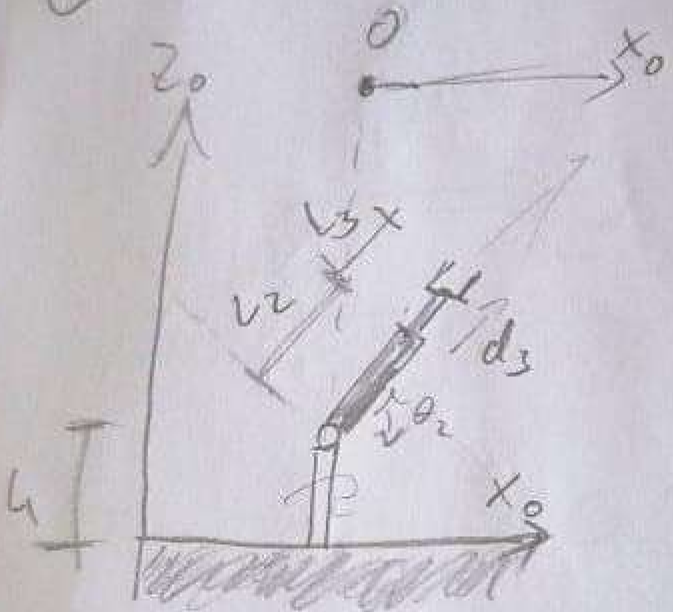
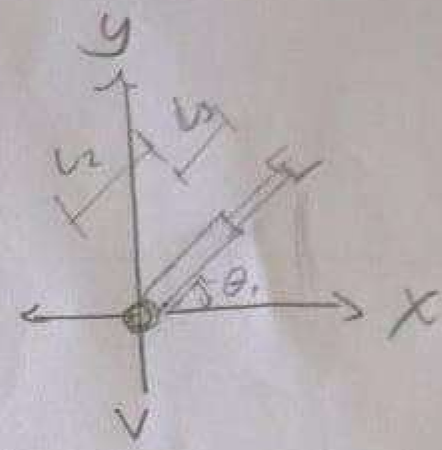
(2)





