Journal of Electrical Engineering and Information Technologies, Vol. **1,** No. 1–2, pp. 37–43 (2016)
125
In print: ISSN 2545–4250
Received: September 18, 2016
On line: ISSN 2545–4269
Accepted: December 6, 2016
UDC:007.52:04.896

Original scientific paper

AVOIDING HEAVY COMPUTATIONS IN INVERSE CALIBRATION PROCEDURE FOR 7 DOF ROBOT MANIPULATOR

Samoil Samak[1], Igor Dimovski[1], Mirjana Trompeska[1], Vladimir Dukovski[2]

[1]*Institute for Advanced Composites and Robotics,*
Prilep, Republic of Mcedonia
[2]*Faculty of Mechanical Engineering, "Ss. Cyril and Methodius" University in Skopje,*
Rugjer Bošković bb, P.O. box 674, 1001 Skopje, Republic of Macedonia
igord@iacr.edu.mk

A b s t r a c t: Procedure for determining commanded coordinates in machine space if desired coordinates are given is inverse calibration. A large amount of data is considered after measurement procedure and it is essential to locate desired point in the real space which is skewed due to measured geometric errors. The machine workspace is divided to cells using measurement points. It is depicted the importance of finding the proper cell in skewed 3D lattice, for calibration of translational axes of ATL machine with large workspace. To calibrate 7 DOF robot manipulator, this algorithm is extended. The problem of finding the proper cell in 7D skewed grid needs heavy computations and takes significant amount of computational time. Few ideas for avoiding these computations are described and the influence on the final precision of the calibration procedure is explored.

Key words: inverse calibration; geometric errors; robot manipulator

ПРОЦЕДУРА НА ИНВЕРЗНА КАЛИБРАЦИЈА ЗА НАМАЛУВАЊЕ НА ПРЕСМЕТКОВНИТЕ ОПЕРАЦИИ КАЈ РОБОТ УПРАВУВАН СО 7 СТЕПЕНИ НА СЛОБОДА

А п с т р а к т: Процедурата за одредување координати кои можат да се контролираат во машинскиот простор со зададени координати се нарекува инверзна калибрација. Во текот на мерната постапка поради големата количина на податоци е особено важно да се одреди точната позиција на посакуваната точка во реалниот простор, која се изместува поради измерените геометриски грешки. Со користење на мерните точки машинскиот работен простор се дели на ќелии. За калибрација на транслаторните оски од ATL машина со голем работен простор е особено важно да се одреди соодветна ќелија во искривената 3D решетка. Алгоритамот е проширен за калибрација на робот со 7 степени на слобода (7 DOF) за маневар. Проблемот на наоѓање на соодветната ќелија во 7D искривената мрежа е комплексен и бара огромни пресметковни капацитети и време за нивно извршување. Во трудот се прикажани неколку начини за избегнување огромни пресметковни операции и објаснето е влијанието на калибрационата постапка врз крајната прецизност.

Клучни зборови: инверзна калибрација; геометриски грешки; робот за маневар

INTRODUCTION

In composite industry, Automated Fiber Placement (AFP) and Automated Tape Layup (ATL) technologies are used for producing large parts. Inaccuracy in position or orientation of the AFP/ATL head, as end-effector of robotized machine, may cause some defects of the laminate and as well of the final product [1], [2].

To enhance accuracy of such robotized machines, comprehensive calibration procedure has been developed for linear axes of a 6 DOF ATL machine [3]. Due to the standards ISO 230-1:2012 [4] and ISO 230-2:2014 [5], as well to the ISO

technical report [6], 21 volumetric errors was taken in account.

Since all measurements were made on the machine with large workspace, large amount of data were obtained. Original algorithm for 3D volumetric calibration is implemented in Matlab and the results are verified using comparative analysis, comparing compensating predictions with the results of established calibration TRAC-CAL software. The algorithm is based on mathematical model for linear approximation of the total displacement error in the interior of the machine's workspace. That is nonparametric calibration, named black-box [7].

A comprehensive calibration algorithm has to cover both forward and inverse calibration. Forward calibration means to find actual coordinates if the nominal ones are given, both in machine space. If the desired coordinates are given and commanded coordinates should be found, it is the inverse calibration.

