mathematics

MDPI

Article

Data-Driven Adaptive Modelling and Control for a Class of Discrete-Time Robotic Systems Based on a Generalized Jacobian Matrix Initialization

América Berenice Morales-Díaz [1], Josué Gómez-Casas [2,*], Chidentree Treesatayapun [1], Carlos Rodrigo Muñiz-Valdez [2], Jesús Salvador Galindo-Valdés [2] and Jesús Fernando Martínez-Villafañe [2]

[1] Department of Robotics and Advanced Manufacturing, CINVESTAV-Saltillo, Ramos Arizpe 25900, Mexico; america.morales@cinvestav.edu.mx (A.B.M.-D.); chidentree@cinvestav.edu.mx (C.T.)
[2] Faculty of Engineering, Autonomous University of Coahuila, Arteaga 25350, Mexico; rodrigo.muniz@uadec.edu.mx (C.R.M.-V.); s_galindo@uadec.edu.mx (J.S.G.-V.); jesusmartinezvillafane@uadec.edu.mx (J.F.M.-V.)
* Correspondence: jogomezc@uadec.edu.mx

Abstract: Data technology advances have increased in recent years, especially for robotic systems, in order to apply data-driven modelling and control computations by only considering the input and output signals' relationship. For a data-driven modelling and control approach, the system is considered unknown. Thus, the initialization values of the system play an important role to obtain a suitable estimation. This paper presents a methodology to initialize a data-driven model using the pseudo-Jacobian matrix algorithm to estimate the model of a mobile manipulator robot. Once the model is obtained, a control law is proposed for the robot end-effector position tasks. To this end, a novel neuro-fuzzy network is proposed as a control law, which only needs to update one parameter to minimize the control error and avoids the chattering phenomenon. In addition, a general stability analysis guarantees the convergence of the estimation and control errors and the tuning of the closed-loop control design parameters. The simulations results validate the performance of the data-driven model and control.

Keywords: data-driven model; Jacobian matrix initialization; robotic system

MSC: 65

Citation: Morales-Díaz, A.B.; Gómez-Casas, J.; Treesatayapun, C.; Muñiz-Valdez, C.R.; Galindo-Valdés, J.S.; Martínez-Villafañe, J.F. Data-Driven Adaptive Modelling and Control for a Class of Discrete-Time Robotic Systems Based on a Generalized Jacobian Matrix Initialization. *Mathematics* **2023**, *11*, 2555. https://doi.org/10.3390/math11112555

Academic Editor: Ivan Lorencin

Received: 27 April 2023
Revised: 26 May 2023
Accepted: 28 May 2023
Published: 2 June 2023

1. Introduction

In the last decade, data-driven control has increased its applications in mechanical, electronic, and robotic systems, see, for instance, the works of [1–3]. A data-driven approach simplifies the modelling in complex processes using only the online input and output signals of the system. A data-driven model is based on the premise that the system is unknown [4–6]. In the same way, the only requirement is the measurable information from the system to approximate its model [7]. Robot manipulators are considered complex and nonlinear systems with parametric uncertainties. Thus, it is difficult to define their model precisely. The Jacobian matrix represents the system for a kinematic control at the velocity level to achieve the position tasks of the robot's end-effector. The robot model is obtained through the approximation of the Jacobian matrix using a data-driven approach. In comparison with the traditional robot model, the data-driven modelling requires less information to approximate the Jacobian matrix. The main difference between the traditional and estimation method for robot modelling is the initialization of the Jacobian matrix. While the traditional method has its initial conditions well defined, in the case of a data-driven model, the initial conditions are unknown.

Data-driven modelling works by using the input and output signals from the system [8–10]. Therefore, there is no discrimination in the class, structure, or type of robot to apply this approach [10–12]. It can be applied to inertial, noninertial, and flexible

robots with multi-input and multioutput signals [13–16]. The Jacobian matrix is computed only with the measurements of the online joint velocities and end-effector velocities [17]. Moreover, data-driven modelling and control can be applied to robots as a complement to industrial processes. Ref. [18] implemented a data-driven model in the joint position control of a robotic arm as single-input single-output (SISO) system. On the other hand, a redundant robot was considered as a discrete-time MIMO system under the principle of a data-driven model, where the time-varying parameter was the estimated Jacobian matrix [19]. In that particular case, Ref. [20] proposed the pseudo-Jacobian matrix (PJM) algorithm to estimate the kinematic model of MIMO robotic systems.

Commonly, the computation of a data-driven model in SISO systems is through the principle of the pseudo-partial derivative (PPD) [21,22]. Hence, the estimated model starts up with a scalar value. However, the challenge for data-driven models in redundant robots is that they require the Jacobian matrix computation, so that the estimated model requires a set of initial values. The initialization of the Jacobian matrix values represents a challenging topic for data-driven modelling, since the system is unknown even from the beginning. The most common proposal used is to use a zero initialization. Thus, the initial Jacobian matrix values tend to update with the same order of magnitude. However, each value of the Jacobian matrix depends on different orders of magnitude with respect to the robot's nature. As a consequence, the implementation of the zero-initialization technique diminishes the quality of the estimation algorithm. On the other hand, a Jacobian matrix with random initialization values loses repeatability and estimation performance. The initialization techniques are generally applied to computer vision applications. Ref. [23] presented the initialization of the segmentation of images for a medical approach. Ref. [24] implemented the initialization of multimodal pair registration algorithm. However, in the case of the initial values of the Jacobian matrix, it is necessary to satisfy the performance of the robot control. Moreover, it is important to guarantee the convergence of the estimation and control errors. In general, each type of robot demands an exclusive Jacobian matrix of its kinematic model. Therefore, the initialization values for the PJM approach represent a challenge for the topological configuration of each robot. The proposal of this work is to present a generalized methodology to find the adequate values for the Jacobian matrix initialization.

As was mentioned above, the common initialization techniques are selected intuitively by the user's experience, which are unsatisfactory for starting up the PJM algorithm. This work proposes a novel methodology for the Jacobian initialization based on the kinematic constraints, rank, and domain values of the Jacobian matrix.

This paper establishes a methodology for the initial values' selection of an estimated Jacobian matrix based on the input and output signals of a redundant robot. The equivalent model is approximated through the PJM algorithm. A control law is also proposed based on a new neuro-fuzzy network that only needs to adapt one parameter to achieve the convergence of the control error. The input to the neuro-fuzzy network is a function in terms of the future error, which avoids the chattering phenomenon [25].

To demonstrate the initialization methodology, the end-effector control of an eight-degree-of-freedom (dof) redundant robot through three different scenarios is performed, including a regulation control task, trajectory tracking task, and position task against disturbance. Furthermore, a general stability analysis is established, including the parameter settings for the estimation model and the proposed control law.

The structure of this work is as follows: Section 2 describes the robotic system representation and the Jacobian matrix initialization, Section 3 introduces the control law design and the Lyapunov analysis, Section 4 exposes the numerical results, and Section 5 provides the conclusions.

2. Robotic System Representation

The representation of the end-effector velocity working within the discrete-time domain is approximated by the following expression:

$$\frac{\chi(k+1)-\chi(k)}{T_s} = J_A^*(k)\frac{q(k)-q(k-1)}{T_s} \tag{1}$$

where $J_A^*(k)$ is an ideal Jacobian matrix, $\nu(k+1) = \frac{\chi(k+1)-\chi(k)}{T_s}$ is the end-effector velocity, $\omega(k) = \frac{q(k)-q(k-1)}{T_s}$ represents the joints' velocities, and T_s is the sampling time.

Assumption 1 below is required for the robot control in a closed-loop configuration.

Assumption 1. *The robotic system needs to satisfy the Lipschitz condition, where a positive constant L limits the relationship between the system's input and output:* $\| \nu(k+1) \| \leq L \| \omega(k) \|$. *This means a change in the output of the system imposes a change in the input of the system.*

The ideal Jacobian matrix approach $J_A^*(k)$ is represented by:

$$J_A^*(k) = \hat{\mathbf{J}}_\mathbf{A}(k) + \epsilon(k) \tag{2}$$

where $\hat{\mathbf{J}}_\mathbf{A}(k)$ is the Jacobian matrix computed for the PJM algorithm, and $\epsilon(k)$ is the estimation error. The Jacobian matrix is the relationship between the output and input signals within the discrete-time domain

$$\hat{\mathbf{J}}_\mathbf{A}(k) = \frac{\nu(k+1)}{\omega(k)} \in \mathbb{R}^{m\times n} \tag{3}$$

where m is equal to the number of end-effector dofs and n is equal to the number of dofs of the robot, whereas the robot is considered a MIMO system. The robotic system requires the observability condition in Assumption 2 in order to apply a data-driven model and control.

Assumption 2. *To apply a data-driven model and control under the concept of the PJM algorithm, the output of the robotic system needs to be observable,* $\nu(k+1) = \hat{\mathbf{J}}_\mathbf{A}(k)\omega(k)\ \forall k > 0$. *An approximated Jacobian matrix is computed as a model identification by the measured output (end-effector velocity).*

The PJM algorithm approximates the Jacobian matrix, see the reference [20] for more details. The Jacobian matrix is updated as follows:

$$\hat{\mathbf{J}}_\mathbf{A}(k+1) = \hat{\mathbf{J}}_\mathbf{A}(k) + \frac{\eta\left[J_A^*(k)\omega(k) - \hat{\mathbf{J}}_\mathbf{A}(k)\omega(k)\right]\omega^T(k)}{\mu + \| \omega(k) \|^2} \tag{4}$$

where $\mu, \eta \in \mathbb{R}^+$ are the weight parameter and the step parameter, respectively.

