^3, d\colon \frac{M}{V}, x2\colon \mathrm{integrate}(\mathrm{integrate}(\mathrm{integrate}(\rho^2\sin(\varphi)\, x^2, \vartheta, 0, 2\pi), \rho, R1, R), \varphi, 0, \pi), y2\colon \mathrm{integrate}(\mathrm{integrate}(\mathrm{integrate}(\rho^2\sin(\varphi)\, y^2, \vartheta, 0, 2\pi), \rho, R1, R), \varphi, 0, \pi), z2\colon \mathrm{integrate}(\mathrm{integrate}(\mathrm{integrate}(\rho^2\sin(\varphi)\, z^2, \vartheta, 0, 2\pi), \rho, R1, R), \varphi, 0, \pi), \mathrm{xy}\colon \mathrm{integrate}(\mathrm{integrate}(\mathrm{integrate}(\rho^2\sin(\varphi)\, x\, y, \vartheta, 0, 2\pi), \rho, R1, R), \varphi, 0, \pi), \mathrm{xz}\colon \mathrm{integrate}(\mathrm{integrate}(\mathrm{integrate}(\rho^2\sin(\varphi)\, x\, z, \vartheta, 0, 2\pi), \rho, R1, R), \varphi, 0, \pi), \mathrm{yz}\colon \mathrm{integrate}(\mathrm{integrate}(\mathrm{integrate}(\rho^2\sin(\varphi)\, y\, z, \vartheta, 0, 2\pi), \rho, R1, R), \varphi, 0, \pi), I\colon \mathrm{zeromatrix}(3, 3), (I_1)_1\colon y2 + z2, (I_1)_2\colon -\mathrm{xy}, (I_1)_3\colon -\mathrm{xz}, (I_2)_1\colon (I_1)_2, (I_2)_2\colon x2 + z2, (I_2)_3\colon -\mathrm{yz}, (I_3)_1\colon (I_1)_3, (I_3)_2\colon (I_2)_3, (I_3)_3\colon x2 + y2, I\colon \mathrm{ratsimp}(\mathrm{expand}(d\, I)), \mathrm{print}(\text{Matrice di Ineriza }), \mathrm{print}(I)\Big)$

```
(%i62) sfera(R,R[1])
```

Matrice di Ineriza

$$\left(\frac{2\,M\,R^4+2\,R_1\,M\,R^3+2\,R_1^2\,M\,R^2+2\,R_1^3\,M\,R+2\,R_1^4\,M}{5\,R^2+5\,R_1\,R+5\,R_1^2}, 0, 0; 0, \frac{2\,M\,R^4+2\,R_1\,M\,R^3+2\,R_1^2\,M\,R^2+2\,R_1^3\,M\,R+2\,R_1^4\,M}{5\,R^2+5\,R_1\,R+5\,R_1^2}, 0; 0, 0, \frac{2\,M\,R^4+2\,R_1\,M\,R^3+2\,R_1^2\,M\,R^2+2\,R_1^3\,M\,R+2\,R_1^4\,M}{5\,R^2+5\,R_1\,R+5\,R_1^2}\right)$$

(%o62) $\left(\frac{2\,M\,R^4+2\,R_1\,M\,R^3+2\,R_1^2\,M\,R^2+2\,R_1^3\,M\,R+2\,R_1^4\,M}{5\,R^2+5\,R_1\,R+5\,R_1^2}, 0, 0; 0, \frac{2\,M\,R^4+2\,R_1\,M\,R^3+2\,R_1^2\,M\,R^2+2\,R_1^3\,M\,R+2\,R_1^4\,M}{5\,R^2+5\,R_1\,R+5\,R_1^2}, 0; 0, 0, \frac{2\,M\,R^4+2\,R_1\,M\,R^3+2\,R_1^2\,M\,R^2+2\,R_1^3\,M\,R+2\,R_1^4\,M}{5\,R^2+5\,R_1\,R+5\,R_1^2}\right)$

(%i63)

Equazioni Eulero-Lagrange

Le equazioni di Eulero-Lagrange si basano sul principio di Hamilton(principio di minimizzazione) in cui, note le espressioni dell'energia cinetica e potenziale, si calcolano le equazioni del moto e quindi permette di simulare il comportamento del sistema.

Sia:

$$T := \text{energica cinetica}, U := \text{energia potenziale}$$

Si definisce l'**azione** come $\int_{\text{ti}}^{t_f} \mathcal{L} \partial t$ in cui $\mathcal{L} := \text{Lagrangiana} = T - U$.

Supponiamo di voler passare da $q(t_i) \rightarrow q(t_f)$:

Si possono percorrere innumerevoli traiettorie, tuttavia dimostra che la traiettoria a costo minimo in (t_i, t_f) coincide con $\dot{q^*}(t)$, cioè un punto di stazionarietà dell'integrale d'azione.

N.B.:

- Si definisce **punto stazionario** un punto $f(x)$ t.c. $\frac{\partial f}{\partial x} = 0$.

Inoltre ,effettuando lo sviluppo di Taylor della fuzione $y = f(x)$ che congiunge i punti dell'ipotesi.Si definisce la variazione $f(x) - f(x_0) = \partial y = J(x_0)(x - x_0) + \dots$ in cui $(x - x_0) = \partial x$. Quindi:

$$\partial y = J(x) \partial x$$

Si definisce un **punto x^* di stazionarietà** se $\partial y = 0 \forall \partial x$.

In altre parole, un punto è di stazionarietà se considere una variazione nel codominio della funzione, ottengo una variazione nulla del dominio:

$$J(x^*) = 0$$

Il principio di Hamilton afferma che il sistema si muove in maniera tale che:

$$\partial J = 0 \qquad \forall q$$

Da cui si ottengono le equazioni di Eulero-Lagrange.Esse tengono conto del tipo di sistema che si sta considerando:

-Sistemi conservativi(assenza di forze esterne e forze d'attrito):

$$\frac{\mathrm{d}}{\mathrm{dt}} \frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q} = 0$$

-Sistemi non conservativi(presenza di forze esterne):

$$\frac{\mathrm{d}}{\mathrm{dt}} \frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q} = u^T$$

Il vettore u^T rappresenta le forze esterne agenti lungo la direzione di q.

In generale, per ottenere l'equazione di un qualsiasi moto, occorre considerare anche la presenza delle forze di attrito. Si ottiene:

$$\frac{\mathrm{d}}{\mathrm{dt}} \frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q} + \frac{\partial \mathbb{F}}{\partial \dot{q}} = u^T$$

In cui $\mathbb{F} = \frac{1}{2}\dot{q}^T F \dot{q}$, $F \succeq 0$.

```
(%i1) inverseLaplace(SI,theta):=block([res,M,MC,aC,b],
                                 M:SI,
                                 MC:SI,
                                 for i:1 thru 3 do(
                                   for j:1 thru 3 do
                                      (
                                        aC:M[i,j],
                                        b:ilt(aC,s,theta),
                                        MC[i,j]:b
                                       )
                                   ),
                                 res:MC

                              )
```

