mathematics

Article

The Importance of Embedding a General forward Kinematic Model for Industrial Robots with Serial Architecture in Order to Compensate for Positioning Errors

Cozmin Cristoiu *, Mario Ivan, Ionuţ Gabriel Ghionea and Cristina Pupăză

Faculty of Industrial Engineering and Robotics, University Politehnica of Bucharest, Splaiul Independentei No. 313, 060042 Bucharest, Romania; andrei_mario.ivan@upb.ro (M.I.); ghionea.gabriel@upb.ro (I.G.G.); cristina.pupaza@upb.ro (C.P.)
* Correspondence: cozmin.cristoiu@upb.ro

Abstract: This paper proposes a methodology for creating simplified structural schemes and forward geometric models for industrial robots with serial architecture, with the goal of reducing thermal deformation errors that negatively impact positioning accuracy during operation. Unlike classical approaches, the proposed methodology introduces modifications to the order of matrix multiplication and incorporates new parameters to create a forward geometric model that better corresponds to the deformation characteristics of these robots. Details are presented on how to build and employ this extended model and integrate it into a thermal error compensation algorithm. The implementation of the algorithm in a software application is presented along with experimental results that demonstrate its effectiveness. This work addresses a real phenomenon that occurs in industrial robot operation and has implications for improving the performance of robots in manufacturing applications.

Keywords: industrial robot; error compensation; general model; simulation; thermal deformation; forward kinematics

MSC: 70B15

check for updates

Citation: Cristoiu, C.; Ivan, M.; Ghionea, I.G.; Pupăză, C. The Importance of Embedding a General forward Kinematic Model for Industrial Robots with Serial Architecture in Order to Compensate for Positioning Errors. *Mathematics* **2023**, *11*, 2306. https://doi.org/10.3390/math11102306

Academic Editor: António Lopes

Received: 31 March 2023
Revised: 8 May 2023
Accepted: 10 May 2023
Published: 15 May 2023

1. Introduction

Industrial robots are widely used in various applications, such as material handling, machining, welding, and assembly, due to their high flexibility, precision, and efficiency. However, during operation, these robots suffer from thermal deformations that negatively impact their positioning accuracy. This paper addresses the problem of reducing thermal deformation errors by proposing a methodology for creating a simplified geometric model for industrial robots with serial architecture, which considers the way in which these robots deform during operation. The control and programming of industrial robots relies on a geometric description of the robot in a digital environment, which typically includes the dimensions of the robot's segments, the locations of the joints or links between the elements, and the angles of inclination or rotation of the segments. This simplified schematic representation is very efficient in terms of the mathematical description of some robot models and facilitates the calculation of their position, but unfortunately this is not enough. In actual operation, the robot may be affected by positioning errors. Positioning deviations of industrial robots can arise from various factors, such as assembly errors, geometric deviations of the components, elastic deformations, or thermal expansion [1]. Overlooking these errors may cause defects in the manipulated parts or in high-precision processes such as assembly, machining, or laser cutting. To address this issue, this paper proposes a novel algorithm for creating a geometric model of industrial robots with serial architecture. Using this procedure, deformations of the robot structure are included into the model, thus improving the accuracy of the calculations and, in the end, the positioning accuracy of the robot. The paper is structured as follows. The next section encompasses a critical review of the state of the art. Section 3 describes how an extended geometric model can be constructed and how error parameters are included. Section 4 presents the place occupied by the extended geometric model in a whole suite of procedures that together compose a thermal error compensation solution. In Section 5, the advantages of using the extended forward geometric model and the benefits brought by the integration of such a model into a thermal error compensation procedure for industrial robots are emphasized.

2. State of the Art

In software control algorithms, the calculations regarding the position of robots are performed based on matrices in which the dimensions of the robot and the angles of the joints are included as arguments. To make it easier to track and assign in the corresponding matrices, the dimensions of the robot and the angles of the joints are first centralized in a tabular form that is usually called a table of parameters of the robot. Since the robot models can be different, some standardization of the geometrical description of the robot elements and the table of parameters is necessary. A general form was first used in 1955 by Jacques Denavit and Richard S. Hartenberg [2]. This Denavit–Hartenberg (DH) convention has remained unchanged and is still employed today to create forward geometric models. The DH convention uses four parameters, namely the link length, the link twist, the link offset, and the joint angle. They describe the transformation between two adjacent links in a robotic arm. This convention provides a systematic approach for modeling the kinematics of a robot and simplifies the process of developing control algorithms for the robot. Since its introduction, numerous researchers have worked on improving the DH convention and developing more accurate and efficient forward kinematic models for industrial robots. One recent example is the work of Cerrillo, D. et al. [3], who proposed a modified DH convention that incorporates additional parameters to account for joint misalignments and other imperfections in robot design. Their model demonstrated improved accuracy over traditional DH models [3]. Another recent study by Maaroof et al. [4] presented an optimization-based approach for determining the optimal DH parameters for a given robot configuration. The authors demonstrated that their approach resulted in more accurate forward kinematic models than traditional methods [4]. Several other researchers have also proposed modifications and extensions to the DH convention. For example, Zuha et al. [5] presented a modified DH convention to be used in modeling the kinematics of robots with redundant degrees of freedom. Their approach extends the DH convention by incorporating additional parameters to account for the redundant joints [5]. Similarly, Huczala et al. [6] proposed an improved DH convention that incorporates the concept of screw theory. Their model uses the screw parameters to describe the transformation between two adjacent links in a robot arm, providing a more efficient and accurate representation than traditional DH models [6]. Overall, the use of the DH convention for modeling the kinematics of industrial robots remains an actual and effective approach. However, ongoing research is expected to redefine and improve the convention to enhance the accuracy and efficiency of forward kinematic modeling. One area of current research is the development of more accurate models for non-standard robot geometries, such as those with flexible links or complex joint structures. The present paper covers a literature gap focusing on the concept that the robot's elements are deformable, as in real applications. To understand how to consider these deformations, first of all, the general method of calculating the relative position between two segments of a robot with serial architecture is presented, and then the DH method is presented in detail with all the advantages, drawbacks, and shortcomings.

2.1. General Method

To utilize robots in tasks such as part handling, machining applications, or precision assembly, it is imperative to determine the position of the mechanical hand and the tool that the robot will employ. This position is identified as the characteristic point of the robot, and its location is calculated through a mathematical computation based on two sets of information—the segment length of the robot and the position of its joints, including the angles of inclination or rotation. Thus, from a mathematical point of view, any other aspects related to the final form of the elements may be omitted. In this way, some aspects are reduced to a minimum, and a simplified geometric representation is defined, which is called the structural scheme. Figure 1a,b depicts the real and structural representations of a robot [7], with the latter approximating the robot's segments as line segments of equivalent length (Figure 1a). The joint positions and rotation angles are indicated by placing Cartesian coordinate systems at those points, and the robot's characteristic point is located at the end of its final segment.

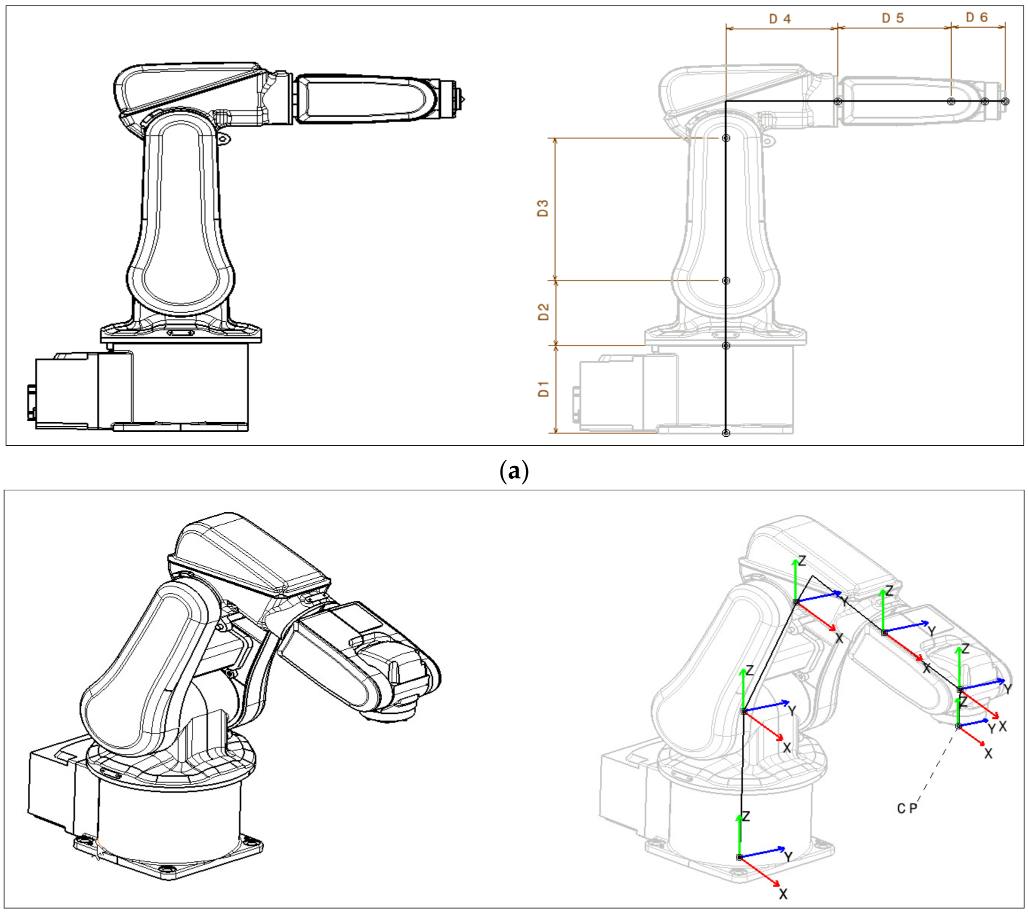


Figure 1. The ABB IR 120 robot model: (**a**) real structure; (**b**) simplified structural diagram [6].

To calculate the position of the characteristic point, from a mathematical point of view, other information about the structure of the robot is not relevant. The mathematical method to determine the position involves the description of each element in the kinematic chain, starting from the base (from the global coordinate system), and then, row by row, the position and orientation of each joint and each individual element is determined until the "top" of the robot is reached. The mathematical description of the locations of the robot's joints is achieved by employing homogeneous coordinate transformations and involves the use of matrices that have two main roles: to describe the translations and to describe the rotations. The matrices that define the translations along the X, Y, and Z axes have the form:

$$\begin{pmatrix} 1 & 0 & 0 & tx \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{1}$$

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & ty \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2}$$

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & tz \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3}$$

where *tx*, *ty*, and *tz* represent the distances between the joints of two successive segments. The matrices that describe the rotations around the X, Y, and Z axes have the form:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & cos\theta & -sin\theta & 0 \\ 0 & sin\theta & cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4}$$

$$\begin{pmatrix} cos\theta & 0 & sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -sin\theta & 0 & cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{5}$$

$$\begin{pmatrix} cos\theta & -sin\theta & 0 & 0 \\ sin\theta & cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{6}$$

The segments of the robot can be moved relative to each other, but also rotated around the joint that connects them. From a mathematical point of view, the geometric description of the position is obtained by composing (multiplying) the matrices of both translation and rotation, obtaining a transformation matrix (usually denoted by T) of the form:

$$T = \begin{pmatrix} \mathrm{R} & \mathrm{p} \\ \eta^T & \sigma \end{pmatrix} \tag{7}$$

This transformation matrix is composed of four sub-matrices as follows: R = fundamental rotational matrix (3×3); p = translation vector (3×1); η = perspective vector (1×3) (in kinematics it is the null vector); σ = scale factor (usually = 1). Starting from the base, row by row, for the position of each joint/segment, these transformation matrices are successively multiplied until the final position of the characteristic point is obtained as follows:

$$T_{\text{base}-\text{end}} = T_{01} \times T_{12} \times T_{23} \times T_{\ldots} \times T_{\ldots} = \begin{pmatrix} R11 & R12 & R13 & Px \\ R21 & R22 & R23 & Py \\ R31 & R32 & R33 & Pz \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{8}$$

2.2. Denavit–Hartenberg Convention

For the kinematic analysis (forward model) of robot mechanisms, the compound homogeneous operators obtained from the product of simple homogeneous operators are important: translation $\times$ rotation, rotation $\times$ translation, and rotation $\times$ rotation. The order in which the operators are applied is essential because their product is not commutative. It should be noted that in a compound operator resulting from the product of several simple ones, each operator acts on the coordinate system obtained after applying the previous operator. A variant (and the most common) of using block matrices is the one that uses the Denavit–Hartenberg notation, which is described in detail in [8]. In brief, the DH and DHM (Denavit–Hartenberg modified) conventions presuppose compliance with the following methodologies.

2.2.1. Classic DH

The notation of parameters and placement of axis systems in order to geometrically describe the position of two successive elements in concordance with DH convention [3] of a robot is described in Figure 2.

