ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/314091763

Path Tracking Simulation for a Two-Degree-of-Freedom Pneumatic Manipulator Robot

Conference Paper · May 2016

CITATIONS
0

READS
267

4 authors, including:

Sergio Daniel Sánchez Solar
Instituto Nacional de Astrofísica, Óptica y Electrónica (INAOE)
2 PUBLICATIONS **0** CITATIONS

SEE PROFILE

Jesus Carlos Pedraza Ortega
Autonomous University of Queretaro
120 PUBLICATIONS **384** CITATIONS

SEE PROFILE

Some of the authors of this publication are also working on these related projects:

Project Machine Learning Algorithms View project

Project Clasification of biomedical signals using Artificial Intelligence View project

All content following this page was uploaded by Sergio Daniel Sánchez Solar on 28 February 2017.

The user has requested enhancement of the downloaded file.

Path Tracking Simulation for a Two-Degree-of-Freedom Pneumatic Manipulator Robot

Sergio-Daniel Sánchez-Solar*, Jesús-Carlos Pedraza-Ortega†, Efrén Gorrostieta-Hurtado‡ and Juan-Manuel Ramos-Arreguín§

*Universidad Autónoma de Querétaro, Facultad de Informática; e-mail: ssanchez37@alumnos.uaq.mx
†Universidad Autónoma de Querétaro, Facultad de Ingeniería; e-mail: caryoko@yahoo.com
‡Universidad Autónoma de Querétaro, Facultad de Ingeniería; e-mail: efrengorrostieta@gmail.com
§Corresponding Author: Universidad Autónoma de Querétaro, Facultad de Ingeniería; e-mail: jramos@mecamex.net

***Abstract*—Nowadays, Robotics is a broad field of research, mainly when it comes to pneumatic robots due to its high nonlinearity. Most works have been developed using electric or hydraulic actuators, there are few works that use pneumatic actuators, this is the principal reason to develop a tool that permits to simulate the behaviour of the manipulator before the physical implementation, it permits to the user to have an idea about the response of the manipulator. In this paper a simulation software for a Two-Degree-of-Freedom Pneumatic Manipulator Robot is presented, using OpenGL libraries for developing the simulator and C++ to calculate the interpolation among the points that are given as input. Matlab is used to calculate the direct and inverse kinematics to find the angles and positions that the manipulator must reach and to obtain the response of the system after the control is applied. Matlab also process the control that is applied to the system to be capable of reaching the points that were established as set-points.**

***Keywords*—Pneumatic Manipulator, Path Tracking, OpenGL Simulation, PID Controller.**

I. Introduction

Pneumatic Actuator Systems have a lot of properties such as, cheapness and cleanness in comparison to hydraulic actuators. Pneumatic actuators present more durability and they are lighter than others. This set of features has made pneumatic actuators very attractive for different applications, mainly in robotics [1].
In the other hand, pneumatic actuators present high non-linearities that make them hard to control, these non-linearities are related to compressibility of the air, the time delay provoked by the slow propagation of the air pressure waves, and the associated large friction forces [2]. Some researches present a simplified termo-mechanical model for pneumatic actuators, so it is necessary to obtain the dynamical model of the manipulator [3], and the dynamic behaviour of a pneumatic manipulator [4].

To track trajectories, is necessary to solve two main problems, trajectory planning and trajectory control [5]. It is very important to select an appropriate method that describes the trajectory that the manipulator has to follow in order to obtain a curve that fits to the original path, that is the reason for using some interpolation techniques that approximate the trajectory to the original curve.
In this paper a simulator of a Two-Degree-of-Freedom Manipulator Robot is presented, it is using a polynomial interpolation technique known as Splines that permits to obtain intermediate points from a few points given to generate a curve that the system will follow. The equations to obtain the direct and inverse kinematics, useful to obtain the points of the final position and the angles of the links, are also presented. The control that is applied to the system was developed in [6]. The morphology of the manipulator robot is similar to the used in [7] but a rotational degree is added using a pneumatic motor.

II. Kinematic Equations

It is necessary to know the kinematic equations that describe the system, they are presented in the following subsections. The system that is being controlled is shown on Fig. 1.