$$x = (L_2 + d_3)\cos\theta_2 \cos\theta_1 = (L_2 + d_3) C\theta_2 C\theta_1$$

$$y = (L_2 + d_3)\cos\theta_2 \sin\theta_1 = (L_2 + d_3) C\theta_2 S\theta_1$$

$$z = L_1 + (L_2 + d_3)\sin\theta_2 = L_1 + (L_2 + d_3) S\theta_2$$

```
clear, clc
syms q1 q2 q3 px py pz

L1 = 12
L2 = L1

EqX = px == (L2+q3)*cos(q2)*cos(q1)
EqY = py == (L2+q3)*cos(q2)*sin(q1)
EqZ = pz == L1 + (L2 + q3)*sin(q2)

[Sol_q1, Sol_q2, Sol_q3] = solve([EqX EqY EqZ], [q1 q2 q3])%, param, cond] ,
 "ReturnConditions",true);
%disp(Sol_q3)

Sol_q1 = simplify(Sol_q1)
Sol_q2 = simplify(Sol_q2)
Sol_q3 = simplify(Sol_q3)
```

```
L1 =

    12


L2 =

    12


EqX =

px == cos(q1)*cos(q2)*(q3 + 12)


EqY =

py == cos(q2)*sin(q1)*(q3 + 12)


EqZ =

pz == sin(q2)*(q3 + 12) + 12

Warning: Solutions are only valid under certain conditions. To include
parameters and conditions in the solution, specify the 'ReturnConditions'
 value
as 'true'.

Sol_q1 =

-2*atan((px - (12*((((px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2) - pz + 12)*(pz +
 (px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2) - 12))^(1/2) + (px^2 + py^2 + pz^2
```

```
 - 24*pz + 144)^(1/2)))/(pz - 12) - (px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2)
 + (pz*((((px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2) - pz + 12)*(pz + (px^2 +
 py^2 + pz^2 - 24*pz + 144)^(1/2) - 12))^(1/2) + (px^2 + py^2 + pz^2 - 24*pz +
 144)^(1/2)))/(pz - 12))/py)
-2*atan((px + (12*((((px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2) - pz + 12)*(pz +
 (px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2) - 12))^(1/2) + (px^2 + py^2 + pz^2
 - 24*pz + 144)^(1/2)))/(pz - 12) + (px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2)
 - (pz*((((px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2) - pz + 12)*(pz + (px^2 +
 py^2 + pz^2 - 24*pz + 144)^(1/2) - 12))^(1/2) + (px^2 + py^2 + pz^2 - 24*pz +
 144)^(1/2)))/(pz - 12))/py)
-2*atan((px - (12*((((px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2) - pz + 12)*(pz +
 (px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2) - 12))^(1/2) - (px^2 + py^2 + pz^2
 - 24*pz + 144)^(1/2)))/(pz - 12) + (px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2)
 + (pz*((((px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2) - pz + 12)*(pz + (px^2 +
 py^2 + pz^2 - 24*pz + 144)^(1/2) - 12))^(1/2) - (px^2 + py^2 + pz^2 - 24*pz +
 144)^(1/2)))/(pz - 12))/py)
-2*atan((px + (12*((((px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2) - pz + 12)*(pz +
 (px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2) - 12))^(1/2) - (px^2 + py^2 + pz^2
 - 24*pz + 144)^(1/2)))/(pz - 12) - (px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2)
 - (pz*((((px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2) - pz + 12)*(pz + (px^2 +
 py^2 + pz^2 - 24*pz + 144)^(1/2) - 12))^(1/2) - (px^2 + py^2 + pz^2 - 24*pz +
 144)^(1/2)))/(pz - 12))/py)


Sol_q2 =

 2*atan(((((px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2) - pz + 12)*(pz + (px^2 +
 py^2 + pz^2 - 24*pz + 144)^(1/2) - 12))^(1/2) + (px^2 + py^2 + pz^2 - 24*pz +
 144)^(1/2))/(pz - 12))
-2*atan(((((px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2) - pz + 12)*(pz + (px^2 +
 py^2 + pz^2 - 24*pz + 144)^(1/2) - 12))^(1/2) + (px^2 + py^2 + pz^2 - 24*pz +
 144)^(1/2))/(pz - 12))
 2*atan(((((px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2) - pz + 12)*(pz + (px^2 +
 py^2 + pz^2 - 24*pz + 144)^(1/2) - 12))^(1/2) - (px^2 + py^2 + pz^2 - 24*pz +
 144)^(1/2))/(pz - 12))
-2*atan(((((px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2) - pz + 12)*(pz + (px^2 +
 py^2 + pz^2 - 24*pz + 144)^(1/2) - 12))^(1/2) - (px^2 + py^2 + pz^2 - 24*pz +
 144)^(1/2))/(pz - 12))


Sol_q3 =

  (px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2) - 12
- (px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2) - 12
- (px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2) - 12
  (px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2) - 12


Sol_q1 =

-2*atan((px + (px^2 + py^2)^(1/2))/py)
-2*atan((px - (px^2 + py^2)^(1/2))/py)
-2*atan((px + (px^2 + py^2)^(1/2))/py)
-2*atan((px - (px^2 + py^2)^(1/2))/py)
```

```
Sol_q2 =

 2*atan(((px^2 + py^2)^(1/2) + (px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2))/(pz -
 12))
-2*atan(((px^2 + py^2)^(1/2) + (px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2))/(pz -
 12))
 2*atan(((px^2 + py^2)^(1/2) - (px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2))/(pz -
 12))
-2*atan(((px^2 + py^2)^(1/2) - (px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2))/(pz -
 12))


Sol_q3 =

  (px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2) - 12
- (px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2) - 12
- (px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2) - 12
  (px^2 + py^2 + pz^2 - 24*pz + 144)^(1/2) - 12
```