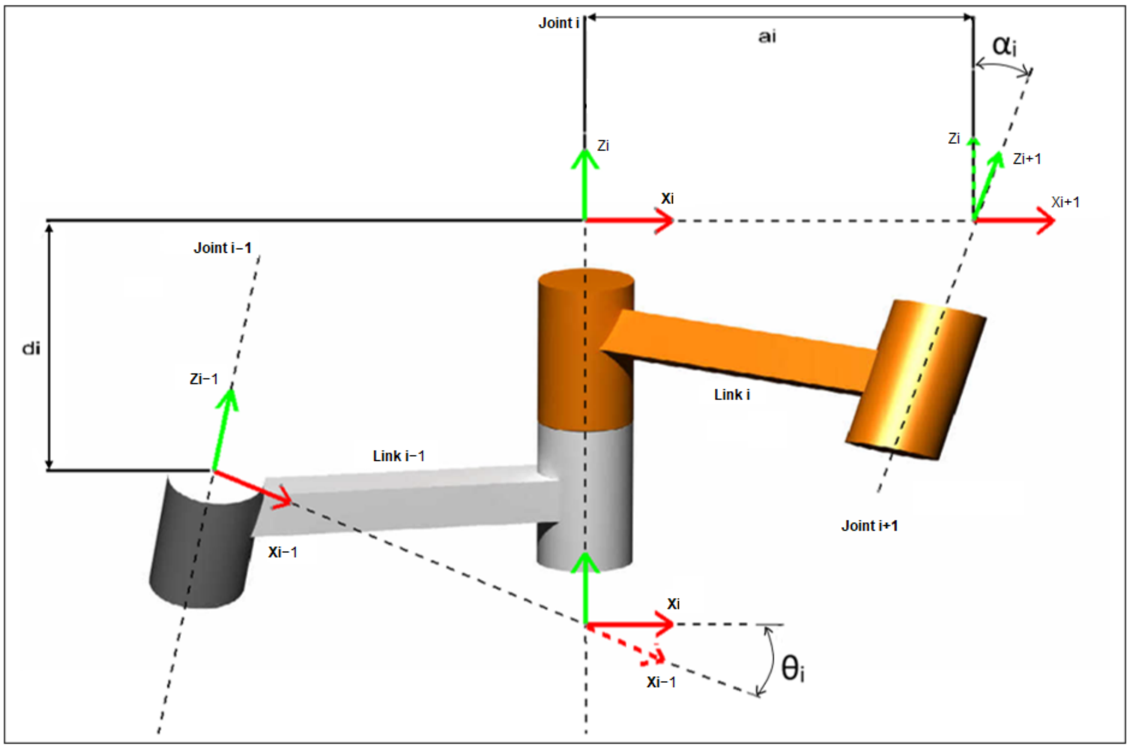


Figure 2. DH convention for axis placement and parameter notation.

The procedure following the DH convention is described by the following steps:

1. draw the axes of the joints;
2. draw the common perpendiculars between the neighboring axes and their points of intersection with the axes of the respective joints;
3. for system {i}, the origin of the system is at the point where the common perpendicular to the axes of the "i" and "i + 1" joints intersects the axis of the "i" joint;
4. the axis Z_i is oriented along the axis of the joint "i";
5. the X_i axis is oriented along the common normal to the axes of the "i" and "i + 1" joints, or if the axes intersect, the X_i axis is normal to the plane of the Z_i and Z_{i+1} axes;
6. the Y_i axis completes the reference system according to the right hand rule;
7. the reference system of the fixed element is denoted by {0}; this system coincides with system {1} when the variable of the first joint is zero;
8. for the {n} system, the location of the origin O_n and the direction of the X_n axis are free, their choice being made so that several parameters of the element are null;
9. a table containing the parameters of the elements is created (Table 1).

Table 1. Example of DH parameters.

Joint Nr.	α_i	a_i	d_i	θ_i/q_i
i	α_{i-1}	a_{i-1}	d_i	θ_i/q_i

Where: a_i is the distance from Z_i to Z_{i+1} measured along X_i; α_i is the angle between Z_i and Z_{i+1} measured around X_i; d_i is the distance between the axes X_{i-1} and X_i measured along Z_i; θ_i is the angle between X_{i-1} and X_i measured around Z_i.

If a joint is both translated and rotated compared to the previous one (so there is a screw displacement), it is common to separate translation and rotation as follows [9]:

$$[Z_i] = Trans_{Z_i}\ (d_i) \cdot Rot_{Z_i}(\theta_i) \tag{9}$$

$$[X_i] = Trans_{X_i}\ (a_{i,i+1}) \cdot Rot_{X_i}(a_{i,i+1}) \tag{10}$$

Using these two notations, each link is described according to DH by a coordinate transformation from the concurrent coordinate to the previous coordinate:

$$T^i_{i-1} = [Z_{i-1}] \cdot\ [X_i] \tag{11}$$

resulting in:

$$T^i_{i-1} = \begin{bmatrix} cos\theta_i & -sin\theta_i cos\alpha_i & sin\theta_i sin\alpha_i & a_i cos\theta_i \\ sin\theta_i & cos\theta_i cos\alpha_i & -cos\theta_i sin\alpha_i & a_i sin\theta_i \\ 0 & sin\alpha_i & cos\theta_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{12}$$

2.2.2. Modified DH (DHM)

The notation of parameters and placement of axis systems in order to geometrically describe the position of two successive elements in concordance with the modified DH convention [10] of the robot is described in Figure 3:

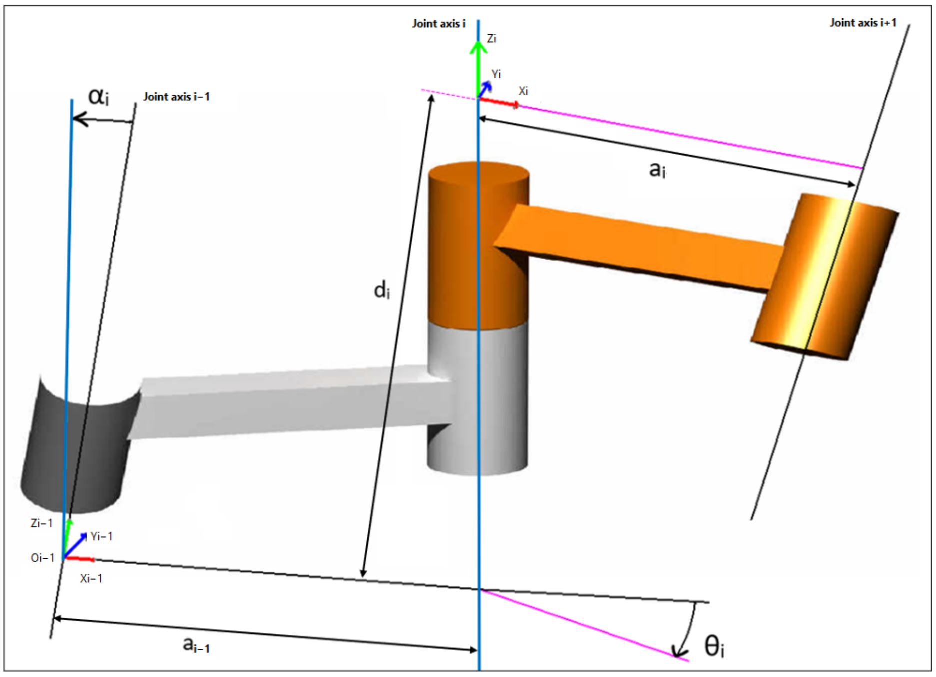


Figure 3. DHM convention for axis placement and parameter notation.

Unlike the standard method, this time, the O_{i-1} coordinate system is attached to the $i-1$ axis and O_i is attached to the i axis. The order of applying the operations is as follows:

1. a rotation around the $\mathrm{Z_{i-1}}$ axis of angle θ_i;
2. a translation along the $\mathrm{Z_{i-1}}$ axis with the distance $\mathrm{d_i}$;
3. a translation along the axis $\mathrm{X_{i-1}}$ rotated and becoming $\mathrm{X_i}$ with distance $\mathrm{a_i}$;
4. a rotation around the axis $\mathrm{X_i}$ with the angle α_i.

Each transformation is represented by a matrix:

$$T_{i-1}^{i} = Rot_{X_{i-1}}\,(\alpha_{i-1}) \times Trans_{X_{i-1}}(a_i) \times Rot_{Z_i}\,(\theta_i) \times Trans_{Z_i}(d_i) \tag{13}$$

resulting in:

$$T_{i-1}^{i} = \begin{bmatrix} cos\theta_i & -sin\theta_i cos\alpha_i & sin\theta_i sin\alpha_i & a_i cos\theta_i \\ sin\theta_i & cos\theta_i cos\alpha_i & -cos\theta_i sin\alpha_i & a_i sin\theta_i \\ 0 & sin\alpha_i & cos\theta_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{14}$$

2.3. Advantages, Disadvantages, and Shortcomings of Using the DH Solutions

The ambiguities involved in the application of the DH solution and the advantages of the application of the new approach have been discussed in previous papers [11–13] in which the methodology for developing simpler geometric models was also presented in order to verify the compatibility of DH together with model validation for several robot models with the help of CAD (CATIA) and specific offline programming and simulation software applications (ABB Robot Studio). For example, the DH convention may lead to the realization of different geometric models for the same robot that are still equivalent to a certain extent, but that do not exactly correspond to the real geometric parameters of the robot. One of the most visible aspects is presented in Figure 4 (Table 2), dimensions in mm, and also in Figures 5 and 6 (Tables 3 and 4), where, for the same type of robot (model IRB 140), different parametrizations are observed in different papers [14–16].

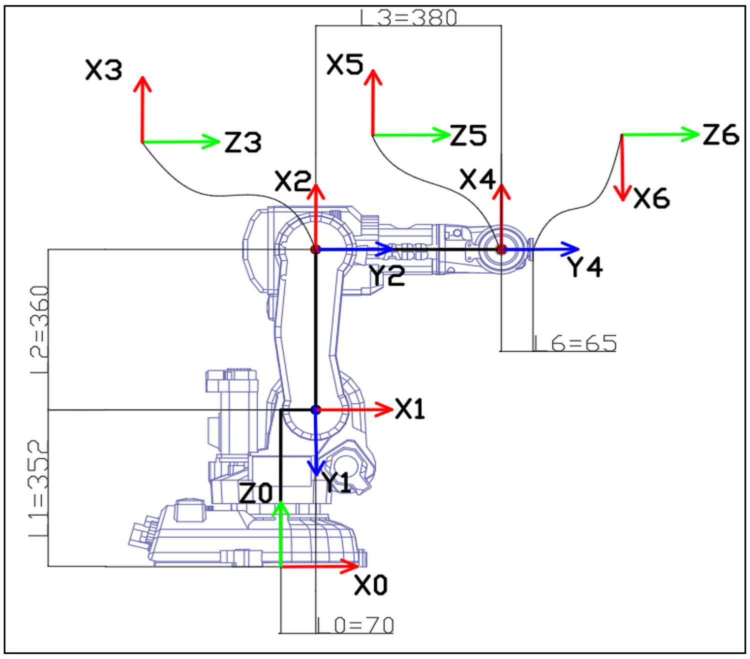


Figure 4. Different structural diagram parametrization of the same robot—example 1 [14].

Table 2. Corresponding DH parameters for Figure 4 [14].

Joint No.	α_i	a_i	d_i	θ_i/q_i
1	$-\pi/2$	70	352	q_1
2	$0°$	360	0	$q_2 - \pi/2$
3	$-\pi/2$	0	0	q_3
4	$\pi/2$	0	380	q_4
5	$-\pi/2$	0	0	q_5
6	$0°$	0	65	$q_6 + \pi/2$

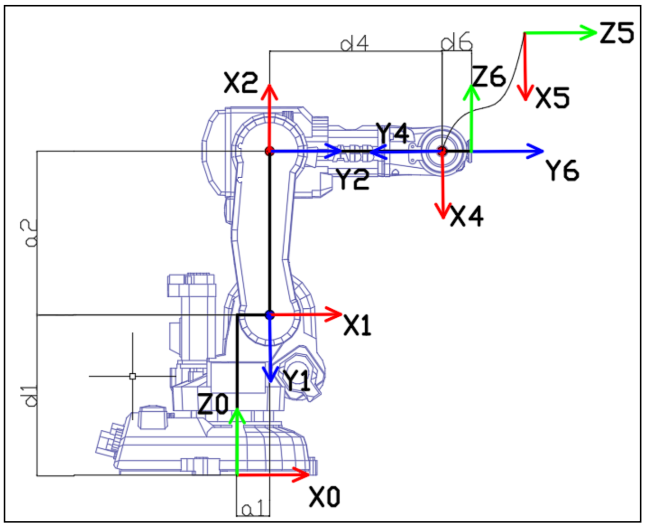


Figure 5. Different structural diagram parametrization of the same robot—example 2 [15].

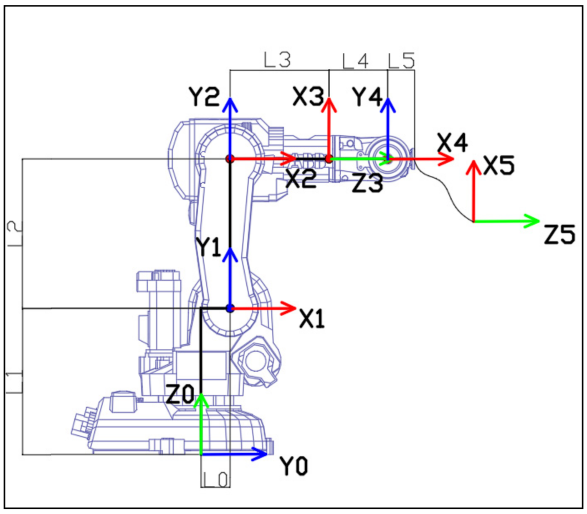


Figure 6. Different structural diagram parametrization of the same robot—example 3 [16].