(%o1) $\mathrm{inverseLaplace}(\mathrm{SI}, \vartheta) := \textbf{block}\ ([\mathrm{res}, M, \mathrm{MC}, \mathrm{aC}, b], M\!: \mathrm{SI}, \mathrm{MC}\!: \mathrm{SI},$
$\textbf{for}\ i\ \textbf{thru}\ 3\ \textbf{do}\ \textbf{for}\ j\ \textbf{thru}\ 3\ \textbf{do}\ (\mathrm{aC}\!: M_{i,j}, b\!: \mathrm{ilt}(\mathrm{aC}, s, \vartheta), \mathrm{MC}_{i,j}\!: b), \mathrm{res}\!: \mathrm{MC})$

```
(%i2) rotLaplace(k,theta):=block([res,S,I],
                             S:ident(3),
                             I:ident(3),
                           for i:1 thru 3 do
                             (
                             for j:1 thru 3 do
                                (
                                  if i=j
                                     then S[i][j]:0
                                  elseif j>i
                                     then (
                                    temp:(-1)^(j-i)*k[3-remainder(i+j,3)],
                                           S[i][j]:temp,
                                           S[j][i]:-temp

                                            )
                                  )
                              ),
                             res:inverseLaplace(invert(s*I-S),theta)


                           )
```

(%o2) $\mathrm{rotLaplace}(k, \vartheta) := \textbf{block}\ ([\mathrm{res}, S, I], S\!: \mathrm{ident}(3), I\!: \mathrm{ident}(3),$
$\textbf{for}\ i\ \textbf{thru}\ 3\ \textbf{do}\ \textbf{for}\ j\ \textbf{thru}\ 3\ \textbf{do}\ \textbf{if}\ i = j\ \textbf{then}\ (S_i)_j\!: 0\ \textbf{elseif}\ j > i\ \textbf{then}\ (\mathrm{temp}\!:$
$(-1)^{j-i} k_{3-\mathrm{remainder}(i+j,3)}, (S_i)_j\!: \mathrm{temp}, (S_j)_i\!: -\mathrm{temp}), \mathrm{res}\!: \mathrm{inverseLaplace}(\mathrm{invert}(s\, I - S), \vartheta))$

```
(%i3) Av(v,theta,d):=block([res,Trot,row,Atemp,A],
                             Trot:rotLaplace(v,theta),
                             row:matrix([0,0,0,1]),
                             Atemp:addcol(Trot,d*transpose(v)),
                             A:addrow(Atemp,row),
                             res:A
                             )
```

(%o3) $\mathrm{Av}(v,\vartheta,d) := \mathbf{block}\ ([\mathrm{res},\mathrm{Trot},\mathrm{row},\mathrm{Atemp},A], \mathrm{Trot}\colon \mathrm{rotLaplace}(v,\vartheta), \mathrm{row}\colon \begin{pmatrix} 0 & 0 & 0 & 1 \end{pmatrix}, \mathrm{Atemp}\colon \mathrm{addcol}(\mathrm{Trot}, d\,\mathrm{transpose}(v)), A\colon \mathrm{addrow}(\mathrm{Atemp},\mathrm{row}), \mathrm{res}\colon A)$

```
(%i4) Q(theta,d,alpha,a):=block([res,tempMat,Qtrasf],
                            tempMat:Av([0,0,1],theta,d).Av([1,0,0],alpha,a),
                            Qtrasf:zeromatrix(4,4),
                            for i:1 thru 4 do
                        (
                        for j:1 thru 4 do
                           (
                            Qtrasf[i][j]:trigreduce(tempMat[i][j])
                                      )
                          ),

                            res:Qtrasf
                            )
```

(%o4) $Q(\vartheta,d,\alpha,a) := \mathbf{block}\ ([\mathrm{res},\mathrm{tempMat},\mathrm{Qtrasf}], \mathrm{tempMat}\colon \mathrm{Av}([0,0,1],\vartheta,d)\cdot\mathrm{Av}([1,0,0],\alpha,a), \mathrm{Qtrasf}\colon \mathrm{zeromatrix}(4,4), \mathbf{for}\ i\ \mathbf{thru}\ 4\ \mathbf{do}\ \mathbf{for}\ j\ \mathbf{thru}\ 4\ \mathbf{do}\ (\mathrm{Qtrasf}_i)_j\colon \mathrm{trigreduce}((\mathrm{tempMat}_i)_j), \mathrm{res}\colon \mathrm{Qtrasf})$

```
(%i5) Qdirect(DH):=block([res,Q,Qtemp],
                      Q:[Q(DH[1][1],DH[1][2],DH[1][3],DH[1][4])],
                      for i:2 thru length(DH) do(

                          Qtemp:Q(DH[i][1],DH[i][2],DH[i][3],DH[i][4]),

                          Q:append(Q,[trigsimp(trigreduce(trigexpand(Q[i-
      1].Qtemp)))])


                         ),
                       res:Q)
```

(%o5) $\mathrm{Qdirect}(\mathrm{DH}) := \mathbf{block}\ ([\mathrm{res},Q,\mathrm{Qtemp}], Q\colon [Q((\mathrm{DH}_1)_1,(\mathrm{DH}_1)_2,(\mathrm{DH}_1)_3,(\mathrm{DH}_1)_4)], \mathbf{for}\ i\ \mathbf{from}\ 2\ \mathbf{thru}\ \mathrm{length}(\mathrm{DH})\ \mathbf{do}\ (\mathrm{Qtemp}\colon Q((\mathrm{DH}_i)_1,(\mathrm{DH}_i)_2,(\mathrm{DH}_i)_3,(\mathrm{DH}_i)_4), Q\colon \mathrm{append}(Q, [\mathrm{trigsimp}(\mathrm{trigreduce}(\mathrm{trigexpand}(Q_{i-1}\cdot\mathrm{Qtemp})))])), \mathrm{res}\colon Q)$

```
(%i6) Qbc(Q,bc,dist):=block([traslBC,Qcap],
                  Qcap:[],
                  ex:matrix([1],[0],[0]), ez:matrix([0],[0],[1]),
                  for j:1 thru length(Q) do(
                       traslBC:addrow(addcol(ident(3),dist[j]),[0,0,0,1]),
                       Qcap:append(Qcap,[trigsimp(Q[j].traslBC)])

                       ),
                  Qcap
                  )
```

(%o6) $\mathrm{Qbc}(Q,\mathrm{bc},\mathrm{dist}) := \mathbf{block}\left([\mathrm{traslBC},\mathrm{Qcap}], \mathrm{Qcap}\colon [], \mathrm{ex}\colon \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \mathrm{ez}\colon \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \mathbf{for}\ j\ \mathbf{thru}\ \mathrm{length}(Q)\ \mathbf{do}\ (\mathrm{traslBC}\colon \mathrm{addrow}(\mathrm{addcol}(\mathrm{ident}(3),\mathrm{dist}_j),[0,0,0,1]), \mathrm{Qcap}\colon \mathrm{append}(\mathrm{Qcap}, [\mathrm{trigsimp}(Q_j\cdot\mathrm{traslBC})])), \mathrm{Qcap}\right)$

(%i7)
```
inerzia(j):=block(
[II],
II:matrix([alpha[xx][i],alpha[xy][i],alpha[xz][i]],
          [alpha[xy][i],alpha[yy][i],alpha[yz][i]],
          [alpha[xz][i],alpha[yz][i],alpha[zz][i]]),
II:subst(j, i, II),
return(II)
)
```

