Direct and Inverse Kinematics of Serial Manipulators - 1st Lab Assignment

Manuel Madeira (83836), Maria Teresa Parreira (83844)

March 27, 2019

1 INTRODUCTION

The present Laboratory Assignment concerns a serial manipulator with 6 degrees-of-freedom (dof). By considering the robot's dimensions but excluding its physical limitations, the objective was to formulate a Direct Kinematics and an Inverse Kinematics function that would, respectively, compute the position and orientation (Euler angles) of the manipulator given a set of angles in the joint space and compute a set of joint angles based off on a set of coordinates (position and orientation) in the work space. After an initial phase of reference frame assignment, where the geometry of the robot and its range of motion were taken into account, these functions were written in MATLAB, recurring to homogeneous transformation matrices, the D-H Convention and to a hybrid geometrical-algebraic approach in the case of the Inverse Kinematics.

Furthermore, a small, mostly intuitive study of the existing singularities in the movement of the manipulator was performed.

2 FRAME ASSIGNMENT

A pivotal part of kinematics is frame assignment, which will allow for the representation of each of the degrees of freedom of the serial manipulator. In order to properly capture the rotations of the joints and links, the frames were positioned and oriented according to the Denavit-Hartenberg Convention. In this convention, the transformation between two frames is defined by 4 parameters, referring to rotations and translations along two axis:

- a_i - Distance from Z_i to Z_{i+1} along X_i.
- α_i - Angle between Z_i and Z_{i+1} around X_i.
- d_i - Distance from X_{i-1} to X_i along Z_i.
- θ_i - Angle between X_{i-1} and X_i around Z_i.

Our model makes use of 9 frames. Frame 0 is set at the base of the manipulator. Two frames are auxiliary frames, meaning that they are not directly linked to any of the degrees of freedom but rather are intermediate steps and help representing translations. The disposition of the frames is represented on Fig.2.1.