A. Inverse Kinematics

When the points that the manipulator must reach are given, it is necessary to get the angles that the links have to take to be able to go to those points.

Equation (4) is used to obtain the rotation angle in relation to z axis. Equations (5) and (6) represents the angles θ_2 and θ_3 in relation to the xy plane. In equation (2) and (3), α and β are auxiliary angles that permit to obtain the angle θ_2. Using Pythagoras theorem, the helper line L_a is calculated.

$$L_a = \sqrt{L_2^2 + L_3^2} \tag{1}$$

$$\alpha = sin^{-1}(\frac{h}{L_a}) \tag{2}$$

$$\beta = cos^{-1}(\frac{L_2}{L_a}) \tag{3}$$

$$\theta_1 = tan^{-1}(\frac{y_p}{x_p}) \quad (4)$$

$$\theta_2 = \alpha + \beta \quad (5)$$

$$\theta_3 = \theta_2 - \frac{\pi}{2} \quad (6)$$

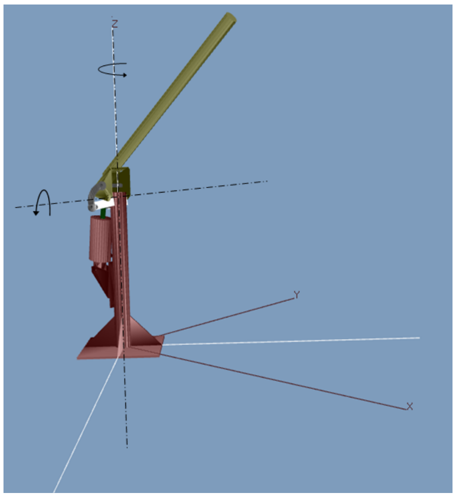

Fig. 1. Pneumatic Manipulator

B. Direct Kinematics

Direct kinematic equations (8-10) are used to obtain the Workspace of the manipulator, this means the area that can be reached by the robot.

$$\theta_2 = \theta_3 + \frac{\pi}{2} \quad (7)$$

$$x_p = (L_2 cos\theta_2 + L_3 cos\theta_3) cos\theta_1 \quad (8)$$

$$y_p = (L_2 cos\theta_2 + L_3 cos\theta_3) sin\theta_1 \quad (9)$$

$$z_p = L_1 + L_2 sin\theta_2 + L_3 sin\theta_3 \quad (10)$$

III. Workspace

Using the direct kinematic equations and specifying the movement limitations of the pneumatic manipulator (Table I), it is possible to obtain the Workspace of the system that is shown on Fig. 2.

TABLE I
ACTUATOR RESTRICTIONS

Actuator	Minimum Value	Maximum Value
Motor	$0°$	$270°$
Cylinder	$-72°$	$86°$

The next algorithm is used to obtain the Workspace of the Manipulator:

Algorithm 1 receives the length of each link of manipulator and angles θ_1, and θ_3 (Lines 2 and 3) as inputs and an index i (Line 1) that indicates the number of data that is being obtained as output for each coordinate x_p, y_p and z_p (Lines 5, 6 and 7).

Algorithm 1 Algorithm to obtain the Workspace

Require: $i, \theta_1, \theta_3, L_1, L_2, L_3$
Ensure: x_p, y_p, x_p

```
1:  i = 1
2:  for θ1 = Motor_min : Motor_max do
3:      for θ3 = Cylinder_min : Cylinder_max do
4:          θ2 = θ3 + π/2
5:          x_p(i) = (L2 cosθ2 + L3 cosθ3) cosθ1
6:          y_p(i) = (L2 cosθ2 + L3 cosθ3) sinθ1
7:          z_p(i) = L1 + L2 sinθ2 + L3 sinθ3
8:          i = i + 1
9:      end for
10: end for
```

On each iteration the angle θ_2 is calculated such as in Line 4 considering that the angle that is composed by L_2 and L_3 is a right angle, this means that they both have the reason shown in direct kinematics.

