

Real Time Computation for Robotic Arm Motion upon a Linear or Circular Trajectory

Liliana Marilena Matica, Cornelia Györödi, Helga Maria Silaghi, Simona Veronica Abrudan Cacioara
Faculty of Electrical Engineering and Information Technology, University of Oradea
Oradea, Romania

Abstract—The computation method proposed in this paper, named ADNIA (Analysis Differential Numeric Interpolate Algorithms), computes waypoints Cartesian coordinates for TCP (tool centre point) of a robotic arm, for a motion on an linear or circular imposed trajectories. At every sampling period of time, considering real-time software implementation of ADNIA, the location matrix of a robotic arm is computed. This computation method works with a well-defined value of motion speed; it results a maximum computation precision (for those motions).

Keywords—Waypoints; location matrix; position vector; orientation of a robotic arm; orientation versors; Analysis Difference Numeric Interpolate Algorithm (ADNIA); linear or circular ADNIA

I. INTRODUCTION

The paper describes the real-time computation method for robotic arm motions, upon imposed trajectories, named ADNIA (Analysis Differential Numeric Interpolate Algorithms).

Some considerations regarding the importance of industrial robots, in manufacturing process, are described in this first paragraph.

Industrial production (manufacturing) is a complex process, the decisions result of combination and use of involved factors. About manufacturing process, the important factors are: environmental and natural resources, scientific and technological resources, quality of management. The multitudes of factors that determine industrial production are identified by the so called socio-political environment generously given. About manufacturing, an important method of qualitative growth is manufacturing automation. Automation, not only refers to the development of the means of production, but also to its integrative aspects, in relation of all system levels of industrial structure, to the human factor.

Many automation facilities have as purpose the replacement of human activity, in the manufacturing process; the mainly functions of them is to help about the automatization of the machining tasks, concerning the technologic processes; tasks requiring the intervention of the hand human operator, under the supervision of the eye (the whole action is coordinated by the human brain). Possibly, such operations are: start-stop equipment, loading and unloading, assembly-disassembly work items (pieces, parts), change or handling tools, testing, processing, inspection, comparison, repair or maintenance.

Regarding industrial production (manufacturing) systems, it is needful to emphasize the means of work; one particularly powerful is: industrial robots [1]-[8]. About industrial production efficiencies and necessarily required organizational activities: leadership, management, correlation in time and space, the industrial robots are very useful, in order to achieve objectives. So, the concept of interest allocation (as a function of effective industrial production plan and management) is implemented by industrial robots, in the manufacturing process. Among the important objectives of manufacturing (industrial production) systems, (regarding the concept of allocation), it is the growth adaptability of those systems to the changing social demand or to the environment, while reducing resource consumption. An important role, particularly in this respect, has the industrial robots [1], [5].

In purpose to define a location and command a motion of an industrial robot, more exactly, of a robotic arm (a specific industrial robot that is similar with human arm), it must be defined the location matrix [2], [4]:

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

The location matrix contains the axes components of orientation versors: $\vec{n}; \vec{o}; \vec{a}$, (three vectors with module value equal with 1 value, $|\vec{n}|=1, |\vec{o}|=1, |\vec{a}|=1$) and the position vector: \vec{p} , Fig. 1, (definition: a versor is a vector having module value equal with 1 value).

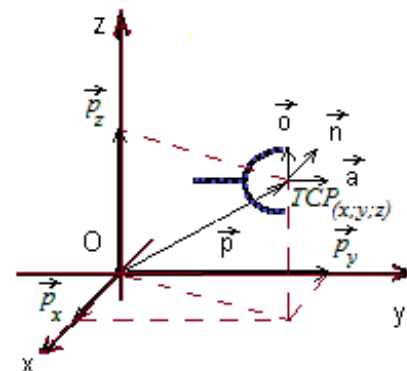


Fig. 1. Orientation versors and position vector.

The motion of a robotic arm may be imposed, upon a linear or a circular trajectory; an imposed trajectory of motion is necessary during an industrial process executed by the robotic arm, as painting or welding industrial process. The imposed trajectory for a robotic arm may be defined by computation of every location matrix for every waypoint, upon the trajectory. This computation, working with ADNIA algorithms, will be described in this paper. Those algorithms were implemented about metal cutting industrial process, on CNC machines. [7] Those algorithms were not been described about robotic arms motion; the paper describes the adaptation of this algorithm for robotic arm motion.