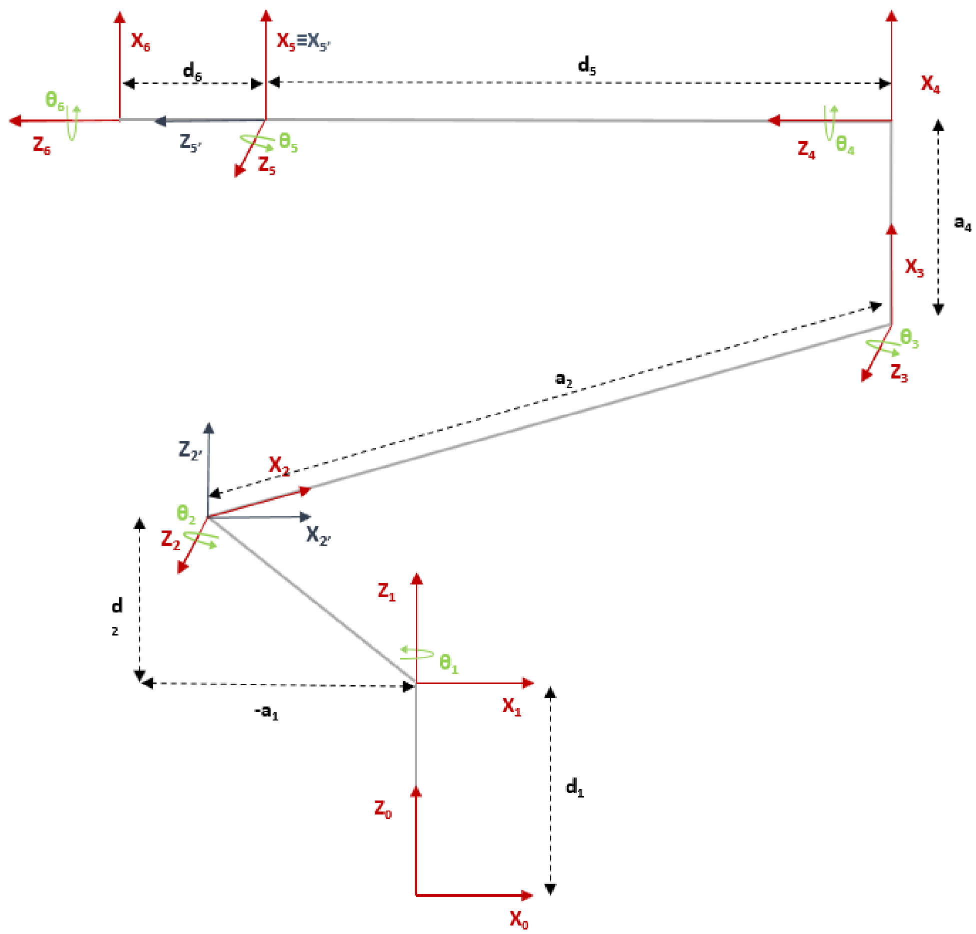

Figure 2.1: Frame assignment used in 6-dof manipulator.

Having assigned the frames, the following step involved determining the D-H parameters in detail. We built a "D-H Matrix", that summed up the transformations between frames as a function of α_{i-1}, a_{i-1}, d_i and θ_i. The distances in the serial manipulator were named according to the D-H parameter they represent. Both the matrix and distances are displayed on Table 3.1.

Table 3.1: D-H Matrix, with parameters displayed per frame (left) and distance parameters in mm (right).

Frame	α_{i-1}	a_{i-1}	d_i	θ_i
1	0	0	d_1	θ_1
$2'$	0	$-a_1$	d_2	0
2	$\pi\backslash 2$	0	0	θ_2
3	0	a_2	0	$\pi\backslash 2+\theta_3$
4	$-\pi\backslash 2$	a_4	0	θ_4
$5'$	0	0	d_5	0
5	$\pi\backslash 2$	0	0	θ_5
6	$-\pi\backslash 2$	0	d_6	θ_6

Parameters	Length (mm)
d_1	47
a_1	30
d_2	45
a_2	120
a_4	20
d_5	130
d_6	29

3 Direct Kinematics

In a serial structure, the composition of all the link transformations is called direct kinematics. It translates information about the angle of each dof into the position and orientation of the end effector of the serial manipulator. Therefore, the function computed received as input an array of length 6 (one angle for each dof) and its output was a set of 6 variables : x, y,z, α, β and γ - the first three refer to a position in space, according to frame 0 (at the base) and the latter three refer to the orientation of the end effector in relation to the orientation of frame 0 (Euler angles).

From the D-H parameters, one can build homogeneous matrices, representing the link transformations. Below is the generalized formula for these matrices (c_{x_i} and s_{x_i} represent the cosine and sine of angle x_i):

$$
{}^{i-1}_{i}T = \begin{pmatrix} c_{\theta_i} & -s_{\theta_i} & 0 & a_{i-1} \\ s_{\theta_i} c_{\alpha_{i-1}} & c_{\theta_i} c_{\alpha_{i-1}} & -s_{\alpha_{i-1}} & -s_{\alpha_{i-1}} d_i \\ s_{\theta_i} s_{\alpha_{i-1}} & c_{\theta_i} s_{\alpha_{i-1}} & c_{\alpha_{i-1}} & c_{\alpha_{i-1}} d_i \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.1}
$$

Obtaining the transformation matrix from the base to the end-effector is then just a matter of multiplying the matrices for each frame, which are built according to matrix 3.1 and using the parameters of Table 3.1.

$$
{}^{tool}_{base}T = {}^{0}_{1}T * {}^{1}_{2'}T * {}^{2'}_{2}T * {}^{2}_{3}T * {}^{3}_{4}T * {}^{4}_{5'}T * {}^{5'}_{5}T * {}^{5}_{6}T \tag{3.2}
$$

This will result in a structure similar to that of 3.1, in which ${}^{tool}_{base}T_{1:3,1:3}$ corresponds to the rotation matrix and ${}^{tool}_{base}T_{1:3,4}$ refers to the position in respect to frame 0.

