sensors

MDPI

Article

Dynamic Model and Inverse Kinematic Identification of a 3-DOF Manipulator Using RLSPSO

Josias Batista [1,*], Darielson Souza [1], Laurinda dos Reis [1], Antônio Barbosa [2] and Rui Araújo [3,4]

1 Robotics, Automation and Control Research Group (GPAR), Federal University of Ceará, Fortaleza-CE 60455-760, Brazil; darielson@dee.ufc.br (D.S.); laurinda@dee.ufc.br (L.d.R.)
2 Federal Institute of Ceará–IFCE Campus Maracanaú, Maracanaú-CE 61925-315, Brazil; barbosa@dee.ufc.br
3 Institute of Systems and Robotics (ISR-UC), University of Coimbra, Pólo II, PT-3030-290 Coimbra, Portugal; rui@isr.uc.pt
4 Department of Electrical and Computer Engineering (DEEC-UC), University of Coimbra, Pólo II, PT-3030-290 Coimbra, Portugal
* Correspondence: josiasgb@dee.ufc.br; Tel.: +55-85-3366-9581

Received: 26 November 2019; Accepted: 8 January 2020; Published: 11 January 2020

Abstract: This paper presents the identification of the inverse kinematics of a cylindrical manipulator using identification techniques of Least Squares (LS), Recursive Least Square (RLS), and a dynamic parameter identification algorithm based on Particle Swarm Optimization (PSO) with search space defined by RLS (RLSPSO). A helical trajectory in the cartesian space is used as input. The dynamic model is found through the Lagrange equation and the motion equations, which are used to calculate the torque values of each joint. The torques are calculated from the values of the inverse kinematics, identified by each algorithm and from the manipulator joint speeds and accelerations. The results obtained for the trajectories, speeds, accelerations, and torques of each joint are compared for each algorithm. The computational costs as well as the Multi-Correlation Coefficient (R^2) are computed. The results demonstrated that the identification accuracy of RLSPSO is better than that of LS and PSO. This paper brings an improvement in RLS because it is a method with high complexity, so the proposed method (hybrid) aims to improve the computational cost and the results of the classic RLS.

Keywords: least Squares; recursive least squares; inverse kinematics; dynamic model; improved RLS with PSO

1. Introduction

The diffusion of several systems in industrial environments has led over the years to the fact that several identification methods were developed to monitor and control various models of plants such as mobile robots or manipulator robots giving them the ability to operate accurately and efficiency [1]. These robots must perform tasks with great perfection and safety. For this purpose, they need appropriate kinematic and dynamic models that represent the real manipulator.

Nonlinearity and time variation are characteristics of some systems and to model and control them one often wants to use linear models. One of the difficulties of some processes is when operating conditions change thus giving a valuable choice of model partitions during the upgrade. Some methodologies of estimation of model parameters were proposed as the recursive least squares method (RLS). According to the work presented in [2] the RLS method updates a vector of parameters and has a lower computational

cost than the non-recursive least squares method. The work of Hafezi et al. [3] addresses two recursive identification methods with ARMA noise with applications in identification of bilinear systems were proposed the generalized extended least squares (GELS) and recursive maximum likelihood (RML) methods.

The quality of data obtained by the system can significantly influence the identification of a manipulator model. Generally some data may have poor quality thus interfering the identification process. Its include insufficient input excitation and low signal to noise ratio. Moreover, brief knowledge of the system model can help in the implementation of the control project. Physical systems may include nonlinear and stochastic behaviors and present data outliers. These systems can interferes in the performance of identification algorithms. The Robust Algorithms Approach has relevance when it comes to systems with outliers according to [4]. The appearance of outliers may also compromise the performance of techniques when there is insertion of Gaussian distribution noise in the data samples as presented in [5].

In [6], the LS is used to solve the problem of four degrees of freedom ship manoeuvring. It is used to perform identification modelling with the full-scale trial data. A new transformed multi-innovation least squares (TMILS) algorithm it was used. Ma, J. et al. in [7] proposes a new approach for identifying a Wiener-based model in which the system can be interpreted by an exogenous autoregressive model coupled with least squares and a support vector machine (LSSVM). The parameters were select by adaptive particle swarm optimization (APSO) that obtain better performance in relation of classical PSO.

The work of [8] presents an identification of dynamic parameters of the lower extremity exoskeleton using the Particle Swarm Optimization (PSO) metaheuristic in the search space defined by Recursive Least Square (RLS), thus making it a hybrid method. During the definition in the PSO search space, the hybrid method not only avoids the convergence of parental identification to the local minimum, but also has very accurate results. Particle Swarm Optimization (PSO)-based identification methods with some variations have been shown in [9]. The identification method is applied to a robotic manipulator where the estimated gaps are used to predict joint torques.

The prototype of a 5-DOF hybrid manipulator was developed in research conducted by [10]. In this research the mechanical structure, the kinematics, the dynamics and the control system were presented. In the results the kinematics and dynamics simulations of this manipulator are presented and tests of accuracy and repeatability of the manipulator path and position. In paper [11] to solve the problem of inverse kinematics (IK), reinforcement learning (RL) was used, designed to balance the lower body of a humanoid 3D robot that has 12 degrees of freedom (DOF). The lower body trajectories are learned by RL which are IK solutions that are converted into positions for NAO robot joints. This reduces the learning dimension because RL-integrated IK eliminates the need to use whole humanoid robot (HR) states. The purpose of the work presented in [12] was to create an ABB IRB120 industrial robot representation for simulating and analyzing dynamics and kinematics of the industrial robots by using MapleSim. In addition the paper presents how linear and nonlinear models of the robot can be obtained and makes available them to public.

Dynamic modeling and kinematics analysis of parallel robot was presented in [13]. In research of [14] its refer to inverse kinematics and a new method to identify the parameters of the dynamic model of the manipulator that was the identification of dynamic parameters based on Particle Swarm Optimization (PSO). The dynamic model taking into account the friction of the manipulator joints is determined and the dynamic parameters are defined as a linear form of the identified parameter. PSO is used to minimize the optimum manipulator trajectory parameters.

In [15] used the torque exerted by each joint when performing periodic excited Fourier trajectories. The parameters were divided into a linear and nonlinear part and used the least square linear parameter estimation (LLS) and the double swarm-based particle swarm optimization (DPso) to calculate the linear and nonlinear parts, respectively. The configurations used were simpler and can identify the dynamic parameters, the friction coefficients of the joints. Already in the paper of [16] techniques were

used to identify the dynamic parameters in an industrial manipulator robot with 5 degrees of freedom. The parameters were identified using LS, Adaline artificial neural networks, Hopfield artificial neural networks and the extended Kalman Filter. To solve manipulator robots identification problems [17] presented an intelligent approach with PSO that was called the elitist learning strategy (ELS) and the proportional-integral-derivative controller (PID) hybridized approach (ELPIDSO). The parameter identification of robots manipulator was performed to evaluate the performance of the approach. The ELPIDSO was superior to the LS method, genetic algorithm (GA) and SPSO in the estimation of the parameters of the robot manipulators kinetic models.

Based on the review done above this paper aims to identify the inverse kinematics of a cylindrical manipulator using LS, RLS, and RLS with PSO (RLSPSO). The positions of each manipulator joint obtained using the inverse kinematics model calculated from a trajectory in the Cartesian space are used as input data. The model of the manipulator dynamics is calculated using Lagrangean mechanics and the equations of torques of each manipulator joint are presented. The results show the trajectories, speeds, accelerations, and torques of each joint, real and estimated. The computational cost of each algorithm used in the identification as well as the Multi-Correlation Coefficient (R^2) of each manipulator joint is presented. A discussion of the results is carried out and the advantages and disadvantages of each method are presented. The inverse kinematics identification can be used to generate real-time manipulator trajectories and to generate collision-free trajectories in static and dynamic obstacle environments as well as being used as an approximation of the inverse kinematics model.

Contributions

This paper improves the results of classic RLS in relation to computational cost of the proposed method. It is used a particle swarm algorithm with an objective function resulting from the RLS covariance matrix. Each method will be presented a quantitative analysis of the results in order to verify the issue of reducing the complexity of calculating the covariance matrices of the algorithms.

The system to be identified is a three phase induction motor driven cylindrical robotic manipulator. The importance of the proposal is due to the issue of different points of operations when the manipulator is driven. The proposed method still works with non-Gassian disturbances in the system inputs, testing its robustness.

This paper is organized as follows. Section 2 presents the characteristics of the manipulator. Section 3 presents the models of direct kinematics and inverse kinematics, the Jacobian model and the dynamic model. Section 4 presents the formulations on how the LS, RLS, and RLSPSO algorithms were used. Section 5 presents the results of the implemented algorithms and the discussions about the work. Finally Section 6 presents the discussions and conclusions are mentioned in Section 7.

2. Cylindrical Manipulator

This section presents the characteristics of the manipulator used in this research, as well as the kinematic and dynamic modelling. The kinematics will be modelled using Denavit–Hartenberg (DH) notation, and the dynamic model is based using the Lagrangean Mechanics (LM) modelling method formulations. A cylindrical robotic manipulator that is driven by three phase induction motors was used in this work. As can be seen in Figure 1 the first joint moves around the main axis of the structure (rotational motion), the second and third joints have linear (prismatic) movements, which defines as a RPP (Rotational-Prismatic-Prismatic).

The three-phase induction motors used are of the squirrel cage type. The power of the motors was chosen so that it was possible to move each joint of the manipulator.

2.1. Characteristics of the Manipulator

This subsection presents some physical characteristics of the manipulator under study. To calculate the torques of each joint it was necessary to find the masses of the robot and the simplest form was through a modelling software Solid Edge© that was able to provide this information [18]. In Figure 2 presents the computational modelling of the manipulator.

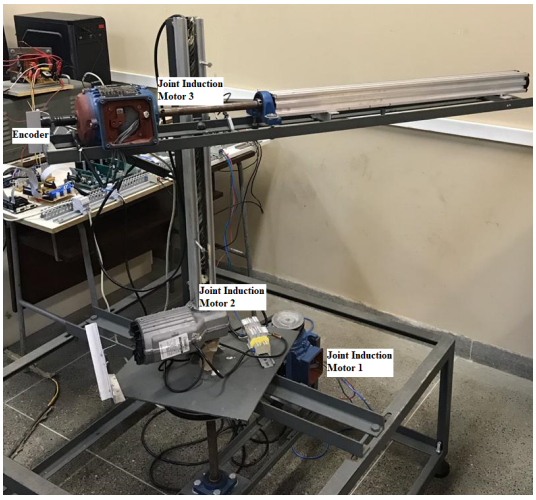

Figure 1. Setup of cylindrical manipulator.

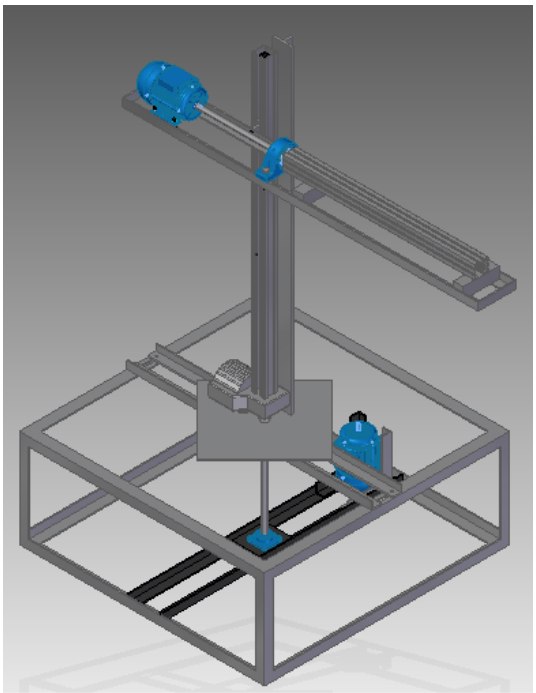

Figure 2. Structure of the cylindrical manipulator—Software Solid Edge©.

Through the software of computer modelling were found the main physical properties of the manipulator as the dimensions, masses and moments of inertia of each joint. Figure 3 shows a modelling software screen with the physical properties of the manipulator only for joint 2.

The values of the masses (m) and lengths (l) of each link of the manipulator are shown in Table 1.

Table 1. Values of masses (m) and lengths (l) of each link.

Link	m (kg)	l (m)
1 (θ_1)	36.367405	0.050
2 (d_2)	12.632222	0.790
3 (d_3)	23.735183	0.900

The information presented in Table 1 will be used for calculating the joints torques 1, 2 and 3.

