

# Lab N.1: Manipulator Modelling

MICHELE TARTARI, STEFANO CASTAGNETTA

October 16, 2017

## Abstract

This paper presents a report for the lab that took place on the 09 October 2017 and during which the symbolic language of MATLAB was used to establish the geometrical models of manipulator robots. The first section will focus on the to the understanding off two matlab functions, namely InvTransHom.m and DHSym.m, which were provided. A second part will aim to analyze the 3R planar robot and present a valid MATLAB script capable to solve the direct and inverse geometrical problem for such robot. Later, in the third section, a similar procedure has been applied to the 6 d.o.f. cartesian robot with perfect spherical wrist.

## Contents

<b>1 Analysis of the two functions InvTransHom.m and DHSym.m</b>	<b>1</b>	<b>3 Modeling of the cartesian robot</b>	<b>5</b>
1.1 InvTransHom.m . . . . .	1	3.1 DGM: Direct Geometrical Model	5
1.2 DHSym.m . . . . .	2	3.2 IGM: Inverse Geometrical Model	7
<b>2 Modeling of the R3 plan robot</b>	<b>2</b>	3.3 Simulation and visualization of the cartesian robot . . . . .	8
2.1 DGM: Direct Geometrical Model	2	<b>Conclusion</b>	<b>11</b>
2.2 IGM: Inverse Geometrical Model	3	Appendix . . . . .	12
2.3 Simulation and visualization of the R3PLAN robot . . . . .	3		

## 1. Analysis of the two functions InvTransHom.m and DHSym.m

### 1.1. InvTransHom.m

This function receives a matrix as input and returns the inverse of that matrix.

The inverse is computed through operations that make use of the property of the transformation matrix inverse as shown in the expression:

$$T^{-1} = \left[ \begin{array}{ccc|c} & & & -s^T \cdot P \\ & R^T & & -n^T \cdot P \\ & & & -a^T \cdot P \\ \hline 0 & 0 & 0 & 1 \end{array} \right]$$

where  $R$  is the rotation matrix extracted from  $T$  and  $P$  is the translation vector.

$$R = \begin{bmatrix} s & n & a \end{bmatrix}$$

## 1.2. DHSym.m

This function receives the MDH<sup>1</sup> table parameters  $(\alpha, d, \theta, r)$  and returns the total transformation matrix between the first frame and the last frame ( ${}^0T_N$ ).

For each frame is computed the corresponding matrix according to the MDH convention and all of this matrices are multiplied together to obtain the  ${}^0T_n$  matrix.

$${}^0T_N = {}^0T_1 \cdot {}^1T_2 \cdot \dots \cdot {}^{i-1}T_i \cdot \dots \cdot {}^{N-1}T_N$$

Where:

$${}^{i-1}T_i = \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) & 0 & d_i \\ \cos(\alpha_i) \cdot \sin(\theta_i) & \cos(\alpha_i) \cdot \cos(\theta_i) & -\sin(\alpha_i) & -r_i \cdot \sin(\alpha_i) \\ \sin(\alpha_i) \cdot \sin(\theta_i) & \sin(\alpha_i) \cdot \cos(\theta_i) & \cos(\alpha_i) & r_i \cdot \cos(\alpha_i) \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

## 2. Modeling of the R3 plan robot

### 2.1. DGM: Direct Geometrical Model

The DGM computes the end effector position and orientation in function of joint variables.

In the case of the R3 plan robot the position is given by  $P_x$  and  $P_y$  while the orientation is given by  $\alpha$ . Since we have 3 rigid bodies there are 3 joints ( $q_1, q_2, q_3$ ). The geometrical parameters (lengths of robot bodies  $L_1, L_2, L_3$ ) are given.

We describe this robot using the MDH notation. We consider the fixed frame and the end effector frame equal respectively to zero frame and the last body frame. The axis and the origins are shown in Figure 1. Positive direction for  $z$  axis is considered the one coming out of the plane and for  $x$  axis the one going from joint  $j$  to joint  $j + 1$ . The MDH table is show in Table 1.

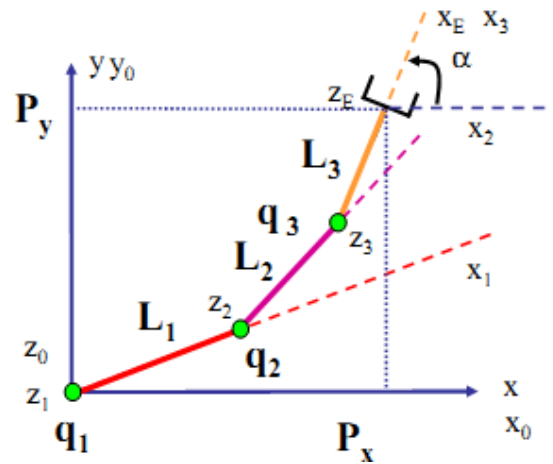


Figure 1: R3 Robot

<sup>1</sup>Modified Denavit-Hartenberg

Table 1: R3 Robot MDH table

	$\sigma_k$	$\alpha_k$	$d_k$	$\theta_k$	$r_k$
$R_0 \rightarrow R_1$	0	0	0	$q_1$	0
$R_1 \rightarrow R_2$	0	0	$L_1$	$q_2$	0
$R_2 \rightarrow R_3$	0	0	$L_2$	$q_3$	0
$R_3 \rightarrow R_e$	0	0	$L_3$	0	0

For this robot is possible to compute the DGM with a geometrical approach. This leads to the following equations :

$$\begin{aligned}
 P_x &= T_{tot}[1,4] = \cos(\alpha) \cdot L_3 + \cos(q_2) \cdot L_2 + \cos(q_1) \cdot L_1. \\
 P_y &= T_{tot}[2,4] = \sin(\alpha) \cdot L_3 + \sin(q_2) \cdot L_2 + \sin(q_1) \cdot L_1. \\
 \alpha &= q_1 + q_2 + q_3
 \end{aligned}$$

By implementing these in the code and asking the user for the values of the joint variables we are able to give  $P_x$ ,  $P_y$  and  $\alpha$ . The solution is unique as for every serial robot in the case of DGM.