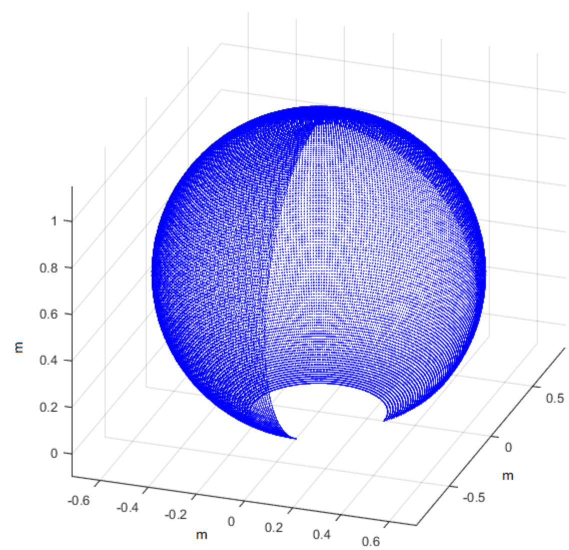

Fig. 2. Workspace of 2DOF Manipulator

IV. Path Tracking

In this section a Splines algorithm to interpolate the points that the Robot must reach is implemented. Cubic Splines are being used due to their smoothness in comparison to other interpolation techniques, such as Newton or Lagrange [8]. Splines is a method used to fit a series of unique cubic polynomials between each of the data points and let the curve obtained be continuous and appear smooth [9].

Given $n+1$ data:

x	x_0	x_1	...	x_n
y	y_0	y_1	...	y_n

A cubic spline that interpolates these points, is a function $S(x)$ defined as follow:

$$s(x) = \begin{cases} s_0(x) & if \quad x \in [x_0, x_1] \\ s_1(x) & if \quad x \in [x_1, x_2] \\ \vdots & \\ s_{n-1}(x) & if \quad x \in [x_{n-1}, x_n] \end{cases} \tag{11}$$

where s_i is a third degree polynomial defined by:

$$s_i(x) = a_i(x-x_i)^3 + b_i(x-x_i)^2 + c_i(x-x_i) + d_i \tag{12}$$

for $i = 1, 2, 3, ..., n-1$. The first and second derivatives of these n-1 equations are fundamental to this process, and they are

$$s_i' = 3a_i(x-x_i)^2 + 2b_i(x - x_i + c_i) \tag{13}$$

$$s_i'' = 6a_i(x-x_i) + 2b_i \tag{14}$$

for $i = 1, 2, ..., n-1$.

For the development of this work, the points given were:

x	-0.1	0.1	0.2	0.4
y	-0.1	0.1	0.5	-0.2
z	1.13853	1.13853	0.871101	0.977196

And the resultant curve of the splines interpolation is shown on Fig. 3.

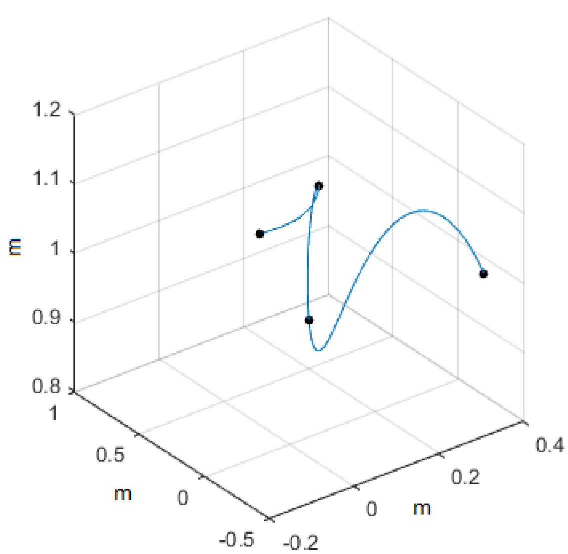

Fig. 3. Result of Cubic Splines Interpolation

V. Termo-mechanical Model

In [10] the following simplified termo-mechanical model is presented (15-24):
For the interval $0 \le X \le L$:

$$\dot{X} = \frac{d}{dt}X \tag{15}$$

$$D\dot{X} = \frac{d^2}{dt^2}X \tag{16}$$

For the interval $0 \le X \le L_{alp}$:

$$\dot{P}_{a1} = g_{21}(X)(\dot{m}_{a1} - \dot{m}_{c1} - 9.176 \times 10^{-10} P_{a1} DX) \times 10^8 \tag{17}$$

$$\dot{P}_{c1} = g_{31}(X)(\dot{m}_{c1} - 3.608 \times 10^{-8} P_{c1} DX) \times 10^6 \tag{18}$$

