algorithms

MDPI

Article

Evaluation of Neural Network Effectiveness on Sliding Mode Control of Delta Robot for Trajectory Tracking

Anni Zhao [1], Arash Toudeshki [1], Reza Ehsani [1], Joshua H. Viers [2] and Jian-Qiao Sun [1,*]

[1] Department of Mechanical Engineering, University of California, Merced, CA 95343, USA; azcoco2021@gmail.com (A.Z.); amohammaditoudeshk@ucmerced.edu (A.T.); rehsani@ucmerced.edu (R.E.)
[2] Department of Civil & Environmental Engineering, School of Engineering, University of California, Merced, CA 95343, USA; jviers@ucmerced.edu
* Correspondence: jqsun@ucmerced.edu

Abstract: The Delta robot is an over-actuated parallel robot with highly nonlinear kinematics and dynamics. Designing the control for a Delta robot to carry out various operations is a challenging task. Various advanced control algorithms, such as adaptive control, sliding mode control, and model predictive control, have been investigated for trajectory tracking of the Delta robot. However, these control algorithms require a reliable input–output model of the Delta robot. To address this issue, we have created a control-affine neural network model of the Delta robot with stepper motors. This is a completely data-driven model intended for control design consideration and is not derivable from Newton's law or Lagrange's equation. The neural networks are trained with randomly sampled data in a sufficiently large workspace. The sliding mode control for trajectory tracking is then designed with the help of the neural network model. Extensive numerical results are obtained to show that the neural network model together with the sliding mode control exhibits outstanding performance, achieving a trajectory tracking error below 5 cm on average for the Delta robot. Future work will include experimental validation of the proposed neural network input–output model for control design for the Delta robot. Furthermore, transfer learnings can be conducted to further refine the neural network input–output model and the sliding mode control when new experimental data become available.

Keywords: delta robot; sliding mode control; neural networks

Citation: Zhao, A.; Toudeshki, A.; Ehsani, R.; Viers, J.H.; Sun, J.-Q. Evaluation of Neural Network Effectiveness on Sliding Mode Control of Delta Robot for Trajectory Tracking. *Algorithms* **2024**, *17*, 113. https://doi.org/10.3390/a17030113

Academic Editor: Andres Iglesias Prieto

Received: 31 January 2024
Revised: 20 February 2024
Accepted: 6 March 2024
Published: 8 March 2024

1. Introduction

Delta robots are fast, accurate, and versatile, suitable for tasks like assembly, pick-and-place, sorting, and classification. The Delta robot is a parallel robot with three arms connected to an end effector mounted on its base, exhibiting an over-actuated configuration. Lagrangian dynamics, inverse kinematics, and screw theory are three common approaches to model the Delta robot [1–3]. Knowing the dynamic model is essential for motion planning and trajectory tracking. It allows the controller to calculate the required joint inputs to achieve a desired end effector position. The control design of a Delta robot is challenging due to its high level of non-linearity and complicated kinematic and dynamic model. Therefore, plenty of nonlinear and adaptive control algorithms have been implemented for trajectory tracking and disturbance rejection of the Delta robot. Sliding mode control is one of the most popular nonlinear control algorithms [4,5]. This algorithm has been extensively applied to the trajectory tracking of the Delta robot combined with the fuzzy neural network [6], nonlinear proportional-derivative (PD) control [7], and synergetic control [8]. Additionally, Radial Basis Function Neural Networks (RBFNNs) combined with sliding mode control were adopted in [8,9] to compensate for the unknown disturbances. The work reported in [10] proposed an online estimation approach to compensate for various uncertainties in the Delta robot and to improve the tracking performance. An

adaptive active disturbance rejection control was adopted for the output-based robust trajectory tracking of the Delta robot in [11]. Iterative learning control has also been applied to the trajectory tracking of the Delta robot with a high performance. Boundjedir, C.E. and Bouri, M. [12] proposed a PD-type ILC combined with PD control to improve the iteration performance. Model-free iterative learning control was further implemented on the Delta robot with non-repetitive trajectories [13]. Moreover, a model reference adaptive control has also been adopted for control of the Delta robot. Combining an identified linear model with mode reference adaptive control has shown a significant performance improvement compared to the three commonly used methods: PID, adaptive control, and sliding mode control algorithms [14].

Besides traditional control algorithms, neural networks were also adopted to improve the trajectory-tracking performance of the Delta robot. An inverse kinematic controller using neural networks was presented in [15] for trajectory control of a Delta robot in real-time by making use of data collected from simulation. Ref. [16] proposed a data-driven optimal control algorithm by approximating the solution of the Hamilton–Jacobi–Bellman equation using neural networks. A neural network model was proposed in [17] to approximate the inverse kinematics of the Delta robot with randomly sampled experimental data and was implemented on the hardware in an open-loop control for trajectory tracking.

Most of the advanced control algorithms mentioned above are either data-driven or model-based. Model-based control algorithms are highly dependent on the accuracy of the model. Therefore, they have high requirements for control robustness in terms of model uncertainties and disturbances. On the other hand, data-driven algorithms operate independent of model accuracy requirements. Given the paramount importance to safety considerations during normal operations, ensuring robustness and maintaining optimal performance pose significant challenges in the control design of Delta robots employing data-driven algorithms. Moreover, another challenge in Delta robot control is the singularity [18,19], which occurs when the end effector is positioned at specific locations where the kinematic model degenerates and the joint angles become indeterminate, which can lead to erratic behavior and potential damage to the robot. Therefore, understanding the robot's geometry and workspace is crucial to avoid singularities, which can cause significant problems during operation.

In this study, we consider a Delta robot equipped with stepper motors as inputs. Once the desired trajectory of the end effector is determined, inverse kinematics can be adopted to solve for the joint inputs analytically or numerically using optimization methods. Moreover, control algorithms can be implemented to further improve the reliability of the motion and the trajectory tracking accuracy of the Delta robot. The objective of this study is to develop a hybrid of data-driven and model-based sliding mode control algorithms instead of using the conventional optimal control to improve the trajectory tracking performance. The inverse kinematics of the Delta robot is analytically difficult to express in the control-affine form, which complicates the control design process. Therefore, to deal with this issue, we propose that neural networks are used to create a control-affine nonlinear input–output dynamic model of the Delta robot, considering the stepper motor angles and velocities as inputs. The neural network input–output model is trained with randomly sampled data in a sufficiently large workspace of the Delta robot. Furthermore, the model-based sliding mode control for trajectory tracking is designed based on the neural network model. The proposed neural networks make full use of the inverse kinematics while the sliding mode control delivers excellent performance in trajectory tracking for a Delta robot. We should point out that the proposed control-affine nonlinear dynamic model of the Delta robot is not derivable either from Lagrangian dynamics or inverse kinematics. It is completely data-driven and can be updated once new experimental data become available. This is also one of the contributions from our work.

Section 2 presents the architecture of the Delta robot and its inverse kinematics. Section 3 presents the control-affine nonlinear dynamic model in terms of the neural networks. Section 4 presents the sliding mode control with the neural network model and extensive numerical simulation results to validate the proposed model and control design. Section 5 concludes this paper.

2. Dynamic Model of Delta Robot

The mechanical structure of a three-DOF Delta robot is shown in Figure 1. Figure 2 shows the geometry and coordinates of the Delta robot in 2D and 3D views. A typical Delta robot is composed of a fixed platform, a moving platform, three active arms, and three passive arms connected to an end effector mounted on the base. The active arms of the robot are driven by the rotation actuators, which are stepper motors in this study. The parameters of the Delta robot are given in Table 1, and the specifications of components used for fabricating the Delta robot can be found in [17].

