MODELLING AND SIMULATION OF A PICK&PLACE SYSTEM USING MODELICA MODELLING LANGUAGE AND AN INVERSE KINEMATICS APPROACH

Alberto Monteiro[1], Filipe Pereira[1], Adriano A. Santos[2,3], José Machado[1], Miguel Oliveira[4]*
[1]MEtRICs Research Center, School of Engineering, University of Minho,
Campus of Azurém, 4800-058 Guimarães, Portugal
[2]School of Engineering (ISEP), Polytechnic of Porto (P.Porto),
R. Dr. António Bernardino de Almeida 431, 4249-015 Porto, Portugal
[3]INEGI, Institute of Science and Innovation in Mechanical Engineering and
Industrial Engineering, R. Dr. Roberto Frias, 4200-465 Porto, Portugal
[4]Aveiro-North Polytechnic School, University of Aveiro, 3720-511 Oliveira de Azeméis, Portugal
Email: d8561@dem.uminho.pt

Abstract - *Modelica* is a powerful modelling language for modelling and simulation of Multiphysics Engineering Systems behaviour. This work explores a model to be implemented in the control design of a 6-axis industrial manipulator whose structure approaches the robotic arm *ABB IRB 140*, through inverse kinematics calculation. The significance of an inverse kinematics model is to bring accuracy to the reference inputs into a closed-loop control system involved in an industrial setting. Inverse kinematics can complement the path planner or trajectory generator function in object-oriented *Modelica* language, optimizing the study of an industrial robot physical process such a *Pick&Place* system. Finally, obtained results are extrapolated for similar cases and applications.

Keywords: ABB IRB-140, Inverse Kinematics, Modelica, Pick&Place, Simulation, Six-axis Robot.

1. Introduction

Modelling and simulation are very important steps for obtaining reliable engineering systems [1]-[7]. Some approaches and languages are used and helpful, according different points of view, but *Modelica* language [7] seems one of the best ones for being used for simulation of Multiphysics systems [10],[11]. *Modelica* language is one of the few that present a comprehensive integrated model-based approach, raising the level of abstraction and allowing to perform virtual prototyping by simulating and optimizing system models in an intuitive way before building physical products [12],[13], such as different objects of industrial scope (transmission elements, spring damping, power supply circuits, etc. [14]), and also the reusability of the developed models, and the absence of symbol processing [15].

The subject of this work consists of the development of an inverse kinematics model. The usefulness of object-oriented modelling, implied in the *Modelica* language, and the study of a manipulator system, is greater when it is accompanied by the specific task for which it was programmed and it is here that the inverse kinematics, defining the angular configuration of the multi-axis system, becomes crucial, specially by its complexity and importance in assist the object-oriented modelling of a 6-axis industrial manipulator. Similar works have introduced this topic, that presents an automated design for industrial robots in object-oriented modelling but consisted in only approaching the design framework in a *Modelica* system model [16], without a detailed analysis of the model development equations and due implementation for inverse kinematics. This work addresses the development of the inverse kinematics model that complements and makes possible an accurate control trajectory generation, based on an analysis of the structure and the equations involved.

A robotic manipulator used in industrial setting is usually programmable, with similar functions to a human arm. The links of such a manipulator are connected by joints allowing either rotational motion or translational (linear) displacement. The joints of the manipulator can be considered to form a kinematic chain in which its business end is called the end-effector, and it is analogous to the human hand [17]. The end-effector can be designed to perform any desired task such as gripping, which is verified in *Pick&Place* industrial systems.

When designing an industrial manipulator through object-oriented modelling, the power supply can be neglected and the dynamic model is usually

composed by the mechanical structure, the controller and the trajectory generator or path planner, that dictates the control orders from the main computer to the robot controller [17].

The study or analysis of the physical process resulting from an industrial manipulator, resorting to object-oriented modelling, is only practical if the physical system modelling is accompanied by a well-designed control system, meeting a real sequence of movements, namely the control orders, from motor controllers, preceded by well-designed reference values [18]. Taking the above assumptions in consideration, Gadaleta *et. al.* [18] developed a mechatronic model of an industrial *KUKA KR210 R3100* robot, supported by the capabilities of Modelica/Dymola, where components such as the mechanical structure and actuators, logic and electronics are modelled based on the calculation of the energy consumed by the system.

Inverse kinematics of a robot is an important method to find the joint variables that satisfy the desired position of the robot during its manipulation [22]. The angular values calculated through inverse kinematics, and respectively integrated into a model, are thus the final values of the angles to be reached at the end of the trajectory to be performed. The added value of implementing an inverse kinematics model is that, by recreating the calculations involved in the main computer and transmitted to the controller, it only requires the user to insert the final cartesian coordinates of the end-effector and not the insertion of all the angular variations involved in the axes. An accurate calculation of the joint angles allows precise manipulation of the robot in relation to the base structure and/or the kinematic animation, considering its final physical limits [18].

For the model definition, the industrial manipulator *ABB IRB 140* was the reference structure, that possess six degrees of freedom. The robot has six revolute joints controlled by ac-motors. It is designed specifically for manufacturing industries to perform a wide range of applications such as welding, packing, assembling, *Pick&Place* and other.

The novelty of the article is its significant contribution to the development of an inverse kinematics model for industrial manipulators. By integrating the Modelica language, the study demonstrates the effectiveness of this approach in modeling and simulating multiphysics systems, providing a powerful framework for virtual prototyping and optimization. The practical implementation of the model, together with a detailed theoretical analysis, offers valuable information about the behavior and performance of the Pick&Place system.

In order to achieve the main proposed goals, this paper is structured as follows: In an initial phase it is presented the development of the inverse kinematics model, accompanied by the theoretical component of the calculation of angular variation and its implementation component in C++ (*text view* editor). In a second instance the architecture involved in the connection of the model with parallel models is analyzed, ending finally with a brief analysis of the obtained results, and respective discussion, followed by main achieved conclusions and considered bibliographic references.

