Inverse Kinematics and Trajectory Planning Analysis of a Robotic Manipulator

Lucas B. de Souza[1], Jônatas F. Dalmedico[1], Henrique S. Kondo[2], Márcio Mendonça[2], Marcio A. F. Montezuma[3], Katarzyna Poczęta[4]

[1]Mechanical Engineering Graduate Program, Federal University of Technology – Paraná, Cornélio Procópio, Brazil
Emails: lucasbotoni@hotmail.com, jdalmedico@alunos.utfpr.edu.br
[2]Electrical Engineering Department, Federal University of Technology – Paraná, Cornélio Procópio, Brazil
Emails: henriquekondo@alunos.utfpr.edu.br, mendonca@utfpr.edu.br
[3]Mechanical Engineering Department, Federal University of Technology – Paraná, Cornélio Procópio, Brazil
Email: montezuma@utfpr.edu.br
[4]Department of Information Systems, Kielce University of Technology, Kielce, Poland
Email: k.piotrowska@tu.kielce.pl

Abstract—*In this work, we pretended to show and compare three methodologies used to solve the inverse kinematics of a 3 DOF robotic manipulator. The approaches are the algebraic method through Matlab® solve function, Genetic Algorithms (GAs), Artificial Neural Networks (ANNs). Another aspect considered is the trajectory planning of the manipulator, which allows the user to control the desired movement in the joint space. We compare polynomials of third, fourth and fifth orders for the solution of the chosen coordinates. The results show that the ANN method presented best results due to its configuration to show only feasible joint values, as also do the GA. In the trajectory planning the analysis lead to the fifth-order polynomial, which showed the smoothest solution.*

Keywords—Robotic Manipulator, Genetic Algorithms, Artificial Neural Networks, Trajectory Planning, Comparative Analysis.

I. INTRODUCTION

In the past years, we can found several approaches to the robotics' research field due to its wide-range applications, such as in industrial production, space exploration and medical surgery (Zou, Hou, Fu, & Tan, 2006).Thus, the study and development of mobile robots, e.g. robotic manipulators, became a recurrent theme in engineering.

A key concept in the research of robotic manipulators is the trajectory planning. To reach a determined coordinate with the smoothest sequence of movements in a plausible solution, and the ability to avoid obstacles in the manipulator's workspace are essential tasks in this application (Zou et al., 2006).

Trajectory planning refers to how a robot goes from one location to another in a controlled manner. Composed of straight-line motions or sequential motions, the use of kinematics and dynamics of a robot is required. In comparison to a simple path, its advantage is the possibility of configuring the trajectory for each portion of the motion segments between the points through desired speed and acceleration(Niku, 2010).

A commonly used methodology of movement analysis of a robotic manipulator is the kinematic study. When this analysis is done, the geometric complexity increases if the manipulator present several DOF, mainly if we use the inverse kinematics method.

Thekinematic study is an important aspect of a manipulator calibration.In this way, two models are used, the direct and inverse kinematics. The inverse kinematics, used in this work, exhibit challenge due to its equations are non-linear, the manipulators present elevate DOF, with the possibility of presenting multiple solutions(Nunes, 2016).

Some traditional methods as geometric, algebraic and numerical-interactive are used for inverse kinematics solution and are from inappropriate usage whenever in a complex manipulator structure (Alavandar & Nigam, 2008). In this way, some alternative approaches in solution and ANNs application can be found in the literature. Hence, its effectiveness to understand the manipulator is due to the flexibility and capability of learning through training.

In order to accomplish a desired purpose, a recurrent methodology is the use of redundant manipulators, which present more degrees of freedom (DOF) than required to a specific task. Otherwise, the manipulator's end-effector will not have the necessary accuracy (Xiao & Zhang, 2014).

In this work, for a two-dimensional (2D) space, a 3 DOF robotic manipulator is used to reach desired points within its workspace. Three different methodologies are used to solve the inverse kinematics of the manipulator: Geometric Equations System through Matlab® *solve* function, Genetic Algorithms (GAs) and Artificial Neural Networks (ANNs). Thus, the second step is the trajectory planning for a specified point in the manipulator's workspace. In this work, all the manipulator's joints have simultaneous movement in the same crossing time.

Some similar papers can be cited. In work (Tian & Collins, 2004), Tian and Collins propose a GA method for trajectory planning with obstacles in workspace. Polynomials represent the trajectory and are formulated for internal points interpolated with GA parameters. The objective is to search for an optimal solution in the manipulator's workspace.

An intelligent posture calibration method is proposed in (Kuo, Liu, Ho, & Li, 2016) for a robot arm calibration which integrates Particle Swarm Optimization (PSO) and ANN methods. The problem describes an error due to a not ideal mechanism design. The results demonstrated the feasibility and practicability of the proposed method.