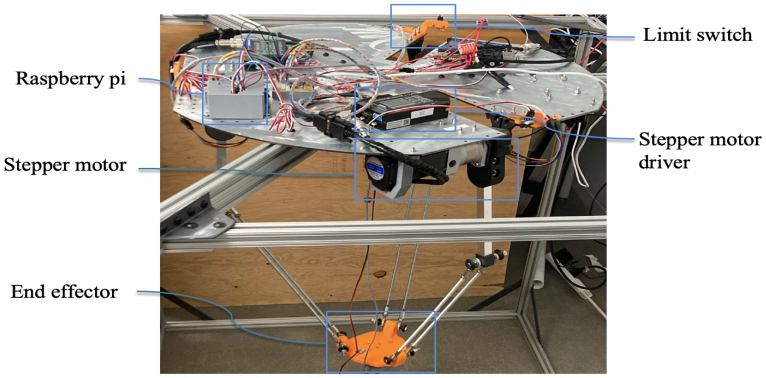

Figure 1. Hardware setup of the Delta robot.

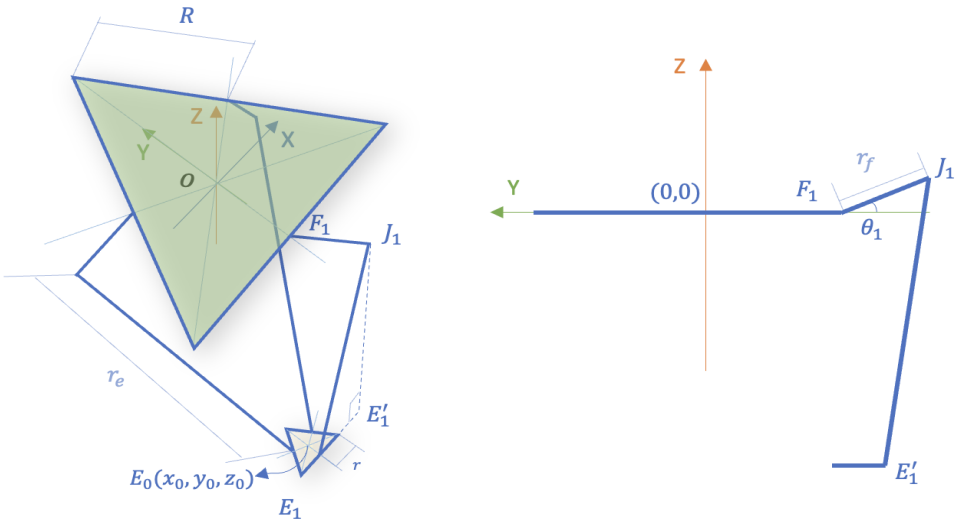

Figure 2. The geometric configuration and coordinate framework required to model the inverse kinematics of the Delta robot. **Left**: geometric structure of the Delta robot in 3D. **Right**: two-dimensional projection of geometry and coordinates of the Delta robot joints and links.

Table 1. Parameters of the Delta robot.

Description	Notation	Value
Radius of the fixed platform	R	0.325 m
Radius of the moving platform	r	0.075 m
Length of the active arm	r_f	0.5 m
Length of the passive arm	r_e	0.25 m
Mass of the active arm	m_f	0.205 kg
Mass of the passive arm	m_e	0.153 kg
Mass of the end effector	m_b	0.653 kg

Inverse Kinematics

Inverse kinematics stands out as one of the most widely adopted algorithms for modeling the Delta robot. Solving the inverse kinematics involves deriving the geometrical relationships between its links and joints. From Figure 2, the relationship between the link positions and angles θ_i can be given as

$$\theta_i = \arctan\left(\frac{Z_{Ji}}{Y_{Fi} - Y_{Ji}}\right) \tag{1}$$

where $i = 1, 2, 3$; Y_{Fi} and Y_{Ji} are the positions of the points F_i and J_i in the Y direction; and Z_{Fi} and Z_{Ji} are the positions of the points F_i and J_i in the Z direction. The positions of the joints Y_{Fi} and Y_{Ji} satisfy the geometric constraints of the Delta robot as described in [17].

The inverse kinematics describes how the angles of the joints relate to the positions of the end effector. The joint velocities $\dot{\theta}_i$ are not considered in Equation (1). In this study, the control inputs include angles and joint velocities $[\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}]$ of the stepper motors. Hence, the dynamic input–output model of the Delta robot should involve joint velocities. One way to compute the joint velocities is by using the Jacobian matrix J that relates to the joint velocities $\dot{\boldsymbol{\theta}} = [\dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3]^T$ and the end effector position velocities $\mathbf{v} = [\dot{x}, \dot{y}, \dot{z}]^T$ [20]:

$$\dot{\boldsymbol{\theta}} = J(\mathbf{r})\mathbf{v} \tag{2}$$

where $\mathbf{r} = (x, y, z)^T$ describes the centroid position of the end effector. Equations (1) and (2) suggest that the joint velocities are highly nonlinear and complex functions of the positions and its velocities of the end effector. In the data-driven study, the joint velocities can be computed using finite difference approximation as, for example,

$$\dot{\theta}_i(k) = (\theta_i(k) - \theta_i(k-1))/\Delta t \tag{3}$$

where k is the discrete time, i represents the stepper motor's index number and Δt is the sample time. However, as discussed in the Section 1, this information alone is not enough to help us build a dynamic input–output model for the Delta robot. Furthermore, if the finite difference method or Jacobian matrix method is used to compute joint velocities, errors in the computed velocities can accumulate over time, leading to a poor position prediction. This affects the performance of trajectory tracking control.

It is commonly known that the control-affine model of a system provides a convenient platform for control design. In this study, we propose to develop a control-affine nonlinear model of the Delta robot by utilizing neural networks. Neural networks can learn and model the underlying physical relationship between the inputs and outputs. A rich set of input–output data from the forward model of the Delta robot can be used to train neural networks in order to build the control-affine nonlinear dynamic model of the Delta robot without the need for the Jacobian matrix, finite difference, and inverse kinematics. The proposed control-affine model is completely data-driven and is intended for control design consideration. The

model is not derivable from Newton's law or Lagrange's equation. The input–output data for training the neural networks are randomly sampled in a sufficiently large workspace. Moreover, the neural network model can be further updated by incorporating the concept of transfer learning when new experimental data become available.

3. Neural Network Model

The nonlinear control-affine model with stepper motor joint angles and velocities as inputs makes the control design relatively straightforward and is given in the following.

$$\ddot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \dot{\mathbf{x}}) + \mathbf{g}(\mathbf{x}, \dot{\mathbf{x}})\mathbf{u} \tag{4}$$

where $\mathbf{x} = [p_x, p_y, p_z]^T$, $\mathbf{u} = [\theta_1, \theta_2, \theta_3, \dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3]^T$, $\mathbf{f}(\cdot) \in \mathbb{R}^{3\times 1}$, and $\mathbf{g}(\cdot) \in \mathbb{R}^{3\times 6}$ are nonlinear functions of their arguments, and p_x, p_y, and p_z represent the position of the end effector at the centroid along the x, y, and z axes.