2. Inverse Kinematics Issues

The first step for the movement of the manipulator along the operation to occur, as desired, is a correct calculation of the involved values that allow its kinematic movement, namely an accuracy in the presentation of the joint values for each axis (initial joint values and final joint values, in other words, the absolute positioning accuracy [23]). For this purpose, an inverse kinematics model is developed (Figure 1), aiming to calculate the variation of the angular values of the joints, so that the path implied in the operation occurs.

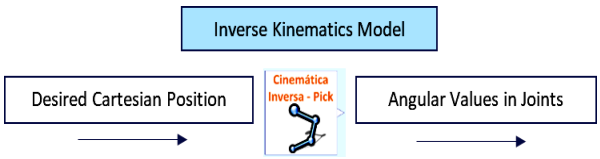


Figure 1: Goal of the Inverse Kinematics Model

The elaboration of this model is performed, in its entirety, using the text view editor, that is, the calculation of the angular variation of the joints is entirely programmed in code.

This model allows simplifying the system right away, since it is easier for the user to know the final cartesian position of the arm tip, in order to perform the Pick, than to know the variation of the angular value of all joints so that the movement occurs as desired (as suggested in the manipulator model present in *Modelica Standard Library* [24]) and the final configuration is reached. This flexibility results from the possibility of using the components provided by the *Modelica Standard Library* which, when combined, can give rise to new models [25].

Since for one operation cycle in a *Pick&Place* system, the implied transport must ensure picking up the package on an input conveyor and dropping the package on another output conveyor, at least two movements must be ensured. In this sense, two

inverse kinematics models are used, one for *Pick* and another for *Place*, where the only differentiating elements between these models are the defined cartesian coordinates (*Pick&Place* points).

Throughout this section it is presented the theoretical component aimed to calculating the inverse kinematics, which is accompanied simultaneously by the implementation of the inverse kinematics model in *OpenModelica* [26],[27].

2.1 Assigning Frames of Reference

This assignment of frames of reference serves as a starting point for the determination of the *Denavit-Hartenberg* (DH) model [28], or the modified *Denavit-Hartenberg* model (MDH) [29], which is based on the dimensions presented by the *IRB-140* industrial manipulator [30]. The different references along the structure is defined by assigning a set of coordinate axes at each link of the structure (Figure 2).

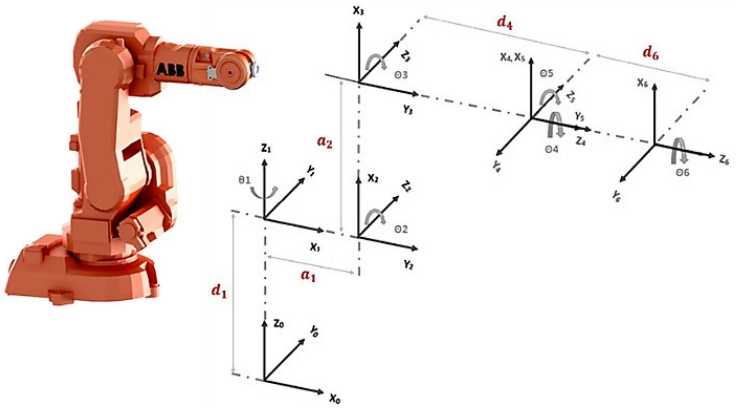


Figure 2 – First excerpt of the code involved in the inverse kinematics model

Each of the dimensions show in Figure 2 dictates the following values (Table 1).

Table 1. Dimensions involved between the different frames of reference

Length	d1	a1	a2	d4	d6
(m)	0.352	0.070	0.360	0.380	0.065

These dimensions have the relevance of being used to define the *Denavit-Hartenberg* parameters, determining the position of the different reference points. However, if the focus is on all the dimensions along the structure, which is useful in the development of the inverse kinematics, they can be described according to the vector {0.352; 0.070; 0.177; 0.360; 0.177; 0.380; 0.065}.

2.2 Denavit–Hartenberg Parameters

Based on Figure 2 it is possible to determine the *Denavit-Hartenberg* parameters, which are present in Table 2.

For each frame of reference there are four parameters, which originate a transformation matrix [31],[32]. This establishes the relationship between the previous reference (i-1) and the following one (i). The parameters a_i and α_i refer to the displacement and rotation on the *X* axis, respectively, considering the following reference.

The parameters d_i and θ_i refer to the displacement and rotation in the *Z* axis, respectively, also taking the next reference into account.

Table 2. Denavit-Hartenberg parameters relating to the distributed coordinate axes for IRB 140 structure

${}^{i-1}T_i$	a_i	α_i	d_i	θ_i
${}^{0}T_1$	0	0	-90	d1
${}^{1}T_2$	90	a1	90	0
${}^{2}T_3$	0	a2	0	0
${}^{3}T_4$	90	0	0	d4
${}^{4}T_5$	-90	0	0	0
${}^{5}T_6$	90	0	0	d6

2.3 Individual Transformation Matrices

Once the *Denavit-Hartenberg* parameters are obtained, one can resort to the matrix shown (Figure 3) in order to obtain each of the individual transformation matrices.

$$ {}^{i-1}T_i = \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) & 0 & a_{i-1} \\ \cos(\alpha_{i-1}).\sin(\theta_i) & \cos(\alpha_{i-1}).\cos(\theta_i) & -\sin(\alpha_{i-1}) & -\sin(\alpha_{i-1}).d_i \\ \sin(\alpha_{i-1}).\sin(\theta_i) & \sin(\alpha_{i-1}).\cos(\theta_i) & \cos(\alpha_{i-1}) & \cos(\alpha_{i-1}).d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Figure 3 - Individual Transformation Matrix

On the other hand, the global transformation matrix of the manipulator is obtained by the product between the individual transformation matrices shown above, according to (1).