2.1. Jacobian Matrix Initialization

The initialization of an estimated model is a critical topic for data-driven modelling and control. The selection of the initial parameters for an estimated Jacobian matrix plays an important role to determine the end-effector trajectory and the convergence time of the control error and the control signals. The first stage of a data-driven model is to determine the initialization values of the Jacobian matrix. Thus, during the second stage, the estimation algorithm approximates the Jacobian matrix. In the first stage, the proposed initialization contains fixed values only to start up the estimation algorithm, and during the second stage the estimation algorithm computes the online Jacobian matrix. The quality of the online estimation algorithm relies on the adequate selection values in the initialization Jacobian matrix. Consequently, the robot can achieve the proper control position task in a closed-loop configuration.

The proposed methodology for the initialization of the Jacobian matrix is presented below.

Assumption 3. *The initialization values from the estimated Jacobian matrix* $\hat{\mathbf{J}}_{\mathbf{A}}(0)$ *and the control signals* $\omega(0)$ *should satisfy* $\epsilon(k) \approx 0$ *when* $k \to \infty$.

Corollary 1. *If the equivalent model obtained by the PJM algorithm fulfils Assumptions 1–3, the initialization of the estimated Jacobian matrix should satisfy the next criteria:*

- *i* *Fulfil the robot kinematic constraints.*
- *ii* *The Jacobian matrix should be full rank, i.e.,* $\sigma(\hat{\mathbf{J}}_{\mathbf{A}}(0)) = m$.
- *iii* *The domain* $\mathcal{D}_i$ *of the initial Jacobian matrix* $\hat{\mathbf{J}}_{\mathbf{A}}(0)$ *should guarantee that the estimation error* $\epsilon(k) \approx 0$ *and the control error* $e(k) \to 0$ *when* $k \to \infty$.
- *iv* *The domain of the Jacobian matrix from a discrete-time MIMO system belongs to a domain* $\hat{\mathbf{J}}_{\mathbf{A}}(k) \in \mathcal{D}$, *and the subdomain from the initial conditions* $\hat{\mathbf{J}}_{\mathbf{A}}(0) \in \mathcal{D}_i$ *belongs to the Jacobian matrix domain, this means* $\mathcal{D}_i \in \mathcal{D}$.
- *v* *Therefore, the error converges to a vicinity of the origin for tracking control in a compact set* Ω_{set}, *when the criteria i to iv are fulfilled.*

To exemplify the initialization method, the analysis of this work was applied to a mobile manipulator robot. KUKA youBot has 8 dofs. The estimated Jacobian matrix represents the whole robot with two parts: the omnidirectional platform and the robotic arm. The omnidirectional platform is composed of 3 dofs: 2 prismatic joints in the x and y direction, and 1 revolute joint. The robotic arm is composed of 5 revolute joints. The topology configuration of the robot is $SE(2) \times \mathbb{T}^5$.

The structure of the initial estimated Jacobian matrix is defined as:

$$\hat{\mathbf{J}}_{\mathbf{A}}(0) = \begin{bmatrix} \begin{bmatrix} & a_0 & \end{bmatrix}_{3\times 3} & \begin{bmatrix} b_0 \end{bmatrix}_{1\times 5} \\ & & \begin{bmatrix} c_0 \end{bmatrix}_{1\times 5} \\ & & \begin{bmatrix} d_0 \end{bmatrix}_{1\times 5} \end{bmatrix} \in \mathbb{R}^{3\times 8} \tag{5}$$

where a_0 represents the holonomic constraint of the omnidirectional mobile platform expressed as follows

$$a_0 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \in \mathbb{R}^{3\times 3} \tag{6}$$

Then, b_0 is the subspace of the estimated Jacobian matrix which represents the minimum change from axis x with respect to the robotic arm joints represented by

$$b_0 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \in \mathbb{R}^{1\times 5}; \tag{7}$$

likewise, c_0 is the subspace of the estimated Jacobian matrix which represents the minimum change from axis y with respect to the robotic arm joints represented by

$$c_0 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \in \mathbb{R}^{1\times 5}; \tag{8}$$

for this case, d_0 is the subspace of the estimated Jacobian matrix which represents the change from axis z with respect to the robotic arm. The selection of the d_0 values becomes essential to fulfil the requirements of the estimated Jacobian matrix initialization. It is important to select a domain of values to satisfy the conditions of a full rank of the Jacobian matrix and the convergence of the estimation error regarding the criteria in Assumption 1. The set of values selected were:

$$d_0 = \begin{bmatrix} -0.06 & -0.05 & -0.03 & 0.02 & 0.03 \end{bmatrix} \in \mathbb{R}^{1\times 5} \tag{9}$$

Therefore, according to the previous conditions, the initial Jacobian matrix was formed as follows

$$\hat{\mathbf{J}}_{\mathbf{A}}(0) = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -0.06 & -0.05 & -0.03 & 0.02 & 0.03 \end{bmatrix} \in \mathbb{R}^{3\times 8} \tag{10}$$

where the selected values in (10) satisfied $\sigma(\hat{\mathbf{J}}_{\mathbf{A}}(0)) = 3$ and they guaranteed an estimation error close to zero, which is demonstrated in Corollary 2.

Corollary 2. *The term $P_k = J_A^*(k)\hat{\mathbf{J}}_{\mathbf{A}}^{\dagger}(k)$ allows a comparison between the ideal Jacobian $J_A^*(k)$ and the estimated Jacobian $\hat{\mathbf{J}}_{\mathbf{A}}(k)$. Therefore, it satisfies the next inequality,*

$$P_k^{min} < \| \ I - J_A^*(k)\hat{\mathbf{J}}_{\mathbf{A}}^{\dagger}(k) \ \| \leq P_k^{max}. \tag{11}$$

Since P_k^{min} and P_k^{max} are the lower and upper values. They depend on the initialization of the Jacobian matrix $\hat{\mathbf{J}}_{\mathbf{A}}(0)$ [26]; $P_k^{min} = 0$ and $P_k^{max} = 1.85$ were obtained by simulations using a proportional controller, and Figure 1 depicts its performance. In terms of the damping factor, the pseudoinverse matrix computation is

$$\hat{\mathbf{J}}_{\mathbf{A}}^{\dagger}(k) = \hat{\mathbf{J}}_{\mathbf{A}}^{T}(k)\left[\hat{\mathbf{J}}_{\mathbf{A}}(k)\hat{\mathbf{J}}_{\mathbf{A}}^{T}(k) + \xi I\right]^{-1} \tag{12}$$

where the domain of values is established for $\xi \in (0, 1)$. The PJM algorithm approximates the Jacobian matrix by maintaining the estimation error close to zero. Consequently,

$$P_k = J_A^*(k)\hat{\mathbf{J}}_{\mathbf{A}}^{\dagger}(k) \approx I \tag{13}$$

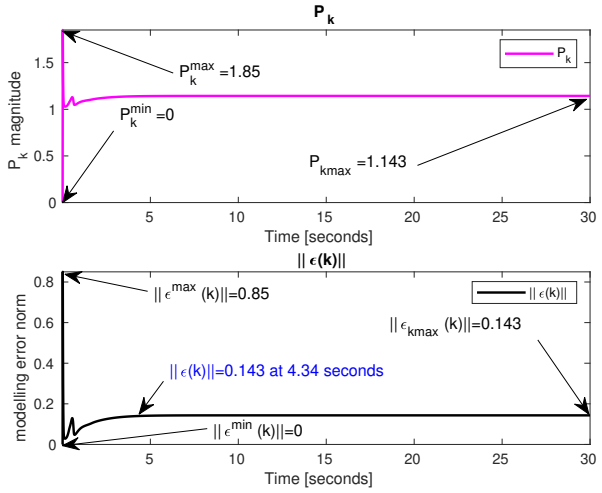


Figure 1. Magnitude of P_k and $\| \ \epsilon(k) \ \|$.

Therefore, an estimation error $\| \ \epsilon(k) \ \| = 0.16$ is guaranteed, according to Figure 1, when the selected values in (10) are applied.