In(Savsani, Jhala, & Savsani, 2013), a robotic manipulator trajectory is optimized through Teaching Learning Based Optimization (TLBO) and Artificial Bee Colony (ABC) optimization techniques. The objective was a trajectory planning with less travelling time and distance between joints. The results show better performance of TLBO and ABC in comparison with a GA.

This work is developed as follows: Section II presents the fundamentals of robotic manipulators and the *solve*, GA and ANN techniques. Section III shows the methodology used and the constructive aspects of the studied manipulator. In Section IV, we show and discuss the obtained results for the three approaches, as the trajectory planning for a desired coordinate. Finally, Section V concludes the paper and addresses future works.

II. BACKGROUND

In this section, we present the concepts of robotic manipulators and three polynomial methods used for trajectory planning. In addition, we briefly discuss the GA and ANN techniques used for the manipulator calibration.

1. Robotic Manipulators

Robotic manipulators are devices used in engineering that interact and execute tasks within a workspace, with similar characteristics to the human arms. Several manipulators are installed in industries to handle objects through stations, to welding, assembling *etc*. (Hexmoor, 2013). Fig. 1 shows an example of robotic manipulator with 2 DOF; l_1 and l_2 are the joints' lengths, θ_1 and θ_2 the angles of the first and second joints, and P the desired point.

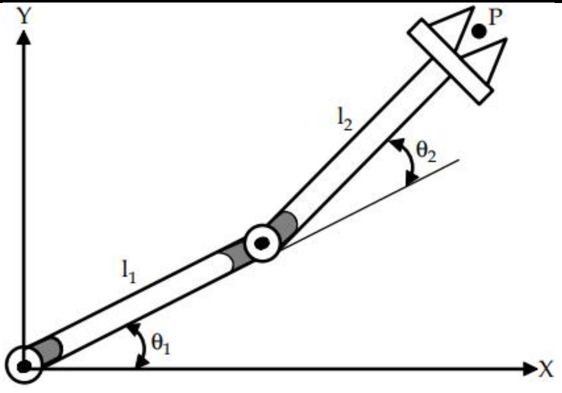

Fig. 1: 2 DOF Robotic manipulator

Robots are unable to respond in emergency situations unless through situation prediction and the response is already included in the system. This scenario divides robotics into the fields of programmed and autonomous robotics. The PUMA, Stanford and others known robots are arms mechanical systems exhibit complex kinematics, static and dynamics, which makes difficult its analysis, control (Niku, 2010)and the interaction between the manipulator and environment (Hu & Xiong, 2018).

With the increase in the use of manipulators, their environments present in several forms. Their interaction with non-static environments led to adaptive manipulator's controllers in order to maintain acceptable performance levels. The applications involve changing loads, varying geometry *etc*.(Zhang & Wei, 2017).

In this way, intelligent systems such as Adaptive Neuro-Fuzzy Inference System (ANFIS), ANNs and GAs have been used in robotics mainly due to the design of autonomous robots and their controllers for unstructured, flexible, and/or partially unknown environments is a very difficult task for a human designer.

Inherent to inverse kinematics is the problem of multiple solutions. In this case, the angles set that lead to the same initial and final points, but with impossible solutions due to the constructive aspect of the manipulator or also through undesirable paths, as also redundant solutions. In addition, the number of possible solutions increases exponentially with increasing DOF.

From the position of origin of a manipulator, i.e. the reference of its base represented by O and the desired position at the other end of the manipulator represented by P, multiple solutions could satisfy the joints' angle configuration. Thus, there are redundancies in the inverse kinematics solution, as shown in Fig. 2, for 2 and 3 DOF manipulators, respectively(Nunes, 2016).

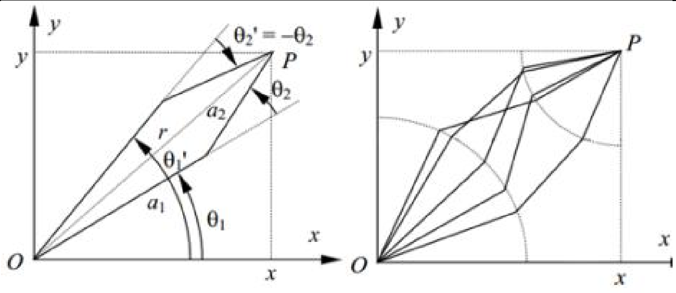

Fig. 2: The problem of multiple solutions

In Fig. 2, θ_1 and θ_2 are the angles of the first and second joints, a_1 and a_2 are the joints lengths and *P* the desired point.