## 2.2. IGM: Inverse Geometrical Model

The IGM computes the joint variables ( $q_1, q_2, q_3$ ) in function of the end effector location ( $P_x, P_y, \alpha$ ). Geometrical considerations give us the following equation:

$$\begin{aligned}
 P_x - \cos(\alpha) \cdot L_3 &= \cos(q_1 + q_2) \cdot L_2 + \cos(q_1) \cdot L_1 \\
 P_y - \sin(\alpha) \cdot L_3 &= \sin(q_1 + q_2) \cdot L_2 + \sin(q_1) \cdot L_1 \\
 \alpha - q_1 - q_2 &= q_3
 \end{aligned}$$

These equations can be solved referring first to an equation of type 8 to find  $q_{21}$  and  $q_{22}$  and after to an equation of type 3 to get the solution for  $q_{11}$  and  $q_{12}$ . With the third equation is possible to determine  $q_{31}$  and  $q_{33}$ .

## 2.3. Simulation and visualization of the R3PLAN robot

For the function mgdmgi.m we used a variable that let us call the two scripts mgi.m and mgd.m without having user input again for the mgi function. The variable is initialized to 1 in the 3rplan.m file and modified in the mgdmgi.m in order not to enter in the input loop. Inside the mgi.m we have:

```

1 if (mgd_mgi)
2 disp('entrer les valeurs des variables operationnelles');
3 Px = input('Px = ');
4 Py = input('Py = ');
5 alpha = input('alpha = ');
6 end

```

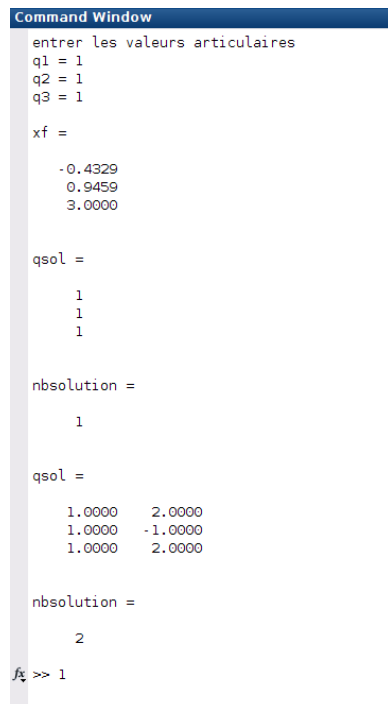
It allows us to have a very simple mgdmgi.m file:

```

1  %faire MGD
2  mgd
3
4  %faire MGI
5  mgd_mgi=1;
6  mgi
7
8  visuart
9  pause

```

In figure 2 and in Figure 3 two examples of the working code are given, it can be observed that user input is not required anymore.



```

Command Window
entrer les valeurs articulaires
q1 = 1
q2 = 1
q3 = 1

xf =

    -0.4329
     0.9459
     3.0000

qsol =

     1
     1
     1

nbsolution =

     1

qsol =

     1.0000     2.0000
     1.0000    -1.0000
     1.0000     2.0000

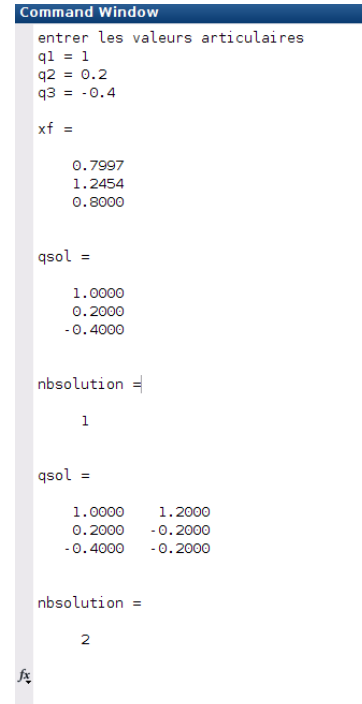
nbsolution =

     2

fx >> 1

```

Figure 2: mgdmgi(1, 1, 1)



```

Command Window
entrer les valeurs articulaires
q1 = 1
q2 = 0.2
q3 = -0.4

xf =

     0.7997
     1.2454
     0.8000

qsol =

     1.0000
     0.2000
    -0.4000

nbsolution =

     1

qsol =

     1.0000     1.2000
     0.2000    -0.2000
    -0.4000    -0.2000

nbsolution =

     2

fx

```

Figure 3: mgdmgi(1, 0.2, -0.4)

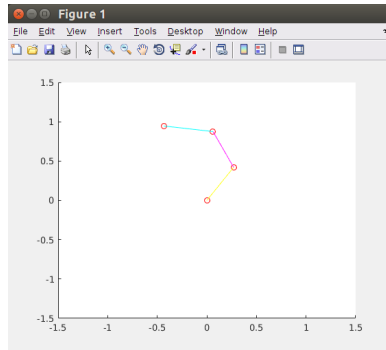


Figure 4: mgdmgi(1, 1, 1) drawing

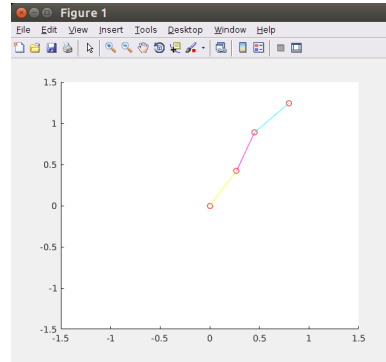


Figure 5: mgdmgi(1, 0.2, -0.4) drawing

In Figure 4 and in Figure 5 is possible to see the the right solution is always shown (thanks to a change in the code).

### 3. Modeling of the cartesian robot

#### 3.1. DGM: Direct Geometrical Model

As for the DGM computes the end-effector position and orientation in function of joint variables. We define the position as  $P = (P_x, P_y, P_z, \theta_1, \theta_2, \theta_3)^T$  where the first 3 terms define the cartesian position and the latter the orientation expressed in Bryant angles. The robot has 3 prismatic joints  $(q_1, q_2, q_3)$  and a spherical wrist  $(q_4, q_5, q_6)$ .

We describe this robot using the MDH notation. The axis and the origins are shown in Figure 6. The MDH table is show in Table 2.

