International Journal of Modern Manufacturing Technologies
ISSN 2067–3604, Vol. XIV, No. 2 / 2022
https://doi.org/ 10.54684/ijmmt.2022.14.2.30

VIRTUAL GEOMETRIC MODEL WITH DYNAMIC PARAMETERS FOR 6 DOF ARTICULATED ARM ROBOT

Cozmin Cristoiu, Stan Laurentiu, Ivan Mario

Department of Robotics and Production Systems, POLITEHNICA University of Bucharest, Splaiul Independentei 313, sector 6, 060042, Bucharest, Romania

Corresponding author: Cozmin Cristoiu, cozmin.cristoiu@upb.ro

Abstract: To calculate joints angles of an articulated arm robot, when the coordinates of the point to be reached are known, different calculation methods or iterative algorithms for inverse kinematics (IK) can be used. IK requires that the dimensions of the robot segments and the initial positions of the joints to be known, described, and implemented mathematically, so it is based on the geometric model of the robot. In practice, the geometric modeling of the robots is done considering that all their structural elements are rigid, and their dimensions and positions are considered constant (while in reality the robots suffer certain deformations that can have different causes). This article considers the thermal deformations that a robot suffers during operation which are leading to positioning errors. The deformations are variable during the warm-up period of the robot and become constant after reaching the thermal stabilization level. From this point of view, if it is desired to consider and possibly compensate these thermal induced errors, the elaboration of the geometric model of the robot in the classical way is no longer possible and the geometric parameters must be somehow described as variables. Thermal deformations produce displacements and torsions of the robot elements. Linear and angular deviations may occur from the initial (theoretical) position in all 3 directions of the cartesian axis systems used in robot modeling. This paper presents a technique for creating a virtual model of the ABB IRB140 robot in CoppeliaSim, programming and modeling environment, with the positioning of the axis systems attached to the joints identical to the real position (unlike simplified versions of Denavit-Hartenberg geometric models) and the logic of a custom written software algorithm for automatic deformation of the model.
Key words: virtual model, dynamic parameters, DOF, arm robot.

1. INTRODUCTION

Among the key aspects of an industrial robot, we can name positioning accuracy and repeatability. Usually, the values for these parameters are specified in the robot's data sheet but those are guaranteed only in stable working environment. These parameters can be influenced by various factors, of which a very important role is played by thermal expansion ("drift" as specified in ISO standard [1]). Provision of a robot heating activity before the start of the production activity it is not desirable because warm-up cycles of a robot can take up to a few hours. Finding solutions of correcting the positioning errors caused by the drift of the robots has been a constant concern for robot manufacturers and researchers and between the approaches we may distinguish:

a) Live tracking of the robot TCP with the help of a laser measurement system and real-time correction of robot position, [2, 3]. This solution is among the most precise but also the most expensive due to the need of a laser measurement system (which can be quite expensive) in the working area of the robot. Also, the continuous visibility of the robot TCP (of a tracking object attached to robot) must be assured. Sometimes robot trajectories and configurations may result in a TCP or effector position that cannot be followed anymore thus resulting in an interruption of the tracking laser beam compromising all the measurements.

b) Pre-calibration of robot errors for known targets [4, 5]. Sometimes the robots are programmed to do simple-tasks (e.g., pick-and-place) not implying a big number of points needed to be reached. For these locations, the drift can be measured by different methods and compensated (usually determining by measurements and regression the function of influence of robot temperature upon TCP drift). This is a solution for programs with a small number of targets. It is hard to determine the TCP drift of the robot in the entire working space or in different configurations.

c) Evaluation of robot thermal drift by means of finite element analysis (FEA), [7]. This approach is very useful, and it can be easily updated when the working conditions are changing due to the possibility of adjusting some analysis parameters and re-run the simulation. Thus, most of the papers regarding this approach are treating robot elements as solid blocks and does not consider internal structure and elements of the robot. The thermal behaviour and drift of solid

blocks with homogenous material is different than a real structure robot with real material and specific wall thickness, joints, motors, speed reducers, gears, free spaces and so on. No paper including a coupled transient Thermal FEA with transient Kinematic FEA was found to determine thermal drift of the robot in every configuration of the robot in the entire working space in a working cycle including the warm-up period. Still, FEA is a solution that provides quite accurate approximations of the robot thermal behaviour, but alone it is not enough for a complete compensation solution.

d)The last frontier in terms of thermal compensation solutions is represented by neural networks used for robot calibration [8, 9] or error prediction based on advanced algorithms such as Kalman filter, [10].

It is hard to decide what compensation method it is suitable to apply, because different studies and solutions were applied for different robot structures working in different environment conditions. Every robot has its own stated positioning and repeatability values, its own thermal behaviour and is programmed to do different tasks, and every compensation method has its own downsides (expensive, difficult to implement, big data samples needed to train, supplementary equipment needed, hard to change etc.). In most studies, the big improvement was upon robot repeatability. Some of the best improvements of robot repeatability are stated in [10] (around 78% obtained by predicting errors using neural networks and Kalman filter) and in [11] (around 85% improvement obtained by determining the error drift of TCP during warm-up as a function of temperature and implementing the equation in a parallel task in the robot program on the controller in order to apply corrections of the targets in real time based on temperature).

The approach that will be further presented is aiming to solve some of the problems of the previously mentioned methods of thermal drift compensation while trying to keep the range of improvements as high as possible. First of all, the method depends on a series of data that are determined experimentally by combining 3 procedures (laser-tracker measurements, thermal behaviour recordings with infrared camera, thermal FEA analysis based on a virtual model constructed including internal components, real wall thickness of elements and real materials applied). Second during the experimental phase, not only the drift measurement of the robots TCP was determined but also the drift of each joint. The third different aspect is the fact that the compensation is done at the geometric level of the robot (not based on classical DH models with 4 parameters but on a constructive model with 6 parameters and with real positioning of each robot joint) by altering the geometric parameters and recalculating the IK based on this continuously changing robot model for each target at every moment meaning that the compensation will be applied in the entire robot workspace for every configuration as long as the robot can reach the target).