$$ {}^{0}T_6 = {}^{0}T_1 \times {}^{1}T_2 \times {}^{2}T_3 \times {}^{3}T_4 \times {}^{4}T_5 \times {}^{5}T_6 \qquad (1) $$

The global transformation matrix (Figure 4) displays in its fourth column the position of the end-effector relative to the base (P_{06}).

$$ {}^{0}T_6 = \begin{bmatrix} & & & P_{06} \\ & & & {}^{0}x_6 \\ & & & {}^{0}y_6 \\ & & & {}^{0}z_6 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Figure 4 - Position of the end-effector relative to the base, taken from the global matrix

From the global matrix it is possible to determine not only the position of the end-effector but also its orientation relative to the base (Figure 5). This orientation is defined by the first three rows and first three columns present in the global matrix, thus defining the matrix *RPY* which is equivalent to the rotation matrix R_{06}, that defines the end-effector orientation relative to the base.

$$ RPY = (r_{ij})_{i,j=1,2,3} = \mathrm{Rot}(z,\phi_t)\,\mathrm{Rot}(y,\psi_t)\,\mathrm{Rot}(x,\theta_t) $$

$$ = \begin{bmatrix} c\phi_t\, c\psi_t & -s\phi_t\, c\theta_t + c\phi_t\, s\psi_t\, s\theta_t & s\phi_t\, s\theta_t + c\phi_t\, s\psi_t\, c\theta_t \\ s\phi_t\, c\psi_t & c\phi_t\, c\theta_t + s\phi_t\, s\psi_t\, s\theta_t & -c\phi_t\, s\theta_t + s\phi_t\, s\psi_t\, c\theta_t \\ -s\psi_t & c\psi_t\, s\theta_t & c\psi_t\, c\theta_t \end{bmatrix} $$

$\hat{x}_6$ $\hat{y}_6$ $\hat{z}_6$

Figure 5 – R_{06} or RPY rotation matrix, based on Roll, Pitch, Yaw, angles, dictates the final orientation of the wrist

The global matrix presented in Figure 4 also has the orientation of the end-effector in its first three rows and columns, where the columns correspond, respectively, to the orientation in x ($\hat{x}_6$), y ($\hat{y}_6$) and z ($\hat{z}_6$).

It is also possible to determine, based on the rotation matrix and through (2) the value of the angles that define the orientation, *Roll* (Φ), *Pitch* (Ψ) and *Yaw* (θ) according to the nomenclature present in (2-i).

$$ RPY = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \qquad \text{(i)} $$

$$ \Psi = atan_2\left(-r_{31}, \sqrt{{r_{32}}^2 + {r_{33}}^2}\right) \qquad \text{(ii)} \qquad (2) $$

$$ \Phi = atan_2(r_{21}, r_{11}) \qquad \text{(iii)} $$

$$ \theta = atan_2(r_{32}, r_{33}) \qquad \text{(iv)} $$

The sufficient condition for solving a six-axis manipulator is the existence of three consecutive revolute axes intersected at a common point, as known as *Pieper* condition [33]. Three consecutive revolute parallel axes are a special case of the above condition, since parallel lines are considered to intersect at infinity, and *ABB IRB 140* obeys this condition.

For the manipulators that meet the *Pieper* condition, it is possible to divide the inverse kinematics into two sub-problems: position and orientation [34],[35]. The calculation of the first three required joint angles is based on the end-effecter position whereas the orientation [36] of the end-effecter allows the calculation of the three final joint angles.

3. Angular Determination and Implementation in the Model

The calculation of each of the angles is now analysed from a theoretical point of view with its practical implementation using *Modelica*, specifically the "*Text View*" section for model development.

Before focusing in detail on each of the angles, the matrices presented in (1) and Figure 5 described in the previous section are implemented. The set of steps performed throughout the implementation is duly described.

- Definition of the length of the links, based on the vector *L* and where the values of the dimensions *d1,a1,a2,d4* and *d6* implied by the *Denavit-Hartenberg* parameters are included;
- Definition of the *Pose_tip* vector, where the coordinates of the final position of the end-effector (P_{06}) are incorporated in the 3 initial values, and the *Roll*, *Pitch* and *Yaw* (RPY) angles are defined in the 3 final values, which are arbitrated with a value of 0;
- Declaration of the rotation matrix R_{06}, based on the declared values for the three angles involved in the end-effector orientation (RPY) and also the matrix associated with the end-effector RPY_{tip} that defines the transformed matrix between the base frame of reference and the end-effector frame of reference;
- Definition of the global matrix ${}^{0}T_6$, based on the rotation matrix ${}^{0}R_6$ and the cartesian position vector of the end-effector (P_{06}).
- Calculation of the wrist position (P_{05}), equivalent to the subtraction between the position specified for the end-effector (P_{06}) and the distance between the end-effector and the wrist (joint 5) multiplied by the z orientation of the rotation matrix RPY (RPY_{tip_z}).

3.1 Determination of Geometric θ_1 and Implementation

The previous determination of the wrist coordinates (P_{05}), allows to obtain geometrically the value of θ_1, based on the projection of the vector p_w in

the xy plane (Figure 6), through the expression (6). In this expression, the importance of using the trigonometric function $arctan_2$ is highlighted, so that the spatial location of the robot is considered when calculating θ_1.