2. Trajectory Planning

The trajectories are composed by a sequence of displacements of a robotic manipulator. It can be seen as a sequence of points in which the end-point must course. Due to the manipulator's discretized movement, we obtain a continuous trajectory. Thus, the trajectory optimization of robotic manipulators is the identification, the optimal combination and the number of intermediary positions (Pires, 1998).

The trajectory planning is presented generally in two forms, the operating space and joint space. In the first one, the trajectory of end-effectors (the manipulator itself) is trivially described. However, it lead to kinematic singularities and manipulator redundancy. In this way, the joint space approach guarantee the smoothness of the joints movement, but reducesthe position accuracy in the operating space. The joint space is the method used in most cases, and the trajectories have been formed by several interpolation functions such as the polynomials used in this work (Huang, Hu, Wu, & Zeng, 2018).

The general form of the polynomials used in the joint space is given as follows. For joints' speed and acceleration, we trivially derive (1) one time for joint speed and again for joint acceleration.

$$\theta(t) = c_0 + c_1.t + c_2.t^2 + \cdots + c_{n-1}.t^{n-1} + c_n.t^n \quad (1)$$

In (1), *t* is the time vector, *θ(t)* represent the angles in time and c_n are the constants associated with the *n*-order polynomial. The *n*-order polynomial enables the user to choose *n+1*specifications, such as initial and final speed and positions, a desired acceleration *etc*.

3. Genetic Algorithms

The development of computational simulations of genetic systems began in the 50s and 60s through many biologists, with John Holland developing the first researches in the area, in 1975 (Holland, 1992). Since then, they have been applied successfully in several real-world search and optimization problems, like regression, feature selection(Kramer, 2017), classification or machine learning (Goldberg, 1989; Pedrycz, Stach, Kurgan, & Reformat, 2005).

GAs are known as a powerful tool for optimization. It is a model designed to emulate natural selection and genetics (Holland, 1992). It has several benefits over conventional optimization methods. The GA use do not require an entire system model, therefore employed trivially to solve optimization problems. According to (Kramer, 2017), GAs are heuristic research approaches applicable to a wide range of optimization problems, which makes them attractive for various problems in practice.

These algorithms are composed of a population of individuals and a set of operators over the population. *A priori*, an initial population consisting of random individuals is selected. Each individual into population represents a solution of the optimization problem, which is coded to the parameter set (chromosome). If the population of individuals is large, the algorithm lacks in efficiency and if it is small GA lacks in diversity.

Posteriori the fitness function is defined. In this work, the Euclidean distance is the fitness function. The individuals are crossed and mutated until the most adequate solution is found. According to the evolutionary theories, through which the GA was developed, the better-adapted individuals to its environment are more likely to survive and reproduce, transmitting their genetic material to the new generations.

In this work, it is used to search for the optimal solution for the inverse kinematics of the manipulator (Tian & Collins, 2004). Fig. 3 show a flowchart used for the GA development (Lopes, Rodrigues, & Steiner, 2013).

In Fig. 3, at first, we generate an initial population of possible solutions. Thus, the individuals are evaluated according to the fitness function. Then, we check if the GA stop criterion was reached: if not, the most adequate individuals are selected through a selection or reproduction method. The selected ones are exposed to the genetic operators (crossing, reproduction, mutation) and a new generation is formed from the previous one. This cycle repeats until the stop criterion is reached. At this moment, the algorithm converges presenting the solution found for the problem (Lopes et al., 2013).

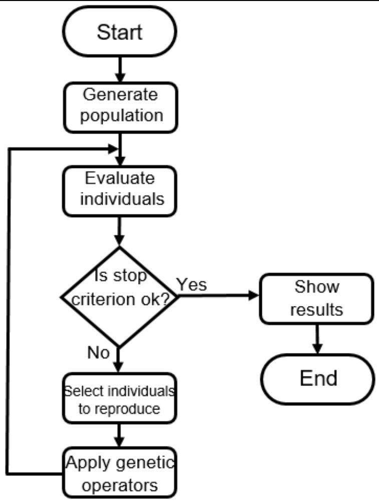

Fig. 3: GA flowchart used

4. Artificial Neural Networks

By a neurobiological analogy, ANNs are based in a fast and powerful brain. In engineering, it as an opportunity to solve complex problems and, to neurobiologists, is a research tool to understand the neurobiology behavior(Haykin, 1998).

The ANNs have a computational power from massively distributed structure and its ability to learn. These two capabilities make it possible for solving complex problems(Haykin, 1998). Thus, ANNs have some capabilities such as nonlinearity, input-output mapping, adaptiveness, evidential response, contextual information, fault tolerance, very-large-scale-integrated (VLSI) implement ability and uniformity of analysis and design (da Silva, Spatti, Flauzino, Liboni, & Alvez, 2017). These concepts are exploited in the next paragraphs.