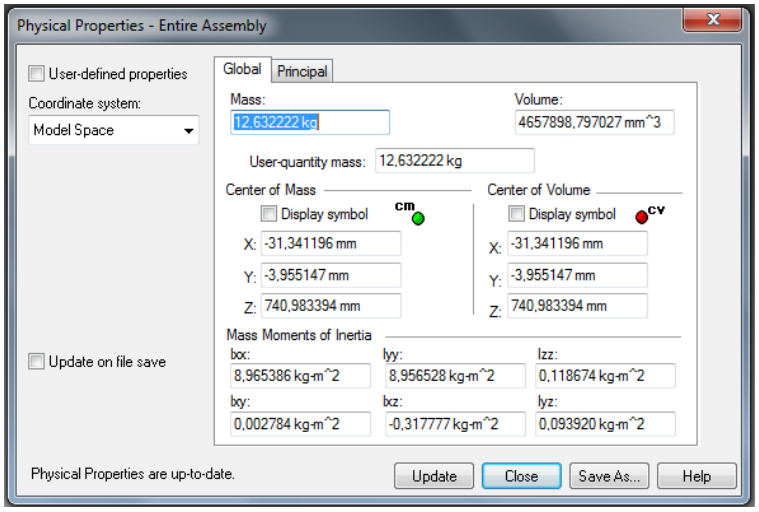

Figure 3. Physical properties of the manipulator joint 2—Software Solid Edge©.

3. Kinematic and Dynamic Modelling of the Robotic Manipulator

The kinematics exposes the relative motion of the reference systems, as the structure moves by relating reference systems to the various portions of the structure [19,20].

3.1. Forward Kinematics

The cylindrical manipulator used in this paper is shown in Figure 4 (Kinematic model of an RPP robotic arm). The kinematic configuration of the manipulator according to the Denavit–Hartenberg (D–H) convention in [21] is established and presented in Table 2. The DH parameters are: twist angles α_i, link lengths a_i, joint displacements q_i, and link offsets d_i, where $i = 1, \ldots, n$. Obviously, the manipulator has a revolute degree-of-freedom (DOF), and two prismatic movements, it an RPP robotic manipulator.

Table 2. DH Parameters of an RPP manipulator.

Link	a_i	α_i	d_i	θ_i
1	0	0	0.245	θ_1
2	0.11	$-\pi/2$	d_2	0
3	0	0	d_3	0

Using the DH conversion to the parameters shown in Table 2 are the corresponding matrices A and T [22] given by:

$$A_1 = \begin{bmatrix} cos(\theta_1) & -sin(\theta_1) & 0 & 0 \\ sin(\theta_1) & cos(\theta_1) & 0 & 0 \\ 0 & 0 & 1 & 0.245 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (1)$$

$$A_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (2)$$

$$A_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (3)$$

$$T_3^0 = A_1 A_2 A_3 = \begin{bmatrix} cos(\theta_1) & 0 & -sin(\theta_1) & -sin(\theta_1)(d_3 + 0.35) \\ sin(\theta_1) & 0 & cos(\theta_1) & cos(\theta_1)(d_3 + 0.35) \\ 0 & -1 & 0 & 0.245 + d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (4)$$

Any final position (end-effector) of the manipulator can be found in the Cartesian space from the coordinates in the joint space, as noted in Equation (5):

$$\begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} = \begin{bmatrix} -sin(\theta_1)(d_3 + 0.35) \\ cos(\theta_1)(d_3 + 0.35) \\ 0.245 + d_2 \end{bmatrix} \quad (5)$$

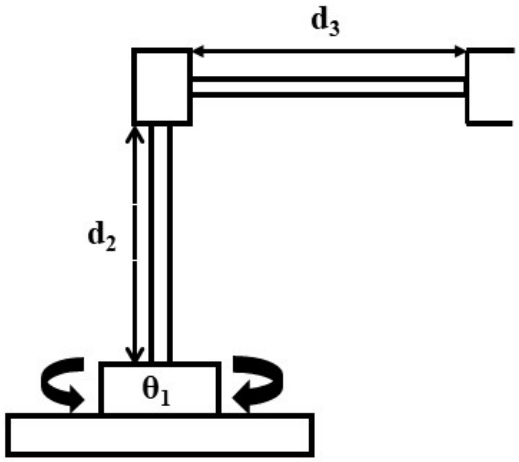

Figure 4. Cylindrical manipulator.

3.2. Inverse Kinematics

Inverse kinematics defines the configuration, the values of the joint variables, that the manipulator must have for the position and orientation of a chosen point. One method to solve the problem of inverse

kinematics of a manipulator is by the geometric method [22]. Applying this method it can be found that θ_1 is defined by:

$$\theta_1 = Atan2(P_x, P_y) \tag{6}$$

The parameter of link 2 is prismatic, as can be seen in Figure 1, and d_2 is in the same axis z_1 given by:

$$d_2 = P_z - 0.245 \tag{7}$$

In the case of the third parameter d_3, it will move in the plane formed by x and y and can be determined by:

$$d_3 = (\sqrt{P_x^2 + P_y^2}) - 0.35 \tag{8}$$

Equations (6)–(8) are the solutions for the cylindrical manipulator inverse kinematics problem and will be used to perform position control and manipulator path and trajectory generation.

3.3. Jacobian

The relationship between the Cartesian (end-effector) and joint speeds of a manipulator is given by the Jacobian [22]. As can be seen in Figure 4 a cylindrical manipulator has the following variables of the joints, $q = (\theta_1,\ d_2,\ d_3)$.

Since the manipulator has a revolute joint and two prismatic joints, i.e., three joints, the Jacobian matrix in this case is of dimensions 6×3 and is of the form:

$$J(q) = \begin{bmatrix} z_0 \times (o_3 - o_0) & z_1 & z_2 \\ z_0 & o_0 & o_0 \end{bmatrix} \tag{9}$$

where we have $z_o = [0\ 0\ 1]^T = z_1$ and $o_0 = [0\ 0\ 0]^T$; z_2 an o_3 are given by:

$$z_2 = \begin{bmatrix} sen(\theta_1) \\ cos(\theta_1) \\ 0 \end{bmatrix} \tag{10}$$

$$o_3 = \begin{bmatrix} -cos(\theta_1)(l_3 + 0.35) \\ -sen(\theta_1)(l_3 + 0.35) \\ 0 \end{bmatrix} \tag{11}$$

Substituting each matrix and performing the necessary operations, considering $d_3 = l_3 + 0.35$, we have the Jacobian matrix 6×3, given by:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} -cos(\theta_1)d_3 & 0 & sen(\theta_1) \\ -sen(\theta_1)d_3 & 0 & cos(\theta_1) \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \dot{\theta}_1 \\ \dot{d}_2 \\ \dot{d}_3 \end{bmatrix} \tag{12}$$

This reveals that it is impossible to perform a rotation around the x_0 and y_0 axes. The Jacobian in relation to the linear speed of the end-effector can be obtained considering only the first three matrix lines that is

$$J = \begin{bmatrix} -cos(\theta_1)(d_3 + 0.35) & 0 & sen(\theta_1) \\ -sen(\theta_1)(d_3 + 0.35) & 0 & cos(\theta_1) \\ 0 & 1 & 0 \end{bmatrix} \tag{13}$$

3.4. Dynamic Modelling

The dynamics of the manipulator displays the between the position-speed-acceleration-torque relationship of the joints. Therefore, the dynamic modelling of an industrial robot aims to know the relationship between the movement of the robot and the forces applied to it [19,23]. The model of the manipulator dynamics can be obtained using the Euler–Lagrange formulation, [24,25]. The model equation is of the form shown below:

$$L(q, \dot{q}) = K(q, \dot{q}) - P(q), \tag{14}$$

where L is the Lagrangian; K is the kinetic energy and P is the potential energy (see Appendix A.1). For the cylindrical manipulator under study was calculated the kinetic and potential energy and then applied the formulation based on the Lagrangian [23].

From the Jacobian Equation (12) one can determine the speeds and the equations of the kinetic energy, after performing some operations and mathematical transformations. Potential energy equations could be obtained from classical mechanics [23]. From the Lagrange Equation (14) the system motion equations given by:

$$\frac{d}{dt}\left[\frac{\partial L}{\partial \dot{q}}\right] - \frac{\partial L}{\partial q} = \tau \tag{15}$$

where $\tau \in \Re^n$, are the torques applied to the joints. Thus, considering the kinetic energy of the manipulator the dynamic equation of the manipulator can be written in simplified form as:

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) = \tau, \tag{16}$$

where, $q, \dot{q}, \ddot{q} \in \Re^n$ indicate the positions, speeds and accelerations joint's, respectively; $M(q)$ is the inertial matrix; $C \in \Re^n$ is the matrix that describes the centripetal and Coriolis forces and $G\frac{\partial g}{\partial q} \in \Re^n$ is the gravity matrix.

Applying the Lagrange formulation Equation (14) and from the equation of motion Equation (15), performing the calculations of the partial derivatives, it can be obtained the torque equations of each joint of the manipulator [22] which are shown below. The first equation of motion describing the torque of joint 1 is:

$$\begin{aligned} \tau_1 = &-[(4m_1 sen\theta_1 - 4m_2 cos\theta_1)d_3 + I_3]\ddot{\theta}_1 \\ &+[(m_1 + m_2)(sen\theta_1 cos\theta_1)d_3]\ddot{d}_3 \\ &+[(m_1 sen\theta_1 - m_2 cos\theta_1)d_3]\dot{\theta}_1^2 \\ &-[m_1 cos\theta_1 + m_2 sen\theta_1]\dot{d}_3^2 \\ &-[(m_1 + m_2)(sen\theta_1 cos\theta_1)d_3]\dot{\theta}_1\dot{d}_3 \end{aligned} \tag{17}$$

Likewise by solving the partial derivatives for the second joint of the manipulator we have the torque of the joint 2 found from the equation of motion of the joint 2:

$$\tau_2 = m_3\ddot{d}_2 + g(m_2 + m_3) \tag{18}$$

Following the same idea the partial derivatives for the third joint of the manipulator, the equation of motion describing the torque of joint 3 is given by:

$$\begin{aligned} \tau_3 = &[m_1 sen\theta_1 cos\theta_1]\ddot{\theta}_1 \\ &-[2(m_1 sen\theta_1 + m_2 cos\theta_1)]\ddot{d}_3 \\ &+[2d_3(m_1 sen\theta_1 - m_2 cos\theta_1)]\dot{\theta}_1^2 \\ &-[(m_1 + m_2)(sen\theta_1 cos\theta_1)]\dot{\theta}_1\dot{d}_3 \end{aligned} \tag{19}$$

The terms in the torque equations, $\ddot{\theta}_1$, $\ddot{d}_2$, $\ddot{d}_3$ are related to the angular accelerations of the links, the terms $\dot{\theta}_1^2$, $\dot{d}_2^2$, $\dot{d}_3^2$ are the centripetal accelerations, and the terms $\dot{\theta}_1\dot{d}_2$, $\dot{\theta}_1\dot{d}_3$, $\dot{d}_2\dot{d}_3$ are the Coriolis accelerations [23].

Equations (17)–(19) were used to calculate the torques of each manipulator joint and the torque values are presented in the following results section.

4. Identification Methods

The practice of identification algorithms is interesting for many applications such as supervision, diagnostics, filtering, prediction, signal processing, detection, and variant parameter tracking for adaptive control. In this section, we presente the methods of identification of the inverse kinematics using Least Squares (LS), Recursive Least Squares (RLS), and Recursive Least Squares with Particle Swarm Optimization (RLSPSO) according to the literature [26–28].

4.1. Least Squares (LS)

The LS method is one of the most well-known and is used in the most diverse areas [29].

Consider the rigid-body dynamics Equation (16). Let us excite it with a control input τ and collect the resulting q, $\dot{q}$, and $\ddot{q}$. Assume we have collected θ samples of each element of τ, q, $\dot{q}$ and $\ddot{q}$ corresponding to time instants $t_1, t_2, \ldots, t_k$, [30]:

$$\Phi\theta = y(q, \dot{q}, \ddot{q}), \tag{20}$$

where

$$\Phi = \begin{bmatrix} \Phi_{[0]} \\ \Phi_{[1]} \\ \vdots \\ \Phi_{[n-1]} \end{bmatrix}, \; y = \begin{bmatrix} y_{[0]}(q, \dot{q}, \ddot{q}) \\ y_{[1]}(q, \dot{q}, \ddot{q}) \\ \vdots \\ y_{[n-1]}(q, \dot{q}, \ddot{q}) \end{bmatrix},$$

and n is the total number of sampled data points. The columns of the matrix Φ should be linearly independent for LS to accurately approximate the parameters. The estimation process can be improved using the total proximity of least squares which also considers uncertainties in the regression matrix.