The information on (x, y, z) is explicit in the position given by the final matrix obtained. Regarding the information about the orientation of the end effector, α, β and γ are angles referring to rotations around moving axis (Euler angles). Consequently, a rotation convention should be picked (in our case, Z-Y-X) and the angles mentioned can be extracted from the rotation matrix, with some algebraic manipulation - note that α will refer to the rotation around Z, β to the rotation around Y (not the original orientation of Y, but the one after the first rotation around Z) and γ to the rotation around the X axis (wherever it is after the two previous rotations).

4 Inverse Kinematics

Inverse Kinematics is a process by which coordinates in the workspace $(x, y, z, \alpha, \beta, \gamma)$ can be transformed into coordinates in the joint space (θ_i)

Two main approaches are considered in this process: Geometrical Approach - which makes use of basic trigonometry relations and planar geometry - and Algebraic Approach - in which the system is reduced to a set of equations. However, hybrid approaches can be used and, in this case, the Geometrical Approach was used to obtain the values of θ_1,θ_2 and θ_3 and, from then, θ_4,θ_5 and θ_6 were derived using the Algebraic Approach.

One of the main problems of Inverse Kinematics is dealing with the multitude of solutions - in practice, this means that the robot has many configurations that are able to achieve the desired position and orientation of the end-effector; mathematically, this stems from using inverse trigonometric functions to obtain the value of θ_i. There are, in total, 8 possible solutions:

- θ_1 has a maximum of two solutions
- for each θ_1, there are two possible values for (θ_2,θ_3).
- for each of the four triplets (θ_1,θ_2,θ_3), there are two values for the triplet (θ_4,θ_5,θ_6).

Initially, the coordinates of the origin of frame 5 are obtained by considering only a translation between that point and the origin of frame 6.

$$
{}^{0}_{5}T * {}^{5}_{6}T = {}^{0}_{6}T \tag{4.1}
$$

Assuming that there is no rotation in this transformation, ${}^{0}_{5}R = {}^{0}_{6}R$ and, knowing that ${}^{5}_{6}P = (0, 0, d_6)$, we get:

$$
\begin{pmatrix} {}^{0}_{5}R & {}^{0}_{5}P \\ 0 & 1 \end{pmatrix} * \begin{pmatrix} {}^{5}_{6}R & {}^{5}_{6}P \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} {}^{0}_{6}R & {}^{0}_{6}P \\ 0 & 1 \end{pmatrix} \tag{4.2}
$$

$$
{}^{0}_{5}P = -{}^{0}_{5}R * \begin{pmatrix} 0 \\ 0 \\ d_6 \end{pmatrix} + {}^{0}_{6}P \tag{4.3}
$$

The value of θ_1 can be obtained as the arctan2 of x_5 and y_5. However, due to the high mobility of our robot and the fact that physical limitations of its movement are being ignored, there can actually be two values for

θ_1 : the one obtained directly via the arctan2 and one obtained by adding π to the first value; we can name these configurations based on the 90-degree angle between a_4 and d_5 being, respectively, *right-armed* or *left-armed*.

Fig. 4.1 illustrates the four possible configurations of the arm for the first situation (*right-armed configuration*). The angles portrayed allow for the determination of θ_2, θ_3.

- t_aux1 *(not represented)* is the angle between d_{2_5} and an horizontal axis placed at the beginning of a_2 (reference frame 2);
- t_aux2 is the angle between a_2 and d_{3_5} , within the triangle [a_2,d_{3_5}, d_{2_5}];
- ϕ is the angle between a_4 and d_{3_5};
- $\theta_A = \pi\backslash 2 + \theta_3$ and it is equal to ($\pi-$t_aux2$-\phi$) for the two configurations on the left ($\theta_A \in [-\phi, \pi - \phi]$) and equal to ($\pi+t_aux2-\phi$) for both configurations on the right ($\theta_A \in [-\pi - \phi, -\phi]$) .