Table 2: MDH Table of the Cartesian Robot

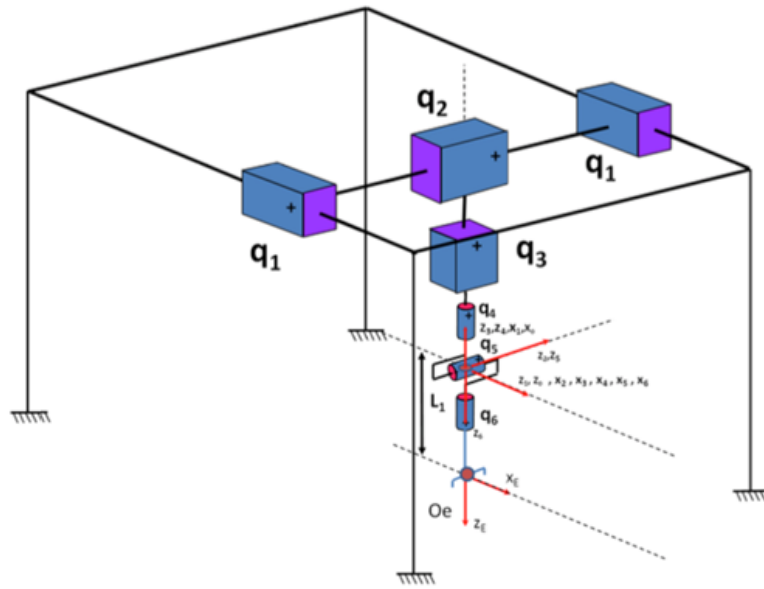
	$\sigma_k$	$\alpha_k$	$d_k$	$\theta_k$	$r_k$
$R_0 \rightarrow R_1$	1	0	$q_1$	0	0
$R_1 \rightarrow R_2$	1	$\pi/2$	$q_2$	0	$\pi/2$
$R_2 \rightarrow R_3$	1	$\pi/2$	$q_3$	0	0
$R_3 \rightarrow R_4$	0	0	0	$q_4$	0
$R_4 \rightarrow R_5$	0	$-\pi/2$	0	$q_5$	0
$R_5 \rightarrow R_6$	0	$-\pi/2$	0	$q_6$	0

The MDH table is then fed to the MATLAB function which will return the homogeneous transofmation matrix  ${}^0T_N$ . In order to have our coordinate in the absolute reference system we define the transformation matix  ${}^AT_E$  which is defined as:

$${}^AT_E = {}^AT_0 \cdot {}^0T_6 \cdot {}^6T_E$$

where  ${}^AT_0$  is the transformation from the reference frame  $O_A$  to  $O_0$  and  ${}^6T_E$  is the transformation from  $O_6$  to end-effector reference frame  $O_E$ . Using Bryant angles for our application the two matrices will be respectively equal to:

Figure 6: Cartesian Robot



$${}^A T_0 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^6 T_E = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & L_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Finally we can define

$${}^A T_E = \begin{bmatrix} s & n & a & p \end{bmatrix} = \begin{bmatrix} s_x & n_x & a_x & p_x \\ s_y & n_y & a_y & p_y \\ s_z & n_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

From this equivalence we can directly obtain the position  $p$  is the position of the end-effector. To obtain its orientation some calculation will be required. From the previous equation we can obtain:

$${}^A T_E = \left\{ \begin{array}{c|c} {}^A R_E & {}^A P_E \\ \hline 0 & 0 & 0 & 1 \end{array} \right\} \Rightarrow {}^A R_E = \begin{bmatrix} s_x & n_x & a_x \\ s_y & n_y & a_y \\ s_z & n_z & a_z \end{bmatrix}$$

By the definitoin of Bryan angles we can state that:

$$\begin{aligned} R_{Bry} &= R(x, \theta_1) \cdot R(y, \theta_2) \cdot R(z, \theta_3) \\ &= \begin{bmatrix} \cos(\theta_2) \cdot \cos(\theta_3) & \cos(\theta_2) \cdot \sin(\theta_3) & \sin(\theta_2) \\ \sin(\theta_1) \cdot \sin(\theta_2) \cdot \cos(\theta_3) + \cos(\theta_1) \cdot \sin(\theta_3) & -\sin(\theta_1) \cdot \sin(\theta_2) \cdot \sin(\theta_3) + \cos(\theta_1) \cdot \sin(\theta_3) & -\sin(\theta_1) \cdot \cos(\theta_2) \\ -\cos(\theta_1) \cdot \sin(\theta_2) \cdot \cos(\theta_3) + \sin(\theta_1) \cdot \sin(\theta_3) & \cos(\theta_1) \cdot \sin(\theta_2) \cdot \sin(\theta_3) + \sin(\theta_1) \cdot \sin(\theta_3) & \cos(\theta_1) \cdot \cos(\theta_2) \end{bmatrix} \end{aligned}$$

From which we can obtain:

$$R_{Bry} = \begin{bmatrix} s_x & n_x & a_x \\ s_y & n_y & a_y \\ s_z & n_z & a_z \end{bmatrix} \Rightarrow \begin{cases} \theta_1 &= \text{atan2}(-a_y, a_z) \\ \theta_2 &= \text{atan2}(-a_x, -a_y \cdot \sin(\theta_1) + a_z \cdot \cos(\theta_1)) \\ \theta_3 &= \text{atan2}(a_y \cdot \cos(\theta_1) + n_y \cdot \sin(\theta_1), n_y \cdot \cos(\theta_1) + n_z \cdot \sin(\theta_1)) \end{cases}$$

there will be only one solution  $X = (p_x, p_y, p_z, \theta_1, \theta_2, \theta_3)^T$  for a given set  $Q = (q_1, q_2, q_3, q_4, q_5, q_6)^T$

### 3.2. IGM: Inverse Geometrical Model

Given the end effector position vector  $X = (p_x, p_y, p_z, \theta_1, \theta_2, \theta_3)^T$ , described in bryant angles and cartesian coordinates we can obtain one or more sets of solutions  $Q = (q_1, q_2, q_3, q_4, q_5, q_6)^T$ . From  $X$  we can describe the transformation from  $O_A$  to  $O_E$  as  ${}^A T_E \downarrow_{Num}$ .