For nonlinearity, ANNs are a nonlinear structure of artificial neurons distributed throughout the network and are an important figure due to the nonlinearity of input signals(Haykin, 1998).The ANNs also present input-output mapping. The synaptic weighs are modified by training the network with a task-determined number of samples randomly picked in order to minimize de difference between output and desired response. The network learns from the examples as mapping the input-output problem(da Silva et al., 2017).

To become adaptive, an ANN may react at minimum changes in the environment of study causing changing by the synaptic weights. A more robust performance for a non-stationary environment depends on a greater adaptability of the system. However, adaptiveness and robustness are not always proportional. Another aspect of the ANNs is the evidential response. It can provide information about pattern selection and its confidence. The objective is for a better performance and pattern classification(Haykin, 1998).

By meanings of a contextual information,the activation and structure of an ANN define the knowledge which affects all neurons in the network by information dealing. In terms of fault tolerance, anANN is a capable of robust computation by adverse conditions. However, it is uncontrollable and the algorithm must be carefully optimized(Haykin, 1998).

The very-large-scale-integrated (VLSI) technologyimplementability means that some complex behavior tasks are well solved due to a parallel computation by the network. In the feature of uniformity of analysis and design,a notation is used in all domains in a network and manifests through neurons, share of theories and algorithms by many applications, and seamless integration of modules building modular networks(Haykin, 1998).

III. MATERIALS AND METHODS

The first step was to determine the joints' physical limitations. In this case, for all the joints we use a maximum angle of 60º.Then, we generate the point cloud considering the first and second quadrants in the *xy* plane using forward kinematics.

The next step was to choose five test points in order to verify the accuracy of the three methods used to compare them and choose the best method to use in the second phase: the trajectory planning. The point cloud and the test points are shown in Fig. 4. The desired points were (-10,20); (-5,22); (3,25); (10,22); (23,18).

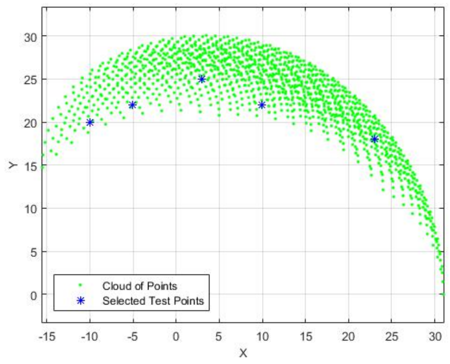

Fig. 4: Point cloud and test points

For the *solve* method, we used (2) to (4) to describe the*(x, y)* positions for each manipulator's joints. Each equation describes one of the DOF of the manipulator, with l_1, l_2 and l_3 being the joint lengths, respectively 7, 10 and 14 cm. $\theta_1, \theta_2$and θ_3 are the angles of joints 1, 2 and 3.

$$X = l_1\,cos(\theta_1) + l_2\,cos(\theta_1+\theta_2) + l_3\,cos(\theta_1+\theta_2+\theta_3) \quad (2)$$

$$Y = l_1\,sin(\theta_1) + l_2\,sin(\theta_1+\theta_2) + l_3\,sin(\theta_1+\theta_2+\theta_3)(3)$$

$$cos^2(\theta_1) + sin^2(\theta_1) = 1(4)$$

For the GA method, several heuristic configurations were tested for initial number of individuals, mutation rate and tournament size. The worst and best configurations tested are shown in Table 1.

Table.1: GA tested configurations

Configuration	Initial population	Tournament size	Stop error [cm]
1	20	5	0.10
2	50	5	0.05

For both GA configurations used in Table 1, we defined a maximum of 500 generations for convergence, simple cross and a 1% mutation. In this work, the second configuration obtained better results, using an initial population of 50 individuals.

For the ANN method, we used a Multilayer Perceptron (MLP) with an offline calibration (training) process. The input layer is composed of two neurons as*x* and *y* coordinates of the manipulator end-effector. There are 100 neurons in the hidden layer and two in the output layer representing the calibration anglesθ_1, θ_2 and θ_3.

The training algorithm used was back-propagation with Levenberg-Marquardt (L-M) method and the sigmoid was used as activation function for the hidden layer and a ramp for the output layer.A limit of 1000 epochs or a network error in order of 10^{-6} for the convergence of the ANN was defined. After the training step, the MLP is able to generalize the output within the point cloud of the manipulator's workspace.

To compare the three methods used, we use the Euclidean error. However, since the geometric *solve* method is exact in this work, it was inconsiderate in the comparisons made. Its use is justified in Section 4.