Published with MATLAB® R2022b

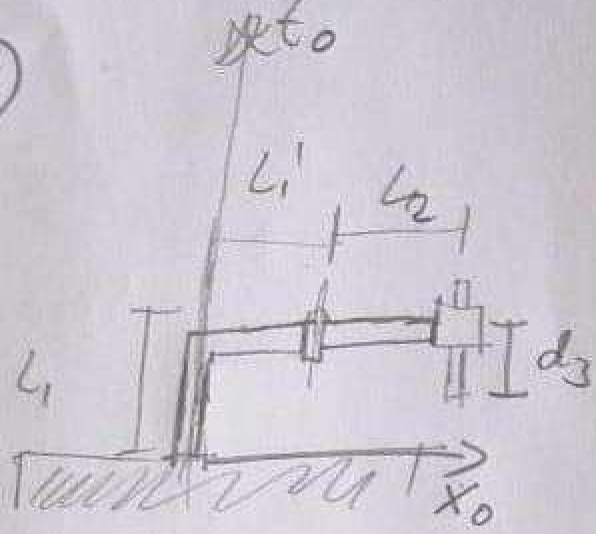
```
syms q1 q2 q3 px py pz

L1 = 12
L2 = L1

EqX = px == L1*cos(q1) + L2*cos(q1+q2)
EqY = py == L1*sin(q1) + L2*sin(q1+q2)
EqZ = pz == L1 - q3

[Sol_q1 Sol_q2 Sol_q3] = solve([EqX EqY EqZ], [q1 q2 q3]);

Sol_q1 = simplify(Sol_q1)
Sol_q2 = simplify(Sol_q2)
Sol_q3 = simplify(Sol_q3)
```

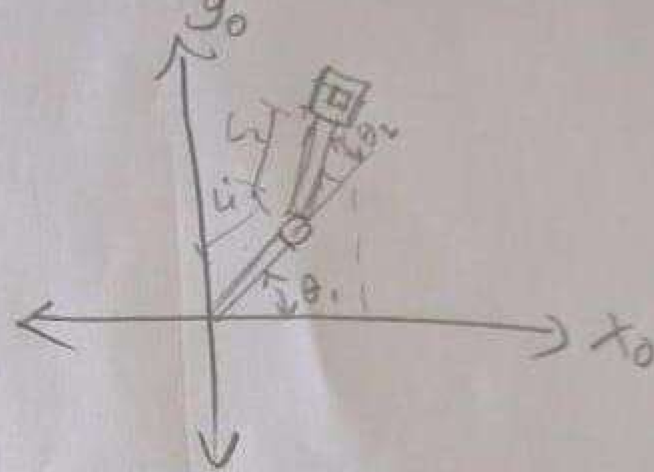
```
L1 =

    12


L2 =

    12


EqX =

px == 12*cos(q1 + q2) + 12*cos(q1)


EqY =

py == 12*sin(q1 + q2) + 12*sin(q1)


EqZ =

pz == 12 - q3


Sol_q1 =

2*atan((24*py + (- px^4 - 2*px^2*py^2 + 576*px^2 - py^4 + 576*py^2)^(1/2))/
(px^2 + 24*px + py^2))
2*atan((24*py - (- px^4 - 2*px^2*py^2 + 576*px^2 - py^4 + 576*py^2)^(1/2))/
(px^2 + 24*px + py^2))


Sol_q2 =

-2*atan((-(px^2 + py^2)*(px^2 + py^2 - 576))^(1/2)/(px^2 + py^2))
```

```
 2*atan((-(px^2 + py^2)*(px^2 + py^2 - 576))^(1/2)/(px^2 + py^2))


Sol_q3 =

12 - pz
12 - pz
```

Published with MATLAB® R2022b

③





$$x = L_1' \cos\theta_1 + L_2 \cos(\theta_1 + \theta_2) = L_1' C\theta_1 + L_2 C(\theta_1 + \theta_2)$$