The measurement points for each axis divide the range of that axis to intervals. Crossing them, entire machine's workspace is divided to cells which dimension depends on the number of degrees of freedom (DOF) of the calibrated machine. In traditional approach, approximation of the total error is spanned over the workspace, taking into consideration the measured errors in the knots of the cells [7], [8], [9]. In our approach, we do linear approximation for every cell separately. Because of that, it is extremely important to determine the proper cell which contains the considered point.

Such idea is extended for calibration machines with more than 3 DOF. Virtual model for a 7 DOF robot manipulator is built. All kinematic calculations needed for calibration are done using screw theory [10]. Errors for all 7 axes are randomly generated, ensuring they are in expected range and distribution, similarly as in the case with 3 linear axes and their errors.

DETERMINE THE PROPER CELL IN 3D CASE

The most sensitive step in the inverse calibration procedure from aspect of computational time is determination of the grid proper cell for given desired coordinates in machine space.

After obtaining measurement results, all measurement points are stored for each axis separately. For 3 translational axes X, Y and Z, measurement was performed and the number of measurement points and their range are shown in Table 1.

Table 1

Number of measurement points

	Number of points	Min (mm)	Max (mm)
X	322	940	8965
Y	142	520	4045
Z	50	-1200	25

This way, the machine's workspace is divided to 2,217,789 3D cells. Combining division point for each axis, nominal coordinates of the knot is obtained:

$$\mathbf{P_{nom}} = [x_{i,\mathrm{nom}}, y_{j,\mathrm{nom}}, z_{k,\mathrm{nom}}]^{\mathrm{T}}. \quad (1)$$

Using the kinematic model of the machine, actual coordinates are calculated and stored:

$$\mathbf{P_{act}} = [x_{i,\mathrm{act}}, y_{j,\mathrm{act}}, z_{k,\mathrm{act}}]^{T}. \quad (2)$$

For storing all actual coordinates for all knots, large amount of memory is needed. That is impractical in higher dimension, so prior calculating and storing the actual knots is not done in the simulation of calibration model for 7 DOF machine.

Also, for each cell separately, coefficients of the error approximation polynomials are calculated and stored.

That way, forward calibration step is completed. Entire machine's workspace has two representations. Knots nominal coordinates determine the ideal workspace with ideal boxes as cells. Actual coordinates determines the real workspace, whose axes are skewed.

Figure 1 shows the way axes are skewed. The ideal workspace and its corresponding skewed workspace are visualized on Figure 2.

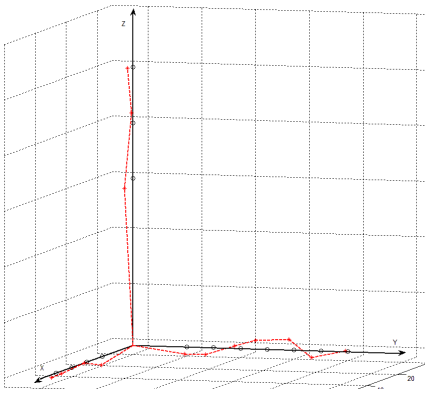

Fig. 1. The skewed coordinate axes

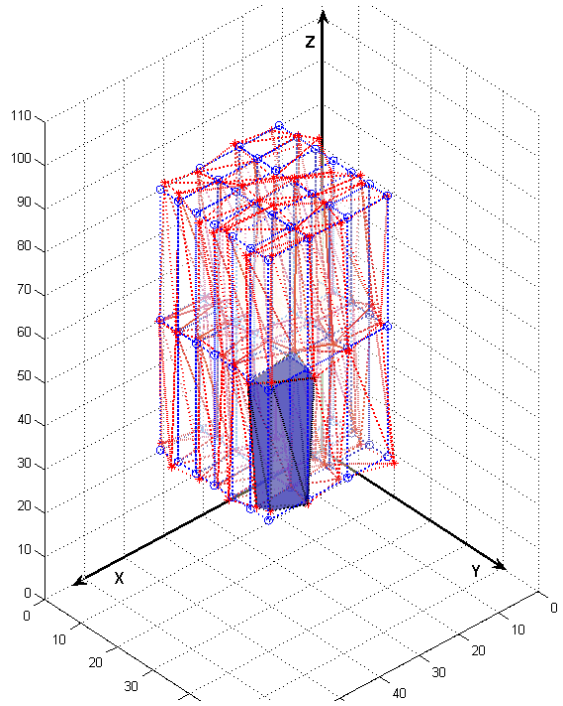

Fig. 2. The ideal and skewed grid of the worksapce

Let

$$\mathbf{Q_{des}} = [x_{\text{des}}, y_{\text{des}}, z_{\text{des}}]^{\text{T}} \tag{3}$$