For the trajectory planning, we chose the method with the lowest relative Euclidean error at the point that presented the smaller error. In this work, due to the offline calibration, the comparison of algorithms' execution times are not discussed. Thus, we show the performance only for the best method in this case.

Once the method has been chosen, the last step is to execute a trajectory planning using the third, fourth and fifth-order polynomials using (1) as reference. In (5) to (13), θ is the angular position of the joint, $\dot{\theta}$ is the angular speed and $\ddot{\theta}$ the angular acceleration.

The third-order polynomial equations are shown in (5), (6) and (7).

$$\theta(t) = c_0 + c_1.t + c_2.t^2 + c_3.t^3(5)$$

$$\dot{\theta}(t) = c_1 + 2.c_2.t + 3.c_3.t^2(6)$$

$$\ddot{\theta}(t) = 2.c_2 + 6.c_3.t(7)$$

The fourth-order polynomial equations are given by (8), (9) and (10).

$$\theta(t) = c_0 + c_1.t + c_2.t^2 + c_3.t^3 + c_4.t^4(8)$$

$$\dot{\theta}(t) = c_1 + 2.c_2.t + 3.c_3.t^2 + 4.c_4.t^3(9)$$

$$\ddot{\theta}(t) = 2.c_2 + 6.c_3.t + 12.c_4.t^2(10)$$

Finally, (11), (12) and (13) describe the fifth-order polynomial system.

$$\theta(t) = c_0 + c_1.t + c_2.t^2 + c_3.t^3 + c_4.t^4 + c_5.t^5(11)$$

$$\dot{\theta}(t) = c_1 + 2.c_2.t + 3.c_3.t^2 + 4.c_4.t^3 + 5.c_5.t^4(12)$$

$$\ddot{\theta}(t) = 2.c_2 + 6.c_3.t + 12.c_4.t^2 + 20.c_5.t^3(13)$$

In this work, in order to obtain a smooth trajectory for the manipulator we design the polynomials to move all the joints simultaneously. For the three cases, we determined initial and final joint speed and position. In the fourth-order polynomial, only the initial acceleration was controlled, while in the fifth-order one both initial and final accelerations were determined.

It is noteworthy that simulation environment is ideal in this work, in other words, it is noise-free, there are no obstacles in the manipulator's workspace and its weight is not considered for the calculations.

IV. RESULTS AND DISCUSSION

In this section, we present and discuss the obtained results for the three methods used for the solution of the robotic manipulator's inverse kinematics and the methods used for its trajectory planning.

1. Robotic manipulator inverse kinematics

In this comparison step, Tables2, 3 and 4 show the results for *solve*, GA and ANN methods used. In Table 2, we zeroed the error due to its Matlab® value was in the order of 10^{-29}, considered as a memory trash. It is also necessary to point out that in Table 4 the ANN method chosen omits the angles values.

Table 2: Matlab® solver results

P	Desired [cm]		Obtained Angles[°]			Obtained [cm]		Error [cm]
	X	*Y*	θ_1	θ_2	θ_3	*X*	*Y*	
1	-10.00	20.00	1.25	-9.39	65.00	-10.00	20.00	0
2	-5.00	22.00	28.00	64.00	50.00	-5.00	22.00	0
3	3.00	25.00	25.00	47.00	44.00	3.00	25.00	0
4	10.00	22.00	-12.97	5.50	3.58	10.00	22.00	0
5	23.00	18.00	27.38	-10.44	-2.84	23.00	18.00	0

Table 3: GA results

P	Desired [cm]		Obtained Angles[°]			Obtained [cm]		Error [cm]
	X	Y	θ_1	θ_2	θ_3	X	Y	
1	-10.00	20.00	52.69	45.18	61.29	-10.21	20.45	0.49
2	-5.00	22.00	38.46	47.47	58.00	-5.13	22.56	0.58
3	3.00	25.00	30.20	35.55	54.25	3.15	24.76	0.28
4	10.00	22.00	17.27	26.46	61.47	10.23	22.50	0.55
5	23.00	18.00	5.24	27.53	24.44	22.66	18.19	0.38

Table 4: ANNresults

P	Desired [cm]		Obtained [cm]		Error [cm]
	X	Y	X	Y	
1	-10.00	20.00	-10.04	19.96	0.05
2	-5.00	22.00	-4.97	22.04	0.05
3	3.00	25.00	3.06	25.09	0.11
4	10.00	22.00	9.97	22.02	0.04
5	23.00	18.00	23.15	18.03	0.16

To graphical represent the obtained points, Fig. 5 present the point cloud, desired and *solve*, GA and ANN reached points.