In the new proposed compensation method, in order to simulate the deformation, error factors were added as variables in the geometric model, both for linear and for angular displacements, factors that can change their value (magnitude) over time or according to another criterion (e.g., temperature). Thus, based on the deformable virtual model, the robot's joints angles can be continuously recalculated and updated (using IK) so that it reaches the programmed points even while the deformation of the robot's structure is happening. Such a deformable virtual model could be used in the development of a thermal drift compensation solution. This can be done in two ways:

- **offline** (The proposed software may be fed with all programmed robot targets (as excel file) and then the angular positions of the robot joints are recalculated (for each target) multiple times for a desired time interval by continuously applying the deformation of the geometric model and constantly recalculating the inverse-kinematics. All results are saved in a text or excel file (.csv) and may be used to replace the targets from the initial robot program.
- **online** (The software must be connected to the robot controller or to the computer that has the control software of the robot installed. Each of the programmed robot targets is read, recalculated, and resent to the robot as a corrected target).

This paper will be further referring only to the **offline** method leaving the online method to be discussed in a following paper.

2. MODELING OF THE GEOMETRIC MODEL

Every virtual robot model is based on two set of components. The parts of the robot and the definition of each robot joint position as an axis or axis system that must be placed in accordingly to the real robot joint. In the following image (figure 1), the orange model is the one downloaded from CoppeliaSIM library, and the white model is a custom version created in Siemens NX12. The geometries of the robots are almost identical (except for some surfaces that do not have an important functional role) but the locations of the joints are different.

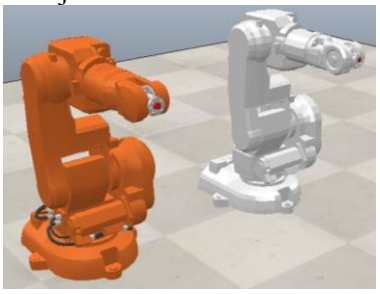


Fig. 1. Robot assembled parts

From the kinematic point of view, robot parts are of no importance. They serve only as a visual representation and the only thing that matters is the length of the parts. No other properties of the parts are considered when computing the robot targets or when moving the robot. Instead, placement of the joints (so of the axis or axis systems) is what is defining the geometric and kinematic model. These are defining and describing where the arm articulations are, where the movement should have place, and the direction of the movement. From software to software, the visual representation of the joints may differ. In CoppeliaSIM, joints are represented as orange cylinders. Length of the cylinders or diameter are not relevant. Only the position and the orientation are important. In figure 2, the joints of two kinematic models of the same robot can be observed. The orange robot is the nominal model while the while the white robot is the custom model that will be used as a deformable model (by deformable geometric model we mean, that during computation of the IK the robot joins positions will be constantly modified meaning that for each iteration the IK is recomputed for a different geometric model. The parts (that are used only as a mock-up for visual representation will not be deformed as well).

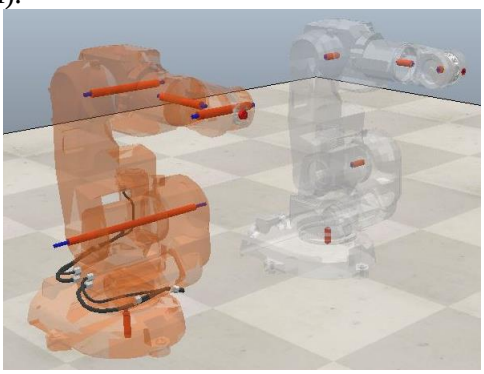


Fig. 2. Joints of the virtual model

As said before, from the kinematic point of view, the only relevant parameters are joint positions, joint angles, and the length of each robot segments. Using these parameters, the coordinates of the endpoint can be calculated via forward-kinematics or, when the targets are known, joint angles can be computed via an inverse-kinematics method. For reasons of mathematical simplification and since other parameters are not necessary for kinematics computation, geometric models of the robot are constructed as simplified models with constant dimensions and rigid elements. This case of ideal simplified models of geometric models is extensively discussed in [12] and [13] from which we draw the following conclusions: simplified geometric models cannot be used in thermal compensation and geometric models should be built using real constructive parameters of the robot (figure 3).

In general, the approach is to place the joints in a single vertical (symmetry) plane and then apply DH (Denavit-Hartenberg) or DHM (Modified Denavit-Hartenberg) algorithm to determine the orientation of each joint related axis system. But form the point of view of the thermal behaviour of the robot, each element deforms leading to deviations of the joints from the real position (not mentioning about that almost no robot has all the joints exactly in a vertical symmetry plane, especially for asymmetrical structures as of IRB 140 robot). Besides that, the ambiguity that comes with DH about the orientation of joints corresponding axis systems, makes it difficult to make use of linear displacements or orientational deviations of each joint (determined by experimental means and FEA methods) as compensation parameters in the geometric model. By trying to introduce some deformation parameters in a geometric model constructed following DH convention, the differential form of transformation equation from one-joint to another (for forward kinematics) is presented in eq (1).