be the 3D point represent desired coordinates in machine space. Clearly, those coordinates are in the real, skewed space. One has to determine the proper real cell where the point $\mathbf{Q_{des}}$ is located. This cell is polyhedron and is the convex hull of the set of 8 vertices, determined with their actual coordinates.

First, bisection method is applied to find the default cell. That means, the nominal values are used as partition for each axis, but appropriate actual knots are used to check whether the point $\mathbf{Q_{des}}$ is inside the hull. If it does, the proper cell is found and it is the default one. If it doesn't, all neighbor cells, maximum 26 cells are tested to find the proper one.

The point on top left on Figure 3 shows the possibility point not to be in the default skewed cell.

An original algorithm is developed and implemented in Matlab, that determines whether the point $\mathbf{Q_{des}}$ is inside the hull or not. This algorithm is based on linear algebra tools and determines whether the given point is on the same side from all the planes determined by triples of polyhedron vertices lying on the same face.

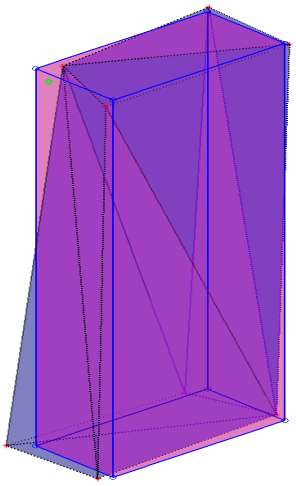

Fig. 3. Desired point outside the default cell

Similar approach is tested using built-in Matlab functions based on Delaunay triangulation and tsearch function. In both cases, the same result as proper cell is obtained, and both procedures are time consuming.

After determining the proper cell, appropriate coefficient are loaded and inverse procedure is performed to find the commanded coordinates

$$\mathbf{Q_{com}} = [x_{\text{com}}, y_{\text{com}}, z_{\text{com}}]^{\text{T}}. \tag{4}$$

With obtaining the commanded coordinates $\mathbf{Q_{com}}$ when desired ones $\mathbf{Q_{des}}$ are given, the inverse calibration procedure is done.

Since determination of the proper cell spends most of entire computational time for the inverse calibration procedure, the question is how this determination is crucial for the accuracy after compensation and is it possible to avoid such computations.

Analysis of the data shown on Table 2, points out the importance of using right coefficients in inverse calibration procedure, since if default cell is taken instead of the proper one, ignoring the possibility of case shown on Figure 3, the precision after the calibration is much worse.

Taking the proper cell, deviation of the end-effector from the desired position is almost zero, since calculated deviations are in range close to the Matlab numerical accuracy. Calculated deviations in case of taking the default cell instead of the proper one are shown in the third column of the Table 2. One can conclude those deviations may be significant and finding the proper cell to use the proper coefficient for error approximation may be crucial.

Table 2

Proper cell vs. default cell in 3D

Point inside procedure	Number of callings	Default cell deviation (mm)	Ratio-computational time proper cell over default cell
Point 1	5	$51.9 \cdot 10^{-6}$	1.98
Point 2	19	$89.2 \cdot 10^{-6}$	2.74
Point 3	3	$11.4 \cdot 10^{-6}$	2.47
Point 4	4	$20.4 \cdot 10^{-6}$	2.11
Point 5	16	$21.6 \cdot 10^{-6}$	1.96
Point 6	4	$20.7 \cdot 10^{-6}$	1.80
Point 7	5	$11.2 \cdot 10^{-6}$	2.18

From aspect of computational time, in 3D case, finding the proper cell takes less than 3 times than the time needed for just taking the default cell. So, we decided always to search for the proper cell in the inverse calibration procedure in our calibration algorithm for 3 DOF machine.