(%o7) $\mathrm{inerzia}(j) := \mathbf{block}\left([\mathrm{II}], \mathrm{II}\colon \begin{pmatrix} (\alpha_{\mathrm{xx}})_i & (\alpha_{\mathrm{xy}})_i & (\alpha_{\mathrm{xz}})_i \\ (\alpha_{\mathrm{xy}})_i & (\alpha_{\mathrm{yy}})_i & (\alpha_{\mathrm{yz}})_i \\ (\alpha_{\mathrm{xz}})_i & (\alpha_{\mathrm{yz}})_i & (\alpha_{\mathrm{zz}})_i \end{pmatrix}, \mathrm{II}\colon \mathrm{subst}(j, i, \mathrm{II}), \mathrm{return}(\mathrm{II}) \right)$

(%i8)
```
massa(k):=M[k];
```

(%o8) $\mathrm{massa}(k) := M_k$

(%i9)
```
ek(DH,dist):=block([Q,Qcap,I,wtemp,w,Si,Tatemp,Ta,Tbtemp,Tb,d,dd,Qend,B,
Btemp,T,Tot,Btot,res],
                    I:[],w:[],Ta:[],Tb:[],B:[],T:[],Ttot:0,
                    depends([q],t),
                    Q:Qdirect(DH),

                    Qcap:Qbc(Q,DH,dist),

                   for i:1 thru length(Qcap) do( I:append(I,[inerzia(i)]),
                        R:matrix([Qcap[i][1][1],Qcap[i][1][2],
Qcap[i][1][3]],[Qcap[i][2][1],Qcap[i][2][2],Qcap[i][2][3]],[Qcap[i][3][1],
Qcap[i][3][2],Qcap[i][3][3]]),
                       dR:diff(R,t),
                      /* for j:1 thru length(DH) do(
                            dR:subst('diff(q[j],t)=omega[j],dR)),*/
                       Sw:dR.transpose(R),
                       wtemp:matrix([Sw[3][2]],[Sw[1][3]],[Sw[2][1]]),
                       w:append(w,[trigreduce(expand(wtemp))]),
                 Tatemp:(1/2)*transpose(wtemp).R.I[i].transpose(R).wtemp,
                 Tatemp:trigsimp(trigreduce(trigexpand(Tatemp))),
                  Ta:append(Ta,[Tatemp]),
                  d:matrix([Qcap[i][1][4]],[Qcap[i][2][4]],[Q[i][3][4]]),
                  dd:diff(d,t),
                 /* for j:1 thru length(DH) do(dd:subst('diff(q[j],
t)=omega[j],dd)),*/
                  Tbtemp:(massa(i)/
2)*trigsimp(trigreduce(trigexpand(transpose(dd).dd))),
                  Tb:append(Tb,[Tbtemp]),
                  T:append(T,[trigreduce(Tatemp+Tbtemp)])),
                  for i:1 thru length(DH) do(
                    Ttot:T[i]+Ttot
                  ),
                  Ttot
                   )
```

(%o9) $\mathrm{ek}(\mathrm{DH}, \mathrm{dist}) := \mathbf{block}\Big([Q, \mathrm{Qcap}, I, \mathrm{wtemp}, w, \mathrm{Si}, \mathrm{Tatemp}, \mathrm{Ta}, \mathrm{Tbtemp}, \mathrm{Tb}, d, \mathrm{dd}, \mathrm{Qend}, B,$

$\mathrm{Btemp}, T, \mathrm{Tot}, \mathrm{Btot}, \mathrm{res}], I\colon [], w\colon [], \mathrm{Ta}\colon [], \mathrm{Tb}\colon [], B\colon [], T\colon [], \mathrm{Ttot}\colon 0, \mathrm{depends}([q], t), Q\colon \mathrm{Qdirect}(\mathrm{DH}),$

Qcap: Qbc(Q, DH, dist), **for** i **thru** length(Qcap) **do** $\Bigg($ I: append(I, [inerzia(i)]), R:
$\begin{pmatrix} ((\mathrm{Qcap}_i)_1)_1 & ((\mathrm{Qcap}_i)_1)_2 & ((\mathrm{Qcap}_i)_1)_3 \\ ((\mathrm{Qcap}_i)_2)_1 & ((\mathrm{Qcap}_i)_2)_2 & ((\mathrm{Qcap}_i)_2)_3 \\ ((\mathrm{Qcap}_i)_3)_1 & ((\mathrm{Qcap}_i)_3)_2 & ((\mathrm{Qcap}_i)_3)_3 \end{pmatrix}$, dR: diff($R, t$), Sw: dR · transpose($R$), wtemp:
$\begin{pmatrix} (\mathrm{Sw}_3)_2 \\ (\mathrm{Sw}_1)_3 \\ (\mathrm{Sw}_2)_1 \end{pmatrix}$, w: append(w, [trigreduce(expand(wtemp))]), Tatemp: $\frac{1}{2}$ transpose(wtemp) · $R \cdot I_i \cdot$
transpose(R) · wtemp, Tatemp: trigsimp(trigreduce(trigexpand(Tatemp))), Ta: append(Ta,
[Tatemp]), d: $\begin{pmatrix} ((\mathrm{Qcap}_i)_1)_4 \\ ((\mathrm{Qcap}_i)_2)_4 \\ ((Q_i)_3)_4 \end{pmatrix}$, dd: diff($d, t$), Tbtemp:
$\frac{\mathrm{massa}(i)}{2}$ trigsimp(trigreduce(trigexpand(transpose(dd) · dd))), Tb: append(Tb, [Tbtemp]), T:
append(T, [trigreduce(Tatemp + Tbtemp)]) $\Bigg)$, **for** i **thru** length(DH) **do** Ttot: T_i + Ttot, Ttot $\Bigg)$

```
(%i10) ep(DH,dist):=block([Q,Qcap,g,U,Utemp,dTemp,prev,Utot],
                        Q:[], Qcap:[],U:[],Utot:zeromatrix(3,3),Utot:0,
                         depends([q,omega],t),
                        g:10*matrix([0],[0],[1]),
                        prev:ident(4),
                        Q:Qdirect(DH),
                         Qcap:Qbc(Q,DH,dist),

                           for i:1 thru length(Qcap) do(

                            dTemp:matrix([Qcap[i][1][4]],[Qcap[i][2][4]],
                                          [Qcap[i][3][4]]),

                            Utemp:M[i]*transpose(g).dTemp,
                            U:append(U,[Utemp])

                          ),
                           for i:1 thru length(U) do(
                               Utot:Utot+U[i]
                           ),
                           ratsimp(trigsimp(trigreduce(trigexpand(Utot))))
                           );
```