$$y = L_1' S\theta_1 + L_2 S(\theta_1 + \theta_2)$$

$$z = L_1 - d_3$$

```
clear, clc

syms px py pz az ax ay
syms q1 q2 q3

L1 = 12
L2 = L1
L3 = L1

DHParam = [q1 L1 0 0; -pi/2 q2 0 -pi/2; pi/2 q3 0 0];

[A, T, Q, Rot, Tra] = DH(DHParam, px, py, pz, az);

    EqX = px == (Tra(1));
    EqY = py == (Tra(2));
    EqZ = pz == (Tra(3));

    Eqs =  [EqX; EqY; EqZ];
    % n, s, a
    %EqAux = az == Rot(2,3);
    %EqAux = az == Rot(3,3);

    EqAux = az == Rot(3,1);

    jointVar = symvar(T);

Q = solve([EqX EqY EqZ], jointVar, "Real",true)%, "PrincipalValue",false)

%%%%%%%%5
```

```
L1 =

    12


L2 =

    12


L3 =

    12

     3

[q1, 12, 0, 0]

[-pi/2, q2, 0, -pi/2]

[pi/2, q3, 0, 0]
```

```
(:,:,1) =

[cos(q1), -sin(q1), 0,  0]
[sin(q1),  cos(q1), 0,  0]
[      0,        0, 1, 12]
[      0,        0, 0,  1]

(:,:,2) =

[      0,        0, 1,  0]
[     -1,        0, 0,  0]
[      0,       -1, 0, q2]
[      0,        0, 0,  1]

(:,:,3) =

[      0,       -1, 0,  0]
[      1,        0, 0,  0]
[      0,        0, 1, q3]
[      0,        0, 0,  1]

[ 0, -sin(q1), cos(q1), q3*cos(q1)]
[ 0,  cos(q1), sin(q1), q3*sin(q1)]
[-1,        0,       0,    q2 + 12]
[ 0,        0,       0,          1]

Warning: Solutions are only valid under certain conditions. To include
parameters and conditions in the solution, specify the 'ReturnConditions'
 value
as 'true'.

Q =

  struct with fields:

    q1: [2×1 sym]
    q2: [2×1 sym]
    q3: [2×1 sym]
```

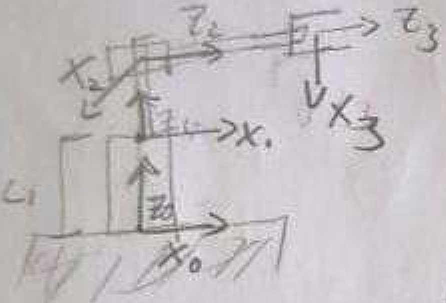
assume(q1>-pi & q1<pi & q2>0 & q3>0)

```
   EqX = px == expand(Tra(1));
   EqY = py == expand(Tra(2));
   EqZ = pz == expand(Tra(3));

   EqAux = az == Rot(3,1);
   EqAux2 = ay == Rot(2,1);
   EqAux3 = ax == Rot(1,1);

   jointVar = symvar(T);

   %Q = solve([EqX EqY EqZ EqAux1 EqAux1b EqAux2 EqAux2b EqAux3 EqAux3b], jointVar, "Real"
```

```
Q = solve([EqX EqY EqZ EqAux EqAux2 EqAux3], jointVar(2:end), "Real",true) % Se puede j
Q2 = solve([EqX EqY EqZ EqAux EqAux2 EqAux3], jointVar, "Real",true) % Se puede jugar c

 %Q.q1 = solve

 %%%%%%%%%%55

% A es el arreglo conteniendo las matrices de transformación de i = 1 a i =
% n. T es la matriz de transformación simplificada, la matriz de cinemática
% directa. Q es un vector conteniendo los resultados para las variables de
% q1 a qn. Rot es la matriz de rotación extraída de T. Y Tra es el ventor
% posición extraído de T.

% syms n_x s_x a_x p_x n_y s_y a_y p_y n_z s_z a_z p_z
% T_syms = [n_x s_x a_x p_x;n_y s_y a_y p_y;n_z s_z a_z p_z;0 0 0 1];
%
% T1 = (A(:,:,1))\T_syms == A(:,:,2)*A(:,:,3)




%inverseKinematics()

%     ia = inv(A(:, :, 1));
%     ter2 = A(:, :, 2) * A(:, :, 3);
%     eq1 = ter2(3,4) == ia (3,1)*px + ia(3,2) *py;
%     %q1 = solve(eq1,q1, 'real',true);
%
%     ib = inv(A(:, :, 2))*ia;
%     ter1 = A(:, :, 3);
%
%     eq2 = ter1(2,4) == ib(2,1)*px + ib(2,2)*py + ib(2,3)*pz + ib(2,4)*1;
%     eq3 = ter1(1,4) == ib(1,1)*px + ib(1,2)*py + ib(1,3)*pz + ib(1,4)*1;
%
%     [q2,q3] = solve([eq2,eq3],[q2 q3], 'real', true);
%
%     q1 = atan2(py,px);
%
%     q2 = eval(q2(1));
%     q3 = eval(q3(1));
%
%
%     th = [q1,q2,q3];
%     th2 = rad2deg(th);
%
%     th2(1)
%     th2(2)
%     th2(3)
```