The paper is structured as follows: Section 2 presents the computation method about robotic arm motion, on a linear or circular imposed trajectory and Section 3 describes the computations about a variable orientation of a robotic arm, during the motion on a linear or circular imposed trajectory. Based on the results of the tests performed, several conclusions are presented in the last section.

II. ADNIA FOR LINEAR OR CIRCULAR TRAJECTORY OF ROBOTIC ARMS

This paragraph describes the computation method about robotic arm motion, on a linear or circular imposed trajectory; the robotic arm has a constant orientation, during the motion.

About a robotic arm, the motion trajectory of TCP (*Tool Centre Point*) is defined by variation of \vec{p} vector, Fig. 1. The axle components of \vec{p} vector: $\vec{p}_x; \vec{p}_y; \vec{p}_z$ defines the Cartesian coordinates of TCP. ADNIA algorithm computes the (indexed k) Cartesian coordinates of waypoints, $P_k(x_k; y_k; z_k)$ upon the linear trajectory. The input data of linear ADNIA algorithm are: Cartesian coordinates of first point (index 0) and end point (index F) of the linear trajectory and the motion speed: v .

The ADNIA computation starts with the number of sampling periods of time, necessarily for traverse the linear trajectory, named N [2], [4] (the value of sampling periods of time is Δt):

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

The N value must be an integer value; about (2), the rounding computation find the next integer value of the value computed (for example, if the result value is 22.1 the round value must be 23).

Then, the linear ADNIA computes the values of linear space steps, necessary to be performed at every sampling period of time, for each axle (executed motion at every sampling period of time), named axle steps: $\delta_x, \delta_y, \delta_z$. The interpolating process of the linear trajectory run and computes the waypoints, $P_k(x_k; y_k; z_k)$, considering [2]:

$$x_k = x_O + k \cdot \delta_x$$

$$y_k = y_O + k \cdot \delta_y \quad (3)$$

$$z_k = z_O + k \cdot \delta_z$$

Let consider the interpolation process of a linear trajectory; defined by Table I:

TABLE I. VALUES OF CARTESIAN COORDINATES

Step No. (index)	Cartesian coordinate of intermediary points		
	X_l	Y_l	Z_l
0 (start ADNIA)	10	20	30
1	10.1	22	30.03
2	10.2	24	30.06
3	10.3	26	30.09
4	10.4	28	30.12
5	10.5	30	30.15
6	10.6	32	30.18
7	10.7	34	30.21
8	10.8	36	30.24
9	10.9	38	30.27
10 (stop ADNIA)	11	40	30,3

About this interpolation process, the axle steps have the values: $\delta_x = 0.1$; $\delta_y = 2$; $\delta_z = 0.03$; and the number of necessarily steps for traverse the linear is: $N = 10$, (for every axle).

Let consider this constant orientation of the robotic arm, during the motion on the linear trajectory: $\vec{n}_x = 1$; $\vec{o}_y = 1$; $\vec{a}_z = 1$; the location matrix about a waypoints upon the trajectory, named G_k , have the axle components values of position vector as listed in Table I. For example, on the seventh step upon the linear trajectory, the location matrix is:

$$G_k = G_7 = \begin{bmatrix} 1 & 0 & 0 & x_{k=7} \\ 0 & 1 & 0 & y_{k=7} \\ 0 & 0 & 1 & z_{k=7} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 10.7 \\ 0 & 1 & 0 & 34 \\ 0 & 0 & 1 & 30.21 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4)$$

The big advantage of ADNIA algorithms is the constant value of motion speed.

Next considerations explain the situation of a variable orientation of robotic arm, (the ADNIA algorithms, explained in this paper, work about a linear or circular trajectory, with a constant or variable orientation of the robotic arm).

The linear ADNIA (previously explained), works with Cartesian coordinates. The circular ADNIA works with spherical coordinates: (R, φ, ϕ) , Fig. 2 [7].