(%o10) ep(DH, dist) := **block** $\Bigg($ [Q, Qcap, g, U, Utemp, dTemp, prev, Utot], Q: [], Qcap: [], U: [],
Utot: zeromatrix(3, 3), Utot: 0, depends([q, ω], t), g: 10 $\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$, prev: ident(4), Q: Qdirect(DH),
Qcap: Qbc(Q, DH, dist), **for** i **thru** length(Qcap) **do** $\Bigg($ dTemp: $\begin{pmatrix} ((\mathrm{Qcap}_i)_1)_4 \\ ((\mathrm{Qcap}_i)_2)_4 \\ ((\mathrm{Qcap}_i)_3)_4 \end{pmatrix}$, Utemp:
M_i transpose(g) · dTemp, U: append(U, [Utemp]) $\Bigg)$, **for** i **thru** length(U) **do** Utot: Utot + U_i,
ratsimp(trigsimp(trigreduce(trigexpand(Utot)))) $\Bigg)$

```
(%i11) eel(dh,f,u,dist):=block([T,U,L,dLq,dLqt,dLqp,F,eq,eqi,vel],
                 print(q[i][i],"Indica la derivata i-esima di",q[i]),
                 T:0,U:0,eq:zeromatrix(length(dh),1),vel:zeromatrix(length(dh),1),
                 dLq:zeromatrix(length(dh),1),dLqt:zeromatrix(length(dh),1),
                 dLqp:zeromatrix(length(dh),1),dF:zeromatrix(length(dh),1),
                 eq:zeromatrix(length(dh),1),depends([q],t),
                 for i:1 thru length(dh) do(vel[i][1]:diff(q[i],t)),
                 T:ek(dh,dist), U:ep(dh,dist),
                 if length(dh)=1 then(F:(1/2)*(transpose(vel).vel)*f)else
                 (F:(1/2)*expand(transpose(vel).f.vel)),
                 L:trigsimp(trigreduce(trigexpand(T-U))),
                 for i:1 thru length(dh) do
                    (dLq[i][1]:diff(L,'diff(q[i],t)),
                     dLqt[i][1]:diff(dLq[i][1],t),
                     dLqp[i][1]:diff(L,q[i]),
                     dF[i][1]:expand(diff(F,'diff(q[i],t))),
                     eq[i][1]:dLqt[i][1]-dLqp[i][1]+dF[i][1]-u[i]),
                 for i:1 thru length(dh) do
                    (T:subst('diff(q[i],t)=q[i][1],T),
                     T:subst('diff(diff(q[i],t),t)=q[i][2],T),
                     F:subst('diff(q[i],t)=q[i][1],F),
                     F:subst('diff(diff(q[i],t),t)=q[i][2],F),
                     U:subst('diff(q[i],t)=q[i][1],U),
                     U:subst('diff(diff(q[i],t),t)=q[i][2],U),
                     L:subst('diff(q[i],t)=q[i][1],L),
                     L:subst('diff(diff(q[i],t),t)=q[i][2],L),
                     dLq:subst('diff(q[i],t)=q[i][1],dLq),
                     dLq:subst('diff(diff(q[i],t),t)=q[i][2],dLq),
                     dLqt:subst('diff(q[i],t)=q[i][1],dLqt),
                     dLqt:subst('diff(diff(q[i],t),t)=q[i][2],dLqt),
                     dLqp:subst('diff(q[i],t)=q[i][1],dLqp),
                     dLqp:subst('diff(diff(q[i],t),t)=q[i][2],dLqp),
                     dF:subst('diff(q[i],t)=q[i][1],dF),
                     dF:subst('diff(diff(q[i],t),t)=q[i][2],dF),
                     eq:subst('diff(q[i],t)=q[i][1],eq),
                     eq:subst('diff(diff(q[i],t),t)=q[i][2],eq)),
                     print("Energia cinetica T=",ratsimp(expand(T))),
                     print("Energia potenziale U=",ratsimp(expand(U))),
                     print("Forze di attrito F=",F),
                     print("Forze esterne u=",u),
                     print("Lagrangiana L=",ratsimp(trigreduce(expand(L)))),
                     print("dL/dq' = ", ratsimp(trigreduce(expand(dLq)))),
                     print("d/dt dL/dq' = ", ratsimp(trigreduce(expand(dLqt)))),
                     print("dL/dq = ",ratsimp(trigreduce(expand(dLqp)))),
                     print("dF/dq' = ",ratsimp(dF)),
                     for i:1 thru length(dh) do(
                          print("Equazione eulero lagrange",i),
                                 print(ratsimp(eq[i][1][1]),"=0")
                     ));
```

(%o11) $\mathrm{eel}(\mathrm{dh}, f, u, \mathrm{dist}) := \mathbf{block}\Big([T, U, L, \mathrm{dLq}, \mathrm{dLqt}, \mathrm{dLqp}, F, \mathrm{eq}, \mathrm{eqi}, \mathrm{vel}], \mathrm{print}((q_i)_i,$ Indica la derivata i-esima di , $q_i), T\colon 0, U\colon 0,$ eq: zeromatrix(length(dh), 1), vel: zeromatrix(length(dh), 1), dLq: zeromatrix(length(dh), 1), dLqt: zeromatrix(length(dh), 1), dLqp: zeromatrix(length(dh), 1), dF: zeromatrix(length(dh), 1), eq: zeromatrix(length(dh), 1), depends($[q], t$),
for i **thru** length(dh) **do** $(\mathrm{vel}_i)_1\colon \mathrm{diff}(q_i, t), T\colon \mathrm{ek}(\mathrm{dh}, \mathrm{dist}), U\colon \mathrm{ep}(\mathrm{dh}, \mathrm{dist}),$ **if** length(dh) =

1 **then** $F{:}\,\frac{1}{2}\mathrm{transpose}(\mathrm{vel})\cdot \mathrm{vel}\; f$ **else** $F{:}\,\frac{1}{2}\mathrm{expand}(\mathrm{transpose}(\mathrm{vel})\cdot f\cdot \mathrm{vel}), L{:}$ $\mathrm{trigsimp}(\mathrm{trigreduce}(\mathrm{trigexpand}(T-U)))$, **for** i **thru** length(dh) **do** $\Big((\mathrm{dLq}_i)_1{:}\,\mathrm{diff}\Big(L, \frac{1}{\mathrm{mtimes}()}\,q_i\Big), (\mathrm{dLqt}_i)_1{:}\,\mathrm{diff}((\mathrm{dLq}_i)_1, t), (\mathrm{dLqp}_i)_1{:}\,\mathrm{diff}(L, q_i), (\mathrm{dF}_i)_1{:}\,\mathrm{expand}\Big(\mathrm{diff}\Big(F, \frac{1}{\mathrm{mtimes}()}\,q_i\Big)\Big), (\mathrm{eq}_i)_1{:}\,(\mathrm{dLqt}_i)_1-(\mathrm{dLqp}_i)_1+(\mathrm{dF}_i)_1-u_i\Big)$, **for** i **thru** length(dh) **do** $\Big(T{:}\,\mathrm{subst}\Big(\frac{1}{\mathrm{mtimes}()}\,q_i=(q_i)_1, T\Big), T{:}\,\mathrm{subst}\Big(\frac{1}{\mathrm{mtimes}()}\,\mathrm{diff}(q_i,t)=(q_i)_2, T\Big), F{:}\,\mathrm{subst}\Big(\frac{1}{\mathrm{mtimes}()}\,q_i=(q_i)_1, F\Big), F{:}\,\mathrm{subst}\Big(\frac{1}{\mathrm{mtimes}()}\,\mathrm{diff}(q_i,t)=(q_i)_2, F\Big), U{:}\,\mathrm{subst}\Big(\frac{1}{\mathrm{mtimes}()}\,q_i=(q_i)_1, U\Big), U{:}\,\mathrm{subst}\Big(\frac{1}{\mathrm{mtimes}()}\,\mathrm{diff}(q_i,t)=(q_i)_2, U\Big), L{:}\,\mathrm{subst}\Big(\frac{1}{\mathrm{mtimes}()}\,q_i=(q_i)_1, L\Big), L{:}\,\mathrm{subst}\Big(\frac{1}{\mathrm{mtimes}()}\,\mathrm{diff}(q_i,t)=(q_i)_2, L\Big), \mathrm{dLq}{:}\,\mathrm{subst}\Big(\frac{1}{\mathrm{mtimes}()}\,q_i=(q_i)_1, \mathrm{dLq}\Big), \mathrm{dLq}{:}\,\mathrm{subst}\Big(\frac{1}{\mathrm{mtimes}()}\,\mathrm{diff}(q_i,t)=(q_i)_2, \mathrm{dLq}\Big), \mathrm{dLqt}{:}\,\mathrm{subst}\Big(\frac{1}{\mathrm{mtimes}()}\,q_i=(q_i)_1, \mathrm{dLqt}\Big), \mathrm{dLqt}{:}\,\mathrm{subst}\Big(\frac{1}{\mathrm{mtimes}()}\,\mathrm{diff}(q_i,t)=(q_i)_2, \mathrm{dLqt}\Big), \mathrm{dLqp}{:}\,\mathrm{subst}\Big(\frac{1}{\mathrm{mtimes}()}\,q_i=(q_i)_1, \mathrm{dLqp}\Big), \mathrm{dLqp}{:}\,\mathrm{subst}\Big(\frac{1}{\mathrm{mtimes}()}\,\mathrm{diff}(q_i,t)=(q_i)_2, \mathrm{dLqp}\Big), \mathrm{dF}{:}\,\mathrm{subst}\Big(\frac{1}{\mathrm{mtimes}()}\,q_i=(q_i)_1, \mathrm{dF}\Big), \mathrm{dF}{:}\,\mathrm{subst}\Big(\frac{1}{\mathrm{mtimes}()}\,\mathrm{diff}(q_i,t)=(q_i)_2, \mathrm{dF}\Big), \mathrm{eq}{:}\,\mathrm{subst}\Big(\frac{1}{\mathrm{mtimes}()}\,q_i=(q_i)_1, \mathrm{eq}\Big), \mathrm{eq}{:}\,\mathrm{subst}\Big(\frac{1}{\mathrm{mtimes}()}\,\mathrm{diff}(q_i,t)=(q_i)_2, \mathrm{eq}\Big)\Big)$, print(Energia cinetica T= , ratsimp(expand(T))), print(Energia potenziale U= , ratsimp(expand(U))), print(Forze di attrito F= , F), print(Forze esterne u= , u), print(Lagrangiana L= , ratsimp(trigreduce(expand(L)))), print(dL/dq' = , ratsimp(trigreduce(expand(dLq)))), print(d/dt dL/dq' = , ratsimp(trigreduce(expand(dLqt)))), print(dL/dq = , ratsimp(trigreduce(expand(dLqp)))), print(dF/dq' = , ratsimp(dF)),
for i **thru** length(dh) **do** (print(Equazione eulero lagrange , i), print(ratsimp($((\mathrm{eq}_i)_1)_1$), =0)) $\Big)$

