International Journal of Applied Engineering Research ISSN 0973-4562 Volume 11, Number 24 (2016) pp. 11741-11748

Design and comparison of automatic control systems with artificial vision for a 2-DOF cartesian robot

Miguel A. Sarmiento[a], Mauricio Mauledoux[a], Oscar I. Caldas[b]

[a] *Department of Mechatronics Engineering, Universidad Militar Nueva Granada, Bogotá, Colombia.*

[b] *Department of Industrial Engineering, Universidad Militar Nueva Granada, Cajicá, Colombia*.

Abstract

This paper describes the modelling and simulation of the operating principle of a two degrees of freedom (2-DOF) Cartesian Robot, including dynamics and kinematics modelling, followed by the design and implementation of the appropriate control technique that eventually will be used for path tracking, developing a comparative analysis between three proposed controllers: PD, PD+ and the inverse tangent reaching law. The control system data input will be drawn either regular and irregular closed trajectories or shapes captured by an artificial vision system based on edge detection and image processing.

Keywords: Dynamics, kinematics, control system, PD, PD+, artificial vision system, image processing.

INTRODUCTION

Since the first industrial revolution with the mechanization of the British textile industry, the progress in the design of machines and the innovation in automation became key pillars for economic development and modern robotics is a logical result [1, 2]. In the last decades, different techniques in robotics and automation, such as trajectory planning and control have been used by industries to be more productive, resource efficient and responsive to consumers, considering the need to determine how well the manipulators or industrial robots perform a given task. Once the end effector and joint trajectories are planned, knowing the task specifications and controller limits, different controllers can be used to make the robot perform the task [3].

This paper aims to improve the performances of a previously build open-loop 2-DOF Cartesian robot, by using different control strategies and imaging processing techniques and thus achieve the maximum tracking resolution and therefore get a more efficient behavior for all possible industrial applications. Figure 1 shows the Cartesian robot, which also includes an artificial vision system that properly allowed a trajectory detection by images as the input for the open-loop tracking system [4], but with 10 mm of position error that must be reduced. These robots and machine vision systems have been widely used together in industry for accuracy and speed demanding tasks, such as cutting [5], painting [6], gluing [7], fail diagnosis [8], shape tracking [9] or quality control [10].

Figure 1. 2-DOF Cartesian robot.

The work is organized as follows. First kinematic and dynamical models are developed in order to properly analyse the robot behaviour in terms of joint variables and end-effector position [11]. These models were simulated with Matlab Simulink to obtain the joints velocity, acceleration and force as responses of inputs, required for controllers design. Then, three different control techniques for path tracking were addressed and compared in both time response and accuracy, considering a point-to-point desired trajectory [12]. Finally, there is a brief description of the artificial vision system and data processing to recognize the path to track, which allowed to performed tests and hence lead to results and concluding remarks.

KINEMATICS

Forward kinematics is used to get the position and orientation of the end-effector, based on joint variables. As known, the manipulator to analyse is a Cartesian robot with two degrees of freedom and a planar movement, i.e. 2D (x, y) work space or a surface with a set of points that can be reached by its end-effector [13]. This forward kinematics could be made by means of a series of homogenous transformations that relate each link coordinate frame with its neighbours, using the four quantities of the Denavit-Hartenberg notation for describing the link itself and its connections [14]. As a result, the transformation matrix that relates the end-effector with base frame origin in terms of positions and rotations, is shown in eq. (1):

$$A_0^2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & y \\ 0 & -1 & 0 & x \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad (1)$$

The matrix that specifies a mapping from velocities in joint space to velocities in Cartesian space is called the Jacobian, which in this case have only translation variables (no rotations) [15]. The Jacobians of position and orientation in eq. 2 are used for the formulation of motion equations, static analysis of configurations and motion planning [16].

$$J_p = \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} \qquad J_o = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \qquad (2)$$

Using the transformation matrix of forward kinematics and the Jacobian it is possible to analyse the behaviour of the robot, using the Inverse Differential Kinematics algorithm shown in equation (3). This technique for inverting kinematics (and then get joint variables based on position and orientation of the end-effector) is independent of the solvability of the kinematic structure, nonetheless, it is necessary that the Jacobian be square and of full rank.

$$q_{n+1} = q_n + J_p(q_0)^{-1} * \vec{V} \frac{d}{|V| * n} \qquad (3)$$

Where q_n is the coordinates vector of a single step in a point to point trajectory, q_0 is the first step, d is distance, n is the current step, $\vec{V}$ is an direction unit vector and $q_{n+1}$is the coordinates vector of the next step in the trajectory.

The behavior of joint variables and end-effector can be analyzed applying the inverse difference kinematics algorithm, in every single step of a point to point movement. See Figure 2.

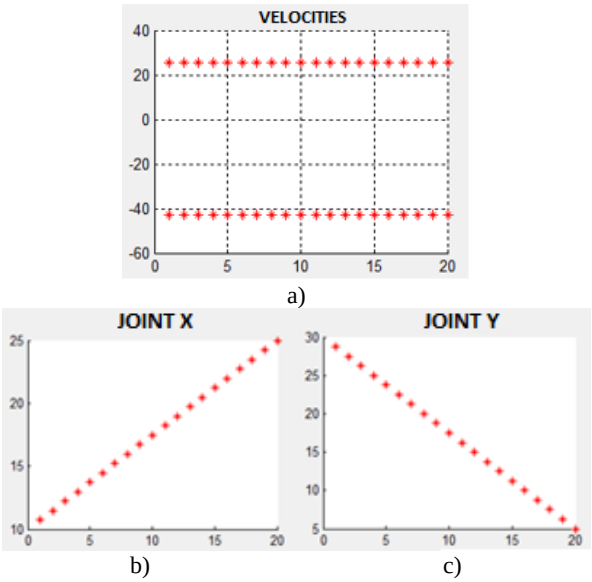


Figure 2. a) X and Y velocities of the end-effector in a 20 point trajectory, during position variation in joints b) X and c) Y.