By the analysis of Tables 2, 3 and 4 we conclude that the *solve* method obtained better results in comparison to GA and ANN. However, its present solution is one among several ones, constituting the multiple solution described in first sections. Hence, this method was discarded since the other methods obtained only one solution.

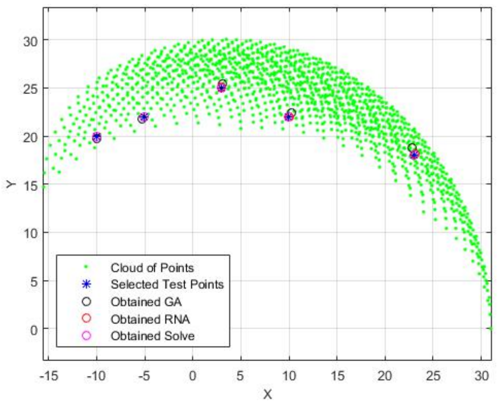

Fig. 5: Point cloud and reached points

Thus, in this case, the ANN presented errors up to ten times lower than the GA approach. Due to that it was chosen for the trajectory planning comparison between the third, fourth and fifth-order polynomials.

2. Trajectory planning

In this step of the work, the trajectory chosen was from the initial point (manipulator in the *x* axis) to Point 2 of Tables2, 3 and 4. The desired trajectory time chosen was 5 s.For all polynomials, we chose initial and final speeds as zero. In the fourth-order one, the final acceleration was designed zero, and in the fifth-order polynomial, both initial and final accelerations are zero.

The angles in discrete time are shown in Tables 5, 6 and 7, respectively for third, fourth and fifth orders. The trajectory planning is shown in Fig. 6. Each joint's angular speed, position and acceleration are seen in Figs. 7(third-order), 8 (fourth-order) and 9 (fifth-order).

Table 5: Third-order polynomial results

Time [s]	θ_1[°]	θ_2[°]	θ_3[°]
0	0.00	0.00	0.00
1	2.88	6.65	5.14
2	9.75	22.50	17.40
3	17.95	41.43	32.04
4	24.82	57.28	44.30
5	27.70	63.93	49.44

Table 6: Fourth-order polynomial results

Time [s]	θ_1[°]	θ_2[°]	θ_3[°]
0	0.00	0.00	0.00
1	5.01	11.56	8.94
2	14.53	33.55	25.95
3	22.73	52.47	40.58
4	26.94	62.19	48.10
5	27.70	63.93	49.44

Table 7: Fifth-order polynomial results

Time [s]	θ_1[°]	θ_2[°]	θ_3[°]
0	0.00	0.00	0.00
1	1.60	3.70	2.86
2	8.79	20.29	15.69
3	18.90	43.63	33.75
4	26.09	60.23	46.58
5	27.70	63.93	49.44

In Tables 5, 6 and 7, we can note that the fourth-order polynomial has the highest acceleration and deceleration. In a real scenario, this fact can cause malfunction and/or damage the motors used to control the joints.

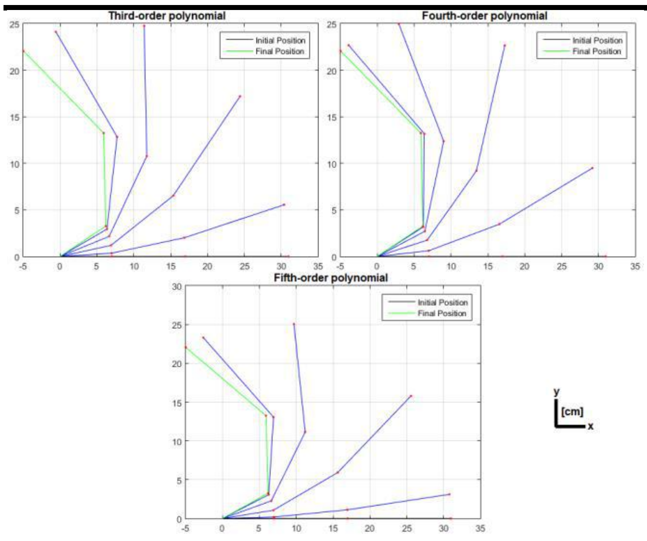

Fig. 6: ANN trajectory planning

In Fig. 6, in its initial position, the manipulator are black and, in the final point, it is green. In its analysis, we can note that third and fifth-order polynomials present higher speeds at the intermediate points of the trajectory, but decelerate in a smaller rate compared to the fourth-order one. Figs. 7, 8 and 9 complement the solution visualization. The *y*-axis represent position (º), speed (º/s) and acceleration (º/s^2) of each joint.

In Fig. 7, the first approach presented a linear response in the acceleration, initiating with high values, which can harm the motors in a possible real application.

