Cartesian Coordinates Computation with Interpolation Algorithms of Differencial Numerical Analysis for Industrial Robot Motion

Liliana Marilena Matica
Faculty of Electrical Engineering and Information Technologies, University of Oradea,
Oradea, Romania
lmatica@uoradea.ro

Helga Maria Silaghi
Faculty of Electrical Engineering and Information Technologies, University of Oradea,
Oradea, Romania
hsilaghi@uoradea.ro

Zoltan Kovendi
Faculty of Electrical Engineering and Information Technologies, University of Oradea,
Oradea, Romania
zkovendi@uoradea.ro

Claudiu Costea
Faculty of Electrical Engineering and Information Technologies, University of Oradea,
Oradea, Romania
ccostea@uoradea.ro

***Abstract*—Interpolate algorithms of numeric difference analyse, named in this paper ADNIA, may be implemented in purpose to generate a complex trajectory of industrial robots tool point. Considering the position matrix of an industrial robot, the linear or the circular ADNIA, according with linear or circular trajectory, may be implemented for interpolating the position vector: $\vec{p}$. The circular algorithm may be used for interpolate the orientation versors: $\vec{n}, \vec{o}, \vec{a}$. Those are the proposed computation algorithms and will be describe in this paper. The purpose of the described method is the value of position matrix elements, about a robotic arm.**

***Index Terms*—interpolating algorithm, numeric difference analyse, sampling time period, trajectory, characteristic point, industrial robots; linear space steps; angle steps.**

I. INTRODUCTION

The management of industrial robot motion contains a lot of complex aspects. An important aspect is the waypoints Cartesian coordinate computation, during a motion on an imposed trajectory [1], [2].

During the motion, the trajectory is important for several industrial robots, as painting or welding industrial process [4], [5],[8]. There for, it is necessary a compromise, about traversing the trajectories. The compromise is accomplished by decomposing the trajectory in simple segments: circular or linear segments.

The coordinates of the segment extremities are the external information of interpolate algorithms, [6]. The coordinates of intermediary points are computed, during the trajectory traverse, [4]. In purpose to compute the coordinates of intermediary points, interpolate software module is called [10]. For industrial robots, the computation has some specific characteristics, some will be described in this paper.

II. ABOUT INDUSTRIAL ROBOTS MOTION

There are a lot of criteria concerning classification of motion, about industrial robots, [3], [7]. For industrial robots, the type of motion may be classified as follow:

A. Point to point motion

Point to point motion involve an unimportant trajectory (the speed is constant for each articulation of the industrial robot, in every movement); only the final position is important and exact defined, such point welding or manipulation of the objects.

B. Articulation interpolation

This motion type commands each articulation of the industrial robot according with inertial principles. The resulted trajectory is unknown, in the working space of the industrial robot, for example objects manipulation process.

C. Cartesian interpolation

About this type, the trajectory is defined exact, in Cartesian space, with three axles: Ox; Oy; Oz.; the trajectory points coordinates are converted in movement commands, for each articulation of the industrial robot, using the geometric model.

For example, using ACL language, an instruction MOVEL describes a movement upon a linear trajectory and an instruction MOVEC describes a movement upon a circular

trajectory (instruction MOVE describes a movement with undefined trajectory), [9].

III. THE PROPOSED LINEAR ADNIA ALGORITHM

About computation of waypoints Cartesian coordinates on a linear trajectory, [5], the linear ADNIA algorithm considers the distance to go, at every sampling period of time (named: distance to go, through algorithm definition). The axle components of this distance are named axle steps. About axle steps, its mean the linear space step that must be performed (executed motion), at every sampling period of time, on every axle; the notations are: $\delta_x, \delta_y, \delta_z$ (considering the three axles: Ox; Oy; Oz) [4].

For each axle, the axle step is a constant value, because the motion speed, named *v*, is constant (about this algorithm, the speed of linear segment traverse is constant and must be defined; it has three component: v_x, v_y, v_z.