$${}^A T_E \downarrow_{Num} = \left[ \begin{array}{ccc|c} R_{Bry} & P \\ \hline 0 & 0 & 0 & 1 \end{array} \right]$$

where

$$\begin{aligned} P &= (p_x, p_y, p_z)^T \\ R_{Bry} &= R(x, \theta_1) \cdot R(y, \theta_2) \cdot R(z, \theta_3) \\ &= \begin{bmatrix} \cos(\theta_2) \cdot \cos(\theta_3) & \cos(\theta_2) \cdot \sin(\theta_3) & \sin(\theta_2) \\ \sin(\theta_1) \cdot \sin(\theta_2) \cdot \cos(\theta_3) + \cos(\theta_1) \cdot \sin(\theta_3) & -\sin(\theta_1) \cdot \sin(\theta_2) \cdot \sin(\theta_3) + \cos(\theta_1) \cdot \sin(\theta_3) & -\sin(\theta_1) \cdot \cos(\theta_2) \\ -\cos(\theta_1) \cdot \sin(\theta_2) \cdot \cos(\theta_3) + \sin(\theta_1) \cdot \sin(\theta_3) & \cos(\theta_1) \cdot \sin(\theta_2) \cdot \sin(\theta_3) + \sin(\theta_1) \cdot \sin(\theta_3) & \cos(\theta_1) \cdot \cos(\theta_2) \end{bmatrix} \end{aligned}$$

From the DGM we can say that

$$\begin{aligned} {}^A T_E \downarrow_{Num} &= ({}^A T_0)^{-1} \cdot {}^0 T_6 \downarrow_{Num} \cdot ({}^6 T_E)^{-1} \\ {}^0 T_6 \downarrow_{Num} &= {}^0 T_6 \downarrow_{DGM} \\ \Rightarrow {}^0 T_6 \downarrow_{DGM} &= \begin{bmatrix} ss_x & nn_x & aa_x \\ ss_y & nn_y & aa_y \\ ss_z & nn_z & aa_z \end{bmatrix} \end{aligned}$$

From this equation we can obtain:

$$\begin{cases} q_1 &= pp_z \\ q_2 &= -pp_y \\ q_3 &= pp_x \end{cases}$$

We can then formalize the problem to:

$${}^0 R_6 \downarrow_{Num} = {}^0 R_6 \downarrow_{DGM}$$

It is now useful to restrict our focus to the wrist. From  ${}^0 R_6 \downarrow_{DGM}$  can notice that  $q_1, q_2, q_3$  are prismatic joint and have no influence on the rotation matrix, and thus:

$$\begin{aligned} {}^3 R_0 \cdot {}^0 R_6 \downarrow_{Num} &= {}^3 R_0 \cdot {}^0 R_6 \downarrow_{Num} \\ {}^3 R_6 \downarrow_{DGM} &= ({}^0 R_3)^{-1} \cdot {}^0 R_6 \downarrow_{Num} \\ {}^3 R_6 &= {}^3 R_6 \downarrow_{Num} \end{aligned}$$

From this equation is still not easy to solve for  $q_4, q_5, q_6$  and it is useful to isolate  $q_4$  by multiplying both sides by  $({}^3 R_4)^{-1}$ .

$$\begin{aligned} ({}^3 R_4)^{-1} \cdot {}^3 R_6 &= ({}^3 R_4)^{-1} \cdot {}^3 R_6 \downarrow_{Num} \\ {}^4 R_6 &= {}^4 R_6 \downarrow_{Num} \end{aligned}$$

By isolating the equations  ${}^4R_6[3,3] = {}^4R_6 \downarrow_{Num} [3,3]$  and  ${}^4R_6[1,3] = {}^4R_6 \downarrow_{Num} [1,3]$  we can obtain the solutions for  $q_5$ .

$$\begin{cases} {}^4R_6[3,3] = {}^4R_6 \downarrow_{Num} [3,3] \\ {}^4R_6[1,3] = {}^4R_6 \downarrow_{Num} [1,3] \end{cases} \Rightarrow \begin{cases} \sin(q_5) = -aa_x \\ \cos(q_5) = \sin(q_{4,1}) \cdot aa_y - \cos(q_{4,1}) \cdot aa_z \end{cases}$$

$$\Rightarrow \begin{cases} q_{5,1} = \text{atan2}(-aa_x, \sin(q_{4,1}) \cdot aa_y - \cos(q_{4,1}) \cdot aa_z) \\ q_{5,2} = \text{atan2}(-aa_x, \sin(q_{4,2}) \cdot aa_y - \cos(q_{4,2}) \cdot aa_z) \end{cases}$$

By isolating the equation  ${}^4R_6[2,1] = {}^4R_6 \downarrow_{Num} [2,1]$  and  ${}^4R_6[2,2] = {}^4R_6 \downarrow_{Num} [2,2]$  we can obtain the solutions for  $q_6$ .

$$\begin{cases} {}^4R_6[2,1] = {}^4R_6 \downarrow_{Num} [2,1] \\ {}^4R_6[2,2] = {}^4R_6 \downarrow_{Num} [2,2] \end{cases} \Rightarrow \begin{cases} \sin(q_6) = \cos(q_4) \cdot ss_y + \sin(q_4) \cdot ss_z \\ \cos(q_6) = -\cos(q_4) \cdot nn_y + \sin(q_4) \cdot nn_z \end{cases}$$

$$\Rightarrow \begin{cases} q_{6,1} = \text{atan2}(\cos(q_{4,1}) \cdot ss_y + \sin(q_{4,1}) \cdot ss_z, -\cos(q_{4,1}) \cdot nn_y + \sin(q_{4,1}) \cdot nn_z) \\ q_{6,2} = \text{atan2}(\cos(q_{4,2}) \cdot ss_y + \sin(q_{4,2}) \cdot ss_z, -\cos(q_{4,2}) \cdot nn_y + \sin(q_{4,2}) \cdot nn_z) \end{cases}$$

