Laboratory 4

Determination of an inverse geometric model

In the previous works we analysed the problem of calculating the position and orientation of a Cartesian system attached on the end-effector, when the robot coordinates of each joint of the robot is known. This exercise proposes an approach a bit more difficult: we want to determine the joint coordinates for a specific position of the end-effector. In other words, determine a set of coordinates of the robot $q_1, q_2, ..., q_n$, that ensure a specific position and orientation of the end-effector. This problem is known as the **inverse geometric model** of a kinematic chain. The principal objective of this exercise is the description of a heuristic method for determining this inverse geometric model of a robotic arm and to apply this method in two robotic kinematic structures.

4.1 Theoretical considerations

The problem of determining the inverse geometric model of a robot is a non-linear problem. For a robotic structure, we are starting with the numerical values of the matrix that describes the direct geometric model (${}^0T_N$) and we aim at determining the set of joint coordinates of the robot ($q_1, q_2, ...q_n$). For a robot with 6 degrees of freedom (6 motor joints), we obtain a system with 12 equations and 6 unknowns. However, from the 9 equations that result from the correspondence of the orientation from matrix $\frac{T}{6}$, only three of them are independent and the other 6 are redundant. This limits us to the determination of maximum 3 independent variables. If, on these three equations, we add the three equations resulting from the correspondence of the position vectors from matrix $\frac{T}{6}$, we obtain a set of 6 independent equations that allow us to determine a maximum of 6 independent variables. These variables are corresponding to the 6 degrees of freedom of the robot. This system of equations is non-linear and transcendent, often proving difficult to solve. Like all non-linear systems of equations, the main problems are raised by the existence of solutions, existence of multiple solutions and the method to use for solving the system.

Considering a robot for which we denote as $\overline{q} = [q_1,\ q_2,\ ...\ q_n]^T$ the vector of coordinates of the robot, and as $\overline{Z} = [P_X,\ P_Y,\ P_Z,\ \phi,\ \theta,\ \psi]^T$ the vector of Cartesian coordinates of the end-effector, the inverse geometric model (IGM) is represented by a system of equations that construct the dependence $\overline{q} = f(\overline{Z})$. The inverse geometric model must

We need to determine the inverse geometric model since the trajectories of the robotic

arm are described in Cartesian coordinates, while the robot is driven by the motor joints and is therefore described in the joint coordinates. The inverse geometric model is determined based either on the direct geometric model, on heuristic methods (e.g. for simple structures), on iterative methods (e.g. for more complicated structures), or on geometric methods. It is possible to determine either the particular solution of a model or a general method (analytic) solution thereof. Applying heuristic methods is convenient but does not guarantee a unique solution. Thus, two people who use this method to solve the same problem can reach a different solution (although equivalent).

Another problem with the determination of the IGM, is the redundancy of the manipulator. A manipulator is said to be redundant when it is able to reach a specific position and orientation under two or more different configurations of its kinematic chain. This translates mathematically in two or more solutions of the IGM and is linked with the appearance of specific mathematical functions, such as the square root or the cosine, which give usually two solutions: a negative and a positive one.

For example, in figure 4.1 we present a redundant manipulator with two degrees of freedom, and two configurations of its kinematic chain that generate the same position of its end-effector.

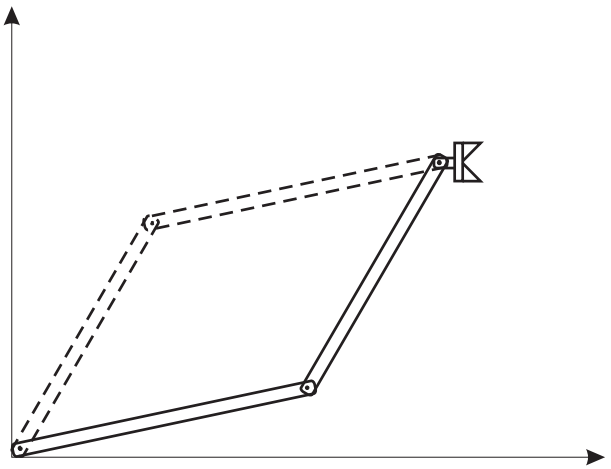

Figure 4.1: Redundant manipulator

The choice of one over the other solution depends on external restrictions and on the construction elements of the robot. However, once a solution has been chosen, it should be used with continued consistency to avoid unnecessary displacement of the robot between the two configurations.

