International Journal of Electrical and Computer Engineering (IJECE)
Vol. 10, No. 3, June 2020, pp. 3066~3073
ISSN: 2088-8708, DOI: 10.11591/ijece.v10i3.pp3066-3073

Trajectory reconstruction for robot programming by demonstration

Reda Hanifi Elhachemi Amar[1], Laredj Benchikh[2], Hakima Dermeche[3], Ouamri Bachir[4], Zoubir Ahmed-Foitih[5]
[1,3,5]Départment d'électronique, Université des Sciences et de la Technologie d'Oran, Algeria
[2]Université d' Evry-Val-d'Essonne, Laboratoire IBISC, France
[4]Département de Technologie, Université de Bechar, Algeria

Article Info

Article history:

Received Mar 11, 2019
Revised Jan 8, 2020
Accepted Jan 20, 2020

Keywords:

Interpolation
Motion capture
Programming by demonstration
Trajectory reconstruction

ABSTRACT

The reproduction of hand movements by a robot remains difficult and conventional learning methods do not allow us to faithfully recreate these movements because it is very difficult when the number of crossing points is very large. Programming by Demonstration gives a better opportunity for solving this problem by tracking the user's movements with a motion capture system and creating a robotic program to reproduce the performed tasks. This paper presents a Programming by Demonstration system in a trajectory level for the reproduction of hand/tool movement by a manipulator robot; this was realized by tracking the user's movement with the ArToolkit and reconstructing the trajectories by using the constrained cubic spline. The results obtained with the constrained cubic spline were compared with cubic spline interpolation. Finally the obtained trajectories have been simulated in a virtual environment on the Puma 600 robot.

Corresponding Author:

Reda Hanifi Elhachemi Amar,
Département d'électronique,
Université des Sciences et de la Technologie d'Oran,
El mnaouar BP 1505, Bir El Djirs 31000 (Oran) Algérie.
Email: reda.hanifi@univ-usto.dz

1. INTRODUCTION

Since the invention of the first robots, the reproduction of human movement is still a challenging subject in robotics. This reproduction can be divided into two categories: the first one is imitation of the human movement as it is by the robot to realize the task. References [1, 2] they imitate the human movement to realize different tasks like writing and opening doors, their approach was derived from the human functioning based on the body schema and the percept. In the work of Jie and al. [3], they focus on the pose imitation between a human and a humanoid robot and proposed a pose similarity metric based on the shared structure of the motion spaces of human and robot .The second reproduces the human movement by taking as reference only the hand (the end-effector) and neglecting how and where the other joints are positioned (shoulder, elbow, and wrist). In the work of Benchikh [4], in which he realized a system of a synchronous reproduction of the human movement, following the movement of a single point of interest.

One can notice two categories of robot programming methods, the first one is manual like the text based systems, graphical systems and the teach-pendant programming; the second one is the automatic programming such as the instruction and the observation based programming. The Figure 1 represents the categorization of these methods made by [5]. For applications such as imitation and reproduction of

human movement; conventional learning methods as text programming, graphical systems or the teach-pendant programming are not suitable. Especially when the trajectories to reproduce are complex and the number of crossing points are high. Most of the recent works for human movement reproduction use the robot observation based programming. In this category, we find what is called Programming by Demonstration or Learning by watching in other readings [1–4, 6, 7].