Table 3. Corresponding DH parameters for Figure 5 [15].

Joint No.	α_i	a_i	d_i	θ_i/q_i
1	$-90°$	a_1	d_1	$0°$
2	$0°$	a_2	0	$-90°$
3	$90°$	0	0	$180°$
4	$-90°$	0	d4	$0°$
5	$90°$	0	0	$0°$
6	$90°$	0	d6	$-90°$

Table 4. Corresponding DH parameters for Figure 6 [16].

Joint No.	α_i	a_i	d_i	θ_i/q_i
1	$\pi/2$	L_0	L_1	$\theta_1 + \pi/2$
2	$0°$	L_2	0	$\theta_2 + \pi/2$
3	$\pi/2$	0	0	θ_3
4	$-\pi/2$	0	L_{3+4}	θ_4
5	$\pi/2$	0	0	θ_5
6	$0°$	0	L_5	θ_6

As can be observed, in practice, when it comes to the geometric and kinematic modeling of robots, simplifications of the theoretical model (of the calculation scheme) are very often used. These simplifications can be successfully represented as theoretical models only if the robot is removed from the context of its real daily life operation cycle. Some of the simplifications that are applied in geometric modeling are creating the structural diagrams as if all the robot's joints are coplanar in a vertical plane, the dimensions of the joints are ignored, the robot's elements are considered to be rigid, and in the particular case of robots with closed kinematic chain architecture, passive joints and mechanical synchronization elements are omitted from the geometric model and are usually treated as mechanisms with only four degrees of freedom. Most of them (such as robots used in palletizing applications) have five degrees of freedom although they actually have only four numerically controlled (motorized) axes. These simplifications lead to aspects that, from a strictly geometric and kinematic point of view (only theoretically), have no influence. On the other hand, if it is desired that the geometric should include additional error parameters and for certain error compensation (such as the errors due to the thermal influence on the robot), then a simplified model no longer corresponds to reality and is no longer sufficient. As a result of the research carried out in [17], it was shown that adding error parameters to the kinematic model leads to a mathematical model that fits the robot better than the nominal kinematic model. In the research carried out on structural and functional optimization in order to increase the performance of an industrial robot (IR), two cases were considered for the development of the analytical calculation model:

- model of the robot unaffected by errors but including real constructive and functional parameters, in addition to the conventional DH parameters;
- model of the robot affected by errors, which includes the modeling of the constructive and functional parameters, the DH parameters, and some error categories that are considered while evaluationg the volumetric error.

Such an extended model was also studied in [18], where, based on DH modeling, a 29-parameter model was used to calibrate a robot (ABB IRB140) to improve the positioning accuracy by approximately three times. In this context, the purpose of this paper is to present the new modeling approach through which a forward geometric model must be developed following only two simple rules and eliminating most of the ambiguities related to the use of DH. This method may be applied to create forward geometric models for any robot type with serial architecture. Moreover, it is a model whose parameterization facilitates the extension of the direct geometric model (by simply adding some translation or rotation parameters to those of the nominal model) in order to include the entire set of constructive and functional parameters of the real robot and also include parameters used later to compensate for certain errors (such as thermal errors). For example, such a model, which is not complete but takes into account as many real constructive parameters as possible (the real position of the active joints including the real offsets, the position of the passive joints, and the real dimensions of the structural elements of the robot), for an articulated arm-type robot with a closed kinematic chain was presented in [19], with the scheme according to Figure 7.

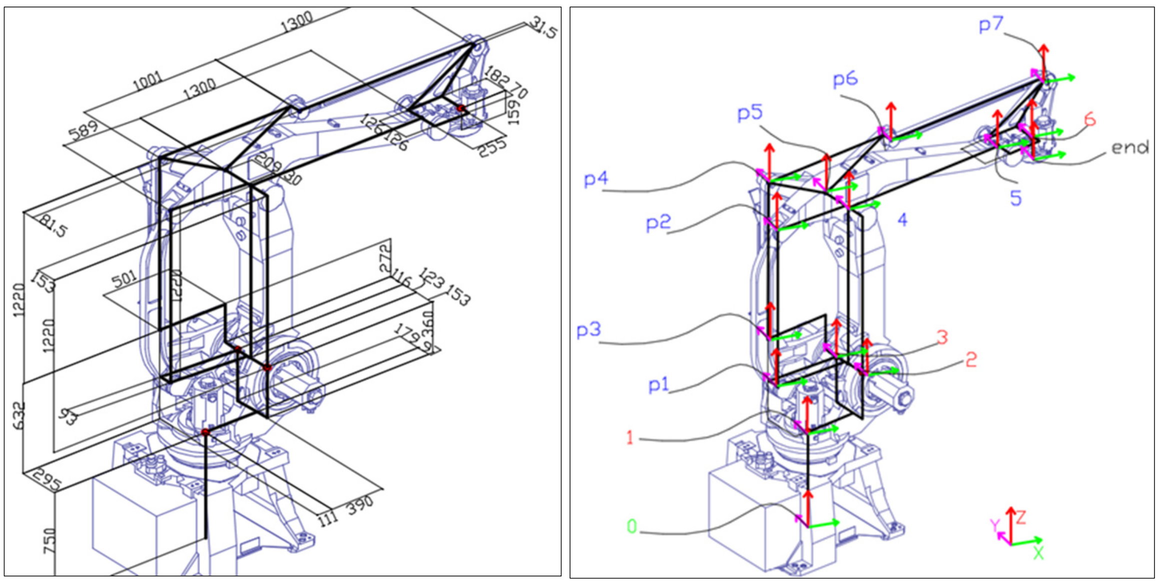


Figure 7. Positioning of axis systems with the same orientation and structural diagram for a closed kinematic chain robot arm (usually, only four joints are used in DH modeling) [19].

To obtain the geometric model, the only two rules to be followed are those presented in papers [11,12]:

- first, apply the translation matrices from the "i" coordinate system to the "i + 1" system without changing the orientation of the initial coordinate system;
- the rotation matrix is applied for the "i + 1" system, which will rotate around one of its own axes (this depends on the rotation axis of that joint).

The use of this method is more efficient to create a forward extended geometric model because, compared to DH, it leads to a more intuitive structural diagram by maintaining the same orientation of the coordinate systems attached to the joints on the entire structure of the robot and a simpler form of the table of parameters (one rotation parameter and three translation parameters for each joint). Next, if deformations of the robot or positioning errors are measured, these can be quantified as displacements along and around the X, Y, and Z axes. Thus, maintaining the same orientation of the axis systems, parameters for measured errors can be easily integrated into the forward model. Measuring instruments include laser interferometers, which are able to perform measurements referencing a global coordinate system. The next section depicts how error compensation parameters could be added to the geometric model.

3. Forward Geometric Model Including Supplementary Parameters

In recent years, the use of industrial robots has increased dramatically in various applications such as assembly, machining, and laser cutting. New trends in advanced applications of industrial robots have been reviewed by Andrius D. et al. in [20]. However, these applications require high precision, and any positioning deviation of the robot can lead to defects in the manipulated parts or the processes into which the robot is integrated. Therefore, it is crucial to improve the positioning accuracy of industrial robots. One common method is to calibrate robots. Despite the benefits of industrial robot calibration, there are also some limitations and downsides to consider [21]. One major limitation is that calibration is a time-consuming and complex process that requires specialized knowledge and equipment (and is usually very expensive too). Additionally, the calibrated parameters may change over time due to wear and tear, which means that periodic recalibration may be necessary. Finally, calibration only addresses geometric errors and does not account for other sources of error, such as thermally induced errors. The solution for this problem is a compensation method, implying a new modeling approach. The purpose of this section is to present this approach, which involves developing a forward geometric model following two simple rules that can be applied to any robot type with serial architecture. This new model allows for easy extension to include all real constructive and functional parameters of a robot, including parameters used to compensate for errors such as thermal errors. The next section will describe how error compensation parameters can be added to the geometric model. However, first, the reason why such a model is necessary must be understood, and for this, the following case study is exemplified.

This case study is a common example in which the position of several successive joints from a simple robot structure must be geometrically described. Figure 8a,b describes the application of transformation matrices from an axis system attached to a base to two successive joints.

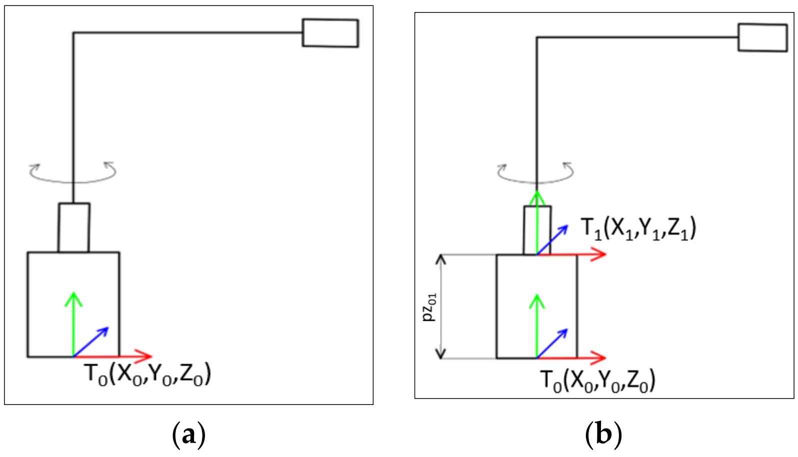


Figure 8. (**a**) Global reference frame (attached to base). (**b**) First joint and parameter for translation along Z axis.

The matrix associated with the coordinate system attached to the base is T_0:

$$T_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{15}$$

The transformation from the coordinate system attached to the base to the coordinate system of the first rotational joint around the Z axis is T_{01}, which is equal to T_0 × translation along Z axis × rotation around Z axis.

$$T_{01} = T_0 \times \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & pz\,01 \\ 0 & 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} cos\theta & -sin\theta & 0 & 0 \\ sin\theta & cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} cos\theta & -sin\theta & 0 & 0 \\ sin\theta & cos\theta & 0 & 0 \\ 0 & 0 & 1 & pz\,01 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{16}$$

The transformation from the first joint to the second joint (with rotation around X axis) is presented in Figure 9.

The associated matrix T_{12} is now equal to T_1 × translation along X and Z× rotation around X.

$$T_{12} = T_{01} \times \begin{pmatrix} 1 & 0 & 0 & px_{12} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & pz_{12} \\ 0 & 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & cos\theta & -sin\theta & 0 \\ 0 & sin\theta & cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & px_{12} \\ 0 & cos\theta & -sin\theta & 0 \\ 0 & sin\theta & cos\theta & pz_{12} \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{17}$$

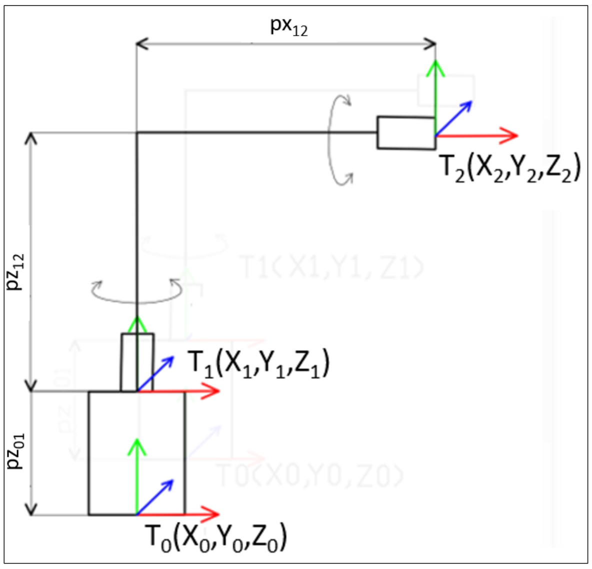


Figure 9. Transformation from first joint to second joint.

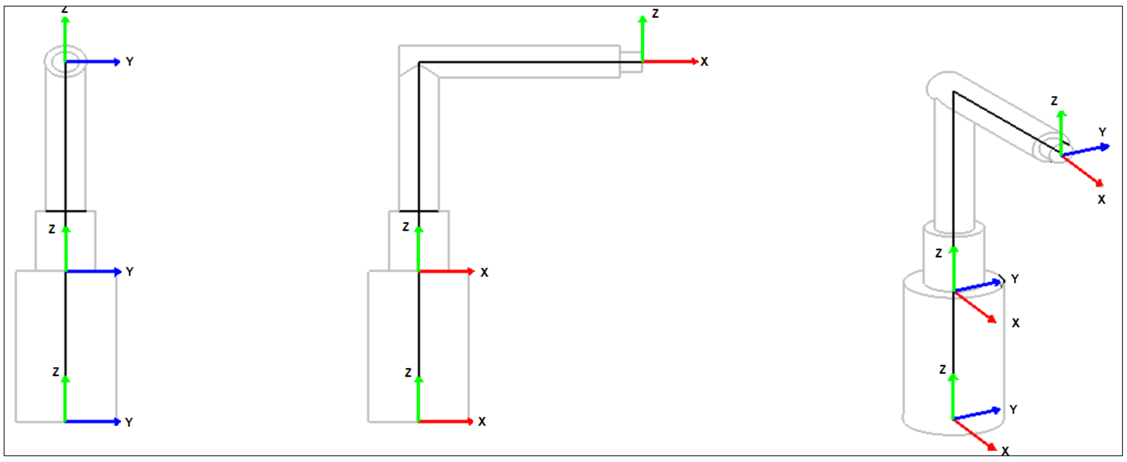
In this case, the position and orientation of the last axis system (characteristic point of the robot at the end flange) is calculated by successively multiplying the transformation matrices: $T_{end} = T_{01} \times T_{12} \times \ldots \times T_{n-1n}$. Thus, Table 5 contains the parameters.