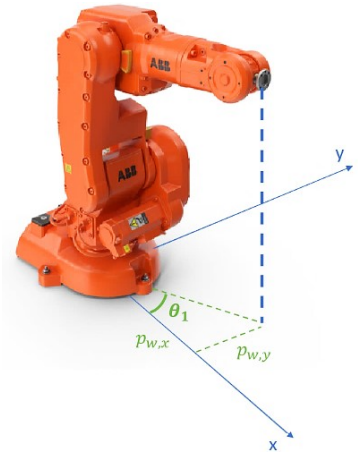


Figure 6 – Projection of vector p_w into the xy plane

$$\theta_1 = \begin{cases} arctan_2(p_{w,y}, p_{w,x}) & se \;\; p_{w,y} \neq 0 \\ 0 & se \;\; p_{w,y} = 0 \end{cases} \quad (3)$$

This is not the only solution for θ_1, since this angle is of rotation about z. Thus, the second solution presents a very simple calculation:

$$\theta_1{}' = \theta_1 + 180^{\circ} \quad (4)$$

The angles responsible for the positioning of the wrist are θ_1, θ_2 and θ_3. It can be taken into consideration that angle 1 defines the positioning of the wrist along the xy plane, while angle 2 and angle 3 define the positioning along the xz plane. Angle 1 is defined as the arctangent of the position of the wrist along the xy plane. If the variation observed along the y axis is close to 0, the angle value can be disregarded and assigned an approximation to 0.

- Defining the values of the two solutions presented for θ_1, $theta1\_L$ (when the left "elbow" is used) and $theta1\_R$ (when the right "elbow" is used).

3.2 Calculation of θ_2 and θ_3 and Implementation

For the calculation of θ_2 and θ_3 the problem can be simplified to a 2-link planar robot (Figure 7), where a vector between the shoulder and wrist are defined. Thus, the determination of θ_2, similarly to θ_1 depends on the previously determined shoulder and wrist coordinates, and the method of obtaining them is purely geometric.

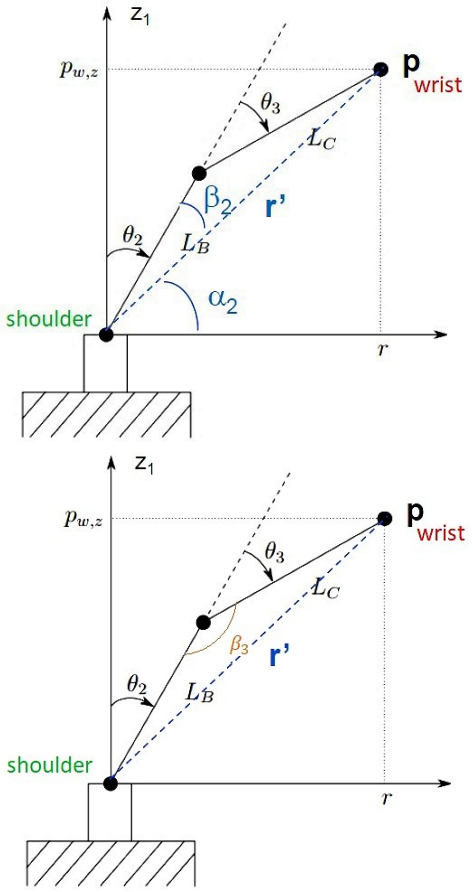


Figure 7 – Simplified problem to a 2-link planar robot (β_2 and β_3 angles represented)

Using the schematic shown and the cosine theorem it is possible to extract the angles β_2 and, consequently, θ_2.

$$\begin{cases} \theta_2 = 90^o - \alpha_2 - \beta_2 \\ \alpha_2 = \arctan({}^B p_{ws,z}, r) \\ L_c^2 = L_B^2 + r'^2 - 2 \times L_B \times r' \times \cos(\beta_2) \end{cases} \quad (5)$$

being that,

$$\beta_2 = \text{acos}\left(\frac{L_B^2 + r'^2 - L_c^2}{2 \times L_B \times r'}\right) \quad (6)$$

only exists if,

$$-1 \leq \frac{L_B^2 + r'^2 - L_c^2}{2 \times L_B \times L_C} < 1 \quad (7)$$

By the cosine theorem it is also possible to determine β_3. This is assumed to be a considered angle that helps in obtaining θ_3. It should also be noted that $r = \sqrt{p_{wrist,x}{}^2 + p_{wrist,y}{}^2}$.

$$\beta_3 = \text{acos}\left(\frac{L_B^2 + L_c^2 - r'^2}{2 \times L_B \times L_C}\right) \quad (8)$$

With β_3 defined, and using the geometric properties of the sketch, it is possible to obtain the equation for angle θ_3:

$$\theta_3 = \beta_3 - 90 \quad (9)$$

The set of steps performed throughout the implementation and aimed at calculating the angles θ_2 and θ_3 are presented:

- Calculation of the length of the vector between the shoulder and the wrist r' ($r3$ in the *script*), according to the following formulation $r3 = \sqrt{{P_{25_x}}^2 + {P_{25_y}}^2 + {P_{25_z}}^2}$. Preceded by this calculation is presented the calculation of P_{25} by subtraction between the vector between the base and the pulse P_{05} and the vector P_{02} defined between the base and the second axis of the manipulator, being $P_{25} = \{P_{25_x},\ P_{25_y},\ P_{25_z}\}$;
- Calculation of $\beta_3 (beta3)$ according to (8), preceded by calculation of the argument $(arg3)\ \frac{L_B^2 + L_C^2 - r'^2}{2 \times L_B \times L_C}$ where L_B and L_C equal the dimensions $a2$ and $d4$, respectively;
- Calculation of the two solutions of θ_3 ($theta3\_L$ e $theta3\_R$). The solution profiling the left elbow is given by (9), while the solution established by the right elbow is calculated using the following formula.

$$\theta_3 = 270 - \beta_3 \quad (10)$$

- Calculation of r' projected along the xy plane (r);
- Calculation of α_2 $(alfa2)$ using the coordinate of vector P_{25} $(P25\_Z)$ and r;
- Calculation of $\beta_2 (beta2)$ according to (9), preceded by calculation of the argument $(arg2)\ \frac{L_B^2 + r'^2 - L_C^2}{2 \times L_B \times r'}$;
- Calculation of the two solutions of θ_2 ($theta2\_L$ e $theta2\_R$). The solution profiling the left elbow is given by (5), while the solution established by the right elbow is calculated using the following formula.

