به نام خدا

دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)

دانشکده مهندسی برق

گزارش پروژه درس مقدمه ای بر رباتیک

استاد درس : دکتر ایمان شریفی

تدریسیار : مهندس سجاد صادق نعل کنانی

محمد برآبادی

سجاد قدیری

مارال مرداد

محیا حقگو

سرژ یقیازاریان تبریزی

بهار 1401

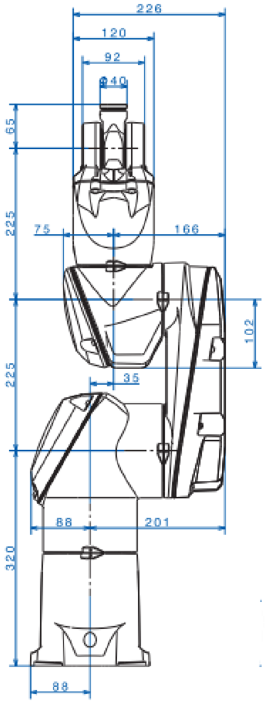
فهرست مطالب

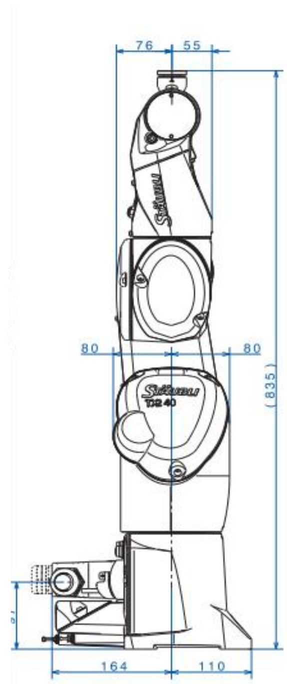
بخش اول

ربات مورد مطالعه در این پروژه، ربات 6 درجه آزادی TX2-40 از شرکت STAUBLI می‌باشد.

نمای کلی ربات به شکل زیر می‌باشد.

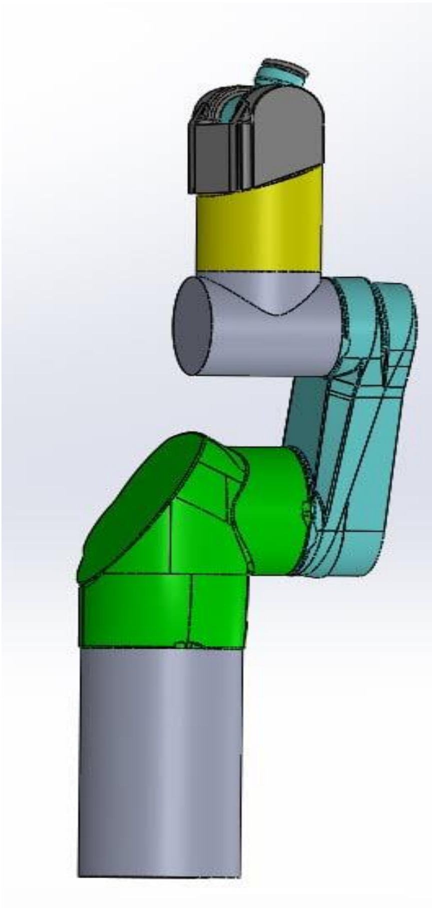
ابعاد ربات و جای‌گیری مفاصل آن در شکل‌های زیر قابل مشاهده می‌باشد

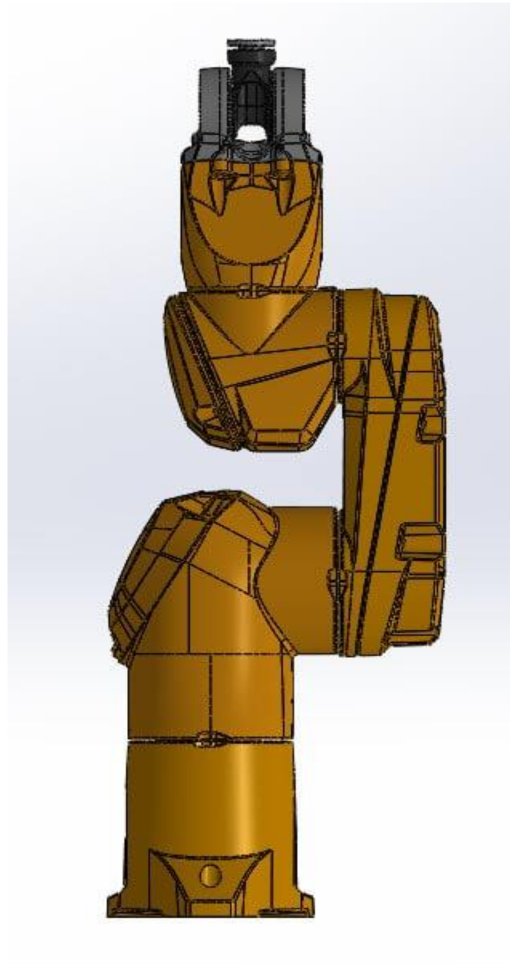




مشخصات و شرایط محیطی مناسب کاری ربات	
حداکثر بار	2 KG
بار نامی(Nominal load)	1.7 Kg
وزن ربات	29 Kg
حداکثر سرعت در مرکز ثقل بار	8.6 m/s
دمای کاری با توجه به استاندارد NF EN 60 204	5-40 C°
دمای کاری با توجه به استاندارد NF EN 60 204	30تا 95 %

طراحی ربات در نرم افزار solidwork به صورت زیر می‌باشد. شکل سمت راست، طراحی انجام شده توسط شرکت سازنده می‌باشد و طراحی شکل سمت چپ توسط این گروه انجام شده است.





ممان اینرسی‌های مورد استفاده در قسمت دینامیک، از فایل solidwork طراحی شده گروه به دست آمده و لینک آن‌ها قرار داده شده است.