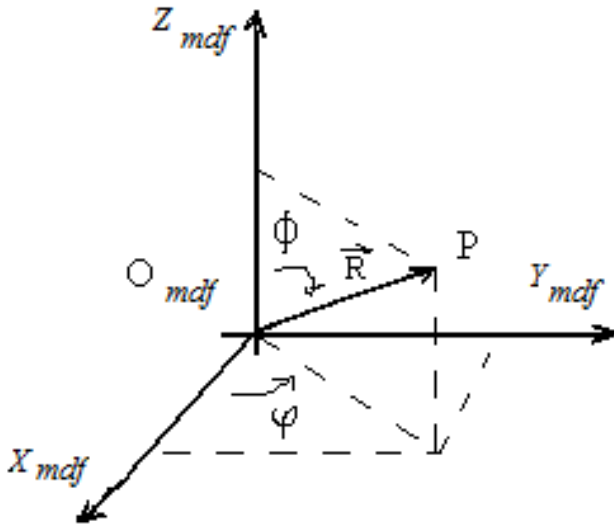


Fig. 2. The spherical coordinates.

In Fig. 2, the Cartesian coordinates system, named $OXYZ_{mdf}$; is properly chosen, in purpose to have a constant value of R , for spherical coordinates of waypoints upon the circular trajectory.

Every waypoint upon the circular trajectory is defined by its spherical coordinates: $P_m(R, \varphi_m, \phi_m)$. Considering the polar angle (indexed m): φ_m and azimuthally angle (indexed m): ϕ_m , the Cartesian coordinates of waypoints, indexed m , (the point that must be reach after m angle steps) are computed with relations [2]:

$$\varphi_m = \varphi_O + m \cdot \delta_\varphi = \varphi_O + m \cdot \left(\frac{\varphi_F - \varphi_O}{N} \right) \quad (5)$$

$$\phi_m = \phi_O + m \cdot \delta_\phi = \phi_O + m \cdot \left(\frac{\phi_F - \phi_O}{N} \right) \quad (6)$$

The notations have the similar meanings as previously explained: index O point upon the first point of the circular trajectory, $P_O(R, \varphi_O, \phi_O)$; index F point upon the last point of the circular trajectory, $P_F(R, \varphi_F, \phi_F)$; N is the number of angle steps, necessary for traverse the circular trajectory, its value may be computed from the speed value of motion. For example, considering: $N=15$; $\varphi_O=0^\circ, \phi_O=92^\circ$; $\varphi_F=90^\circ, \phi_F=47^\circ$; $R=22$, so the spherical coordinates of initial and final point are: $P_O(22; 0^\circ; 92^\circ); P_F(22; 90^\circ; 47^\circ)$; it results the value of angle steps that must be performed, on every sampling time period:

$$\delta_\varphi = \frac{90^\circ - 0^\circ}{15} = 6^\circ \quad (7)$$

$$\delta_\phi = \frac{47^\circ - 92^\circ}{15} = -3^\circ$$

The Cartesian coordinates of waypoints: $P_m(x_m, y_m, z_m)$, upon the circular trajectory, may be computed:

$$\begin{aligned} x_m &= R \cdot \sin(\phi_0 + m \cdot \delta_\phi) \cdot \cos(\varphi_0 + m \cdot \delta_\varphi) = \\ &= 22 \cdot \sin(90^\circ - m \cdot 3^\circ) \cdot \cos(0^\circ + m \cdot 6^\circ) \\ y_m &= R \cdot \sin(\phi_0 + m \cdot \delta_\phi) \cdot \sin(\varphi_0 + m \cdot \delta_\varphi) = \\ &= 22 \cdot \sin(90^\circ - m \cdot 3^\circ) \cdot \sin(0^\circ + m \cdot 6^\circ) \\ z_m &= R \cdot \cos(\phi_0 + m \cdot \delta_\phi) = 22 \cdot \cos(90^\circ - m \cdot 3^\circ) \end{aligned} \quad (8)$$

For example, after three steps, the Cartesian coordinates of waypoint $P_3(x_3, y_3, z_3)$ are:

$$\begin{aligned} x_3 &= 22 \cdot \sin(90^\circ - 3 \cdot 3^\circ) \cdot \cos(0^\circ + 3 \cdot 6^\circ) \\ y_3 &= 22 \cdot \sin(81^\circ) \cdot \sin(18^\circ) \\ z_3 &= 22 \cdot \cos(81^\circ) \end{aligned} \quad (9)$$