$$\theta_2 = 90^o - \alpha_2 + \beta_2 \quad (11)$$

$$ {}^3\boldsymbol{R_6} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} = \begin{bmatrix} \cos(\theta_4)\cos(\theta_5)\cos(\theta_6) - \sin(\theta_4)\sin(\theta_6) & -\cos(\theta_6)\sin(\theta_4) - \cos(\theta_4)\cos(\theta_5)\sin(\theta_6) & -\cos(\theta_4)\sin(\theta_5) \\ \sin(\theta_5)\cos(\theta_6) & -\sin(\theta_5)\sin(\theta_6) & \cos(\theta_5) \\ -\cos(\theta_4)\sin(\theta_6) - \cos(\theta_5)\cos(\theta_6)\sin(\theta_4) & \cos(\theta_5)\sin(\theta_4)\sin(\theta_6) - \cos(\theta_4)\cos(\theta_6) & \sin(\theta_4)\sin(\theta_5) \end{bmatrix} $$

Figure 8 – Rotation matrix ${}^3R_6$

3.3 Calculation of θ_4 , θ_5 and θ_6 and Implementation

Using the *Denavit-Hartenberg* parameters, it is possible to calculate the first 3 individual transformation matrices and from there extract the corresponding rotation matrix ${}^0R_3$. Considering ${}^0R_6$, the rotation matrix *RPY* (which is known), it is possible to solve the equation for ${}^3R_6$.

$${}^3R_6 = {}^3R_0 \times {}^0R_6 = inv({}^0R_3) \times {}^0R_6 \quad (12)$$

Having obtained the final values of the matrix ${}^3R_6$ and knowing that its elements may be defined in the way illustrated in Figure 8, it's possible to calculate θ_4, θ_5 e θ_6.

Resorting to the element r_{23} and knowing that $\sin(\theta) = \pm\sqrt{1 - \left(cos(\theta)\right)^2}$, the determination of θ_5 becomes possible, whose dual solution can be taken from the following equation.

$$\theta_5 = arctan_2\ (\pm\sqrt{1 - (r_{23})^2}, r_{23}) \quad (13)$$

Similarly, using the formulations taken from matrix present in allows, based on elements r_{33} and r_{13}, to extract the value of θ_4 as well as, using r_{22} and r_{21}, the value of θ_6 .

$$\begin{cases} \theta_4 = arctan2\ (r_{33}, -r_{13}\ ) \\ \theta_6 = arctan2\ (-r_{22}, r_{21}\ ) \end{cases} \quad (14)$$

The deductions of these values present, however, the condition that $\sin(\theta_5) \neq 0$ aimed at avoiding singularities. It is in this sense that the following conditional definition for both angles θ_4 and θ_6 is further considered.

$$\begin{cases} \theta_4 = 0\ \wedge \theta_6 = arctan_2\ (r_{12}, r_{32}) & if\ \theta_5 = 0 \\ \theta_4 = 0\ \wedge \theta_6 = -\ arctan_2(-r_{12}, -r_{32}) & if\ \theta_5 = \pi \end{cases} \quad (15)$$

The set of steps performed throughout the implementation and aimed at calculating the angles θ_4 , θ_5 and θ_6 are presented:

- Statement of the *Denavit-Hartenberg* parameters a_i, α_i, d_i and θ_i for the different frames of reference $(DH\_alpha, DH\_a, DH\_theta, DH\_d)$;
- Calculation of the first three individual transformation matrices ${}^0T_1$, ${}^1T_2$ e ${}^2T_3$ ($T01$, $T12$ e $T23$);
- Calculation of the transformation matrix ${}^0T_3$ ($T03$) and consequent extraction of the rotation

matrix ${}^{0}R_{3}$ ($R03$) involved in the first 3 frames of reference, proceeded by its inverse ${}^{3}R_{0}$ ($R30$);
- Calculation of the matrix ${}^{3}R_{6}$ ($R36$), necessary for the calculation of the 3 angles. Based on the definition of this matrix, each of the constituent elements is subsequently determined (r_{11}, r_{12}, r_{13}, etc);
- Compute the two solutions for θ_5, using the elements present in (13);
- Calculation of the generic solutions for θ_4 and θ_6, as well as the conditional solutions dependent on the value of θ_5;
- Final definition of the joints vector consisting of the 6 angles, being this vector equivalent to the only output defined for the model.

4. Architecture

Once the inverse kinematics model has been developed, it can be connected to a trajectory generation model (Figure 9) where the final angular position values are transmitted as a vector and the trajectory is subsequently calculated based on pre-set initial position values.

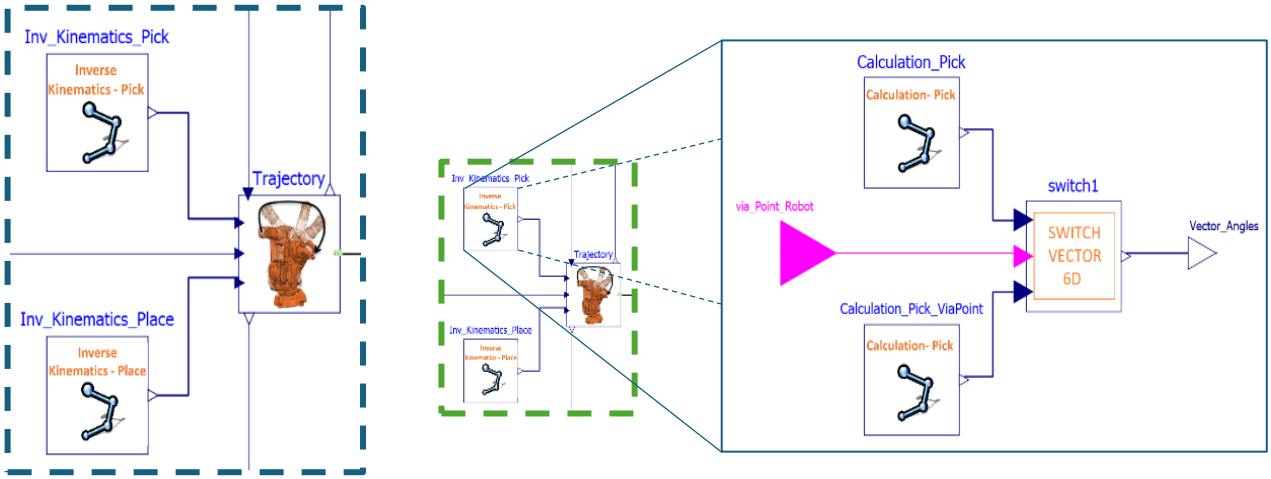


Figure 9 – Connection of two inverse kinematics models with the trajectory generation model (left). Higher hierarchy model in which two inverse kinematics models are incorporated, and only one is activated (right).