Let consider $P_O(x_O, y_O, z_O)$ the initial point of the linear segment. The Cartesian coordinates of every waypoint (of every intermediary point), named: $P_l(x_l, y_l, z_l)$ are:

$$\begin{aligned} x_l &= x_O + l \cdot \delta_x \\ y_l &= y_O + l \cdot \delta_y \\ z_l &= z_O + l \cdot \delta_z \end{aligned} \tag{1}$$

The linear ADNIA computations may be explained by those relations:

$$x_l = x_O + \sum_{m=1}^{l}\left(\frac{(x_F - x_O)\cdot \Delta t}{t_P}\right)_m = x_O + \sum_{m=1}^{l}\left(\frac{x_F - x_O}{N}\right)_m =$$

$$= x_O + \sum_{m=1}^{l}(\delta_x)_n = x_O + l \cdot \delta_x = x_O + l \cdot v_x \cdot \Delta t \tag{2}$$

$$y_l = y_O + \sum_{m=1}^{l}\left(\frac{(y_F - y_O)\cdot \Delta t}{t_P}\right)_m = y_O + \sum_{m=1}^{l}\left(\frac{y_F - y_O}{N}\right)_m =$$

$$= y_O + \sum_{m=1}^{l}(\delta_y)_m = y_O + l \cdot \delta_y = y_O + l \cdot v_y \cdot \Delta t \tag{3}$$

$$z_l = z_O + \sum_{m=1}^{l}\left(\frac{(z_F - z_O)\cdot \Delta t}{t_P}\right)_m = z_O + \sum_{m=1}^{l}\left(\frac{z_F - z_O}{N}\right)_m =$$

$$= z_O + \sum_{m=1}^{l}(\delta_z)_m = z_O + l \cdot \delta_z = z_O + l \cdot v_z \cdot \Delta t \tag{4}$$

In the previous relations, the notation means: $P_F(x_F, y_F, z_F)$ is the final point; t_p is time value that is necessary for traverse the entire linear segment: $P_O P_F$; *l* is the index for the intermediary points: $P_l(x_l, y_l, z_l)$; Δt is the value of sampling period of time, (a numeric system, that command industrial process, works with a constant value of sampling time period); v_x, v_y, v_z are the speed components values upon the three axles; *N* is the number of sampling periods of time, necessary for traverse the linear segment.

Considering previous relations and notations, the theoretical explanation of linear ADNIA algorithm is based on the definition of the distance constant values that must be executed (travelled), at every sampling period of time, on every axle, named axle steps: $\delta_x = \frac{x_F - x_O}{N}$; $\delta_y = \frac{y_F - y_O}{N}$; $\delta_z = \frac{z_F - z_O}{N}$.

The algorithm begins with the computation of the linear segment length, named *L*; knowing the coordinates of initial and final points. Then, the algorithm determinates the number of sampling periods of time, necessarily for traverse the linear segment:

$$N = round\left(\frac{t_p}{\Delta t}\right) = round\left(\frac{L}{v \cdot \Delta t}\right)$$

$$= round\left(\frac{\sqrt{(x_F - x_O)^2 + (y_F - y_O)^2 + (z_F - z_O)^2}}{v \cdot \Delta t}\right) \tag{5}$$

The *N* value must be an integer value, the rounding process find the next integer value of the number computed with (4).

Then, the algorithm determines the values of linear space steps: $\delta_x, \delta_y, \delta_z$. With those values, the interpolating process run and computes the intermediary points coordinates values, considering relations 1.

For example, considering the Cartesian coordinates of initial point: $P_O(x_O, y_O, z_O) = (1;2;3)$ [mm] and the final point: $P_F(x_F, y_F, z_F) = (2;4\sqrt{2};7)$ [mm]; considering the speed $v = 2.5 \cdot 10^3 mm/s$ and sampling time period $\Delta t = 0.2 \cdot 10^{-3} s$; it results the number of sampling periods of time (necessary for traversing this linear segment), it is:

$$\mathrm{N} = \frac{\sqrt{(2-1)^2 + (4\sqrt{2}-2)^2 + (7-3)^2}}{2.5 \cdot 10^3 \cdot 0.2 \cdot 10^{-3}} = 10 \tag{6}$$

On every axle, the space steps that must be performed (executed, during the motion), on every sampling period of time (named axle step), are (mm):

$$\delta_x = \frac{x_F - x_O}{N} = 0.1$$

$$\delta_y = \frac{y_F - y_O}{N} \cong 0.36$$

$$\delta_z = \frac{z_F - z_O}{N} = 0.4 \quad (7)$$

The values of intermediary points coordinates are computed, considering those linear space steps, see table 1.

About industrial robots, the orientation of robotic arm is characterized by versors: $\vec{n}, \vec{o}, \vec{a}$, $|\vec{n}| = 1, |\vec{o}| = 1, |\vec{a}| = 1$ and the position, by position vector: $\vec{p}$.

The position matrix of an industrial robot, named G, has the components upon the three axles: Ox; Oy; Oz, of orientation versors and position vector, described by (7).