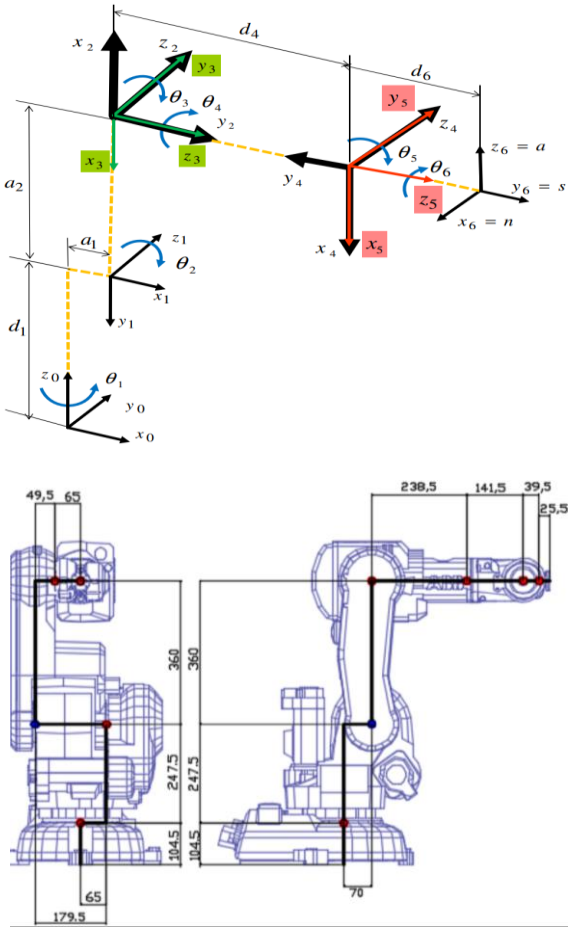


Fig. 3. General simplified geometric models differ from constructive model, [17], [12]

$$dT_{i-1}^{i} = \frac{\delta T(T)_{i-1}^{i}}{\delta\theta}\Delta\theta + \frac{\delta T(T)_{i-1}^{i}}{\partial\alpha}\Delta\alpha + \frac{\delta T(T)_{i-1}^{i}}{\delta a}\Delta a + \frac{\delta T(T)_{i-1}^{i}}{\delta d}\Delta d \quad (1)$$

where T is transformation matrix and θ, α, a and d are DH parameters for describing position of each joint. In case of considering some displacements caused by the thermal deformation of the robot, parameters $\Delta\theta, \Delta\alpha, \Delta a$ and Δd could not be sufficient. In reality, the robot structure deforms so that each joint can deviate from their initial position along X, Y or Z direction and as well can suffer changes from initial orientation (rotation around X, Y or Z). That means not 4 but 6 parameters are needed to consider deformation parameters in the geometric model (the equation should look something like eq. 2).

$$dT_{i-1}^{i} = \frac{\delta T(T)_{i-1}^{i}}{\delta X}\Delta X + \frac{\delta T(T)_{i-1}^{i}}{\delta Y}\Delta Y \frac{\delta T(T)_{i-1}^{i}}{\delta Z}\Delta Z + \frac{\delta T(T)_{i-1}^{i}}{\delta \Delta\theta x}\Delta\theta x + \frac{\delta T(T)_{i-1}^{i}}{\partial\theta y}\Delta\theta y + \frac{\delta T(T)_{i-1}^{i}}{\delta\theta z}\Delta\theta z \quad (2)$$

where: $\Delta X, \Delta Y, \Delta Z$ are linear displacements and $\theta x, \theta y, \theta z$ are angular deviation from initial orientation of each joint.

The key to develop a software solution for thermal compensation is to determine the magnitude of deformations (done previously in [14]) and adjust values continuously with respect to the temperature of robot elements. Each time the temperature changes, the parameters should adjust, and the IK of the robot should be recalculated based on the new values thus correcting the angular values of the robot joints in order to still reach the programmed targets even during robot deformation. CoppeliaSIM was chosen for the modeling and programming of the virtual model because this software allows a programmatically definition of the geometric model with variable parameters that can be altered at demand and that permits programmatically iterative computation for IK solution of a robot based on such geometric model. So, in order to create a script for this, some information is needed to be well-known (previously measured or determined experimentally) such as: exact position of each joint of the real robot, the thermal behaviour of the robot (how it is heading during warm-up or during a known state of the environment where the robot operates) and quantitative information about linear and angular displacements of the joints relative to their initial positions. As a procedure, proposed software compensation can be generally applied to other articulated arm robots but with a particular set of information gathered for each different working scenario (e.g., the warm-up of the robot could happen faster and reach higher values if the robot is set to a faster working speed, or if the robot is exposed to external heating sources that can influence differently it's thermal behaviour).

In CoppeliaSIM the placement of the virtual joints was done considering the position measured on the real robot and keeping the same orientation of the axis systems (all with Z axis up, X facing forward and Y to the left of the robot) with respect to the methodology presented in [12]. The location of each joint on the virtual model can be observed in figure 4 (the white robot model).

If rotational axes are defined and positioned, the only two things needed to be done further are: stacking of the joints hierarchically (figure 5) from base to top and then associate a visual part for each joint so that the parts will move simultaneous with the joint.

3. PROGRAMMING AND SIMULATION

Actions and simulations in CoppeliaSIM can be programmed using scripts and LUA programming language. A simple user interface (UI) was developed with several options for: manual jog, settings and input of thermal deformation data, simulation settings and robot communication. As mentioned before, this paper is focused only on the offline compensation method and the needed input data should be loaded as two tables with:

a). Targets table (programmed robot points)

b). Deformations table (displacements – experimentally determined previously).

A part of the UI is shown in figure 6. Robot targets were defined in RoboDK (robot programming and simulation software with postprocessor for multiple robot models including RAPID for ABB Robots) as equally distanced points as specified in [15] and illustrated in figure 7. The coordinates of programmed targets were exported in a .CSV table format that can be used as input data in the form presented in table 1. The linear and angular deformations of each joint were completed in table 2.

A robot program was created in ROBODK with 45 targets and linear trajectories between the targets. The program was converted in RAPID language and a functional robot program was created ready to be loaded on the robot controlled. At the same time, all joint angles computed in RoboDK were exported as well in tabular .CSV format. After the loading of the input tables in CoppeliaSIM the program was first launched keeping the deformations static (the case when the robot reaches thermal stabilization at the end of the warm-up period). In figure 8 it can be observed how the robot structure tends to warp thanks to applied deformations. The deviations were too small for the eye to be noticed so on the right side they were magnified 20 times.