2.2. Equivalent Model Stability Analysis

This section presents the stability analysis for the data-model of the PJM algorithm in (4) using the Lyapunov candidate function in terms of the estimation error. The estimation error is defined as:

$$\epsilon(k+1) = J_A^*(k+1) - \hat{\mathbf{J}}_{\mathbf{A}}(k+1) \tag{14}$$

The ideal Jacobian matrix is the reference model; this Jacobian Matrix is computed by the kinematic model considering the Denavit–Hartenberg parameters. Under a classical approach, the robot model requires us to know the physical and mechanical parameters such as the types of joints (revolute or prismatic ones), the link distance, the centre of mass, and the gravity. In contrast, the proposed data-driven model only requires the inputs and outputs to estimate the Jacobian matrix as a model of the robot. For the case where the estimation error achieves to be zero, the updated Jacobian matrix satisfies $J_A^*(k+1) = J_A^*(k)$; then, the estimated Jacobian matrix becomes:

$$\hat{\mathbf{J}}_{\mathbf{A}}(k+1) = \hat{\mathbf{J}}_{\mathbf{A}}(k) + \Theta_k \parallel \omega(k) \parallel^2 \epsilon(k). \tag{15}$$

Hence, the next inequality should be fulfilled

$$\Theta_k = \frac{\eta}{\mu + \parallel \omega \parallel^2} > 0, \in \mathbb{R}^+ \tag{16}$$

and by substituting (15) in (14), the estimation error is

$$\begin{aligned} \epsilon(k+1) &= J_A^*(k+1) - \hat{\mathbf{J}}_{\mathbf{A}}(k+1) \\ &= J_A^*(k) - \hat{\mathbf{J}}_{\mathbf{A}}(k) - \Theta_k \parallel \omega(k) \parallel^2 \epsilon(k) \\ &= \epsilon(k) - \epsilon(k)\Theta_k \parallel \omega(k) \parallel^2 \end{aligned} \tag{17}$$

and the change in the estimation error is

$$\begin{aligned} \epsilon(k+1) - \epsilon(k) &= -\epsilon(k)\Theta_k \parallel \omega(k) \parallel^2 \\ \Delta\epsilon(k+1) &= -\epsilon(k)\Theta_k \parallel \omega(k) \parallel^2 \end{aligned} \tag{18}$$

Theorem 1. *As the KUKA youBot is considered a discrete-time MIMO system, the PJM algorithm can compute an estimated Jacobian matrix, when the system in (3) satisfies the observability and Lipschitz conditions. Therefore, the data-driven model approach is applied when* $\epsilon(k) \approx 0$.

Proof. The candidate discrete Lyapunov function is

$$V_{\text{sys}}(k+1) = \frac{1}{2}\epsilon(k+1)\epsilon^T(k+1). \tag{19}$$

The Lyapunov function change is represented by

$$\Delta V_{\text{sys}}(k+1) = V_{\text{sys}}(k+1) - V_{\text{sys}}(k). \tag{20}$$

The change in the Lyapunov function in terms of the estimation error $\Delta\epsilon(k+1)$ is

$$\Delta V_{\text{sys}}(k+1) = \Delta\epsilon(k+1)\left[\epsilon(k) + \frac{1}{2}\Delta\epsilon(k+1)\right]^T. \tag{21}$$

Substituting (18) in (21), it is necessary to look at the Lyapunov stability condition

$$\begin{aligned} \Delta V_{\text{sys}}(k+1) = &-\epsilon(k)\epsilon^T\Theta_k \parallel \omega(k) \parallel^2 \\ &\times \left[1 - \frac{1}{2}\Theta_k \parallel \omega(k) \parallel^2\right]. \end{aligned} \tag{22}$$

The Lyapunov conditions are $V_{\text{sys}}(k+1) > 0$ and $\Delta V_{\text{sys}}(k+1) < 0$, and the next inequality condition should satisfy

$$\Phi_k = 1 - \frac{1}{2}\frac{\eta \parallel \omega(k) \parallel^2}{\mu + \parallel \omega(k) \parallel^2} > 0, \in \mathbb{R}^+ \tag{23}$$

Remark 1. *The details of Theorem 1 are discussed in* [27]*, the stability analysis of the PJM algorithm and the convergence of the estimation error are relevant for the global stability analysis of a data-driven control in a closed-loop configuration presented below in Theorem 3.*

Remark 2. *The upper and lower joints' velocities depend on the actuators' saturation, and mechanical damage in the robot needs to be prevented. Hence, the estimation error must be close to zero.*

For $k \to \infty$, $\omega(k) \approx 0$, and the stability condition Φ_k is accomplished by using (23). The saturation of the actuators $\omega_{sat}(k)$ depends on the robot characteristics (revolute or prismatic joints). Nevertheless, the damages into the KUKA youBot actuators can be prevented under the operating range $\omega(k)_{sat} = \pm 0.6 \frac{rad}{sec}$. From (23), the next inequality should satisfy

$$0 < \eta \leq \frac{2\left[\mu + \parallel \omega_{sat}(k) \parallel^2\right]}{\parallel \omega(k)_{sat} \parallel^2} \tag{24}$$

Accordingly, $\Phi_k \leq \mathrm{Y}$, where Y is the upper limit, when $\Phi_k(||\omega(k)||, \eta_{\mathrm{Max}})$, so that (22) becomes:

$$\Delta V_{\mathrm{sys}}(k+1) \leq -||\epsilon(k)||^2 \Theta_k \parallel \omega(k) \parallel^2 \Phi_k \tag{25}$$

Since, $\Delta V_{\mathrm{sys}}(k+1) < 0$ when (24) is fulfilled and $\mu > 0$, $\epsilon(k) \approx 0$, *i.e.*, $J_A^*(k)\hat{\mathbf{J}}_{\mathbf{A}}^{\dagger}(k) \approx I$ as $k \to \infty$. □

3. Control Law

The proposed control law is a proportional controller with adaptive gains for each ith axis of the robot in the x, y, and z directions. A neuro-fuzzy network was applied to tune the adaptive gains of the controller. The position error of the end-effector is

$$e_i(k+1) = \chi_i(k+1) - \chi_{d_i}(k+1) \tag{26}$$

where $\chi_i(k+1)$ represents the position of the end-effector axis, and $\chi_{d_i}(k+1)$ is the desired position task of the robot. The controller design uses the function $s_i(k+1)$ in the input of the neuro-fuzzy network in order to improve the tracking of the position error. Hence, the function is defined as:

$$s_i(k+1) = C_1 e_i(k+1) + C_2 e_i(k) \tag{27}$$

where C_1 and $C_2 \in \mathbb{R}^+$ are positive constants. The novelty of the proposed controller is that the function $s_i(k+1)$ is the input to the neuro-fuzzy network, and the output of the neuro-fuzzy network tunes the gains of the proportional controller K_{s_i}. The architecture of the adaptive gains is based on the fuzzy rules emulated network (FREN) structure proposed by [28]. The proposed topology based on a FREN to tune the adaptive gains for the proportional controller is depicted in Figure 2.

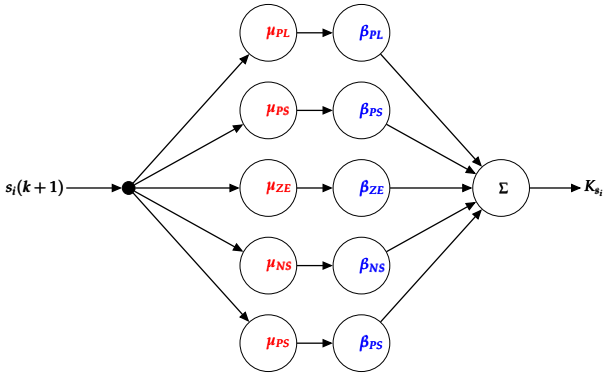


Figure 2. Artificial neuro-fuzzy network architecture.

The layers of the FREN are:

Layer 1: The function $s_i(k+1)$ is defined as the input to the neuro-fuzzy structure.

Layer 2: The second layer contains the linguistic variables as membership functions. The output at the jth node of this layer is calculated by ϕ_{ij} as

$$\phi_{ij} = \mu_{ij}(s_i) \tag{28}$$

where μ_{ij} denotes the linguistic variable at the jth node ($j = 1, 2, ..., N$) for the ith axis. The five linguistic variables are PL for positive large, PS for positive small, ZE for zero, NS for negative small, and NL for negative large.

Layer 3: This layer may be considered as a defuzzification step. In this case, the parameters β_{ij} remain constant.

Layer 4: This is the output of the artificial neuro-fuzzy network, where the gains of the controller are updated for the proportional controller as

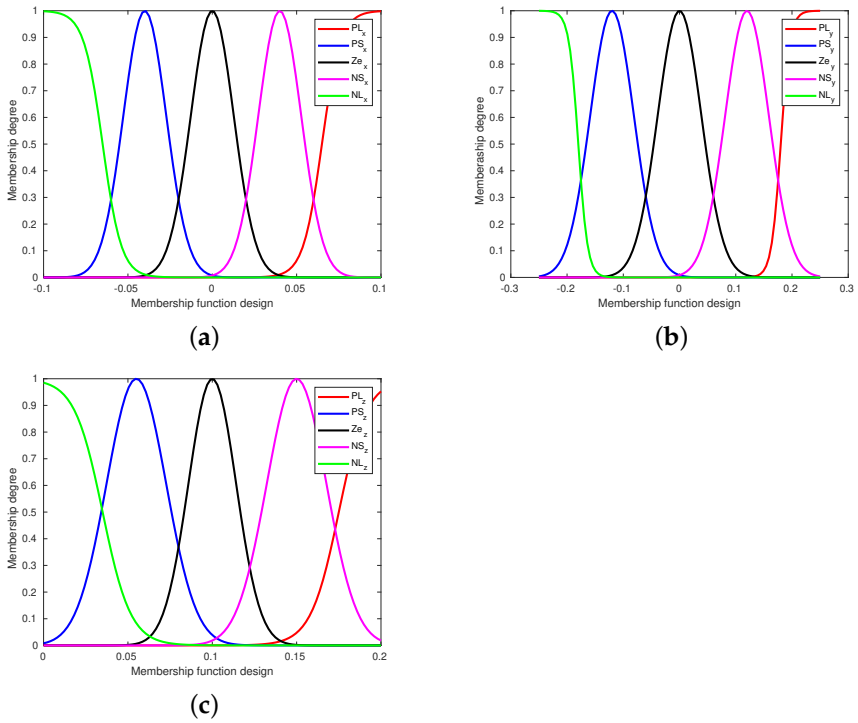
$$K_{s_i} = \sum_{j=1}^{N} \beta_{ij}\phi_{ij} \tag{29}$$

where N is the number of linguistic variables. The output of the FREN, K_{s_i}, contains positive values according to the membership functions taking values between 0 and 1, and positive values for $\beta_{ij} \in \mathbb{R}^+$.