Table 5. Parameters for each joint and rotation angle (without error parameters).

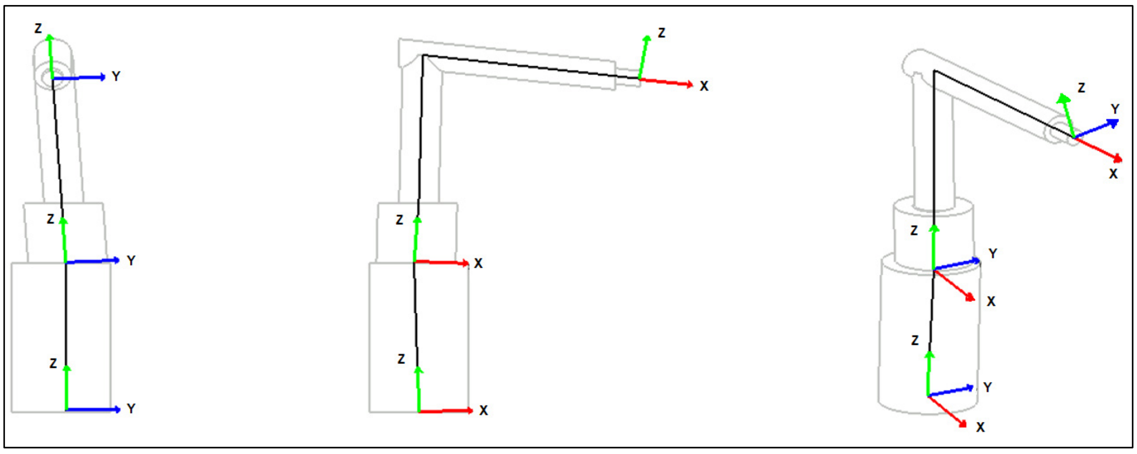
Joint No. (Axis No.)	Rotation Angle	Translations		
		Along X	Along Y	Along Z
0 (base)	θ_0	0	0	0
1	θ_1	0	0	pz_{01}
2	θ_2	px_{12}	0	pz_{12}

Generally, for a system with a higher number of joints, a row corresponding to each new joint has to be added to this table. The procedure is simple to understand and follow, and the resulting table with parameters is clear and easy to fill out. The development of an extended geometric model that is useful for compensating certain errors, the consistency of the models obtained, and the clear form of the table with parameters facilitate the possibility of quickly incorporating external factors/parameters in order to apply the corrections. For example, after completing a study to determine the elastic behavior of the structural elements and the identification of the quantitative yield values, the values could be easily integrated into the geometric model by adding additional terms in the table such as ΔX, ΔY, ΔZ, and $\Delta\theta 1$, $\Delta\theta 2$, etc. The values of these terms are influenced by several factors such as the loading of the robot, accelerations, inertial loads, etc. In the case of thermal displacements, these terms must be correlated with the effective temperature of the robot (considering the values of the thermal expansions and torsions of the elements being proportional and directly influenced by the temperature) or the operating time if the heating curves and thermal stabilization periods are known for a specific robot and working conditions. If necessary, it is possible for a more complex model to be developed in which multiple error factors (from different sources) can be considered simultaneously. The key point is that after the development of the geometric model, the identification of some error parameters and their quantification according to the sources that generate them must be carried out. This can be achieved in multiple ways—from direct measurements and the identification of some functions and corresponding mathematical relations to numerical methods and finite element analysis.

Taking all these into account, it can be considered that the previous theoretical model is deformable as the real robot is during operation, due to the heating of its components, joints, motors, fluids, electrical circuits, etc. The representation of the structural kinematic scheme in a single plane is no longer sufficient. In Figure 10a,b, such a model is presented in a simplified manner (initial, at start-up) and deformed (later, after suffering some deformations due to heating during operation).



(**a**)



(**b**)

Figure 10. (**a**) Initial structural diagram (cold start). (**b**) Structural diagram after deformation.

Even the real deformations are not of the same order of magnitude as the required positioning accuracy; due to the serial structure of the robot, even small elongations or torsions of the elements will cause significant deviations of the characteristic point because errors chain and accumulate. For this reason, for each of the changes in the position or orientation of the structural elements, the robot operator must include additional parameters in the geometric model. Considering that the deformations can twist a structure in 3D, each joint position local coordinate system must be characterized by three additional matrices of shifting the orientation that may appear (additional rotation around the X, Y, and Z axes) but also by three additional matrices of deviation from the initial position (three additional translations along the X, Y, and Z axes). Table 6 presents the parameters encompassing these additional factors.

Table 6. Parameters of angular deviations and translations.

Joint No. (Axis No.)	Angular Deviations	Rotation Angle	Translations		
			Along X	Along Y	Along Z
0 (base)	$D\theta_{0x}$, $D\theta_{0y}$, $D\theta_{0z}$	θ_0	$0 + D\,X_0$	$0 + D\,Y_0$	$0 + D\,Z_0$
1	$D\theta_{1x}$, $D\theta_{1y}$, $D\theta_{1z}$	θ_1	$0 + D\,X_1$	$0 + D\,Y_1$	$pz01 + D\,Z_1$
2	$D\theta_{2x}$, $D\theta_{2y}$, $D\theta_{2z}$	θ_2	$p \times 12 + D\,X_2$	$0 + D\,Y_2$	$pz12 + D\,Z_2$

Returning to the matrix multiplication rule, and each time before moving to the next joint, the robot operator must first update the current position to take into account the error caused by the deformation. This assumes that first, the matrix describing the initial position must be multiplied in a row by the three translation matrices that represent the displacement of the joint from the initial position, followed by multiplication by the three additional rotation matrices that represent the angular deviation. The obtained result has to be multiplied by the translation matrix (which represents the length of the segment that is connected to the next joint) in the direction of the axis corrected in the previous step. In a mathematical form, this is described as follows:

(1) Correction of linear displacements (lc = linear correction)

$$T_{01c} = T_0 \times \begin{pmatrix} 1 & 0 & 0 & \Delta X_0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & \Delta Y_0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta Z_0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & \Delta X_0 \\ 0 & 1 & 0 & \Delta Y_0 \\ 0 & 0 & 1 & \Delta Z_0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{18}$$

(2) Correction of angular deviations (ac = angular correction)

$$T_{0\,\text{ac}} = T_{01c} \times \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & cos\Delta\theta 0x & -sin\Delta\theta 0x & 0 \\ 0 & sin\Delta\theta 0x & cos\Delta\theta 0x & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} cos\Delta\theta 0y & 0 & sin\Delta\theta 0y & 0 \\ 0 & 1 & 0 & 0 \\ -sin\Delta\theta 0y & 0 & cos\Delta\theta 0y & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} cos\Delta\theta 0z & -sin\Delta\theta 0z & 0 & 0 \\ sin\Delta\theta 0z & cos\Delta\theta 0z & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{19}$$

(3) Transformation of the segment length from the base to first joint (M = modified, with all corrections included)

$$T_{01M} = T_{0\,\text{ac}} \times \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & pz\,01 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{20}$$

To go from the first joint to the second joint, the previous algorithm must be repeated, first applying the linear corrections for the location of joint 1, then applying the angular corrections and multiplying in the last phase with the transformation matrices related to the lengths of the segments connecting joint 1 to joint 2. The previous example represents a model with bilateral symmetry, that is, in which the structural scheme is located right in the vertical plane of symmetry. In reality, even robots that look symmetrical from the outside have inside components that are different on one side compared to the other, and this different disposition of the components influences the way in which heat is transmitted and in which the structure is distorted. All the geometrical parameters that describe a robot model are three linear dimensions that describe the position of a joint compared to the previous one and the angle at which the respective joint is rotated. If we consider that the structure is also deformable, then three additional parameters are added that represent the additional linear displacements caused by the deformations, along with three more angular parameters that represent the orientation deviations (not only around the joint axis but in relation to all axes because the distortions could occur in 3D). Therefore, knowing all the parameters, a general table of parameters can be created that is suitable for calculating the transformations from one joint to another for any type of robot with serial architecture. The universal form of the parameters table is presented in Table 7.

Table 7. Universal form of parameters table.

Joint No. (Axis No.)	Angular Deviations	Rotation Angle	Translations		
			Along X	**Along Y**	**Along Z**
0 (base)	$D\theta_{0x}$, $D\theta_{0y}$, $D\theta_{0z}$	θ_0	px_{00} + D X_0	py_{00} + D Y_0	py_{00} + D Z_0
1	$D\theta_{1x}$, $D\theta_{1y}$, $D\theta_{1z}$	θ_1	px_{01} + D X_1	py_{01} + D Y_1	pz_{01} + D Z_1
2	$D\theta_{2x}$, $D\theta_{2y}$, $D\theta_{2z}$	θ_2	px_{12} + D X_2	py_{12} + D Y_2	pz_{12} + D Z_2
...	...	...	...	...	...
6	$D\theta_{6x}$, $D\theta_{6y}$, $D\theta_{6z}$	θ_6	px_{56} + D X_6	py_{56} + D Y_6	pz_{56} + D Z_6

Now, applying the same rules as those presented previously and using Formulas (18) and (19), we obtain a universal form of the correction matrix. To simplify the matrix writing, we will first note the following terms:

$$A = \cos D\theta_{xi};\ B = \sin D\theta_{xi};\ C = \cos D\theta_{yi};\ D = \sin D\theta_{yi};\ E = \cos D\theta_{zi};\ F = \sin D\theta_{zi}$$

The correction matrix (T_C) becomes:

$$\begin{pmatrix} EC & -CF & D & -CF\text{py} + EC\text{px} + D\text{pz} + \Delta X \\ AF + EBD & EA - BDF & -BC & \text{px}(AF + EBD) + \text{py}(EA - BDF) - BC\text{pz} + \Delta Y \\ BF - EAD & ADF + EB & AC & \text{px}(BF - EAD) + \text{py}(ADF + EB) + AC\text{pz} + \Delta Z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{21}$$

The process of passing from one joint to another to obtain the final position of the deformed robot now becomes:

$$T_{end} = (T_{C0} \times R_0) \times (T_{C1} \times R_1) \times \ldots (T_{Cn} \times R_n) \tag{22}$$

where we recall that R represents the rotation matrices around the axis of the respective joint (Formulas (4)–(6) presented in Section 2.1). Considering the additional parameters and writing the mathematical model in this way is more difficult than creating a classic DH model, but this must be carried out only once, at the implementation into the software algorithm. These additional steps seem difficult to achieve, but in the actual applications in which industrial robots are increasingly involved (such as machining, welding, or precision assembly) reducing errors as much as possible is extremely important. By comparison, the transformation matrices within a geometric conventional DH model contain four parameters for each joint, three of which are constant and one variable (only the angle of the joint is variable). The general model proposed in this paper involves 10 parameters, of which only 3 are constant and 7 are variable. A comparison of computing efficiency between the classical DH model and the proposed model has not been performed for two reasons: the software limitation of a robot controller does not allow the implementation of such an extended model, and different machines will provide different and irrelevant results in computing time caused by hardware capabilities. Even in CoppeliaSIM, there is no classic DH model implemented. Instead, each joint of serial structures is, by default, characterized by seven parameters. Six of these parameters are constants (joint position and orientation) and one is variable (joint rotation angle). A comparison was made regarding the simulation time using the default model from CoppeliaSIM and the model in which the deformations are applied continuously. In both cases, a robot operation program of 200 min was simulated, in which the robot moved continuously, reaching a total of 10,050 target points. The complete program ended after 3 min and 40 s for the unreformable robot case and only 2 s more in the case of the deformable robot whose deformation variables were constantly updated. At this point, it is desired that the software compensation solution be an offline one. From this point of view, the time of simulation and obtaining the results is not relevant. Moreover, in this phase, the initial program of the robot must be manually updated with the new angles on the corrected joints. Aspects to be improved will be discussed in more detail in the conclusions section. To emphasize the importance and usefulness of the new modified forward model, the exemplification of the use of such an extended model is presented in the next section.

4. Case Study

The elements presented in the previous sections, related to the development of a directly extended geometric model for robots with serial architecture, are part of a series of research/papers aiming to obtain a thermal error compensation solution for an industrial robot that suffers deformations during operation due to its heating. This extended model fits the following procedure:

1. experimental evaluation of the thermal behavior of a robot during operation;
2. experimental determination of induced thermal deformations and positioning errors;
3. creating an extended geometric model that includes the necessary parameters that also characterize the previously measured errors;
4. implementation of the geometric defect model from point 3 in a software algorithm for inverse kinematics computation based on this deformed geometric model;
5. using the results calculated through inverse kinematics based on the deformed model to apply angle corrections in the robot's program so that, even when deformed, it will reach the programmed points with the highest possible accuracy.