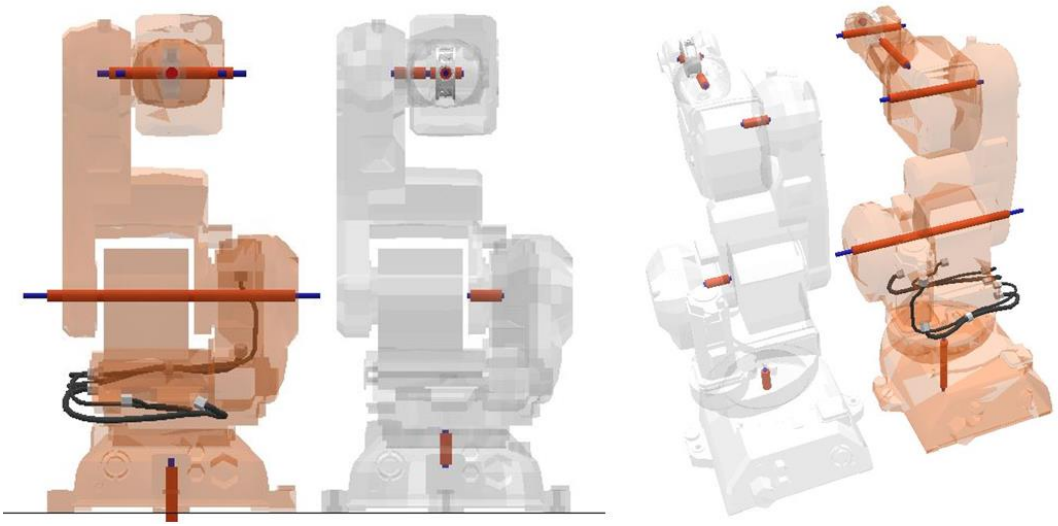


Fig. 4. Placement of the joints of virtual model according to constructive dimensions (white model) vs. general model (orange model)

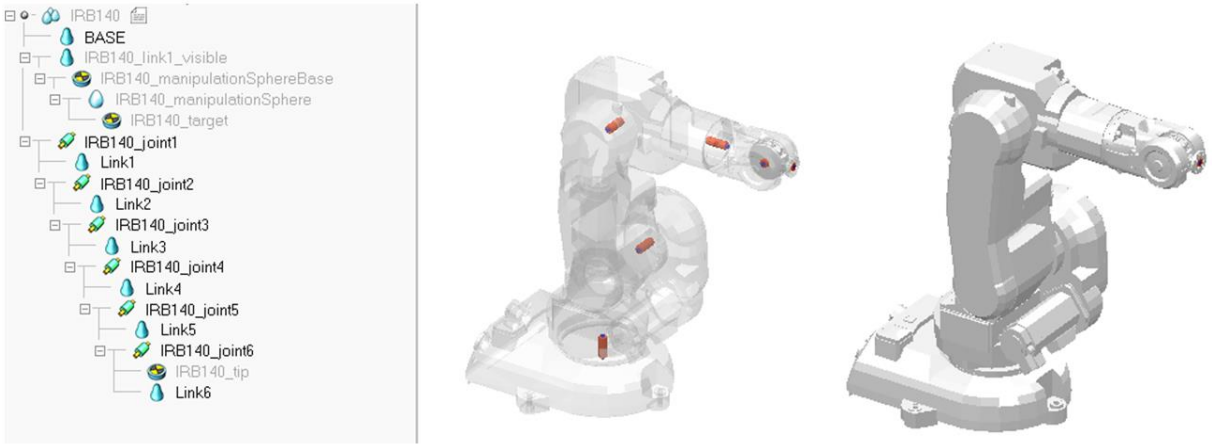


Fig. 5. Stacking of joints and parts in CoppeliaSIM assembly tree

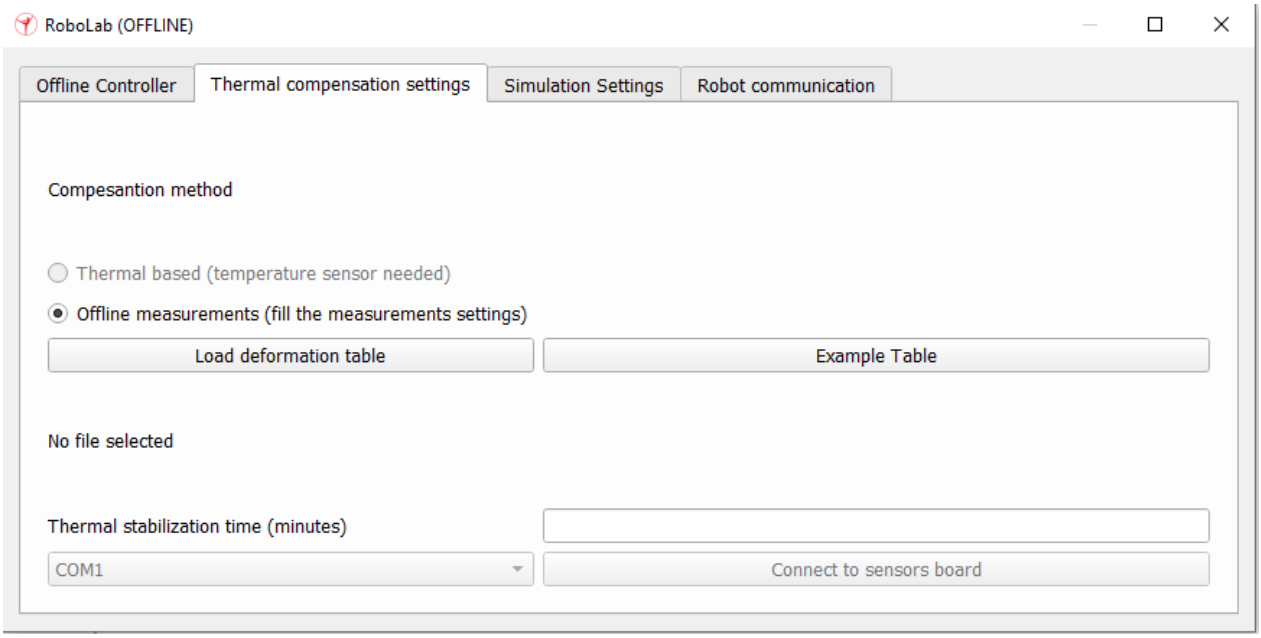


Fig. 6. Screen of application UI

Table 1. Format of robot targets

No. point	X[mm]	Y[mm]	Z[mm]
P1	400	300	250
P2	400	150	250
P3	400	0	250
P4	400	-150	250
P5	400	-300	400