2DOF

```
(%i12) dueDof:[[q[1],0,0,L[1]],[q[2],0,0,L[2]]];
```

(%o12) $[[q_1, 0, 0, L_1], [q_2, 0, 0, L_2]]$

```
(%i13) u:matrix([u[1]],[u[2]])
```

(%o13) $\begin{pmatrix} u_1 \\ u_2 \end{pmatrix}$

```
(%i14) F:matrix([K[11],0],[0,K[22]]);
```

(%o14) $\begin{pmatrix} K_{11} & 0 \\ 0 & K_{22} \end{pmatrix}$

```
(%i15) distance:[matrix([-L[1]/2],[0],[0]),matrix([-L[2]/2],[0],[0])];
```

(%o15) $\left[\begin{pmatrix} -\frac{L_1}{2} \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} -\frac{L_2}{2} \\ 0 \\ 0 \end{pmatrix}\right]$

```
(%i16) eel(dueDof,F,u,distance);
```

$(q_i)_i$ Indica la derivata i-esima di q_i

Energia cinetica T= $((4\,(q_1)_1\,(q_2)_1+4\,(q_1)_1^2)\,L_1\,L_2\,M_2\cos(q_2)+(4\,(q_2)_1^2+8\,(q_1)_1\,(q_2)_1+4\,(q_1)_1^2)\,(\alpha_{zz})_2+(((q_2)_1^2+2\,(q_1)_1\,(q_2)_1+(q_1)_1^2)\,L_2^2+4\,(q_1)_1^2\,L_1^2)\,M_2+4\,(q_1)_1^2\,(\alpha_{zz})_1+(q_1)_1^2\,L_1^2\,M_1)/8$

Energia potenziale U= 0

Forze di attrito F= $\dfrac{(q_2)_1^2\,K_{22}+(q_1)_1^2\,K_{11}}{2}$

Forze esterne u= $\begin{pmatrix} u_1 \\ u_2 \end{pmatrix}$

Lagrangiana L= $((4\,(q_1)_1\,(q_2)_1+4\,(q_1)_1^2)\,L_1\,L_2\,M_2\cos(q_2)+(4\,(q_2)_1^2+8\,(q_1)_1\,(q_2)_1+4\,(q_1)_1^2)\,(\alpha_{zz})_2+(((q_2)_1^2+2\,(q_1)_1\,(q_2)_1+(q_1)_1^2)\,L_2^2+4\,(q_1)_1^2\,L_1^2)\,M_2+4\,(q_1)_1^2\,(\alpha_{zz})_1+(q_1)_1^2\,L_1^2\,M_1)/8$

dL/dq' = $\Big(((2\,(q_2)_1+4\,(q_1)_1)\,L_1\,L_2\,M_2\cos(q_2)+(4\,(q_2)_1+4\,(q_1)_1)\,(\alpha_{zz})_2+(((q_2)_1+(q_1)_1)\,L_2^2+4\,(q_1)_1\,L_1^2)\,M_2+4\,(q_1)_1\,(\alpha_{zz})_1+(q_1)_1\,L_1^2\,M_1)/4;\ \frac{2\,(q_1)_1\,L_1\,L_2\,M_2\cos(q_2)+(4\,(q_2)_1+4\,(q_1)_1)\,(\alpha_{zz})_2+((q_2)_1+(q_1)_1)\,L_2^2\,M_2}{4}\Big)$

d/dt dL/dq' = $(-((2\,(q_2)_1^2+4\,(q_1)_1\,(q_2)_1)\,L_1\,L_2\,M_2\sin(q_2)+(-2\,L_1\,(q_2)_2-4\,L_1\,(q_1)_2)\,L_2\,M_2\cos(q_2)+(-4\,(q_2)_2-4\,(q_1)_2)\,(\alpha_{zz})_2+((-(q_2)_2-(q_1)_2)\,L_2^2-4\,L_1^2\,(q_1)_2)\,M_2+(-4\,(\alpha_{zz})_1-L_1^2\,M_1)\,(q_1)_2)/4;-(2\,(q_1)_1\,(q_2)_1\,L_1\,L_2\,M_2\sin(q_2)-2\,L_1\,(q_1)_2\,L_2\,M_2\cos(q_2)+(-4\,(q_2)_2-4\,(q_1)_2)\,(\alpha_{zz})_2+(-(q_2)_2-(q_1)_2)\,L_2^2\,M_2)/4)$

dL/dq = $\begin{pmatrix} 0 \\ -\frac{((q_1)_1\,(q_2)_1+(q_1)_1^2)\,L_1\,L_2\,M_2\sin(q_2)}{2} \end{pmatrix}$

dF/dq' = $\begin{pmatrix} (q_1)_1\,K_{11} \\ (q_2)_1\,K_{22} \end{pmatrix}$

Equazione eulero lagrange 1

$-((2\,(q_2)_1^2+4\,(q_1)_1\,(q_2)_1)\,L_1\,L_2\,M_2\sin(q_2)+(-2\,L_1\,(q_2)_2-4\,L_1\,(q_1)_2)\,L_2\,M_2\cos(q_2)-4\,(q_1)_1\,K_{11}+(-4\,(q_2)_2-4\,(q_1)_2)\,(\alpha_{zz})_2+((-(q_2)_2-(q_1)_2)\,L_2^2-4\,L_1^2\,(q_1)_2)\,M_2+(-4\,(\alpha_{zz})_1-L_1^2\,M_1)\,(q_1)_2+4\,u_1)/4=0$