The inverse difference kinematics algorithm shows the operating principle of the robot in a simple coordinate movement. The joint velocities need to be proportional and constant to achieve that movement, but due to the lack of data used in the kinematic model, the behavior shown is still inappropriate for control design.

DYNAMICS

Obtaining the dynamical model of a manipulator plays an important role for simulation of motion, analysis of manipulator structures and design of control algorithms. Basically, forward dynamical model computes joints acceleration with given joint forces, while inverse model does the opposite job with link masses as parameters [17]. Simulating manipulator motion allows control strategies and motion planning techniques to be tested without the need to use a physically available system. Furthermore, computation of the forces and torques required for the execution of typical motions provides useful information for designing joints, transmissions and actuators.

To get the dynamic model, it was used the Lagrange-Euler formulation, as the equations of motion can be derived in a systematic way independently of the base coordinate frame and effectively describes the link positions of the 2-DOF manipulator, using the mechanical system Lagrangian to obtain the dynamics matrix, as shown in eq. (4) and eq. (5).

$$L = K - U \qquad (4)$$

Where L is the Lagrangian matrix, and T and U denote the total kinetic and potential energy, respectively [15].

$$\begin{bmatrix} F_1 \\ F_2 \end{bmatrix} = \begin{bmatrix} m_1 + m_2 & 0 \\ 0 & m_2 \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{y} \end{bmatrix} \qquad (5)$$

Using Simulink, the dynamic model can be simulated to obtain velocity, acceleration and joint forces as responses to an input, and then design the appropriate control technique.
As can be seen in Figure 3, eq. 5 was driven to a block diagram that shows the sequential derivatives of position inputs and the final products with the links masses. Figures 4, 5 and 6 shows an example of X and Y inputs, the joints velocities and their forces, respectively, which suggests that dynamical model allows to analyze more variables that kinematics model.

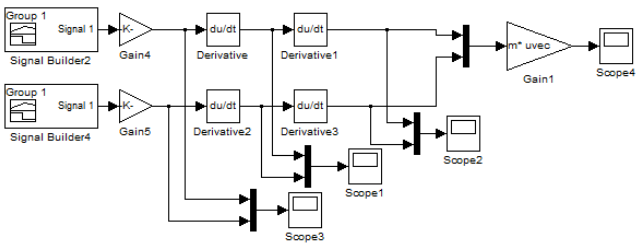


Figure 3. Dynamic model block diagram.

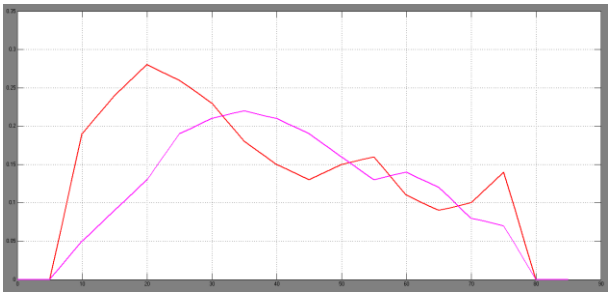


Figure 4. Inputs: X and Y signals.

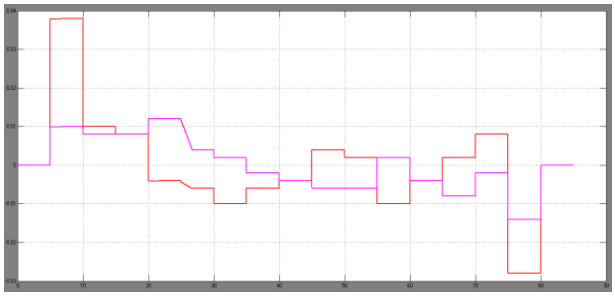


Figure 5. Prismatic joints velocities.

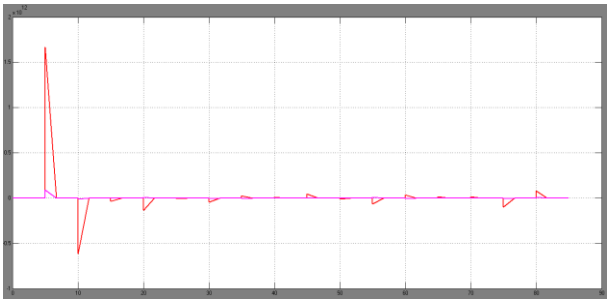


Figure 6. Prismatic joints forces.

CONTROLLERS DESIGN

According to the dynamical and kinematic models, different control techniques are available to this type o mechanism, this paper addresses 3 of them:

- Inverse tangent numeric control with position feedback.
- PD controller.
- PD+ controller.

The inverse tangent numeric control, is based in the point to point movement, assuming constant joint velocities and applying the following algorithm:

θ = atan2((yf-y0),(xf-x0))
Va = 1500 Hz
Vy = Va*sin θ
Vx = Va*cos θ

Algorithm 1. Inverse tangent algorithm.