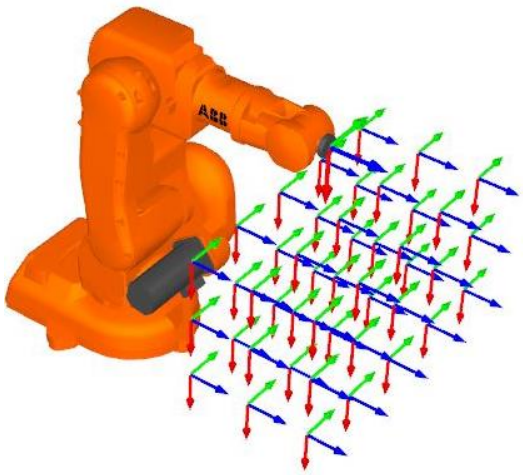


Fig. 7. Robot targets

Table 2. Format of input deformation data

Dy [mm]	Dy[mm]	Dz[mm]	DOx[deg]	DOy[deg]	DOz[deg]
-0.0200698	0.000154	-0.05297	0	-0.011	-0.0137
0	0	0	0.0108	0	0
0.100863597	0.11485519	-0.001420766	0	0.0081	0.0098
-0.0044	0.0241	-0.0149	0	0	0
0	0	0	0	0	0
-0.0162926	0	0	0	0	0

Values were determined experimentally in [14] for IRB.

While the script is running, the robot in CoppeliaSIM goes through all targets that were fed in as input table. At the same time, during operation (simulation), the deformation parameters (loaded as second input table) are applied to the joints altering the geometric model of the robot. The robot position is continuously recalculated (via pseudo-inverse IK method [16]) so that for each iteration, joint values of the robot have different values while trying to reach the same target at different time. The most important aspect is the ability of the robot to reach the programmed targets even if the geometric model of the robot is changing. To test the accuracy of the custom-made script, first the robot program was run in RoboDK (figure 9) and for each robot target, cartesian coordinates as well as orientation angles were recorder.

The script was run in CoppeliaSIM and for each robot target, for every iteration (so considering every time a deformation has occurred) the robot coordinates of the endpoint were recorded as well as it's orientation angles (figure 10).

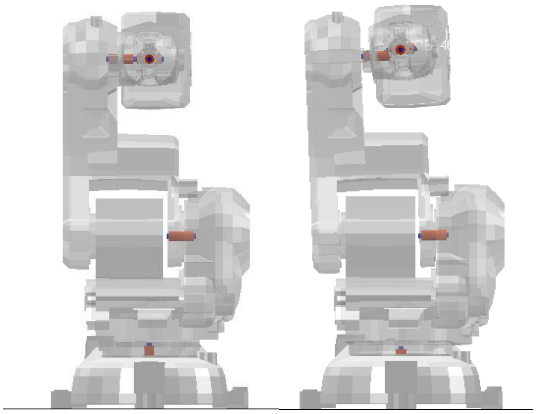
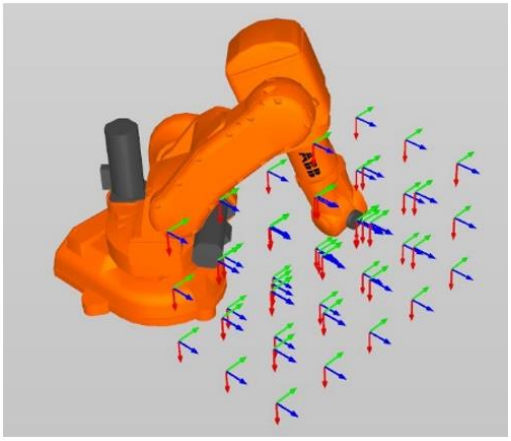


Fig. 8. Robot virtual model before and after the algorithm applies deformations to joints



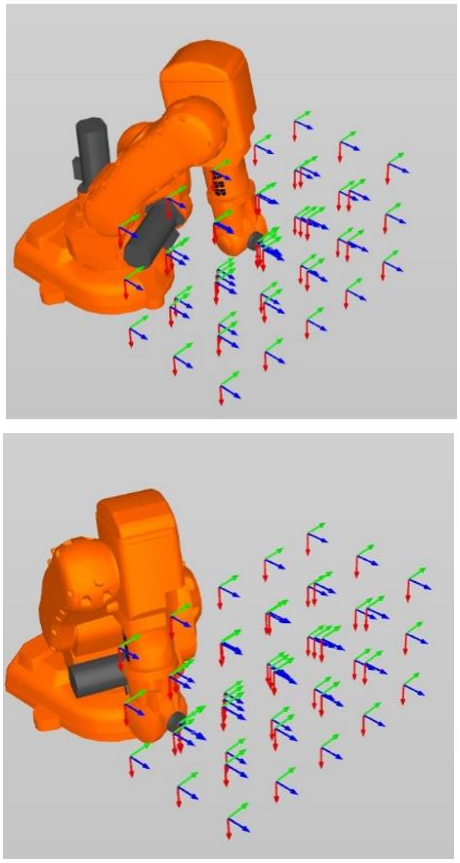
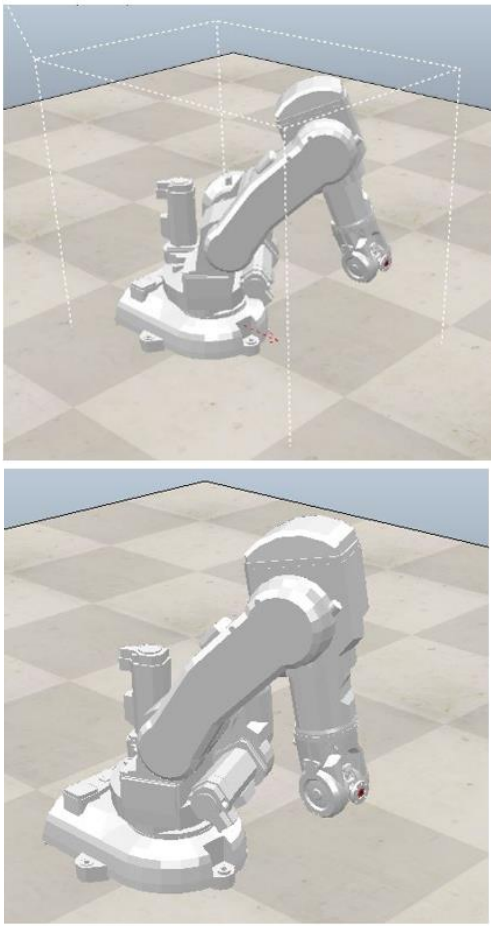