As seen in Fig. 8, the fourth-order polynomial present values that, in a real prototype, may harm motors' functioning due to the drastic changes in acceleration and higher speeds in comparison to the other two methods used.

Finally, for Fig. 9, the fifth-order polynomial provides a smooth curve due to its full controllability of position, speed and acceleration, generating a continuous trajectory without discontinuities.

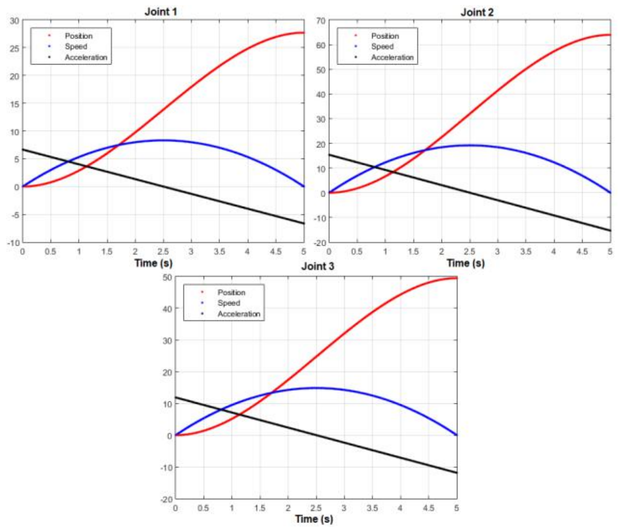

Fig. 7: Third-order ANN trajectory planning

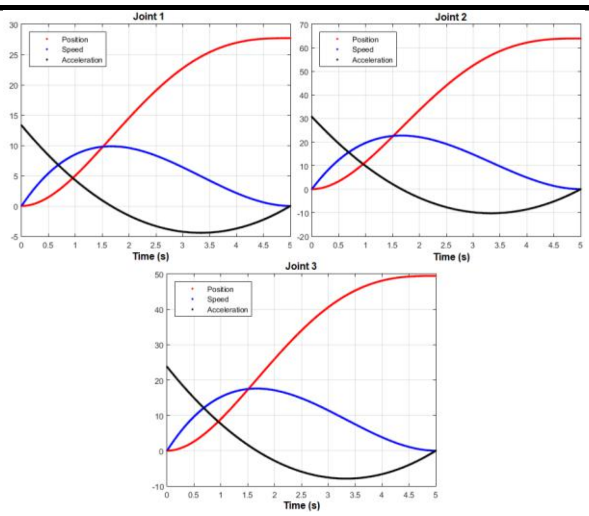

Fig. 8: Fourth-order ANN trajectory planning

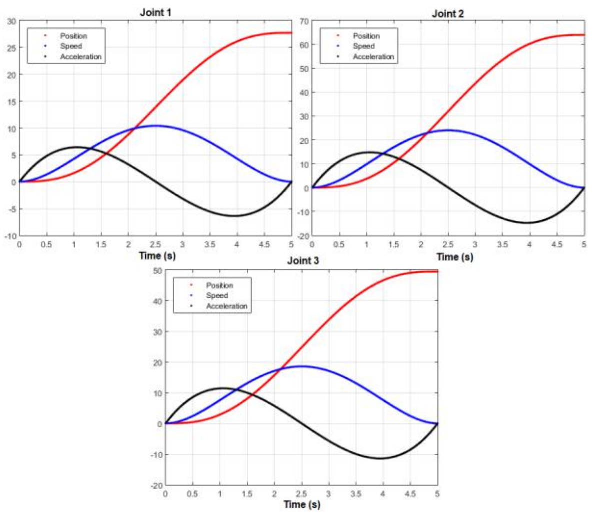

Fig. 9: Fifth-order ANN trajectory planning

V. CONCLUSION

The results provided key-information to future works. Its analysis of the inverse kinematics solution prove that different topologies of MLP can be tested and get feasible results to a three dimensional space and to a more DOF manipulators. It is also noteworthy that this study can be extended to serial and parallel educational and industrial robots.

The fifth-order polynomial trajectory planning results presented the desired smooth curves principally for speed and acceleration, which can be totally user-controlled, that is, assuming values to preserve the components used, e.g. servo or step motors.

This work next step is to use computer vision, through a calibration of a two-camerastereo system in order to use the images to coordinate the robotic manipulator to identify and pick up targets, such as screws, or even execute more complex tasks like welding and cutting operations in three dimensions. The trajectory planning will be done in order to enable the manipulator to avoid possible static or dynamic obstacles autonomously, using intelligent systems like Fuzzy and Fuzzy Cognitive Maps (FCM) to the decision-making.