As pointed out before, the control-affine form of the model is not directly derivable from Newton's law, Lagrangian dynamics, or inverse kinematics. Although various dynamic models for the Delta robot that use the computed-torque approach are available in the literature, they cannot be applied to this study, since the inputs of the Delta robot are only the joint angles and velocities. Here, the neural networks are adopted to approximate the nonlinear functions of the dynamic model for the Delta robot. Specifically, two neural networks are constructed to approximate the nonlinear functions $\mathbf{f}(\cdot)$ and $\mathbf{g}(\cdot)$. Notably, with an extra hidden layer and an additional input, a single neural network can be customized to approximate the two nonlinear functions. The structure of the proposed neural network model for the Delta robot is shown in Figure 3.

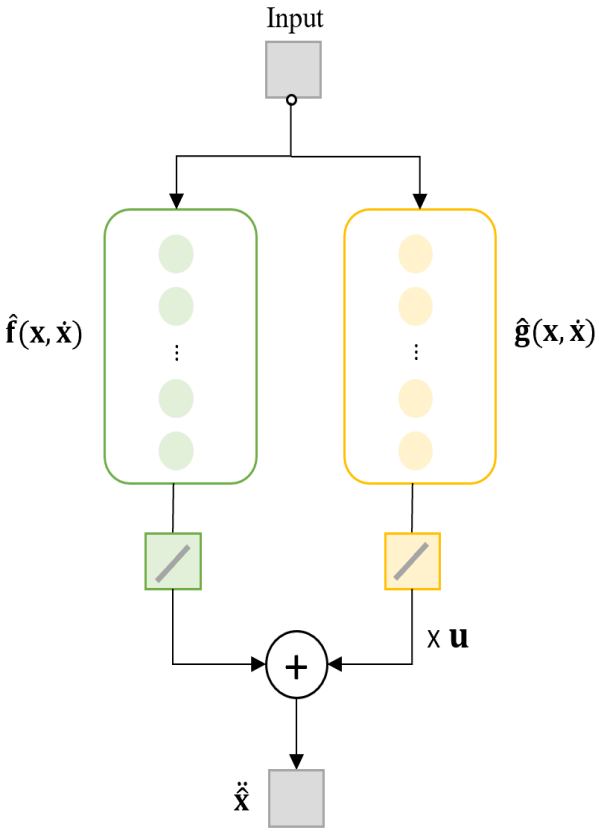

Figure 3. A flowchart illustrating the neural network model of the Delta robot.

The inputs for neural networks are $\mathbf{x}$, $\dot{\mathbf{x}}$, and $\mathbf{u}$. The output for the neural networks is $\ddot{\mathbf{x}}$, which can be expressed in terms of the two neural networks $\hat{\mathbf{f}}(\mathbf{x}, \dot{\mathbf{x}})$ and $\hat{\mathbf{g}}(\mathbf{x}, \dot{\mathbf{x}})$:

$$\ddot{\mathbf{x}}(n) = \hat{\mathbf{f}}(\mathbf{x}(n), \dot{\mathbf{x}}(n)) + \hat{\mathbf{g}}(\mathbf{x}(n), \dot{\mathbf{x}}(n))\mathbf{u} \tag{5}$$

where n is the sample time. The loss function is defined as

$$\mathbf{J} = \frac{1}{2}\sum_{n=1}^{n_s}(\ddot{\hat{\mathbf{x}}}(n) - \ddot{\mathbf{x}}(n))^2 \tag{6}$$

where $\ddot{\hat{\mathbf{x}}}(n)$ is the prediction from neural networks and $\ddot{\mathbf{x}}(n)$ is the system response. Data can be collected from simulations or experiments, and n_s is the total number of sampled points.

Detailed information of the number of hidden layers, type of activation function, and number of neurons are shown in Table 2. Here, the sigmoid neuron is chosen as the activation function. Other activation functions such as tanh or ReLU can also be considered based on the universal approximation theorem of neural networks. The number of training epochs is set as 20,000. The chosen optimization algorithm is stochastic gradient descent (SGD). Dropout with a frequency of 0.5 is adopted to prevent overfitting. The SGD algorithm is one of the most popular stochastic optimization methods in the machine learning community [21]. In this study, it has been adopted as the optimization algorithm to train the neural networks. The stochastic gradient descent maintains a single learning rate to update all weights compared to Adam [22]. The loss value reached 0.000281 after training for 20,000 epochs.

A remark on the choices of various hyper-parameters for the neural networks is in order. We have to say that the parameters we used are not guaranteed to be optimal. Since there is no established general rule for selecting the hyper-parameters of neural networks, some level of trial and error is involved with specific data set of different applications.

Table 2. Summary of neural networks.

Function	No. of Hidden Layers	Activation Function	No. of Neurons
$\hat{\mathbf{f}}(\mathbf{x})$	1	sigmoid	100
$\hat{\mathbf{g}}(\mathbf{x})$	1	sigmoid	100

With the help of the dynamic model built in Section 2, a SimScape model of the Delta robot is developed and used to generate the data of time series $\mathbf{x}, \dot{\mathbf{x}}, \ddot{\mathbf{x}}$ and $\dot{\theta}_i$. The SimScape model of the Delta robot is shown in Figure 4 and created based on the actual Delta robot that is to be used for the experiment, as shown in Figure 1 and Table 1. The parameters of this modeled Delta robot are digital twins of those listed in Table 1. To generate training data, we randomly sample joint angles in a bounded range of $\theta_i \in [-30^\circ, 120^\circ]$, as shown in Figure 5, in order to cover the typical workspace of the Delta robot. A total of 2500 data points are sampled in the workspace, and 75% of the data are chosen as training data set, and 25% of the data are chosen as validation data set. The corresponding positions of the end effector in 3D are shown in Figure 6. The boundary of these points is associated with the maximum of the positions in the workspace of the end effector of the targeted Delta robot.

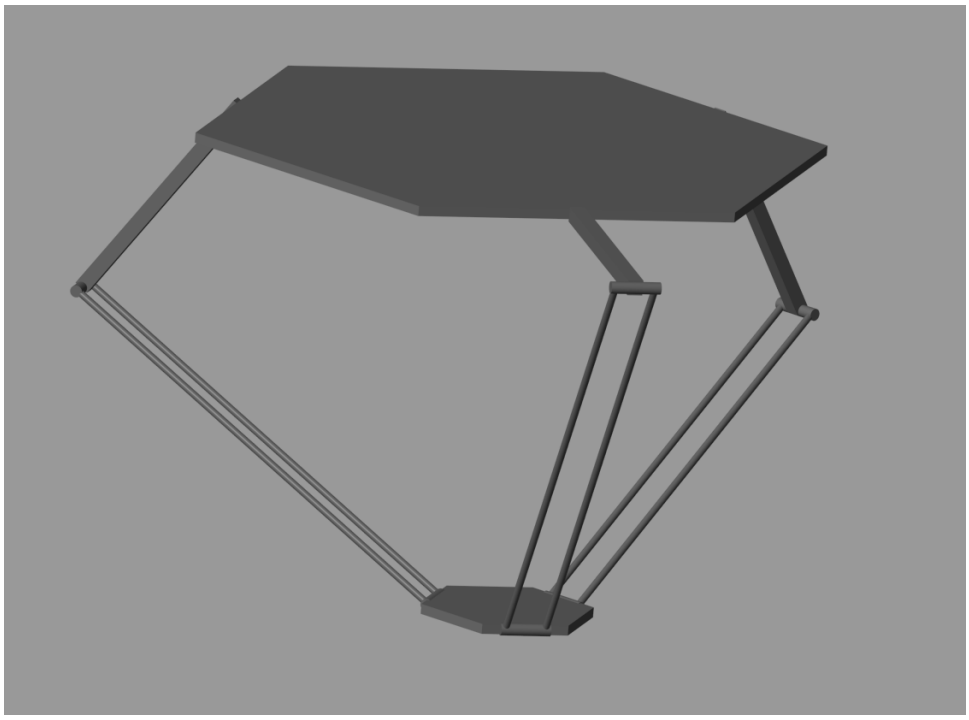

Figure 4. SimScape model of Delta robot.