Equazione eulero lagrange 2

$(2\,(q_1)_1^2\,L_1\,L_2\,M_2\sin(q_2)+2\,L_1\,(q_1)_2\,L_2\,M_2\cos(q_2)+4\,(q_2)_1\,K_{22}+(4\,(q_2)_2+4\,(q_1)_2)\,(\alpha_{zz})_2-4\,u_2+((q_2)_2+(q_1)_2)\,L_2^2\,M_2)/4=0$

(%o17) **done**

(%i18)

Robot Cilindrico

```
(%i12) cilindrico:[[q[1],L[1],0,0],[0,q[2],-%pi/2,0],[0,q[3],0,0]];
```

(%o12) $\left[[q_1, L_1, 0, 0], \left[0, q_2, -\frac{\pi}{2}, 0\right], [0, q_3, 0, 0]\right]$

```
(%i13) u:matrix([u[1]],[u[2]],[u[3]]);
```

(%o13) $\begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}$

```
(%i14) F:matrix([K[11],0,0],[0,K[22],0],[0,0,K[33]]);
```

(%o14) $\begin{pmatrix} K_{11} & 0 & 0 \\ 0 & K_{22} & 0 \\ 0 & 0 & K_{33} \end{pmatrix}$

```
(%i15) distance:[matrix([0],[0],[-L[1]/2]),matrix([0],[-L[2]/2],[0]),matrix([0],
       [0],[-L[3]/2])];
```

(%o15) $\left[\begin{pmatrix} 0 \\ 0 \\ -\frac{L_1}{2} \end{pmatrix}, \begin{pmatrix} 0 \\ -\frac{L_2}{2} \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ -\frac{L_3}{2} \end{pmatrix}\right]$

```
(%i16) eel(cilindrico,F,u,distance);
```

$(q_i)_i$ Indica la derivata i-esima di q_i

Energia cinetica T= $(4\,(q_1)_1^2\,(\alpha_{yy})_3 + 4\,(q_1)_1^2\,M_3\,q_3^2 - 4\,(q_1)_1^2\,L_3\,M_3\,q_3 + ((q_1)_1^2\,L_3^2 + 4\,(q_3)_1^2 + 4\,(q_2)_1^2)\,M_3 + 4\,(q_1)_1^2\,(\alpha_{yy})_2 + 4\,(q_2)_1^2\,M_2 + 4\,(q_1)_1^2\,(\alpha_{zz})_1)/8$

Energia potenziale U= $(10\,q_2 + 10\,L_1)\,M_3 + 10\,M_2\,q_2 + (5\,L_2 + 10\,L_1)\,M_2 + 5\,L_1\,M_1$

Forze di attrito F= $\frac{(q_3)_1^2\,K_{33} + (q_2)_1^2\,K_{22} + (q_1)_1^2\,K_{11}}{2}$

Forze esterne u= $\begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}$

Lagrangiana L= $(4\,(q_1)_1^2\,(\alpha_{yy})_3 + 4\,(q_1)_1^2\,M_3\,q_3^2 - 4\,(q_1)_1^2\,L_3\,M_3\,q_3 + ((q_1)_1^2\,L_3^2 - 80\,q_2 - 80\,L_1 + 4\,(q_3)_1^2 + 4\,(q_2)_1^2)\,M_3 + 4\,(q_1)_1^2\,(\alpha_{yy})_2 - 80\,M_2\,q_2 + (-40\,L_2 - 80\,L_1 + 4\,(q_2)_1^2)\,M_2 + 4\,(q_1)_1^2\,(\alpha_{zz})_1 - 40\,L_1\,M_1)/8$

dL/dq' = $\begin{pmatrix} \frac{4\,(q_1)_1\,(\alpha_{yy})_3 + 4\,(q_1)_1\,M_3\,q_3^2 - 4\,(q_1)_1\,L_3\,M_3\,q_3 + (q_1)_1\,L_3^2\,M_3 + 4\,(q_1)_1\,(\alpha_{yy})_2 + 4\,(q_1)_1\,(\alpha_{zz})_1}{4} \\ (q_2)_1\,M_3 + (q_2)_1\,M_2 \\ (q_3)_1\,M_3 \end{pmatrix}$

d/dt dL/dq' = $((4\,(q_1)_2\,(\alpha_{yy})_3 + 4\,(q_1)_2\,M_3\,q_3^2 + (8\,(q_1)_1\,(q_3)_1 - 4\,(q_1)_2\,L_3)\,M_3\,q_3 + ((q_1)_2\,L_3^2 - 4\,(q_1)_1\,(q_3)_1\,L_3)\,M_3 + 4\,(q_1)_2\,(\alpha_{yy})_2 + 4\,(\alpha_{zz})_1\,(q_1)_2)/4;\ (q_2)_2\,M_3 + (q_2)_2\,M_2;\ (q_3)_2\,M_3)$

dL/dq = $\begin{pmatrix} 0 \\ -10\,M_3 - 10\,M_2 \\ \frac{2\,(q_1)_1^2\,M_3\,q_3 - (q_1)_1^2\,L_3\,M_3}{2} \end{pmatrix}$

dF/dq' = $\begin{pmatrix} (q_1)_1\,K_{11} \\ (q_2)_1\,K_{22} \\ (q_3)_1\,K_{33} \end{pmatrix}$

Equazione eulero lagrange 1

$(4\,(q_1)_1\,K_{11} + 4\,(q_1)_2\,(\alpha_{yy})_3 + 4\,(q_1)_2\,M_3\,q_3^2 + (8\,(q_1)_1\,(q_3)_1 - 4\,(q_1)_2\,L_3)\,M_3\,q_3 + ((q_1)_2\,L_3^2 - 4\,(q_1)_1\,(q_3)_1\,L_3)\,M_3 + 4\,(q_1)_2\,(\alpha_{yy})_2 + 4\,(\alpha_{zz})_1\,(q_1)_2 - 4\,u_1)/4=0$

Equazione eulero lagrange 2

$(q_2)_1\,K_{22} + ((q_2)_2 + 10)\,M_3 - u_2 + ((q_2)_2 + 10)\,M_2=0$

Equazione eulero lagrange 3

$\frac{2\,(q_3)_1\,K_{33} - 2\,u_3 - 2\,(q_1)_1^2\,M_3\,q_3 + ((q_1)_1^2\,L_3 + 2\,(q_3)_2)\,M_3}{2}=0$