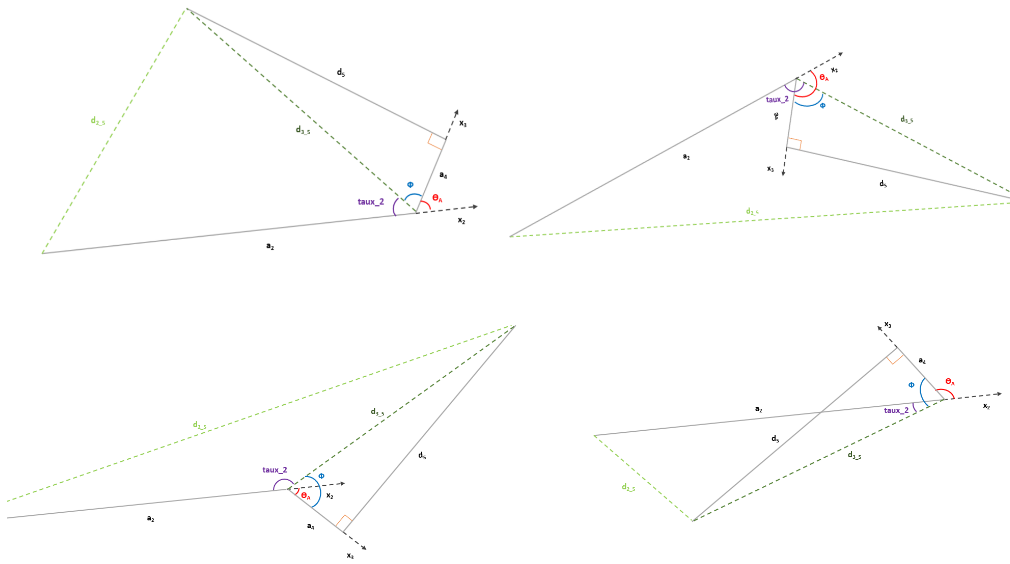

Figure 4.1: Four possible geometrical configurations for the robot, in the right-armed configuration.

For these four configurations, one must take into account that the angles should be measured anticlockwise, as the z-axis of the relevant frames will be pointing outwards ("outside" the page). The configurations (4) for the left-armed configuration are not displayed as the expressions for the values of θ_2, θ_3 are analogous.

Therefore, after getting 4 sets of values for $(\theta_1, \theta_2, \theta_3)$, the Algebraic Approach is used to get the remaining angles.

The knowledge of the first three dof allows us to build the homogeneous transformation matrix ${}^0_3T$, using the D-H matrix defined initially (Table 3.1). From 3.2 we can reformulate:

$$ {}^{tool}_{base}T = {}^0_6 T = {}^0_3 T * {}^3_6 T \tag{4.4}$$

$$ {}^0_3T^{-1} * {}^0_6 T = {}^3_6 T \tag{4.5}$$

$$ \begin{pmatrix} {}^0_3R^T & -{}^0_3R^T * {}^0_3P \\ 0 & 1 \end{pmatrix} * {}^0_6 T = {}^3_6 T \tag{4.6}$$

where the properties of the rotation matrix were taken into account (${}^0_3R^{-1} = {}^0_3 R^T$). ${}^3_6T$ is given by:

$$ {}^3_6T = \begin{pmatrix} c_4c_5c_6 - s_4s_6 & -c_4c_6s_6 - s_4c_6 & -c_4s_5 & -d_6c_4s_5 + a_4 \\ s_5c_6 & -s_5s_6 & c_5 & d_6c_5 + d_5 \\ -s_4c_5c_6 - c_4s_6 & s_4c_5s_6 - c_4c_6 & s_4s_5 & d_6s_4s_5 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4.7}$$

c_i, s_i are the cosine and sine of the angle θ_i. The values of $\theta_4, \theta_5, \theta_6$ are derived from 4.6 by using ${}^3_6T_{2,3}$ (for θ_5), ${}^3_6T_{2,1}$, ${}^3_6T_{2,2}$ (for θ_6), and ${}^3_6T_{1,3}$, ${}^3_6T_{3,3}$ (for θ_4).