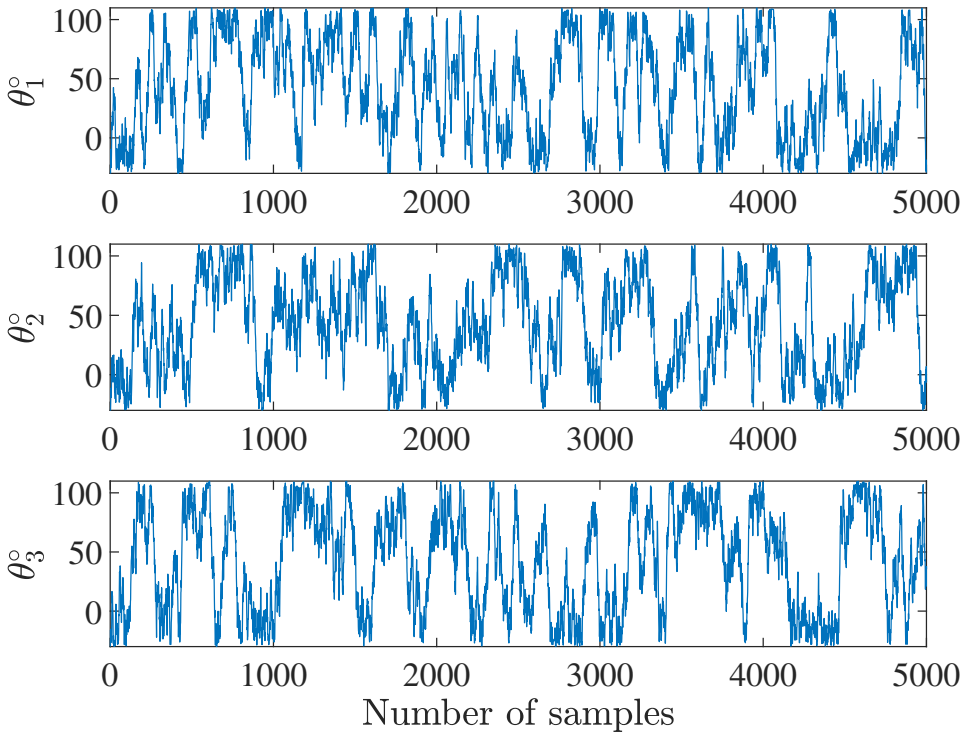

Figure 5. Random joint angles for SimScape Delta robot model. Top: joint angle θ_1 for stepper motor 1. Middle: joint angle θ_2 for stepper motor 2. Bottom: joint angle θ_3 for stepper motor 3.

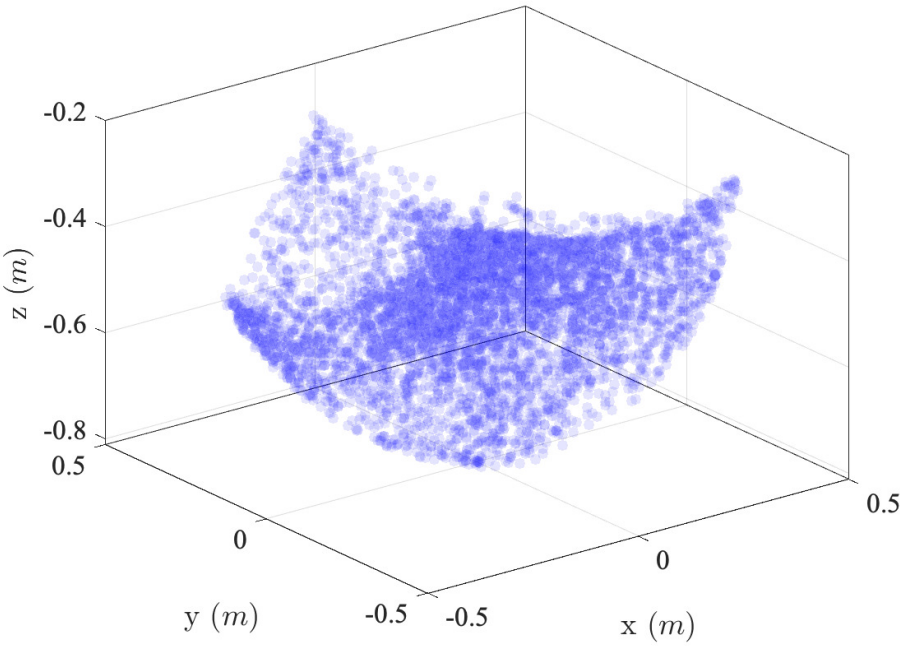

Figure 6. Random sampled points in the 3D workspace of the Delta robot.

4. Sliding Mode Control

Recall the control-affine nonlinear model in terms of the neural networks in Equation (5). We have

$$\ddot{\mathbf{x}} = \hat{\mathbf{f}}(\mathbf{x}, \dot{\mathbf{x}}) + \hat{\mathbf{g}}(\mathbf{x}, \dot{\mathbf{x}})\mathbf{u} \tag{7}$$

Assume that the Delta robot will track the desired trajectory $\mathbf{x}_d(t)$. The tracking error and its derivative are given as

$$\mathbf{e}(t) = \mathbf{x}_d(t) - \mathbf{x}(t), \;\; \dot{\mathbf{e}}(t) = \dot{\mathbf{x}}_d(t) - \dot{\mathbf{x}}(t) \tag{8}$$

We select a sliding surface as

$$\mathbf{s}(t) = \mathbf{C}\mathbf{e}(t) + \dot{\mathbf{e}}(t) \tag{9}$$

where $\mathbf{C} \in \mathbb{R}^{3\times 3}$ is positive-definite and satisfies the Hurwitz stability condition. Thus, the sliding mode control can be designed as [23]

$$\mathbf{u}(t) = \hat{\mathbf{g}}^{-1}\left(\gamma \tanh\left(\frac{\mathbf{s}}{\epsilon}\right) + \alpha\mathbf{s} + \mathbf{C}(\dot{\mathbf{x}}_d - \dot{\mathbf{x}}) + \ddot{\mathbf{x}}_d - \hat{\mathbf{f}}\right) \tag{10}$$

where $\gamma > 0$ and $\alpha > 0$ are constant gains, and $\epsilon > 0$. Chattering is a prevalent concern associated with sliding mode controllers with switching term $sgn(\mathbf{s})$. To reduce the chattering in the sliding mode control, the switching terms for the control law and the sliding surface have to be carefully chosen [24]. In this paper, the continuous function $\tanh(\cdot)$ is adopted to replace the discontinuous sign function [25]. ϵ determines the steepness of $\tanh(\cdot)$ as an approximation of the sign function. To prove the stability of the sliding mode control, we shall need the following lemma.