The input $\tau(t)$ it is appropriate to stimulate robot dynamics. With this stimulus the vector of identifiable parameters θ can be estimated from the least-squares (LS) sense using some generalized inverse of the information matrix Φ,

$$\hat{\theta} = \Phi^* y, \tag{21}$$

where "*" denotes a generalized matrix inverse.

Equation (21) is a solution by the LS method, which is equal to the solution obtained using the pseudo-inverse matrix. This solution will be used to perform the inverse of kinematic identification with LS.

4.2. Recursive Least Squares (RLS)

The LS method is based on set of measures and is unsuitable for real-time application. It is necessary to build, update, have available a model of the system during on-line operation [26,28].

A dynamic model taken over a set of data generates constraints that can be presented by a matrix equation which can be written in the regressor form as,

$$y = \phi\hat{\theta} + \xi, \tag{22}$$

where ϕ is called the matrix of regressors and ξ is the residue. The RLS solution for $\theta_{[k]}$ takes the following form [31]:

$$\theta_{[k]} = \theta_{[k-1]} + L_{[k]}\left[y_{[k]} - \phi_{[k]}^{T}\theta_{[k-1]}\right], \tag{23}$$

where

$$L_{[k]} = \frac{P_{[k-1]}\phi_{[k]}}{\lambda_{[k]} + \phi_{[k]}^{T}P_{[k-1]}\phi_{[k]}}, \tag{24}$$

and

$$P_{[k]} = \frac{1}{\lambda_{[k]}}\left[P_{[k-1]} - \frac{P_{[k-1]}\phi_{[k]}\phi_{[k]}^{T}P_{[k-1]}}{\lambda_{[k]} + \phi_{[k]}^{T}P_{[k-1]}\phi_{[k]}}\right]. \tag{25}$$

4.3. Recursive Least Squares with Particle Swarm Optimization (RLSPSO)

Particle Swarm Optimization (PSO) is a metaheuristic inspired on social behavior proposed by [32]. The main objective of the algorithm is to search in a given space, through the data permutation of the particles, consequently each particle will be a trajectory in the search space. PSO excels at other algorithms in aspects such as easy implementation and fast convergence. As with other search algorithms, PSO may have particles trapped in local minima locations [33].

The PSO metaheuristic has particles similar to a set of birds that seek the best way to fly taking into account the position and speed of each particle. A convergence curve is used during the execution of the algorithm. Each particle will have its resulting goal depending also on the behaviour of the general population of particles [33]. The position at time t is updated by $x_i(t)$ and at future time $t+1$ will be given in (26).

$$x_i(t+1) = x_i(t) + v_i(t+1), \tag{26}$$

where $v_i(t)$ is the speed [34]. Each particle will present a cognitive component which will be a relation of the distance between itself and the best (optimal solution) besides the social component that is the understanding of the set on the existence of a given particle. For this problem, we used the global PSO (Global best PSO) in which the particle speed is updated by:

$$\begin{aligned} v_{ij}(t+1) = v_{ij}(t) &+ c_1 r_1(t)[y_{ij}(t) - x_{ij}(t)] \\ &+ c_2 r_2(t)[\hat{y}_{ij}(t) - \hat{x}_{ij}(t)], \end{aligned} \tag{27}$$

for $v_{ij}(t)$ being the speed of the particle, in a given dimension at time t. Again, c_1 and c_2 are the acceleration parameters. The best particle information is given by $\hat{y}_{ij}$ and y_{ij} is the best position from the beginning [34].

Unlike other evolutionary computing techniques in PSO each particle is associated with a speed. Particles fly through the search space with speeds that are dynamically adjusted according to their historical behavior. Finally, the particles have a tendency to traverse the best areas for research to a solution during the search process [34].

For the PSO algorithm (see Algorithm 1) the following values of the elements were used:

- Number of particles = 60 particles;
- Cognitive and social parameters (learning rates): c_1 = 3.1 and c_2 = 3.9;
- Iterations = 10 iterations;
- Inertia factor (w) = 1.0;
- Initial population generation = used a rand in a generic equation that is restricted to the interval [0.01, 50].

Algorithm 1: PSO Algorithm

1: initiate the swarm of particles and define P Matrix ;
2: **repeat**
3: **for** $i = 1$ **to** m
4: **if** $f(x_i) < f(p_i)$ **then**
5: $p_i = x_i$;
6: **if** $f(x_i) < f(g)$ **then**
7: $g = x_i$;
8: **end if**
9: **end if**
10: **for** $j = 1$ **to** n
11: $r_1 = rand()$, $r_2 = rand()$;
12: $v_{ij} = wv_{ij} + c_1 r_1(p_i - x_{ij}) + c_2 r_2(g_j - x_{ij})$;
13: **end for**
14: $x_i = x_i + v_i$;
15: **end for**
16: **to** satisfy the stopping criterion
17: Optimal value of P convariance matrix

The stopping criterion used in the PSO algorithm was the number of iterations of the algorithm. The PSO metaheuristic has as its mission to minimize the objective function given in Equation (28), with the number of iterations equal to 10, and the algorithm was executed 10 times for obtaining the best result.

$$J_{min} = 1 - mean(R^2_{J_i}),\ i = 1, 2, 3 \tag{28}$$

where $R^2_{J_i}$ is the multiple correlation coefficient applied to joints 1, 2 and 3.

5. Results

In this section, we present the results of the identification of the inverse kinematics of the manipulator. Comparisons of actual and estimated values are presented. The results of speeds, accelerations and torques are also presented for each trajectory generated from the identification of LS, RLS, and RLSPSO.

To perform the manipulator trajectory, an algorithm describing a trajectory shown in Figure 5 was developed where the displacement of each manipulator joint in the cartesian space is performed. This

trajectory provides the final manipulator positions collected from the encoders of each manipulator joint that were used as inputs to the algorithms that perform inverse kinematic identification.

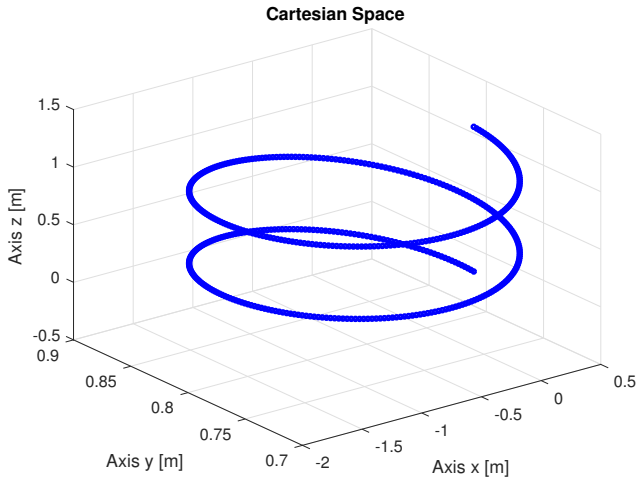

Figure 5. Trajectory executed by the manipulator in Cartesian space.

In this study, we will consider both noise-free case and noised case in the measurements of position, speed, acceleration. Measurement noises are all considered as the white noise with standard deviation $\sigma = 0.05$ and the signal to noise ratio is 10 dB.

It is noteworthy that the initial states of the estimation of each algorithm for each joint were initialized with values equal to zero.

5.1. Noise-Free results

The present results are data without noise, to make a final analysis. The results of the LS, RLS, and RLSPSO methods will be discussed in this section.

5.1.1. Results of LS

The following are the figures with trajectories, speeds, accelerations and torques of joints 1, 2, and 3 of the manipulator. The trajectories in the space of the joints were obtained from the resolution of the inverse kinematics and the identification using LS using as input the points of the trajectory in the Cartesian space shown in Figure 5.

Figure 6 shows the trajectories (displacement) of the joint 1, 2, and 3 to perform the path in Cartesian space shown in Figure 5.

Figure 7 shows the results of errors in identification the trajectories of each joint using the least squares method.

Figures 8 and 9 shows the speeds and accelerations of joints 1, 2, and 3 to perform the trajectories shown in Figure 6, respectively.

The joint torques were obtained from the dynamic model, Equations (17)–(19) of the manipulator and are shown in Figure 10. Torques were calculated by taking the trajectories, speeds and accelerations shown in Figures 6, 8 and 9.

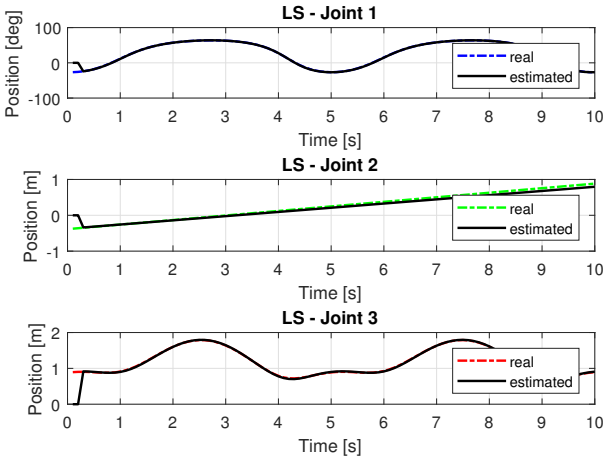

Figure 6. Trajectory in the joint space identified with LS.

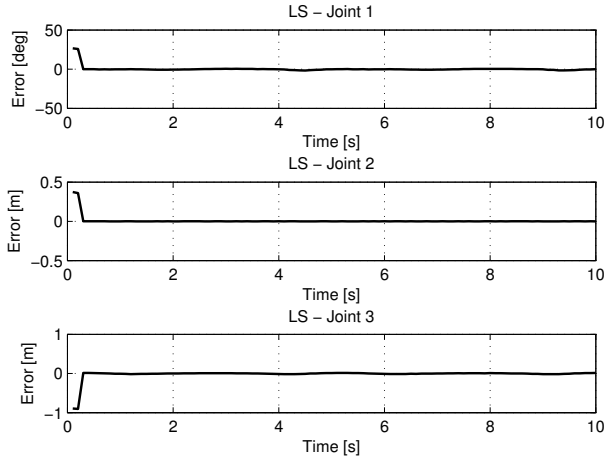

Figure 7. Error of Trajectories identifications with LS.

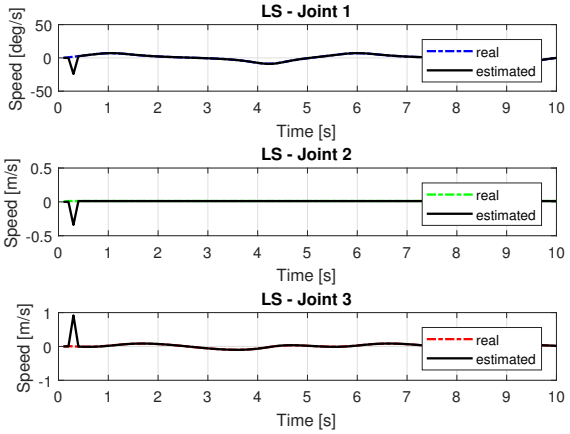

Figure 8. Speed of joints with LS.

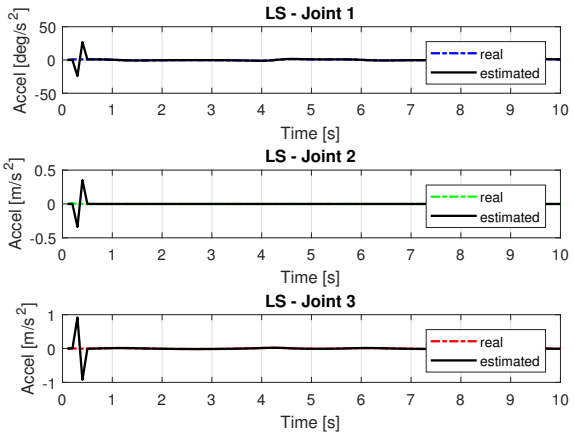

Figure 9. Accelerations of joints with LS.

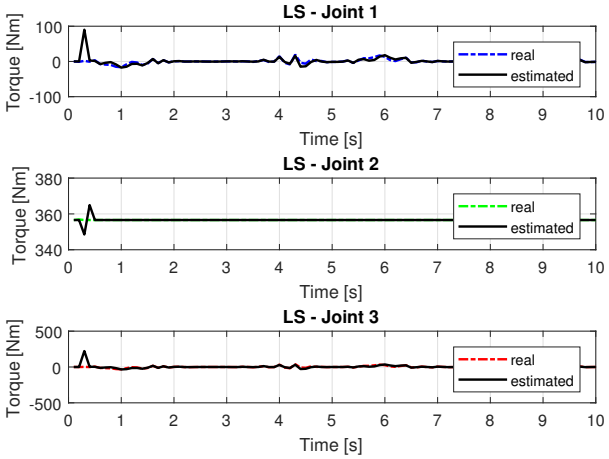

Figure 10. Torque of joints with LS.