Beyond the multiple solutions, there are other concerns when using the Inverse Kinematics method. For instance, the coordinates inserted might be out of reach for the robot. When calculating the variables on the joint space, there may be situations in which there are infinite solutions to a certain variable - these cases will be further explored in the next section.

5 Singularities

The robot is said to be in a **singularity** whenever it loses degrees of freedom. This happens when the motion of a joint or combination of joints doesn't produce any effect on the position and orientation of the end effector, meaning that the robot is unable to move itself in a direction or a set of directions in the work space. Mathematically, this phenomenon implies that the determinant of the Jacobian Matrix of the transformation is zero, *i.e.*, a situation where the matrix is rank deficit.

For the robot studied, the following situations should be pointed out:

- **Alignment of rotational joints**: when two or more of rotational joints (*i.e.* joints in which the rotation axis (Z) is along the link with the next frame) are aligned, we have infinite solutions for their θ angles which result in the same position and orientation for the end effector. For our robot, we have the following possible cases:

  a) **z_1 aligned with z_6 (Fig. 5.1.a)** - this condition is necessary and sufficient to lead us to a situation of *orientation singularity* given that, if we compensate the rotation around z_6 with a symmetric rotation around z_1, we still have the same orientation for the end effector. Moreover, we expect the motion in this configuration to be possible only along Z (analogous to the case of PUMA 560), since, in order to keep the referred alignment, we can only change θ_2 and θ_3 (planar structure) and not arbitrarily (we should keep a relation between these angles). Consequently, this is a *position singularity*, as well.

     In our code, this singularity is noted before the definition of θ_1, by testing if the x and y coordinates of the end effector and those of the origin of frame 5 in relation to frame 0 are all zero. If so, then we're in the presence of the mentioned singularity and the function displays the following warning: "SINGULARITY: There are infinite solutions for the value of theta1 and theta6. By default, theta1 will assume the values 0 and pi. The value of theta6 must take into account other angles, and will be defined accordingly".

  b) **z_4 aligned with z_6 (Fig. 5.1.b)** - this condition is necessary and sufficient to lead us again to a situation of *orientation singularity* given that, if we compensate the rotation around z_6 with a symmetric rotation around z_4, we still have the same orientation for the end effector. However, this one is *not* a position singularity as we can easily conclude that the other angles allow for a 3-dimensional motion (for example, θ_1 enables motion of the end effector along the xy-plane and θ_2 enables motion along Z).

     In our code, this singularity is noted by testing if the cosine of θ_5 is 1 or -1. If so, then we're in the presence of the mentioned singularity and the function displays the following warning:" SINGULARITY: There are infinite solutions for the value of theta4 and theta6. These angles will then assume symmetrical values.";

  c) **z_1, z_4 and z_6 all aligned (Fig. 5.1.c)** - this condition is a conjunction of the two previous conditions. Therefore, it is clear that we're in a *orientation* and *position singularity* case. We should mention that this configuration is slightly different from the completely stretched one: for the latter, d_{3_5} has to be aligned with a_2 and d_6. In our code, since the two previous conditions (from situations **a)** and **b)**) are verified, both warnings are displayed.

  The representation of all the cases mentioned above is made for the case in which the mentioned z-axis all have the same direction (angle between them is 0), but we should refer that the situation is the same if the angle between them is π. We should also mention the interesting case in which z_1 and z_4 align - despite it being the case of rotational joints alignment, it is *not* a singularity as there are no constraints on the end effector's position and orientation (for example, θ_5 and θ_6 are free to move and therefore the end effector has a 3-dimensional range of motion).