Lemma 1. *Let* $f, V : [0 : \infty) \in R$, *then* $\dot{V} \leq -\alpha V + f, \forall t \geq t_0 \geq 0$ *implies that [26]*

$$V(t) \leq e^{-\alpha(t-t_0)}V(t_0) + \int_{t_0}^{t} e^{-\alpha(t-\tau)}f(\tau)d\tau \tag{11}$$

Define a Lyapunov function as

$$V = \frac{1}{2}\mathbf{s}^T\mathbf{s} \tag{12}$$

Then, we have

$$\dot{V} = \mathbf{s}^T\dot{\mathbf{s}} \tag{13}$$

Since

$$\begin{aligned}\dot{\mathbf{s}} &= \mathbf{C}\dot{\mathbf{e}}(t) + \ddot{\mathbf{e}}(t) \\ &= \mathbf{C}(\dot{\mathbf{x}}_d - \dot{\mathbf{x}}) + (\ddot{\mathbf{x}}_d - \ddot{\mathbf{x}}) \\ &= \mathbf{C}(\dot{\mathbf{x}}_d - \dot{\mathbf{x}}) + (\ddot{\mathbf{x}}_d - \hat{\mathbf{f}} - \hat{\mathbf{g}}\mathbf{u}) \\ &= -\gamma \tanh\left(\frac{\mathbf{s}}{\epsilon}\right) - \alpha\mathbf{s},\end{aligned} \tag{14}$$

therefore, we obtain

$$\begin{aligned}\dot{V}(t) &= \mathbf{s}^T\left(-\gamma \tanh\left(\frac{\mathbf{s}}{\epsilon}\right) - \alpha\mathbf{s}\right) \\ &= -\alpha\mathbf{s}^T\mathbf{s} - \gamma\mathbf{s}^T \tanh\left(\frac{\mathbf{s}}{\epsilon}\right) \leq 0\end{aligned} \tag{15}$$

Hence, the closed-loop system is stable. Furthermore, we have

$$\dot{V}(t) = \begin{cases} -2(\alpha + \gamma/\epsilon)V(t) & s_i \leq \epsilon \text{ for all } i \\ -2\alpha V(t) & s_i > \epsilon \text{ for all } i \end{cases} \tag{16}$$

According to Lemma 1, we conclude that $V(t)$ exponentially decreases at the rate $2(\alpha + \gamma/\epsilon)$ when $\mathbf{s}$ is outside the boundary layer defined by ϵ, and at the rate of 2α when $\mathbf{s}$ is located inside the boundary layer. In both cases, we have

$$\lim_{t\to\infty} V(t) \to 0 \tag{17}$$

Hence, the tracking error of the sliding mode control for the system in Equation (7) converges asymptotically to zero at the exponential rate. On the other hand, when the neural network model of the Delta robot has a sufficiently small error, it can be expected that the sliding mode control will deliver similar tracking performance in experiments. This is a subject of on-going work. In this paper, we applied neural network model-based sliding mode control to the physics- and geometry-based model of the Delta robot, and carried out numerical simulations to observe and evaluate the validity of the proposed neural network modeling and control design approach.

In the following examples, we take $\mathbf{C}$ as $10 \times \mathbf{I}$ where $\mathbf{I}$ is the identity matrix, $\alpha = 0.1$, $\gamma = 0.1$, and $\epsilon = 1$. Three different paths are adopted to test the performance of the proposed sliding mode control. The mathematical expressions of three paths, namely heart curve, logarithmic spiral, and spiral, are given as follows.

$$\text{Heart curve:} \begin{cases} x = 0.01(16\sin^3(2.1\pi t/t_e)) \\ y = 0.01(13\cos(2.1\pi t/t_e) - 5\cos(4.1\pi t/t_e) \\ \qquad -2\cos(6.1\pi t/t_e) - \cos(8.1\pi t/t_e)) \\ z = -0.6 \end{cases} \tag{18}$$

$$\text{Logarithmic spiral:} \begin{cases} a = 0.2e^{(-0.48\pi t/t_e)} \\ x = a\cos(8\pi t/t_e) \\ y = -a\sin(8\pi t/t_e) \\ z = 0.1\sin(0.5\pi t/t_e) - 0.55 \end{cases} \tag{19}$$

$$\text{Spiral:} \begin{cases} x = 0.05\sin(5\pi t/t_e) \\ y = 0.22\cos(5\pi t/t_e) \\ z = -0.25t/t_e - 0.35 \end{cases} \tag{20}$$

where t_e is the end of the time. The 3D tracking results and part of the control inputs θ_i for different paths are shown in Figures 7–12. The neural network model-based sliding mode control demonstrates remarkable performance in trajectory tracking of various paths while maintaining bounded control inputs. This indicates that the trained neural network model successfully captures the Delta robot's nonlinear property with an acceptable magnitude of tracking error. The trajectory tracking errors of the sliding mode control $\mathbf{e}(t) = [e_{p_x}, e_{p_y}, e_{p_z}]^T$ for different curves are shown in Figure 13. The initial conditions for the tracking of all trajectories are chosen as $[\mathbf{x}; \dot{\mathbf{x}}] = [\mathbf{x}_d; \dot{\mathbf{x}}_d] - [0.1; 0.1; 0.1; 0.1; 0.1; 0.1]$.

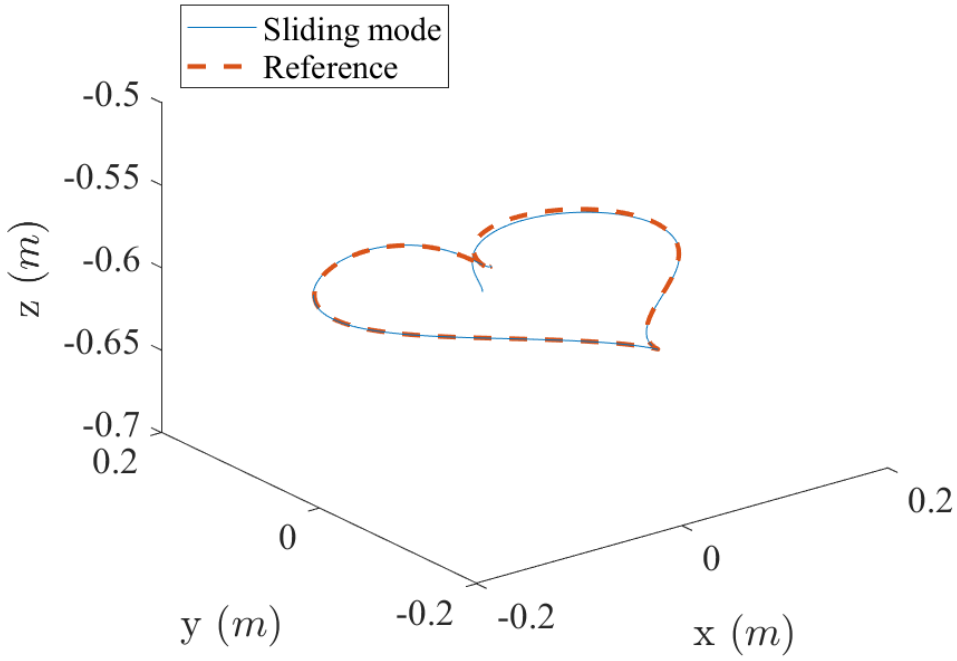

Figure 7. Three-dimensional trajectory tracking of heart curve using the sliding mode control with the neural network control-affine model.