The generated trajectory may adopt a trapezoidal velocity profile or a polynomial profile, depending on the design determined by the user.

In Figure 9 it is possible to observe the use of the inverse kinematics model, for the definition of a direct trajectory *Pick* and for the definition of a direct trajectory *Place*. A direct trajectory is defined as a trajectory without *via points* [37]. The use of via points, on the other hand, requires a restructuring of the system architecture, since it requires the execution of a larger number of inverse kinematic models, trying to determine the angular variations required for each of the *via points* along the Cartesian space. One of the ways of implementing *via points* would be, for example, to define a Boolean signal that establishes the existence of *via points* and depending on the state of the signal, determined by the user, the Cartesian point initially considered would change as well as, consequently, the inverse kinematics model to be executed in an initial phase, with the other models for the remaining points, in the case of *via points*, being implemented and executed within the trajectory generation model (for example).

5. Results and Discussion

Once the final values that dictate the angular position in each of the axes for the desired final configuration have been calculated, these values serve as the basis for trajectory generation, entering the respective model. In Figure 10 it is possible to see the evolution of each of the trajectories, determining the rotational movement of the 6 axes for a Pick operation, where the final values of the angles, which dictate the reference values for trajectory generation, are the values generated by the inverse kinematics model.

The generated trajectory, in turn, will be displayed as a set of reference values in the control cycle for each one of the servomotors present in the joints.

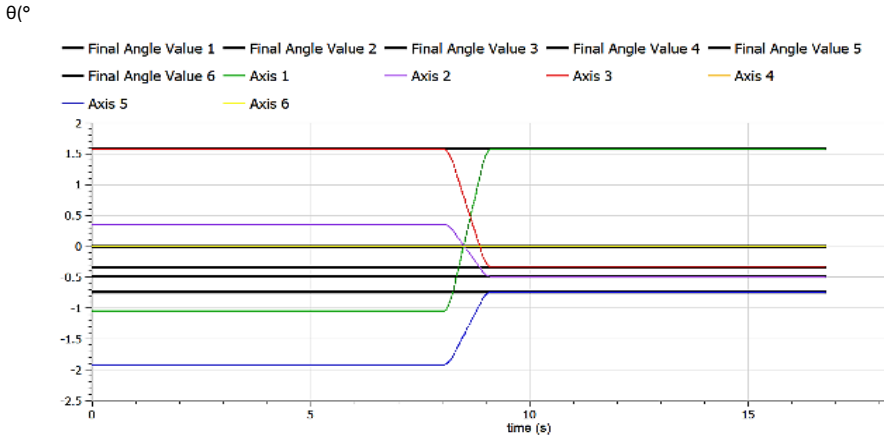
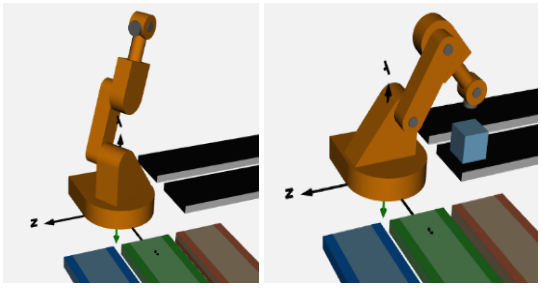
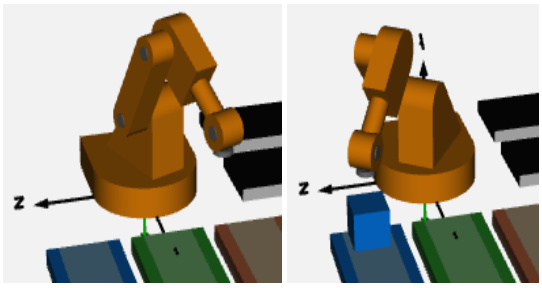


Figure 10 – Evolution of joint angles (colours) to the desired angles (black), with no movement in joints 4 and 6

The animation of the Pick step can be seen in Figure 11(i), in which the initial position of the wrist is randomly established. After the Pick movement and the operation of a gripper to be duly attached to the end of the arm, the Place movement is performed by the manipulator Figure 11(ii)).



(i)



(ii)

Figure 11 – Visual animation that follows the variation of joint values in the Pick(i) and Place (ii) process

The degree through which the inverse kinematics model is used along the system's model depends on the complexity of the movement conceived. A global trajectory in which two via points are considered (one for picking and one for delivering), for example, demands 4 models for each one of the angular values calculations. For this case, the global trajectory command generated can be seen in Figure 12, in which position and velocity are presented for axis 3.

We can thus conclude that only with a correct calculation of the joint angles involved in a movement under study can the trajectory be well designed, thus leading to a more realistic study of the different subsystems adjacent to the industrial manipulator, whether from the structural point of view (mechanical), servomotors (electrical), control (logic blocks) or any other area that one wishes to analyse in depth.