TABLE I. VALUES OF CARTESIAN COORDINATES

Step No. (index)	**Cartesian coordinate of intermediary points**		
	***X**I*	***Y**I*	***Z**I*
0 (start ADNIA)	1	2	3
1	1.1	2.36	3.4
2	1.2	2.72	3.8
3	1.3	3.08	4.2
4	1.4	3.44	4.6
5	1.5	3.80	5.0
6	1.6	4.16	5.4
7	1.7	4.52	5.8
8	1.8	4.88	6.2
9	1.9	5.24	6.6
10 (stop ADNIA)	2	5.6 $\cong 4\sqrt{2}$	7

$$G = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (8)$$

Considering the orientation of the industrial robot with the versor $\vec{n}$ along the axle Ox, the versor $\vec{o}$ along the axle Oy and the versor $\vec{a}$ along the axle Oz, the position matrix, index l, named: G_l, has those values in the fourth column:

$$G_l = \begin{bmatrix} 1 & 0 & 0 & x_l \\ 0 & 1 & 0 & y_l \\ 0 & 0 & 1 & z_l \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (9)$$

The position matrix index l, rel.8, define the intermediary point: $P(x_l, y_l, z_l)$, index *l*, the point that be reached after *l* steps (linear space steps).

For example, on the seventh step, the position matrix is (the seventh step values are shown in table 1):

$$G_7 = \begin{bmatrix} 1 & 0 & 0 & 1.7 \\ 0 & 1 & 0 & 4.52 \\ 0 & 0 & 1 & 5.8 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (10)$$

If the orientation of the industrial robot is not according with the described considerations, the orientation versors components may be computed using the circular ADNIA (as it will be described in the next paragraph). It results others components values about first, second and third column of position matrix.

Also, the circular ADNIA is useful for generate the circular segments of trajectories.

IV. THE PROPOSED CIRCULAR ADNIA ALGORITHM

In purpose of intermediary coordinates points computation, the spherical coordinates, named (R, φ, ϕ) are indicated, figure 1; the ADNIA algorithm is applied about polar angle: φ and azimuthal angle: ϕ.

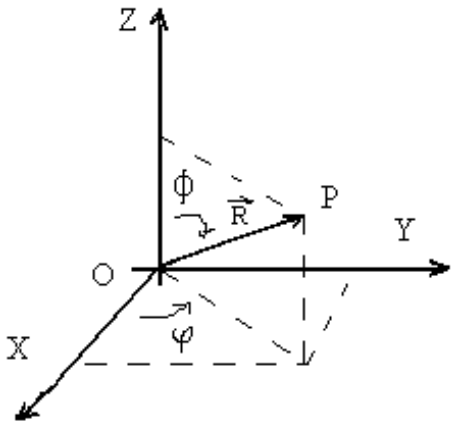

Figure 1. About spherical coordinates.

The circular ADNIA works with angle steps that must be performed, on every sampling time period, named: $\delta_\varphi, \delta_\phi$.

$$\delta_\varphi = \frac{\varphi_F - \varphi_O}{N}$$
$$\delta_\phi = \frac{\phi_F - \phi_O}{N} \quad (11)$$

Considering the polar angle (indexed *l)* : φ_l and azimuthally angle (indexed *l)* : ϕ_l, the Cartesian coordinates of intermediary points, index *l* , (the point that must be reach after *l* angle steps) are computed with next formulas [4] (the notations have the similar meanings as previously described):

$$\varphi_l = \varphi_O + \sum_{m=1}^{l}\left(\frac{(\varphi_F - \varphi_O)\cdot\Delta t}{t_P}\right)_m = \varphi_O + \sum_{m=1}^{l}\left(\frac{\varphi_F - \varphi_O}{N}\right)_m =$$
$$= \varphi_O + \sum_{m=1}^{l}\left(\delta_\varphi\right)_m = \varphi_O + l\cdot\delta_\varphi \quad (12)$$

$$\phi_l = \phi_O + \sum_{m=1}^{l}\left(\frac{(\phi_F - \phi_O)\cdot\Delta t}{t_P}\right)_m = \phi_O + \sum_{m=1}^{li}\left(\frac{\phi_F - \phi_O}{N}\right)_m =$$
$$= \phi_O + \sum_{m=1}^{l}\left(\delta_\phi\right)_k = \phi_O + l\cdot\delta_\phi \quad (13)$$

The number of angle steps, named N, is computed considering the length of the circular trajectory, named *L*, the tangential speed, named *v*, and the sampling period of time, the rounding considerations are similar with (5):

$$\mathrm{N} = \mathrm{round}\left(\frac{\mathrm{L}}{\mathrm{v}\cdot\Delta\mathrm{t}}\right) \quad (14)$$

For example, considering: $N = 10$; the spherical coordinates of initial point (1; 0^0; 90^0) and final point (1; 90^0; 45^0) , it results $\varphi_O = 0^0, \phi_O = 90^0$; $\varphi_F = 90^0, \phi_F = 45^0$; $R = 1$.