(%o16) **done**

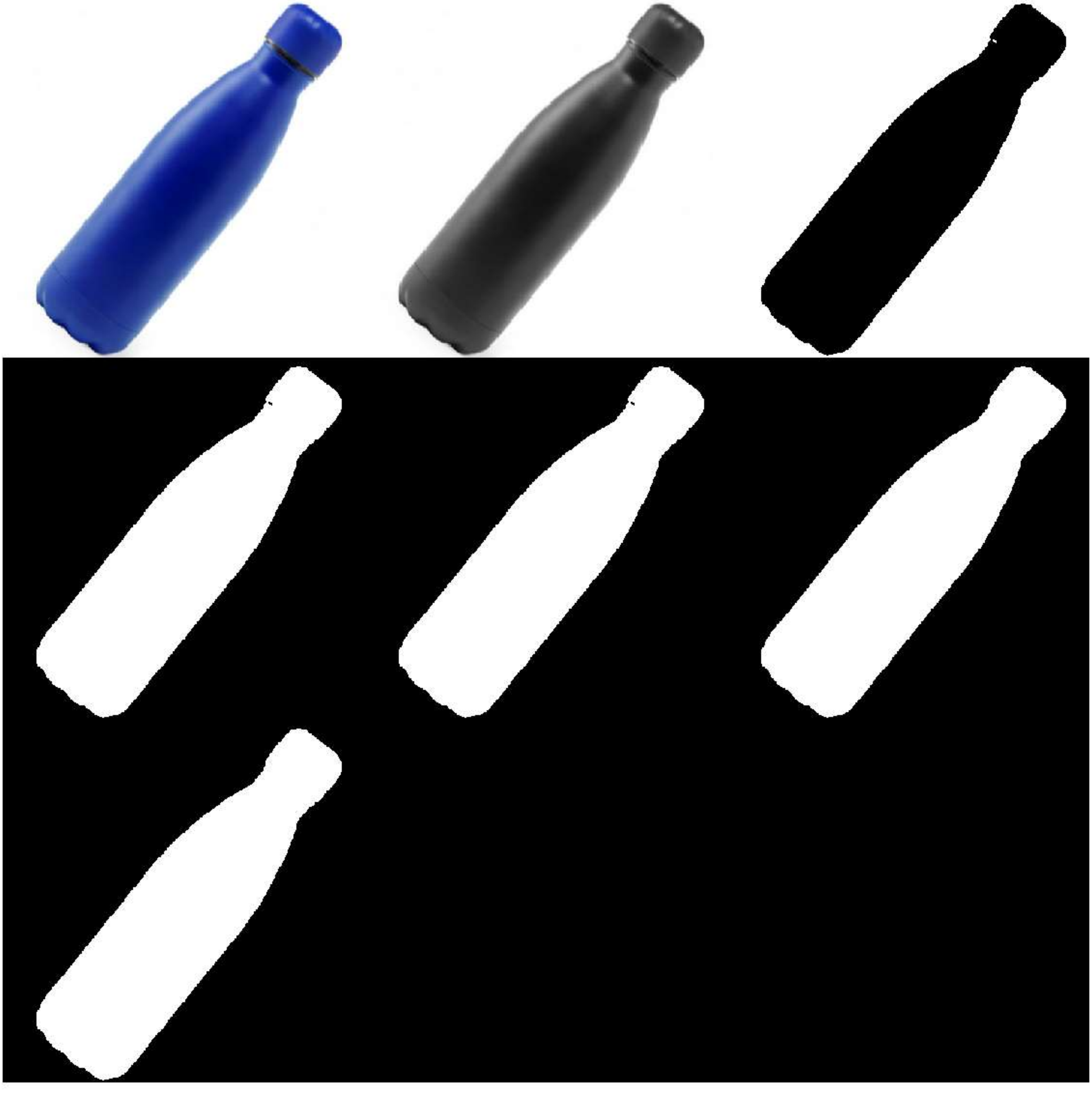
(%i17)

```
%%Visione artificiale

clearvars
close all
clc
syms x diag1(x) diag2(x)

%% Lettura immagine
% imgRGB=imread('triangolo.jpg');
% imgRGB=imread('rondella.jpg');
imgRGB=imread('bottiglia.jpg');

prefRes=[480,640];
width = size(imgRGB, 2);     % Larghezza iniziale dell'immagine
height = size(imgRGB, 1);    % Altezza iniziale dell'immagine
%% Ridimensionamento Immagine
if (height > prefRes(1))
    imgRGB = imresize(imgRGB, [prefRes(1) NaN]);
end
if (width > prefRes(2))
    imgRGB = imresize(imgRGB, [NaN prefRes(2)]);
end
width = size(imgRGB, 2);     % Larghezza dell'immagine post-compressione
height = size(imgRGB, 1);    % Altezza dell'immagine post-compressione
res = width*height;          % Risoluzione dell'immagine post-compressione
center = [floor((width+1)/2), floor((height+1)/2)];
%% RGB to gray scale
I= rgb2gray(imgRGB);

%% grey scale to BW
BW = imbinarize(I);
BW1=BW;
%% Verifica della luminosità dell'immagine, se troppo chiara effettuo il complementare
% Somma righe e colonne dei pixel dell'immagine
nPixelW = sum(sum(BW1));
if(nPixelW >= (prefRes(1)*prefRes(2))/2)
 %%Complement Image from WB to->BW
    BW1 = ~BW1;
end
%% Rimozione rumore
% utile per eliminare artefatti grafici (es: ombre)
 pxToDel = 40;        % Soglia di pixel da considerare come rumore.
BW2 = bwareaopen(BW1,pxToDel);

%%  Maschere di Dilatazione ed Erosione
% Definisco operatore morfologico: Applico maschere quadrate
mask = strel('square', 10);
% imclose() applica operatori morfologici su immagini in b/n  o greyscale
BW3 = imclose(BW2, mask);
%% Eliminazione dei "buchi" dall'immagine
BW4 = imfill(BW3, 'holes');

%%

%traces the exterior boundaries of objects, as well as boundaries of holes
```