The generalized rules depend on:

- If s_{ij} is j, then $K_{s_{ij}}$ is j.

Figure 3a–c show the memberships function for the x, y, and z axes, respectively. The five membership functions were designed by considering the input function to the artificial neuro-fuzzy network $s_i(k+1)$ in (27). As the function $s_i(k+1)$ was in terms of the position errors, this depended on the physical characteristics of the KUKA youBot, where the omnidirectional platform moved on the x–y plane and the robotic arm in the z direction.



(**a**) (**b**) (**c**)

Figure 3. Membership function design. (**a**) The 5 membership functions designed in terms of $s_x(k)$; (**b**) the 5 membership functions designed in terms of $s_y(k)$; (**c**) the 5 membership functions designed in terms of $s_z(k)$.

Theorem 2. *We can tune the parameters β_{xj}, β_{yj} and β_{zj} in Table 1 through the conditions in (A10).*

The novelties of the proposed neuro-fuzzy network are:

- The five membership functions are designed according to the physical characteristics of the robot axes.
- The proposed function $s_i(k+1)$ is the input of the neuro-fuzzy network. This avoids the chattering action, and it benefits the trajectory tracking control.
- The proposed neuro-fuzzy network needs to update only one parameter K_{s_i} for each robot axis in order to minimize the control errors.

Table 1. Value of β_{ij} parameters.

Parameters	s_x	s_y	s_z
β_{PL}	1	1	8.16
β_{NL}	0.75	0.75	7.5
β_{ZE}	0.5	0.5	5
β_{NS}	0.75	0.75	2.5
β_{NL}	1	1	1

3.1. Robot Control

Figure 4 shows the block diagram of the proposed control scheme. The function $s_i(k+1)$ is fed with the end-effector position error, which in turn is considered the input to the neuro-fuzzy FREN, where the tuning and adaptation of the gains K_{s_i} is achieved. Later, the gains K_{s_i}, the control error $e(k)$, and the estimated pseudoinverse Jacobian matrix $\hat{\mathbf{J}}_{\mathbf{A}}^{\dagger}(k)$ are used in the kinematic control law. The equivalent model is derived from the estimation of the Jacobian matrix through the PJM algorithm. By consequence, the loop closes when the end-effector position $\chi(k+1)$ reaches the desired position $\chi_d(k+1)$.

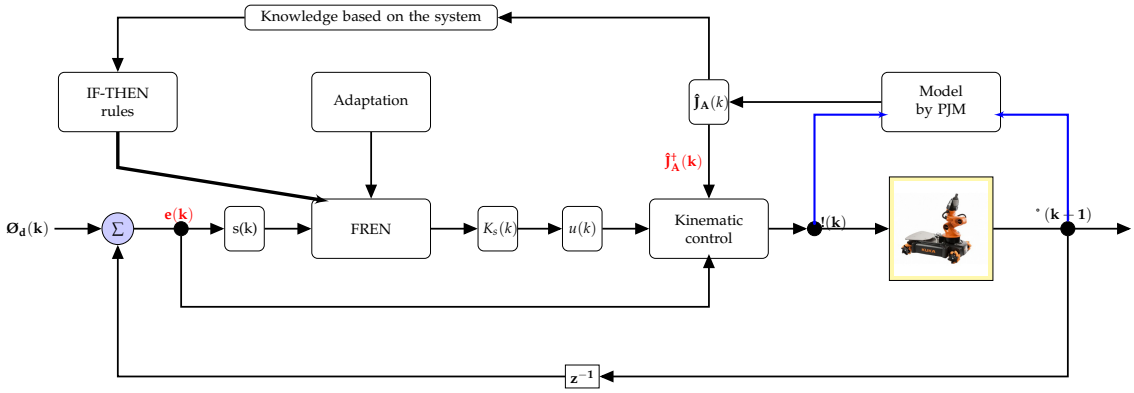


Figure 4. Control block diagram of the neuro-fuzzy control applying PJM algorithm.

The position error of the end-effector was defined in (26). Starting from this, the position of the robot's end-effector is now

$$\begin{aligned}\chi(k+1) =& \chi(k) + \bar{J}_A(k)\omega(k) \ \in \mathbb{R}^m \\ =& \chi(k) + T_s\hat{\mathbf{J}}_{\mathbf{A}}(k)\omega(k) + T_s\epsilon(k)\omega(k) \ \in \mathbb{R}^m \end{aligned} \tag{30}$$

where the terms $\bar{J}_A(k) = J_A^*(k)T_s \in \mathbb{R}^{m\times n}$ include the ideal Jacobian matrix and the sampling time, and from (2), the ideal Jacobian is $J_A^*(k) = \hat{\mathbf{J}}_{\mathbf{A}}(k) + \epsilon(k), \in \mathbb{R}^{m\times n}$. Substituting the current position (30) in the control error (26):

$$\begin{aligned} e(k+1) =& \chi(k) + \bar{J}_A(k)\omega(k) - \chi_d(k+1) \\ =& \chi(k) + J_A^*(k)\Delta q(k) - \chi_d(k+1) \end{aligned} \tag{31}$$

The pseudoinverse Jacobian matrix $\hat{\mathbf{J}}_{\mathbf{A}}^{\dagger}(\mathbf{k})$ resolves the inverse kinematics, the joint's velocities are the control signals

$$\omega(k) = -\hat{\mathbf{J}}_{\mathbf{A}}^{\dagger}(k)u(k) \ \in \mathbb{R}^n \tag{32}$$

and $u(k)$ contains the control law of u_x, $u_y(k)$, and $u_z(k)$ as follows

$$u(k) = [K_s e(k) - v_d(k+1)] \ \in \mathbb{R}^m \tag{33}$$

where $K_s \in \mathbb{R}^{m \times m}$ is a diagonal matrix which contains the adaptive gains K_{s_x}, K_{s_y} and K_{s_z}. The desired velocity $v_d(k+1) \in \mathbb{R}^m$ of the end-effector is included in the tracking control. The next equation solves the pseudoinverse Jacobian matrix $\hat{\mathbf{J}}_{\mathbf{A}}^{\dagger}(k)$ problem

$$\hat{\mathbf{J}}_{\mathbf{A}}^{\dagger}(k) = \hat{\mathbf{J}}_{\mathbf{A}}^{T}(k)\left[\hat{\mathbf{J}}_{\mathbf{A}}(k)\hat{\mathbf{J}}_{\mathbf{A}}^{T}(k) + \xi I\right]^{-1} \tag{34}$$

where the value of the damping factor is $\xi = 0.1$. The updated positions are

$$q(k+1) = q(k) + \omega(k)T_s \tag{35}$$

3.2. Controller Stability Analysis

Theorem 3. *For a closed-loop control using the PJM to estimate the Jacobian matrix of a discrete-time MIMO system such as the KUKA youBot, the error position of the end-effector converges to a vicinity of the origin for the trajectory tracking control. When it satisfies the Lipschitz condition,* $\| I - J_A^*(k)\hat{\mathbf{J}}_{\mathbf{A}}^{\dagger}(k) \| \leq P_k^{max}$, *and* $\epsilon(k) \approx 0$.

Proof. The candidate discrete Lyapunov function for a closed-loop control is

$$V(k+1) = \frac{1}{2}e(k+1)e^T(k+1) \tag{36}$$

for the stability analysis, the Lyapunov condition is $V(k+1) > 0$. The change in the Lyapunov function for the controller is

$$\begin{aligned} \Delta V(k+1) &= -K_s e(k)e^T(k)\left[I - \frac{1}{2}K_s\right] \\ &\quad + \frac{1}{2}\Omega_k\Omega_k^T \\ \Delta V(k+1) &\leq -\frac{1}{2} \| e(k) \|^2 + \frac{1}{2}\Gamma_2 \end{aligned} \tag{37}$$

Ω_k and Γ_2 are discussed in Appendix A, and Γ_2 is the upper-bounded value corresponding to the undefined sign terms in (A8) and (A9) within a compact set. Therefore, $\Delta V(k+1) < 0 \ \forall \ K_s \leq I$, i.e., $e(k)$ approaches a uniformly ultimately bounded (UUB) system. □

3.3. General Stability Analysis

The aim of this section is to demonstrate the stability analysis through a Lyapunov approach, which involves the dynamic of the estimated model and the control law in a closed-loop configuration. This means the Lyapunov candidate function includes the control and estimation errors.