6. Conclusions and Future Work

This work was undertaken to build an inverse kinematic model of the *ABB IRB 140* industrial manipulator, in *OpenModelica,* allowing to study the control design of an industrial 6 axis manipulator, aside from the physical design. The multidisciplinary study of the physical process conferred by the manipulator, in the mechanical and electrical domain, for example, is only relevant if accompanied by the set of movements that best define the real process destined to the robot. In this way, the presence of a model responsible for generating a multi-axis trajectory is mandatory, and whose connection with an inverse kinematics model allows the user to simplify the general input parameters that define the trajectory, since the joint angles required are directly computed. The relevance of using an inverse kinematics model becomes even greater when, throughout the study of the operation, via points are considered, as it requires a considerable number of joint values to be calculated through the different stopping/slowing points of the global trajectory. The development of the inverse kinematics model was therefore successfully implemented in the study of a *Pick&Place* system in *Modelica*, whose reference manipulator was the *IRB 140*.

This research has important implications for industrial automation, increasing the efficiency and accuracy of robotic operations in manufacturing environments.

One of the challenges to future works would be the development of an inverse kinematics model that universally add other types of manipulators. This task will be possible to perform, in a systematic way, if based on the presented developments and achievements of this work.

References

[1] A. Polenghi, L. Fumagalli e I. Roda (2018). Role of simulation in industrial engineering: focus on manufacturing systems, IFAC PapersOnLine, Vol. 51, Issue 11, pp. 496-501. https://doi.org/10.1016/j.ifacol.2018.08.367.

[2] J. M. Aughenbaugh e C. J. Paredis (2008). The Role and Limitations of Modeling and Simulation in Systems Design, in *ASME* International Mechanical Engineering Congress and RD&D Expo, Anaheim, California USA, pp. 13-22. https://doi.org/10.1115/IMECE2004-59813.

[3] J. Machado, E. Seabra, J. Campos, F. Soares and C.P. Leão (2011) Safe controllers design for industrial automation systems. Computers and Industrial Engineering, 60 (4), pp. 635 – 653. https://doi.org/10.1016/j.cie.2010.12.020.

[4] J. Campos, J. Machado and E. Seabra (2008) Property patterns for the formal verification of automated production systems. IFAC Proceedings Volumes (IFAC-PapersOnline), 17 (1 Part 1). Doi: https://doi.org/10.3182/20080706-5-KR-1001.00858

[5] R.-P. Nikula, M. Paavola, M. Ruusunen e J. Keski-Rahkonen (2020). Towards online adaptation of digital twins, Open Engineering, Vol. 10, no. 1, pp. 776–783. https://doi.org/10.1515/eng-2020-0088.

[6] F. Masmoudi, Y. Masmoudi and Y. A. Maalej (2006). Optimization of Product Transfer with Constraint n Robotic Cell Using, The International Journal of Simulation Modelling, Vol. 5, no. 3, pp. 89-100. https://ijsimm.com/Full_Papers/Fulltext2006/text5-3_89-100.pdf.

[7] Santos, A.A., Pereira, F. & Felgueiras, C. Optimization and improving of the production capacity of a flexible tyre painting cell. Int J Adv Manuf Technol (2024). https://doi.org/10.1007/s00170-024-13208-4.

[8] Santos, A.A., Haladus, J., Pereira, F., Felgueiras, C., Fazenda, R. (2024). Simulation Case Study for Improving Painting Tires Process Using the Fanuc Roboguide Software. In: Silva, F.J.G., Pereira, A.B., Campilho, R.D.S.G. (eds) Flexible Automation and Intelligent Manufacturing: Establishing Bridges for More Sustainable Manufacturing Systems. FAIM 2023. Lecture Notes in Mechanical Engineering. Springer, Cham. https://doi.org/10.1007/978-3-031-38241-3_58

[9] Modelica (2023). Modelica Language, Last accessed 20 Octuber 2023. Available: https://modelica.org/language/.

[10] P. J. Zufiria, L. J. Yebra e F. M. Márquez (2020). Port-Hamiltonian modeling of multiphysics systems and object-oriented implementation with the Modelica language, IEEE Access, Vol. 8, pp. 105980-105996. https://doi.org/10.1109/ACCESS.2020.3000129.

[11] A. Smirnov, A. Burt, H. Zhang e I. Celik (2010). Component-based modeling of multi-physics systems, International Journal of Modelling and Simulation, Vol. 30, no 4, pp. 409-415. https://doi.org/10.1080/02286203.2010.11442598.

[12] L. Buffoni, L. Ochel, A. Pop, P. Fritzson, N. Fors, G. Hedin, W. Taha e M. Sjölund (2021). Open Source Languages and Methods for Cyber-Physical System Development: Overview and Case Studies, Electronics (Switzerland), Vol. 10, no. 8, 902. https://doi.org/10.3390/electronics10080902.

[13] Seppo Sierla, Mohammad Azangoo, Kari Rainio, Nikolaos Papakonstantinou, Alexander Fay, Petri Honkamaa, Valeriy Vyatkin (2022). Roadmap to semi-automatic generation of digital twins for brownfield process plants, Journal of Industrial Information Integration, Vol. 27, 100282. https://doi.org/10.1016/j.jii.2021.100282.

[14] Jinjiang Wang, Xiaotong Niu, Robert X. Gao, Zuguang Huang, Ruijuan Xue (2023). Digital twin-driven virtual commissioning of machine tool, Robotics and Computer-Integrated Manufacturing, Vol. 81, 102499. https://doi.org/10.1016/j.rcim.2022.102499.

[15] Lin Zhang, Fei Ye, Kunyu Xie, Pengfei Gu, Xiaohan Wang, Yuanjun Laili, Chun Zhao, Xuesong Zhang, Minjie Chen, Tingyu Lin, Zhen Chen (2022). An Integrated Intelligent Modeling and Simulation Language for Model-based Systems Engineering. Journal of Industrial Information Integration, Vol. 28, 100347. https://doi.org/10.1016/j.jii.2022.100347.