[ممان اینرسی لینک اول]

[ممان اینرسی لینک دوم]

[ممان اینرسی لینک سوم]

[ممان اینرسی لینک چهارم]

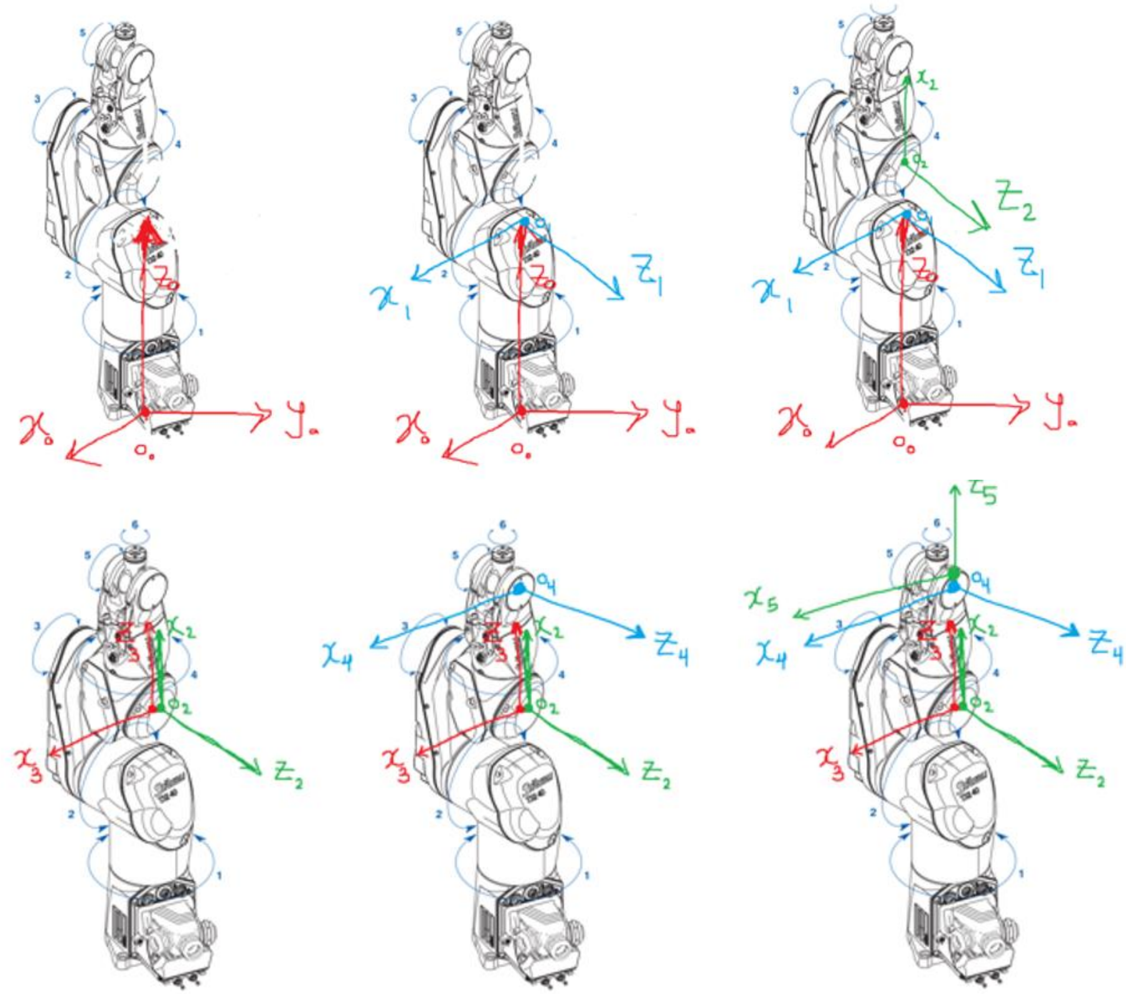
[ممان اینرسی لینک پنجم]

[ممان اینرسی لینک ششم]

بخش دوم

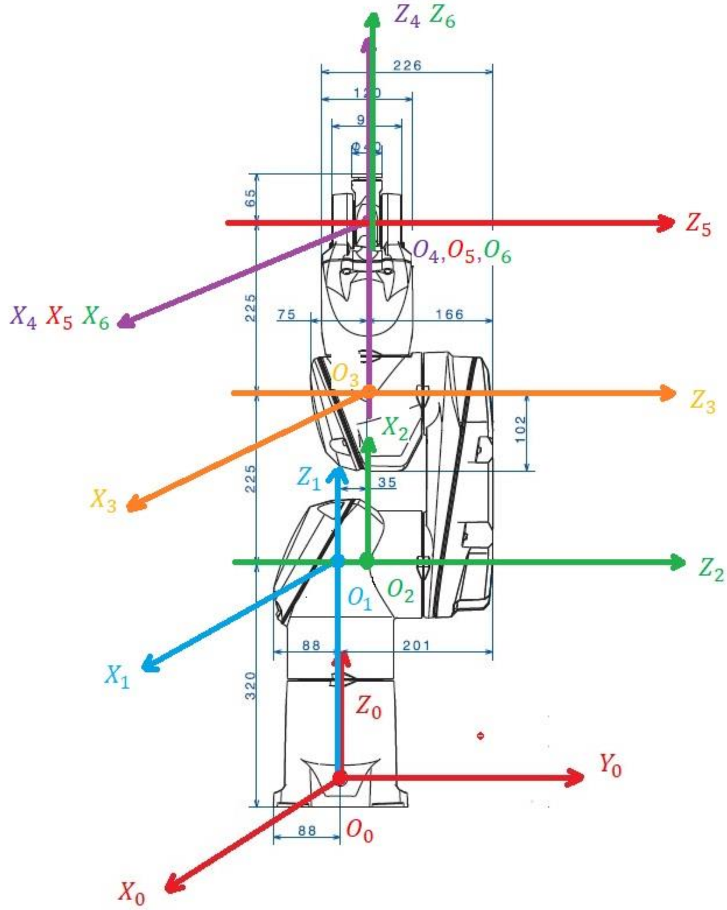
در این بخش محورگذاری‌ها به دو صورت classic DH و modified DH انجام شده است که در ادامه هردو روش آورده شده است ولی برای بخش های بعدی از پارامترهای modified DH استفاده شده است.

دناویت-هارتنبرگ به روش کلاسیک (Classic)



شماره مفصل	$\propto_{i-1}$	$\boldsymbol{a_{i-1}}$	$\boldsymbol{d_i}$	$\boldsymbol{\theta_i}$
units	[°]	[*mm*]	[*mm*]	[°]
1	-90°	0	L1	θ1
2	0	L2	-35	-90+θ2
3	90°	0	0	θ3
4	-90°	0	L4	θ4
5	90°	0	L5	θ5
6	0	0	0	θ6

دناویت-هارتنبرگ به روش اصلاح شده (Modified)



شماره مفصل	$\propto_{i-1}$	a_{i-1}	d_i	θ_i
units	[°]	[*mm*]	[*mm*]	[°]
1	0	0	P_1	θ1
2	$-\frac{\pi}{2}$	0	P_2	$-\frac{\pi}{2}+\theta 2$
3	0	P_3	0	$\frac{\pi}{2}+\theta 3$
4	$\frac{\pi}{2}$	0	P_4	θ4
5	$-\frac{\pi}{2}$	0	0	θ5
6	$\frac{\pi}{2}$	0	0	θ6