Fig. 9. Undeformed robot running program and going through targets in RoboDK



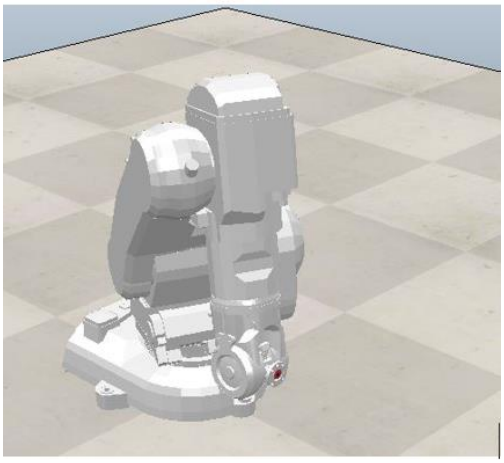


Fig. 10. Robot running program and going through targets while running custom deformation script in CoppeliaSIM

Joint angles of the deformed robot were also recorded at every time and for every target and exported in an Excel file. After this run (including only static deformations as in the case of the robot that passed the warm-up period) another simulation was run but this time by applying variable deformations in time. Figure 11 depicts the logic of the program.

4. RESULTS AND DISCUSSION

As stated in previous chapter, the program created in CoppeliaSIM was run two times. Once with static deformations (simulating the behaviour of the robot after it reaches thermal stabilization – end of warm-up period) and second time with variable deformations (simulating the behaviour of the robot during warm-up). The input data that were fed to the robot program were:

a) parameters of thermal deformation determined previously for each joint (experiments were conducted during warm-up period of the robot working continuously for 3 hours in a relatively cold environment with a start temperature of about 7 deg. C).

b) target coordinates and angles (45 targets disposed as a cube in front of the robot, targets used in the procedure of calibration of the robot and as well targets that composed the robot program and between which the robot was performing continuously movements during previous mentioned warm-up).

The purpose of the project is to compensate thermal drift of the robot which translates into continuously recalculation of robot joints for same targets (but using a slightly modified geometric model of the robot). That means, they key aspect is translating cartesian coordinates of programmed targets in modified (corrected) joint targets of the robot. By applying these joint targets, it must be seen with what degree of accuracy the programmed targets cand still be reached. For both cases (static and variable deformations) programmed target coordinates and orientation were

compared to coordinates and angles of reached targets. Original robot joint values were also compered with deformed robot joint values (compensated values). Due to the big amount of data, in table 3 and table 4 an example of data comparison for only three targets are presented.

The static case resulting:

a) maximum deviation between programmed target coordinates and reached target after deformation and compensation: 0.007 mm.

b) maximum orientation deviation between programmed target orientation and reached target orientation: 0.001 deg.

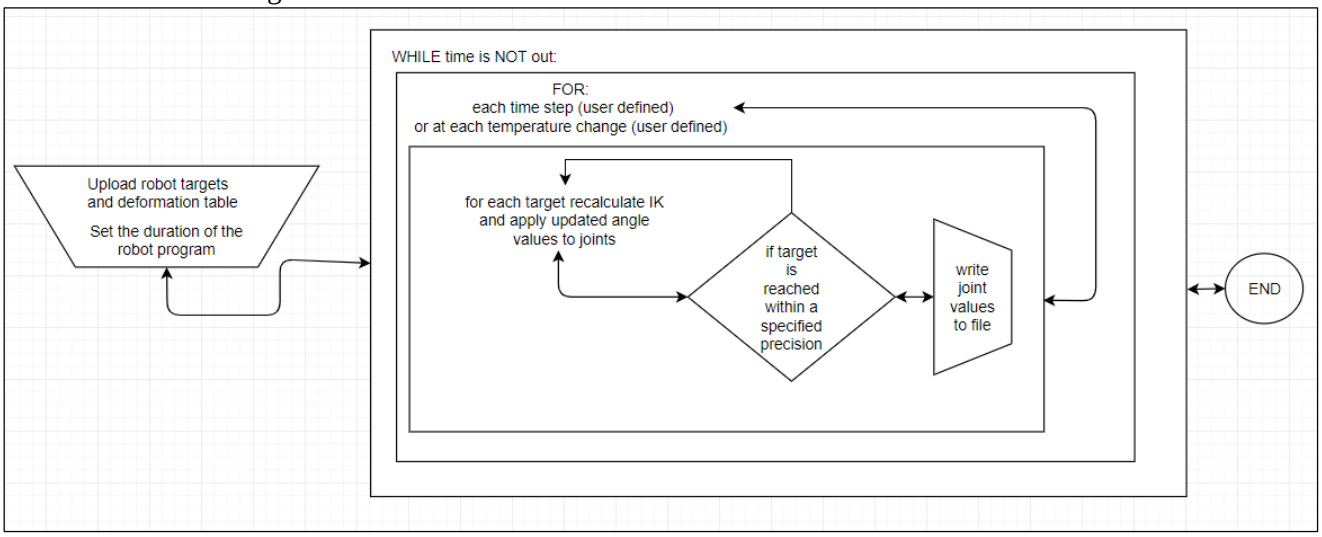


Fig. 11. Robot running program and going through targets while running custom deformation script in CoppeliaSIM