The usefulness is proven by the connection with the previous research for a real case of an industrial robot of the articulated arm type, and the results must be presented as a whole methodology, which is reached by applying the entire procedure with the previously listed steps. Through thermal recording techniques using an infrared camera and monitoring using sensors located on the robot's structure, the thermal behavior of an industrial robot model IRB 140 (in a certain cycle and working environment) was determined in [22]. Based on these results, in [23], a corroboration between the results of a measurement procedure using a laser tracker and finite element analysis (FEA) was carried out in order to determine and locate the deformations and displacements of the robot elements on the whole structure. At this point, only static deformations were considered, but the model may be extended by determining the deformations using a transient FEA analysis [24]. Next, a forward geometric model (as presented in this paper) could be constructed, including parameters for identified errors. How these errors behave is strongly related to the warm-up of the robot, so in [25], a warm-up prediction analysis was conducted for the robot (using regression analysis). The measurements from [22] showed that the robot heats up continuously for a period of approximately 180 min until it reaches thermal stabilization. During the approximately 3 h of work, the heating curves of the segments are not exactly linear but follow a certain curve (not very steep). In [26], the coefficients of the heating curves were determined through regression analysis, and the most accurate results regarding the anticipation of heating curves were obtained through cubic regression. However, the linear regression was not very biased, the correlation coefficient being a high one (0.87). For this reason, in this paper, it was considered that the deformations occur linearly and proportionally to the working time. Therefore, the maximum deformations identified on each segment of the robot were applied increasing linearly in each minute of the work cycle. The final step of implementing the forward geometric model in a software algorithm for the computation of the inverse kinematics (based on the deformed model) in order to obtain corrected joint angle values to compensate for the thermal errors was presented in [27]. The structural diagram used for constructing the forward geometric model already included real constructive parameters (Figure 11 [19]).

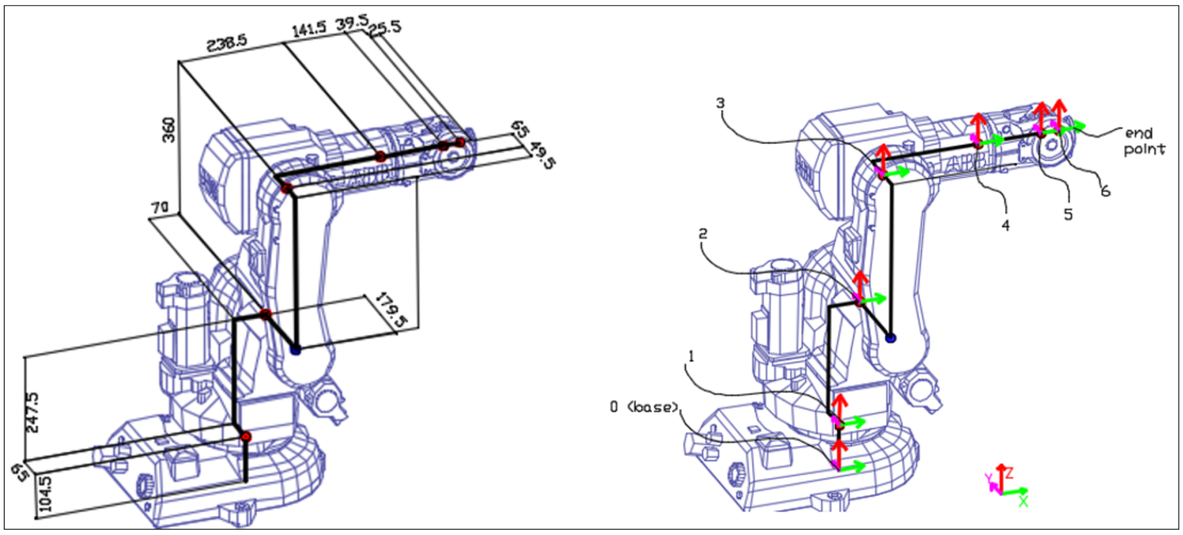


Figure 11. Structural diagram used for construction of the robot's geometric model [19].

Based on constructive parameters and error parameters experimentally determined, the following tables with parameters were conceived.

Based on the structural diagram and values from the tables of parameters (Tables 8 and 9), transformation matrices were multiplied according to methodology presented in Section 2. Then, the virtual model (corresponding to the deformed robot) was constructed in CoppeliaSIM (computer-aided design and programming software application). CoppeliaSIM is a hybrid application with CAD capabilities and scripting programming in different languages. A deformed robot virtual model could also be constructed in any other CAD application (such as CATIA) and then controlled via API programming specific to the CAD application, as in [28]. By changing the position and orientation of the robot joints on the virtual model, the entire structure is distorted, as seen in Figure 12.

Table 8. Angular parameters and angular deviation parameters [19].

Joint No. (Axis No.)	Angular Deviations [°]			Rotation Angle
	$D\theta_x$	$D\theta_y$	$D\theta_z$	
0 (base)	0	0	0	θ_0
1	-8.44×10^{-5}	-0.011	-0.0137	θ_1
2	0.0108	0	0	θ_2
3	6.665×10^{-6}	0.0081	0.0098	θ_3
4	0	0	0	θ_4
5	0	0	0	θ_5
6	0	0	0	θ_6

Table 9. Translation parameters [19].

Ref. Axis No.	Angular Deviations [mm]					
	X	DX	Y	DY	Z	DZ
1	0	−0.02006	0	0.000154	104.5	−0.05297
2	70	0	65	0	247.5	0
3	0	0.10086	−123	0.114855	360	−0.00142
4	238.5	−0.00440	58	0.0241	0	−0.0149
5	141.5	0	0	0	0	0
6	39.5	0	0	0	0	0
7	−0.0166	−0.01662	0	0	0	0

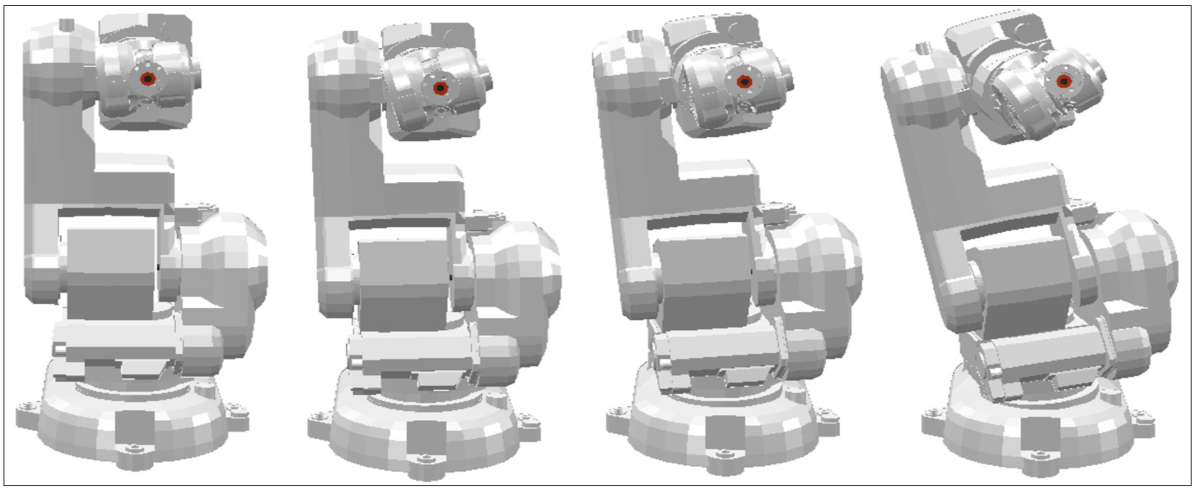


Figure 12. Virtual model of the robot during deformation (deformations are magnified up to 1000 times).

After building the deformed virtual model, the script for calculating the inverse kinematics was launched into execution. The main functions of the script are presented in Appendix A, which includes a link to the complete station (file) that can be downloaded and loaded in CoppeliaSIM. The software algorithm runs according to the diagram presented in Figure 13.

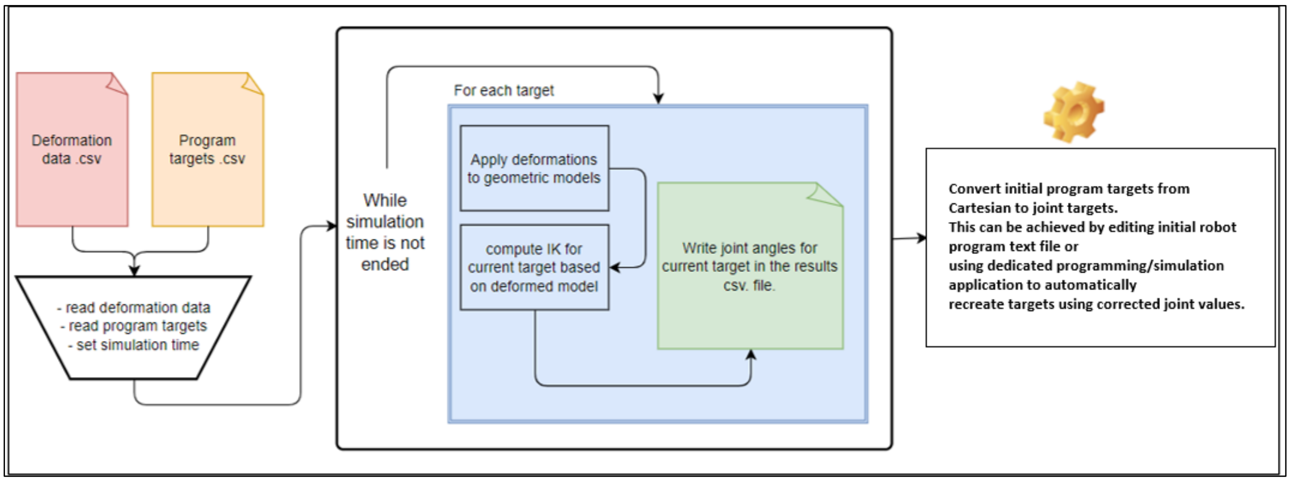

Deformation data .csv
Program targets .csv
- read deformation data
- read program targets
- set simulation time
While simulation time is not ended
For each target
Apply deformations to geometric models
compute IK for current target based on deformed model
Write joint angles for current target in the results csv. file.
Convert initial program targets from Cartesian to joint targets.
This can be achieved by editing initial robot program text file or
using dedicated programming/simulation application to automatically
recreate targets using corrected joint values.

Figure 13. Software algorithm for IK based on extended geometric model of the deformed robot.

The robot's targets must be extracted from a dedicated offline programming and simulation program (in this case, RoboDK) or they can be extracted from a CAD model (sketches, trajectories) according to the methodology presented in [26]. After completing the entire procedure, the main result obtained in the previous tests and published in [25] showed that the deformed model reaches the programmed targets with an improved accuracy ranging from 24% to 93% (depending on the simulation time and general state of the robot). In the case that the robot has reached thermal stabilization, the positioning accuracy is about 0.007 mm (by compensating thermally induced errors with a measured magnitude of 0.097 mm). The maximum angular deviation obtained after compensation was 0.02°. It was observed that the thermal behavior of industrial robots is particular to each individual robot and depends on many factors, such as load, working speed, working time, and conditions. For the robot in the laboratory, the entire thermal procedure based on the directly proposed geometric model led to the mentioned results for a particular case of working conditions. It should be mentioned that the determined displacements and the entire compensation procedure are based on experimental measurements and procedures presented in previous works that were carried out in relatively cold environmental conditions (ambient temperature of 5–7 degrees). Checking the state of the robot, calibrating it, and measuring the positioning accuracy using a laser tracker system was carried out and presented in [19]. The procedure for recording the heating behavior of the robot, using contact sensors placed on the robot as well as recordings with an infrared camera, was carried out in [22]. Based on the recordings made in [22], FEA simulations and analyses were performed to determine the deformations on the entire structure of the robot and are presented in [23]. In the work [24], studies were carried out to predict the temperatures of the robot during a known work cycle. Through linear regression, the coefficients of the approximation functions of the heating curves were determined. The first attempt to apply the deformations on the virtual model of the robot was presented in paper [25]. In this study, the geometric model was altered with very small values because the robot suffered small deformations due to the cold environment and working at a maximum speed of 60% of the robot's working speed and without load (because the reflector necessary for laser tracking was mounted on the robot's flange). Thus, the maximum temperature reached by the robot in certain areas was only 32 degrees Celsius. Moreover, in this study, only 45 positions were analyzed (without changing the configuration of the robot). The 45 target points were the pre-programmed equally spaced positions from the calibration procedure. Although the software algorithm worked, the results provided were not exactly relevant for two reasons:

1. There could be a chance that the few target points on which the test was carried out were particular positions that do not involve problems for determining the position of the robot.
2. The deformations applied were very small and for some elements, even zero. In this case, there is also the possibility of a particular case in which the deformations of some elements cancel others out or that the software algorithm works only for these particular values.

This paper aims to prove the usefulness of the implementation of the extended geometric model in a software compensation solution for thermal errors that has general applicability and does not work only in certain particular cases. For this reason, the following case study was analyzed, in which the robot suffers exaggerated deformations. Each joint of the robot is 1 mm off from its initial position along each axis direction and rotated by 1 degree around each axis from its initial orientation. Moreover, this time, the Cartesian coordinates of the target points were randomly generated. The parameters are presented in Tables 10 and 11.

Table 10. Angular parameters and angular deviation parameters—case study.