The angle steps that must be performed, on every sampling time period, are:

$$\delta_\varphi = \frac{90^0 - 0^0}{10} = 9^0$$
$$\delta_\phi = \frac{45^0 - 90^0}{10} = -4^0 30' \quad (15)$$

The Cartesian coordinates of intermediary point $P_l(x_l, y_l, z_l)$ may be computed with next formulas:

$$\mathrm{x_1} = \mathrm{R}\cdot\sin(\phi_0 + 1\cdot\delta_\phi)\cdot\cos(\varphi_0 + 1\cdot\delta_\varphi) =$$
$$= \sin\left[90^0 - 1\cdot(4^0 30')\right]\cdot\cos(0^0 + 1\cdot 9^0) \quad (16)$$

$$\mathrm{y_1} = \mathrm{R}\cdot\sin(\phi_0 + \cdot\delta_\phi)\cdot\sin(\varphi_0 + 1\cdot\delta_\varphi) =$$
$$= \sin\left[90^0 - 1\cdot(4^0 30')\right]\cdot\sin(0^0 + 1\cdot 9^0) \quad (17)$$

$$\mathrm{z_1} = \mathrm{R}\cdot\cos(\phi_0 + 1\cdot\delta_\phi) =$$
$$= \cos(90^0 - 1\cdot(4^0 30')) \quad (18)$$

This example may represent the interpolating process for the versor $\vec{a}$ and may compute the polar angle for $\vec{a}$, named: $\varphi_{l,a}$ and azimuthally angle, named: $\phi_{l,a}$.

Considering the versor characteristic: $|\vec{a}| = 1$; it results the value of radium, $R = 1$.

The versor components, fig.2, are:

$$\mathrm{a_{l,x}} = \sin\left[90^0 - 1\cdot(4^0 30')\right]\cdot\cos(0^0 + 1\cdot 9^0) \quad (19)$$

$$\mathrm{a_{l,y}} = \sin\left[90^0 - 1\cdot(4^0 30')\right]\cdot\sin(0^0 + 1\cdot 9^0) \quad (20)$$

$$\mathrm{a_{l,z}} = \cos(90^0 - 1\cdot(4^0 30')) \quad (21)$$

The versors: $\vec{i}, \vec{j}, \vec{k}$ are those that define the three axles: Ox, Oy and Oz, fig.2.

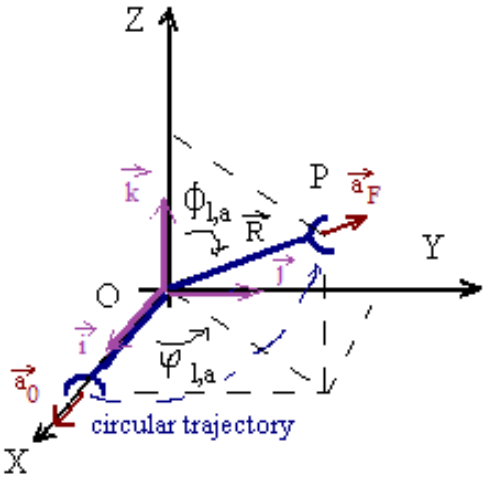

Figure 2. Circular ADNIA about vector $\vec{a}$.

It result the versor $\vec{a}_l$, on every *l* sampling time period (index l):

$$\vec{\mathrm{a}}_1 = \mathrm{a_{l,x}}\cdot\vec{\mathrm{i}} + \mathrm{a_{l,y}}\cdot\vec{\mathrm{j}} + \mathrm{a_{l,z}}\cdot\vec{\mathrm{k}} \quad (22)$$

Similar, the algorithm may be applied about interpolate the versor $\vec{o}_l$, (23) and the versor $\vec{n}_l$ (24):

$$\vec{o}_l = o_{l,x} \cdot \vec{i} + o_{l,y} \cdot \vec{j} + o_{l,z} \cdot \vec{k} \quad (23)$$

$$\vec{n}_l = n_{l,x} \cdot \vec{i} + n_{l,y} \cdot \vec{j} + n_{l,z} \cdot \vec{k} \quad (24)$$

The acceleration and the deceleration of the movement are not included in those computations; it may be another part of the computations. The described algorithms begin after acceleration stage and stop before deceleration stage of motion.