Figure 7 is the simulation of this first controller on Simulink, and Figure 8 is the obtained approximate result to the input data

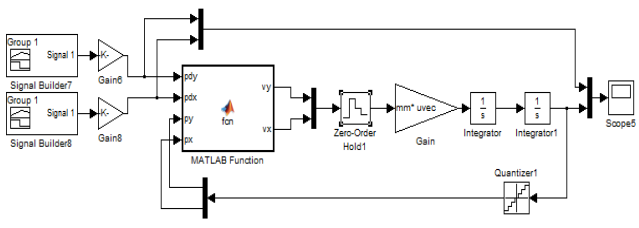


Figure 7. Inverse tangent block diagram.

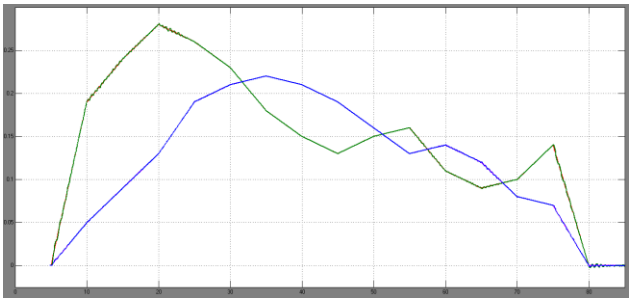


Figure 8. Inverse tangent output.

The PD and the PD+ controllers are designed to obtain a critical damping to minimize the error, as being efficient for general control, despite the nonlinearity and uncertainty of the robots dynamics [18]. In fact, one of the practical methods to control nonlinear systems is to design linear controllers via the linearization of the system about an operating point and the definition of a control law as simple as shown in Eq. 6 [19].

$$u = K_p e - K_d \dot{e} \tag{6}$$

Where K_p and K_d are the proportional and the derivative gain matrices; $e = q_d - q$ and $\dot{e} = \dot{q}_d - \dot{q}$ denote the position and velocity error vectors, defined using the following criteria:

$$K_p = {w_n}^2 \ , \ K_d = 2w_n \ ,$$
$$w_n = \sqrt{K_p} \ , \ 2\delta\sqrt{K_p} = K_v \tag{7}$$
$$\delta = 1$$

The PD controller is a MIMO linear controller that ensures position tracking, based on the structure shown in Figure 9:

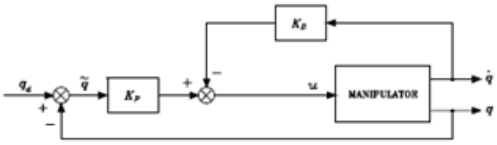


Figure 9. PD controller structure [20].

Using Simulink, the PD controller is simulated with the path shown in Figure 10, acting as position inputs X and Y. Figure 11 corresponds to the block simulation developed, which uses de joints velocities and accelerations (Figures 12 and 13) to obtained the end-effect position as the control system output (Figure 14).

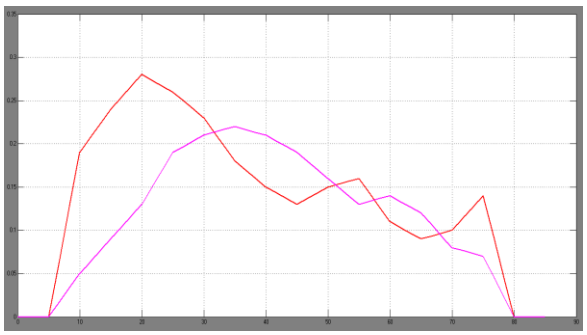


Figure 10. Input path.

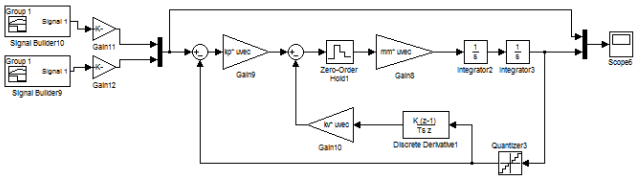


Figure 11. PD block diagram.

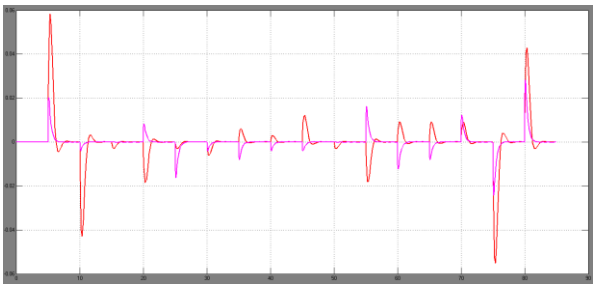


Figure 12. Joints acceleration.

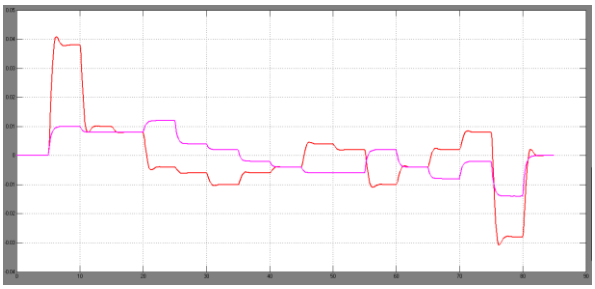


Figure 13. Joints velocities.