Theorem 4. *Considering that the MIMO system in (3) satisfies the following criteria:*

- *Assumption 1: Lipschitz condition.*
- *Assumption 2: observability condition.*
- *Assumption 3: estimation error convergence.*
- *Corollary 1: initialization methodology.*
- *Corollary 2: bounded estimation error value.*
- *Theorem 1: estimation model stability analysis.*
- *Theorem 2: robot control stability analysis.*

If Assumptions 1–3, Corollaries 1 and 2, and Theorems 1 and 2 are fulfilled, then the general closed-loop stability of the system is ensured through the model and control Lyapunov functions. The control error converges to a vicinity of the origin and the estimation error approximates zero.

Proof. The stability proof is developed by the general Lyapunov function $V_G(k+1) > 0$ and its change in the Lyapunov function $\Delta V_G(k+1) < 0$.

Consider the following general Lyapunov function

$$\begin{aligned} V_G(k+1) =& V_{\text{sys}}(k+1) + V(k+1) \\ =& \frac{1}{2}\left[\epsilon(k+1)\epsilon^T(k+1) + e(k+1)e^T(k+1)\right] \end{aligned} \tag{38}$$

considering (19) and (36) to propose the general Lyapunov function, where $V_G(k+1) > 0$. The change in the general Lyapunov function is

$$\Delta V_G(k+1) = \Delta V_{\text{sys}}(k+1) + \Delta V(k+1) \tag{39}$$

and substituting (25) and (37) in (39), it is obtained that

$$\begin{aligned} \Delta V_G(k+1) \leq & -||\epsilon(k)||^2 \Theta_k \parallel \omega(k) \parallel^2 \Phi_k \\ & - \frac{1}{2} \parallel e(k) \parallel^2 + \frac{1}{2}\Gamma_2 \end{aligned} \tag{40}$$

This means $e(\epsilon(k))$ when $||\epsilon(k)|| = 0.143$ in Figure 1, and $e(k) = 0$, as long as $k \to \infty$ and Γ_2 is within a compact set.

Using Corollary 2,

$$\lim_{k\to\infty} ||\epsilon(k)|| = 0.143 \tag{41}$$

The estimation error converges at 4.34 s in Figure 1; meanwhile, the control error converges at 10 s in Figure 5b. This means the estimation error should converge before the control error. According to (30), the control error is in terms of the estimation error. Therefore, $V_G(k+1) > 0$, $\Delta V_G(k+1) < 0$ $\forall k$ and the Lyapunov stability analysis is guaranteed in a vicinity of the origin using (40). □

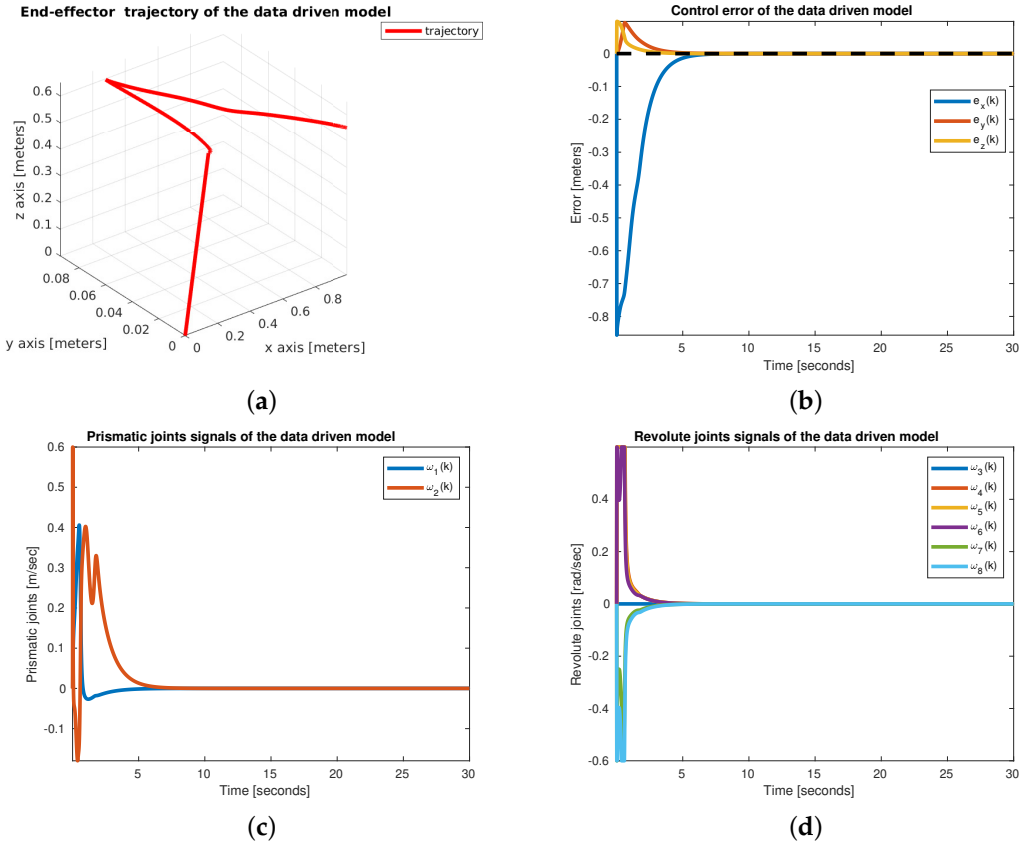


Figure 5. Simulation results for regulation control: (**a**) end-effector position, (**b**) control errors, (**c**) prismatic velocities, and (**d**) revolute velocities. (**a**) End-effector reaching a fixed position in the space and the performance in the x, y, and z directions; (**b**) control errors for the x, y and z directions of the end-effector; (**c**) performance of the prismatic velocities $\omega_1(k)$ and $\omega_2(k)$ in the mobile platform; (**d**) performance of the revolute velocities: $\omega_3(k)$ in the mobile platform and from $\omega_4(k)$ to $\omega_8(k)$ in the robotic arm.

4. Results

To start the simulations, it was necessary to consider the initialization conditions mentioned in Corollary 1 and according to the values selected in (10) for the estimated Jacobian matrix $\hat{\mathbf{J}}_{\mathbf{A}}(0)$. The operating values were within $\pm 0.6 \frac{rad}{sec}$ in order to avoid risky damage to the robot structure. The initial values for the prismatic and revolute joints were $\omega_{\text{pris}}(0) = 0.2 \frac{m}{sec}$ and $\omega_{\text{rev}}(0) = 0.2 \frac{rad}{sec}$.

The parameter settings were selected according to Theorems 1–3 in Table 2.

Table 2. Parameter settings for the robot model and control.

Parameters	Values	Remark
η_x	0.5726	(24)
η_y	0.5745	(24)
η_z	0.1045	(24)
μ	1	(24)
C_1	0.7	(A12)
C_2	0.5	(A12)
$\tilde{\xi}$	0.01	(12)

The performance of the data-driven control was validated by three different simulations.

The first simulation was the regulation control of the end-effector. That is, the end-effector reached a fixed desired position. Figure 5a shows the end-effector for a fixed position task. The convergence of the three position errors are observed in Figure 5b. The z-axis position error was the first to converge at 6 s. The x-axis error converged at 10 s. Finally, the y-axis error converged at 13 s. Figure 5c shows the signals of the prismatic velocities $\omega_1(k)$ and $\omega_2(k)$; the maximum linear velocity was 0.6 $\frac{m}{sec}$. Figure 5d depicts the revolute joints' signals from the robot base and the robotic arm. The maximum angular velocity was $0.6\frac{rad}{sec}$ for $\omega_6(k)$ and the minimum velocity was $-0.6\frac{rad}{sec}$, achieved for $\omega_8(k)$.

For the second simulation, while the end-effector reached the object position, a disturbance was applied in the desired position. This means the desired position was changed in order to probe the performance of the robot's control against disturbances. The trajectory of the end-effector is observed in Figure 6a, where the change in the trajectory once the objective was moved is clearly observed. The position error is shown in Figure 6b. It is seen that after 15 s, the disturbance in the end-effector position was introduced, increasing the control errors in the three axes. However, after 20 s, the errors in the three axes converged again. In the case of prismatic joints, it can be seen in Figure 6c that when applying the disturbance, the maximum linear velocity was $0.6\frac{m}{sec}$. On the other hand, the revolute joints reached their minimum and maximum value in $\pm$ 0.6 $\frac{rad}{sec}$ during the disturbance.