CALIBRATION OF 7 DOF ROBOT MANIPULATOR

To test possibility of extension of this calibration procedure including rotational axes, virtual 7 DOF robotized machine is simulated. The kinematic chain contains 3 translational and 4 rotational axes. Due to the ISO standards [4], [5], [6], in total 61 geometric errors must be considered. All of them are included in the kinematic model. Forward and inverse kinematics procedures are implemented in similar manner as in [11] and [12], based on screw theory. Ideal forward kinematics and error forward kinematics procedures are implemented and the last one includes all 61 geometric errors. That makes possible to compare nominal and actual pose in pose space and determine the deviations for position, but for orientation as well.

There are 42 position dependent geometric errors, 6 errors for each axis. Therefore, measurement points must be considered for all 7 axes. Taking only 20 – 50 measurement points for the rotational axes, the number of combination for calculation of number of knots in the workspace in machine coordinates is extremely enlarged. That makes simple extension of the calibration procedure from 3 to 7 DOF impossible. It is not possible to calculate and to store actual machine coordinates for all the knots and error approximation polynomial coefficients for all the cells.

For given desired coordinates in machine space

$$\boldsymbol{\Theta}_{\mathbf{des}} = [\theta_{1,\mathrm{des}}, \theta_{2,\mathrm{des}}, \theta_{3,\mathrm{des}}, \theta_{4,\mathrm{des}}, \theta_{5,\mathrm{des}}, \theta_{6,\mathrm{des}}, \theta_{7,\mathrm{des}}]^{\mathrm{T}} \tag{5}$$

real time calibration procedure is designed, including determination of the default cell based on bisection method, calculations of actual coordinates of all knots, generating error approximation polynomial coefficients for the default cell and algorithm to determine the proper cell in skewed 7D space. The last one is critical, since it is time most consuming. The goal is to obtain the appropriate commanded coordinates in machine space

$$\mathbf{Q}_{\mathbf{com}} = [\theta_{1,\mathrm{com}}, \theta_{2,\mathrm{com}}, \theta_{3,\mathrm{com}}, \theta_{4,\mathrm{com}}, \theta_{5,\mathrm{com}}, \theta_{6,\mathrm{com}}, \theta_{7,\mathrm{com}}]^{\mathrm{T}} \tag{6}$$

in real time.

Large data set for 42 position dependent geometric errors is generated randomly. These errors are in the similar range and distribution as the data obtained for calibrating 3 DOF machine. Remaining 19 position independent geometric errors are randomly generated in the same manner. All 61 geometric errors are included in the calibration procedures.

The simulation for calibration of a 7 DOF robot manipulator is implemented in Matlab as well.

Forward calibration procedure is performed in reasonable computational time. That means if the nominal coordinates in machine space are given, the actual coordinates could be calculated in real time.

For inverse calibration procedure, searching for proper cell in a 7D skewed space is too much time consuming and makes idea impossible to realize in real time.

AVOIDING HEAVY COMPUTATIONS IN 7D

A heavy computations are needed to determine whether some 7D point in inside the convex hull of a set of 7D points.

Mathematical foundation of the problem is in combinatorial topology [13]. There are 128 vertices in 7D case and the cell in ideal space is simplicial complex. Its underlying space is linear polyhedron in dimension 7. When actual coordinates are calculated, the set of 128 points are obtained, and one needs to find the convex hull of such set of 7D points. Partition of that hull to 7-simplices is needed and test if the given point is inside any 7-simplex.

The fastest way to make such partition is extension of Delaunay triangulation method [14] in higher dimension [15], [16].

Since entire procedure is implemented in Matlab, its built-in functions Delaunayn and tsearchn are used. Correctness is visually verified in 2D and 3D and few tests are performed to ensure the same answer is obtained with using these functions and using topological test over all combinations of 7-simplees.

For most points given with desired coordinates in machine space, only one call of these two functions is sufficient, since most of that points would lie in the default cell and searching for that cell is easy and fast, due to bisection method. But, even one calling of these functions is time consuming and makes entire calibration procedure impracticable for real time computing. For such "good" point, approximately 94% of computational time takes the check it is inside the default cell. The function Delaunayn spends approximately 6.8 seconds and the function tsearchn spends around 3 seconds. It is clear, that these heavy computations should be avoided.