Published with MATLAB® R2022b

①



A	θ_i	d_i	a_i	α_i
1	q_1	L_1	0	0
2	-90	q_2	0	-90
3	90	q_3	0	0

$$^{i-1}A_i = \begin{pmatrix} C\theta & -C\alpha S\theta & S\alpha S\theta & aC\theta \\ S\theta & C\alpha C\theta & -S\alpha C\theta & aS\theta \\ 0 & S\alpha & C\alpha & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$T = {}^0A_3 = {}^0A_1\,{}^1A_2\,{}^2A_3$$

$$^0A_1 = \begin{pmatrix} C\theta_1 & -S\theta & 0 & 0 \\ S\theta_1 & C\theta & 0 & 0 \\ 0 & 0 & 1 & L_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$^1A_2 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & q_2 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$^2A_3 = \begin{pmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & q_3 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

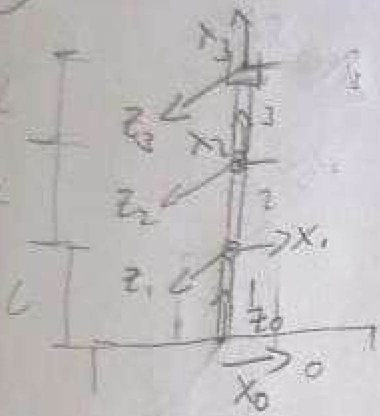
```
clear, clc

syms px py pz az ax ay
syms q1 q2 q3

L1 = 12
L2 = L1
L3 = L1

DHParam = [q1 L1 0 pi/2; q2 0 L2 0; q3 0 L3 0];

[A, T, Q, Rot, Tra] = DH(DHParam, px, py, pz, az);

%assume(q1>-pi & q1<pi & q2>-pi & q2<pi & q3>-pi & q3<pi)
assume(pz>0 & q1<0 & q1 > -pi)% & q2>0 & q3>0)

    EqX = px == simplify(Tra(1));
    EqY = py == simplify(Tra(2));
    EqZ = pz == simplify(Tra(3));

    jointVar = symvar(T);

%     Q3 = solve([EqZ], jointVar(3), 'real',true) % Se puede jugar con las
 variables a resolver y a partir de ahí sacar la última solución. Usando la
 primera variable como parámetro. Dado el caso de que las ecuaciones no son
 linealmente indemendientes entre sí
%     Q3b = Q3(1) %== q3
%     %q3 = Q3(1)
%     EqX = simplify(subs(EqX, q3, Q3b))
%     EqY = simplify(subs(EqY, q3, Q3b))
%     Q12 = solve([EqX EqY], jointVar(1:2)) % Se puede jugar con las
 variables a resolver y a partir de ahí sacar la última solución. Usando la
 primera variable como parámetro. Dado el caso de que las ecuaciones no son
 linealmente indemendientes entre sí

%     Q1 = solve([EqX], jointVar(1), 'real',true) % Se puede jugar con las
 variables a resolver y a partir de ahí sacar la última solución. Usando la
 primera variable como parámetro. Dado el caso de que las ecuaciones no son
 linealmente indemendientes entre sí
%     Q1b = Q1(2) %== q3
%     %q3 = Q3(1)
%     EqY = simplify(subs(EqY, jointVar(1), Q1b))
%     EqZ = simplify(subs(EqZ, jointVar(1), Q1b))
%     Q23 = solve([EqX EqY], jointVar(2:3)) % Se puede jugar con las
 variables a resolver y a partir de ahí sacar la última solución. Usando la