Table 3. Programmed target coordinates, angles, and robot joints values in RoboDK (undeformed robot)

Target	Coordinates			Orientations			Joint angles – RoboDK (undeformed robot)					
	X	*Y*	*Z*	*Ox*	*Oy*	*Oz*	*J1*	*J2*	*J3*	*J4*	*J5*	*J6*
P19	550	150	550	0	90	0	17.19	14.3	9.09	37.92	-28.74	-34.34
P20	550	300	550	0	90	0	31.74	23.57	-3.25	60.69	-37.11	-54.85
P21	550	300	400	0	90	0	31.74	35.6	4.49	43.85	-49.41	-32

Table 4. Reached target coordinates, angles, and robot joints values in CoppeliaSIM (deformed robot)

Target	Reached target coord. . Coppelia			Reached target orient. Coppelia			Joint angles – Coppelia (deformed robot)					
	X	*Y*	*Z*	*Ox*	*Oy*	*Oz*	*J1*	*J2*	*J3*	*J4*	*J5*	*J6*
P19	549.997	150.002	549.995	0	90	-0.01	17.1851	14.29629	9.09239	37.92219	-28.7301	-34.3396
P20	549.998	300	549.999	0	90	0	31.73811	23.56822	-3.24707	60.68823	-37.1049	-54.855
P21	549.998	299.999	400	0	90	0	31.73809	35.60116	4.48659	43.84593	-49.4092	-32.0036

Differences have been computed for all targets and for For the second case (variable deformations during robot-warmup) the program in CoppeliaSIM was set to run for about three hours. For each iteration, reached targets, orientation and joint angles were recorded. In the same manner as previous, a file with 8145 rows was saved at the end, containing all the specified data recalculated with deformations applied to the robot every minute. For this case the results were following:

a) maximum deviation between programmed target coordinates and reached target after deformation and compensation: 0.073 mm.

b) maximum orientation deviation between programmed target orientation and reached target orientation: 0.01 deg.

Values obtained in the case of variable deformations do not seem as accurate as in the static case. It seems that the error increases with the simulation time. The error of reaching programmed targets is rising from 0.03 mm up to 0.073 during the 180 min of simulation time. Increasing error values cand be graphically observed in figure 12.

The measurements on the real robot [14] showed a positioning error caused by thermal deformations (in

the given working condition and environment) of about 0.097 mm.
Given this magnitude of deviation of the endpoint of the robot it means that using the proposed program for compensation of the thermal deformation in the static case (deformed robot after warm-up) the improvement of positioning accuracy is about 93% (deviation of thermally compensated robot of 0.007 mm vs. thermal deviation of characteristic point of the robot 0.097 mm).
In the transitory case, during 180 min of warm-up of the robot, the improvement of accuracy is about 70% in the first 60 min (max deviation about 0.03 mm from programmed targets), about 38% from min 60 up to min 120 (0.06 mm deviation) and about 24.7 % (max deviation of 0.073 mm) in the last 60 operating minutes. At this moment, this could be improved by forcing the last part of simulation with values from the static case where accuracy is better by a factor of ten.
It is unclear why the error is increasing in steps, and it is an issue to be further investigated and improved but even in this state the improvements of the robot accuracy by compensating the thermal deformations by proposed method is significant.

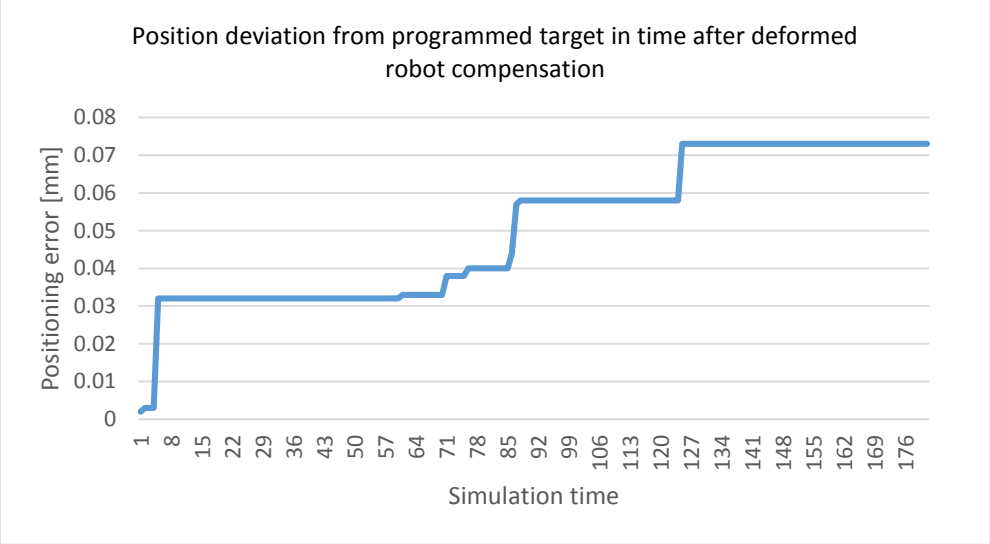


Fig. 12. Increase of error values during 180 min of simulation time.

Acknowledgment: „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 Sectoral Operational Program Human Capital 2014-2020.” Work done as postdoctoral research activity at University Politehnica of Bucharest, Faculty of Industrial Engineering and Robotics.