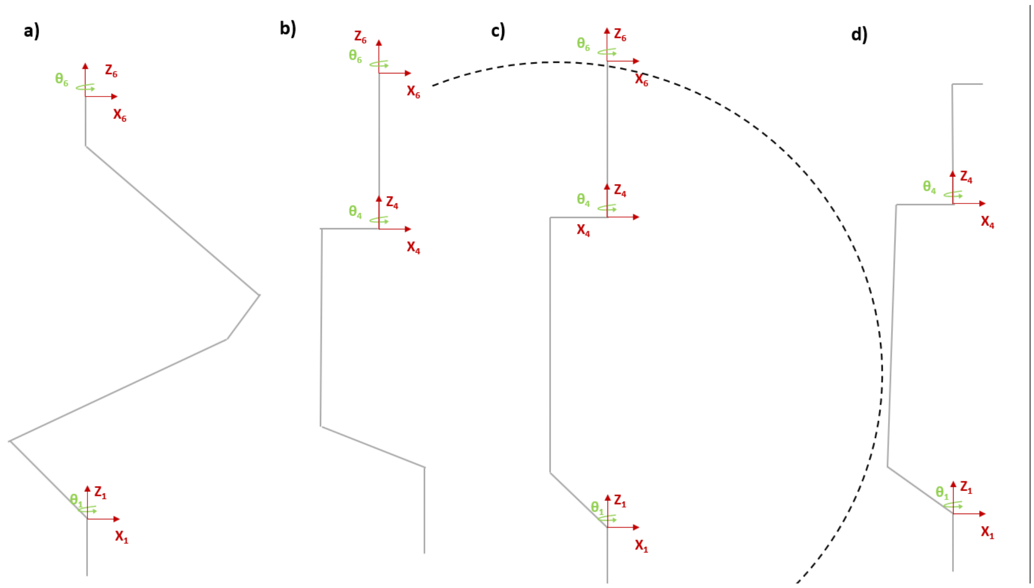

Figure 5.1: a), b), c) - Representations of the singularities of the robot; d) clarification about the alignment of z_1 and z_4

- **Completely stretched arm**: we highlight the case that was mentioned before - if the arm is completely stretched, we're in the presence of a *position singularity* imposed by the robot's physical geometry. In this case, it can't reach any point beyond its own position range. For a given θ_1, there's a circumference centered on the origin of frame 2 which limits that range, as represented in Fig. 5.1.c.
  In our code, this singularity is noted by the geometric method itself: if the absolute value of cosine of t_aux2 (c1) or the absolute value of the cosine of t_aux3 (c3) are higher than 1, it means that the point is out of reach. Therefore, our function "inv_kinematics" assigns the value NaN to the six angles and displays a warning: "for theta1 = ..., the coordinates inserted are out of reach".
  For the case in which the arm is completely withdrawn, we're in an analogous situation (that's why it is not represented), meaning that we're in a *position singularity* as well.

6 Testing

6.1 Default Position

```
>>d_kinematics([0,0,0,0,0,0])
  ans =  -69.0000   -0.0000  112.0000    0.7854   -1.5708   -0.7854
```

This tests the Direct Kinematics code for the case where every θ is 0. Interpreting the result, it is easy to check its correctness. $(-69, 0, 112)$ corresponds to the position of the end effector (see Fig. 2.1 - note that $\theta_2 = 0$ corresponds to an horizontal orientation of a_2). Regarding the orientation, we get the correspondence $(\alpha, \beta, \gamma) = (\pi/4, -\pi/2, -\pi/4)$. Considering the rotations Z-Y-X, we get first a rotation around the Z-axis, of $\pi/4$, then around Y $(-\pi/2)$ and, finally, around X $(-\pi/4)$. The final frame (frame 6) will be oriented as can be seen on Fig.2.1.

```
>>inv_kinematics([-69.0000  -0.0000  112.0000  0.7854  -1.5708  -0.7854])
  ans =
 -3.1416   -1.1547   -0.4276    3.1416    1.5593   -0.0000
 -3.1416   -1.1547   -0.4276   -0.0000   -1.5593    3.1416
 -3.1416    1.7113    0.7329    0.0000    0.6974    3.1416
 -3.1416    1.7113    0.7329   -3.1416   -0.6974   -0.0000
  0.0000    0.0000   -0.0000   -0.0000    0.0000   -0.0000
  0.0000    0.0000   -0.0000    3.1416   -0.0000    3.1416
  0.0000    4.0689    0.3053   -0.0000    1.9090   -0.0000
  0.0000    4.0689    0.3053    3.1416   -1.9090    3.1416
```