بخش سوم

برای به دست آوردن موقعیت مجری نهایی، باید ماتریس های تبدیل را محاسبه کنیم که به صورت زیر می‌باشند:

$$T_{modified} = \begin{bmatrix} c\theta_i & -s\theta_i & 0 & a_{i-1} \\ s\theta_i c\alpha_{i-1} & c\theta_i c\alpha_{i-1} & -s\alpha_{i-1} & -s\alpha_{i-1} d_i \\ s\theta_i s\alpha_{i-1} & c\theta_i s\alpha_{i-1} & c\alpha_{i-1} & c\alpha_{i-1} d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{1\_0} = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 & 0 \\ 0 & 0 & 1 & P_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad T_{2\_1} = \begin{bmatrix} \sin\theta_2 & \cos\theta_2 & 0 & 0 \\ 0 & 0 & 1 & P_2 \\ \cos\theta_2 & -\sin\theta_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{3\_2} = \begin{bmatrix} -\sin\theta_3 & -\cos\theta_3 & 0 & P_3 \\ \cos\theta_3 & -\sin\theta_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad T_{4\_3} = \begin{bmatrix} \cos\theta_4 & -\sin\theta_4 & 0 & 0 \\ 0 & 0 & -1 & P_4 \\ \sin\theta_4 & \cos\theta_4 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{5\_4} = \begin{bmatrix} \cos\theta_5 & -\sin\theta_5 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -\sin\theta_5 & -\cos\theta_5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad T_{6\_5} = \begin{bmatrix} \cos\theta_6 & -\sin\theta_6 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ \sin\theta_6 & \cos\theta_6 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{7\_6} = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & P_7 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

موقعیت مجری نهایی به صورت زیر به دست می آید:

$$T_{7\_0} = T_{1\_0} * T_{2\_1} * T_{3\_2} * T_{4\_3} * T_{5\_4} * T_{6\_5} * T_{7\_6}$$

بخش چهارم

چهارگانه یکه (Quaternion)

پارامتر های اویلر :

پارامتر های اویلر بر حسب محور معادل $\widehat{K} = \begin{bmatrix} k_x \\ k_y \\ k_z \end{bmatrix}$ و زاویه معادل θ به صورت زیر تعیین می‌شود:

$$\in_1 = k_x \sin\frac{\theta}{2}$$
$$\in_2 = k_y \sin\frac{\theta}{2}$$
$$\in_3 = k_z \sin\frac{\theta}{2}$$
$$\in_4 = \cos\frac{\theta}{2}$$

به وضوح این چهار پارامتر از همدیگر مستقل نیستند و توسط معادله زیربا یکدیگر ارتباط دارند :

$$\in_1{}^2 + \in_2{}^2 + \in_3{}^2 + \in_4{}^2 = 1$$

$$R_{\in} = \begin{bmatrix} 1-2(\in_2{}^2 + \in_3{}^2) & 2(\in_1\in_2 - \in_3\in_4) & 2(\in_1\in_3 + \in_2\in_4) \\ 2(\in_1\in_2 + \in_3\in_4) & 1-2(\in_1{}^2 + \in_3{}^2) & 2(\in_2\in_3 - \in_1\in_4) \\ 2(\in_1\in_3 - \in_2\in_4) & 2(\in_2\in_3 + \in_1\in_4) & 1-2(\in_1{}^2 + \in_2{}^2) \end{bmatrix}$$

حال با محاسبه ماتریس دوران ${}^0_6R$ و معلوم بودن آن پارامتر های اویلر معادل از روابط زیر بدست می‌آیند:

$${}^0_6R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

$$\in_1 = \frac{r_{32} - r_{23}}{4\in_4} \quad , \quad \in_2 = \frac{r_{13} - r_{31}}{4\in_4} \quad , \quad \in_3 = \frac{r_{21} - r_{12}}{4\in_4}$$

$$\in_4 = \frac{\sqrt{1 + r_{11} + r_{22} + r_{33}}}{2}$$

MATLAB Code : [Quaternion]

Results:

$$r_{11} = -S_6[C_4S_1 + C_1C_2C_3S_4 - C_1S_2S_3S_4]$$

$$- C_6[C_5S_1S_4 + C_1C_2S_3S_5 + C_1C_3S_2S_5 + C_1C_4C_5S_2S_3 - C_1C_2C_3C_4C_5]$$

```
>> pretty(simplify(T6_0(1,1)))
- sin(Th6)  (cos(Th4) sin(Th1) + cos(Th1) cos(Th2) cos(Th3) sin(Th4) - cos(Th1) sin(Th2) sin(Th3)

  sin(Th4)) - cos(Th6)  (cos(Th5) sin(Th1) sin(Th4) + cos(Th1) cos(Th2) sin(Th3) sin(Th5) + cos(Th1)

  cos(Th3) sin(Th2) sin(Th5) + cos(Th1) cos(Th4) cos(Th5) sin(Th2) sin(Th3) - cos(Th1) cos(Th2) cos(Th3)

  cos(Th4) cos(Th5))
```

$$r_{12} = S_6[C_5S_1S_4 + C_1C_2S_3S_5 + C_1C_3S_2S_5 + C_1C_4C_5S_2S_3 - C_1C_2C_3C_4C_5] - C_6[C_4S_1 + C_1C_2C_3S_4 - C_1S_2S_3S_4]$$

```
>> pretty(simplify(T6_0(1,2)))
sin(Th6)  (cos(Th5) sin(Th1) sin(Th4) + cos(Th1) cos(Th2) sin(Th3) sin(Th5) + cos(Th1) cos(Th3) sin(Th2)

  sin(Th5) + cos(Th1) cos(Th4) cos(Th5) sin(Th2) sin(Th3) - cos(Th1) cos(Th2) cos(Th3) cos(Th4) cos(Th5))

   - cos(Th6)  (cos(Th4) sin(Th1) + cos(Th1) cos(Th2) cos(Th3) sin(Th4) - cos(Th1) sin(Th2) sin(Th3)

  sin(Th4))
```

$$r_{12} = S_6[C_1C_4 - C_2C_3S_1S_4 + S_1S_2S_3S_4] - C_6[C_2S_1S_3S_5 - C_1C_5S_4 + C_3S_1S_2S_5 - C_2C_3C_4C_5S_1 + C_4C_5S_1S_2S_3]$$

```
>> pretty(simplify(T6_0(2,1)))
sin(Th6)  (cos(Th1) cos(Th4) - cos(Th2) cos(Th3) sin(Th1) sin(Th4) + sin(Th1) sin(Th2) sin(Th3) sin(Th4))

   - cos(Th6)  (cos(Th2) sin(Th1) sin(Th3) sin(Th5) - cos(Th1) cos(Th5) sin(Th4) + cos(Th3) sin(Th1)

  sin(Th2) sin(Th5) - cos(Th2) cos(Th3) cos(Th4) cos(Th5) sin(Th1) + cos(Th4) cos(Th5) sin(Th1) sin(Th2)

  sin(Th3))
```