Heuristic method for determining the IGM

I. We equate the homogeneous transformation matrices of the manipulator (direct geometric model) with the general homogeneous transformation matrix (relation 4.6).

If we are seeking a specific position, the homogeneous transformation matrix of the manipulator is equated with a matrix that describes this specific position.

II. We inspect both matrices, observing if:

a. there are elements that contain only one variable

b. there are pairs of elements that can produce expressions with a single variable after a division. We especially use divisions that produce an ***arctan*** function.

c. there are combinations of elements that can be simplified using trigonometric identities.

III. Once we select such an element, we equalise it with its correspondent from the second matrix, and we solve the obtained equation.

IV. We repeat step **III** until we solve all equations that contain elements identified in step **II**.

V. If we obtain redundant or unsatisfactory results, we try again starting from a different equation, keeping the solution aside for later use. We prefer to obtain solutions in function of the vector $P = [P_X,\ P_Y,\ P_Z]^T$ and not in function of the vectors X, Y or Z, since this imposes in general position only and not orientation.

VI. If we cannot identify all the robot joint coordinates, we pre-multiply both parts of the equation with the inverse transformation matrix of the first kinematic joint, obtaining a new set of equations. Alternatively we can try post-multiplication of both parts of the equation with the transformation matrix of the last kinematic joint.

VII. We repeat steps **II** - **VI** until we obtain all the solutions or until we exhaust all the pre- or post-multiplication matrices.

VIII. If we cannot obtain a suitable solution for a variable, we can consider one of the solutions obtained in step **V**.

IX. If we cannot obtain a solution for one of the variables, it means that the manipulator cannot reach the respective position (it is outside of the space of operation of the robot)

For example, for the manipulator with two degrees of freedom from figure 4.2 we can write the direct geometric model as:

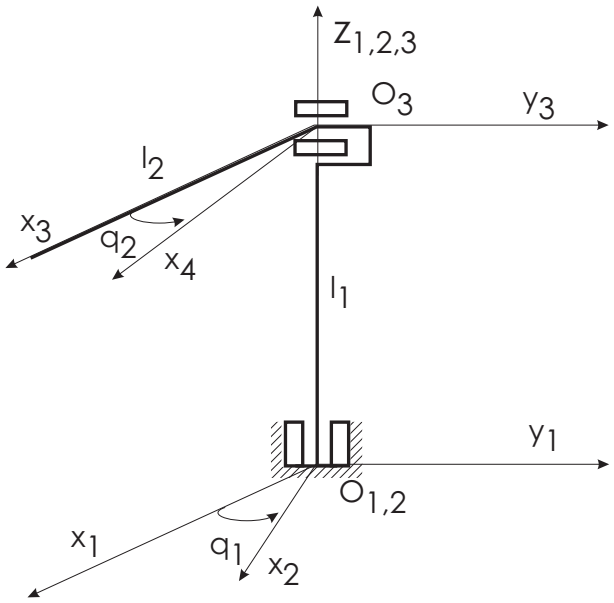

Figure 4.2: Robot RR

$$T_1 = Rot(Z, q_1) = \begin{bmatrix} c_1 & -s_1 & 0 & 0 \\ s_1 & c_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.1}$$

$$T_2 = Trans(Z, l1) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.2}$$

$$T_3 = Rot(Y, q_2) = \begin{bmatrix} c_2 & 0 & s_2 & 0 \\ 0 & 1 & 0 & 0 \\ -s_2 & 0 & c_2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.3}$$

$$T_4 = Trans(X, l_2) = \begin{bmatrix} 1 & 0 & 0 & l_2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.4}$$

resulting in a direct geometric model in form:

$$T = T_1 \cdot T_2 \cdot T_3 \cdot T_4 = \begin{bmatrix} c_1 \cdot c_2 & -s_1 & c_1 \cdot s_2 & c_1 \cdot c_2 \cdot l_2 \\ s_1 \cdot c_2 & c_1 & s_1 \cdot s_2 & s_1 \cdot c_2 \cdot l_2 \\ -s_2 & 0 & c_2 & l_1 - l_2 \cdot s_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.5}$$