Naive idea to assume the point is "good" and it lies in the default cell, could lead to undesired deviations as 3D case showed.

If any coordinate of desired point (5) is near the measurement point in some predefined tolerance, there is chance desired point not to be "good" one. Ideally, in that case, the neighbor cells should be checked, but it is totally impractical, since the number of neighbors is large, and computational time for one check is large as well. But, taking the neighbor cells into account, without insisting to determine which one is the proper, is the right idea for avoiding heavy computations.

That means, for such potentially "bad" point with desired coordinates in machine space, to extend the neighborhood and instead of taking only one linear polyhedron in dimension 7 as proper cell, to take few of them, at most $128 = 2^7$ such cells. Calculating the actual coordinates and approximation coefficient for all neighbour cells is much cheaper than performing Delaunay 7D test.

Taking the average of appropriate coefficients of all such neighbour cells, for each dimension separately, the new approximation polynomial is obtained. The inverse calibration procedure is performed and the commanded coordinates are obtained, without calling the heavy computation functions.

One may expect to have losing of accuracy, so obtained commanded coordinates in machine space should be transformed in pose space and the deviations of position and orientations from the ideal ones should be calculated, and they should be compared against the appropriate deviations if the proper cell was taken in inverse calibration procedure.

RESULTS

Nine 7D points were sampled, all with desired coordinates (5), and simulation is done for all of them. Three of sampled points were "good" – they are inside the default cell, and 6 of them were critical under the predefined tolerance, determining different numbers of neighbour cells that must be taken into account in polynomial coefficient calculations.

For the desired coordinates (5), ideal pose coordinates are calculated, in pose space, calling the ideal forward kinematics procedure. In that procedure, it is assumed ideally, no geometric errors exist. The position of the end-effector is represented with 3D vector $\mathbf{R_p}$, whose coordinates are expressed in millimeters. The orientation is represented with 2 unit 3D vectors $\mathbf{R_{o1}}$ and $\mathbf{R_{o2}}$, so their coordinates are dimensionless numbers.

$$\text{Pose}_{\text{ideal}} = [\mathbf{R}_\text{p}\ \mathbf{R}_{\text{o1}}\ \mathbf{R}_{\text{o2}}] \tag{7}$$

To analyze how the choice of the cell in skewed 7D space influences to the accuracy, 3 different strategies are applied:

- Strategy 1: Looking for the proper cell, no matter how long it lasts.
- Strategy 2: Taking the default cell, no matter whether the point is critical or not.
- Strategy 3: Taking neighbor cells into account if appropriate coordinate is near measurement point under predefined tolerance and calculating the error approximation polynomial coefficients as average of the appropriate coefficient for all that cells.

For every strategy applied, different commanded coordinates are obtained. Calling the error forward kinematics procedure, for commanded coordinates obtained by i-th strategy (i = 1, 2, 3), the real pose is calculated:

$$\text{Pose}_{\text{real},i} = [S_{p,i}\ S_{o1,i}\ S_{o2,i}]. \tag{8}$$

Accuracy estimation is made calculating the deviations from real pose to ideal pose.

Position deviation for i-th strategy (i = 1, 2, 3) is 3D vector with coordinates expressed in millimeters:

$$\text{Delta}_i = \text{R}_\text{p} - \text{S}_{\text{p,i}} \tag{9}$$

In the Table 3, norm of Delta deviation vector is given for each strategy.

Table 3

Position deviations

	Number of callings Delaunayn and tsearchn	Position deviation (mm)		
		Strategy 1	Strategy 2	Strategy 3
Pt. 1	1	$269.6\cdot10^{-6}$	$269.6\cdot10^{-6}$	$190.3\cdot10^{-6}$
Pt. 2	1	$685.2\cdot10^{-6}$	$685.2\cdot10^{-6}$	$584.1\cdot10^{-6}$
Pt. 3	1	$365.7\cdot10^{-6}$	$365.7\cdot10^{-6}$	$490.9\cdot10^{-6}$
Pt. 4	16	$447.4\cdot10^{-6}$	$684.5\cdot10^{-6}$	$177.9\cdot10^{-6}$
Pt. 5	19	$211.3\cdot10^{-6}$	$264.5\cdot10^{-6}$	$270.6\cdot10^{-6}$
Pt. 6	28	$993.8\cdot10^{-6}$	$1108\cdot10^{-6}$	$43.2\cdot10^{-6}$
Pt. 7	28	$557.9\cdot10^{-6}$	$716.2\cdot10^{-6}$	$708.2\cdot10^{-6}$
Pt. 8	275	$556.7\cdot10^{-6}$	$811.6\cdot10^{-6}$	$339.6\cdot10^{-6}$
Pt. 9	343	$203.6\cdot10^{-6}$	$242.4\cdot10^{-6}$	$44.6\cdot10^{-6}$