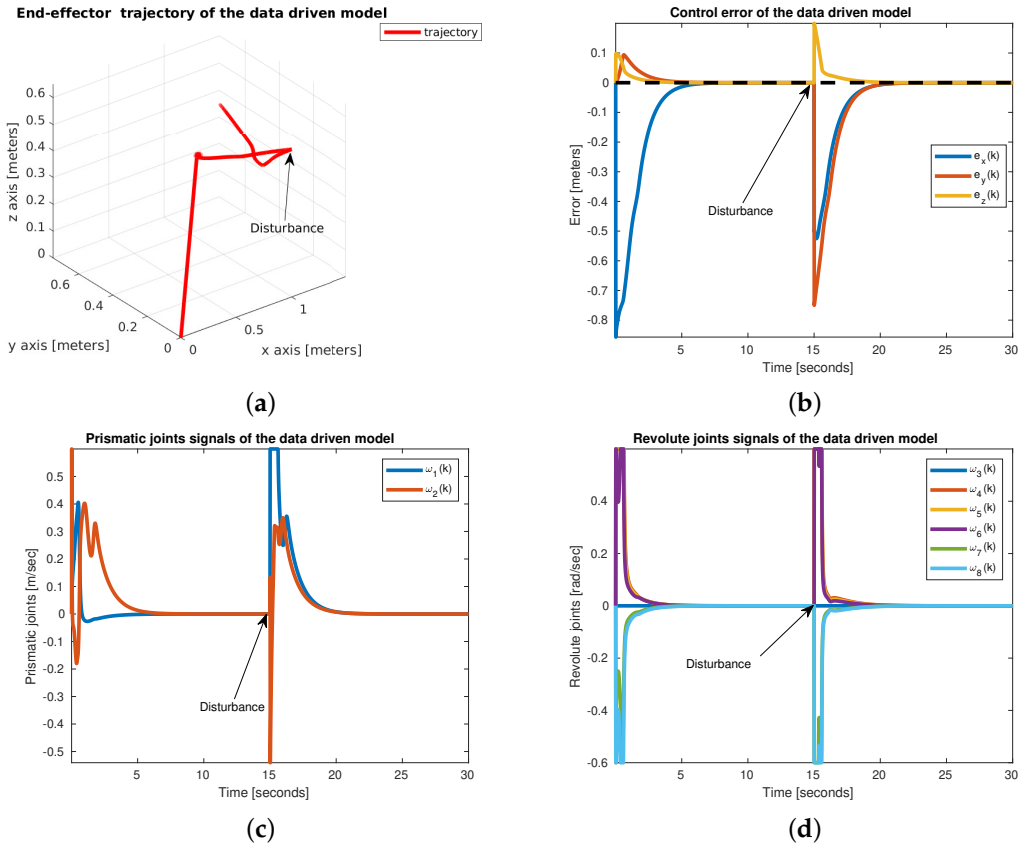


Figure 6. Simulation results for an external disturbance in the desired object position: (**a**) end-effector position, (**b**) control errors, (**c**) prismatic velocities, and (**d**) revolute velocities. (**a**) End-effector performance in the x, y, and z directions during an external disturbance; (**b**) control errors for the x, y, and z directions during an external disturbance; (**c**) performance of the prismatic velocities $\omega_1(k)$ and $\omega_2(k)$ in the mobile platform during an external disturbance; (**d**) performance of the revolute velocities, $\omega_3(k)$ in the mobile platform and from $\omega_4(k)$ to $\omega_8(k)$ in the robotic arm, during an external disturbance.

In the third simulation, the end-effector followed a circular trajectory. Figure 7a shows the position of the end-effector on its three axes. The x and y axes followed the trajectory,

since the z-axis was maintained constant; in Figure 7b is shown the position error of the end-effector during tracking control. Figure 7c depicts the periodic trajectory followed by the prismatic joints in the x–y plane. Figure 7d shows that the revolute joints' velocities remained close to zero in order to maintain the posture of the robotic arm. Figure 7e shows the circular trajectory of the end-effector.

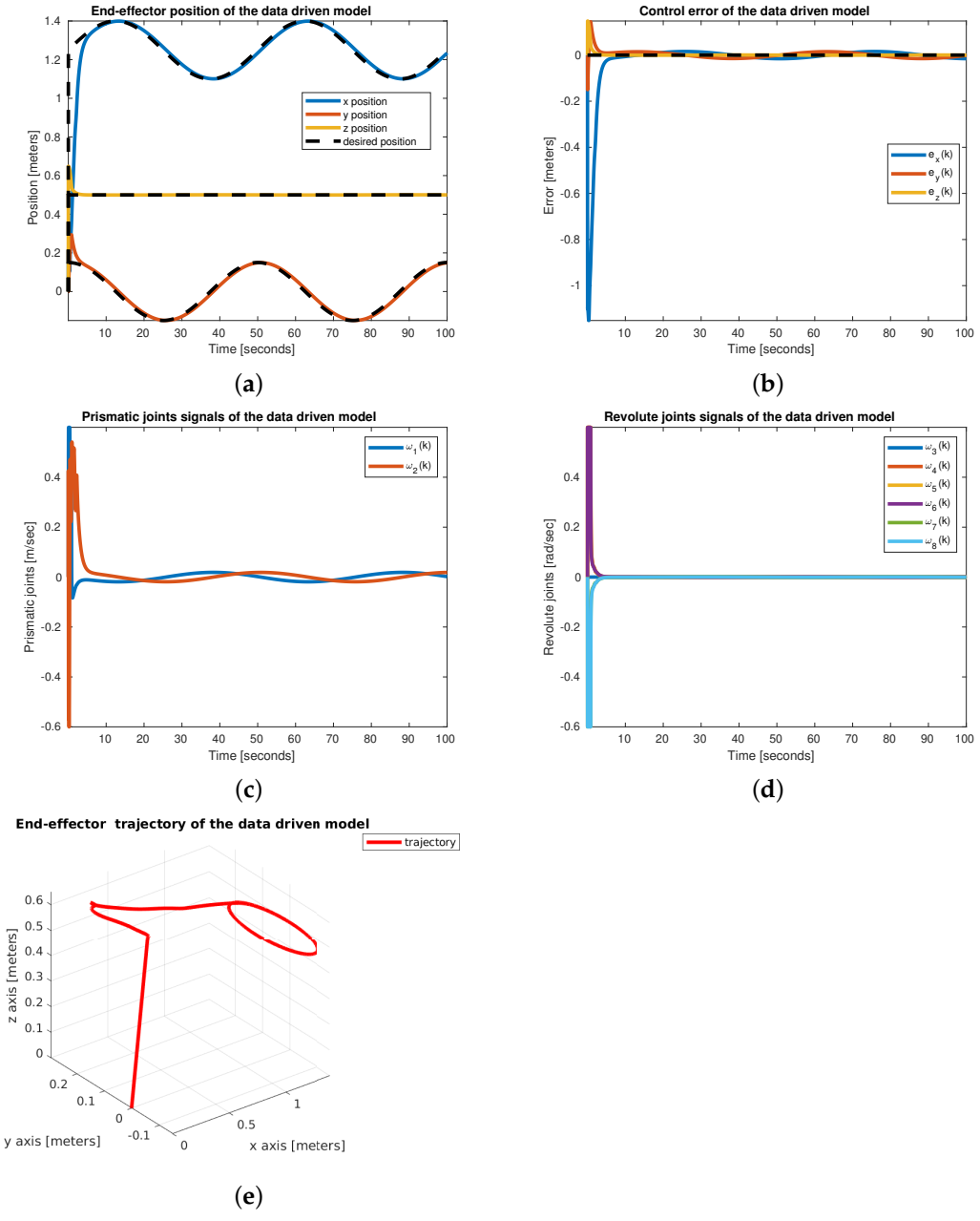


Figure 7. Simulation results for the trajectory tracking control: (**a**) end-effector position, (**b**) control errors, (**c**) prismatic velocities, (**d**) revolute velocities, and (**e**) the end-effector circular trajectory in space. (**a**) End-effector trajectory tracking control performance in the x, y and z directions; (**b**) control errors for the x, y, and z directions during the end-effector trajectory tracking control; (**c**) performance of the prismatic velocities $\omega_1(k)$ and $\omega_2(k)$ in the mobile platform during the end-effector trajectory tracking control; (**d**) performance of the revolute velocities, $\omega_3(k)$ in the mobile platform and from $\omega_4(k)$ to $\omega_8(k)$ in the robotic arm, during the end-effector trajectory tracking control; (**e**) end-effector trajectory.

The drawback of the proposed initialization method is to be able to identify the four criteria in Corollary 1. It is important to have an understanding of the kinematic constraint, matrix rank, and the stability analysis of the closed-loop control.

5. Conclusions

This work presented a methodology based on a generalized Jacobian matrix initialization applied to a data-driven model approach for a robotic system. A redundant robot is a class of discrete-time MIMO systems. From the data-driven model methods, the system is considered as unknown even from the beginning. The PJM algorithm requires the input–output signals to approximate the Jacobian matrix. Hence, the estimated Jacobian matrix represents the model for a first-order kinematic control in a redundant robot.

The initialization of the Jacobian matrix is an indispensable research topic for data-driven model and control, since the control error and estimation error guarantee their convergence. It was possible to determine the conditions for the initial values of the estimated Jacobian matrix. The conditions were closely tied to the Jacobian matrix rank, the holonomic constraint, the domain of the Jacobian matrix, and the guarantees of control and estimation errors' convergence. The proposed methodology introduced a specific procedure to identify and select the adequate set of initialization values of the estimated Jacobian matrix by applying a data-driven model in a closed-loop control.

Moreover,de novelty of the proposed proportional controller was the adaptation of its gains using a neuro-fuzzy architecture. The neuro-fuzzy network adapted only one parameter for each end-effector axis; hence, the control errors converged to zero. The control stability analysis included the convergence of the estimation and control errors. Moreover, the conditions of the initial Jacobian matrix, the PJM algorithm, and the proposed robot control were tested in simulations.