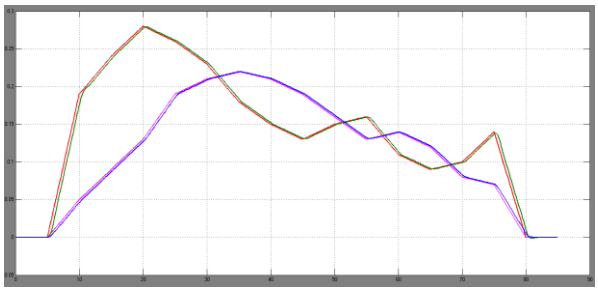


Figure 14. End-effector position.

Nevertheless, the PD controller has no velocity control, so it would have position error and will affect the end result accuracy at considerable levels, just as it can be seen in Figure 14, where both X and Y inputs and outputs were compared. Therefore, the next controller should be considered in order to look for better results.

The PD+ controller is a MIMO lineal controller that involves the dynamic model using stabilizing linear control and nonlinear compensation, via the following structure:

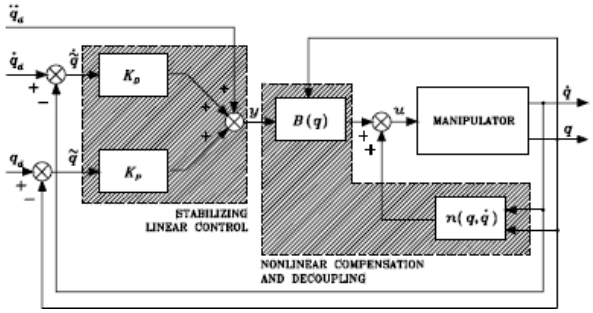


Figure 15. PD+ controller structure [15].

Using Matlab Simulink, the PD+ controller is simulated with the same input path previously shown in Fig 10. Hence, Figures 16, 17, 18 and 19 describes the block diagram of simulation, joints acceleration, joints velocities and end-effector position as the system output, respectively.

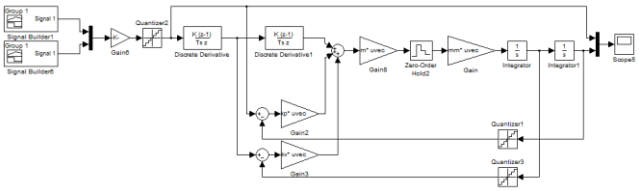


Figure 16. PD+ block diagram.

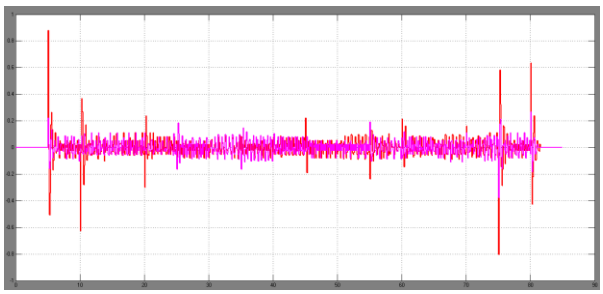
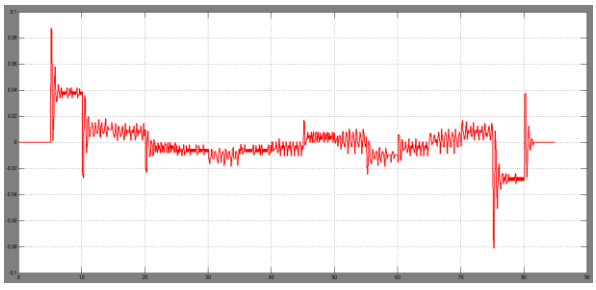


Figure 17. Joints acceleration.



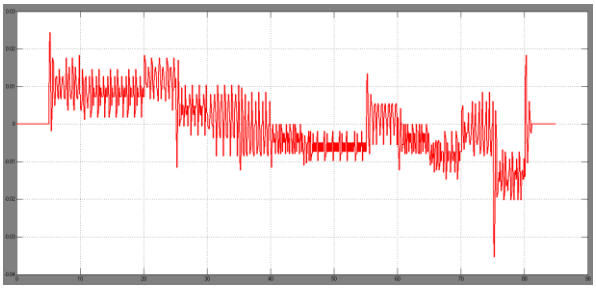


Figure 18. Joint velocities.

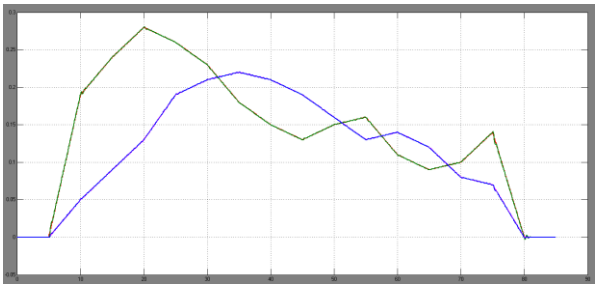


Figure 19. End-effector position.

Figure 19 let to conclude that PD+ controller shows and efficient way to control position and tracking velocity, as the output result is more accurate than the other controllers.

ARTIFICIAL VISION SYSTEM

The artificial vision system is based on a correspondence method used to get the camera intrinsic and extrinsic parameters in order to find the pixel-real world measure correspondence, trying to accomplish the computer vision main aim to duplicate the effect of human vision by electronically perceiving and understanding an image [21].

A C# user interface was developed in order to find the intrinsic and extrinsic camera parameters, via camera calibration with the image-processing library Emgu CV, which makes it possible for OpenCV functions to be called from .NET programming languages, such as C# [22]. The result is shown in Figure 20, where an automatic chessboard detection is performed, with no information supplied regarding the number of rows or columns, as a valuable tool for camera calibration [23].