Joint No. (Axis No.)	**Angular Deviations [°]**			**Rotation Angle**
	$D\theta_x$	$D\theta_y$	$D\theta_z$	
0 (base)	1	1	1	θ_0
1	1	1	1	θ_1
2	1	1	1	θ_2
3	1	1	1	θ_3
4	1	1	1	θ_4
5	1	1	1	θ_5
6	1	1	1	θ_6

Table 11. Translation parameters—case study.

Ref. Axis No.	Angular Deviations [mm]					
	X	DX	Y	DY	Z	DZ
1	0	1	0	1	104.5	1
2	70	1	65	1	247.5	1
3	0	1	−123	1	360	1
4	238.5	1	58	1	0	1
5	141.5	1	0	1	0	1
6	39.5	1	0	1	0	1
7	−0.0166	1	0	1	0	1

A robot program was created with 50 randomly generated targets placed in the front of the robot. Two working cases were then considered. In the first case, the displacements of the robot joints were statically applied. The script was then executed based on the already deformed virtual model. The programmed targets are presented in Table 12.

Table 12. Programmed targets (static case).

Programmed Coordinates				Programed TCP Orientation			Programmed Coordinates				Programed TCP Orientation		
	X	Y	Z	R_X	R_Y	R_Z		X	Y	Z	R_X	R_Y	R_Z
P1	641	21	473	0	90	0	P26	695	288	364	0	90	0
P2	536	235	281	0	90	0	P27	483	298	550	0	90	0
P3	656	242	373	0	90	0	P28	417	184	294	0	90	0
P4	510	276	526	0	90	0	P29	700	36	537	0	90	0
P5	459	−118	333	0	90	0	P30	452	103	282	0	90	0
P6	610	99	466	0	90	0	P31	580	25	303	0	90	0
P7	465	−55	492	0	90	0	P32	547	−113	431	0	90	0
P8	436	237	298	0	90	0	P33	544	−83	447	0	90	0
P9	445	−117	511	0	90	0	P34	493	187	285	0	90	0
P10	547	29	431	0	90	0	P35	644	53	525	0	90	0
P11	415	171	268	0	90	0	P36	592	−10	393	0	90	0
P12	490	300	336	0	90	0	P37	626	143	336	0	90	0
P13	619	7	521	0	90	0	P38	434	95	515	0	90	0
P14	563	296	267	0	90	0	P39	635	−10	469	0	90	0
P15	571	270	520	0	90	0	P40	547	245	397	0	90	0
P16	414	101	432	0	90	0	P41	514	234	468	0	90	0
P17	431	50	452	0	90	0	P42	622	−144	277	0	90	0
P18	462	−94	399	0	90	0	P43	600	−103	393	0	90	0
P19	692	76	390	0	90	0	P44	462	109	354	0	90	0
P20	536	248	280	0	90	0	P45	439	103	506	0	90	0
P21	619	18	314	0	90	0	P46	414	136	536	0	90	0
P22	527	287	353	0	90	0	P47	464	−149	491	0	90	0
P23	647	222	381	0	90	0	P48	628	276	275	0	90	0
P24	405	244	479	0	90	0	P49	522	−143	294	0	90	0
P25	415	192	396	0	90	0	P50	537	−37	325	0	90	0

All the target points were programmed keeping the same orientation of the TCP to not only easily check if, following the execution of the script from CoppeliaSIM, the Cartesian coordinates are respected but also to check the orientation of the robot flange. After executing the script, solutions were determined for all 50 target points. The minimum and maximum values of the orientation deviation are presented in Table 13. The coordinates that the deformed robot reached are presented in Table 14.

Table 13. Deviation from programmed orientation (static case).

	Orientation Deviation [°]		
	$\Delta\theta_X$	$\Delta\theta_Y$	$\Delta\theta_Z$
MIN	0	0	−0.002
MAX	0	0.02	0.005

Table 14. CoppeliaSIM targets (static case).

	CoppeliaSIM Coordinates				**Programmed Coordinates**		
	X	**Y**	**Z**		**X**	**Y**	**Z**
P1	640.998	20.999	472.999	P26	694.996	287.991	363.999
P2	535.999	235.000	281.000	P27	483.003	297.998	550.004
P3	656.000	241.991	373.001	P28	416.992	184.010	294.000
P4	510.003	275.996	526.004	P29	700.003	35.999	537.002
P5	459.000	−118.003	332.999	P30	451.998	103.000	281.999
P6	610.001	99.002	466.001	P31	579.996	24.994	302.997
P7	464.997	−55.004	491.997	P32	547.001	−113.001	431.001
P8	435.999	237.000	298.000	P33	544.002	−83.001	447.001
P9	445.001	−117.001	511.002	P34	492.999	187.001	285.000
P10	547.005	29.010	431.004	P35	644.002	53.000	525.001
P11	414.999	171.000	268.000	P36	591.998	−9.990	392.999
P12	490.003	299.996	336.001	P37	626.003	143.005	336.002
P13	619.002	6.999	521.001	P38	434.000	95.002	515.001
P14	562.999	296.000	267.000	P39	635.000	−10.002	469.000
P15	571.001	270.001	520.000	P40	546.999	245.001	396.999
P16	413.997	101.006	431.998	P41	514.001	233.996	468.002
P17	430.998	49.996	451.998	P42	622.000	−144.003	276.999
P18	461.994	−93.997	398.995	P43	600.002	−103.008	393.001
P19	692.005	75.999	390.002	P44	462.005	109.008	354.005
P20	536.002	248.010	280.002	P45	439.001	103.001	506.001
P21	618.993	17.993	313.997	P46	414.002	135.998	536.003
P22	527.005	287.004	353.003	P47	464.000	−149.004	491.000
P23	646.997	221.993	380.999	P48	627.999	276.001	275.000
P24	405.003	244.002	479.003	P49	521.992	−143.001	293.996
P25	414.997	192.003	395.998	P50	537.004	−37.001	325.002

The minimum and maximum values of the position deviation are presented in Table 15.

Table 15. Position deviation (static case).

	Position Deviation [mm]		
	ΔX	**ΔY**	**ΔX**
MIN	−0.0053	−0.0099	−0.0047
MAX	0.0084	0.00929	0.005

The joint angles necessary to reach the programmed points calculated with IK in CoppeliaSIM based on the deformed model (with values of the errors shown previously) are presented in Table 16.

Table 16. Robot joint angles for deformed robot (compensated errors).

	Joint Angles Computed for Each Target					
	J1	**J2**	**J3**	**J4**	**J5**	**J6**
P1	2.810995	26.28615	2.378532	−0.15608	−31.019	−6.68297
P2	26.62761	43.21751	14.09162	25.74425	−63.459	−18.008
P3	22.93873	44.08226	−8.68216	29.66404	−43.2045	−29.2832
P4	33.23419	15.92531	9.233602	49.5101	−42.2511	−49.3862
P5	−17.4076	24.26864	36.40971	−23.7015	−63.9363	5.716024
P6	11.09623	24.11561	7.243542	14.21492	−35.2344	−18.658
P7	−7.59358	0.09331	32.73793	−17.8784	−35.6293	7.620256
P8	32.53324	34.05742	30.30792	30.47852	−71.1767	−16.7415
P9	−17.0201	−3.04336	32.26777	−35.4684	−35.0908	22.97249
P10	3.81082	18.29642	22.26341	0.828017	−42.9847	−7.14365
P11	25.54021	36.07047	38.04107	22.07303	−78.3546	−9.73248
P12	35.63698	35.49974	16.15504	36.95549	−61.9528	−26.3381
P13	1.610659	19.50852	4.022058	−2.31556	−25.8131	−5.04101
P14	30.95777	49.89616	2.903374	31.68392	−60.8979	−22.8277
P15	29.44616	23.32127	−0.1367	47.25214	−38.1531	−48.4764
P16	16.39806	2.722649	42.67031	17.36198	−50.1682	−18.3206
P17	8.01806	0.308295	40.74881	7.228396	−44.0928	−12.18
P18	−13.719	12.96507	36.47092	−21.7382	−52.6882	7.528158
P19	7.457114	41.02561	−7.01741	7.177614	−36.8931	−12.1077
P20	27.90518	43.90233	12.88649	27.24821	−63.3535	−18.9197
P21	1.908118	40.54725	9.667821	−2.20752	−52.4979	−4.18649
P22	32.36995	35.64228	11.64176	35.08456	−57.1776	−27.7245
P23	21.54627	41.26615	−5.08687	27.38036	−43.2764	−27.3789
P24	36.74781	5.304382	30.8585	45.23446	−51.8357	−40.4683
P25	29.09069	13.72432	37.74581	30.13305	−59.585	−23.4212

Due to the large size of the results table, the calculated angle values are presented only for the first 25 target points. The complete results files are available for download from the GitHub repository, whose link is written at the end of Appendix A.

In the second case, the deformations were constantly applied during a 200 min operating cycle. The robot program was executed in a loop for this whole time. The forward geometric model was updated minute by minute, and at the same interval, the IK was recalculated for each point that had to be reached. In this interval, the robot traveled a total number of 10,050 target points. For the same reason of a very large table of results, in Tables 17 and 18, only the maximum and minimum position and angular deviations recorded are presented.

Table 17. Position deviation (transient case).

	Position Deviation [mm]		
	ΔX	**ΔY**	**ΔX**
MIN	−0.199	−0.198	−0.199
MAX	0.1	0.1	0.1

Table 18. Deviation from programmed orientation (transient case).

	Orientation Deviation [°]		
	$\Delta\theta_X$	**$\Delta\theta_Y$**	**$\Delta\theta_Z$**
MIN	0	0	−0.007
MAX	0	0.1	0.004

Results Analysis and Synthesis

In the laboratory, measurements were performed simultaneously for recording the position of the robot using a laser tracking system and recording the thermal evolution during work using an infrared camera, as presented in Figure 14.

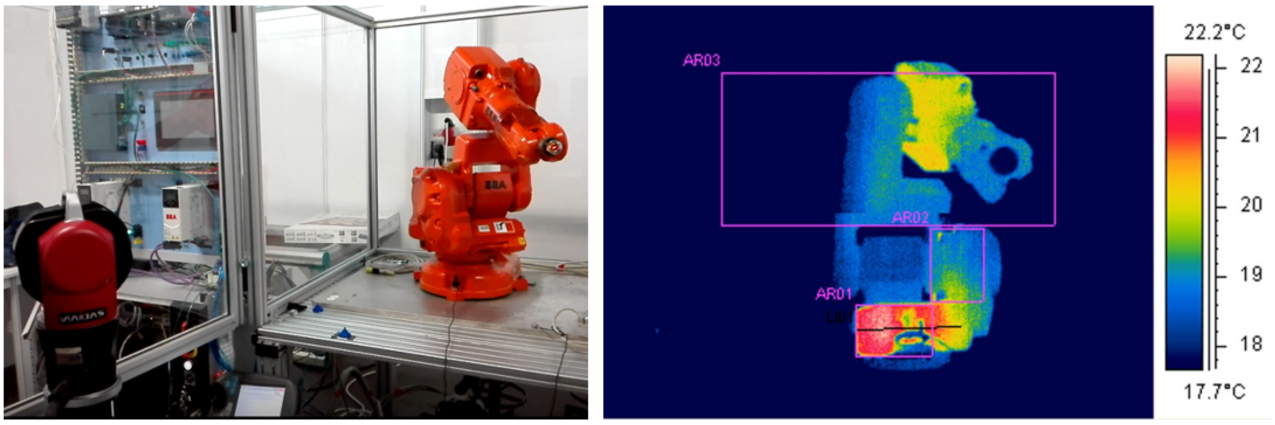


Figure 14. The robot in the laboratory during measurements.

Due to the thermal deformations, for the IRB 140 industrial robot in the laboratory, the experimental measurements showed a deviation of the characteristic point of approximately 0.097 µm. By using the deformed virtual model, based on the extended geometric model with included error parameters and applying the entire error compensation procedure, the precision of reaching the pre-programmed points of the deformed robot (thus reaching thermal stabilization) was 0.07 mm. For the situation in which the robot was still in the warm-up status, the application of the presented compensation methods showed a possibility of improving the positioning precision by between 0.07 mm and 0.073 mm. There are still many aspects to be studied regarding the improvement of the accuracy and during the thermal transition period. At this stage, the Improvement of the positioning accuracy by up to 93% proves that the extended geometric model, with a structural diagram based on the real constructive elements of the robot and including error parameters, is important and its use is mandatory in software procedures for thermal error compensation. Moreover, in the case study where exaggerated displacements of 1 mm in all directions for each joint of the robot and 1 degree of angular deviation, also around each axis, were considered, the ability of the procedure to provide compensated angles was shown. The robot structure being deformed to such a magnitude led to severe deviations of the characteristic point compared to the theoretical position. In the image, the deformations are visible with the naked eye even if they have not been magnified. Deviations of the characteristic point of even 20 mm are observed. From the results presented in Section 4, for this excessively deformed structure, it was possible to determine the angles of the joints (compensating for thermal errors) so that the accuracy of reaching the programmed targets is about 0.01 mm and the maximum angular deviation is 0.02 degrees. For the transient case, in which the deformations were applied progressively throughout the 200 min of robot operation, the maximum accuracy was approximately 0.2 mm, and the maximum angular deviation was 0.01 degrees, which represents an improvement with an order of magnitude two times higher than the deformations (Figure 15).