5. CONCLUSIONS

The work presented in this paper focuses on developing a correction algorithm that uses CoppeliaSIM as a framework. The developed algorithm essentially represents a software module integrated with the user interface and 3D working environment of CoppeliaSIM that can be used to improve robot reached targets by considering the errors induced by the thermal drift issue. The improvement of path targets is achieved by altering the corresponding joint parameters in accordance with both experimental results obtained through laser tracking and thermal camera measurements, as well as performing FEA on the deformable model of the robot. The methods used to determine the robot deformations are not essential to the development of the correction algorithm, because the input data required consists only of a set of displacement values (errors) in .csv format.
The development of the correction algorithm represents the main original contribution of the project. The research performed regarding the state of the art in the field had not previously revealed any drift compensation method based on inverse kinematics recalculation for a variable geometric model. Thus, the development of the correction algorithm presented in this paper can be used as a basis for future research projects and developments in the field of error corrections for robot positioning and path accuracy as well as new methods of trajectory planning. It is the first model that includes six correction parameters for each of the six joints (three for linear displacement and three for angular displacement). Although the developed algorithm includes only corrections for errors induced by thermal drift, it can be easily adapted to include correction variables for other sources of errors (for example, joint elastic displacement errors).
Even in this early stage of algorithm development, the results discussed above show that an improvement of 70% can be obtained for the robot repeatability if the thermal stabilisation threshold has been achieved. For the robot warm-up session, the implementation of the

algorithm showed different improvements, from 70% during the first hour and dropping to 38% in the second hour and to 24% the end of the 3-hour warm-up period. Regarding this aspect, one of the future research directions will be focused on improving the robot repeatability during the warm-up sessions. This will have an impact on the general productivity of robotic systems, given that it will create the conditions for robot tasks to be carried more precisely during the warm-up.

6. REFERENCES

1. ISO 9283:1998. *Manipulating Industrial Robots: Performance Criteria and Related Test Methods*. 2; *ISO*: Geneva, Switzerland.
2. AccuBeam DYNALOG System (2019): *Robot Temperature Compensation & TCP/Robot*, Mastering Recovery; Bloomfield Hills, MI, USA; Available online: https://www.dynalog- us.com/accubeam-robot-temperature-compensation-system.htm.
3. WIEST AG-Kalibriersysteme (2019): *Temperature Compensation*; Neusäß, Germany, Available online: https://www.wiest-ag.de/en/temperaturkompensation.html
4. Lubrano E, Clavel R (2010). *Thermal Calibration of a 3 DOF Ultra High-Precision Robot Operating in Industrial Environment*, Proceedings of the 2010 IEEE International Conference on Robotics and Automation, Anchorage, AK, USA, 3692–3697, ISBN 978-1-4244-5038-1.
5. Mares M., Otakar H, Lukáš H. (2020). *Thermal error compensation of a 5-axis machine tool using indigenous temperature sensors and CNC integrated Python code validated with a machined test piece*, Precis. Eng., 66, 21–30.
6. Elatta A.Y., Li P.G, Fan L.Z., Yu D. (2003). *An Overview of Robot Calibration*, Inf. Technol. J., 3, 74–78.
7. Li R., ZhaoY. (2016). *Dynamic error compensation for industrial robot based on thermal effect model*, Measurement, 88, 113–120.
8. Abdulshahed, A., Longstaff A.P., Fletcher S., Myers A. (2013). *Application of GNNMCI (1, N) to environmental thermal error modelling of CNC machine tools*, The 3rd International Conference on Advanced Manufacturing Engineering and Technologies; KTH Royal Institute of Technology: Stockholm, Sweden; 253–262.
9. Wang D., Bai Y., Zhao J., (2012). *Robot manipulator calibration using neural network and a camera-based measurement system*, Trans. Inst. Meas. Control, 34, 105–121.
10. Liu B., Zhang F., Qu X., (2015). *A Method for Improving the Pose Accuracy of a Robot Manipulator Based on Multi-Sensor Combined Measurement and Data Fusion*. Sensors, 15, 7933–7952.
11. Vocetka M., Bobovský Z., Babjak J., Suder J., Grushko S., Mlotek J., Krys V., Hagara M. (2021). *Influence of Drift on Robot Repeatability and Its Compensation*, Appl. Sci., 11, 10813. https://doi.org/10.3390/app112210813
12. Cristoiu C., Nicolescu (2017). *A. New approach for forward kinematic modeling of industrial robots*, Research and Science Today Supplement, 2, 136-144.
13. Cristoiu C., (2019). *Research on the influence of the thermal behaviour of industrial robots on their performanc*e, Phd. Thesis, Politehnica University of Bucharest. Available online: https://rei.gov.ro/teze-doctorat, search thesis code: F-CA-28543/11.02.2020
14. Cristoiu C, Zapciu M, Nicolescu A. (2020). *Thermal deformation analysis of ABB IRB 140 industrial robot*, U.P.B. Scientific Bulletin, Series D: Mechanical Engineering, 82, 61-72.
15. Nicolescu A., Cristoiu. C, (2018). *Status check and calibration method for robot ABB IRB 140*, The 8th International Conference on Advanced Concepts in Mechanical Engineering, IOP Conf. Series: Materials Science and Engineering, 444, 1-14.
16. SR Buss (2009). *Introduction to Inverse Kinematics with Jacobian Transpose, Pseudoinverse and Damped Least Squares methods*, Department of Mathematics, University of California, San Diego; Available online: http://graphics.cs.cmu.edu/nsp/course/15-464/Spring11/handouts/iksurvey.pdf
17. Djuric A.., Urbanic J., (2012). *Utilizing the Functional Work Space Evaluation Tool for Assessing a System Design and Reconfiguration Alternatives*, Robotic Systems - Applications, Control and Programming, ISBN: 978-953-307-941-7, 361-386.

Received: May 19, 2022 / Accepted: December 15, 2022 / Paper available online: December 20, 2022