The dynamic of industrial robots may be considered a performant one, the acceleration and deceleration process is a fast one and it may be done in few steps (for example, one or two steps about acceleration). In this conditions, the number of steps, N, must be adapted of those considerations (the lengths L is shorter, the difference is the linear distance necessary for acceleration and deceleration).

After those computations, the parameters of industrial robot articulations may be computed, considering the cinematics model of the robot, (starting with components values of position matrix).

V. Conclusions

During the motion upon linear or circular trajectories, the numeric difference analyse interpolate algorithms, named (in this paper) ADNIA, may be applied for compute intermediary points Cartesian coordinates of those trajectories.

The ADNIA formulas compute (using a system with numeric microprocessor or microcontroller) the linear space steps or angle steps that must be performed, every sampling period of time, during the motion.

The motion speed must be a constant value. This value and the coordinates of initial and final points of the trajectories are the necessaries data (input data), for those algorithms.

Those algorithms were implemented in the software of an automated welding cell and the computation precision was satisfying. Similar algorithms have o lot of implementations, concerning milling or turning process.

The ADNIA algorithms, described in this paper, may be applied about tool point movements of industrial robots, on, linear or circular trajectories. The defined trajectory imposes the components values of position vector: $\vec{p}$; its components upon the three axles are identical with intermediary points Cartesian coordinates, computed by similar algorithms.

The modification of the industrial robot orientation, during the trajectory traverse, may involve those algorithms; the components values of orientation versors may be computed according with circular ADNIA. About computation of those versors components, the number of angle steps is equal with steps number computed for trajectory traverse (it must not be computed); the computation about steps number involve the trajectory, (5) or (14).

The link between ADNIA algorithms and position matrix of a robotic arm is described in this paper. Frequently, those algorithms were implemented about metal cutting industrial process, on CNC machines. Those algorithms were not been described linked with robotic arms movements.

The originality of described method consists in the definition of the computation purpose. The purpose of the described method is the value of position matrix elements, about a robotic arm. Usually, the purpose of ADNIA computation is the value of waypoints Cartesian coordinates, upon an imposed trajectory.

References

[1] E. Ciupan, F. Lungu, C. Ciupan, "ANN Method for Control of Robots to Avoid Obstacles", International Journal of Computers, Communications & Control, Vol. 9, Nr. 5, 2014, ISSN 1841-9836, pp. 165-173.

[2] S. Dale, H. Silaghi, D. Zmaranda, U. Rohde, "Intelligent Design Environment for Second-Order Positioning Systems", International Journal of Computers, Communications & Control, Vol. 10, Nr. 2, April, 2015, ISSN 1841-9836, pp. 165-173.

[3] Y. Guan, K. Yokoi, and A, Kheddar, "On robotic trajectory planning using polynomial interpolations". Proceeding of the IEEE International Conference on Robotics and Biomimetics, 2005. 111-116.

[4] L.M. Matica, Conducerea robotilor industriali, Editura Universitatii din Oradea, 2008, ISBN 978-973-759-481-5.

[5] L.M. Matica, "About Adapting Traversing Trajectories ADN Interpolating Algorithms for Industrial Robots", Journal of Computer Science & Control Systems; 2009, Vol. 2, Issue 2, pp.107-110.

[6] J. M. Rosário, C. de Oliveira, C.E.A. de Sá; C.R.E. Lima, "Proposal of Methodology for the Modeling and Control Systems", Journal of the Brazilian Society of Merchanical Sciences, Vol.24, No.3, 2002

[7] B. N. Saeed, Introduction to Robotics Analysis, System, Application. Prentice Hall 2001, 29 – 172.

[8] B. Siciliano, O. Khatib, Springer Handbook of Robotics, Springer Press, 2008.

[9] G.X. Zhang, J.X. Cheng, M. Gheorghe, "Dynamic behavior analysis of membraneinspired evolutionary algorithms", International Journal of Computers Communications & Control, ISSN 1841-9836, 9(2), 2014, pp. 227-242.

[10] D. Zmaranda, Helga Silaghi, G. Gabor, C. Vancea, "Issues on Applying Knowledge-Based Techniques in Real-Time Control Sysrems" International Journal of Computers Communications & Control, ISSN 1841-9836, Vol 8, No 1, 2013: INT J COMPUT COMMUN, F.I.0,694, pp. 166-175.