$$r_{22} = C_6[\, C_1C_4 - C_2C_3S_1S_4 + S_1S_2S_3S_4\,] - S_6[C_2S_1S_3S_5 - C_1C_5S_4 + C_3S_1S_2S_5 - C_2C_3C_4C_5S_1 + C_4C_5S_1S_2S_3\,]$$

```
>> pretty(simplify(T6_0(2,2)))
cos(Th6) (cos(Th1) cos(Th4) - cos(Th2) cos(Th3) sin(Th1) sin(Th4) + sin(Th1) sin(Th2) sin(Th3) sin(Th4))

   + sin(Th6) (cos(Th2) sin(Th1) sin(Th3) sin(Th5) - cos(Th1) cos(Th5) sin(Th4) + cos(Th3) sin(Th1)

  sin(Th2) sin(Th5) - cos(Th2) cos(Th3) cos(Th4) cos(Th5) sin(Th1) + cos(Th4) cos(Th5) sin(Th1) sin(Th2)

  sin(Th3))
```

$$r_{13} = C_1C_2C_5S_3 - S_1S_4S_5 + C_1C_3C_5S_2 - C_1C_2C_3C_4S_5 - C_1C_4S_2S_3S_5$$

```
>> pretty(simplify(T6_0(1,3)))
cos(Th1) cos(Th2) cos(Th5) sin(Th3) - sin(Th1) sin(Th4) sin(Th5) + cos(Th1) cos(Th3) cos(Th5)

  sin(Th2) + cos(Th1) cos(Th2) cos(Th3) cos(Th4) sin(Th5) - cos(Th1) cos(Th4) sin(Th2) sin(Th3) sin(Th5)
```

$$r_{31} = S_{23}S_4S_6 - C_6\,[C_2C_3S_5 + S_2S_3S_5 - C_2C_4C_5S_3 - C_3C_4C_5S_2]$$

```
>> pretty(simplify(T6_0(3,1)))
sin(Th2 + Th3) sin(Th4) sin(Th6) - cos(Th6) (cos(Th2) cos(Th3) sin(Th5) - sin(Th2) sin(Th3)

  sin(Th5) + cos(Th2) cos(Th4) cos(Th5) sin(Th3) + cos(Th3) cos(Th4) cos(Th5) sin(Th2))
```

$$r_{23} = C_1S_4S_5 - C_2C_5S_1S_3 + C_3C_5S_1S_2 - C_2C_3C_4S_1S_5 - C_4S_1S_2S_3S_5$$

```
>> pretty(simplify(T6_0(2,3)))
cos(Th1) sin(Th4) sin(Th5) + cos(Th2) cos(Th5) sin(Th1) sin(Th3) + cos(Th3) cos(Th5) sin(Th1)

  sin(Th2) + cos(Th2) cos(Th3) cos(Th4) sin(Th1) sin(Th5) - cos(Th4) sin(Th1) sin(Th2) sin(Th3) sin(Th5)
```

$$r_{32} = S_6[C_2C_3S_5 - S_2S_3S_5 + C_2C_4C_5S_3 - C_3C_4C_5S_2] + S_{23}C_6S_4$$

```
>> pretty(simplify(T6_0(3,2)))
sin(Th6) (cos(Th2) cos(Th3) sin(Th5) - sin(Th2) sin(Th3) sin(Th5) + cos(Th2) cos(Th4) cos(Th5)

  sin(Th3) + cos(Th3) cos(Th4) cos(Th5) sin(Th2)) + sin(Th2 + Th3) cos(Th6) sin(Th4)
```

$$r_{33} = C_2C_3S_5 - C_5S_2S_3 - C_2C_4S_3S_5 - C_3C_4S_2S_5$$

```
>> pretty(simplify(T6_0(3,3)))
cos(Th2) cos(Th3) cos(Th5) - cos(Th5) sin(Th2) sin(Th3) - cos(Th2) cos(Th4) sin(Th3) sin(Th5) - cos(Th3)

  cos(Th4) sin(Th2) sin(Th5)
```

زوایای اویلری

$$R_{zyz} = R_{z,\phi}\,.\,R_{y,\theta}\,.\,R_{z,\varphi} = \begin{bmatrix} C_\emptyset & -S_\emptyset & 0 \\ S_\emptyset & C_\emptyset & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} C_\theta & 0 & S_\theta \\ 0 & 1 & 0 \\ -S_\theta & 0 & C_\theta \end{bmatrix}\begin{bmatrix} C_\varphi & -S_\varphi & 0 \\ S_\varphi & C_\varphi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R_{zyz} = \begin{bmatrix} C_\emptyset C_\theta C_\varphi - S_\emptyset C_\varphi & -C_\emptyset C_\theta S_\varphi - S_\emptyset C_\varphi & C_\emptyset S_\theta \\ S_\emptyset C_\theta C_\varphi + C_\emptyset S_\varphi & -S_\emptyset C_\theta S_\varphi + C_\emptyset C_\varphi & S_\emptyset S_\theta \\ S_\theta C_\varphi & S_\theta S_\varphi & C_\theta \end{bmatrix}$$

زوایای ثابت **X–Y–Z** :

$$ {}^{A}_{B}R_{XYZ}\,(\gamma\,,\beta\,,\alpha) = \; R_Z(\alpha)\;\; R_Y(\beta)\; R_X(\gamma) $$

$$ = \begin{bmatrix} C_\alpha & -S_\alpha & 0 \\ S_\alpha & C_\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} C_\beta & 0 & S_\beta \\ 0 & 1 & 0 \\ -S_\beta & 0 & C_\beta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & C_\gamma & -S_\gamma \\ 0 & S_\gamma & C_\gamma \end{bmatrix} $$

تا زمانی که $C_\beta \neq 0$ می‌توان α و γ را با گرفتن $Atan2$ محاسبه کنیم :

$$ \beta = Atan2(-r_{31}\,,\sqrt{(r_{11})^2 + (r_{21})^2}) $$

$$ \alpha = Atan2(\frac{r_{31}}{C_\beta}\,,\frac{r_{11}}{C_\beta}) $$

$$ \gamma = Atan2(\frac{r_{32}}{C_\beta}\,,\frac{r_{33}}{C_\beta}) $$

چنانچه $\beta = \pm 90$ باشد ، جواب بدست آمده از روابط بالا بی معنی خواهد شد. در این صورت تنها می‌توان تفاضل α و γ را محاسبه کرد. در این حالت فرض می‌کنیم $\alpha = 0$ و به دو حالت زیر تقسیم می‌کنیم :