The determination of the inverse geometric model, assumes the equation of the direct transformation matrix (T) with the general matrix of a homogeneous transformation (T_g) (see also relation 1.13):

$$T = T_g \tag{4.6}$$

$$\begin{bmatrix} c_1 \cdot c_2 & -s_1 & c_1 \cdot s_2 & c_1 \cdot c_2 \cdot l_2 \\ s_1 \cdot c_2 & c_1 & s_1 \cdot s_2 & s_1 \cdot c_2 \cdot l_2 \\ -s_2 & 0 & c_2 & l_1 - l_2 \cdot s_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} X_X & Y_X & Z_X & P_X \\ X_Y & Y_Y & Z_Y & P_Y \\ X_Z & Y_Z & Z_Z & P_Z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.7}$$

For the determination of the joint coordinates vector of the robot $q = [q_1,\ q_2]^T$ we construct a system of equations:

$$\begin{cases} c_1 \cdot c_2 \cdot l_2 &= P_X \\ s_1 \cdot c_2 \cdot l_2 &= P_Y \\ l_1 - l_2 \cdot s_2 &= P_Z \end{cases} \tag{4.8}$$

$$\frac{P_Y}{P_X} = \frac{s_1}{c_1} \implies q_1 = \arctan\left(\frac{P_Y}{P_X}\right) \tag{4.9}$$

For the determination of q_2 we resort to the pre-multiplication of both parts of the matrix equation (4.7) with the inverse of the transformation matrix of the first kinematic joint (step **VI**), obtaining a new set of equations:

$$T_1^{-1} \cdot T_1 \cdot T_2 \cdot T_3 \cdot T_4 = T_1^{-1} \cdot T_g \tag{4.10}$$

$$\begin{bmatrix} c_2 & 0 & s_2 & c_2 \cdot l_2 \\ 0 & 1 & 0 & 0 \\ -s_2 & 0 & c_2 & l_1 - l_2 \cdot s_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} =$$

$$\begin{bmatrix} X_X \cdot c_1 + X_Y \cdot s_1 & Y_X \cdot c_1 + Y_Y \cdot s_1 & Z_X \cdot c_1 + Z_Y \cdot s_1 & P_X \cdot c_1 + P_Y \cdot s_1 \\ X_Y \cdot c_1 - X_X \cdot s_1 & Y_Y \cdot c_1 - Y_X \cdot s_1 & Z_Y \cdot c_1 - Z_X \cdot s_1 & P_Y \cdot c_1 - P_X \cdot s_1 \\ X_Z & Y_Z & Z_Z & P_Z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.11}$$

resulting in the system

$$\begin{cases} c_2 \cdot l_2 &= P_X \cdot c_1 + P_Y \cdot s_1 \\ s_2 \cdot l_2 &= l_1 - P_Z \end{cases} \tag{4.12}$$

$$\frac{l_1 - P_Z}{P_X \cdot c_1 + P_Y \cdot s_1} = \frac{s_2}{c_2} \implies q_2 = \arctan\left(\frac{l_1 - P_Z}{P_X \cdot c_1 + P_Y \cdot s_1}\right) \tag{4.13}$$

4.2 Proposed problems

1. We consider the robotic structure with 3 degrees of freedom from figure 4.3, for which $l_1 = 0.5\ m$.

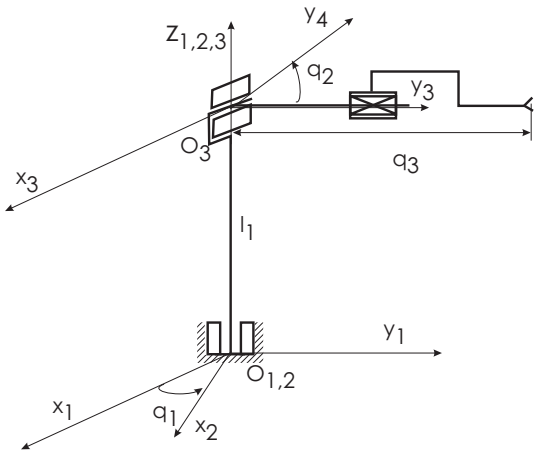

Figure 4.3: Robot RRT

a) Determine the direct geometric model.

b) Determine the inverse geometric model.

2. We consider the robotic structure with 3 degrees of freedom from figure 4.4, for which $l_1 = 0.5\ m, l_2 = l_3 = 0.2\ m$.