5.1.2. Results of RLS

The following are the paths, speeds, accelerations and torques of joints 1, 2, and 3 of the manipulator. The trajectories in the space of the joints were obtained from the resolution of the inverse kinematics, and the identification using RLS. For this case, we used an exhaustive search to find the best $P_{[k]}$ weights in order to get better results. The matrix result is:

$$P_{1_{[k]}} = P_{2_{[k]}} = P_{3_{[k]}} = \begin{bmatrix} 4.999 & 0 & 0 & 0 \\ 0 & 4.999 & 0 & 0 \\ 0 & 0 & 4.999 & 0 \\ 0 & 0 & 0 & 4.999 \end{bmatrix}$$

Figure 11 show the trajectories (displacement) of the seals 1, 2, and 3 to perform the path in Cartesian space shown in Figure 5.

Figure 12 shows the results of errors in identification the trajectories of each joint using the recursive least square method.

Figures 13 and 14 shows the speeds and accelerations of joints 1, 2, and 3 to perform the trajectories shown in Figure 11.

The joint torques were obtained from the dynamic model shown in Equations (17)–(19) of the manipulator presented in Figure 15. Torques were calculated by taking the trajectories, speeds, and accelerations shown in Figures 11, 13, and 14, respectively.

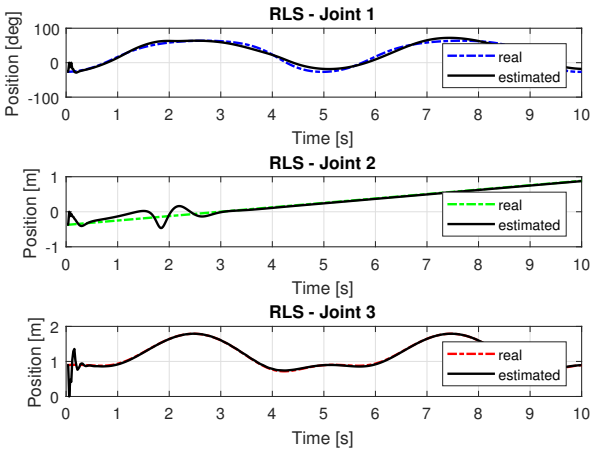

Figure 11. Trajectory in the joint space identified with RLS.

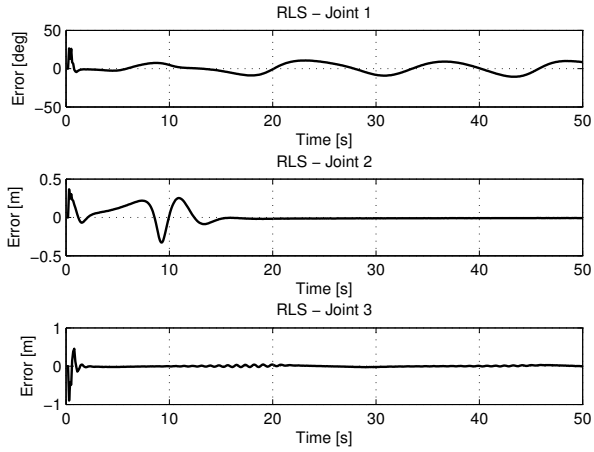

Figure 12. Error of Trajectories identifications with RLS.

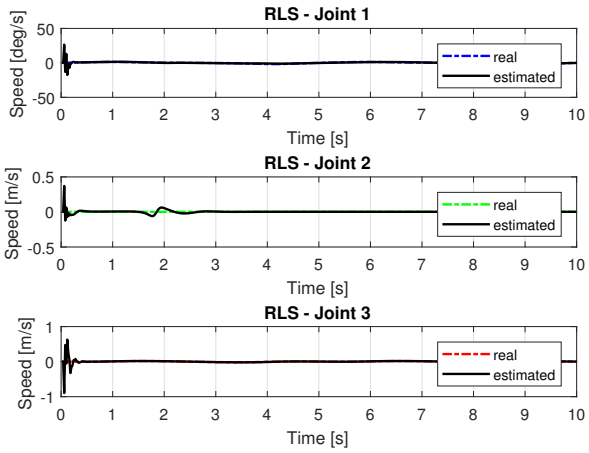

Figure 13. Speed of joints with RLS.

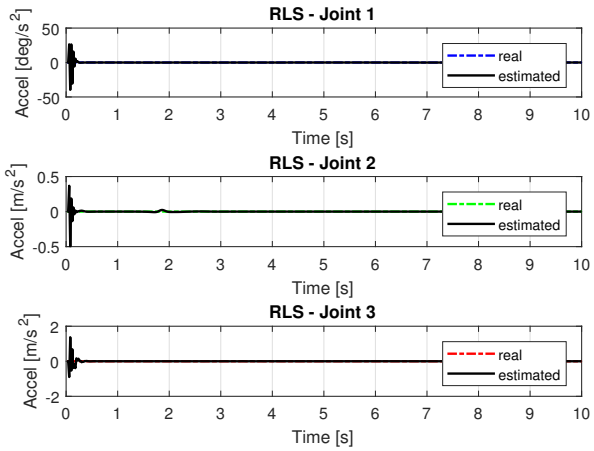

Figure 14. Accelerations of joints with RLS.

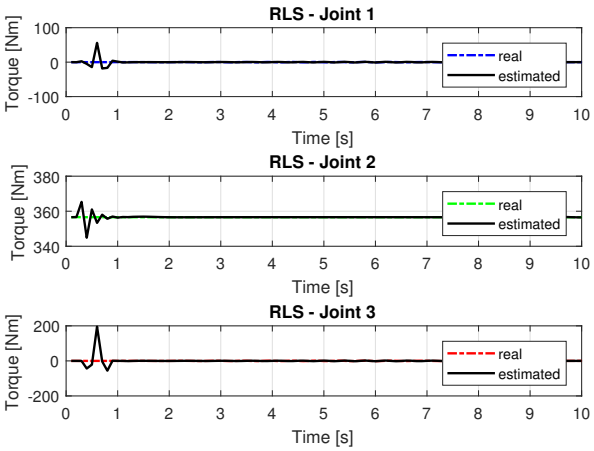

Figure 15. Torque of joints with RLS.

5.1.3. Results of RLSPSO

The following are the paths, speeds, accelerations and torques of joints 1, 2, and 3 of the manipulator. The trajectories in the space of the joints were obtained from the resolution of the inverse kinematics and the identification using RLSPSO. The PSO was used to perform an optimization to find the weights of the matrix $P_{[k]}$ of the RLS. The matrix $P_{[k]}$ found by the PSO was

$$P_{1_{[k]}} = P_{2_{[k]}} = P_{3_{[k]}} = \begin{bmatrix} 10.7521 & 0 & 0 & 0 \\ 0 & 39.3081 & 0 & 0 \\ 0 & 0 & 26.7325 & 0 \\ 0 & 0 & 0 & 36.9065 \end{bmatrix}$$

Figure 16 shows the iterations and the cost in which can be observed that PSO algorithm converges to six iterations with the best cost.

The stopping criterion of the algorithm is number of iterations.

Figure 17 show the trajectories (displacement) of the seals 1, 2, and 3 to perform the path in Cartesian space shown in Figure 5.

Figure 18 shows the results of errors with trajectories identifications using the recursive least square with PSO method.

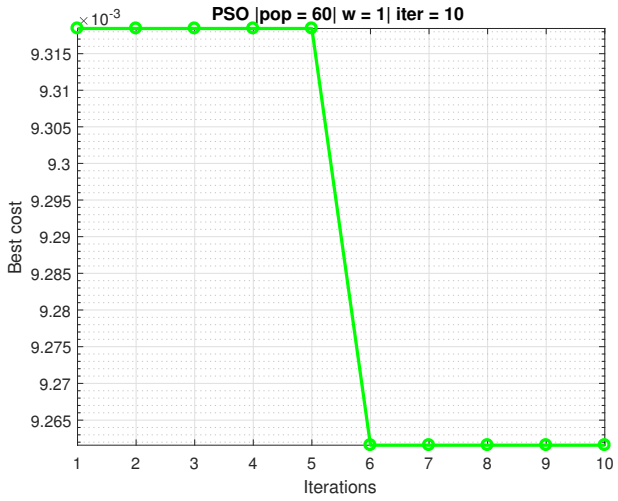

Figure 16. PSO graph converging to 60 particles and 10 iterations.

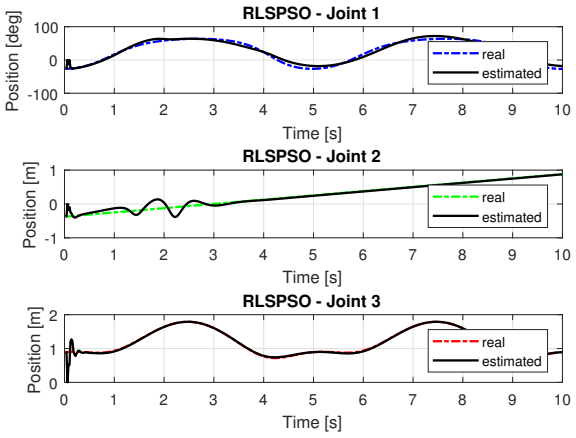

Figure 17. Trajectory in the joint space identified with RLSPSO.

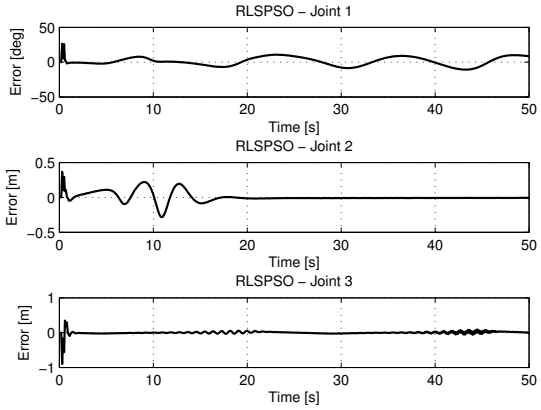

Figure 18. Error of Trajectories identifications with RLSPSO.

Figures 19 and 20 shows the speeds and accelerations of joints 1, 2, and 3 to perform the trajectories shown in Figure 17.

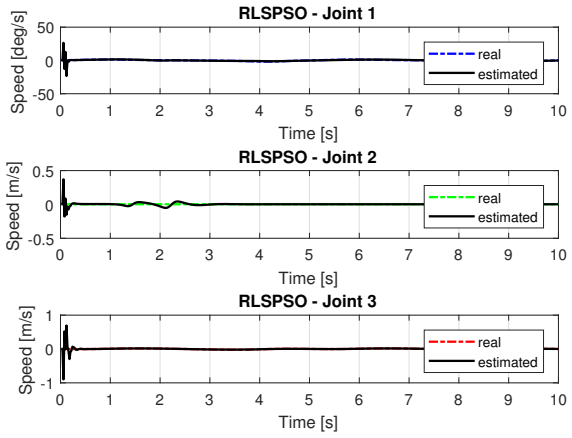

Figure 19. Speed of joints with RLSPSO.

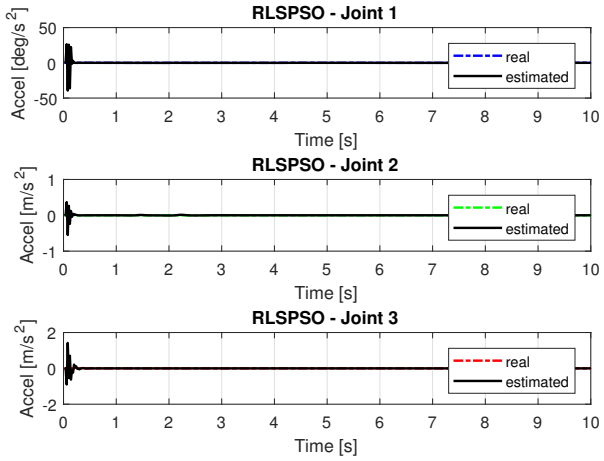

Figure 20. Accelerations of joints with RLSPSO.

The joint torques were obtained from the dynamic model in Equations (17)–(19) of the manipulator and are shown in Figure 21. Torques were calculated by taking the trajectories, speeds and accelerations shown in Figures 17, 19 and 20.