For the interval $L_{alp} < X \le L$:

$$\dot{P}_{a1} = g_{22}(X)(\dot{m}_{a1} - 3.7 \times 10^{-8} P_{a1} DX) \times 10^{11} \tag{19}$$

$$\dot{P}_{c1} = g_{32}(X)(\dot{m}_{c1} - 3.7 \times 10^{-8} P_{c1} DX) \times 10^{11} \tag{20}$$

For the interval $0 \le X \le (L - L_{alv})$:

$$\dot{P}_{c2} = g_{41}(X)(\dot{m}_{c2} + 3.469 \times 10^{-8} P_{c2} DX) \times 10^{11} \tag{21}$$

$$\dot{P}_{a2} = g_{51}(X)(\dot{m}_{a2} + 3.469 \times 10^{-8} P_{a2} DX) \times 10^{11} \tag{22}$$

For the interval $(L - L_{alv}) < X \le L$:

$$\dot{P}_{c2} = g_{42}(X)(\dot{m}_{c2} + 3.352 \times 10^{-8} X_4 X_6) \times 10^{13} \tag{23}$$

$$\dot{P}_{a2} = g_{52}(X)(9.983 \times 10^3 (\dot{m}_{a2} - \dot{m}_{c2}) + 1.168 \times 10^{-5} X_5 X_6) \times 10^4 \tag{24}$$

VI. Control of the Pneumatic Manipulator

Fig. 4 shows the mechanical-pneumatic system used to generate the movement of the manipulator.

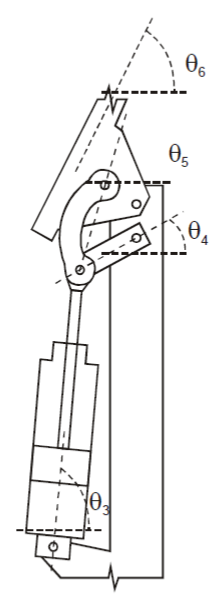

Fig. 4. Mechanical-Pneumatic System for the Pneumatic Manipulator

The displacement of the rod is what must be controlled, because it produces the output angle θ_6 of the impulse mechanism and it affects directly the movement of the final link of the manipulator.
The control that is implemented in this system is a PID control developed in [6] . The inputs of the Termo-mechanical model are the free effective area of the air flow and are shown in (25).

$$u = [A_1, A_2, A_3] \tag{25}$$

Where:
A_1 $\rightarrow$ The effective area the air flow in the base of the cylinder.
A_2 $\rightarrow$ The effective area the air flow in the rod.
A_3 $\rightarrow$ The effective area for air flow return.

The diagram of control that represents the system is shown in Fig. 5, where (e) is the error vector entering the PID controller algorithm that gives the opening percentage vector (u), that enter the pneumatic actuator system and generate the displacement of the rod (X), and this gives the enough force to generate the movement of the manipulator and to obtain the output of the plant (θ).

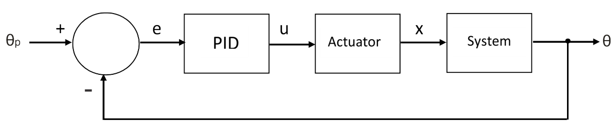

Fig. 5. Control of the angle θ of the pneumatic manipulator

Equation (26) and (27) give as result the values of (25) and they shows the control equation of each valve and the error vector.

$$e = [e_p, e_i, e_d] \tag{26}$$

$$A_j = K_p e_p + K_i e_i + K_d e_d \tag{27}$$

In (27) $j = 1, 2, 3$ indicates the valve to be controlled. K_p, K_i and K_d are the proportional, integral and derivative constants, shown in Table II. e_p, e_i and e_d are the proportional, integral and derivative errors that can be obtained from (28-30).