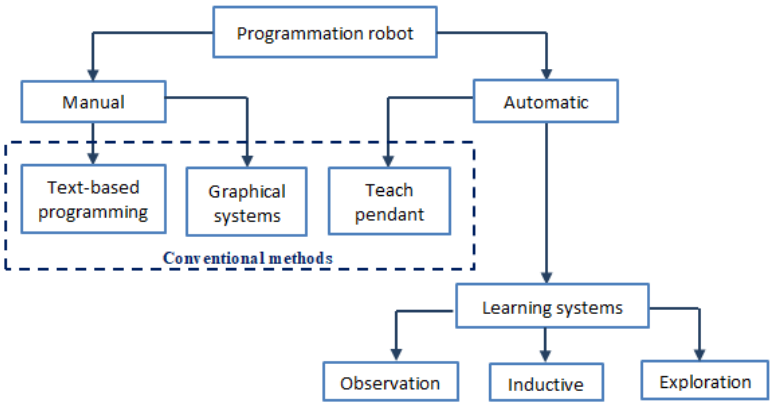

Figure 1. Robot programming methods

Programming by Demonstration (PbD) is a robot programming method based on the extraction of data directly from the visualization of the user's performance. It is a promising automatic method, which can permit to a user with little or no expertise to program robot tasks [8, 9]. In Pbd systems, the teacher performs the task while a learning interface records the movement and actions carried out during the performance. Different interfaces can be used. Kinesthetic guidance teacher moves the robot links manually and the trajectories are recorded [7]. The direct control and teleoperation interfaces are also used in Pbd systems [10]. Interfaces based on sensors (vision, magnetic and inertia) are widely used in Pbd systems. There is a variety of tracking techniques in vision systems [11–14] and it has an advantage over the other sensor-based methods, because no sensors need to be attached to the teacher. And the vision systems allow the teacher to have more natural performance without being disturbed by the material.The sequence measurements data usually goes through a scaling step through interpollations. For the time series data and from different types of interpolation techniques, polynomial interpolations are the mostly used. The cubic spline interpolation was used to reconstruct the Cartesian trajectories in [15] and Non-Uniform Rational B-Splines (NURBS) for the trajectory approximation [16]. The L1 splines for the interpolation and preservation of the trajectory [17-19].

In this paper, we propose a Pbd system in trajectory level based on visual tracking system (ArToolKit) to track the hand/tool movement. In this work we used the cubic spline interpolation to reconstruct the Cartesian trajectories. Compared to previous works; the constrained cubic spline interpolation is simple to implement and efficient. Finaly the caputed movements are reproduced by a manipulator robot in a simulated environment.

2. THE TRACKING SYSTEM

For the hand/Tool tracking, we propose to use a vision-based tracking system. These systems use image-processing methods to calculate the camera pose relative to real world objects and give the position and orientation of these objects. In our work, we use the ARToolKit for tracking the user's hand/tool. We fixed the markers on the faces of a cube (the same marker for all faces). This will allow the camera to see at least one marker and permit us to avoid occlusions. Based on the works [20, 21] we have used simple markers with a 30% border width to have a better detection of the marker and avoid at the maximum the false identifications. In the Figure 2, we can see an example of the markers that are using. The operator will perform the movements and the system will track the position of the marker for the hand/tool and gives the position (x, y, z) every sampling time (the position is calculated from the centre of the cube). The acquisition camera was set at 20 frames/s to limit marker miss identifications. The slow frame rate and some miss identifications created gapes and therefore, an interpolation of the acquired tracking data was mandatory.

Figure 2. Example of the ArToolKit markers

3. THE CONSTRAINED CUBIC SPLINE

The principle behind constrained cubic spline is to prevent overshooting and eliminating oscillation by sacrificing smoothness. In this interpolation, we replace the equality of the second order derivatives at every point by a specified first order derivatives [22, 23]. The construction of the constrained cubic spline function F_i is based on the following criteria:

- Curves are third order polynomials

$$F_i(x) = a_i x^3 + b_i x^2 + c_i x + d_i \quad (1)$$

- Curves pass through all the known points
- First order derivative, is the same for both functions on each side of a point.

$$F_i'(x) = F_{i+1}'(x) \quad (2)$$

- Boundary conditions are the same as for the natural cubic spline.

$$F_1''(x_0) = F_n''(x_n) = 0 (3)$$

- The second order derivative is replaced by a specified first order derivative at every point.