Because $90^\circ - 3 \cdot 3^\circ = 81^\circ; 0^\circ + 3 \cdot 6^\circ = 18^\circ$. About robotic arm, after three steps upon the circular trajectory, the location matrix is (considering the defined constant orientation of the robotic arm):

$$G_3 = \begin{bmatrix} 1 & 0 & 0 & 22 \cdot \sin(81^\circ) \cdot \cos(18^\circ) \\ 0 & 1 & 0 & 22 \cdot \sin(81^\circ) \cdot \sin(18^\circ) \\ 0 & 0 & 1 & 22 \cdot \cos(81^\circ) \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (10)$$

The computation method was implemented about motion command for a welding process, in a mining equipment manufacturing company. Computation precision was: $1 \cdot 10^{-5}$ and imposed precision was: $1 \cdot 10^{-2}$. The sampling period of time was 18 ms ($18 \cdot 10^{-3}$ seconds). Welding execution precision was the one imposed by the beneficiary.

The execution precision is limited by the motion precision of the mechanical part of the equipment. Another limitation of the motion command process was the chosen place for mounting the measuring transducer, it was on axis of electrical engines (it was not on the TCP); possible malfunctions of the motion transmission chain can't be detected.

The software implementation worked with Assembly language for Intel microprocessor. The challenge for the future is to implement the same computation with Assembly language for Intel microcontroller.

III. ADNIA FOR VARIABLE ORIENTATION OF ROBOTIC ARM DURING THE MOTION

This paragraph describes the computations about a variable orientation of a robotic arm, during the motion on a linear or circular imposed trajectory.

Let considers the versors: $\vec{i}, \vec{j}, \vec{k}$; those versors define the three axes: OX, OY and OZ, Fig. 3:

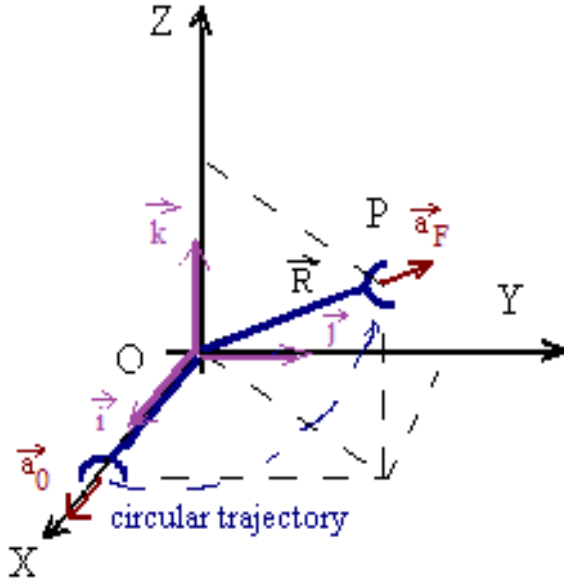


Fig. 3. Circular ADNIA about versor \vec{a} .

Considering those versors, the orientation versor \vec{a}_m , on every m sampling time period (indexed m), from the first position (indexed 0) to the last position (indexed F) is:

$$\vec{a}_m = a_{m,x} \cdot \vec{i} + a_{m,y} \cdot \vec{j} + a_{m,z} \cdot \vec{k} \quad (11)$$

Let considers the previous variation of polar and azimuthal angle, from example described about circular trajectory interpolation; after 3 steps, the axle components of versor \vec{a} computed with ADNIA algorithm are (in (9), the radius R have the value equal with 1 value):

$$\begin{aligned} a_{3;x} &= 1 \cdot \sin(81^\circ) \cdot \cos(18^\circ) \\ a_{3;y} &= 1 \cdot \sin(81^\circ) \cdot \sin(18^\circ) \\ a_{3;z} &= 1 \cdot \cos(81^\circ) \end{aligned} \quad (12)$$

Let considers a linear trajectory for robotic arm motion as described in Table I (Step No. 3) and the interpolation process of versor \vec{a} as described before; it may compute the location matrix, after three steps of motion process, with next relation:

$$G_3 = \begin{bmatrix} 1 & 0 & 1 \cdot \sin(81^\circ) \cdot \cos(18^\circ) & 10.3 \\ 0 & 1 & 1 \cdot \sin(81^\circ) \cdot \sin(18^\circ) & 26 \\ 0 & 0 & 1 \cdot \cos(81^\circ) & 30.09 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (13)$$

The acceleration and the deceleration of the movement are not included in those computations; it may be another part of the computations [4].