As future work, a comparison will be performed between the proposed Jacobian matrix initialization method and an artificial neural network learning approach. Furthermore, the validation of the data-driven modelling and control approach presented in this research will be extended to an experimental setup. Finally, the entire control of the robot pose will be considered, including the position and orientation of the end-effector.

Author Contributions: Conceptualization, J.G.-C.; methodology, A.B.M.-D.; software, J.S.G.-V.; validation, C.R.M.-V.; formal analysis, J.F.M.-V.; investigation, C.R.M.-V. and J.G.-C.; resources, J.S.G.-V.; data curation, A.B.M.-D.; writing—original draft preparation, J.G.-C.; writing—review and editing, J.S.G.-V.; visualization, J.F.M.-V.; supervision, A.B.M.-D.; project administration, C.T.; funding acquisition J.F.M.-V. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data and methods used in the research are presented in sufficient detail in the document for other researchers to replicate the work.

Acknowledgments: The authors gratefully acknowledge the financial support of PRODEP and CONACYT Mexico.

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

Appendix A

Proof of the Theorem 2. The candidate discrete-time Lyapunov function is:

$$V(k+1) = \frac{1}{2} e(k+1) e^T(k+1) \tag{A1}$$

The change in the discrete-time Lyapunov function is defined as:

$$\Delta V(k+1) = V(k+1) - V(k) \tag{A2}$$

The Lyapunov function in terms of the control error is

$$\Delta V(k+1) = \Delta e(k+1)\left[e(k) + \frac{1}{2}\Delta e(k+1)\right] \tag{A3}$$

Since $\chi_d(k+1) = \chi_d(k) + \Delta\chi_d(k+1)$, the position error (31) is simplified as follows:

$$\begin{aligned} e(k+1) =& \chi(k) + T_s\hat{\mathbf{J}}_{\mathbf{A}}(k)\omega(k) + T_s\epsilon(k)\omega(k) \\ & - \chi_d(k) - \Delta\chi_d(k+1) \end{aligned} \tag{A4}$$

for $J_A^*(k) = \hat{\mathbf{J}}_{\mathbf{A}}(k) + \epsilon(k)$. The change in error $\Delta e(k+1)$ can be converted into

$$\begin{aligned} \Delta e(k+1) =& T_s\hat{\mathbf{J}}_{\mathbf{A}}(k)\omega(k) + T_s\epsilon(k)\omega(k) \\ & - \Delta\chi_d(k+1) \\ \Delta e(k+1) =& T_s\hat{\mathbf{J}}_{\mathbf{A}}(k)\frac{\Delta q(k)}{T_s} + T_s\epsilon(k)\frac{\Delta q(k)}{T_s} \\ & - \Delta\chi_d(k+1) \\ \Delta e(k+1) =& \hat{\mathbf{J}}_{\mathbf{A}}(k)\Delta q(k) + \epsilon(k)\Delta q(k) - \Delta\chi_d(k+1) \\ =& - J_A^*(k)\hat{\mathbf{J}}_{\mathbf{A}}^{\dagger}(k)[K_s e(k) - \Delta\chi_d(k+1)] \\ & - \Delta\chi_d(k+1) \end{aligned} \tag{A5}$$

From [29], $\hat{\mathbf{J}}_{\mathbf{A}}^{\dagger}(k)$ is a full row rank matrix $\hat{\mathbf{J}}_{\mathbf{A}}(k) \in \mathbb{R}^{m\times n}$ with $m < n$, and it satisfies the condition $P_k = J_A^*(k)\hat{\mathbf{J}}_{\mathbf{A}}^{\dagger}(k)$ being a positive definite matrix using (11). Hence,

$$\begin{aligned} \Delta e(k+1) =& - P_k K_s e(k) + [P_k - I]\Delta\chi_d(k+1) \\ =& - P_k K_s e(k) + \Omega_k \end{aligned} \tag{A6}$$

Meanwhile, $\Omega_k = [P_k - I]\Delta\chi_d(k+1) \leq \Gamma_1$ if $\mathcal{B}_1 \triangleq P_k - I$, $\Gamma_1 \triangleq \lambda_{max}(\mathcal{B}_1) \parallel \Delta\chi_d(k+1) \parallel$. Replacing the change in control error into a change in the Lyapunov function yields

$$\begin{aligned} \Delta V(k+1) =& \Delta e(k+1)\left[e(k) + \frac{1}{2}\Delta e(k+1)\right]^T \\ =& - P_k K_s e(k) e^T(k)\left[I - \frac{1}{2}P_k K_s\right] \\ & + \Omega_k e^T(k)[I - P_k K_s] \\ & + \frac{1}{2}\Omega_k\Omega_k^T \end{aligned} \tag{A7}$$

where the term $\Omega_k e(k)$ is undefined in sign, and it is necessary to cancel it under the following condition

$$I - P_k K_s = 0 \tag{A8}$$

where as mentioned in Corollary 2, $P_k \approx I$ when $\epsilon(k) \approx 0$, by $K_s \approx I$.In addition, $\parallel \Omega_k\Omega_k^T \parallel^2 \leq \Gamma_2$ if $\mathcal{B}_2 = K_s - I$, $\Gamma_2 \triangleq \left[\lambda_{max}(\mathcal{B}_2) \parallel \Delta\chi_d(k+1) \parallel^2\right]$. The stability condition should satisfy

$$\begin{aligned} \Delta V(k+1) =& - K_s e(k) e^T(k)\left[I - \frac{1}{2}K_s\right] \\ & + \frac{1}{2}\Omega_k\Omega_k^T \end{aligned} \tag{A9}$$

where the Lyapunov stability condition is $\Delta V(k+1) < 0$. Considering that $K_s \in \mathbb{R}^{3\times 3}$ is a diagonal matrix with positive time-varying parameters K_{s_x}, K_{s_y}, and K_{s_z}, the membership

functions ϕ_{ij} contains values from zero to one and the β_{ij} parameters remain positive. In consequence, the β_{ij} parameters are set by considering the performance of the control gains K_{s_x}, K_{s_y}, and K_{s_z}. The maximum value of β_{ij} is determined by the maximum value of K_{s_i} in Figure A1 and $\phi_{ij} = 1$, where $\beta_{ij} = K_{s_i}\phi_{ij}^{-1}$. Therefore, the β_{ij} values are as follows

$$\begin{aligned} 0 < \beta_{xj} &\leq 1.06 \\ 0 < \beta_{yj} &\leq 1.00 \\ 0 < \beta_{zj} &\leq 8.16 \end{aligned} \tag{A10}$$

The selected values used for β_{xj}, β_{yj}, and β_{zj} in Table 1 are in agreement for condition (A10).

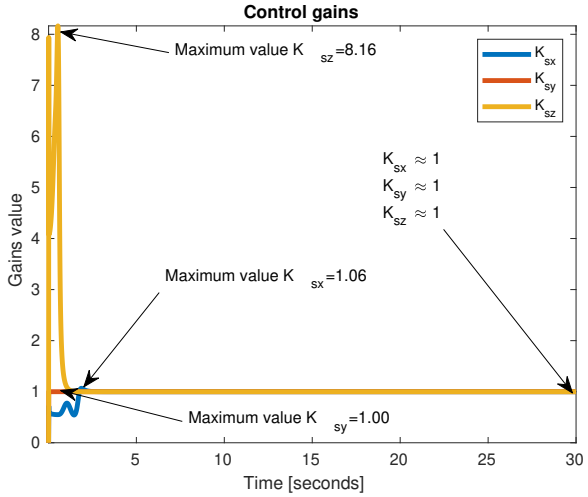


Figure A1. Adaptive gains K_{s_x}, K_{s_y} and K_{s_z}.

The tuning of the design parameters C_1 and C_2 involved in the function in (27) is presented below. The function $s_i(k+1)$ is the input to the artificial neuro-fuzzy network to adapt the gain K_{s_i}, where C_1 and $C_2 \in \mathbb{R}^+$ are positive constants. Thus, when $k \to \infty$, $\epsilon(k) \approx 0$, and $e(k) \to 0$, the function $s(k+1) = 0$ is as depicted in Figure A2. Then,

$$e(k+1) = -\frac{C_2}{C_1}e(k) \tag{A11}$$

In general, (A11) can be considered $e(k+1) = A_k e(k)$, thus the convergence can be guaranteed when $0 < A_k < 1$, where the condition is

$$0 < \left|\frac{C_2}{C_1}\right| < 1 \tag{A12}$$

in order to satisfy the condition $C_1 > C_2$.