$$F_i'(x) = F_{i+1}'(x) = F'(x) \quad (4)$$

The main step becomes the calculation of the slope for each point. Naturally, we know the slope will be between the slopes of the adjacent straight lines, and it should approach zero if the slope of either line approaches zero.

$$F'(x) = 2 \Big/ \left(\frac{x_{i+1}-x_i}{y_{i+1}-y_i} + \frac{x_i-x_{i-1}}{y_i-y_{i-1}} \right) \quad (5)$$

$F'(x) = 0$, if the slop changes sign at this point.
The equation (5) is used only for the intermediate points, in the end points:

$$F_1'(x_0) = \frac{3(y_1-y_0)}{2(x_1-x_0)} - \frac{F'(x_1)}{2} \quad (6)$$

$$F_n'(x_n) = \frac{3(y_n-y_{n-1})}{2(x_n-x_{n-1})} - \frac{F'(x_{n-1})}{2} \quad (7)$$

In this interpolation, there is no necessity to solve a system of equation because the slope at each point is known. Based on the two adjacent points on each side. We can calculate every spline function; as given by (1) by using (8) to (13).

$$F_i''(x_{i-1}) = \frac{2[F_i'(x_i)+2\,F_i'(x_{i-1})]}{(x_i-x_{i-1})} + \frac{6\,(y_i-y_{i-1})}{(x_i-x_{i-1})^2} \quad (8)$$

$$F_i''(x_i) = \frac{2[2\,F_i'(x_i)+\,F_i'(x_{i-1})]}{(x_i-x_{i-1})} + \frac{6\,(y_i-y_{i-1})}{(x_i-x_{i-1})^2} \quad (9)$$

Finally, every polynomial is calculated from the following parameters:

$$a_i = \frac{F_i''(x_i)-\,F_i''(x_{i-1})}{6\,(x_i-x_{i-1})} \quad (10)$$

$$b_i = \frac{x_i F_i''(x_{i-1}) - x_{i-1} F_i''(x_i)}{2\,(x_i - x_{i-1})} \quad (11)$$

$$c_i = \frac{(y_n - y_{n-1}) - b_i(x_i^2 - x_{i-1}^2) - a_i(x_i^3 - x_{i-1}^3)}{2(x_n - x_{n-1})} \quad (12)$$

$$d_i = y_{i-1} - a_i x_{i-1}^3 - b_i x_{i-1}^2 - c_i x_{i-1} \quad (13)$$

4. THE ROBOT'S MODEL OF MOTION

In our work, we use a manipulator robot model, the Puma 600. The parameters of this robot are shown in Table 1 [24, 25]. Motion model of such a mechanism is usually described by the following matrix equation:

$$\Gamma = \mathrm{M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) + F(\dot{q})} \quad (14)$$

where:

Γ : Vector of actuator joint torque
$\mathrm{M(q)}$: Inertia matrix
$\mathrm{C(q,\dot{q})\dot{q}}$: Vector of centrifugal and Coriolis torque
$\mathrm{G(q)}$: Vector of gravitational torques
$\mathrm{F(\dot{q})}$: Vector of actuator joint friction forces
$\mathrm{q, \dot{q}, \ddot{q}}$: Are respectively, the joint angle, velocity, and acceleration vectors

To ensure the linearization of the nonlinear system described by (14) in closed loop, we introduce a linearization control system based on exacting knowledge of the robot model and its implementation. In this control system, the loop of the linearization is achieved by choosing a torque Γ applied to the robot, as follow:

$$\Gamma = M(q)\Gamma_0 + C(q,\dot{q})\dot{q} + G(q) + F(\dot{q}) \quad (15)$$

Γ_0 is an auxiliary input of the select controller. A proportional derivative control (PD) is a typical choice and it is given by the equation:

$$\Gamma_0 = \ddot{q}_d + k_v(\dot{q}_d - \dot{q}) + k_p(q_d - q) \quad (16)$$

Table 1. Parameters of the Puma 600 manipulator robot

Parameters	Values
Mass of the first body	10.521 Kg
Mass of the second body	10.236 Kg
Mass of the third body	8.767 Kg
Coefficient of viscous friction	2.52 N.m.s/rd
Coefficient of viscous friction	7 N.m.s/rd
Coefficient of viscous friction	1.75 N.m.s/rd
Coefficient of dry friction	3.6 N.m.s/rd
Coefficient of dry friction	10 N.m.s/rd
Coefficient of dry friction	2.5 N.m.s/rd
Length of the first body	0.149 m
Length of the second body	0.432 m
Length of the third body	0.431 m

By replacing $\mathrm{\ddot{q} = \Gamma_0}$ in the (16), we get:

$$\Gamma_0 = \ddot{e} + k_v\dot{e} + k_p e \quad (17)$$

$\mathrm{e = q_d - q}$: Vector of the position error
$\mathrm{\dot{e} = \dot{q}_d - \dot{q}}$: Vector of the velocity error
$\mathrm{\ddot{e} = \ddot{q}_d - \ddot{q}}$: Vector of the acceleration error
$\mathrm{q_d, \dot{q}_d, \ddot{q}_d}$: Are respectively vectors of desired position, velocity and acceleration.
$\mathrm{k_p}$, $\mathrm{k_v}$: Gain matrices of the PD controller

The error (17) is a linear differential equation of second order. Where K_p and K_v are defined positive diagonal matrices, so the closed loop system becomes linear decoupled. The values of the K_p and K_v in the simulation are:

$$K_p = \begin{pmatrix} 350 & 0 & 0 \\ 0 & 350 & 0 \\ 0 & 0 & 350 \end{pmatrix} K_v = \begin{pmatrix} 35 & 0 & 0 \\ 0 & 30 & 0 \\ 0 & 0 & 35 \end{pmatrix}$$

In the Figure 3, we can see the implementation of the computed Torque controller on the Puma robot.