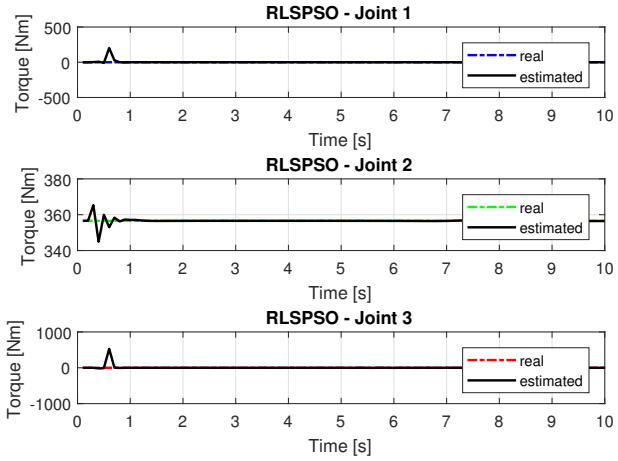

Figure 21. Torque of joints with RLSPSO.

5.2. Results with Noise

Non-Gaussian noise data were used to verify the method identifications with the input data, as well as the appearance of outlies making the estimates difficult.

5.2.1. LS with Noise

The recursive least squares method obtained a high computational effort trying to find the best solution with noisy inputs. Figure 22 show the trajectories (displacement) of the seals 1, 2 and 3 to perform the path in Cartesian space. The speeds and accelerations.

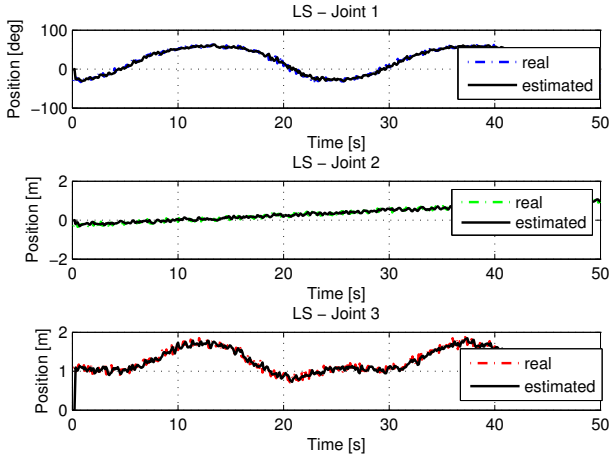

Figure 22. Trajectory in the joint space identified with LS with noise.

Figure 23 shows the results of errors in identification the trajectories of each joint using the least square with method with noise.

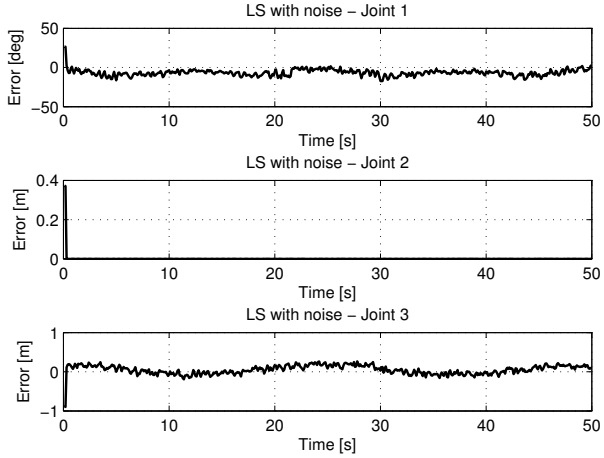

Figure 23. Error of Trajectories identifications with LS with noise.

Figures 24 and 25 shows the the speeds and accelerations of joints 1, 2, and 3 to perform the trajectories shown in Figure 22.

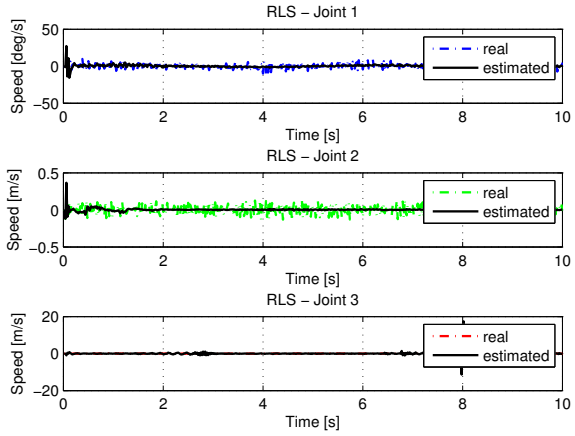

Figure 24. Speed of joints with LS with noise.

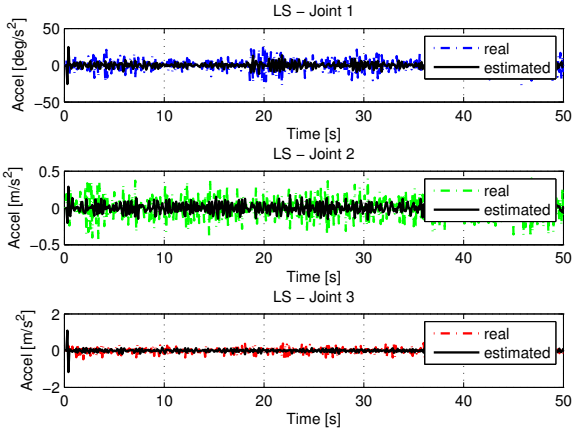

Figure 25. Accelerations of joints with LS with noise.

Figure 26 shows the results with torque noises using the least square method.

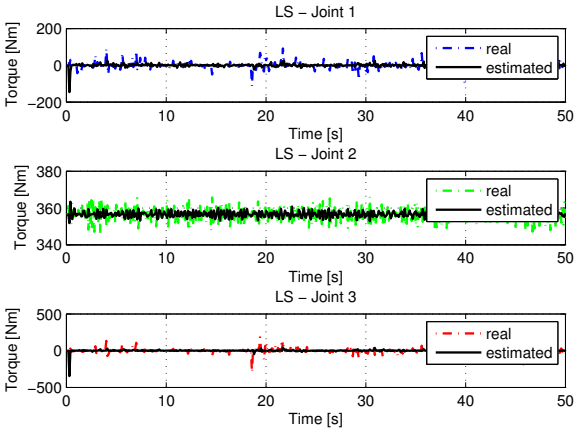

Figure 26. Torque of joints with LS with noise.

5.2.2. RLS with Noise

The use of noise in the RLS obtained a higher computational effort than no noise where the covariance matrix found was:

$$P_{1_{[k]}} = P_{2_{[k]}} = P_{3_{[k]}} = \begin{bmatrix} 8.999 & 0 & 0 & 0 \\ 0 & 8.999 & 0 & 0 \\ 0 & 0 & 8.999 & 0 \\ 0 & 0 & 0 & 8.999 \end{bmatrix}$$

Figure 27 present the results using the noisy RLS method. Input data is the trajectory values shown in the Figure 5.

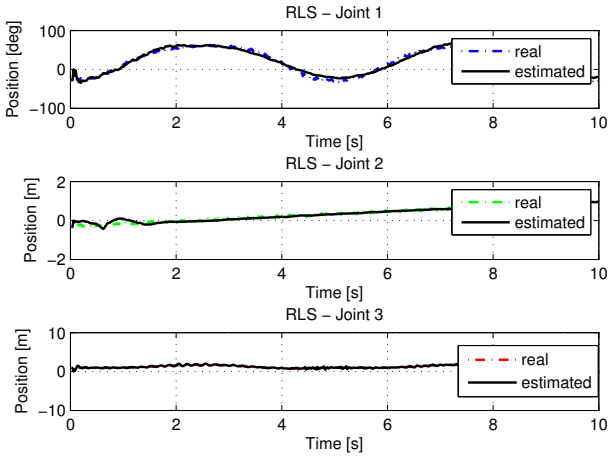

Figure 27. Trajectory in the joint space identified with RLS with noise.

Figure 28 shows the results of errors with trajectories identifications using the recursive least square with method with noise.

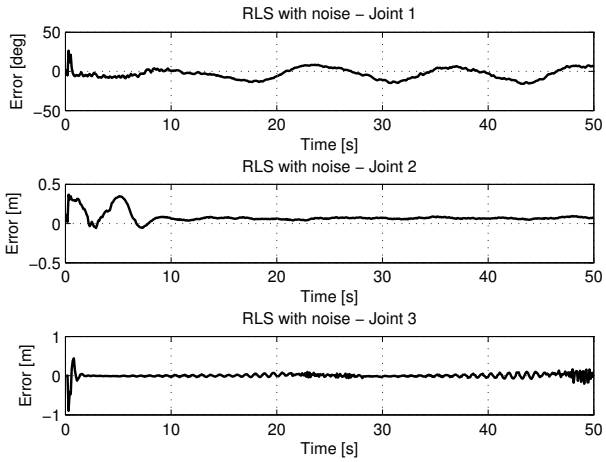

Figure 28. Error of Trajectories identifications with RLS with noise.

Figures 29 and 30 shows the the speeds and accelerations of joints 1, 2, and 3 to perform the trajectories shown in Figure 27.

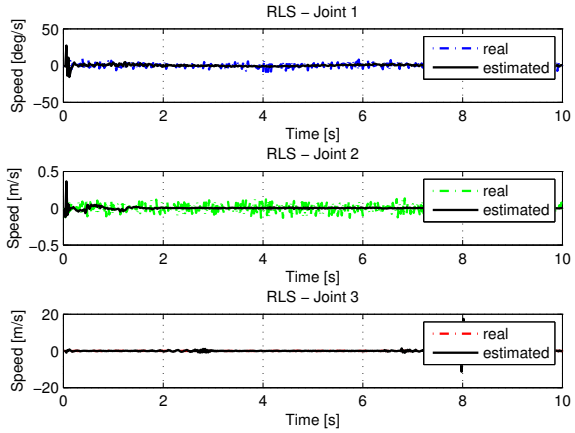

Figure 29. Speed of joints with RLS with noise.

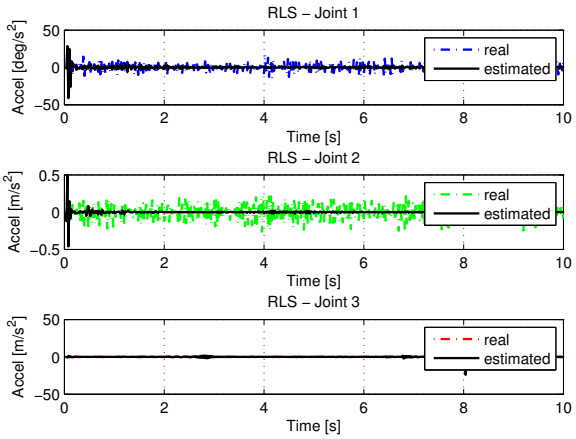

Figure 30. Accelerations of joints with RLS with noise.

The Figure 31 shows the results with torque noises using the recursive least square method.

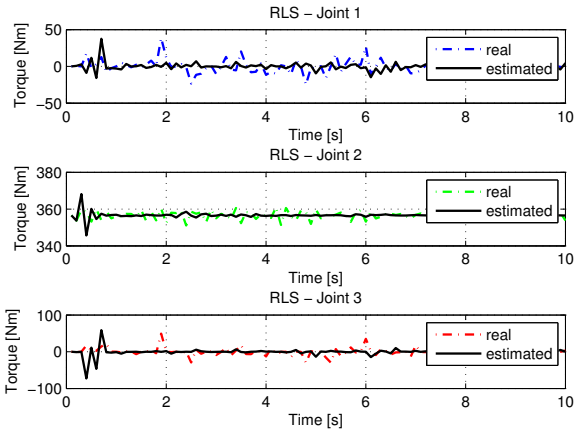

Figure 31. Torque of joints with RLS with noise.

5.2.3. RLSPSO with Noise

This section will present the results of the RLSPSO with noise. The matrix $P_{[k]}$ found by the PSO for RLS with noise was:

$$P_{1_{[k]}} = P_{2_{[k]}} = P_{3_{[k]}} = \begin{bmatrix} 21.7933 & 0 & 0 & 0 \\ 0 & 36.4084 & 0 & 0 \\ 0 & 0 & 36.3718 & 0 \\ 0 & 0 & 0 & 36.1714 \end{bmatrix}$$

In Figure 32 shows the iterations and the cost where it can be observed that the PSO algorithm converges to 45 iterations with the best cost. The stopping criterion of the algorithm is number of iterations.