To summarize the IGM of a cartesian model has 2 solutions that are:

$$\begin{cases} q_1 = pp_z \\ q_2 = -pp_y \\ q_3 = pp_x \\ q_{4,1} = \text{atan2}(-aa_y, aa_z) \\ q_{5,1} = \text{atan2}(-aa_x, \sin(q_{4,1}) \cdot aa_y - \cos(q_{4,1}) \cdot aa_z) \\ q_{6,1} = \text{atan2}(\cos(q_{4,1}) \cdot ss_y + \sin(q_{4,1}) \cdot ss_z, -\cos(q_{4,1}) \cdot nn_y + \sin(q_{4,1}) \cdot nn_z) \end{cases}$$

and

$$\begin{cases} q_1 = pp_z \\ q_2 = -pp_y \\ q_3 = pp_x \\ q_{4,2} = \text{atan2}(-aa_y, aa_z) + \pi \\ q_{5,2} = \text{atan2}(-aa_x, \sin(q_{4,2}) \cdot aa_y - \cos(q_{4,2}) \cdot aa_z) \\ q_{6,2} = \text{atan2}(\cos(q_{4,2}) \cdot ss_y + \sin(q_{4,2}) \cdot ss_z, -\cos(q_{4,2}) \cdot nn_y + \sin(q_{4,2}) \cdot nn_z) \end{cases}$$

### 3.3. Simulation and visualization of the cartesian robot

For the function `mgdmg.m` we used a variable that let us call the two scripts `mg.m` and `mgd.m` without having user input again for the `mg` function. In figure 7 and 8 are presented two examples for the DMG. Underneath each figure it is visible the respective output drawing of the configuration.

Similarly figures from 11 and 14 give two example of the IGM.



```

Command Window

>> menu_tp
entrer les valeurs articulaires
q1 = 1
q2 = 1
q3 = 1
q4 = 0
q5 = 0
q6 = 0

xf =

    1.0000
    1.0000
    0.5000
   -3.1416
    0.0000
   -0.7854

```

Figure 7:  $\text{mgd}(1, 1, 1, 0, 0, 0)$ 

```

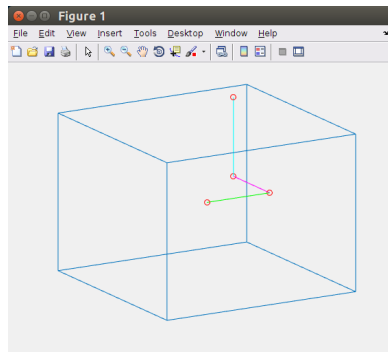
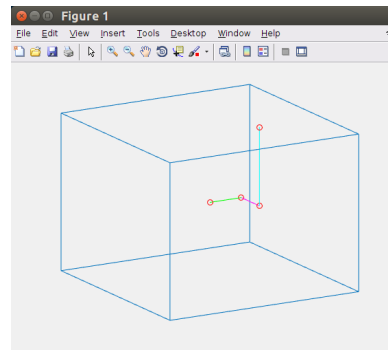
Command Window

menu_tp
entrer les valeurs articulaires
q1 = .5
q2 = -.5
q3 = 1
q4 = pi
q5 = pi-2
q6 = -pi/2

xf =

    0.9546
   -0.5000
    0.7919
    3.1416
    1.1416
    0.0000

```

Figure 8:  $\text{mgd}(0.5, 0.5, 1, \pi, \pi - 2, -\pi/2)$ Figure 9:  $\text{mgd}(1, 1, 1, 0, 0, 0)$  drawingFigure 10:  $\text{mgd}(0.5, 0.5, 1, \pi, \pi - 2, -\pi/2)$  drawing

```

Command Window

>> menu_tp
Entrer les valeurs des variables operationnelles
Px = 1
Py = 1
Pz = 1
tetax = pi
tetay = pi/2
tetaz = pi/4

qsol =

    0.5000    0.5000
    1.0000    1.0000
    1.0000    1.0000
   -0.0000    3.1416
    3.1416    0.0000
    2.3562   -0.7854

```

Figure 11:  $\text{mgi}(1, 1, 1, \pi, \pi/2, \pi/4)$ 

```

Command Window

>> menu_tp
Entrer les valeurs des variables operationnelles
Px = .5
Py = -.5
Pz = 0
tetax = pi
tetay = 0
tetaz = 0

qsol =

    0.5000    0.5000
   -0.5000   -0.5000
    0.5000    0.5000
   -1.5708    1.5708
    1.5708    1.5708
    0.0000    3.1416

```

Figure 12:  $\text{mgi}(0.5, -0.5, 0, \pi, 0, 0)$

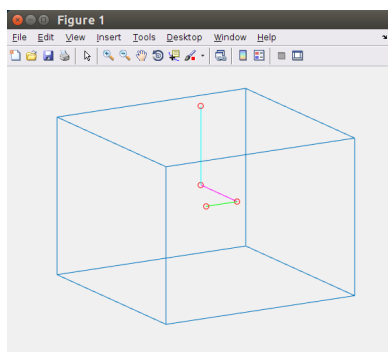


Figure 13:  $\text{mgi}(1, 1, 1, \pi, \pi/2, \pi/4)$  drawing

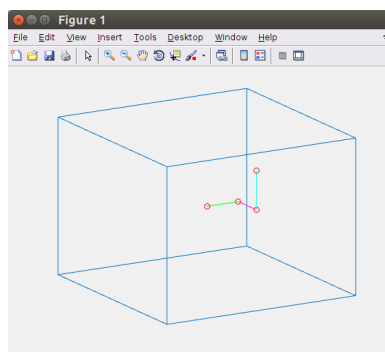


Figure 14:  $\text{mgi}(0.5, -0.5, 0, \pi, 0, 0)$  drawing

## Conclusions

As shown in section 2.3 and 3.3 our models are capable to obtain the position, both the DMG gives valid responses for the input set of data.

While approaching the resolution we first tried to give a general resolution for all type of robots but it appeared very difficult to implement and often too complex to solve. As visible in Figure 15 our attempt led to an internal error due to index overflow after an execution time of roughly 8000s. For such reason it is simpler to conduct a case-by-case study for each robot. While solving and coding we also noticed that it is much easier to proceed by dividing the problem in smaller peaces that can be implemented as MATLAB functions. Future works could include the analysis of singularities for the models.