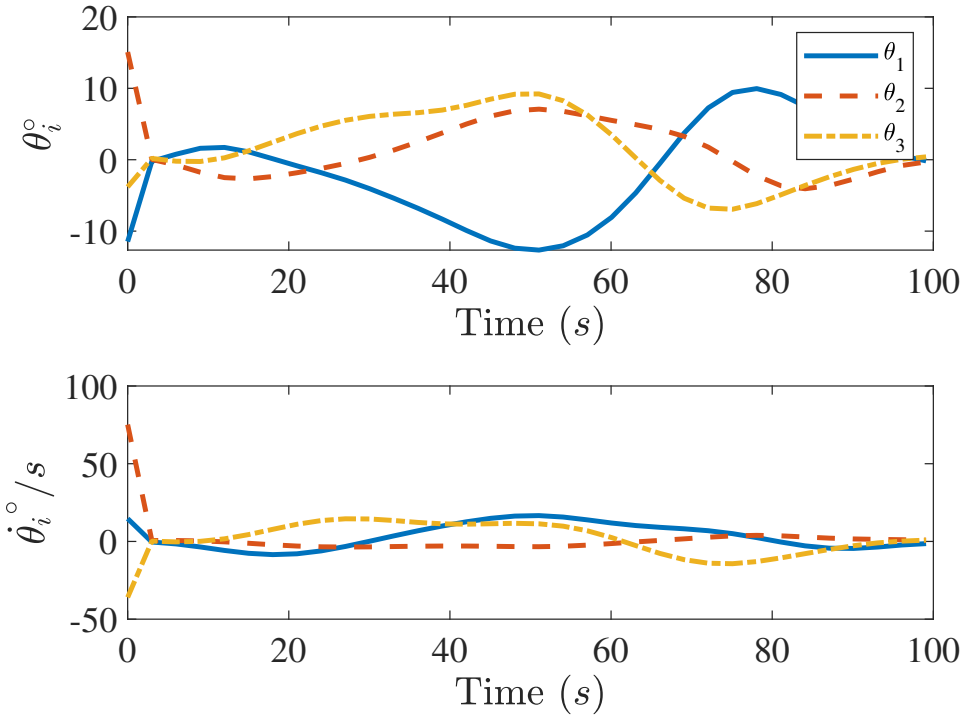

Figure 8. Magnitude of the sliding mode control with the neural network control-affine model for heart curve trajectory tracking.

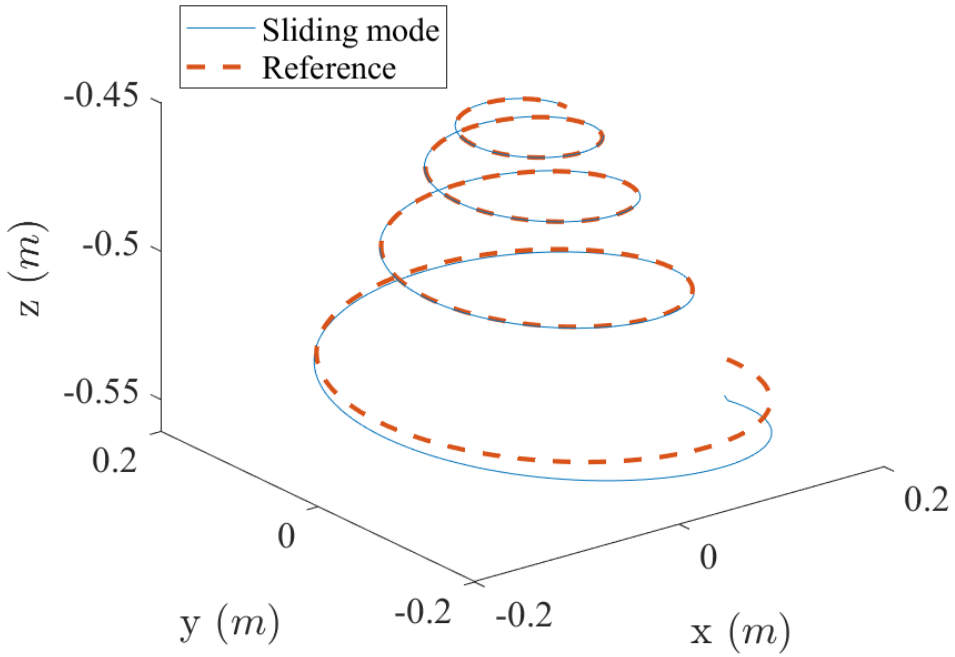

Figure 9. Three-dimensional trajectory tracking of logarithmic spiral curve using the sliding mode control with the neural network control-affine model.

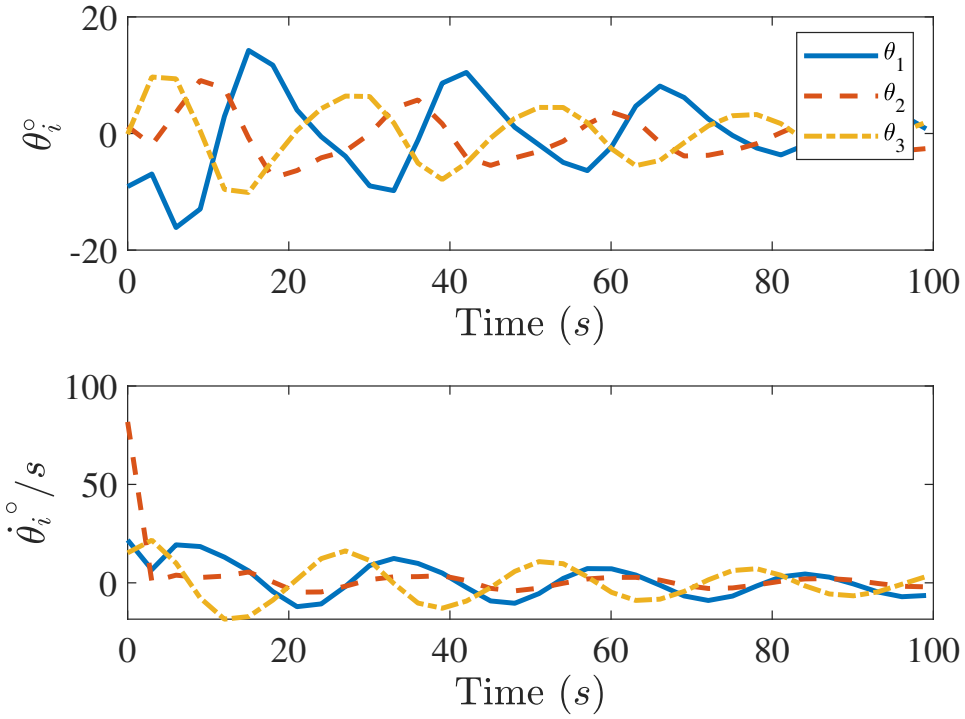

Figure 10. Magnitude of the sliding mode control with the neural network control-affine model for logarithmic spiral curve trajectory tracking.