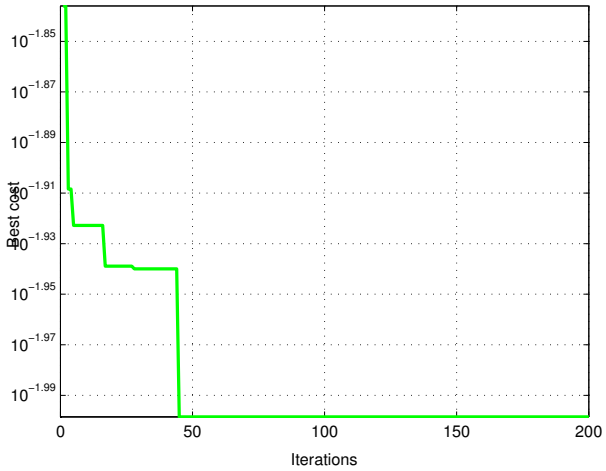

Figure 32. PSO graph converging to 200 particles and 45 iterations.

Figure 33 show the trajectories (displacement) of the seals 1, 2 and 3 with noisy entries, can be seen below:

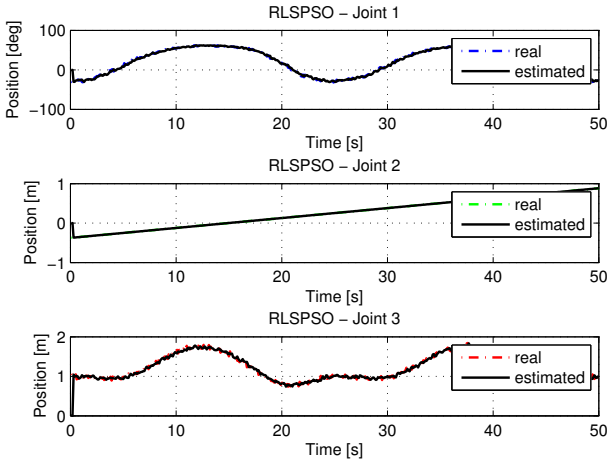

Figure 33. Trajectory in the joint space identified with RLSPSO with noise.

Figure 34 shows the results of errors with trajectories identifications using the recursive least square method with PSO with noise.

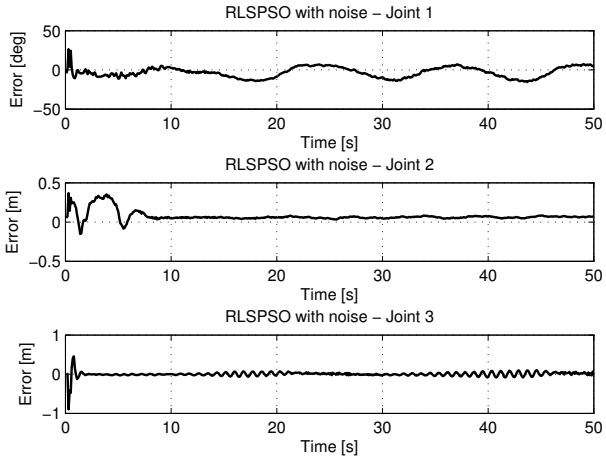

Figure 34. Error of Trajectories identifications with RLSPSO with noise.

Figures 35 and 36 shows the the speeds and accelerations of joints 1, 2, and 3 to perform the trajectories shown in Figure 33.

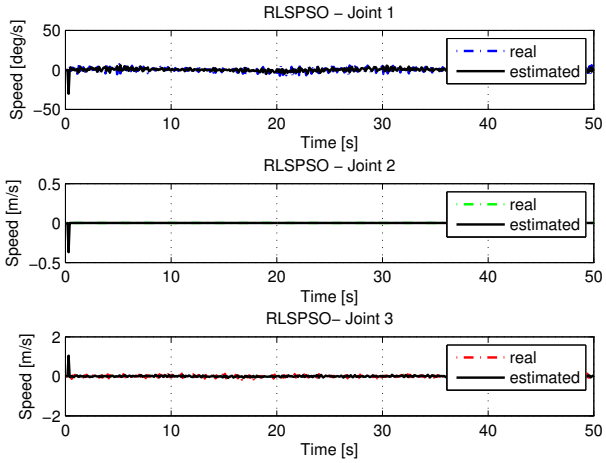

Figure 35. Speed of joints with RLSPSO with noise.

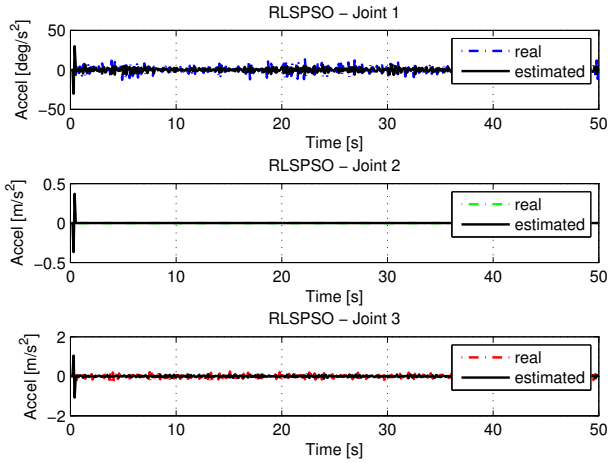

Figure 36. Accelerations of joints with RLSPSO with noise.

The torques obtained with the noisy inputs can be seen in Figure 37.

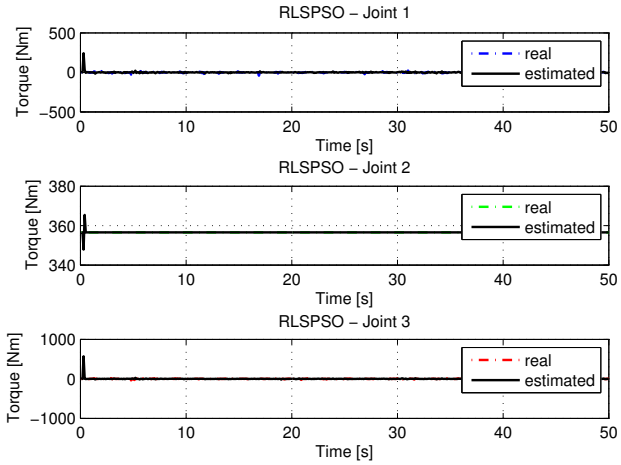

Figure 37. Torque of joints with RLSPSO with noise.

5.3. Comparison of Algorithms

A quantitative analysis of the each algorithm in the identification of the paths of each joint is given in Table 3 by the performance indexes: Multiple Correlation Coefficient, (R^2) and Computational Cost of each algorithms. Equation (29) presents R^2 given by,

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2} \tag{29}$$

where y_i are the observed data, $\bar{y}$ is mean of the observed data, and $\hat{y}$ data estimated by the model.

Can be observed from Table 3, the R^2 values of each joint are assumed values varies from 0 to 1, the closer to 1 means that the estimate is good.

Table 3. Indexes R^2 (Joint 1, 2 and 3) and computational cost of algorithms.

Method	$R^2_{J_1}$	$R^2_{J_2}$	$R^2_{J_3}$	Comp. Cost (s)
LS	0.8873	0.7858	0.6829	2.565149
RLS	0.7946	0.7652	0.8408	65.039719
RLSPSO	0.8016	0.8017	0.8510	37.585912

Table 4 shows the results of the R^2 and computational cost of algorithms with noises.

Table 4. Indexes R^2 (Joint 1, 2 and 3) and computational cost of algorithms with noises.

Method	$R^2_{J_1}$	$R^2_{J_2}$	$R^2_{J_3}$	Comp. Cost (s)
LS	0.8129	0.7275	0.6129	2.851231
RLS	0.7321	0.7118	0.8012	73.989122
RLSPSO	0.7971	0.7912	0.8221	69.969319

From Table 3 can be observed that the index R^2 of the algorithm RLSPSO has a better result than the LS and RLS as well as a lower computational cost compared to the conventional RLS but was higher than that of the LS. For this application the proposed algorithm presented a better performance than the conventional RLS algorithm. The RLSPSO algorithm in this work is presented as a form of improvement of conventional RLS. Noise input methods achieved satisfactory results when compared to noisy methods.

Table 5 presents the complexity of the LS, RLS, and RLSPSO algorithms in terms of number of sums, multiplication, and divisions. It can be assumed that regressor vector has length M.

Table 5. Complexity of Algorithms per input Sample.

Method	*Additions*	*Multiplications*	*Divisions*
LS	$2M^2 + 2M$	$4M^2 + 6M + 1$	1
RLS	$3M^2 + 4M - 1$	$6M^2 + 11M + 1$	1
RLSPSO	$2M^2 + 3M - 1$	$4M^2 + 8M + 1$	1

6. Discussion

Regarding the convergence of the RLSPSO algorithm: noise-free in the sixth iteration the algorithm converges for the best result and noise also begins to converge in the sixth iteration and fully converges in the forty-fifth iteration. Compared to classic RLS obtained an improvement in computational cost and overall result.

The main difficulty in the classical method is in weighting of covariance matrix that is empirically initialized. The metaheuristic pondered this matrix in a search space optimally so the covariance matrix was found faster and more efficiently than empirically or exhaustively searching. However this fact took into account the robustness of the RLSPSO algorithm because even being injected noise in the inputs managed to converge quickly and obtaining satisfactory results. The non-linearity of the system and the changes of operating points made identification difficult it is worth noting that the proposal may be valid as an alternative for nonlinear and variant systems.

Inadequate choice of covariance matrix may compromise method identification, so PSO was able to obtain a covariance matrix that could be robust enough to perform well even with data noise.

The speed and acceleration of a manipulator while performing a manipulation task depend on: grip stability, working environment, material shape, weight, material and stiffness of the object to be manipulated, type of grip or tool used. A cylindrical manipulator can be designed for high rigidity and load capacity and is suitable for transferring oversized materials, handling some parts or handling simple tools, not suitable for other tasks such as welding, assembling, grinding and usually work at low speeds [22,23]. For material handling tasks, the end-effector consists of a jaw of appropriate shape and size, determined by the object to be grasped. For machining and assembly tasks, the end-effector is a specialized tool or device, for example, a welding torch, a spray gun, a mill, a drill bit or a screwdriver [23].

For this paper, the data were collected at low speed because it is intended to use the manipulator for the displacement of high mass loads and lower speeds will be necessary because the material of the tool may slip, which also depends on the type of grip used, high speeds may occur as the material comes loose causing accidents. Another task that is intended to be used with the manipulator under study is the inspection where a camera will be used in place of the end-effector, to perform product quality inspection. The speeds we want to apply are in the range 0.1 to 1 m/s (linear speeds) and 5 to 50 deg/s (angular speeds) [23] for manipulation tasks. For inspection tasks the speed of the camera (which will be mounted on the end-effector) will be in the range of 0.10 to 0.30 m/s [35].

6.1. More Method Results

More results of the identification of each method are presented here, where a more detailed comparison was made for a better visualization. Figures 38–41 show the identifications of the methods noise-free noise and Figures 42–45 with noise

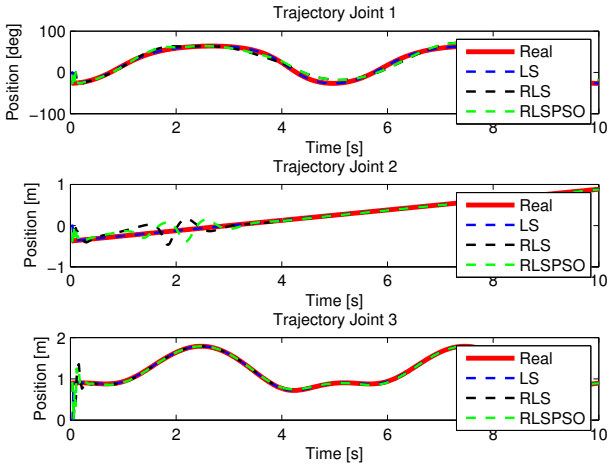

Figure 38. Trajectories identifications.

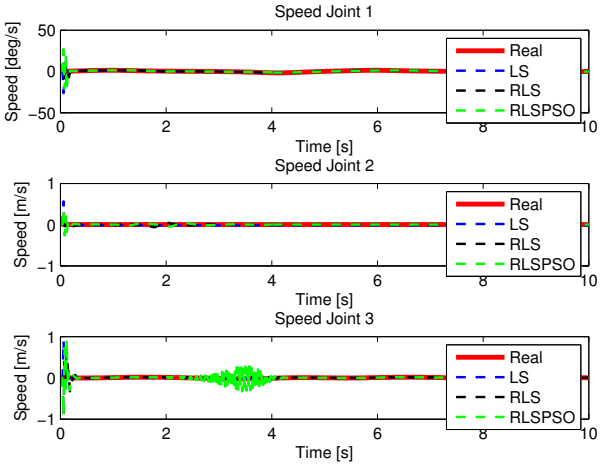

Figure 39. Speed identifications.