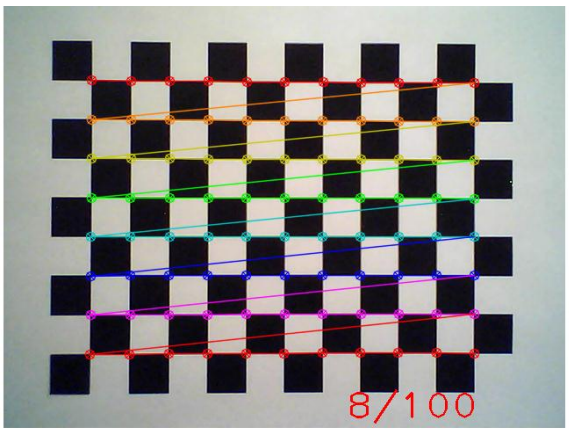


Figure 20. Camera calibration.

Once the camera is calibrated, the C# app corrects the camera distortion and ensure the pixel-mm correspondence according to the chessboard dimensions. In Fig 21. There is an example of a chessboard used to show the results of the camera calibration.

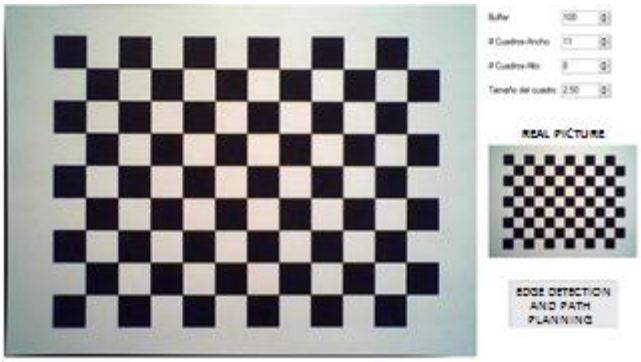


Figure 21. 25x17.5 cm workspace for camera calibration.

Once the camera is calibrated, a shapes detection algorithm is applied to detect closed paths or shapes (Fig. 22) [4].

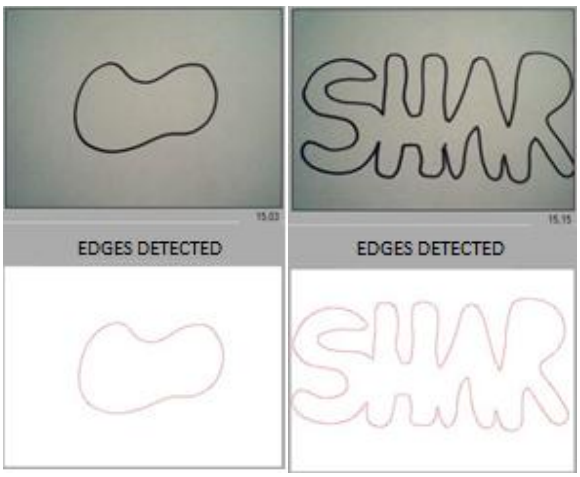
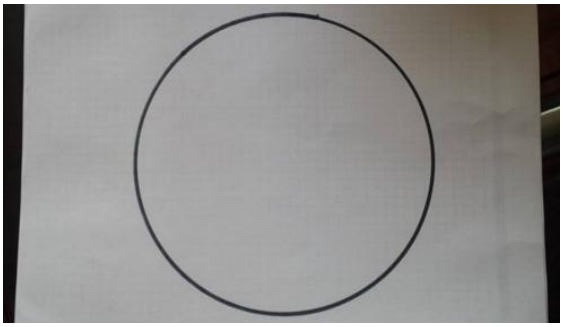


Figure 22. Shapes detection.

The shapes detection algorithm detects all kinds of paths, regular and irregular. The detection resolution depends of the camera pixel resolution, which in this case, was 640x480. To increase the path detection it is required to increase the camera technical features.

FINAL RESULTS

To test the entire system, two different paths were used, as shown in Figure 23. One of them had a regular shape (circular) and the other one was irregular. Both of them were used to test the behavior of the artificial vision system and the control techniques applied in section III. The results are shown in Figure 24.



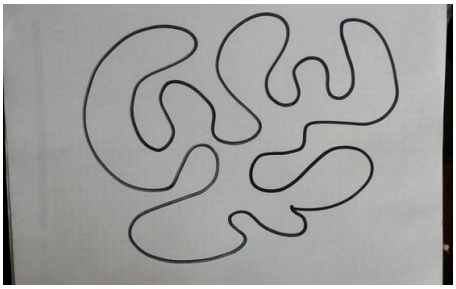


Figure 23. Regular and irregular paths.

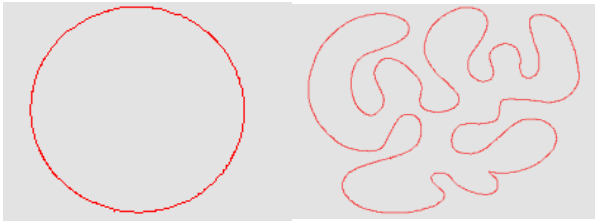


Figure 24. Paths acquired by the artificial vision system.

Once the paths are acquired by the artificial vision system, they are introduced as inputs of the each control technique that were described in section II to analyze final results and velocity behavior.

Figure 25 shows the joint velocities for the circular path input and the Inverse Tangent Numeric Controller. The final result was painted by the robot as shown in Figure 26.