1)

$$ \beta = +90 $$

$$ \alpha = 0 $$

$$ \gamma = Atan2(r_{12}\,,r_{22}) $$

2)

$$ \beta = -90 $$

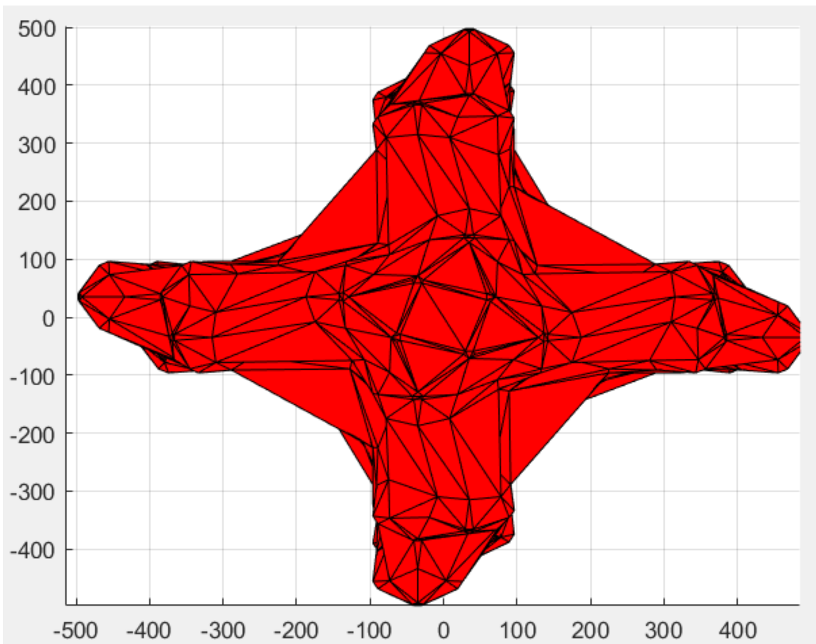
$$ \alpha = 0 $$

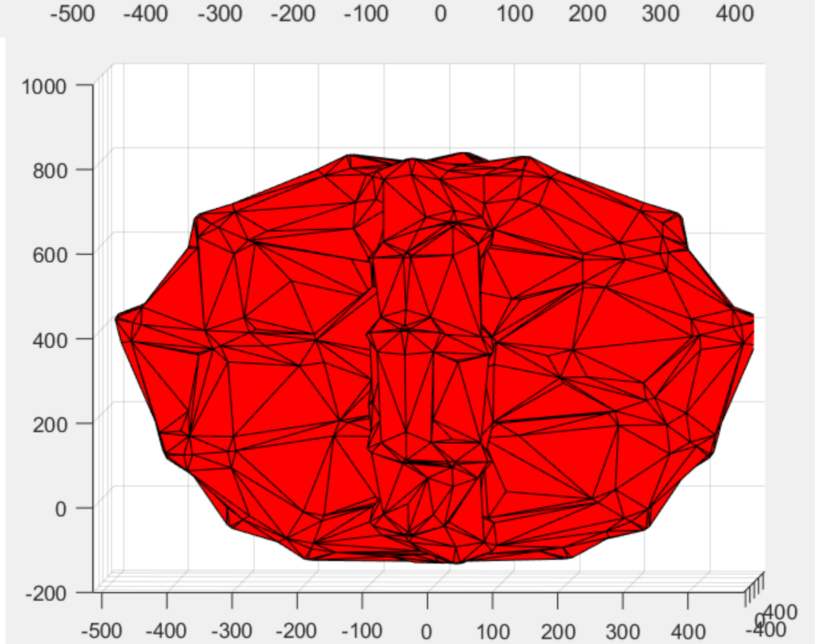
$$ \gamma = -\,Atan2(r_{12}\,,r_{22}) $$

بخش پنجم

این بخش با توجه به بازه های محورها که در کاتالوگ ذکر شده انجام شده است. لازم به ذکر است که این زوایا بر اساس بازه های به دست آمده در جدول DH تغییر داده شده اند.

فضای کاری ربات مورد مطالعه به شکل زیر است.





انیمیشن ربات نیز در کنار دیگر فایل‌های مربوط به پروژه قرار داده شده است.

❋ قابل ذکر است که فضای کاری ربات ما در فضای سه بعدی یک کره است ولی برای نمایش کامل و رسم آن در متلب نیازمند یک کامپیوتر بسیار قوی بودیم تا بازه های متناظر با زوایای هر مفصل را به تعداد بیشتری

تقسیم کنیم . شکل فوق نتیجه رسم با تعداد 5^6حالت مختلف است. در واقع هریک از بازه های مدنظر برای مفاصل را که از کاتالوگ و با توجه به جدول DH بدست آمده بود را به 5 قسمت تقسیم کردیم.

بخش ششم

سینماتیک معکوس

در رهیافت هندسی حل معادلات سینماتیکی بازوی مکانیکی ماهر سعی می شود هندسه فضایی به چند مسئله هندسه مسطحه تجزیه شود.مطابق شکل رسم شده زیر موقعیت O_1 همواره ثابت بوده و با دوران θ_1 حول محور z_1 نسبت به صفحه x_0y_0 در موقعیت $(0,0, L_1)$ می‌باشد.

در ادامه موقعیت O_2 نسبت به صفحه $x_0y_0z_0$ تابعی از دوران θ_1 حول محور z_1 است بنابراین مطابق آنچه که نوشته شده است مولفه z موقعیت O_2 نسبت به صفحه x_0y_0 همواره ثابت و برابر L1 است ولی مولفه x وy در این صفحه تابع θ_1 بوده و به ترتیب برابر $-L_2\sin\theta_1$ و $L_2 cos\,\theta_1$ ، بنابراین موقعیت O_2 نسبت به صفحه x_0y_0 برابر $(-L_2\sin\theta_1, L_2 cos\,\theta_1, L_1)$ می‌باشد.

حال به بررسی موقعیت O_3 که تابعی از دوران θ_1و θ_2 به ترتیب حول محور z_1 و z_2 می‌پردازیم.

ابتدا موقعیت بازو را تحت دوران θ_2 حول محور z_2 نسبت به صفحه x_1y_1 بدست می آوریم که ارتفاع آن برابر $L_3cos\theta_2$ و تصویر طول L_3 نسبت به این صفحه برابر $L_3sin\theta_2$ می‌باشد.

حال در ادامه مطابق روند بدست آوردن موقعیت O_2 نسبت به صفحه x_0y_0 ، موقعیت تصویر O_3 *در صفحه* x_1y_1 *را نسبت به صفحه* $x_0O_0y_0$ که مولفه z آن همواره ثابت و برابر L3cosθ_2+L_1 بوده بدست می آوریم.