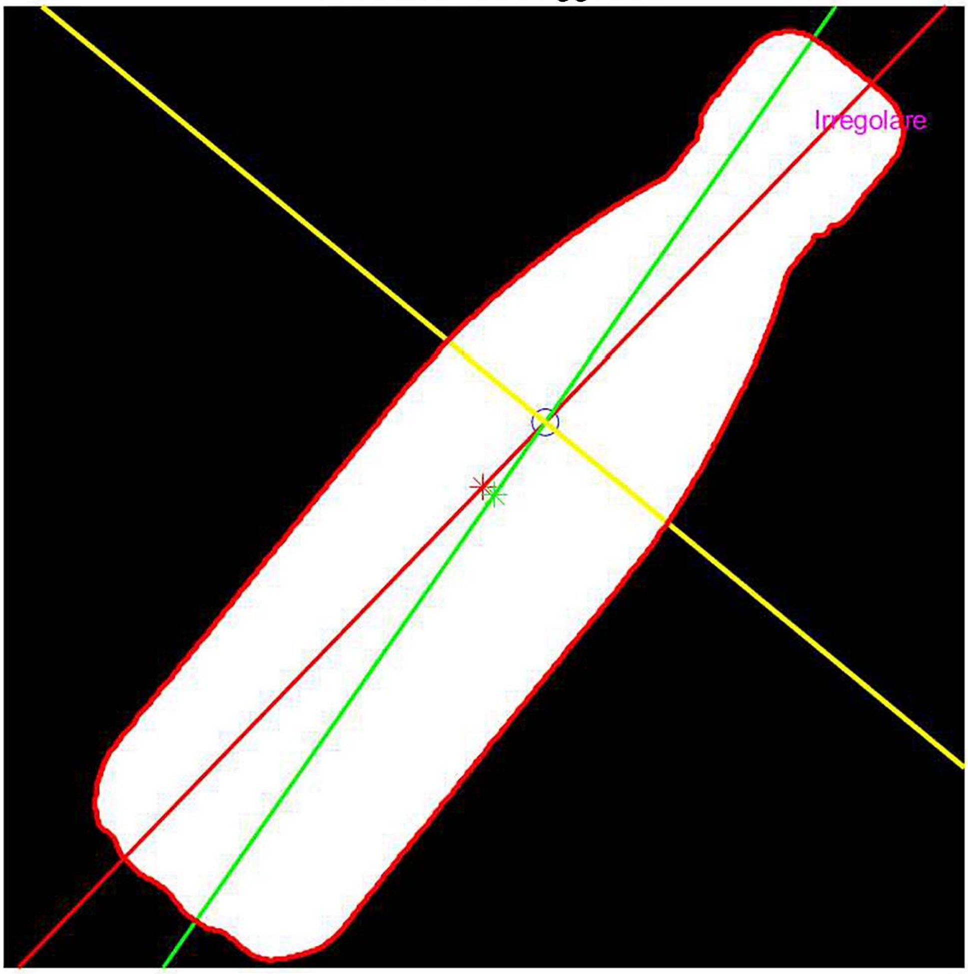
```
%inside these objects, in the binary image BW.
[B,L,n,A] = bwboundaries(BW4,'noholes');
%B:=exterior boundaries of the objects founds;
%L:= label matrix L where objects and holes are labeled.
%n:= number of object found
%A:= adiajency matrix of BW image
%regionprops returns measurements for the current identified region
stats = regionprops(L,'all');

 %%Proprieta dell'oggetto identificato:Area e Perimetro
      p=max([stats.Perimeter]);
      a=max([stats.Area]);
  %compute the roundness metric
  roundness = 4*pi*a/p^2;
  % Circolarità dell'oggetto se prossimo ad 1->cerchio
   circularity=roundness^(-1);
for k=1:size(stats,1)
    if stats(k).Area == a && stats(k).Perimeter==p

      boundary=B{k};
      apothem = a*2/p;
      epsilon=0.03;
     objShape = "Irregolare";

 fixed = [0.289,0.5,0.688,0.866];
polig = ["Triangolo", "Quadrilatero", "Pentagono", "Esagono"];
%In base all'apotema,perimetro ed area si identifica la forma dell'oggetto
for i=1:4
    numLati = i+2;
    disp(apothem/p*numLati)
    if((apothem/p*numLati) < (fixed(i)+epsilon) && (apothem/p*numLati) > (fixed(i)-↙
epsilon))
        objShape = polig(i);
        break;

    end
end
if (strcmp(objShape, "Irregolare") == 1)
    epsilon = 0.1;
   circularity=roundness^(-1);
    if( circularity< 1+epsilon && circularity > 1-epsilon )
            objShape = "Circolare";
    end
end

    end
end

%% rasformata di Radon per determinare baricentro ed orientamento
% Si calcolano le proiezioni radiali dell'oggeto , per identificare
%l'orientamento dell'oggeto e per tracciare le diagonali principali dell'oggetto
%appena riconosciuto. L'intersezione delle diagonali determina la posizione
%del baricentro
% Definiscl'intervallo in gradi su cui effettuare le proiezioni
theta = 0:179;
```

```
% Eseguo le proiezioni.
% R è la proiezione dell'oggetto, Xp è il vettore delle coordinate di ogni riga↙
dell'immagine.
[R, xp] = radon(BW4, theta);
% Determino la proiezione con altezza maggiore per determinare la diagonale
maxRadon = max(R);

%% Determino il massimo per identificare il punto più alto della
% proiezione con altezza maggiore.
% [pk, locs] = findpeaks() identifica il massimo locale del vettore dato in
% ingresso e l'indice di quel valore all'interno del vettore.
% SosrtStr specifica che i risultati andranno ordinati
% NPeaks specifica quanti massimi locali trovare nel vettore.
[pk, locs] = findpeaks(maxRadon, 'SortStr', 'descend', 'NPeaks', 2);

% Trovo angolo in radianti il cui indice corrisponde all'elemento di una
% delle colonne di R il cui valore è pari al picco individuato da pk
theta1 = locs(1);
offset1 = xp(R(:, locs(1)) == pk(1));
theta2 = locs(2);
offset2 = xp(R(:, locs(2)) == pk(2));

% Determino le diagonali
diag1(x) = tand(theta1+90) * (x - offset1*cosd(theta1)) + offset1*sind(theta1);
diag2(x) = tand(theta2+90) * (x - offset2*cosd(theta2)) + offset2*sind(theta2);

% Identifico il baricentro nel punto d'intersezione delle diagonali
x_bc = solve(diag1 == diag2);
y_bc = diag1(x_bc);

% Determino Orientamento a partire da due angoli identificati dalla
% trasformata di Radon

orient = (theta1+theta2)/2;
% Controllo se c'è discordanza tra i quadranti identificati dagli angoli
if(sign(sind(theta1)) ~= sign(sind(theta2)) || sign(cosd(theta1)) ~= sign(cosd↙
(theta2)) )
    orient = orient-90;
end
% Mi assicuro che l'angolo trovato sia tra -90° and 90°
orient = atand(sind(orient)/cosd(orient));
 %% Visualizzazione immagine pre e post processamento
montage({imgRGB,I,BW,BW1,BW2,BW3,BW4})
figure()
imshow(BW4)
title('Identificazione Oggetto');

hold on;

%% Grafico delle Diagonali
plot(center(1) + offset1*cosd(theta1), center(2) - offset1*sind(theta1), 'r*',↙
'MarkerSize', 10);
plot(center(1) + offset2*cosd(theta2), center(2) - offset2*sind(theta2), 'g*',↙
'MarkerSize', 10);
plot(center(1) + x_bc, center(2) - y_bc, 'bo', 'MarkerSize', 10);
```

```
diag1Plot = center(2) - tand(theta1+90) * (x - center(1) - offset1*cosd(theta1)) -↙
offset1*sind(theta1);
diag2Plot = center(2) - tand(theta2+90) * (x - center(1) - offset2*cosd(theta2)) -↙
offset2*sind(theta2);
lineOrient = center(2) - tand(orient) * (x - center(1) - x_bc) - y_bc;
fplot(diag1Plot, 'r', 'LineWidth', 1.5);
fplot(diag2Plot, 'g', 'LineWidth', 1.5);
fplot(lineOrient, 'y', 'LineWidth', 2);
%% Visualizzo il contorno dell'oggetto identificato
  text(boundary(1,1)+10,boundary(2,2)+10,objShape,'Color','magenta');
  plot(boundary(:,2),boundary(:,1),'red','LineWidth',2);
 txt=sprintf("Perimeter:%f\tArea:%f\nForma:%s\nRoundness:%d\nOrientation:%↙
f\nBaricentro=%d\t%d",p,a,objShape,roundness,orient,x_bc,y_bc);
  disp(txt);
```



Identificazione Oggetto




```
Perimeter:1270.038000   Area:67507.000000
Forma:Irregolare
Roundness:5.259267e-01
Orientation:-39.500000
Baricentro=31   32
```