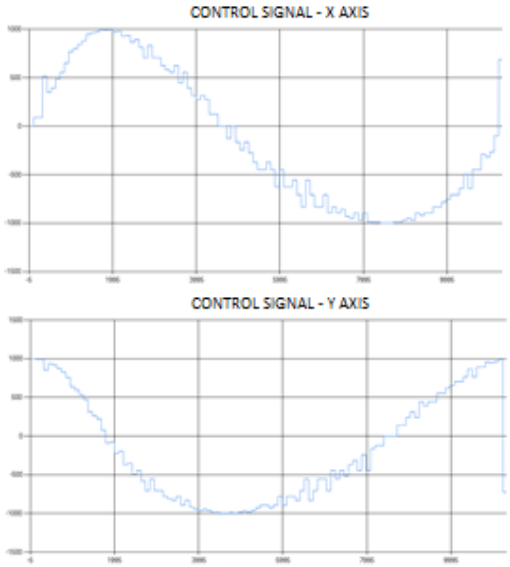


Figure 25. X-Y joint velocities with ITN Controller (circular path)

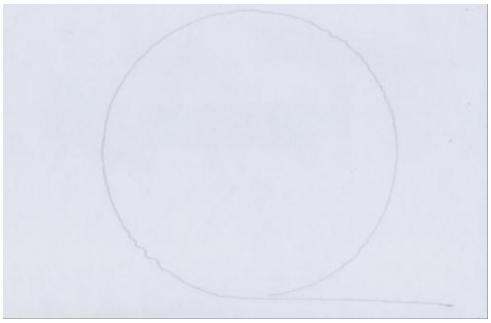


Figure 26. Circular path tracked with ITN Controller.

Figure 27 shows the joint velocities with the irregular path input for the Inverse Tangent Numeric Controller too. The final result was painted by the robot as shown in Figure 28.

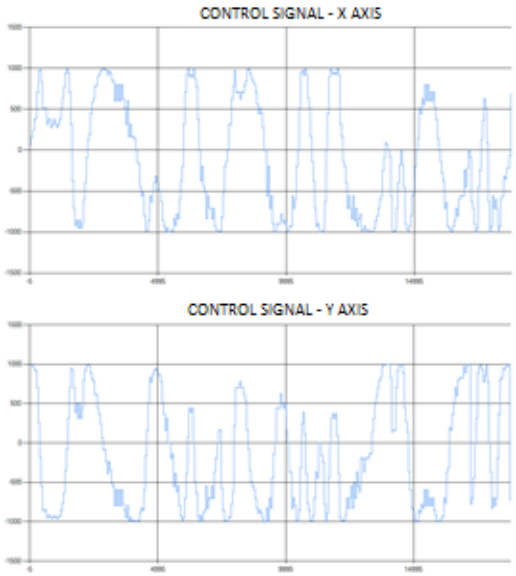


Figure 27. X-Y joint velocities with ITN Controller (irregular path)

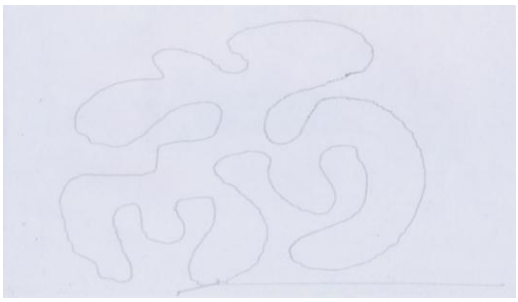


Figure 28. Irregular path tracked with ITN Controller.

Figures 29 to 32 show the same work performed by the PD controller, i.e. joints velocities acquired by the C# app and the path painted by the 2-DOF Cartesian robot for both circular and irregular paths:

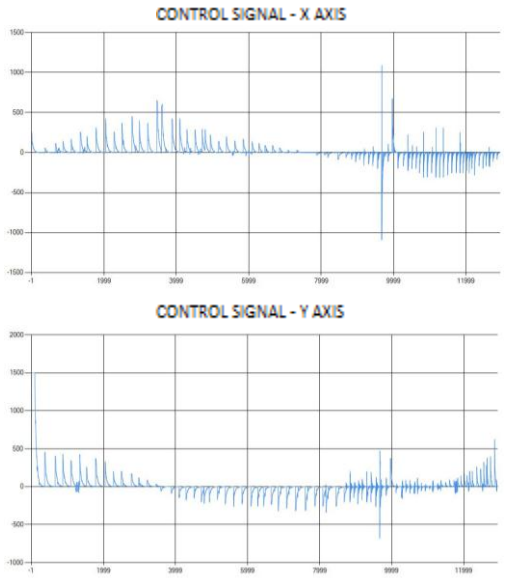


Figure 29. X-Y joint velocities with PD Controller (circular path)

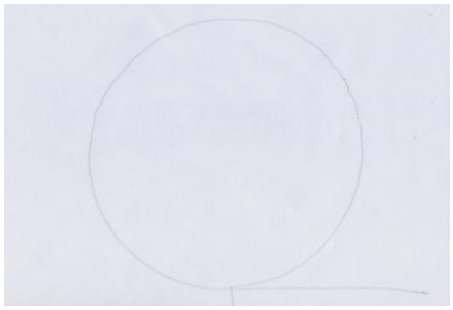


Figure 30. Circular path tracked with PD Controller.