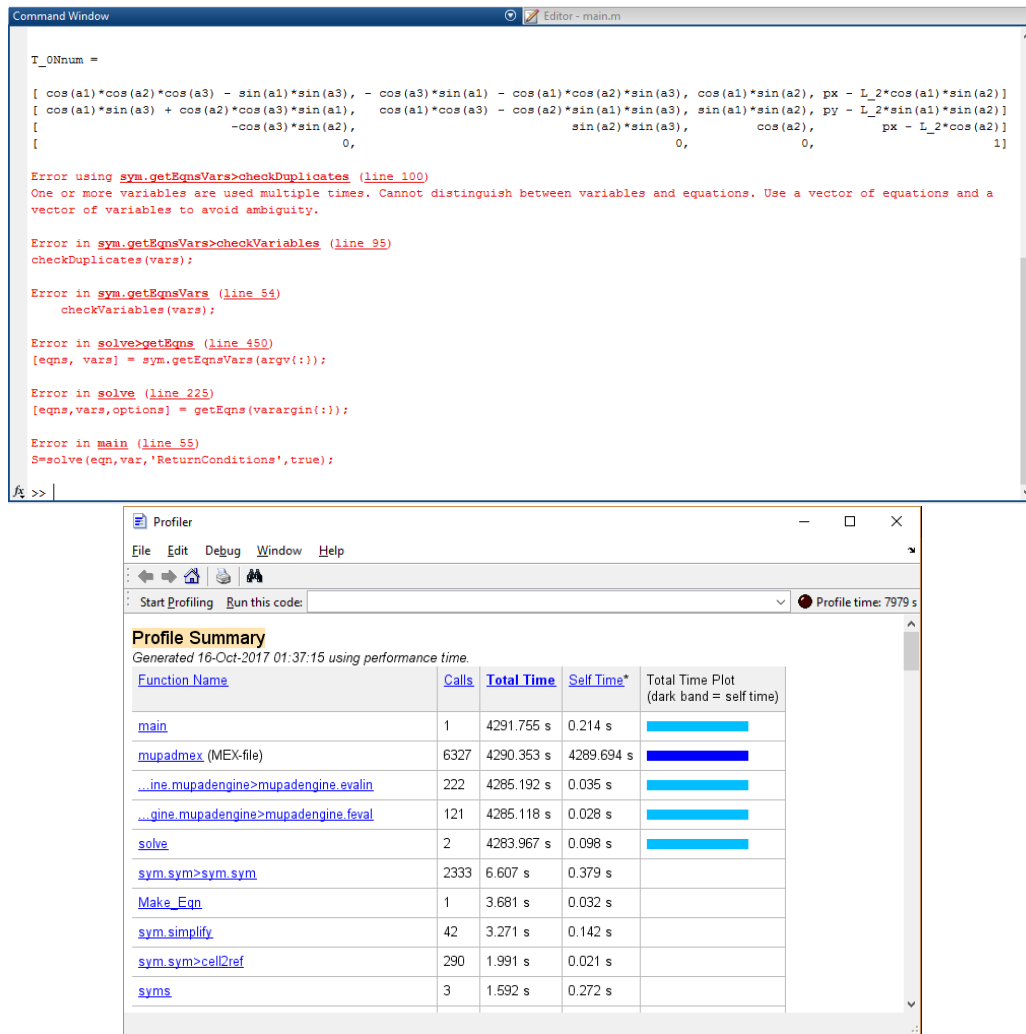


Figure 15: General case approach result often computationally demanding

## Appendix

This appendix contains a listing by alphabetical order of all MATLAB files that have been utilized in this paper.

### 3R planar robot

#### • menu\_tp.m

```

1  %MENU    TP de Robotique Genie Electrique
2  %    Copyright (C) D. Khadraoui , P. Martinet
3  %    Novembre 1996%  ROBOT 3RPLAN
4
5  % Initialisation des donnees specifiques au robot
6  close all
7  clc
8
9  r3plan
10 while 1,
11
12   which = menu('Modele Geometrique Robot 3RPLAN',...
13               'Calcul Modele Geometrique Direct', ...
14               'Calcul Modele Geometrique Inverse', ...
15               'Validation : MGD puis MGI', ...
16               'Visualisation etat articulaire', ...
17               'Exit');
18
19   if which == 1,
20       mgd
21   elseif which == 2,
22       mgi
23   elseif which == 3,
24       mgdmgi
25   elseif which == 4,
26       visuart
27       pause
28   elseif which == 5,
29       break;
30   end
31 end

```

#### • mgd.m

```

1  % Compute DGM of a 3R planar robot
2  % input section
3
4  disp('entrer les valeurs articulaires');
5  q1=input('q1 = ');
6  q2=input('q2 = ');
7  q3=input('q3 = ');
8  q=[q1;q2;q3];
9
10 % To be modified
11 Px = L1*cos(q1)+L2*cos(q1+q2)+L3*cos(q1+q2+q3);
12 Py = L1*sin(q1)+L2*sin(q1+q2)+L3*sin(q1+q2+q3);
13 alpha = q1+q2+q3;
14

```

```

15
16 % Displaying the results
17 xf=[Px;Py;alpha]
18
19 qsol=q
20
21 nbsolution=1

```

### • mgdmgi.m

```

1 % Calcul du Modelle geometrique Directe
2
3 mgd
4
5 %MGI
6
7 % Set variable to 0 so in the mgi the input won't be asked
8 mgd_mgi = 0;
9
10 mgi
11
12 % Reset variable to 1 for next runs
13 mgd_mgi = 1;

```

### • r3plan.m

```

1 % Script to initialize the parameters of the 3R robot
2
3 L1=0.5; % Longueur du premier segment
4 L2=0.5; % Longueur du premier segment
5 L3=0.5; % Longueur du premier segment
6
7 % Etat articulaire initial par default
8 q1m=0;q2m=0;q3m=0;
9
10 % Initialize to 1 variable that determines whether is asked user input
11 % in mgi or not
12 mgd_mgi=1;
13
14 kv1=1.0; % 1 rad/seconde
15 kv2=1.0; % 1 rad/seconde
16 kv3=1.0; % 1 rad/seconde
17
18 ka1=1.0; % 1 rad/seconde*seconde
19 ka2=1.0; % 1 rad/seconde*seconde
20 ka3=1.0; % 1 rad/seconde*seconde

```

### • visuart.m

```

1 hold on;
2 axis([-1.5 1.5 -1.5 1.5]);
3 for indice=1:1:nbsolution
4
5 q1m=qsol(3*(indice-1)+1);
6 q2m=qsol(3*(indice-1)+2);
7 q3m=qsol(3*(indice-1)+3);
8
9 % Trace des origine des reperes