REFERENCES

[1] Alavandar, S., & Nigam, M. J. (2008). Fuzzy PD+I control of a six DOF robot manipulator. *Industrial Robot: An International Journal*, *35*(2), 125–132. https://doi.org/10.1108/01439910810854610

[2] da Silva, I. N., Spatti, D. H., Flauzino, R. A., Liboni, L. H. B., & Alvez, S. F. R. (2017). *Artificial neural networks: a practical curse* (1st ed.). Cham, Switzerland: Springer International Publishing. https://doi.org/10.1007/978-3-319-43162-8

[3] Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning* (1st ed.). Boston, MA: Addison-Wesley Longman Publishing Co., Inc.

[4] Haykin, S. (1998). *Neural networks: a comprehensive foundation* (2nd ed.). Upper Saddle River, USA: Prentice Hall. Retrieved from http://www.journals.cambridge.org/abstract_S0269888998214044

[5] Hexmoor, H. (2013). *Essential Principles for Autonomous Robotics (Synthesis Lectures on Artificial Intelligence and Machine Learning)* (1st ed.). San Rafael, CA, USA: Morgan & Claypool. https://doi.org/10.2200/S00506ED1V01Y201305AIM021

[6] Holland, J. H. (1992). *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. *The University of Michigan Press* (1st ed.). Cambridge, USA: MIT Press. https://doi.org/10.1137/1018105

[7] Hu, J., & Xiong, R. (2018). Contact Force Estimation for Robot Manipulator Using Semiparametric Model and Disturbance Kalman Filter. *IEEE Transactions on Industrial Electronics*, *65*(4), 3365–3375. https://doi.org/10.1109/TIE.2017.2748056

[8] Huang, J., Hu, P., Wu, K., & Zeng, M. (2018). Optimal time-jerk trajectory planning for industrial robots. *Mechanism and Machine Theory*, *121*, 530–544. https://doi.org/10.1016/J.MECHMACHTHEORY.2017.11.006

[9] Kramer, O. (2017). *Genetic Algorithm Essentials* (1st ed.). Basel, SW: Springer International Publishing. https://doi.org/10.1007/978-3-319-52156-5

[10] Kuo, P. H., Liu, G. H., Ho, Y. F., & Li, T. H. S. (2016). PSO and neural network based intelligent posture calibration method for robot arm. In *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)* (pp. 3095–3100). https://doi.org/10.1109/SMC.2016.7844711

[11] Lopes, H. S., Rodrigues, L. C. de A., & Steiner, M. T. A. (Eds.). (2013). *Meta-Heuristics in Operational Research* (1st ed.). Curitiba, PR, Brazil. Retrieved from Omnipax Ltda

[12] Niku, S. B. (2010). *Introduction to Robotics: Analysis, Systems, Applications* (2nd ed.). Hoboken, New Jersey, USA: John Wiley & Sons.

[13] Nunes, R. F. (2016). *Inverse Kinematics Mapping of a Robotic Manipulator Using Parallel Configuration of Artificial Neural Networks*. Paulista State University (UNESP), Ilha Solteira.

[14] Pedrycz, W., Stach, W., Kurgan, L., & Reformat, M. (2005). Genetic learning of fuzzy cognitive maps. *Fuzzy Sets and Systems*, *153*(3), 371–401.

[15] Pires, E. J. S. (1998). *Genetic Algortihms: Application to Robotics*. University of Porto.

[16] Savsani, P., Jhala, R. L., & Savsani, V. J. (2013). Optimized trajectory planning of a robotic arm using teaching learning based optimization (TLBO) and artificial bee colony (ABC) optimization techniques. In *2013 IEEE International Systems Conference (SysCon)* (pp. 381–386). Orlando, Florida: IEEE. https://doi.org/10.1109/SysCon.2013.6549910

[17] Tian, L., & Collins, C. (2004). An effective robot trajectory planning method using a genetic algorithm. *Mechatronics*, *14*(5), 455–470. https://doi.org/10.1016/J.MECHATRONICS.2003.10.001

[18] Xiao, L., & Zhang, Y. (2014). A new performance index for the repetitive motion of mobile manipulators. *IEEE Transactions on Cybernetics*, *44*(2), 280–292. https://doi.org/10.1109/TCYB.2013.2253461

[19] Zhang, D., & Wei, B. (Eds.). (2017). *Adaptive Control for Robotic Manipulators* (1st ed.). Oshawa, ON: CRC Press.

[20] Zou, A., Hou, Z., Fu, S., & Tan, M. (2006). Neural Networks for Mobile Robot Navigation : A Survey. *Advances in Neural Networks - ISNN 2006*, *II*, 1218–1226. https://doi.org/10.1007/11760023_177