که تحت دوران θ_1 حول محور z_1 ، موقعیت O_3 نسبت به صفحه $x_0O_0y_0$ برابر عبارت زیر می‌باشد:

$$([L_3\sin\theta_2]\, cos\,\theta_1 - L_2 sin\,\theta_1, [L_3\sin\theta_2]\, sin\,\theta_1 + L_2 cos\,\theta_1, L_3\cos\theta_2 + L_1\ )$$

به همین ترتیب موقعیت O_4 نسبت به صفحه $x_0O_0y_0$ برابر مختصات زیر خواهد بود:

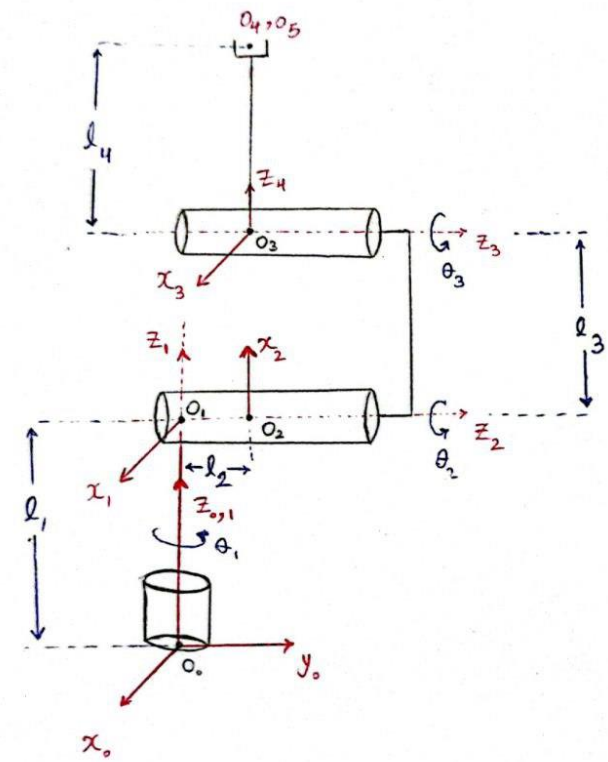
$$X_4 = [L_3\sin\theta_2 + L_4\sin\,(\theta_2+\theta_3)]\, cos\,\theta_1 - L_2 sin\,\theta_1$$

$$Y_4 = [L_3\sin\theta_2 + L_4\sin\,(\theta_2+\theta_3)]\, sin\,\theta_1 + L_2 cos\,\theta_1$$

$$Z_4 = L_1 + L_4\cos(\theta_2+\theta_3) +\ L_3\cos(\theta_2)$$

همچنین محاسبات دستی در تصویر زیر قابل مشاهده است:

سینماتیک معکوس:



$l_1 = 320$

$l_2 = 35$

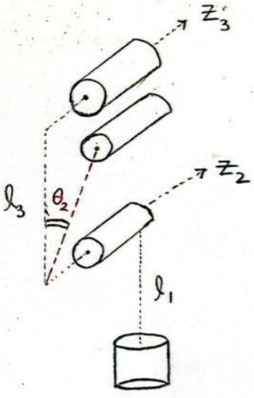
$l_3 = 225$

$l_4 = 225$

$$O_1 \rightarrow \begin{cases} x_1 = 0 \\ y_1 = 0 \\ Z_1 = l_1 \end{cases}$$

$$O_2 \rightarrow \begin{cases} x_2 = -l_2 \sin\theta_1 \\ y_2 = l_2 \cos\theta_1 \\ Z_2 = l_1 \end{cases}$$

$$O_3 \rightarrow \begin{cases} x_3 = \left[l_3 \sin\theta_2\right]\cos\theta_1 - l_2 \sin\theta_1 \\ y_3 = \left[l_3 \sin\theta_2\right]\sin\theta_1 + l_2 \cos\theta_1 \\ Z_3 = l_1 + l_3 \cos\theta_2 \end{cases}$$



$$O_4 \rightarrow \begin{cases} x_4 = \left[l_3 \sin\theta_2 + l_4 \left[\overbrace{\cos\theta_2 \sin\theta_3 + \cos\theta_3 \sin\theta_2}^{\sin(\theta_2+\theta_3)}\right]\right]\cos\theta_1 - l_2 \sin\theta_1 \\ y_4 = \left[l_3 \sin\theta_2 + l_4 \left[\overbrace{\cos\theta_2 \sin\theta_3 + \cos\theta_3 \sin\theta_2}^{\sin(\theta_2+\theta_3)}\right]\right]\sin\theta_1 + l_2 \cos\theta_1 \\ Z_4 = l_1 + l_4 \cos(\theta_2+\theta_3) + l_3 \cos\theta_2 \end{cases}$$

کد مربوط به این بخش را بطور کامل می‌توانید از لینک زیر مشاهده کنید :

MATLAB Code: [Geometric inverse]

در قسمت های بالاتر از این بخش از کد، ابتدا پارامتر های مورد نیاز و ماتریس های تبدیل را تعریف کردیم. سپس با برای مثال یک موقعیت برای مجری نهایی درنظرگرفتیم و سپس با استفاده از ماتریس تبدیل T4_0 را برابر قرار دادن نقطه Intersect به عنوان موقعیت مجری نهایی سه معادله تشکیل دادیم و سپس آن ها را حل کردیم.

سپس مقادیر بدست آمده را در بازه منطقی کاتالوگ در متغییر R3_0 با دستور *Subs* جایگذاری می‌کنیم.

```
%% inverse kinematic
%Find Three Angles(theta1, theta2, theta3) from origin of joint4
eq1 = T4_0(1, 4) == Xee ;
eq2 = T4_0(2, 4) == Yee ;
eq3 = T4_0(3, 4) == Zee ;
[angle1, angle2, angle3] = vpasolve([eq1, eq2, eq3],[theta1, theta2, theta3]);
%angle1 = vpa(mod(mod(angle1, 2 * vpa(pi)), 2 * vpa(pi)) * 180 / pi);
%angle2 = vpa(mod(mod(angle2, 2 * vpa(pi)), 2 * vpa(pi)) * 180 / pi);
%angle3 = vpa(mod(mod(angle3, 2 * vpa(pi)), 2 * vpa(pi)) * 180 / pi);

T6_3 = T4_3 * T5_4 * T6_5;
R6_3 = T6_3(1:3,1:3);

R3_0 = T3_0(1:3, 1:3);
R3_0 = subs(R3_0,'theta1', mod(mod(angle1, vpa(pi)), vpa(pi)));
R3_0 = subs(R3_0,'theta2', mod(mod(angle2, 125 / 180 * vpa(pi)), 125 / 180 * vpa(pi)));
R3_0 = subs(R3_0,'theta3', mod(mod(angle3, 138 / 180 * vpa(pi)), 138 / 180 * vpa(pi)));
R3_0 = vpa(R3_0);
```

همچنین سه زاویه آخر را نیز از طریق کد زیر بدست آوردیم:

```
%Euler's matrix
R_z_phi =[cos(phi) -sin(phi) 0;sin(phi) cos(phi) 0;0 0 1];
R_y_theta = [cos(theta) 0 sin(theta); 0 1 0;-sin(theta) 0 cos(theta)];
R_z_psi = [cos(psi) -sin(psi) 0;sin(psi) cos(psi) 0;0 0 1];
R_zyz = inv(R3_0) * (R_z_phi * R_y_theta * R_z_psi);

theta5 = solve(R_zyz(2,3) == R6_3(2,3));
R6_3 = subs(R6_3,'theta5' ,vpa(theta5(1,1)));
theta6 = solve(R_zyz(2,2) == R6_3(2,2));
R6_3 = subs(R6_3,'theta6' ,vpa(theta6(1,1)));
theta4 = solve(R_zyz(1,3) == R6_3(1,3));
R6_3 = subs(R6_3,'theta4' ,vpa(theta4(1,1)));
%angle4 = vpa(theta4(1,1) * 180 / pi);
%angle5 = vpa(theta5(1,1) * 180 / pi);
%angle6 = vpa(theta6(1,1) * 180 / pi);
```

به این صورت که ابتدا از همان ماتریس R6_3 که بیانگر آن است که از نقطه intersect به end effector چه چرخشی صورت گیرد تا جهت گیری مجری نهایی دقیقا همان جهت گیری مطلوب باشد و همچنین با استفاده از روش اویلر سه معادله برای سه زاویه آخر بدست آوردیم. همچنین مقادیر R3_0 را نیز در ماتریس های اویلر به عنوان پارامتر های معلوم جایگذاری کردیم تا 3 معادله و 6 مجهول به 3 معادله و 3 مجهول تبدیل شود.