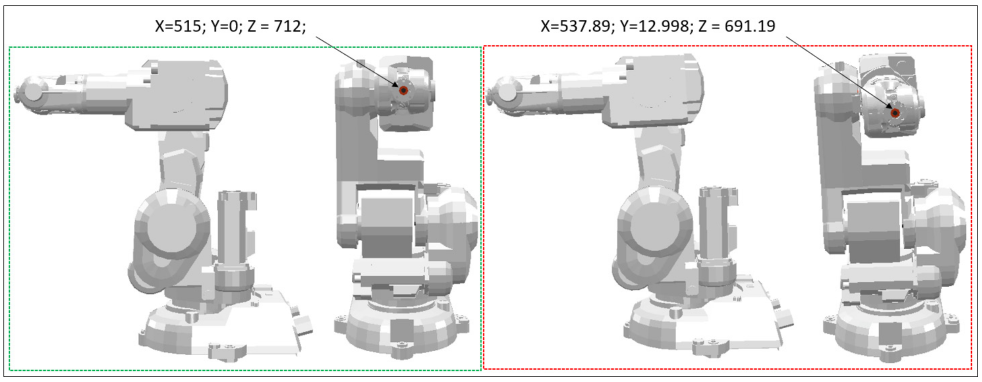


Figure 15. Ideal robot structure and home position coordinates and deformed robot structure and home position coordinates.

5. Conclusions

In this paper, it was proved that in the context of the integration of industrial robots in applications that require high precision, simple modeling (applying theoretical assumptions such as the non-deformability of structural elements or not respecting the real configuration of the robot) is no longer sufficient. The conception of the forward geometric model (which actually represents the basis from which the control of industrial robots starts) must respect as much as possible the structure of the robot and if it is imposed to improve their precision, errors that may occur (such as errors due to thermal deformations) must be found, analyzed, interpreted, and corrected. The use of an extended geometric model in which the same orientation of the axis systems is maintained facilitates the construction of the parameter tables and simplifies the integration of additional correction factors. These should be considered in the multiplication of the matrices that mathematically describe the structure of the robot. The use of this extended geometric model is only a necessary step in a suite of procedures starting from experimental measurements up to the development of a software algorithm for error compensation. Even if the introduction of additional parameters in the calculation model means additional mathematical operations, considering modern computational capabilities, with a relatively small additional effort, the improvements regarding the positioning accuracy of the robot (by compensating for thermally induced errors) are very important. The entire thermal error compensation methodology can be used, but care must be taken to ensure that all experimental determinations and geometric modeling are carried out individually for each type of robot and for known and stable working conditions. Because the proposed compensation method is an off-line one, it is assumed that the working environment, the robot program, and the thermal behavior are known and well determined previously and that they do not change. Considering this aspect, it is desired that the methodology to be improved in the future. Among the aspects that will be researched and further developed are:

(a) development of an online software link between CoppeliaSIM and the robot controller. The possibility of online transmission of the corrected target points or the possibility of direct control of the robot from CoppeliaSIM would remove the manual post-processing part of the files containing the robot's work programs.
(b) the integration of a hardware–software technical solution for measuring the temperature of the robot structure in real time and automatically adjusting the parameters related to the deformations of the structural elements.
(c) carrying out analysis with finite elements and experiments to determine the thermal behavior of the robot and the deformation of its structure in various environmental conditions, from very low temperatures to very high temperatures.
(d) adjusting the parameters of the calculation functions for inverse kinematics to obtain faster and more accurate solutions.

(e) identifying a solution for the problem of decreasing the degree of precision in providing results for the transient case.
(f) the implementation of a software solution for processing and compensation and for programs that involve tracking curves.

Author Contributions: Writing—review & editing, C.C., M.I., I.G.G. and C.P. All authors have read and agreed to the published version of the manuscript.

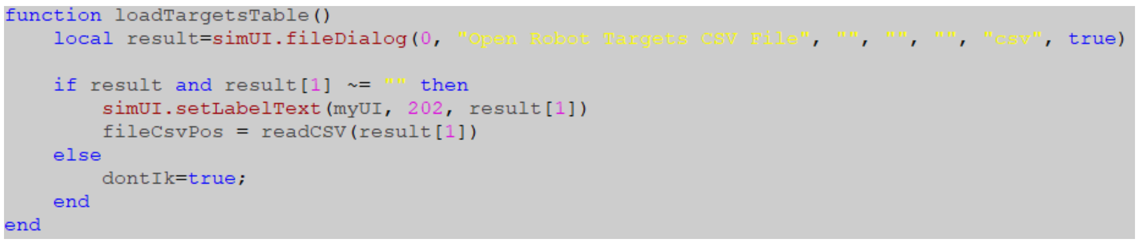
Funding: These studies were carried out as a research activity within the postdoctoral research project of the first author. This work was supported by the grant POCU/993/6/13/153178, “Performanță în cercetare” (“Research performance”) co-financed by the European Social Fund within the Sectorial Operational Program Human Capital 2014–2020.

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: https://github.com/Cozmin90/coppeliasim_irb140_deformable_robot.git (accessed on 28 March 2023).

Conflicts of Interest: The authors declare no conflict of interest.

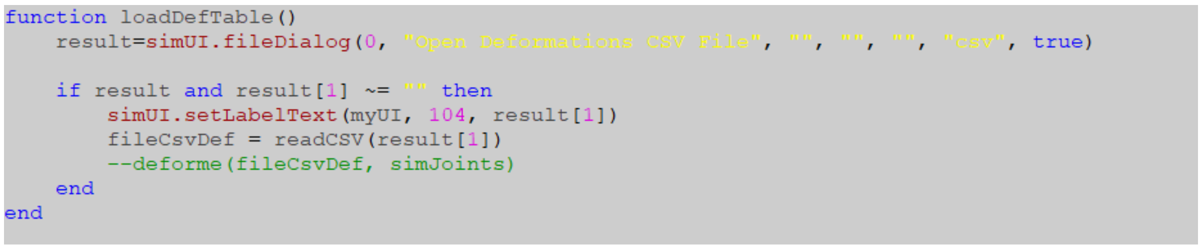
Appendix A

The main functions of the script are:



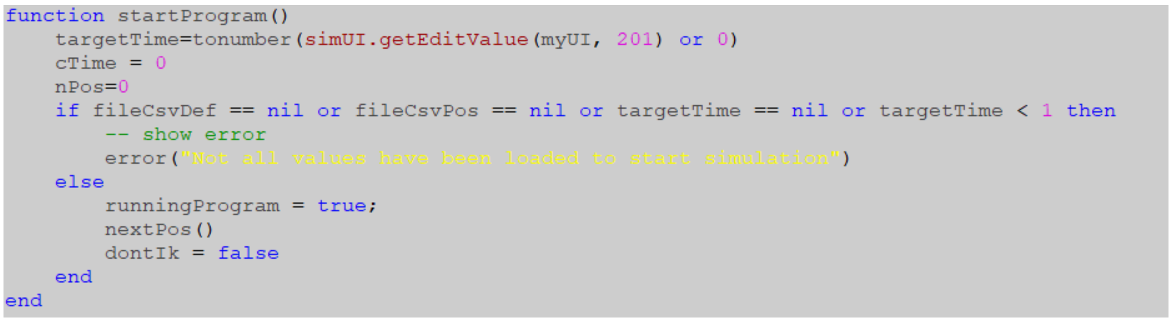
```
function loadTargetsTable()
    local result=simUI.fileDialog(0, "Open Robot Targets CSV File", "", "", "", "csv", true)

    if result and result[1] ~= "" then
        simUI.setLabelText(myUI, 202, result[1])
        fileCsvPos = readCSV(result[1])
    else
        dontIk=true;
    end
end
```

Figure A1. Read targets from csv. file.



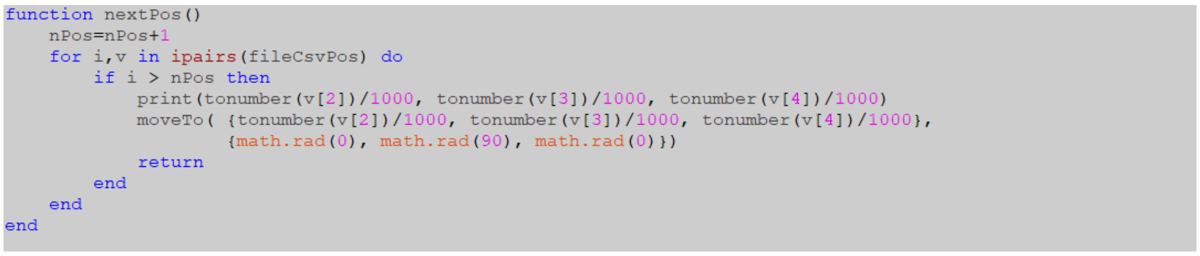
```
function loadDefTable()
    result=simUI.fileDialog(0, "Open Deformations CSV File", "", "", "", "csv", true)

    if result and result[1] ~= "" then
        simUI.setLabelText(myUI, 104, result[1])
        fileCsvDef = readCSV(result[1])
        --deforme(fileCsvDef, simJoints)
    end
end
```

Figure A2. Read deformations values from csv. file.



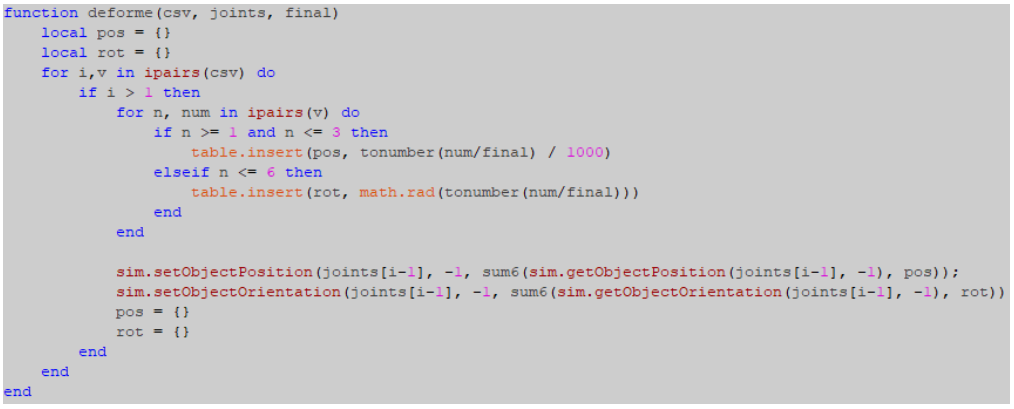
```
function startProgram()
    targetTime=tonumber(simUI.getEditValue(myUI, 201) or 0)
    cTime = 0
    nPos=0
    if fileCsvDef == nil or fileCsvPos == nil or targetTime == nil or targetTime < 1 then
        -- show error
        error("Not all values have been loaded to start simulation")
    else
        runningProgram = true;
        nextPos()
        dontIk = false
    end
end
```

Figure A3. Run script for each target or until time limit.



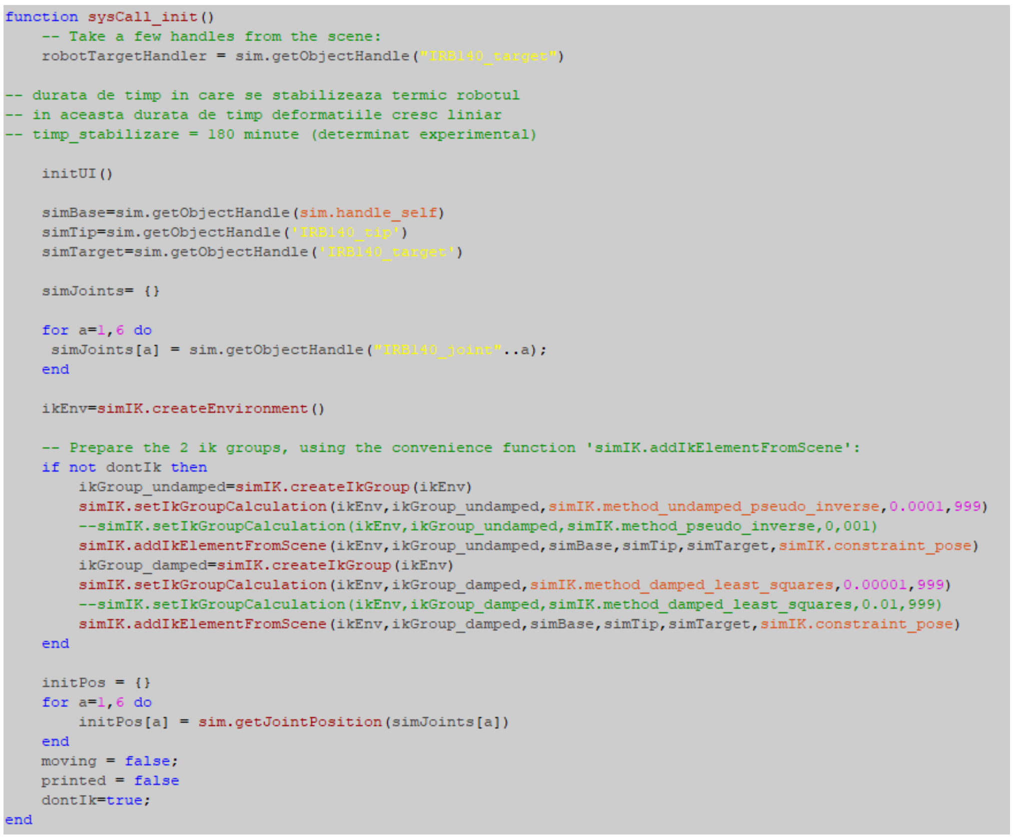
```
function nextPos()
    nPos=nPos+1
    for i,v in ipairs(fileCsvPos) do
        if i > nPos then
            print(tonumber(v[2])/1000, tonumber(v[3])/1000, tonumber(v[4])/1000)
            moveTo( {tonumber(v[2])/1000, tonumber(v[3])/1000, tonumber(v[4])/1000},
                    {math.rad(0), math.rad(90), math.rad(0)})
            return
        end
    end
end
```

Figure A4. Set next robot position.