In the Table 4 accuracy estimation for orientation of the end-effector is made taking the average of the norms of 2 Epslion deviations:

$$\begin{aligned}\text{Epsilon1}_\text{i} &= \text{R}_{\text{o1}} - \text{S}_{\text{o1,i}} \\ \text{Epsilon2}_\text{i} &= \text{R}_{\text{o2}} - \text{S}_{\text{o2,i}}\end{aligned} \tag{10}$$

Table 4

Orientation deviations

	Number of callings Delaunayn and tsearchn	Orientation deviation		
		Strategy 1	Strategy 2	Strategy 3
Pt. 1	1	$0.35\cdot10^{-6}$	$0.35\cdot10^{-6}$	$0.82\cdot10^{-6}$
Pt. 2	1	$0.82\cdot10^{-6}$	$0.82\cdot10^{-6}$	$0.93\cdot10^{-6}$
Pt. 3	1	$0.55\cdot10^{-6}$	$0.55\cdot10^{-6}$	$0.79\cdot10^{-6}$
Pt. 4	16	$0.59\cdot10^{-6}$	$0.72\cdot10^{-6}$	$0.15\cdot10^{-6}$
Pt. 5	19	$0.18\cdot10^{-6}$	$0.26\cdot10^{-6}$	$0.22\cdot10^{-6}$
Pt. 6	28	$1.25\cdot10^{-6}$	$1.32\cdot10^{-6}$	$0.05\cdot10^{-6}$
Pt. 7	28	$0.47\cdot10^{-6}$	$0.71\cdot10^{-6}$	$0.95\cdot10^{-6}$
Pt. 8	275	$0.88\cdot10^{-6}$	$0.51\cdot10^{-6}$	$0.42\cdot10^{-6}$
Pt. 9	343	$0.23\cdot10^{-6}$	$0.35\cdot10^{-6}$	$0.02\cdot10^{-6}$

Strategy 1 is expected to be dominant in accuracy reaching, since heavy computations are made to find the proper 7D cell. In strategy 3, extension of the local space is made and more knots in skewed space are taking into account to have influence on error approximation in such point. It allows avoiding the heavy computations as Delanuay procedure is. It was expected accuracy to be worse and hopefully near to the accuracy of strategy 1. Surprisingly, strategy 3 gives the best results for 6 of sampled points, reaching the smallest position deviations and the best results for 4 of the sampled points, reaching the smallest orientation deviations.

Strategy 2 has the best computational time in all cases, since no Delanuay procedure is taken and the smallest number of knots needed to be taken into account. In the Table 5 ratios between computational times of strategy 1 and strategy 3 over strategy 2 are given. For "good" points, strategy 1 spends 8.7–11.4 times more computational time than strategy 2. For "bad" points, this ratio is between 182.9 and 3825.0. That shows that searching for proper cell is practically impossible in 7D.

Strategy 3 spends 2.3–106.2 times more computational time than strategy 2 and it depends on the number of neighbor cells taken into account. Most of computational time is spend to calculate the actual coordinates of the knots, so it could be reduced using strategy for cashing the actual coordinates of the knots.

Table 5

Computational time

	Number of callings Delaunayn and tsearchn	Ratio-computational time		Number of neighbor cells taken in Strategy 3
		Strategy 1 over Strategy 2	Strategy 3 over Strategy 2	
Pt. 1	1	11.4	2.3	2
Pt. 2	1	11.4	5.3	4
Pt. 3	1	8.7	13.7	16
Pt. 4	16	182.9	24.7	32
Pt. 5	19	196.9	7.2	8
Pt. 6	28	352.1	113.0	128
Pt. 7	28	411.2	27.5	32
Pt. 8	275	3317.6	51.0	64
Pt. 9	343	3825.0	106.2	128

CONCLUSION

The comprehensive calibration procedure for 3 DOF robotized machine is implemented and its results are verified. To extend this concept to calibration model for 7 DOF robotized machine, the most computational time consuming step is identified and ideas for avoiding it in 3D are explored, from aspect of level of losing the needed accuracy.