[16] E. Safavi, M. Tarkian e J. Ölvander (2010). Rapid concept realization for conceptual design of modular industrial robots, in Proceedings of NordDesign 2010, the 8th International NordDesign Conference, Göteborg, Sweden. Crossref.

[17] R. Aparnathi e V. V. Dwivedi (2014). The Novel of Six axes Robotic Arm for Industrial Applications, *International Journal of Robotics and Automation,* Vol. 3, No. 3, pp. 161-167. http://doi.org/10.11591/ijra.v3i3.pp161-167.

[18] Canadas N., Machado J., Soares F., Barros C., Varela L. "Simulation of cyber physical systems behaviour using timed plant models" (2018) Mechatronics, 54, pp. 175 - 185, doi: 10.1016/j.mechatronics.2017.10.009

[19] Martins L., Varela M.L.R., Fernandes N.O., Carmo–Silva S., Machado J. "Literature review on autonomous production control methods" (2020) Enterprise Information Systems, 14 (8), pp. 1219 - 1231, doi: 10.1080/17517575.2020.1731611

[20] Pereira, F.; Magalhães, L.; Santos, A.A.; da Silva, A.F.; Antosz, K.; Machado, J. Development of an Automated Wooden Handle Packaging System with Integrated Counting Technology. Machines 2024, 12, 122. https://doi.org/10.3390/machines12020122

[21] Michele Gadaleta, Giovanni Berselli, Marcello Pellicciari (2017). Energy-optimal layout design of robotic work cells: Potential assessment on an industrial case study. Robotics and Computer-Integrated Manufacturing, Vol. 47, pp. 102-11. https://doi.org/10.1016/j.rcim.2016.10.002.

[22] A. A. Hayat, R. Sadanand e S. K. Saha (2015). Robot manipulation through inverse kinematics, in Proceedings of the 2015 Conference on Advances In Robotics, Goa, India, No. 48. https://doi.org/10.1145/2783449.2783497.

[23] Yonghong Deng, Xi Hou, Bincheng Li, Jia Wang, Yun Zhang (2024): A highly powerful calibration method for robotic smoothing system calibration via using adaptive residual extended Kalman filter, Robotics and Computer-Integrated Manufacturing, Vol. 86, 102660. https://doi.org/10.1016/j.rcim.2023.102660

[24] Modelica (2023). Mechanics. MultiBody. Examples. Systems. RobotR3 – FullRobot. accessed 20 October 2023. https://build.openmodelica.org/Documentation/Modelica.Mechanics.MultiBody.Examples.Systems.html accessed on:20.03.2014.

[25] P. Aivaliotis, Z. Arkouli, K. Georgoulias, S. Makris (2021). Degradation curves integration in physics-based models: Towards the predictive maintenance of industrial robots, *Robotics and* Computer-Integrated Manufacturing, Vol. 71, 102177. https://doi.org/10.1016/j.rcim.2021.102177.

[26] Open Source Modelica Consortium (2023). OpenModelica – Introduction. https://openmodelica.org/ accessed on:18.03.2024.

[27] P. Aivaliotis, S. Aivaliotis, C. Gkournelos, K. Kokkalis, G. Michalos, S. Makris (2019): Power and force limiting on industrial robots for human-robot collaboration, Robotics and Computer-Integrated Manufacturing, Vol. 59, pp. 346-360. https://doi.org/10.1016/j.rcim.2019.05.001.

[28] A. A. Hayat, R. Chittawadigi, A. D. Udai e S. K. Saha (2013). Identification of Denavit-Hartenberg Parameters of an Industrial Robot, in *AIR '13:* Proceedings of Conference on Advances In Robotics, pp. 1-6, Pune, India. https://doi.org/10.1145/2506095.2506121.

[29] Mohammadreza Dehghani, Ryan A. McKenzie, Rishad A. Irani, Mojtaba Ahmadi (2023): Robot-mounted sensing and local calibration for high-accuracy manufacturing, Robotics and Computer-Integrated Manufacturing, Vol. 79, 102429. https://doi.org/10.1016/j.rcim.2022.102429.

[30] ABB (2023). Product specification - IRB 140 (Revision: Q). https://library.e.abb.com/public/a7121292272d40a9992a50745fdaa3b2/3HAC041346%20PS%20IRB%20140-en.pdf accessed on:19.12.2023.

[31] J. J. Craig (2018). Introduction to Robotics: Mechanics and Control (4td Edt.), Pearson Prentice Hall, ISBN: 0133489795.

[32] B. Siciliano, L. Sciavicco , L. Villani e G. Oriolo (2010). Robotics: Modelling, Planning and Control, Springer Science & Business Media. ISBN: 1846286417.

[33] D. L. PIEPER (1969). The kinematics of manipulators under computer control. Ph.D. Dissertation. Stanford University. https://www.proquest.com/openview/b6d841ffecfe56af9a84ac48038472ac/1?pq-origsite=gscholar&cbl=18750&diss=y accessed on: 23.03.2024.

[34] J. N. Pires (2007), Industrial Robots Programming: Building applications for the factories of the future. Springer Science+Business Media, 2007. https://doi.org/10.1007/b101252.

[35] J. Uricek, T. Galbavy, V. Bulej e P. Durec (2014). The Calculation of Inverse Kinematics for 6DOF Serial Robot, Journal Komunikácie, Vol. 16, No. 3, pp. 154-160. https://komunikacie.uniza.sk/pdfs/csl/2014/11/24.pdf.

[36] Almaged, M. (2017): Forward and Inverse Kinematic Analysis and Validation of the ABB IRB 140 Industrial Robot, International Journal of Electronics Mechanical and Mechatronics Engineering, Vol. 7(2), pp. 1383-1401. https://dergipark.org.tr/en/pub/ijemme/issue/28650/319197.

[37] R. Zhao, D. Sidobre e W. He (2014). Online Via-Points Trajectory Generation for Reactive Manipulations, in IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM), Besançon, France, pp. 1243-1248. https://doi.org/10.1109/AIM.2014.6878252.