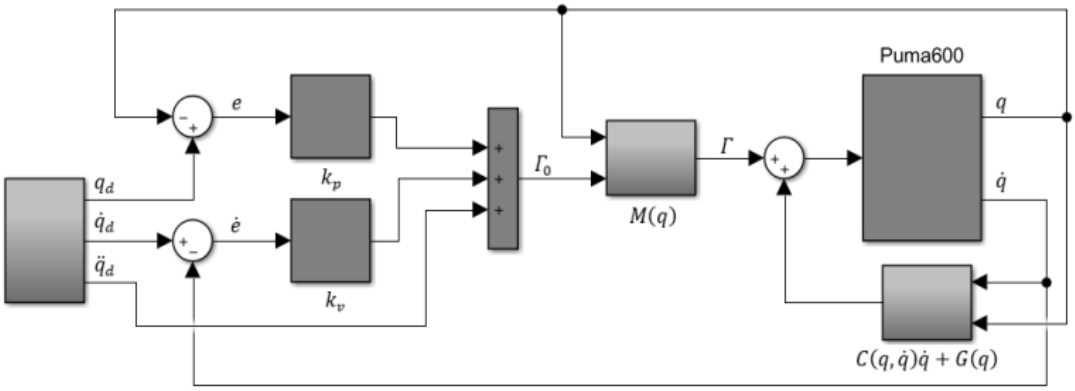

Figure 3. Implemented computed torque controller

5. SIMULATION AND RESULTS

In this section, Since the trajectories obtained from the tracking system should be interpolated, we started by testing the constrained cubic spline interpolation on different data sets. The Figures 4 and 5 show a comparison between the constrained cubic in blue line and the conventional cubic spline interpolation in red line, the red circles are the interpolated data points. In the first tests, we used both of the interpolation methods to reconstruct handwriting trajectory made at almost the same speed. We have chosen to reconstruct the letter M using both of the cubic spine interpolation and the constrained cubic spline because this letter contains sudden directional changes which will help us to see the behaviour of the used interpolation method. In the Figure 4 (a) and (b), we can see respectively the results of interpolation for the X-axis and Y-axis coordinates and in the Figure 4 (c) we can see the reconstructed trajectory. For the results in the Figure 4 (b) we see that both of the methods gave the same results this due to the fact that there was no large data variations but in the Figure 4 (a) we can see an oscillation and an overshooting of the reconstructed data. The Figure 4 (c) we have the result of the reconstructed trajectory where we can clearly see that constrained cubic spline gave the better results and the oscillation of the cubic spine interpolation affected the obtained trajectory. For the Figure 5, we have used the following data:

(a): x=[0,1,2,3,4,5,6,7,8,9,10] ; y=[0,0,0,0,0,1,1,1,1,1,1]

(b): x=[0,1,2,3,4,5,6,7,8,9,10,11,12,13,14] ; y=[0,0,0,0,0,1,1,0,0,0,-1,0,0,0,0]

In Figure 5, it is noted that the constrained cubic spline does not oscillate after large amplitude variation and reduces the overshooting comparing to the cubic spline. This fact makes the constrained cubic spline more stable and has less oscillations which makes it better for the reconstruction of the trajectories. The interpolation programs were made with Matlab 2017a on a computer with the following configuration: I7 3770 3.4 Ghz and 8 Gb Ram. Table 2 shows interpolation execution time according to interpolated point number. One can notice two behaviours. The first is for number of points below 200: the constrained cubic spline is faster than the cubic spline since it does not solve a system of equations. The second case is when the number of interpolated points exceeds 200 the cubic spline overcomes the constrained cubic spline. In the Figure 6 one can see a handwriting trajectory reconstructed with the constrained cubic spline interpolation. Figure 7 shows a simulation of Puma 600 robot executing the same trajectory. The advantage of the simulation process is reconstructed path visualization. The proposed system depends on the teacher, and do not generate collision-free trajectories.