The described algorithm (about \vec{a}_m) may be applied for interpolate the versor \vec{o}_m , (14) and the versor \vec{n}_m (15):

$$\vec{o}_m = o_{m,x} \cdot \vec{i} + o_{m,y} \cdot \vec{j} + o_{m,z} \cdot \vec{k} \quad (14)$$

$$\vec{n}_m = n_{m,x} \cdot \vec{i} + n_{m,y} \cdot \vec{j} + n_{m,z} \cdot \vec{k} \quad (15)$$

IV. CONCLUSIONS

The ADNIA algorithms may be successfully applied for computing the location matrix for every waypoint upon an imposed linear or circular trajectory, in the case of the robotic arm motion. The imposed trajectory enforces the axle components values of position vector: \vec{p} ; its components upon the three axes must be computed with (3) for a linear trajectory and with (7) for a circular trajectory; the motion speed is imposed and it has a constant value.

At every sampling period of time, during the real-time running of ADNIA algorithm, the Cartesian coordinates of another waypoint is computed, thus it results the maximum computation precision, (for motion command process, upon a linear or circular imposed trajectory).

ADNIA algorithms work with an imposed constant value for motion speed, upon the trajectory, this being an advantage.

During the trajectory traverse with a variable orientation of robotic arm, about computation of orientation versors (vector with module equal to 1), the number of angle steps is equal with steps number computed for trajectory traverse (it is imposed by interpolation process of the trajectory). The axle components of orientation versors must be computed according with circular ADNIA.

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, Available: http://univagora.ro/jour/index.php/ijccc/article/view/813/pdf_141, accessed January 2018.
- [2] L.M. Matica, "About Adapting Traversing Trajectories ADN Interpolating Algorithms for Industrial Robots", Journal of Computer Science & Control Systems; 2009, Vol. 2, Issue 2, Available: https://www.researchgate.net/publication/40422467_About_adapting_traversing_trajectories_ADN_interpolating_algorithms_for_industrial_robots, accessed January 2018.

- [3] Z. Kovendi, I.C. Rada, L. Magdoiu, A. Corha, C. Bondici, "Checking Algorithms on Differential Equations with Known Analytical Solution" *Journal of Computer Science & Control Systems*, 2015/5/1, Available: <https://scholar.google.ro/citations?user=8ouqKx0AAAAAJ&hl=ro>, accessed January 2018.
- [4] L.M.Matica, H. Oros, "Speed Computation for Industrial Robots Motion Followed by Accurate Positioning. *International Journal of Computers, Communications & Control*, February 2017, <http://univagora.ro/jour/index.php/ijccc/article/download/2785/1061> , accessed January 2018.
- [5] T. Kunz, M. Stilman, "Optimal Trajectory Generation for Path with Bounded Acceleration and Velocity", *Robotics: Science and Systems 8th Conference*, 2012, Sydney, ISBN 978-0-262-51968-7, Available: https://books.google.ro/books?hl=en&lr=&id=NOxwAAQBAJ&oi=fnd&pg=PA209&dq=kunz+time-Optimal&ots=dMImeillUZ&sig=0N0K2_OJa6H17-4jqwCMrQOz8dk&redir_esc=y#v=onepage&q=kunz%20time-Optimal&f=false accessed February 2018.
- [6] B. J. Evans, C. D. Cook, "Evaluation of Real-Time Robot Control Systems", *Conference proceedings: Field and Service Robotics*, 1998, Springer, London, Available: https://link.springer.com/chapter/10.1007/978-1-4471-1273-0_57 accessed February 2018.
- [7] L.M. Matica, *Conducerea robotilor industriali*. Editura Universitatii din Oradea, ISBN 978-973-759-481-5, 2008
- [8] K. Krishnaswamy, J. Sleeman, T. Oates, "Real-Time Path Planning for Robotic Arm", *Proceedings of the 4th International Conference on Pervasive Technologies Related to Assistive Environments* , May 2011, University of Maryland, Baltimore, Available: https://ebiquity.umbc.edu/_file_directory_/papers/575.pdf accessed February 2018.