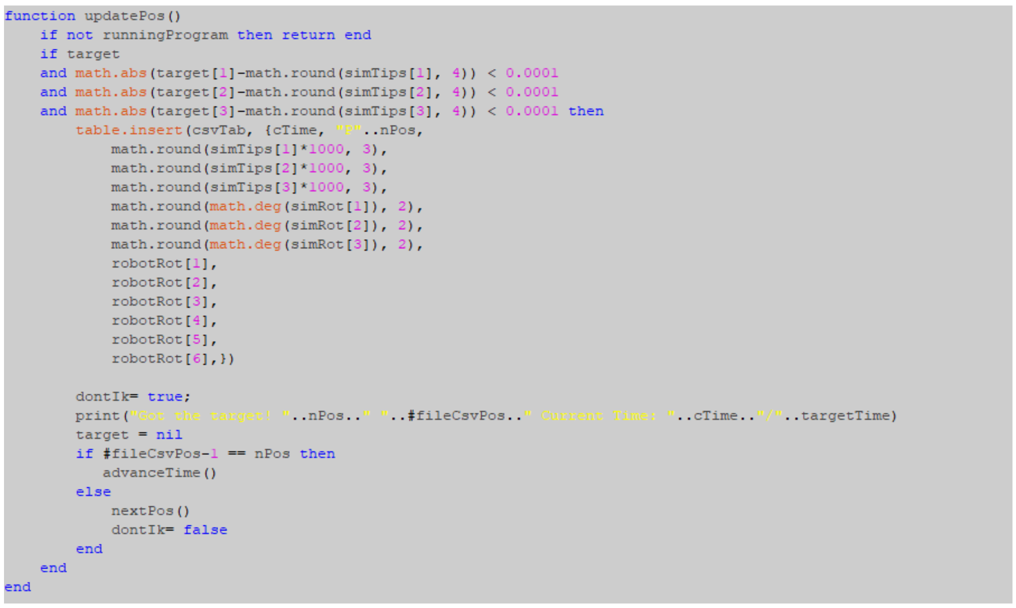
```
function deforme(csv, joints, final)
    local pos = {}
    local rot = {}
    for i,v in ipairs(csv) do
        if i > 1 then
            for n, num in ipairs(v) do
                if n >= 1 and n <= 3 then
                    table.insert(pos, tonumber(num/final) / 1000)
                elseif n <= 6 then
                    table.insert(rot, math.rad(tonumber(num/final)))
                end
            end

            sim.setObjectPosition(joints[i-1], -1, sum6(sim.getObjectPosition(joints[i-1], -1), pos));
            sim.setObjectOrientation(joints[i-1], -1, sum6(sim.getObjectOrientation(joints[i-1], -1), rot))
            pos = {}
            rot = {}
        end
    end
end
```

Figure A5. Deform robot.



```
function sysCall_init()
    -- Take a few handles from the scene:
    robotTargetHandler = sim.getObjectHandle("IRB140_target")

-- durata de timp in care se stabilizeaza termic robotul
-- in aceasta durata de timp deformatiile cresc liniar
-- timp_stabilizare = 180 minute (determinat experimental)

    initUI()

    simBase=sim.getObjectHandle(sim.handle_self)
    simTip=sim.getObjectHandle('IRB140_tip')
    simTarget=sim.getObjectHandle('IRB140_target')

    simJoints= {}

    for a=1,6 do
     simJoints[a] = sim.getObjectHandle("IRB140_joint"..a);
    end

    ikEnv=simIK.createEnvironment()

    -- Prepare the 2 ik groups, using the convenience function 'simIK.addIkElementFromScene':
    if not dontIk then
        ikGroup_undamped=simIK.createIkGroup(ikEnv)
        simIK.setIkGroupCalculation(ikEnv,ikGroup_undamped,simIK.method_undamped_pseudo_inverse,0.0001,999)
        --simIK.setIkGroupCalculation(ikEnv,ikGroup_undamped,simIK.method_pseudo_inverse,0,001)
        simIK.addIkElementFromScene(ikEnv,ikGroup_undamped,simBase,simTip,simTarget,simIK.constraint_pose)
        ikGroup_damped=simIK.createIkGroup(ikEnv)
        simIK.setIkGroupCalculation(ikEnv,ikGroup_damped,simIK.method_damped_least_squares,0.00001,999)
        --simIK.setIkGroupCalculation(ikEnv,ikGroup_damped,simIK.method_damped_least_squares,0.01,999)
        simIK.addIkElementFromScene(ikEnv,ikGroup_damped,simBase,simTip,simTarget,simIK.constraint_pose)
    end

    initPos = {}
    for a=1,6 do
        initPos[a] = sim.getJointPosition(simJoints[a])
    end
    moving = false;
    printed = false
    dontIk=true;
end
```

Figure A6. Compute IK.



```
function updatePos()
    if not runningProgram then return end
    if target
    and math.abs(target[1]-math.round(simTips[1], 4)) < 0.0001
    and math.abs(target[2]-math.round(simTips[2], 4)) < 0.0001
    and math.abs(target[3]-math.round(simTips[3], 4)) < 0.0001 then
        table.insert(csvTab, {cTime, "P"..nPos,
            math.round(simTips[1]*1000, 3),
            math.round(simTips[2]*1000, 3),
            math.round(simTips[3]*1000, 3),
            math.round(math.deg(simRot[1]), 2),
            math.round(math.deg(simRot[2]), 2),
            math.round(math.deg(simRot[3]), 2),
            robotRot[1],
            robotRot[2],
            robotRot[3],
            robotRot[4],
            robotRot[5],
            robotRot[6],})

        dontIk= true;
        print("Got the target! "..nPos.." "..#fileCsvPos.." Current Time: "..cTime.."/"..targetTime)
        target = nil
        if #fileCsvPos-1 == nPos then
           advanceTime()
        else
            nextPos()
            dontIk= false
        end
    end
end
```

Figure A7. Update robot position.

The full CoppeliaSIM scene with the virtual model of the robot and script integrated can be downloaded from GitHub at: https://github.com/Cozmin90/coppeliasim_irb140_deformable_robot.git (accessed on 28 March 2023, a demonstration video along with example and results files are also provided).

References

1. Jiang, Y.; Yu, L.; Jia, H.; Zhao, H.; Xia, H. Absolute Positioning Accuracy Improvement in an Industrial Robot. *Sensors* **2020**, *20*, 4354. [CrossRef] [PubMed]
2. Denavit, J.; Hartenberg, R. A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices. *J. Appl. Mech.* **1955**, *22*, 215–221. [CrossRef]
3. Cerrillo, D.; Barrientos, A.; Del Cerro, J. Kinematic Modelling for Hyper-Redundant Robots—A Structured Guide. *Mathematics* **2022**, *10*, 2891. [CrossRef]
4. Maaroof, O.W.; Dede, M.İ.C.; Aydin, L.A. Robot Arm Design Optimization Method by Using a Kinematic Redundancy Resolution Technique. *Robotics* **2022**, *11*, 1. [CrossRef]
5. Zuha, A.; Saifullah, S.; Arbab, N.; Akhta, U.; Muhammad, A.; Reza, A. Design and Modeling of 9 Degrees of Freedom Redundant Robotic Manipulator. *J. Robot. Control.* **2022**, *3*, 800–808. [CrossRef]
6. Huczala, D.; Kot, T.; Pfurner, M.; Heczko, D.; Oščádal, P.; Mostýn, V. Initial Estimation of Kinematic Structure of a Robotic Manipulator as an Input for Its Synthesis. *Appl. Sci.* **2021**, *11*, 3548. [CrossRef]
7. *ABB Product Manual: IRB 120—3/0.6, IRB 120T—3/0.6*; Document ID: 3HAC035728-001, Revision N; ABB Robotics: Västerås, Sweden, 2017; p. 73.
8. Craig, J.J. *Introduction to Robotics*, 4th ed.; Pearson: London, UK, 2017; pp. 19–102. ISBN 0133489795.
9. Giovanni, L.; Federico, C.; Paolo, R.; Bruno, Z. A Homogenous Matrix Approach to 3D Kinematics and Dynamics—I. Theory. *Mech. Mach. Theory* **1996**, *31*, 573–587. [CrossRef]
10. Waldron, K.; Schmiedeler, J. Kinematics. In *Springer Handbook of Robotics*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 9–33. [CrossRef]
11. Cristoiu, C.; Nicolescu, A. New approach for forward kinematic modeling of industrial robots. *Res. Sci. Today* **2017**, *13*, 136.
12. Cristoiu, C.; Nicolescu, A. New approach for forward kinematic modeling of industrial robots with closed kinematic chain. *Res. Res. Sci. Today* **2017**, *13*, 145–154.
13. Cristoiu, C.; Nicolescu, A. Validation of forward geometric models for ABB robots using virtual models and the software applications CATIA and ABB Robot Studio. *Proc. Manuf. Syst.* **2017**, *12*, 145–153.
14. Gil, A.; Reinoso, Ó.; Marín, J.; Payá, L.; Ruiz, R. Development and Deployment of a New Robotics Toolbox for Education. *Comput. Appl. Eng. Educ.* **2014**, *23*, 443–454. [CrossRef]
15. Djuric, A.; Urbanic, R.J. Chapter 18: Utilizing the Functional Work Space Evaluation Tool for Assessing a System Design and Reconfiguration Alternatives. In *Robotic Systems*; IntechOpen: London, UK, 2012; pp. 361–384. [CrossRef]
16. Baquero-Suárez, M.; Ricardo Ramírez, H. Kinematics, Dynamics and Evaluation of Energy Consumption for ABB IRB-140 Serial Robots in the Tracking of a Path. In Proceedings of the 2nd International Congress on Mechatronics Engineering and Automation, Universidad de la Salle, Bogota, Colombia, 23–25 October 2013. [CrossRef]

17. Cherif, M.; Knevez, J.Y.; Ballu, A. Thermal aspects on robot machining accuracy. In Proceedings of the IDMME—Virtual Concept 2010, Bordeaux, France, 20–22 October 2010; Volume 48, pp. 1–4.
18. Lubrano, E. Calibration of Ultra-High-Precision Robots Operating in an Unsteady Environment. Ph.D. Thesis, École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland, 2011.
19. Cozmin, C. Cercetări privind influența comportării termice a roboților industriali asupra performanțelor acestora. Ph.D. Thesis, "Politehnica" University of Bucharest, Bucharest, Romania, 2020.
20. Andrius, D.; Jurga, S.; Ernestas, S.; Urte, S.; Vytautas, B. Advanced Applications of Industrial Robotics: New Trends and Possibilities. *Appl. Sci.* **2022**, *12*, 135. [CrossRef]
21. Seemal, A.; Philip, W. Realtime Calibration of an Industrial Robot. *Appl. Syst. Innov.* **2022**, *5*, 96. [CrossRef]
22. Nicolescu, A.F.; Cozmin, C.; Dumitrascu, C.; Parpala, R. Recording procedure of thermal field distribution and temperature evolution on ABB IRB 140 industrial robot. *IOP Conf. Ser. Mater. Sci. Eng.* **2018**, *444*, 052023. [CrossRef]
23. Cristoiu, C.; Zapciu, M.; Nicolescu, A.F.; Pupaza, C. Thermal deformation analysis of ABB IRB 140 industrial robot. *UPB Sci. Bull. Ser. D Mech. Eng.* **2002**, *82*, 61–72.
24. Constantin, G.; Dogariu, C.; Bisu, C.F.; Gheorghita, A.; Arotaritei, D.; Ghionea, I. Complex thermomechanical analysis of externally driven main spindles—A case study. *Acta Tech. Napoc. Ser. Appl. Math. Mech. Eng.* **2020**, *63*, 27–38.
25. Cristoiu, C.; Mario, I. Warm-up Temperature Predictions of IRB 140 Robot with Regression Analysis. *Proc. Manuf. Syst.* **2021**, *16*, 69–74.
26. Cristoiu, C.; Stan, L.; Mario, I. Virtual geometric model with dynamic parameters for 6 dof articulated arm robot. *Int. J. Mod. Manuf. Technol.* **2022**, *XIV*, 30–39.
27. Baizid, K.; Cukovic, S.; Iqbal, J.; Yousnadj, A.; Chellali, R.; Meddahi, A.; Devedzic, G.; Ghionea, I. Industrial robotics simulation design planning and optimization platform based on CAD and knowledgeware technologies. *J. Robot. Comput. Integr. Manuf.* **2016**, *42*, 121–134. [CrossRef]
28. Ghionea, I.; Tarbă, C.; Ćuković, S. *CATIA v5: Advanced Parametric and Hybrid 3D Design*; CRC Press: Boca Raton, FL, USA, 2022; ISBN 9781032250069. [CrossRef]