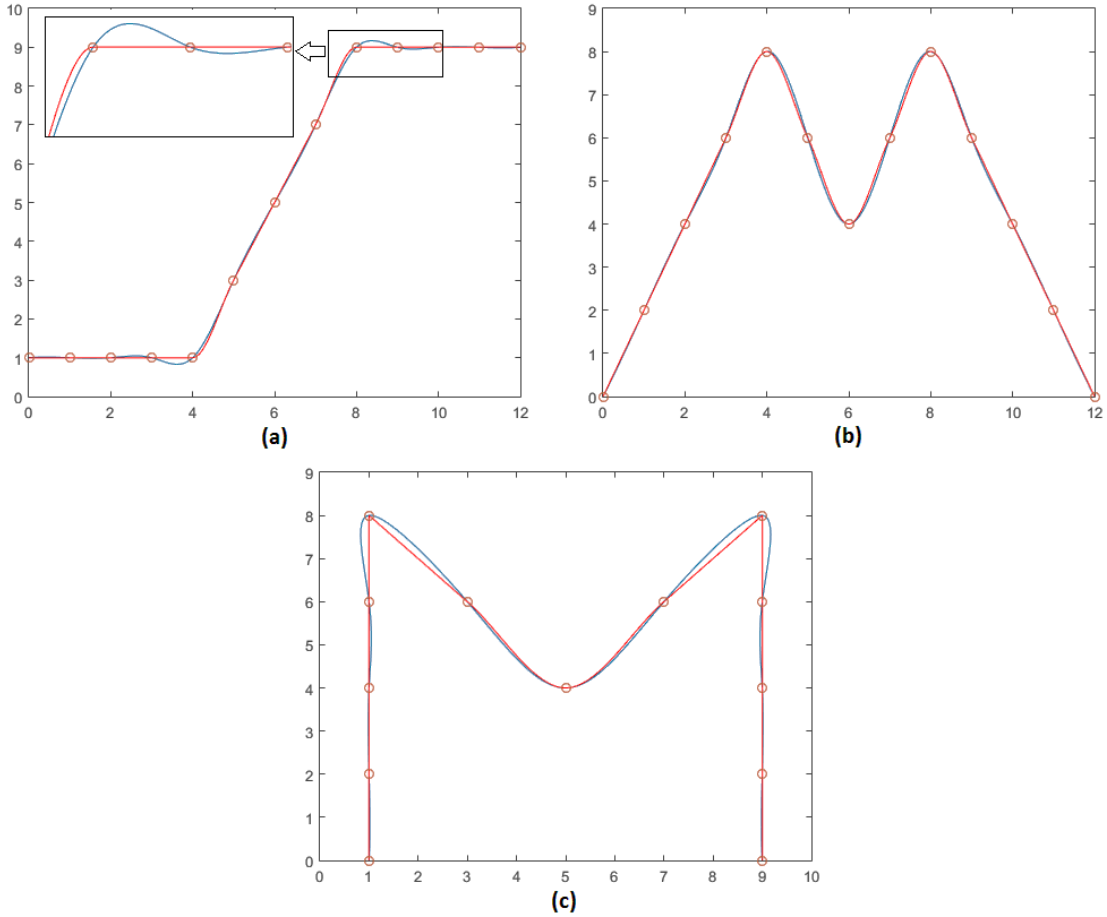

Figure 4. Trajectory reconstruction using cubic spline and constrained cubic spline interpolations

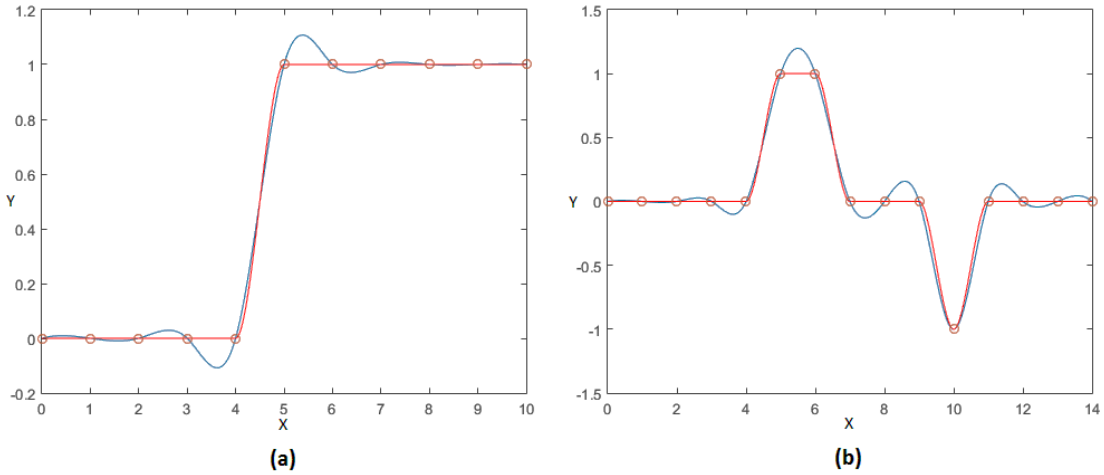

Figure 5. Interpolation of different data sets

Table 2. Parameters of the Puma 600 manipulator robot

Number of points	Constrained spline (ms)	Cubic spline (ms)
25	1.383	2.332
50	1.900	2.481
100	2.502	3.121
200	4.084	3.390