سپس با استفاده از دستور solve سه معادله فوق را حل کردیم .

*روش دیگری که برای محاسبه سینماتیک معکوس طی کردیم ، روش تحلیلی مطرح شده در کتاب craig می‌باشد که برای ربات ما با در نظر گرفتن زوایا به صورت real نیز جواب موهومی می‌داد و به سرانجام نرسید.

کد آن را می‌توانید از طریق زیر ببینید:

MATLAB Code: inverse craig

بخش هفتم

برای دسترسی به سرعت مفاصل لازم است ماتریس ژاکوبین برای هر یک از آن‌ها تشکیل شود. در این قسمت برای محاسبه سرعت خطی و زاویه ای مجری نهایی از دو روش alternative و محاسبه مستقیم مشتق استفاده شده است.

در روشalternative از روابط مخصوص این روش برای پارامتر های دناویت هارتنبرگ اصلاح شده استفاده شده است.

روش alternative

ماتریس ژاکوبین که متشکل از J_v و J_ω است و یک ماتریس با ابعاد 6×1 است، با توجه به نوع مفاصل که revolute است به شکل زیر به دست می‌آید:

$$J = \begin{bmatrix} J_v \\ J_\omega \end{bmatrix} \begin{bmatrix} z_1 \times (o_7 - o_1) & z_2 \times (o_7 - o_2) & z_3 \times (o_7 - o_3) & z_4 \times (o_7 - o_4) & z_5 \times (o_7 - o_5) & z_6 \times (o_7 - o_6) \\ z_1 & z_2 & z_3 & z_4 & z_5 & z_6 \end{bmatrix}$$

که در کد متلب به شکل زیر پیاده سازی شده است:

```
jacobian_endeffector = [cross(z1, o7-o1) cross(z2, o7-o2) cross(z3, o7-o3) cross(z4, o7-o4) cross(z5, o7-o5) cross(z6, o7-o6); z1 z2 z3 z4 z5 z6];
```

روش مشتق گیری مستقیم

برای پیاده سازی این روش در متلب متغیرهای مفصلی که در این پروژه همان زوایای مفصلی هستند بر حسب زمان تعریف شده اند.(مانند شکل زیر)

```
syms theta1(t) theta2(t) theta3(t) theta4(t) theta5(t) theta6(t) real;
```

برای دستیابی به ژاکوبین سرعت خطی از موقعیت مجری نهایی نسبت به زمان مشتق گرفته شده است و سپس حاصل این مشتق به صورت ضرب یک ماتریس با ابعاد 3×6 در مشتق زوایای مفصلی نسبت به زمان نوشته شده است. ماتریس J_v به این صورت به دست آمده که از سرعت خطی مجری نهایی نسبت به مشتق متغیرهای مفصلی مشتق گرفته شده است. این کار برای نوشتن سرعت مجری نهایی به شکل ضرب ماتریس J_v در ماتریس مشتق متغیرهای مفصلی انجام شده است.

```
o7 = T7_0(1:3, 4); %% position of end effector accordng to frame 0
v7 = diff(o7, t);
theta = [theta1(t) theta2(t) theta3(t) theta4(t) theta5(t) theta6(t)];
Jv = sym(zeros(3, 6));
%  decoupling coefficients of q1_dot ... q6_dot
for i=1:3
    for j=1:6
        Jv(i, j) = diff(v7(i), diff(theta(j), t));
    end
end
```

براس دست‌یابی به ژاکوبین سرعت زاویه‌ای نیز از ماتریس R مجری نهایی نسبت به زمان مشتق گرفته شده است. و با در نظر گرفتن اینکه $s = \dot{R} \times R^T$ یک ماتریس متقارن به شکل زیر است،

$$s\left(\begin{bmatrix}\omega_x\\ \omega_y\\ \omega_z\end{bmatrix}\right) = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}$$

سرعت زاویه‌ای مفصل نهایی و از طریق نوشتن آن بر اساس ضرب یک ماتریس با ابعاد 3×6 در مشتق متغیرهای زاویه ای ژاکوبین سرعت زاویه‌ای نیز به دست می‌آید.

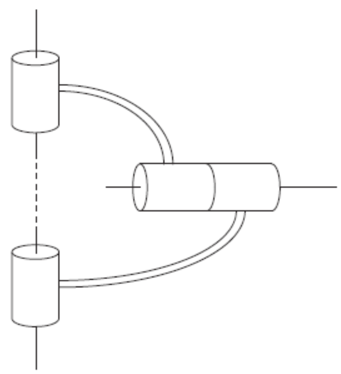
```
R7 = T7_0(1:3, 1:3);
R7_dot = diff(R7, t);
s = R7_dot*transpose(R7);
w = [s(3,2) s(1,3) s(2,1)]';

 theta = [theta1(t) theta2(t) theta3(t) theta4(t) theta5(t) theta6(t)];
 Jw = sym(zeros(3, 6));
%  decoupling coefficients of q1_dot ... q6_dot
 for i=1:3
     for j=1:6
         Jw(i, j) = diff(w(i), diff(theta(j), t));
      end
 end
```

در نهایت ژاکوبین به دست آمده از هردو روش را از هم تفریق کردیم و مشاهده شد که حاصل صفر می‌شود.

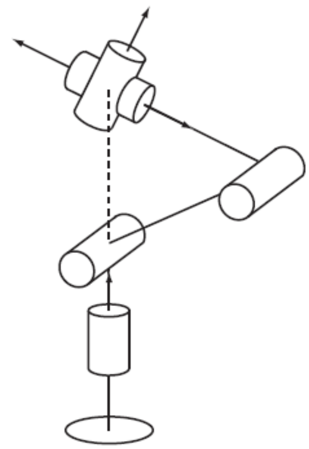
```
J_direct = [Jv; Jw];

Th1 = theta1(t); Th2 = theta2(t); Th3 = theta3(t); Th4 = theta4(t); Th5 = theta5(t); Th6 = theta6(t);
J_alternative = subs(jacobian_endeffector);
% define differences of 2 methods jacobians
J_difference = J_direct - J_alternative;
```

در این بخش از کد برای امکان پذیر بودن تفریق ژاکوبین محاسبه شده، متغیرهای سیمبولیکی که برای روش مشتق گیری مستقیم تعریف شدند را با دستور subs در متغیرهای در نظر گرفته شده برای محاسبه ژاکوبین از روش استفاده از کراس، جایگذاری کردیم. مشاهده می‌شود که حاصل تفاضل آن‌ها با دستور simplify با حدودا 50 گام صفر می‌شود.