Testing the Inverse Kinematics for this case, one can verify that, as expected, the solution ([0,0,0,0,0,0]) is present. The remaining values correspond to different configurations of the arm that will result in the same final position and orientation (although not all are physically achievable). To cross check, we can input the joint values into the Direct Kinematics function:

```
>> d_kinematics([-3.1416   -1.1547   -0.4276    3.1416    1.5593   -0.0000])
   ans = -68.9973    0.0003  112.0003    3.1301   -1.5708   -3.1301
>> d_kinematics([-3.1416   -1.1547   -0.4276   -0.0000   -1.5593    3.1416])
   ans = -68.9973    0.0005  112.0003    2.3562   -1.5708   -2.3562
>> d_kinematics([-3.1416    1.7113    0.7329    0.0000    0.6974    3.1416])
   ans = -68.9981    0.0005  111.9985    0.7854   -1.5708   -0.7854
>> d_kinematics([-3.1416    1.7113    0.7329   -3.1416   -0.6974   -0.0000])
   ans = -68.9981    0.0004  111.9985   -0.6540   -1.5708    0.6540
>> d_kinematics([ 0.0000    0.0000   -0.0000    3.1416   -0.0000    3.1416])
   ans = -69.0000   -0.0000  112.0000    1.5708   -1.5708   -1.5708
>> d_kinematics([0.0000    4.0689    0.3053   -0.0000    1.9090   -0.0000])
   ans = -69.0004   -0.0000  111.9995    3.1416   -1.5708   -3.1416
>> d_kinematics([0.0000    4.0689    0.3053    3.1416   -1.9090    3.1416])
   ans = -69.0004   -0.0002  111.9995    2.8192   -1.5708   -2.8191
```

As can be seen, the values for the final position are approximately the same as in the first example. The orientation is also equivalent to the first one - as long as α and γ are symmetric and β rotates by $(-\pi/2)$, the orientation will match the one on frame 6. The lack of precision is due to the truncation of the output of the Inverse Kinematics.

6.2 STRETCHED FORWARD

```
>> inv_kinematics([-309 0 72  0.7854  -1.5708  -0.7854])

    for theta1 =-3.1416, the coordinates inserted are out of reach.
    ans =
       NaN       NaN       NaN       NaN       NaN       NaN
       NaN       NaN       NaN       NaN       NaN       NaN
       NaN       NaN       NaN       NaN       NaN       NaN
       NaN       NaN       NaN       NaN       NaN       NaN
    0.0000   -3.1416   -2.8363   -3.1416    0.3053    3.1416
    0.0000   -3.1416   -2.8363    0.0000   -0.3053   -0.0000
    0.0000   -2.9819    3.1416   -3.1416    0.1597    3.1416
    0.0000   -2.9819    3.1416    0.0000   -0.1597   -0.0000
```

When the arm is completely stretched forward (x<0) we lose the duality of θ_1 and it can only assume one value. In this case, the Inverse Kinematics function will display a warning and the first 4 solutions (when the robot is *right-armed* - thus $\theta_1 = -\pi$) are arrays of NaN, which illustrates that, in this case, there is a total of only four solutions in the joint space.