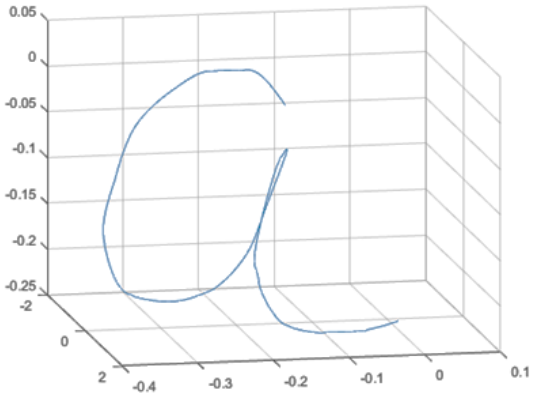

Figure 6. 3D Trajectory reconstructed by the constrained cubic spline

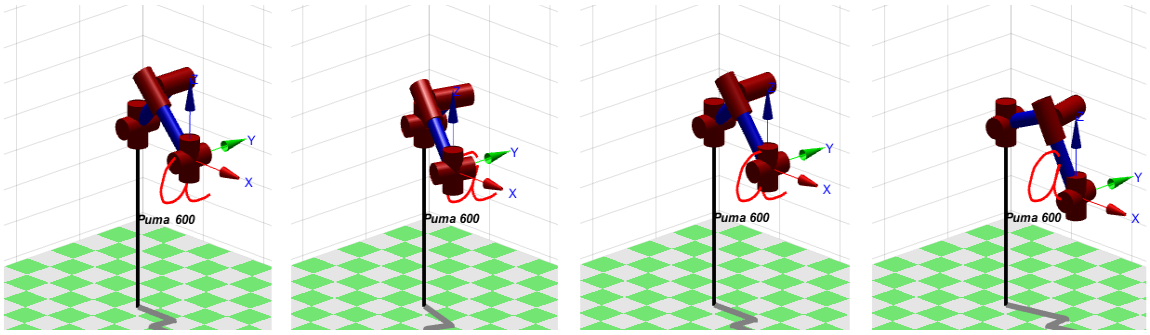

Figure 7. Simulation of robot trajectory reproduction

6. CONCLUSION

This paper presented a robot programming by demonstration system in a trajectory level. The system is based on human movement reproduction by following a single point of interest (the hand/tool). We used ARToolkit as a visual tracking system. The slow frame rate and the miss identifications of the markers was creating gapes in the trajectories. The natural cubic spline and constrained cubic spline interpolations were used to correct these weaknesses and to reconstruct the trajectory. A comparison was made between those both interpolation methods for 3D trajectories reconstruction, 1D data sets and as per execution time. Tests show that constrained cubic pline interpolation overcomes the cubic spline interpolation. The constrained cubic spline interpolation shows good results and the fact that it generates less oscillation and prevents overshooting makes it suitable for the trajectory reconstruction. In the future works this it will be used in a real time trajectory reconstruction.

REFERENCES

[1] Acosta-calderon, C.A., and H. Hu, "Robot imitation: Body schema and body percept," *Applied Bionics and Biomechanics*, vol. 2, no. 3, pp. 131-148, 2005.
[2] Acosta-calderon, C.A., and H. Hu, "Robot imitation from human body movements," *The 3rd International Symposium on Imitation in Animals and Artifacts*, pp. 1-9, UK, 2005.
[3] Lei, J., Song, M., Li, Z.N. and Chen, C., "Whole-body humanoid robot imitation with pose similarity evaluation," *Signal Processing*, vol. 108, pp. 136-146, 2015.
[4] Benchikh, L., "Method for training a robot or the like, and device for implementing said method," U.S. Patent Application No 12/812 792, 2011.
[5] Biggs, G., and MacDonald, B., "A survey of robot programming systems," *Proceedings of Australasian Conference on Robotics and Automation*, Brisbane, Australia, pp. 1-102, 003.
[6] Kuniyoshi, M. Inaba, and H. Inoue, "Learning by watching: Extracting reusable task knowledge from visual observation of human performance," *IEEE transactions on robotics and automation*, vol 10, no. 2, pp. 799-822, 1994.