Two collinear Revolute joint Axes



Four Revolute Joint Axes Intersecting at a Common Point



ربات مورد مطالعه ما دارای نقاط سینگولار بسیاری است. در واقع یکی از حالت های معروف سینگولاریتی حالتی است به نام *Two collinear Revolute joint Axes* که همانطور که در شکل می‌بینید و در قسمت های قبل برای ربات خود تحلیل کردیم سه درجه آخر ربات ما در این حالت قرار دارد.

یکی دیگر از حالت هایی که ممکن است ربات را سینگولار کند شکل سمت راست می‌باشد ولی ربات ما این نوع سینگولاریتی را ندارد زیرا اگر به نحوه محورگذاری در قسمت های قبل دقت کنید مفصل اول (z_1) و مفصل چهارم (z_4) در یک راستا قرار ندارند و یک مقدار ثابت 35 میلیمتر در راستای محور (y_0) فاصله دارند.

یکی دیگر از حالت های سینگولار ربات ما حالت zero configuration آن است زیرا اگر مفصل شماره یک را به اندازه دلخواه دوران دهید همچنان موقعیت مجری نهایی در همان حالی که بوده است باقی می‌ماند و هیچ تغییری در موقعیت و جهت گیری آن رخ نمی‌دهد.

❋ همچنین برای بدست آوردن نقاط سینگولار از برابر قرار دادن دترمینان ماتریس ژاکوبین با صفر و حل معادله بدست آمده در متلب نیز، همانطور که انتظار داشتیم به جواب نرسیدیم.

بخش نهم

برای به دست آوردن دینامیک ربات با روش اویلر-لاگرانژ، معادلات دینامیکی به صورت زیر نوشته می‌شود:

$$D(q) * \ddot{q} + C(q, \dot{q}) * \dot{q} + G(q) = \tau$$

طبق معادله بالا باید ماتریس‌های D و C و G محاسبه شوند.

ماتریس D به صورت زیر به دست می‌آید:

$$D(q) = \sum_{i=1}^{n} (m_i * J_{v_i}(q)^T * J_{v_i}(q) + J_{w_i}(q)^T * R_i(q) * I_i * R_i(q)^T * J_{w_i}(q)$$

ماتریس اینرسی در دستگاه مختصات پایه طبق رابطه زیر به دست می‌آید:

$$\mathrm{I} = R_i(q) * I_i * R_i(q)^T$$

که در آن R ماتریس دوران و I ماتریس اینرسی در مختصات بدنه می‌باشد.

ماتریس های اینرسی در دستگاه مختصات مرجع با استفاده از طراحی ربات در نرم‌افزار Solidwork به دست آمد. ماتریس های ژاکوبین (J_v و J_W) نیز در بخش هفتم محاسبه شدند.

ماتریس D در متلب به صورت زیر نوشته شده است:

```
%%D
for i=1:6
    D = D + (m(i)*transpose(Jv(:, ct_j:ct_j+5))*Jv(:, ct_j:ct_j+5)+transpose(Jw(:, ct_j:ct_j+5))*I(:,ct_i:ct_i+2)*Jw(:, ct_j:ct_j+5));
    ct_i = ct_i + 3;
    ct_j = ct_j + 6;
end
```

* قابل ذکر است در این کد کلیه ماتریس‌های ژاکوبین سرعت خطی در یک ماتریس J_v،کلیه ماتریس‌های ژاکوبین سرعت زاویه‌ای در یک ماتریس J_W و کلیه ماتریس‌های اینرسی در دستگاه مختصات مرجع در یک ماتریس I ریخته شده و ماتریس های مربوط به هر مفصل توسط شمارنده ct مربوطه مشخص می‌شوند.

ماتریس C به صورت زیر به دست می‌آید:

$$C_{ijk} = \frac{1}{2}\left\{\frac{\partial d_{kj}}{\partial q_i} + \frac{\partial d_{ki}}{\partial q_j} - \frac{\partial d_{ij}}{\partial q_k}\right\}$$

در این رابطه i شماره سطر و k شماره ستون می‌باشد.

ماتریس C در متلب به صورت زیر نوشته شده است:

```
%%C
C = sym(zeros(6, 6));
sum = 0;
for k=1:6
    for i=1:6
        for j=1:6
%          C(i, j+ct) = 1/2*(diff(D(k,j), Theta(i))+diff(D(k,i), Theta(j))-diff(D(i,j),Theta(k)));
         sum = sum + 1/2*(diff(D(k,j), Theta(i))+diff(D(k,i), Theta(j))-diff(D(i,j),Theta(k)));
        end
        C(k, i) = sum;
        sum = 0;
    end
end
```

برای محاسبه ماتریس G به صورت زیر عمل می‌شود:

$$P = m_i * g * h_i$$

$$G = \begin{bmatrix} \frac{\partial P}{\partial q_1} \\ \frac{\partial P}{\partial q_2} \\ \cdot \\ \cdot \\ \cdot \end{bmatrix}$$

که در این رابطه h_i ارتفاع مرکز جرم هر لینک از زمین می‌باشد.

h_i ها به صورت زیر در متلب محاسبه شده‌اند:

```
h1 = P(1)/2;
h2 = P(1);
h3 = P(1)+P(3)*cos(Theta(2)) ;
h4 = P(1)+(P(3)+(P(4)/2)*cos(Theta(3))*cos(Theta(2)));
h5 = P(1)+(P(3)+(P(4)*cos(Theta(3)))*cos(Theta(2)));
h6 = P(1)+(P(3)+((P(4)+P(6)*cos(Theta(5)))*cos(Theta(3)))*cos(Theta(2)));
h = [h1 h2 h3 h4 h5 h6];
```

در نهایت محاسبه ماتریس G در متلب به صورت زیر انجام شده‌است:

```
P = 0;
g = 9.8;
for i=1:6
    P = P+m(i)*g*h(i);
end

phi_1 = diff(P, Theta(1));
phi_2 = diff(P, Theta(2));
phi_3 = diff(P, Theta(3));
phi_4 = diff(P, Theta(4));
phi_5 = diff(P, Theta(5));
phi_6 = diff(P, Theta(6));
G = [phi_1 phi_2 phi_3 phi_4 phi_5 phi_6]';
```

MATLAB Code: [Dynamic]