$$e_p = \theta_p - \theta \tag{28}$$

$$e_i = \sum_i e(T_i) \tag{29}$$

$$e_d = e(T_i) - e(T_{i-1}) \tag{30}$$

TABLE II
CONTROL VALUES USED IN PID CONTROLLER

Valve	K_p	K_i	K_d
A_1	4.00	0.0	100.0
A_2	-4.00	0.0	-100.0
A_3	0.45	0.0	0.0

VII. RESULTS

Using the graphic interface developed in OpenGL and C++ [11], it is necessary to add the trajectories obtained from the interpolation algorithm and kinematic equations, because of the computational cost of these operations, they are processed in Matlab because it has a better performance working with matrix operations. The result of the previous operations is saved in a text file and loaded into the simulator program. Resultant movement in degrees of the pneumatic motor is displayed in Fig. 6. The movement described for the second link of the manipulator in degrees is shown in Fig. 7.

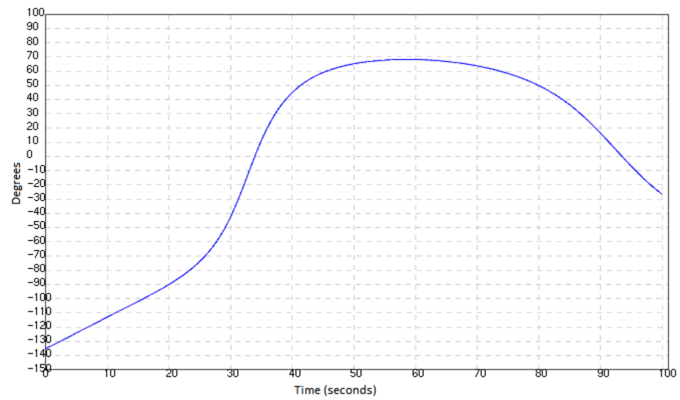

Fig. 6. Movement of the motor according to interpolation

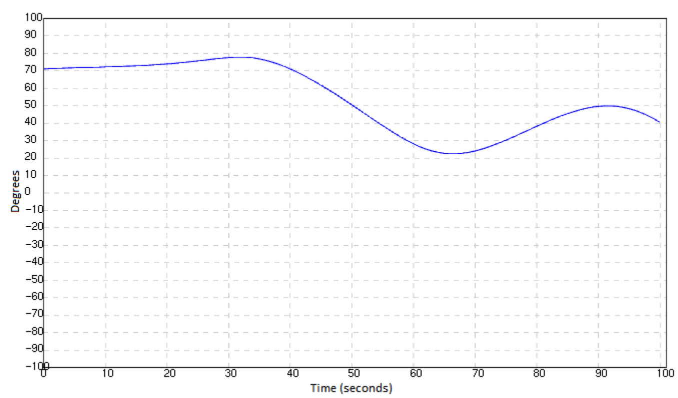

Fig. 7. Movement of the second link of the manipulator according to interpolation

After obtaining these graphics, the next step is to implement the control algorithm developed in [6] to obtain the real behaviour of the pneumatic cylinder considering the simplified termo-mechanical model and PID controller. The result is presented in Fig. 8. In this case, the behaviour of the pneumatic motor is the same than in Fig. 6 due to its termo-mechanical model and dynamics have not been already obtained.

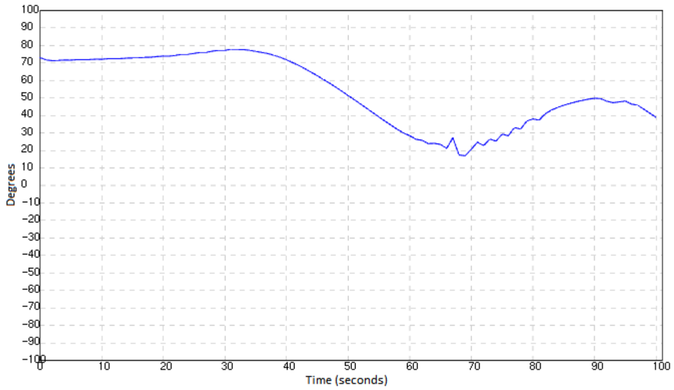

Fig. 8. Movement of the second link of the manipulator considering termo-mechanical model and applying PID control

Fig. 9 shows an interpolated trajectory of 4 points with 50 intermediate points generated between the first point and the last point. It is observed that after applying the controller, the response obtained approaches the reference trajectory but when the displacement of the rod moves out of the cylinder, it does not achieve it, and when it moves into the cylinder, it presents oscillations that are undesired.