[7] M. Calinon, S., Billard, A., "A probabilistic programming by demonstration framework handling constraints in joint space and task space," *International Conference on Intelligent Robots and Systems IROS*, Nice, France, pp. 367-372, 2008.
[8] Aleotti, J., Caselli, S., and Reggiani, M., "Leveraging on a virtual environment for robot programming by demonstration," *Robotics and Autonomous Systems*, vol. 47, pp. 153-161, 2004.
[9] Herrero, Héctor, *et al*., "Skill based robot programming: Assembly, vision and Workspace Monitoring skill interaction," *Neurocomputing*, pp. 61-70, 2017.
[10] Shimizu, M., Yoon, W., Kitagaki, K., "Experimental validation of task skill transfer approach using a humanoid robot," *International Symposium on Assembly and Manufacturing ISAM'07*, pp. 141-146, 2007.
[11] Kurien, M., Kim, M.K., Kopsida, M., and Brilakis, I., "Real-time simulation of construction workers using combined human body and hand tracking for robotic construction worker system," *Automation in Construction*, vol. 86, pp. 125-137, 2018.
[12] Valentini, P.P., "Natural interface for interactive virtual assembly in augmented reality using Leap Motion Controller," *International Journal on Interactive Design and Manufacturing (IJIDeM)*, vol. 22, no. 4, pp. 1-9, 2018.
[13] Khairudin, M., *et al*., "Control of a movable robot head using vision-based object tracking," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 9, no. 4, pp. 2503-2512, 2019.
[14] Shi, Qing, *et al*., "Design and implementation of an omnidirectional vision system for robot perception," *Mechatronics*, vol. 41, pp. 58-66, 2017.
[15] Calinon, Sylvain, and Aude Billard, "Stochastic gesture production and recognition model for a humanoid robot," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 3, pp. 2769-2774, 2004.
[16] Aleotti, J. and Caselli, S., "Robust trajectory learning and approximation for robot programming by demonstration," *Robotics and Autonomous Systems*, vol. 54, no. 5, pp. 409-413, 2006.
[17] Lavery, J.E., "Univariate cubic Lp splines and shape-preserving, multiscale interpolation by univariate cubic L1 splines," *Computer Aided Geometric Design*, vol. 17, no. 4, pp. 319-336, 2000.
[18] Lavery, J.E., "Shape-preserving, first-derivative-based parametric and nonparametric cubic L1 spline curves," *Computer Aided Geometric Design*, vol. 23, no. 3, pp. 276-296, 2006.
[19] Hernoux, F., Bearee, R., Gajny, L., *et al*., "Leap Motion pour la capture de mouvement 3D par spline L1," *Journées du Groupe de* Travail *en Modélisation Géométrique*, Marseille, France, 2013.
[20] Khan, D., Ullah, S., Rabbi, I., "Factors affecting the design and tracking of ARToolKit markers," *Computer Standards & Interfaces*, vol. 41, pp. 56-66, 2015.
[21] Rabbi, Ihsan, and Sehat Ullah, "Extending the tracking distance of fiducial markers for large indoor augmented reality applications," *Advances in Electrical and Computer Engineering*, vol. 15, no. 2, pp. 59-64, 2015.
[22] C. J. Kruger, "Constrained cubic spline interpolation," *Chemical Engineering Applications*, 2003.
[23] Kokes, J. and Nghien, N.B., "Using constrained cubic spline instead of natural cubic spline to eliminate overshoot and undershoot in Hilbert Huang Transform," *The 13th International Carpathian Control Conference*, High Tatras, Slovakia, pp. 300-306, 2012.
[24] Ouamri, B., and Ahmed-Foitih Z., "Adaptive neuro-fuzzy inference system based control of puma 600 robot manipulator," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 2, no. 1, pp. 90-97, 2011.
[25] Ouamri, B., and Ahmed-Foitih Z., "Computed Torque Control of a Puma 600 Robot by using Fuzzy Logic," *International Review of Automatic Control*, vol. 4, no. 2, pp. 248-252, 2011.