Calibration procedure is extended for 7 DOF machine and all functions are created, including the new kinematic model. All 61 geometric errors are taken into account. Simulation is done and 9 sample points are tested.

Finding the proper cell in skewed 7D space is extremely time consuming and should be avoided. For given 7D point with desired coordinates in machine space and predefined tolerance, default cell is found using bisection method and all neighbor cells are determined if the point is critical (strategy 3).

This way, for critical points, wider local space is used to calculate error approximation polynomial coefficients. That lead to better approximation of such coefficients and results with smaller position and orientation deviations even in comparison with the deviations obtained in strategy 1, where always proper cell is found and only the knots of that cell influence on the interior point error estimation.

This strategy allows avoiding the heavy computations to find the proper cell, without significant lose in accuracy, if any.

Its computational time is predictable and in worst case is 128 times computational time of strategy 2. It could be additionally reduced if actual coordinates of the knots are cashed, which makes this strategy conducive for real time implementtation of calibration procedure for a 7 DOF robot manipulator.

REFERENCES

[1] S. Zhu: *An automated fabric layup machine for the manufacturing of fiber reinforced polymer composite*, Graduate Theses and Dissertation – Paper 13170, Iowa State University, 2013.

[2] A. Jordaens, T. Steensels: *Formation of defects in flat laminates during automatic tape laying* (framework of a master's thesis), Faculty of Engineering Technology, Leuven, Belgium, 2015.

[3] S. Samak, I. Dimovski, V. Dukovski, M. Trompeska: Volumetric calibration for improving accuracy of AFP/ATL machines, *7th International Scientific Conference of Defensive Technologies, OTEH 2016*, unpublished.

[4] ISO 230-1:2012: *Test code for machine tools – Part 1: Geometric accuracy of machines operating under no-load or quasi-static conditions.* An International Standard, by International Standards Organization, 2012.

[5] ISO 230-2:2014: *Test code for machine tools – Part 2: Determination of accuracy and repeatability of positioning of numerically controlled axes*. An International Standard, by International Standards Organization, 2014.

[6] Technical report: *Machine tools – numerical compensation of geometric errors*, ISO/TR 16907:2015, ISO, 2015.

[7] B. W. Mooring, Z. S. Roth, M. R. Driels: *Fundamentals of Manipulator Calibration*, John Wiley & Sons Inc., 1991.

[8] J. S. Shamma, D. E. Whitney: A Method for Inverse Robot Calibration. *Journal of Dynamic Systems, Measurement, and Control* **109**. 1, 36–43 (1987).

[9] D. C. Lu, M. J. D. Hayes: Robot Calibration Using Relative Measurements, *The 14th IFToMM World Congress,* Taipei, Taiwan, October 25–30, 2015.

[10] R. M. Murray, Z. Li, S. S. Sastry: *A Mathematical Introduction to Robotic Manipulation*. CRC Press; 1994.

[11] S. Xiang, Y. Altintas: Modeling and compensation of volumetric errors for five-axis machine tools. *International Journal of Machine Tools and Manufacture,* **101**, 65–78 (2016).

[12] J. Yang, Y. Altintas: Generalized kinematics of five-axis serial machines with non-singular tool path generation. *International Journal of Machine Tools and Manufacture* **75**, 119–132 (2013).

[13] M. K. Agoston: *Computer Graphics and Geometric Modeling – Mathematics*, Springer, 2005.

[14] M. K. Agoston: *Computer Graphics and Geometric Modeling – Implementation and Algorithms*, Springer, 2005.

[15] R. A. Dwyer: Higher-dimensional Voronoi diagrams in linear expected time. *Discrete & Computational Geometry* **6**. 3, 343–367 (1991).

[16] C. B. Barber, D. P. Dobkin, H. Huhdanpaa: The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software* (TOMS) **22**, 4, 469–483 (1996).