 primera variable como parámetro. Dado el caso de que las ecuaciones no son
 linealmente indemendientes entre sí


    %%%%%%%%%55

% A es el arreglo conteniendo las matrices de transformación de i = 1 a i =
```

```
% n. T es la matriz de transformación simplificada, la matriz de cinemática
% directa. Q es un vector conteniendo los resultados para las variables de
% q1 a qn. Rot es la matriz de rotación extraída de T. Y Tra es el ventor
% posición extraído de T.

% syms n_x s_x a_x p_x n_y s_y a_y p_y n_z s_z a_z p_z
% T_syms = [n_x s_x a_x p_x;n_y s_y a_y p_y;n_z s_z a_z p_z;0 0 0 1];
%
% T1 = (A(:,:,1))\T_syms == A(:,:,2)*A(:,:,3)




%inverseKinematics()

    ia = inv(A(:, :, 1));
    ter2 = A(:, :, 2) * A(:, :, 3);
    eq1 = ter2(3,4) == ia (3,1)*px + ia(3,2) *py;
    %q1 = solve(eq1,q1, 'real',true);

    ib = inv(A(:, :, 2))*ia;
    ter1 = A(:, :, 3);

    eq2 = ter1(2,4) == ib(2,1)*px + ib(2,2)*py + ib(2,3)*pz + ib(2,4)*1;
    eq3 = ter1(1,4) == ib(1,1)*px + ib(1,2)*py + ib(1,3)*pz + ib(1,4)*1;

    [q2,q3] = solve([eq2,eq3],[q2 q3], 'real', true);

    q1 = atan2(py,px);

    q2 = eval(q2(1));
    q3 = eval(q3(1));


    th = [q1,q2,q3];
    th2 = rad2deg(th);

    th2(1);
    th2(2);
    th2(3);


% fun = @(q) norm([Tra(1);Tra(2);Tra(3)]);


L1 =

    12


L2 =

    12
```

```
L3 =

    12

     3

[q1, 12, 0, pi/2]

[q2, 0, 12, 0]

[q3, 0, 12, 0]


(:,:,1) =

[cos(q1),        0,  sin(q1),          0]
[sin(q1),        0, -cos(q1),          0]
[      0,        1,        0,         12]
[      0,        0,        0,          1]

(:,:,2) =

[cos(q2), -sin(q2),        0, 12*cos(q2)]
[sin(q2),  cos(q2),        0, 12*sin(q2)]
[      0,        0,        1,          0]
[      0,        0,        0,          1]

(:,:,3) =

[cos(q3), -sin(q3),        0, 12*cos(q3)]
[sin(q3),  cos(q3),        0, 12*sin(q3)]
[      0,        0,        1,          0]
[      0,        0,        0,          1]

[cos(q2 + q3)*cos(q1), -sin(q2 + q3)*cos(q1),  sin(q1), 12*cos(q1)*(cos(q2 +
 q3) + cos(q2))]
[cos(q2 + q3)*sin(q1), -sin(q2 + q3)*sin(q1), -cos(q1), 12*sin(q1)*(cos(q2 +
 q3) + cos(q2))]
[        sin(q2 + q3),          cos(q2 + q3),        0,   12*sin(q2 + q3) +
 12*sin(q2) + 12]
[                   0,                     0,        0,
            1]

Warning: Solutions are only valid under certain conditions. To include
parameters and conditions in the solution, specify the 'ReturnConditions'
 value
as 'true'.
```