6.3 z_1 ALIGNED WITH z_6

```
>> inv_kinematics([0  0  300  0  0  0])

    SINGULARITY: There are infinite solutions for the value of theta1 and theta6. By default,
    theta1 will assume the values 0 and pi. The value of theta6 must take into account other
    angles, and will be defined accordingly.
    ans =
         0    0.5950   -1.4575    3.1416    0.7083   -3.1416
         0    0.5950   -1.4575   -0.0000   -0.7083    0.0000
         0    2.2145    1.7628    0.0000    0.7351   -0.0000
         0    2.2145    1.7628   -3.1416   -0.7351    3.1416
    3.1416    0.5950   -1.4575    3.1416    0.7083    0.0000
    3.1416    0.5950   -1.4575   -0.0000   -0.7083   -3.1416
    3.1416    2.2145    1.7628    0.0000    0.7351   -3.1416
    3.1416    2.2145    1.7628   -3.1416   -0.7351    0.0000
```

In a case in which the orientation is maintained and the position is picked in order to have the end effector frame centered with the origin of frame 0, we're facing a situation where z_1 is aligned with z_6. Therefore, as mentioned by the warning, θ_1 takes two arbitrary values and θ_6 is defined in consistence with the other angles (it depends on the value of θ_1, but also of θ_4/θ_5). Therefore, the situation portrayed in Fig. 5.1.a) can be one of the first two solutions of the function. Furthermore, this example very clearly shows the dependency of θ_6 on θ_4 and θ_5).

6.4 z_4 ALIGNED WITH z_6

```
>> inv_kinematics([-10  0  371  0  0  0])

    for theta1 =3.1416, the coordinates inserted are out of reach.
    SINGULARITY: There are infinite solutions for the value of theta4 and theta6. These
    angles will then assume symmetrical values.
    ans =
       NaN       NaN       NaN       NaN       NaN       NaN
       NaN       NaN       NaN       NaN       NaN       NaN
       NaN       NaN       NaN       NaN       NaN       NaN
       NaN       NaN       NaN       NaN       NaN       NaN
    6.2832    1.4111   -2.8363    3.1416    0.1456   -3.1416
    6.2832    1.4111   -2.8363   -0.0000   -0.1456    0.0000
    6.2832    1.5708    3.1416    0.2450         0   -0.2450
    6.2832    1.5708    3.1416   -2.8966         0    2.8966
```

Given that the tested set of values forces the robot to be almost completely stretched when $\theta_1 = 2\pi \Leftrightarrow \theta_1 = 0$ (last four solutions), when $\theta_1 = \pi$ (first four solutions), the desired position becomes out of the robot's reach - similarly to the situation tested on Section 6.2. However, in this case, the last two solutions with $\theta_1 = 2\pi$ have $\theta_5 = 0$, *i.e.*, z_4 is aligned with z_6. Therefore, we're in a situation of singularity. These two solutions are shown in Fig. 5.1.b and we can verify that θ_4 and θ_6 have symmetrical values. In the remaining solutions (5^{th} and 6^{th}), z_4 and z_6 are not aligned (again, a similar case to that of Section 6.2).

6.5 ARM REACH AND PRECISION

If we want to stretch the arm at its maximum length upwards, we'll have to align z_2 with d_{3_5} and z_6. However, d_{3_5} = 131.52946438... ≈ 130 = $a2$. Therefore, despite their slight differences (change the arguments to: $x = -30$ and $z = 372.52$. Keep all the others), testing this case is redundant with the example 6.4, since the results are similar. We should mention that in this case θ_5 won't be 0 due to the fact that our reach is defined by an irrational number and, therefore, we have to truncate that limit value. However, if we try to use $z = 372.53$ instead, we already receive the warning that the point is out of reach for both θ_1. Actually, our function has precision until the 7^{th} decimal order since for $z = 372.5294643$ it still has one θ_1 with four solutions, but for $z = 372.52946438$ it finally fails and returns no solution.

7 CONCLUSION

The objectives of this laboratory assignment were successfully achieved, in our opinion. Solving the Direct Kinematics and the Algebraic Approach for the Inverse Kinematics consisted mostly of mathematical manipulation within an algorithm; however, some challenging problems came up during our implementation of the Geometrical Approach for the Inverse Kinematics. After overcoming those difficulties, we came up with a solution with a precision of $10^{-10}m$ = 1Å.

As a suggestion, we think that it would be interesting if there was some way of testing our functions directly on the robot.