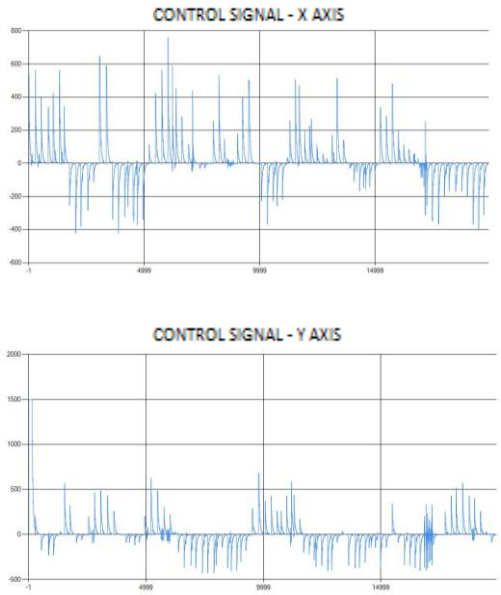


Figure 31. X-Y joint velocities with PD Controller (irregular path)

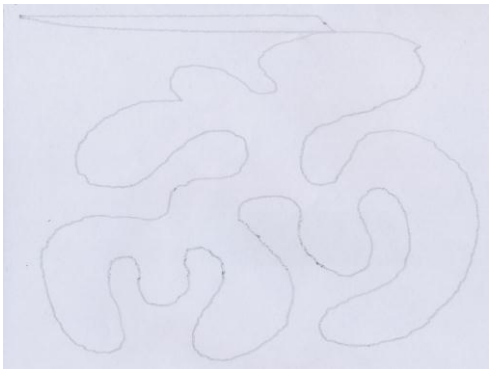


Figure 32. Irregular path tracked with PD Controller.

Finally, Figures 33 to 36 show the same work performed by the PD+ controller, i.e. joints velocities acquired by the C# app and the path painted by the 2-DOF Cartesian robot for both circular and irregular paths:

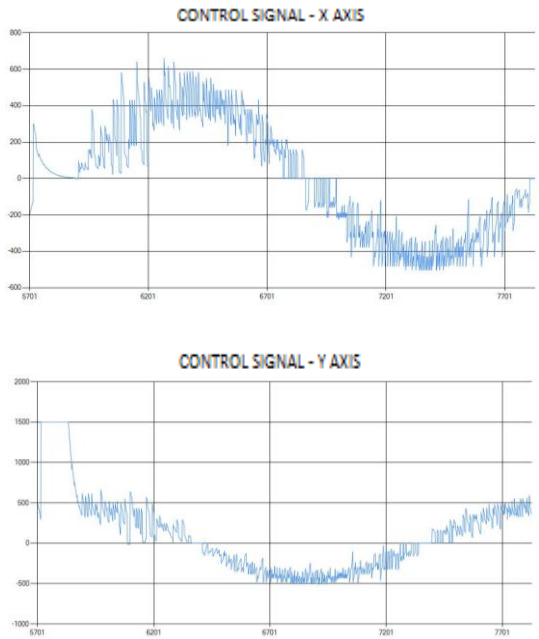


Fig 33. X-Y joint velocities with PD+ Controller (circular path)

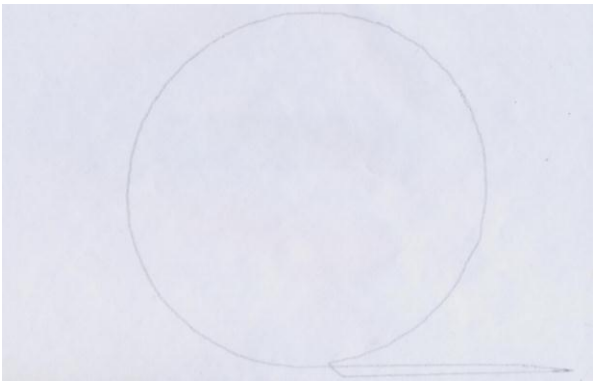


Figure 34. Circular path tracked with PD+ Controller.

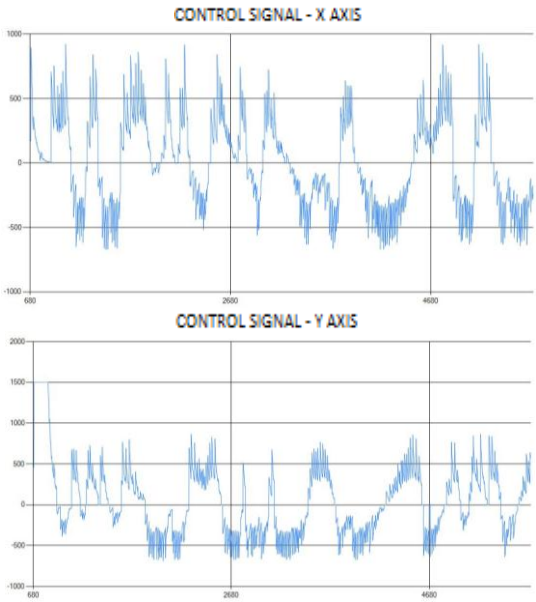


Figure 35. X-Y joint velocities with PD+ Controller (irregular path).

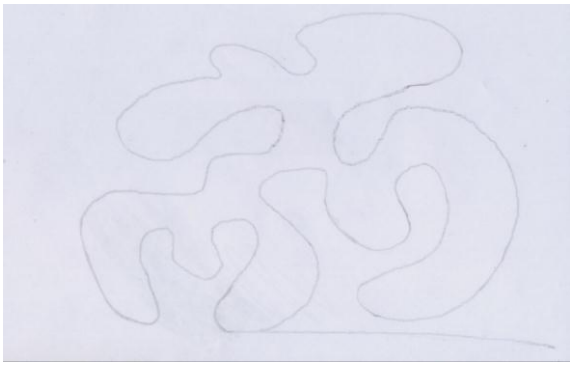


Figure 36. Irregular path tracked with PD+ Controller.