In Fig. 10 the number of intermediate points is increased to 100 points and it is observed that when controller is applied the rod reaches the set point, but when the displacement changes its direction there are overshoots of up to 30 percent of the desired value and they remain during the rest of the movement, and this can affect the manipulator due to the limitations of the system, for example, the maximum displacement of the rod.

On the other hand, in Fig. 11 the number of intermediate points of the interpolation is increased to 500, and the response of the system after controller is applied is smoother and is more in line with the set-point given as reference. This behaviour is more convenient to the system because it does not present overshoots or oscillations that would not be acceptable to a specific application and could damage the system or even the operator.

In Fig. 12 an image that shows the simulator developed in OpenGL and C++ is presented, the coordinates x_p, y_p and z_p of the final effector are displayed, and the angles of each link. In the simulator the trajectory generated is shown, this permits to the user to supervise that the trajectory that is being followed corresponds to the established one. It is controlled by keys from the keyboard that allows to rotate the manipulator and the axis, and to execute the animation of the path tracking.

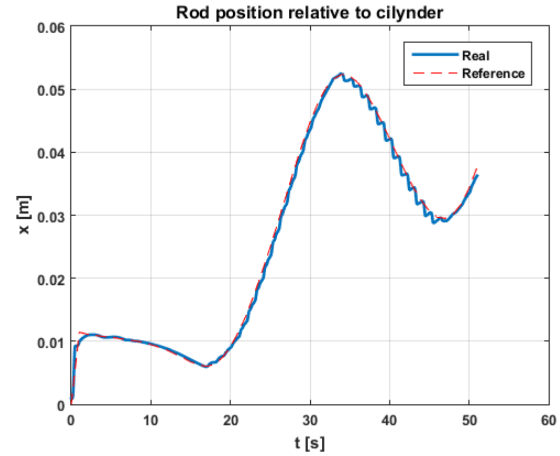

Fig. 9. Rod position of the manipulator following a 50-point trajectory generated

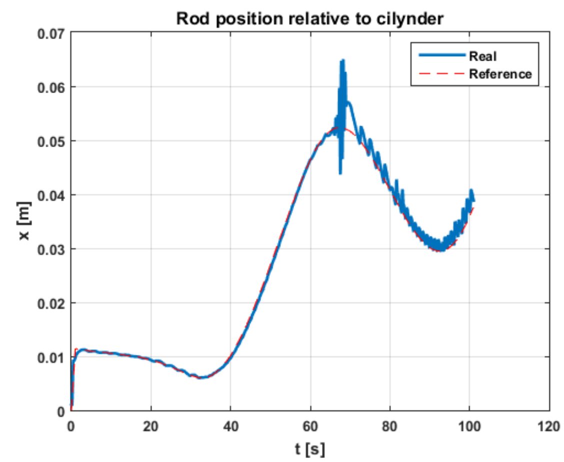

Fig. 10. Rod position of the manipulator following 100-point trajectory generated

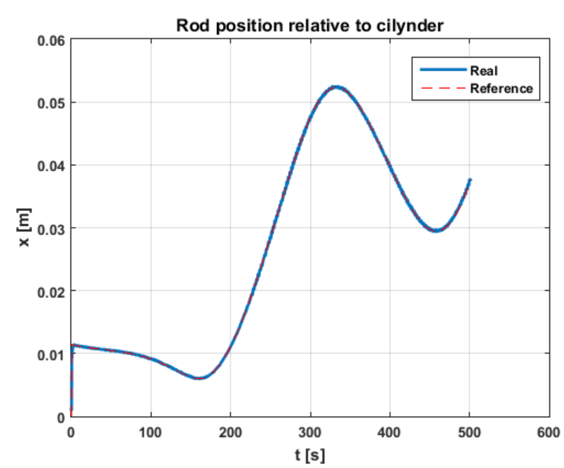

Fig. 11. Rod position of the manipulator following a 500-point trajectory generated