Considering that $K_s \approx I$ and the term $I - \frac{1}{2}K_s$ must be positive to fulfil the Lyapunov stability condition $\Delta V(k+1) < 0$ in (A9),

$$0 < I - \frac{1}{2}K_s; \tag{A13}$$

hence, the next term satisfies the stability condition

$$K_s < 2I \tag{A14}$$

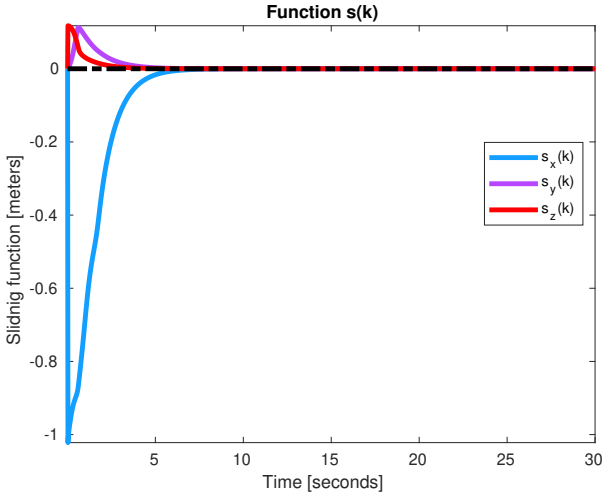


Figure A2. Function response: $s_x(k)$, $s_y(k)$ and $s_z(k)$.

Then, the Lyapunov condition $\Delta V(k+1)$ satisfies Theorem 3 in (36). It is possible to fulfil the condition according to (A9), where the term $\frac{1}{2}\Omega_k\Omega_k^T$ is bounded, the term $-K_s e(k)e^T(k)$ remains negative, and by $K_s \leq I$, the result is the positive definite matrix $\frac{1}{2}I$. (A9) is

$$\Delta V(k+1) \leq -\frac{1}{2} \parallel e(k) \parallel^2 + \frac{1}{2}\Gamma_2 \tag{A15}$$

by constructing $V(k+1) > 0$, see (A1); moreover, regarding (A15), $\Delta V(k+1) < 0$ in a vicinity of the origin. Therefore, $e(k)$ also approaches a compact set near to a vicinity of the origin. Lastly, the control (11) stabilizes the robotic system in (3). □

References

1. van der Veen, G.J.; van Wingerden, J.W.; Fleming, P.A.; Scholbrock, A.K.; Verhaegen, M. Global data-driven modeling of wind turbines in the presence of turbulence. *Control. Eng. Pract.* **2013**, *21*, 441–454. [CrossRef]
2. Treesatayapun, C. Fuzzy Rules Emulated Discrete-Time Controller Based on Plant Input-Output Association. *J. Control. Autom. Electr. Syst.* **2019**, *30*, 902–910. [CrossRef]
3. Hui, Y.; Chi, R.; Huang, B.; Hou, Z. Extended state observer-based data-driven iterative learning control for permanent magnet linear motor with initial shifts and disturbances. *IEEE Trans. Syst. Man, Cybern. Syst.* **2019**, *51*, 1881–1891. [CrossRef]
4. Liu, S.; Sun, J.; Ji, H.; Hou, Z.; Fan, L. Model Free Adaptive Control for the Temperature Adjustment of UGI Coal Gasification Process in Synthetic Ammonia Industry. In Proceedings of the 2020 IEEE 9th Data Driven Control and Learning Systems Conference (DDCLS), Liuzhou, China, 20–22 November 2020; pp. 976–981.
5. Ruiz-Martinez, O.; Mayo-Maldonado, J.; Escobar, G.; Valdez-Resendiz, J.; Maupong, T.; Rosas-Caro, J. Data-driven stabilizing control of DC–DC converters with unknown active loads. *Control. Eng. Pract.* **2020**, *95*, 104266. [CrossRef]
6. Treesatayapun, C. Data input-output adaptive controller based on IF-THEN rules for a class of non-affine discrete-time systems: The robotic plant. *J. Intell. Fuzzy Syst.* **2015**, *28*, 661–668. [CrossRef]
7. Chen, F.; Selvaggio, M.; Caldwell, D.G. Dexterous Grasping by Manipulability Selection for Mobile Manipulator with Visual Guidance. *IEEE Trans. Ind. Inform.* **2018**, *15*, 1202–1210. [CrossRef]
8. Treesatayapun, C. Prescribed performance of discrete-time controller based on the dynamic equivalent data model. *Appl. Math. Model.* **2020**, *78*, 366–382. [CrossRef]
9. Yu, X.; Hou, Z.; Polycarpou, M.M.; Duan, L. Data-driven iterative learning control for nonlinear discrete-time MIMO systems. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *32*, 1136–1148. [CrossRef]
10. Xiong, S.; Hou, Z. Model-Free Adaptive Control for Unknown MIMO Nonaffine Nonlinear Discrete-Time Systems with Experimental Validation. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *33*, 1727–1739. [CrossRef]
11. Liu, S.; Hou, Z.; Zhang, X.; Ji, H. Model-free adaptive control method for a class of unknown MIMO systems with measurement noise and application to quadrotor aircraft. *IET Control. Theory Appl.* **2020**, *14*, 2084–2096. [CrossRef]
12. Bruder, D.; Fu, X.; Gillespie, R.B.; Remy, C.D.; Vasudevan, R. Data-driven control of soft robots using Koopman operator theory. *IEEE Trans. Robot.* **2020**, *37*, 948–961. [CrossRef]
13. Guo, Y.; Hou, Z.; Liu, S.; Jin, S. Data-Driven Model-Free Adaptive Predictive Control for a Class of MIMO Nonlinear Discrete-Time Systems With Stability Analysis. *IEEE Access* **2019**, *7*, 102852–102866. [CrossRef]

14. Li, M.; Kang, R.; Branson, D.T.; Dai, J.S. Model-free control for continuum robots based on an adaptive kalman filter. *IEEE/ASME Trans. Mechatronics* **2018**, *23*, 286–297. [CrossRef]
15. Treesatayapun, C. A discrete-time stable controller for an omni-directional mobile robot based on an approximated model. *Control. Eng. Pract.* **2011**, *19*, 194–203. [CrossRef]
16. Aly, A.A.; The Vu, M.; El-Sousy, F.F.M.; Alotaibi, A.; Mousa, G.; Le, D.-N.; Mobayen, S. Fuzzy-Based Fixed-Time Nonsingular Tracker of Exoskeleton Robots for Disabilities Using Sliding Mode State Observer. *Mathematics* **2022**, *10*, 3147. [CrossRef]
17. Chen, D.; Zhang, Y.; Li, S. Tracking control of robot manipulators with unknown models: A Jacobian-matrix-adaption method. *IEEE Trans. Ind. Inform.* **2018**, *14*, 3044–3053. [CrossRef]
18. Treesatayapun, C. Discrete-time adaptive controller for unfixed and unknown control direction. *IEEE Trans. Ind. Electron.* **2017**, *65*, 5367–5375. [CrossRef]
19. Zaplana, I.; Hadfield, H.; Lasenby, J. Singularities of serial robots: Identification and distance computation using geometric algebra. *Mathematics* **2022**, *10*, 2068. [CrossRef]
20. Hou, Z.; Jin, S. Data-driven model-free adaptive control for a class of MIMO nonlinear discrete-time systems. *IEEE Trans. Neural Netw.* **2011**, *22*, 2173–2188.
21. Hou, Z.; Chi, R.; Gao, H. An overview of dynamic-linearization-based data-driven control and applications. *IEEE Trans. Ind. Electron.* **2016**, *64*, 4076–4090. [CrossRef]
22. Hou, Z.; Zhu, Y. Controller-dynamic-linearization-based model free adaptive control for discrete-time nonlinear systems. *IEEE Trans. Ind. Inform.* **2013**, *9*, 2301–2309. [CrossRef]
23. Prabhu, V.; Kuppusamy, P.; Karthikeyan, A.; Varatharajan, R. Evaluation and analysis of data driven in expectation maximization segmentation through various initialization techniques in medical images. *Multimed. Tools Appl.* **2018**, *77*, 10375–10390. [CrossRef]
24. Yang, G.; Stewart, C.V.; Sofka, M.; Tsai, C.L. Registration of challenging image pairs: Initialization, estimation, and decision. *IEEE Trans. Pattern Anal. Mach. Intell.* **2007**, *29*, 1973–1989. [CrossRef] [PubMed]
25. Ji, P.; Li, C.; Ma, F. Sliding Mode Control of Manipulator Based on Improved Reaching Law and Sliding Surface. *Mathematics* **2022**, *10*, 1935. [CrossRef]
26. Spong, M.; Vidyasagar, M. Robust linear compensator design for nonlinear robotic control. *IEEE J. Robot. Autom.* **1987**, *3*, 345–351. [CrossRef]
27. Gómez, J.; Morales, A.; Treesatayapun, C.; Muñiz, R. Data-driven-modelling and Control for a Class of Discrete-Time Robotic System Using an Adaptive Tuning for Pseudo Jacobian Matrix Algorithm. In *Advances in Computational Intelligence, Proceedings of the 21st Mexican International Conference on Artificial Intelligence, MICAI 2022, Monterrey, Mexico, 24–29 October 2022*; Springer: Berlin/Heidelberg, Germany, 2022; Part II, pp. 291–302.
28. Treesatayapun, C.; Uatrongjit, S. Adaptive controller with fuzzy rules emulated structure and its applications. *Eng. Appl. Artif. Intell.* **2005**, *18*, 603–615. [CrossRef]
29. Dietrich, A.; Ott, C.; Albu-Schäffer, A. An overview of null space projections for redundant, torque-controlled robots. *Int. J. Robot. Res.* **2015**, *34*, 1385–1400. [CrossRef]