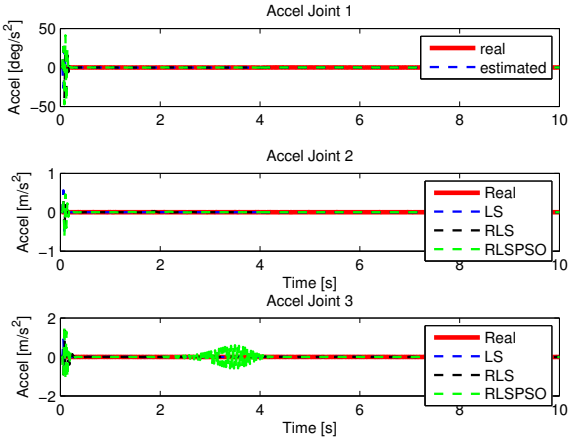

Figure 40. Acceleration identifications.

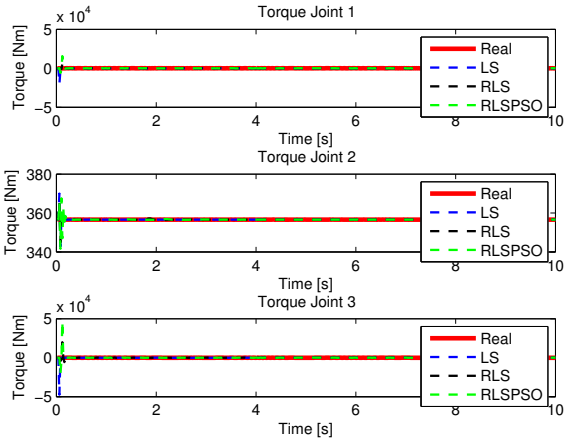

Figure 41. Torque identifications.

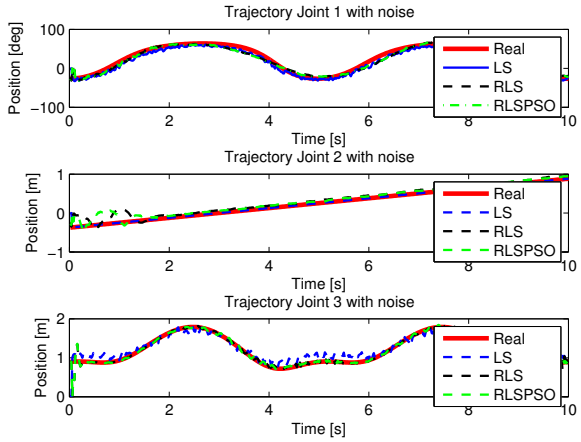

Figure 42. Trajectories with noise identifications.

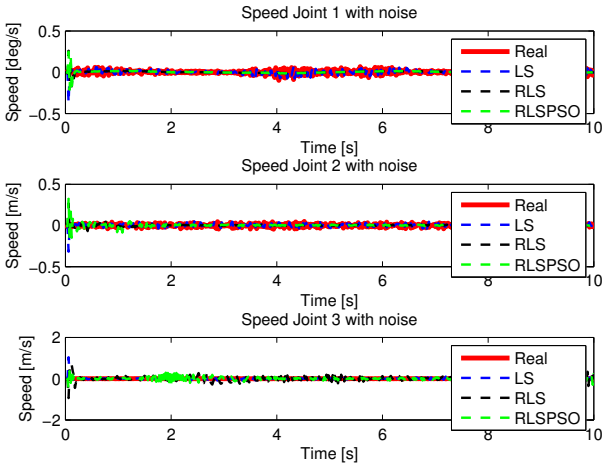

Figure 43. Speed with noise identifications.

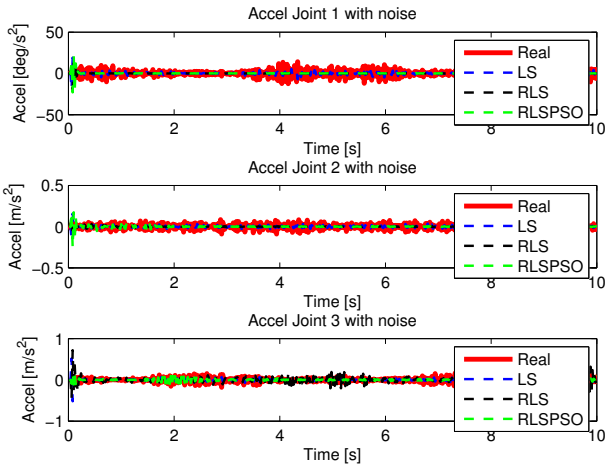

Figure 44. Acceleration with noise identifications.

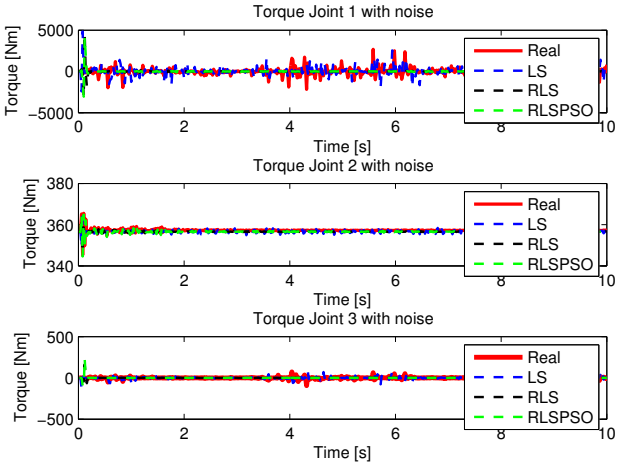

Figure 45. Torque with noise identifications.

7. Conclusions

This work presents an alternative algorithm for calculating the inverse kinematics of robot manipulators based in RLS with PSO identification methods. Other methods were used, assessed and compared, namely LS and RLS. The results shown to be consistent and satisfactory in the identification of the inverse kinematics of the manipulator. Noises have also been added to the data to make estimates more difficult and to check their robustness when working with outlies. To show the efficiency of the algorithms, the R^2 of each algorithm, for each joint was calculated. The RLSPSO algorithm presented a better result than the conventional RLS both in R^2 and in computational cost. This algorithm is a form of improvement on the conventional RLS. This research also presented the kinematic and dynamic modelling of the manipulator. The dynamic model is important for the control of the manipulator. Also research is being performed on trajectory planning in a collision-free environment.

Author Contributions: Conceptualization, J.B. and D.S.; methodology, J.B. and D.S.; software, J.B. and D.S.; validation, L.d.R., A.B. and R.A.; writing–original draft preparation, J.B. and D.S.; writing–review and editing, J.B., D.S, L.d.R. and R.A.; visualization, A.B.; supervision, L.d.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by CAPES.

Acknowledgments: Rui Araújo thanks the support of Portuguese national funds of FCT/MCTES (PIDDAC) under project UIDP/00048/2020.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Dynamic Model of the Cylindrical Manipulator

The Lagrangian formulation based on the mechanical system is defined as

$$L(q, \dot{q}) = K(q, \dot{q}) - P(q) \tag{A1}$$

For the cylindrical manipulator under study the kinetic and potential energy was calculated and then the Lagrangian based formulation was applied.

Appendix A.1. Kinetic Energy

The total kinetic energy of the drive with drive for the three joints is given by [19]:

$$K = K_1 + K_2 + K_3 \tag{A2}$$

where

$$K_1 = \frac{1}{2}[m_1 v_{c1}^2 + \omega_1^T I_1 \omega_1], \tag{A3}$$

$$K_2 = \frac{1}{2}[m_2 v_{c2}^2 + \omega_2^T I_2 \omega_2] \tag{A4}$$

and

$$K_3 = \frac{1}{2}[m_3 v_{c3}^2 + \omega_3^T I_3 \omega_3] \tag{A5}$$

From the Jacobian presented in (12) one can determine the speeds and the equations of the kinetic energy are

$$K_1 = \frac{1}{2}[m_1(-cos\theta_1 d_3\dot{\theta}_1 + sen\theta_1\dot{d}_3)^2], \tag{A6}$$

$$K_2 = \frac{1}{2}[m_2(-sen\theta_1 d_3\dot{\theta}_1 + cos\theta_1\dot{d}_3)^2], \tag{A7}$$

and

$$K_3 = \frac{1}{2}[m_3\dot{d}_2^2 + \dot{\theta}_1^2 I_3] \tag{A8}$$

After performing all the operations and some trigonometric transformations we have the equation that represents the total kinetic energy

$$\begin{aligned} K = \frac{1}{2}[(m_1 + m_2)(-2cos\theta_1 sen\theta_1)(d_3\dot{d}_3\dot{\theta}_1) + (m_1 cos^2\theta_1 + m_2 sen^2\theta_1)(d_3^2\dot{\theta}_1^2) + \\ (m_1 sen^2\theta_1 + m_2 cos^2\theta_1)(\dot{d}_3^2) + m_3\dot{d}_2^2 + \dot{\theta}_1^2 I_3] \end{aligned} \tag{A9}$$

Appendix A.2. Potential Energy

Starting from the definitions of the classical mechanics of reference point (zero of potential energy) the potential energy for each joint of the manipulator is

$$P_1 = m_1 g l_1 sen\theta_1 = 0 \tag{A10}$$

because $l_1 = a_1 = 0$,

$$P_2 = m_2 g d_2 \tag{A11}$$

and

$$P_3 = m_3 g d_2 \tag{A12}$$

As $P = P_1 + P_2 + P_3$, we have

$$P = g d_2(m_2 + m_3) \tag{A13}$$

Appendix A.3. Lagrange Equation

The equations of motion of the system are given by

$$\frac{d}{dt}[\frac{\partial L}{\partial \dot{q}}] - \frac{\partial L}{\partial q} = \tau \tag{A14}$$

where $\tau \in \Re^n$ are the torques (forces) applied to the joints. Thus, considering the kinetic energy of the manipulator, the dynamic equation of the manipulator can be written in simplified form as

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \tau \tag{A15}$$

where $C \in \Re^n$ is the matrix that describes the centripetal and Coriolis forces, and $G = \frac{\partial g}{\partial q} \in \Re^n$ is the gravity vector.

The effects of joint friction and external forces on the end-effector can be included in the dynamic model of the manipulator

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + F(q)q + G(q) = \tau - f_{ext} \tag{A16}$$

On what f_{ext} is the external force applied at the end-effector and $F(q) \in \Re^{\times n}$ represents the effects of dynamic and static friction forces on the joints. This vector also represents disturbances and dynamics not modeled as gaps in couplings and mechanical transmissions.

Applying the Lagrange formulation (A1), the Lagrangian for the system will be

$$\begin{aligned} L = \frac{1}{2}[(m_1+m_2)(-2cos\theta_1 sen\theta_1)(d_3\dot{d}_3\dot{\theta}_1) + (m_1 cos^2\theta_1 + m_2 sen^2\theta_1)(d_3^2\dot{\theta}_1^2) + \\ (m_1 sen^2\theta_1 + m_2 cos^2\theta_1)(\dot{d}_3^2) + m_3\dot{d}_2^2 + \dot{\theta}_1^2 I_3] - g d_2(m_2+m_3) \end{aligned} \tag{A17}$$

The equation of the manipulator motion from the Lagrangian formulation is obtained by the partial derivatives of the Lagrangian Equation (A17). The following are the partial derivatives for the first manipulator joint

$$\frac{\partial L}{\partial \dot{\theta}_1} = (m_1+m_2)(-sen\theta_1 cos\theta_1)(d_3\dot{d}_3) + (m_1 cos\theta_1^2 + m_2 sen\theta_1^2)(d_3^2\dot{\theta}_1) + \dot{\theta}_1 I_3 \tag{A18}$$

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta}_1} = (m_1+m_2)(cos\theta_1 sen\theta_1)(\dot{d}_3\ddot{d}_3) - 2(2m_1 sen\theta_1 - 2m_2 cos\theta_1)(\dot{d}_3\ddot{\theta}_1) + \ddot{\theta}_1 I_3 \tag{A19}$$

$$\frac{\partial L}{\partial \theta_1} = (m_1+m_2)(sen\theta_1 cos\theta_1)(d_3\dot{d}_3\dot{\theta}_1) - (m_1 sen\theta_1 - m_2 cos\theta_1)(\dot{d}_3\dot{\theta}_1^2) + (m_1 cos\theta_1 - m_2 sen\theta_1)(\dot{d}_3^2) \tag{A20}$$