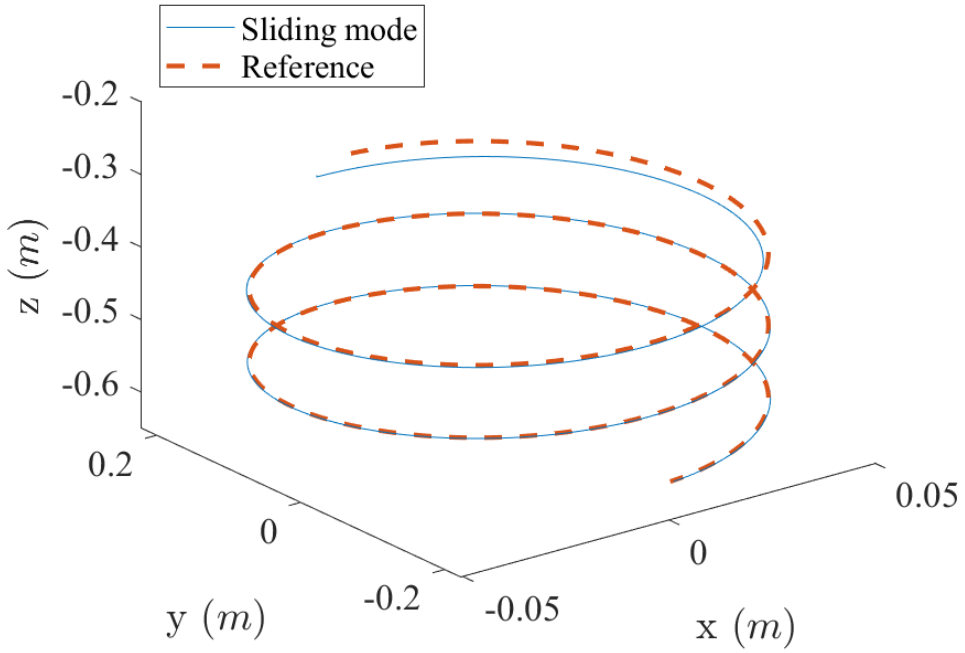

Figure 11. Three-dimensional trajectory tracking of spiral curve using the sliding mode control with the neural network control-affine model.

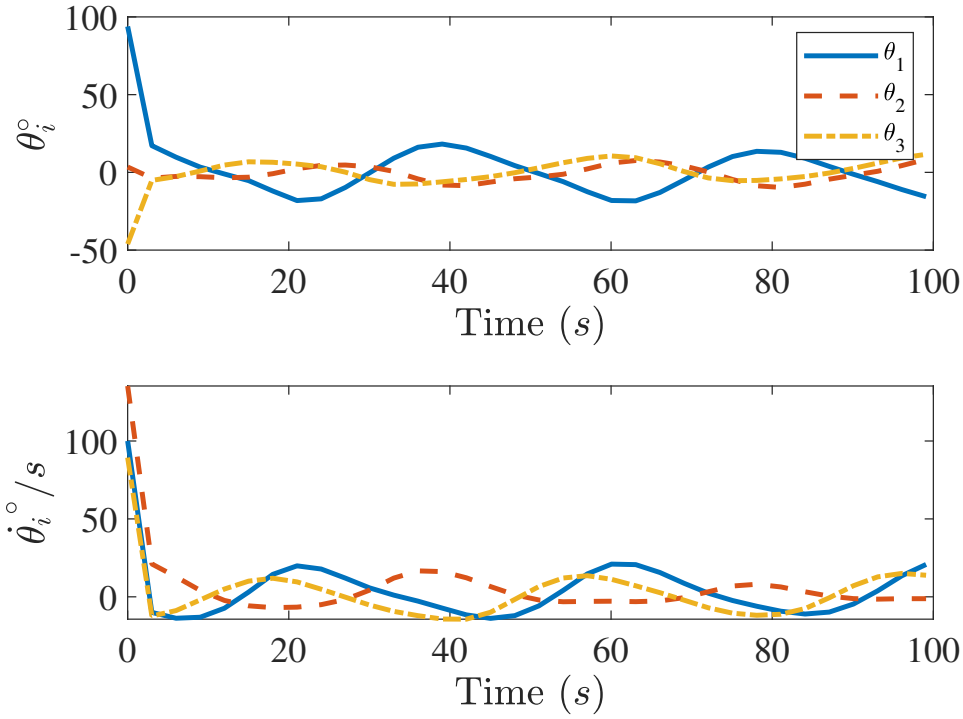

Figure 12. Magnitude of the sliding mode control with the neural network control-affine model for spiral curve trajectory tracking.

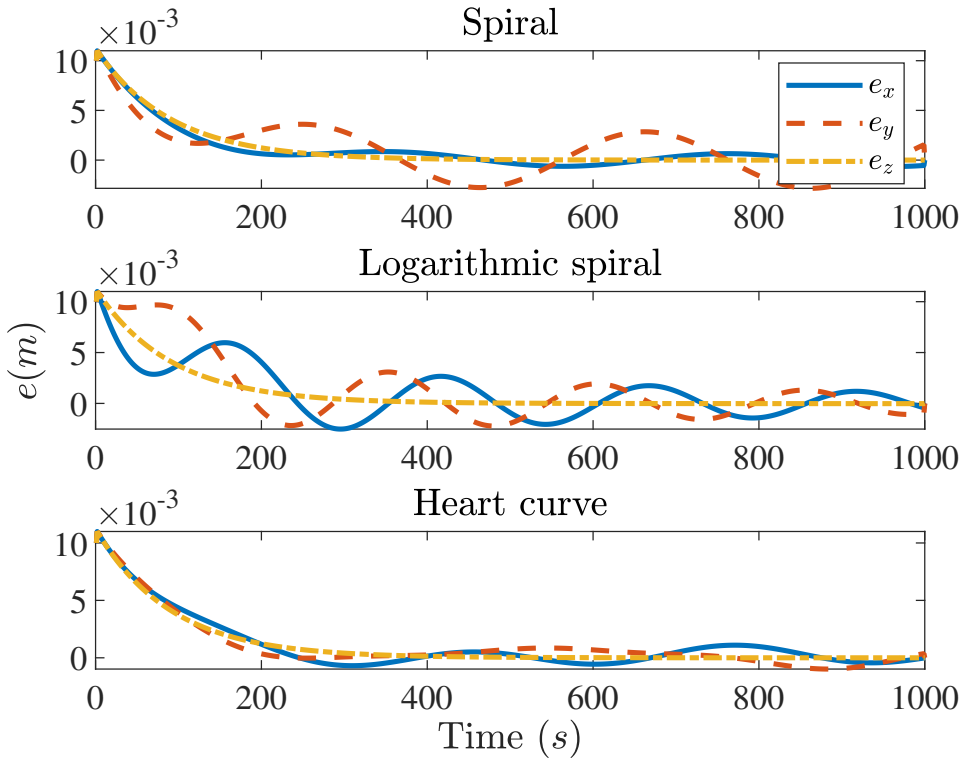

Figure 13. Tracking error of the sliding mode control in Equation (8) for different curves. Up: the error for spiral trajectory tracking. Middle: the error for logarithmic spiral trajectory tracking. Bottom: the error for heart curve trajectory tracking.

A detailed summary of the trajectory tracking errors are shown in Table 3. By evaluating these results, it is found that the maximum value of tracking error for different curves are the same, and the trajectory tracking errors all converge to zero in a finite time.

Table 3. Summary of different trajectory tracking errors in m.