```

```

10 O1=[0;0];
11 O2=[cos(q1m)*L1;sin(q1m)*L1];
12 O3=[cos(q1m)*L1+cos(q2m+q1m)*L2;sin(q1m)*L1+sin(q2m+q1m)*L2];
13 OE=[cos(q1m)*L1+cos(q2m+q1m)*L2+cos(q3m+q2m+q1m)*L3;sin(q1m)*L1+sin(q2m+q1m)*L2+sin(q3m+q2m+q1m)*L3];
14 plot(O1(1),O1(2),'ro');
15 plot(O2(1),O2(2),'ro');
16 plot(O3(1),O3(2),'ro');
17 plot(OE(1),OE(2),'ro');
18
19 %premier segment du robot
20 xmin=O1(1);
21 xmax=O2(1);
22 pas=(xmax-xmin)/1000;
23 x1=0:pas:cos(q1m)*L1;
24 y1=tan(q1m)*x1;
25 plot(x1,y1,'y');
26
27 %deuxieme segment du robot
28 xmin=O2(1);
29 xmax=O3(1);
30 pas=(xmax-xmin)/1000;
31
32 x2=O2(1):pas:O2(1)+cos(q1m+q2m)*L2;
33 y2=tan(q1m+q2m)*(x2-O2(1));
34
35 y2=y2+O2(2);
36 plot(x2,y2,'m');
37
38 %troisieme segment du robot
39 xmin=O3(1);
40 xmax=OE(1);
41 pas=(xmax-xmin)/1000;
42
43 x3=O3(1):pas:O3(1)+cos(q1m+q2m+q3m)*L3;
44 y3=tan(q1m+q2m+q3m)*(x3-O3(1));
45
46 y3=y3+O3(2);
47 plot(x3,y3,'c');
48
49 pause
50
51 end;

```

## Cartesian robot

### • cart.m

```

1 % Caracteristiques geometriques
2 q1min= -1.5;
3 q2min= -1.5;
4 q3min= -1;
5 q1max= +1.5;
6 q2max= +1.5;
7 q3max= +1;
8 q4min= -pi/2;
9 q5min= -pi/2;
10 q6min= -pi/2;
11 q4max= +pi/2;

```

```

12 q5max= +pi/2;
13 q6max= +pi/2;
14
15 % Etat articulaire initial par default
16 q1m=0;q2m=0;q3m=0;
17 q4m=0;q5m=0;q6m=0;
18
19 % initialization of the variable that accounts
20 % if the mgi is used alone or in the mgdmgi
21 mgd_mgi=1;
22
23 L1 = .5;
24
25 % Caracteristiques cinematiques
26 kv1=1.0; % 1 m/seconde
27 kv2=1.0; % 1 m/seconde
28 kv3=1.0; % 1 m/seconde
29 kv4=1.0; % 1 rad/seconde
30 kv5=1.0; % 1 rad/seconde
31 kv6=1.0; % 1 rad/seconde
32
33 % Caracteristiques dynamiques
34 ka1=1.0; % 1 m/seconde*seconde
35 ka2=1.0; % 1 m/seconde*seconde
36 ka3=1.0; % 1 m/seconde*seconde
37 ka4=1.0; % 1 rad/seconde*seconde
38 ka5=1.0; % 1 rad/seconde*seconde
39 ka6=1.0; % 1 rad/seconde*seconde

```

### • DHSym.m

```

1 function T0Tn = DHSym(alpha, d, theta, r)
2 %-----
3 % This function calculates the symbolic transformation matrix for the
4 % Direct Geometric Model for the 6 links Cartesian Robot with a 3dof concurrent wrist.
5
6 %This function can be transformed into a general one with the parameters of the robots as arguments:
7 %function T0Tn =
8 %and the sigma, alpha, d, theta, r should be defined as symbols as follows for the first three degrees of sta
9
10 % syms RL2 t1 t2 r3
11 % alpha = sym([0;-pi/2;pi/2]);
12 % sigma = sym([0;0;1]);
13 % theta = sym([0;0;0]);
14 % d = sym([0;0;0]);
15 % r = sym([0;RL2;r3]);
16
17
18 %-----
19 n = length(alpha);
20
21 T0Tn = eye(4);
22
23
24 % Calculates the transformation matrix for the current link.
25 for j = n:-1:1
26
27     Ct = cos(theta(j));
28     St = sin(theta(j));
29     Ca = cos(alpha(j));

```

```

30     Sa = sin(alpha(j));
31
32     T = [Ct, -St, 0, d(j);
33          Ca*St, Ca*Ct, -Sa, -r(j)*Sa;
34          Sa*St, Sa*Ct, Ca, r(j)*Ca;
35          0, 0, 0, 1];
36
37     % if theta(j) == pi/2 || theta(j) == -pi/2
38     % T(1,1)= 0;
39     % T(2,2)= 0;
40     % T(3,2)= 0;
41     % end
42     % if theta(j) == pi || theta(j) == -pi
43     % T(1,2)= 0;
44     % T(2,1)= 0;
45     % T(3,1)= 0;
46     % end
47     %
48     % if alpha(j) == pi/2 || alpha(j) == -pi/2
49     % T(2,1)= 0;
50     % T(2,2)= 0;
51     % T(3,3)= 0;
52     % T(3,4)= 0;
53     % end
54     % if alpha(j) == pi || alpha(j) == -pi
55     % T(2,3)= 0;
56     % T(2,4)= 0;
57     % T(3,1)= 0;
58     % T(3,2)= 0;
59     % end
60
61     IT = InvTransHom(T);
62
63     % to display intermediate matrices, may be commented out
64     % T
65     % IT
66     % T0Tn = simplify(T*T0Tn);
67     T0Tn = (T*T0Tn);
68
69 end
70
71 T0Tn;
72
73 end

```

#### • InvTransHom.m

```

1 function IT = InvTransHom(T)
2 %-----
3 % #####
4 % #
5 % #   Procedure d inversion d une matrice de
6 % #   transformation homogene
7 % #
8 % #   Entree : Matrice de transformation homogene 4x4 #
9 % #   Sortie : Matrice de transformation homogene 4x4 #
10 % #   T=( R  p)
11 % #   (000 1)
12 % #   IT=( Rt  -Rt*p)
13 % #   (000  1 )