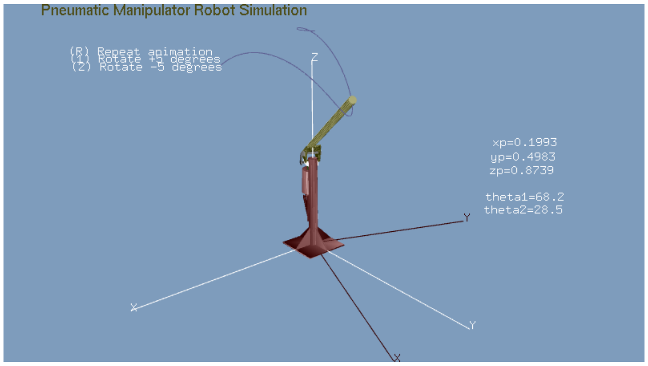

Fig. 12. Simulator Developed using OpenGL and C++

VIII. Conclusion

After applying different amounts of intermediate points resulting from the interpolation it is concluded that the higher number of points, the controller is able to bring closer the output to the set-point given which makes the system more stable and this allows to implement the path to the physical robot without damaging it. In other words, the simulator developed is a helpful tool that presents the possibility of probing the behaviour of a Pneumatic Manipulator Robot before being implemented.

This work has presented four different trajectories: a trajectory that just follows the path of the points that resulted from the interpolation, and three more trajectories where the control developed in previous works is implemented and the responses of each one of them are monitored through the graphics generated in C++ and OpenGL.

As future work, this simulator will be extended to Three-Degrees-of-Freedom which will permit to reach more points in the space. The mathematical model of the pneumatic motor will be obtained too. This will permit to obtain a better simulation that considers the dynamics of the whole system and termo-mechanical model of each actuator.

References

[1] R. R. M. Richter, C. V. Zamberlan, A. C. Valdiero, and L. A. Rasia, "Trajectory Tracking Control of a Pneumatic Actuator Special," vol. 1, no. 6, pp. 371–375, 2014.

[2] X. Wang and G. Peng, "Modeling and control for pneumatic manipulator based on dynamic neural network," in *Systems, Man and Cybernetics, 2003. IEEE International Conference on*, vol. 3, pp. 2231–2236, IEEE, 2003.

[3] J. M. Ramos, E. Gorrostieta, J. E. Vargas, and J. C. Pedraza, "New Polynomial Model of Pneumatic Cylinder Model," *Revista Internacional Información Tecnológica*, vol. 17, no. 3, 2006.

[4] J. M. Ramos-Arreguin, E. Gorrostieta-Hurtado, J. C. Pedraza-Ortega, R. J. Romero-Troncoso, M. A. Aceves, and S. Canchola, "Dynamic Behavior of a Pneumatic Manipulator with Two Degrees of Freedom," in *Robot Manipulators Trends and Development*, InTech, 2010.

[5] S. Y. Qiu, P. Yi, R. B. Yuan, H. Yang, S. Hui, and R. P. Xue, "Trajectory control for cartesian pneumatic manipulator using the terminal sliding mode control method," in *Applied Mechanics and Materials*, vol. 444, pp. 1354–1359, Trans Tech Publ, 2014.

[6] J. M. Ramos Arreguin, "Control de un Actuador Neumático," pp. 33–43, Centro de Ingeniera y Desarrollo Industrial (CIDESI), 2008.

[7] F. F. Kiyama Miramontes, "Modelado de un robot manipulador flexible," pp. 9–32, Centro de Ingeniera y Desarrollo Industrial (CIDESI), 2005.

[8] S. V. Vaseghi, *Advanced digital signal processing and noise reduction*. John Wiley & Sons, 2008.

[9] C. H. Reinsch, "Smoothing by spline functions," *Numerische mathematik*, vol. 10, no. 3, pp. 177–183, 1967.

[10] J. Ramos, E. Gorrostieta, E. Vargas, J. Pedraza, *et al.*, "Pneumatic cylinder control for a flexible manipulator robot,"

[11] M. Gamino, J. C. Pedraza, J. M. Ramos, and E. Gorrostieta, "Matlab-c++ interface for a flexible arm manipulator simulation using multi-language techniques," in *Artificial Intelligence, 2006. MICAI'06. Fifth Mexican International Conference on*, pp. 369–378, IEEE, 2006.

View publication stats