Desired Trajectory	$\lvert e_x \rvert$		$\lvert e_y \rvert$		$\lvert e_z \rvert$	
	max	**min**	**max**	**min**	**max**	**min**
Heart curve	0.011	4.6135×10^{-7}	0.011	3.0669×10^{-7}	0.011	1.7682×10^{-7}
Logarithmic spiral	0.011	3.2416×10^{-6}	0.011	2.1551×10^{-7}	0.011	9.8488×10^{-9}
Spiral	0.011	2.5787×10^{-7}	0.011	6.1408×10^{-7}	0.011	1.7664×10^{-7}

5. Conclusions

Delta robots have highly nonlinear dynamics. Finding a control-affine model to describe the input–output relationship of the Delta robot analytically with joint angles and velocities as inputs is difficult. Since it is difficult to obtain the analytical representation of the input–output model, a neural network representation was developed to represent the model. This is a completely data-driven model intended for control design consideration, and it is not derivable from Newton's law or Lagrange's equation. The neural networks are trained with randomly sampled data in a sufficiently large workspace. Then, the sliding mode control is designed by making use of the control-affine model to track the desired trajectory. Extensive numerical results have shown that the neural network model-based sliding mode control can be highly effective for trajectory tracking for paths with varying complexity. However, in real-world applications, this control approach can be challenging due to differences in parameters of the frames, arms, and joints, as well as the presence of backlash and flexibility in the joints of the Delta robot. These factors can affect the implementation of neural network-based sliding mode control on hardware. Moreover, the parameters of the sliding mode control need to be fine-tuned based on the specific hardware being used, and high-frequency data collection is required for successful implementation. To overcome these challenges, further work is required to implement neural network-based sliding mode control on hardware. With real data collected from experiments, the method of transfer learning can be applied to refine the neural network input–output model and to update the parameters of the sliding mode control.

Author Contributions: Conceptualization, A.Z. and J.-Q.S.; methodology, A.Z. and J.-Q.S.; software, A.Z.; hardware, A.T. and A.Z.; writing—original draft preparation, A.Z.; writing—review and editing, A.Z., A.T., R.E. and J.-Q.S.; supervision, R.E., J.H.V. and J.-Q.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the AI Research Institutes program supported by NSF and USDA-NIFA under the AI Institute: Agricultural AI for Transforming Workforce and Decision Support (AgAID) award No. 2021-67021-35344.

Data Availability Statement: Data are contained within the article.

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

References

1. Angel, L.; Viola, J. Fractional order PID for tracking control of a parallel robotic manipulator type delta. *ISA Trans.* **2018**, *79*, 172–188. [CrossRef] [PubMed]
2. Zubizarreta, A.; Larrea, M.; Irigoyen, E.; Cabanes, I.; Portillo, E. Real time direct kinematic problem computation of the 3PRS robot using neural networks. *Neurocomputing* **2018**, *271*, 104–114. [CrossRef]
3. Abed Azad, F.; Ansari Rad, S.; Hairi Yazdi, M.R.; Tale Masouleh, M.; Kalhor, A. Dynamics analysis, offline-online tuning and identification of base inertia parameters for the 3-DOF Delta parallel robot under insufficient excitations. *Meccanica* **2022**, *57*, 473–506. [CrossRef]
4. Utkin, V.I. Sliding mode control design principles and applications to electric drives. *IEEE Trans. Ind. Electron.* **1993**, *40*, 23–36. [CrossRef]
5. Perruquetti, W.; Barbot, J.P. *Sliding Mode Control in Engineering*; CRC Press: Boca Raton, FL, USA, 2002.
6. Xu, J.; Wang, Q.; Lin, Q. Parallel robot with fuzzy neural network sliding mode control. *Adv. Mech. Eng.* **2018**, *10*, 1687814018801261. [CrossRef]

7. Boudjedir, C.E.; Boukhetala, D.; Bouri, M. Nonlinear PD plus sliding mode control with application to a parallel Delta robot. *J. Electr.-Eng.-Elektrotechnicky Cas.* **2018**, *69*, 329–336. [CrossRef]
8. Pham, P.C.; Kuo, Y.L. Robust adaptive finite-time synergetic tracking control of Delta robot based on radial basis function neural networks. *Appl. Sci.* **2022**, *12*, 10861. [CrossRef]
9. Yen, V.T.; Nan, W.Y.; Van Cuong, P. Robust adaptive sliding mode neural networks control for industrial robot manipulators. *Int. J. Control. Autom. Syst.* **2019**, *17*, 783–792. [CrossRef]
10. Zhao, R.; Wu, L.; Chen, Y.H. Robust control for nonlinear Delta parallel robot with uncertainty: An online estimation approach. *IEEE Access* **2020**, *8*, 97604–97617. [CrossRef]
11. Castañeda, L.A.; Luviano-Juárez, A.; Chairez, I. Robust trajectory tracking of a Delta robot through adaptive active disturbance rejection control. *IEEE Trans. Control Syst. Technol.* **2014**, *23*, 1387–1398. [CrossRef]
12. Boudjedir, C.E.; Bouri, M.; Boukhetala, D. Iterative learning control for trajectory tracking of a parallel Delta robot. *At-Autom.* **2019**, *67*, 145–156. [CrossRef]
13. Boudjedir, C.E.; Bouri, M.; Boukhetala, D. Model-free iterative learning control with nonrepetitive trajectories for second-order MIMO nonlinear systems—Application to a Delta robot. *IEEE Trans. Ind. Electron.* **2020**, *68*, 7433–7443. [CrossRef]
14. Ghafarian Tamizi, M.; Ahmadi Kashani, A.A.; Abed Azad, F.; Kalhor, A.; Masouleh, M.T. Experimental study on a novel simultaneous control and identification of a 3-DOF Delta robot using model reference adaptive control. *Eur. J. Control* **2022**, *67*, 100715. [CrossRef]
15. Gholami, A.; Homayouni, T.; Ehsani, R.; Sun, J.Q. Inverse Kinematic Control of a Delta Robot Using Neural Networks in Real-Time. *Robotics* **2021**, *10*, 115. [CrossRef]
16. Gholami, A.; Sun, J.Q.; Ehsani, R. Neural Networks Based Optimal Tracking Control of a Delta Robot With Unknown Dynamics. *Int. J. Control. Autom. Syst.* **2023**, *21*, 3382–3390. [CrossRef]
17. Zhao, A.; Toudeshki, A.; Ehsani, R.; Sun, J.Q. Data-Driven Inverse Kinematics Approximation of a Delta Robot with Stepper Motors. *Robotics* **2023**, *12*, 135. [CrossRef]
18. Gosselin, C.; Angeles, J. Singularity analysis of closed-loop kinematic chains. *IEEE Trans. Robot. Autom.* **1990**, *6*, 281–290. [CrossRef]
19. Romdhane, L.; Affi, Z.; Fayet, M. Design and singularity analysis of a 3-translational-DOF in-parallel manipulator. *J. Mech. Des.* **2002**, *124*, 419–426. [CrossRef]
20. Mueller, A. Modern robotics: Mechanics, planning, and control. *IEEE Control Syst. Mag.* **2019**, *39*, 100–102. [CrossRef]
21. Bottou, L. Stochastic gradient descent tricks. In *Neural Networks: Tricks of the Trade*, 2nd ed.; Springer: Berlin/Heidelberg, Germany, 2012; pp. 421–436.
22. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
23. Liu, J. *Sliding Mode Control Using MATLAB*; Academic Press: Cambridge, MA, USA, 2017.
24. Ahmad, S.; Uppal, A.A.; Azam, M.R.; Iqbal, J. Chattering free sliding mode control and state dependent Kalman filter design for underground gasification energy conversion process. *Electronics* **2023**, *12*, 876. [CrossRef]
25. Aghababa, M.P.; Akbari, M.E. A chattering-free robust adaptive sliding mode controller for synchronization of two different chaotic systems with unknown uncertainties and external disturbances. *Appl. Math. Comput.* **2012**, *218*, 5757–5768. [CrossRef]
26. Ioannou, P.A.; Sun, J. *Robust Adaptive Control*; PTR Prentice-Hall: Upper Saddle River, NJ, USA, 1996; Volume 1.