```



```

14 % #####
15 %-----
16
17
18
19 % Rotation extraction
20 R = T(1:3, 1:3);
21
22 % Translation extraction
23 P = T(1:3, 4);
24
25
26 % inverse rotation
27 IR = R'; % transpose R
28
29 % translation transformation
30 IP = -IR * P;
31
32 % result composition
33 IT = [IR, IP; 0, 0, 0, 1];
34
35 %IT=[ IR(1,1), IR(1,2), IR(1,3), IP(1,1);
36 %      IR(2,1), IR(2,2), IR(2,3), IP(2,1);
37 %      IR(3,1), IR(3,2), IR(3,3), IP(3,1);
38 %      0,      0,      0, 1;    ];
39
40 end

```

#### • menu\_tp.m

```

1 %MENU   TP de Robotique Genie Electrique
2 %   Copyright (C) D. Khadraoui , P. Martinet
3 %   Novembre 1996%  ROBOT 3RPLAN
4
5 % Initialisation des donnees specifiques au robot
6 close all
7 clc
8
9 r3plan
10 while 1,
11
12 which = menu('Modele Geometrique Robot 3RPLAN',...
13             'Calcul Modele Geometrique Direct', ...
14             'Calcul Modele Geometrique Inverse', ...
15             'Validation : MGD puis MGI', ...
16             'Visualisation etat articulaire', ...
17             'Exit');
18
19 if which == 1,
20     mgd
21 elseif which == 2,
22     mgi
23 elseif which == 3,
24     mgdmgi
25 elseif which == 4,
26     visuart
27     pause
28 elseif which == 5,
29     break;
30 end

```

31 **end**

### • mgd.m

```

1  % % Calcul du Modele geometrique Direct
2  % Initialization of the joint variables
3  disp('entrer les valeurs articulaires');
4  q1 = input('q1 = ');
5  q2 = input('q2 = ');
6  q3 = input('q3 = ');
7  q4 = input('q4 = ');
8  q5 = input('q5 = ');
9  q6 = input('q6 = ');
10 q = [q1;q2;q3;q4;q5;q6];
11
12 % MDH table parameters
13 alpha = [0, pi/2, pi/2, 0, -pi/2, -pi/2];
14 theta = [0, pi/2, 0, q4, q5, q6];
15 d = [0, 0, 0, 0, 0, 0];
16 r = [q1, q2, q3, 0, 0, 0];
17
18 % Using DHSym function to compute the DGM:
19 T06= DHSym(alpha , d, theta, r);
20 TA0 = [0 0 1 0; 0 -1 0 0; 1 0 0 0; 0 0 0 1];
21 T6E = [0 1 0 0; -1 0 0 0; 0 0 1 L1; 0 0 0 1];
22 Ttot= TA0*T06*T6E;
23
24 % Extraction des coordonnees operationnelles (cosinus-directeur)
25 sx = Ttot(1,1);
26 nx = Ttot(1,2);
27 ax = Ttot(1,3);
28 Px = Ttot(1,4);
29 sy = Ttot(2,1);
30 ny = Ttot(2,2);
31 ay = Ttot(2,3);
32 Py = Ttot(2,4);
33 sz = Ttot(3,1);
34 nz = Ttot(3,2);
35 az = Ttot(3,3);
36 Pz = Ttot(3,4);
37
38 % Configuration of the robot
39 tetax = atan2(-ay, az);
40 ctx = cos(tetax);
41 stx = sin(tetax);
42 tetay = atan2(ax, -ay * stx + az * ctx);
43 tetaz = atan2(ctx * ay + stx * sz, ctx * ny + stx * nz);
44
45 % Displaying the results
46 xf = [Px;Py;Pz;tetax;tetay;tetaz]
47 qsol = [q1;q2;q3;q4;q5;q6];
48 nbsolution = 1;

```

### • mgdmgi.m

```

1  %faire MGD
2  mgd
3
4  %faire MGI

```

```

5  mgd_mgi=1;
6  mgi
7
8  visuart
9  pause
10 viseff
11 pause

```

### • tourner.m

```

1  % script pour tourner autour d une vue 3D
2  [az,el]=view;
3  angle=az;
4  angle1=el;
5  for i = 1:1:100
6  angle=angle+360/100;
7  view([angle,el]);
8  pause(0.001);
9  end;
10 %for i = 1:1:100
11 %angle1=angle1+360/100;
12 %view([az,angle1]);
13 %pause(0.001);
14 %end;

```

### • visuart.m

```

1  hold on;
2
3  for indice=1:1:nbsolution
4
5  q1m=qsol(6*(indice-1)+1);
6  q2m=qsol(6*(indice-1)+2);
7  q3m=qsol(6*(indice-1)+3);
8  q4m=qsol(6*(indice-1)+4);
9  q5m=qsol(6*(indice-1)+5);
10 q6m=qsol(6*(indice-1)+6);
11
12 % Trace des origines des reperes dans le repere atelier
13 O0=[0;0;0];
14 O1=[q1m;0;0];      % a verifier le sens pris dans Denavit
15 O2=[q1m;q2m;0];    % a verifier le sens pris dans Denavit
16 O3=[q1m;q2m;q3m];  % a verifier le sens pris dans Denavit
17 OE=[q1m;q2m;q3m];  % a verifier le sens pris dans Denavit
18
19 plot3(O0(1),O0(2),O0(3),'ro');
20 plot3(O1(1),O1(2),O1(3),'ro');
21 plot3(O2(1),O2(2),O2(3),'ro');
22 plot3(O3(1),O3(2),O3(3),'ro');
23
24 %premier segment du robot
25 X=[O0(1),O1(1)];
26 Y=[O0(2),O1(2)];
27 Z=[O0(3),O1(3)];
28 segment1=line(X,Y,Z);
29 set(segment1,'color','g');
30
31 %deuxieme segment du robot
32 X=[O1(1),O2(1)];

```

```
33 Y=[O1(2),O2(2)];
34 Z=[O1(3),O2(3)];
35 segment2=line(X,Y,Z);
36 set(segment2,'color','m');
37
38 %troisieme segment du robot
39 X=[O2(1),O3(1)];
40 Y=[O2(2),O3(2)];
41 Z=[O2(3),O3(3)];
42 segment3=line(X,Y,Z);
43 set(segment3,'color','c');
44
45 pause
46 end;
```