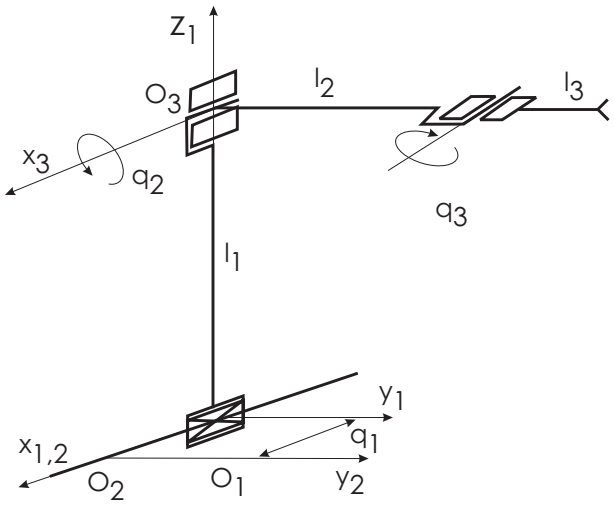

Figure 4.4: Robot TRR

a) Determine the direct geometric model using the DH convention.

b) Determine the inverse geometric model.

c) Construct the robot's model using the robotic toolbox.

d) Using the robotic toolbox, calculate the trajectory for the robot from pose A to pose B. Then animate the trajectory.

$$A = \begin{bmatrix} 0 & 0 & -1 & 0.1414 \\ 0 & 1 & 0 & -0.3 \\ 1 & 0 & 0 & 0.8414 \\ 0 & 0 & 0 & 1 \end{bmatrix}, B = \begin{bmatrix} -1 & 0 & 0 & 0.4 \\ 0 & 1 & 0 & 0.2 \\ 0 & 0 & -1 & 0.5 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

4.3 MATLAB toolbox

In the previous laboratory, we saw how to define link and how to combine them into a robotic structure, using the *Link* and *SerialLink* commands. We also saw how to calculate the end effector position and orientation using the *fkine* and how to visualise the robot using the *plot* commands.

The toolbox has a lot of useful commands for solving the inverse kinematics model as well. In the examples above, it is easy to calculate the inverse kinematics models by hand, but for more complex robots, we need to solved it numerically. The toolbox can do this using the *ikine* command (from inverse kinematics). The command works by providing a desired end-effector pose (position and orientation) and gives back the joint coordinates the result in the desired pose:

```
>> T = transl(0,2,3)*trotx(pi/2);
>> q = bot.ikine(T);
```

As we know, the inverse kinematics model might have more than just one solution for a specific pose. The way numerical methods work, they start looking for a solution around a specific set of joint coordinates, and slowly converge to the joint coordinates that result in the desired pose. To assist the algorithm, we might want to provide an initial guess for the joint coordinates.

```
>> q0 = [0, 0.2, 0.3];
>> q = bot.ikine(T,q0);
```

Unless our robot has at least six degrees of freedom, it will not be always able to obtain the desired pose (why?). In that case, we need to specify which parts of the pose we care about obtaining. We do this using a row vector M which has six element, each one corresponding to a degree of freedom. The first three represent position (X,Y, and Z) and the next three the orientation around X, Y and Z. To specify that we want a specific degree of freedom to be matched, we add 1 in the respective position, otherwise we add 0. For

example, if for a three degree of freedom robot we care about obtaining only position, but orientation is irrelevant, we say that:

```
>> q0 = [0, 0.2, 0.3];
>> M = [1 1 1 0 0 0];
>> q = bot.ikine(T,q0,M);
```

We can solve the inverse kinematics model not just for a specific pose, but for a series of different poses. If our variable containing the desired pose is an array of $4 \times 4 \times N$, then the *ikine* command will solve the inverse kinematics for N different poses. To generate a series of poses starting from an initial pose T_1 until a final pose T_2, in e.g. 50 steps, we can use the *ctraj* command:

```
>> T = ctraj(T1,T2,50)
>> q = bot.ikine(T,q0,M)
```

The result will be an array of $N \times M$, where M is the number of joint coordinates. We can finally feed the result into the *plot* command to visualise the robot's motion from pose T_1 until pose T_2.

```
>> bot.plot(q, 'workspace',[xmin xmax ymin ymax zmin zmax])
```