Published with MATLAB® R2022b

(4)



A	θ_i	d_i	a_i	α_i
1	q_1	L_1	0	90
2	q_2	0	L_2	0
3	q_3	0	L_3	0

$$T = {}^0A_3 = {}^0A_1\,{}^1A_2\,{}^2A_3$$

$${}^{i-1}A_i = \left(\begin{array}{ccc|c} C\theta & -C\alpha S\theta & S\alpha S\theta & aC\theta \\ S\theta & C\alpha C\theta & -S\alpha C\theta & aS\theta \\ 0 & S\alpha & C\alpha & d_i \\ \hline 0 & 0 & 0 & 1 \end{array}\right)$$

$${}^0A_1 = \left(\begin{array}{ccc|c} C\theta_1 & 0 & S\theta_1 & 0 \\ S\theta_1 & 0 & -C\theta_1 & 0 \\ 0 & 1 & 0 & L_1 \\ \hline 0 & 0 & 0 & 1 \end{array}\right) \qquad {}^1A_2 = \left(\begin{array}{ccc|c} C\theta_2 & -S\theta_2 & 0 & L_2C\theta_2 \\ S\theta_2 & C\theta_2 & 0 & L_2S\theta_2 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right)$$

$${}^1A_3 = \left(\begin{array}{ccc|c} C\theta_3 & -S\theta_3 & 0 & L_3C\theta_3 \\ S\theta_3 & C\theta_3 & 0 & L_3S\theta_3 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right)$$

```
clear, clc
tic
syms px py pz az ax ay
syms q1 q2 q3

L1 = 13.5
L2 = 15
L3 = 20

DHParam = [q1 L1 0 pi/2; (pi/2+q2) 0 L2 0; q3 0 L3 0];

[A, T, Q, Rot, Tra] = DH(DHParam, px, py, pz, az);

%assume(q1>-pi & q1<pi & q2>-pi & q2<pi & q3>-pi & q3<pi)
assume(pz>0 & q1<0 & q1 > -pi)% & q2>0 & q3>0)

    EqX = px == simplify(Tra(1));
    EqY = py == simplify(Tra(2));
    EqZ = pz == simplify(Tra(3));

    jointVar = symvar(T);


    %%%%%%%%%%55

% A es el arreglo conteniendo las matrices de transformación de i = 1 a i =
% n. T es la matriz de transformación simplificada, la matriz de cinemática
% directa. Q es un vector conteniendo los resultados para las variables de
% q1 a qn. Rot es la matriz de rotación extraída de T. Y Tra es el ventor
% posición extraído de T.

% syms n_x s_x a_x p_x n_y s_y a_y p_y n_z s_z a_z p_z
% T_syms = [n_x s_x a_x p_x;n_y s_y a_y p_y;n_z s_z a_z p_z;0 0 0 1];
%
% T1 = (A(:,:,1))\T_syms == A(:,:,2)*A(:,:,3)


    ia = inv(A(:, :, 1));
    ter2 = A(:, :, 2) * A(:, :, 3);
    eq1 = ter2(3,4) == ia (3,1)*px + ia(3,2) *py;
    %q1 = solve(eq1,q1, 'real',true);

    ib = inv(A(:, :, 2))*ia;
    ter1 = A(:, :, 3);

    eq2 = ter1(2,4) == ib(2,1)*px + ib(2,2)*py + ib(2,3)*pz + ib(2,4)*1;
    eq3 = ter1(1,4) == ib(1,1)*px + ib(1,2)*py + ib(1,3)*pz + ib(1,4)*1;

    %assume(q2>0 & q3>0)
    [q2,q3] = solve([eq2,eq3],[q2 q3], 'real', true);
```

```
    q1 = atan2(-py,-px); %ajustado a mano para dar órdenes de valores
 negativos

    q2a = eval(q2(1));
    q3a = eval(q3(1));

    q2b = eval(q2(2));
    q3b = eval(q3(2));

    toc

%     th = [q1,q2,q3];
%     th2 = rad2deg(th);
%
%     th2(1);
%     th2(2);
%     th2(3);


% fun = @(q) norm([Tra(1);Tra(2);Tra(3)]);
```

```
L1 =

   13.5000


L2 =

    15


L3 =

    20

     3

[q1, 27/2, 0, pi/2]

[q2 + pi/2, 0, 15, 0]

[q3, 0, 20, 0]


(:,:,1) =

[       cos(q1),               0,  sin(q1),                   0]
[       sin(q1),               0, -cos(q1),                   0]
[             0,               1,        0,                27/2]
[             0,               0,        0,                   1]

(:,:,2) =
```

```
[cos(q2 + pi/2), -sin(q2 + pi/2),        0, 15*cos(q2 + pi/2)]
[sin(q2 + pi/2),  cos(q2 + pi/2),        0, 15*sin(q2 + pi/2)]
[             0,               0,        1,                 0]
[             0,               0,        0,                 1]

(:,:,3) =

[       cos(q3),        -sin(q3),        0,       20*cos(q3)]
[       sin(q3),         cos(q3),        0,       20*sin(q3)]
[             0,               0,        1,                0]
[             0,               0,        0,                1]

[-sin(q2 + q3)*cos(q1), -cos(q2 + q3)*cos(q1),  sin(q1), -5*cos(q1)*(4*sin(q2
 + q3) + 3*sin(q2))]
[-sin(q2 + q3)*sin(q1), -cos(q2 + q3)*sin(q1), -cos(q1), -5*sin(q1)*(4*sin(q2
 + q3) + 3*sin(q2))]
[         cos(q2 + q3),         -sin(q2 + q3),        0,     20*cos(q2 + q3) +
 15*cos(q2) + 27/2]
[                    0,                     0,        0,
                 1]

Warning: Solutions are only valid under certain conditions. To include
parameters and conditions in the solution, specify the 'ReturnConditions'
value as 'true'.
Elapsed time is 44.052097 seconds.
```

Published with MATLAB® R2022b