CONCLUSSIONS

According with the final results obtained from the controllers, it was easy to conclude that the PD+ controller had the most complete behavior of all three. The inverse tangent controller was the fastest to complete the path tracking, but with mayor position error; the PD controller was the slowest of all and also complete the path tracking with a not so good accuracy. Finally, the PD+ controller complete the path tracking at high speed with high accuracy. The control techniques applied have multiple applications like: industrial mechanism, CNC machines, 3D printers and others.

The artificial vision system was a low cost device that worked perfectly with the given circumstances. It showed high accuracy and efficiency with low limitations. This kind of systems can be applied in different fields of engineering, such as virtual reality, space recognizing, 3D reconstruction, 3D scanner, among others.

REFERENCES

[1] G.-Z. Yang, V. Pawar, J. Law y C. Maple, "Manufacturing Robotics: The Next Robotic Industrial Revolution," *UK-RAS White Papers,* 2016.

[2] B. Struijk, "Robot production volume data trends and analysis," *Debreceni Muszaki Közlemények,* vol. 1, pp. 1-10, 2012.

[3] S. R. Munasinghe, M. Nakamura, S. Goto y N. Kyura, "Trajectory Planning for Inustrial Robot MManipulator Considering Assigned Velocity and Allowance Under Joint Acceleration Limit," *International Journal of Control, Automation and Systems,* vol. 1, nº 1, pp. 68-75, March 2003.

[4] R. González y R. Woods, "Digital Image Processing," 3rd ed., Pearson, 2007.

[5] M. Y. Sarmiento y J. P. Rojas, "Construcción de una Máquina para Corte Térmico de Siluetas Metálicas Asistida por Computador," *Revista Colombiana de Tecnologías de Avanzada,* 2006.

[6] H. Chen, W. Sheng, N. Xi, M. Song y Y. Chen, "Automated robot trajectory planning for spray painting of free-form surfaces in automotive manufacturing," of *IEEE International Conference onRobotics and Automation.*, 2002.

[7] Z. Xu, S. Wei, N. Wang y X. Zhang, "Trajectory Planning with Bezier Curve in Cartesian Space for Industrial Gluing Robot," de *International Conference on Intelligent Robotics and Applications*, 2014.

[8] J. Gosalbez, A. Salazar, I. Bosch, R. Miralles y L. Vergara, "Application of Ultrasonic Nondestructive Testing to the Diagnosis of Consolidation of a Restored Dome," *Materials Evaluation*, 2006.

[9] J. L. López, M. A. Santiago y J. Francisco Domínguez, "Robot Cartesiano: Seguimiento de Trayectorias Irregulares Arbitrarias Mediante Computadora,"Universidad Autónoma del Estado de Hidalgo, 2007.

[10] H. F. Sánchez y A. R. Morales, "Sistema de visión artificial para la inspección, selección y control de calidad de fresas," Universidad Industrial de Santander, 2009.

[11] T. Khan, M. Arshad y M. Choudhry, "Modeling and Control of Cartesian Robot Manipulator," de *9th International Multitopic Conference, IEEE INMIC*, 2005.

[12] C. Gosselin y S. Foucault, "Dynamic Point-to-Point Trajectory Planning of a Two-DOF Cable-Suspended Parallel Robot," *IEEE Transactions on Robotics,* vol. 30, nº 3, pp. 728-736, 2014.

[13] J. Craig, "Introduction to Robotics: Mechanics and Control", Pearson, 2005.

[14] J. Denavit y R. Hartenberg, "A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices," *Journal of Applied Mechanics,* pp. 215-221, 1955.

[15] B. Siciliano y L. Sciavicco, "Modeling and Control of Robot Manipulators," M. Hill, Ed., Springer, 2000.

[16] P. Sánchez-Sánchez y F. Reyes-Cortés, «Cartesian Control for Robot Manipulators,» de *Robot Manipulators Trends and Development*, A. Jiménez y B. Hadithi, Edits., InTech Open, 2010, pp. 166-212.

[17] R. D. Robinett, C. R. Dohrmann, G. R. Eisler y J. T. Feddema, "Flexible Robot Dynamics and Controls," I. F. for Systems Research International Series on Systems Science y Engineering, Edits., Springer, 2002.

[18] J. Huang, C. Yang y J. Ye, "Nonlinear PD Controllers with Gravity Compensation for Robot Manipulators," *Cybernetics and Information Technologies,* vol. 14, nº 1, pp. 141-150, 2014.

[19] P. Ouyang y W. Zhang, "Comparison of PD-Based Controllers for Robotic Manipulators," of *ASME 2004 Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, 2004.

[20] F. L. Lewis, D. M. Dawson y C. T. Abdaliah, "Robot Manipulator Control: Theory and Practice," N. Munro, Ed., Marcel Dekker, 2004.

[21] M. Sonka, V. Hlavac y R. Boyle, "Image Processing, Analysis and Machine Vision, "Springer US, 1993.

[22] S. Shi, "Emgu CV Essentials," PACKT Publishing, 2013.

[23] A. de la Escalera y J. M. Armingol, "Automatic Chessboard Detection for Intrinsic and Extrinsic Camera Parameter Calibration," *Sensors,* vol. 10, nº 3, pp. 2027-2044, March 2010.