The first equation of motion describing the torque of joint 1 will be:

$$\begin{aligned} \tau_1 = &-[(4m_1 sen\theta_1 - 4m_2 cos\theta_1)d_3 + I_3]\ddot{\theta}_1 \\ &+[(m_1+m_2)(sen\theta_1 cos\theta_1)d_3]\ddot{d}_3 \\ &+[(m_1 sen\theta_1 - m_2 cos\theta_1)d_3]\dot{\theta}_1^2 \\ &-[m_1 cos\theta_1 + m_2 sen\theta_1]\dot{d}_3^2 \\ &-[(m_1+m_2)(sen\theta_1 cos\theta_1)d_3]\dot{\theta}_1\dot{d}_3 \end{aligned} \tag{A21}$$

In the same way, the partial derivatives for the second joint of the manipulator

$$\frac{\partial L}{\partial \dot{d}_2} = m_3\dot{d}_2 \tag{A22}$$

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{d}_2} = m_3\ddot{d}_2 \tag{A23}$$

$$\frac{\partial L}{\partial d_2} = -g(m_2 + m_3) \tag{A24}$$

The equation of motion describing the torque of the joint 2 will be

$$\tau_2 = m_3\ddot{d}_2 + g(m_2 + m_3) \tag{A25}$$

Following the same idea, the partial derivatives for the third joint of the manipulator will be

$$\frac{\partial L}{\partial \dot{d}_3} = (m_1 + m_2)(-sen\theta_1 cos\theta_1 d_3\dot{\theta}_1) + 2(m_1 cos\theta_1 - 2m_2 sen\theta_1)\dot{d}_3 \tag{A26}$$

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{d}_3} = [m_1 sen\theta_1 cos\theta_1]\ddot{\theta}_1 - [2(m_1 sen\theta_1 + m_2 cos\theta_1)]\ddot{d}_3 + [m_2 sen\theta_1 cos\theta_1]\dot{d}_3 \tag{A27}$$

$$\frac{\partial L}{\partial d_3} = (m_1 + m_2)(sen\theta_1 cos\theta_1)(\dot{d}_3\dot{\theta}_1) - (2m_1 sen\theta_1 - 2m_2 cos\theta_1)(d_3\dot{\theta}_1^2) \tag{A28}$$

The equation of motion describing the torque of the joint 3 will be

$$\begin{aligned} \tau_3 = {} & [m_1 sen\theta_1 cos\theta_1]\ddot{\theta}_1 \\ & -[2(m_1 sen\theta_1 + m_2 cos\theta_1)]\ddot{d}_3 \\ & +[2d_3(m_1 sen\theta_1 - m_2 cos\theta_1)]\dot{\theta}_1^2 \\ & -[(m_1 + m_2)(sen\theta_1 cos\theta_1)]\dot{\theta}_1\dot{d}_3 \end{aligned} \tag{A29}$$

Appendix A.4. Dynamics of the Matrix form Manipulator

Writing Equations (A21), (A25) and (A29) in the standard matrix form as shown in (A15) we have

$$\begin{aligned} \begin{bmatrix} \tau_1 \\ \tau_2 \\ \tau_3 \end{bmatrix} = {} & \begin{bmatrix} (4m_1 sen\theta_1 - 4m_2 cos\theta_1)d_3 + I_3 & 0 & (m_1 + m_2)(sen\theta_1 cos\theta_1)d_3 \\ 0 & m_3 & 0 \\ m_1 sen\theta_1 cos\theta_1 & 0 & 2(m_1 sen\theta_1 + m_2 cos\theta_1) \end{bmatrix} \times \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{d}_2 \\ \ddot{d}_3 \end{bmatrix} \\ & + \begin{bmatrix} (m_1 sen\theta_1 - m_2 cos\theta_1)d_3 & 0 & -m_1 cos\theta_1 + m_2 sen\theta_1 \\ 0 & 0 & 0 \\ 2d_3(m_1 sen\theta_1 - m_2 cos\theta_1) & 0 & 0 \end{bmatrix} \times \begin{bmatrix} \dot{\theta}_1^2 \\ \dot{d}_2^2 \\ \dot{d}_3^2 \end{bmatrix} \\ & + \begin{bmatrix} 0 & -(m_1 + m_2)(sen\theta_1 cos\theta_1)d_3 & 0 \\ 0 & 0 & 0 \\ 0 & -(m_1 + m_2)(sen\theta_1 cos\theta_1) & 0 \end{bmatrix} \times \begin{bmatrix} \dot{\theta}_1\dot{d}_2 \\ \dot{\theta}_1\dot{d}_3 \\ \dot{d}_2\dot{d}_3 \end{bmatrix} \\ & + \begin{bmatrix} 0 \\ g(m_2 + m_3) \\ 0 \end{bmatrix} \end{aligned} \tag{A30}$$

In Equation (A30), the terms $\ddot{\theta}_1$, $\ddot{d}_2$, $\ddot{d}_3$ are related to the angular accelerations of the links, the terms $\dot{\theta}_1^2$, $\dot{d}_2^2$, $\dot{d}_3^2$ are centripetal accelerations, and the terms $\dot{\theta}_1\dot{d}_2$, $\dot{\theta}_1\dot{d}_3$, $\dot{d}_2\dot{d}_3$ are the Coriolis accelerations.

References

1. Pinto, M.F.; Mendonça, T.R.; Olivi, L.R.; Costa, E.B.; Marcato, A.L. Modified approach using variable charges to solve inherent limitations of potential fields method. In Proceedings of the 2014 11th IEEE/IAS International Conference on Industry Applications, Juiz de Fora, Brazil, 7–10 December 2014.
2. Vijaysai, P.; Gudi, R.D.; Lakshminarayanan, S. Identification on demand using a blockwise recursive partial least-squares technique. *Ind. Eng. Chem. Res.* **2003**, *42*, 540–554. [CrossRef]
3. Hafezi, Z.; Mohammad, M.A. Recursive generalized extended least squares and RML algorithms for identification of bilinear systems with ARMA noise. *ISA Trans.* **2018**, *88*, 50–61. [CrossRef] [PubMed]
4. Stojanovic, V.; Novak, N. Joint state and parameter robust estimation of stochastic nonlinear systems. *Int. J. Robust Nonlinear Control* **2016**, *26*, 3058–3074. [CrossRef]
5. Stojanovic, V.; Novak, N. Identification of time-varying OE models in presence of non-Gaussian noise: Application to pneumatic servo drives. *Int. J. Robust Nonlinear Control* **2016**, *26*, 3974–3995. [CrossRef]
6. Zhang, G.; Xianku, Z.; Hongshuai, P. Multi-innovation auto-constructed least squares identification for 4 DOF ship manoeuvring modelling with full-scale trial data. *ISA Trans.* **2015**, *58*, 186–195. [CrossRef] [PubMed]
7. Ma, J.; Zhao, L.; Han, Z.; Tang, Y. Identification of Wiener model using least squares support vector machine optimized by adaptive particle swarm optimization. *J. Control Autom. Elect. Syst.* **2015**, *26*, 609–615. [CrossRef]
8. Zha, F.; Sheng, W.; Guo, W.; Qiu, S.; Deng, J.; Wang, X. Dynamic Parameter Identification of a Lower Extremity Exoskeleton Using RLS-PSO. *Appl. Sci.* **2019**, *9*, 324. [CrossRef]
9. Mizuno, N.; Nguyen, C.H. Parameters identification of robot manipulator based on particle swarm optimization. In Proceedings of the 2017 13th IEEE International Conference on Control & Automation (ICCA), Ohrid, Macedonia, 3–6 July 2017.
10. Guo, W.; Li, R.; Cao, C.; Gao, Y. Kinematics, dynamics, and control system of a new 5-degree-of-freedom hybrid robot manipulator. *Adv. Mech. Eng.* **2016**, *8*, 1687814016680309. [CrossRef]
11. Tutsoy, O.; Duygun, E.B.; Sule, C. Learning to balance an NAO robot using reinforcement learning with symbolic inverse kinematic. *Trans. Inst. Meas. Control* **2017**, *39*, 1735–1748. [CrossRef]
12. Tutsoy, O.; Calikusu, I.; Colak, S.; Vahid, O.; Barkana, D.E.; Gongor, F. Developing Linear and Nonlinear Models of ABB IRB120 Industrial Robot with MapleSim Multibody Modelling Software. *Eurasia Proc. Sci. Technol. Eng. Math.* **2017**, *12*, 273–285.
13. Nazari, A.A.; Ali, S.A.M.; Ayyub, H. Kinematics analysis, dynamic modeling and verification of a CRRR 3-DOF spatial parallel robot. In Proceedings of the 2nd International Conference on Control, Instrumentation and Automation, Shiraz, Iran, 27–29 December 2011.
14. Guo, X.; Lei, Z.; Kai, H. Dynamic parameter identification of robot manipulators based on the optimal excitation trajectory. In Proceedings of the 2018 IEEE International Conference on Mechatronics and Automation (ICMA), Changchun, China, 5–8 August 2018.
15. Yuan, J.J.; Wan, W.; Fu, X.; Wang, S.; Wang, N. A novel LLSDPso method for nonlinear dynamic parameter identification. *Assem. Autom.* **2017**, *37*, 490–498. [CrossRef]
16. Urrea, C.; Pascal, J. Parameter identification methods for real redundant manipulators. *J. Appl. Res. Technol.* **2017**, *15*, 320–331. [CrossRef]
17. Yan, D.; Lu, Y.; Levy, D. Parameter identification of robot manipulators: A heuristic particle swarm search approach. *PLoS ONE* **2015**, *10*, e0129157. [CrossRef] [PubMed]
18. *Edge, Solid Software*; Siemens Global Website; Siemens PLM Software: Stuttgart, Germany, 2019.
19. Sanz, P. Robotics: Modeling, planning, and control (siciliano, b. et al; 2009) [on the shelf]. *IEEE Robot. Autom. Mag.* **2009**, *16*, 101. [CrossRef]
20. Spong, M.W.; Hutchinson, S.; Vidyasagar, M. *Robot Modeling and Control*; Wiley: New York, NY, USA, 2006.
21. Hartenberg, R.; Danavi, J. *Kinematic Synthesis of Linkages*; McGraw-Hill: New York, NY, USA, 1964.
22. Spong, M.W.; Mathukumalli, V. *Robot Dynamics and Control*; John Wiley & Sons: Hoboken, NJ, USA, 2008.
23. Siciliano, B.; Sciavicco, L.; Villani, L.; Oriolo, G. *Robotics: Modelling, Planning and Control*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2010.

24. Potkonjak, V. *Dynamics of Manipulation Robots: Theory and Application*; Springer: Berlin/Heidelberg, Germany, 1982.
25. Kozlowski, K.R. *Modelling and Identification in Robotics*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2012.
26. Coelho, A.A.R.; Santos, L.C. *Identificação de sistemas dinâmicos lineares*; Editora UFSC: Santa Catarina, Brazil, 2004.
27. Ljung, L.; Soderstom, T. *Theory and Practice of Recursive Identification*; MIT Press: Cambridge, MA, USA, 1983.
28. Kjaer, A.P.; Heath, W.P.; Wellstead, P.E. Identification of cross-directional behaviour in web production: Techniques and experience. *Control Eng. Pract.* **1995**, *3*, 21–29. [CrossRef]
29. Aguirre, L.A. *Introdução à Identificação de Sistemas—Técnicas lineares e não-lineares aplicadas a sistemas reai*; Editora UFMG, 3a: Belo Horizonte, Brazil, 2007.
30. Viola, J.; Angel, L. Tracking control for robotic manipulators using fractional order controllers with computed torque control. *IEEE Latin Am. Trans.* **2018**, *16*, 1884–1891. [CrossRef]
31. Ljung, L. *System Identification: Theory for the User Pers*; Tsinghua University Press and Prentice: Beijing, China, 2002.
32. Eberhart, R.; Kennedy, J. A new optimizer using particle swarm theory, Micro Machine and Human Science. In Proceedings of the MHS'95—Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, 4–6 October 1995; pp. 39–43.
33. Paiva, F.A.P.; Costa, J.A.F.; Silva, C.R.M. A Serendipity-Based Approach to Enhance Particle Swarm Optimization Using Scout Particles. *IEEE Latin Am. Trans.* **2017**, *15*, 1101–1112. [CrossRef]
34. Engelbrecht, A.P. *Computational Intelligence: An Introduction*; John Wiley & Sons: Hoboken, NJ, USA, 2007.
35. Mineo, C.; Pierce, S.G.; Nicholson, P.I.; Cooper, I. Robotic path planning for non-destructive testing—A custom MATLAB toolbox approach. *Robot. Comput.-Integr. Manuf.* **2016**, *37*, 1–12. [CrossRef]