Research Article

Abdonoor Kalibala*, Ayman A. Nada, Hiroyuki Ishii, and Haitham El-Hussieny

Real-time force/position control of soft growing robots: A data-driven model predictive approach

https://doi.org/10.1515/nleng-2025-0099
received October 9, 2024; accepted February 3, 2025

Abstract: Vine robots represent a novel class of soft robots that achieve mobility through tip extension, a mechanism inspired by the natural growth processes of vine plants. This unique movement strategy enables effective navigation in constrained and cluttered environments, offering significant advantages over conventional robotic systems. However, the continuum nature and inherent compliance of vine robots introduce complex modeling and control challenges. Deep learning offer a powerful alternative for modeling systems with such complex dynamics. In this article, we present a data-driven dynamic model for a pressure-driven everting vine robot, utilizing a deep neural network (DNN)-based discrete-time dynamic model. This model was integrated into a model predictive control (MPC) framework, and a comparative analysis was conducted against the MPC framework using a nonlinear first-principle model of the vine robot. The results demonstrate that the DNN-MPC framework offers a better control performance and significantly improved computational efficiency compared to the MPC based on the nonlinear first-principles model. The DNN-MPC reduced computation time by a factor of 11, making it highly viable for real-time control applications.

Keywords: vine robot, soft robots, model predictive control, nonlinear control, machine learning

1 Introduction

The animal kingdom, a rich source of inspiration for science and engineering, has long influenced the development of robotics. Recently, researchers have turned their attention to the plant kingdom, creating robots that emulate plant properties such as growth, energy efficiency, sustainability, and adaptability to unknown environments. Plants provide a vast array of mechanisms and materials, from roots to leaves and seeds to flowers, offering abundant bio-inspiration that can be harnessed to develop sustainable and efficient technologies for solving complex problems [1]. Unlike animals, which primarily rely on locomotion to navigate their environments [2], plants utilize growth as their primary means of movement. This unique strategy, characterized by tip extension toward stimuli such as sunlight, operates at a slower rate and is a hallmark of the plant kingdom [3,4]. This type of growth-driven movement can be observed across various scales in nature, including in neurons [5], pollen tubes [6], trailing vines searching for support, and climbing plants that develop tendrils [7,8]. Recently, there has been a significant progress in replicating this form of movement in robotic systems inspired by vine plants. These robots, often referred to as “vine robots” or “growing robots,” aim to mimic the growth behavior of plants to navigate and interact with their surroundings in a manner similar to natural vines.

Replicating growth in robotic systems is archived by employing an ingenious method called pressure-driven eversion of flexible thin-walled material. This technique involves turning a thin-walled membrane inside out. When pressurized, the internal pressure forces the material to extend from the core of the existing body, causing it to evert at the tip and elongate. Robots that utilize this mechanism are known as everting vine robots. These robots offer several advantages over traditional rigid robots: there is no relative movement between the body and the environment since only the tip moves relative to the ground, reducing energy dissipation due to minimal friction and inertia. The tip’s movement creates a three-dimensional (3D) structure, which can be used to deliver

*** Corresponding author: Abdonoor Kalibala**, Department of Mechatronics and Robotics Engineering, Egypt-Japan University of Science and Technology, E-JUST, Alexandria, Egypt, e-mail: abdonoor.kalibala@ejust.edu.eg
Ayman A. Nada: Department of Mechatronics and Robotics Engineering, Egypt-Japan University of Science and Technology, E-JUST, Alexandria, Egypt, e-mail: ayman.nada@ejust.edu.eg
Hiroyuki Ishii: Faculty of Science and Engineering, Waseda University, Tokyo, Japan, e-mail: hiro.ishii@waseda.jp
Haitham El-Hussieny: Department of Mechatronics and Robotics Engineering, Egypt-Japan University of Science and Technology, E-JUST, Alexandria, Egypt, e-mail: haitham.elhussieny@ejust.edu.eg

payloads, transfer force, and form structures for various applications [9]. Everting vine robots are especially useful in navigating unstructured and space-constrained environments [10,11]. The application areas of vine robots are diverse and include minimally invasive surgery, surveillance and monitoring, soft burrowing devices, soft artificial muscles, and reconfigurable antennas [12–15].

However, the continuum nature and inherent compliance of the vine robot introduce a new set of modeling and control challenges compared to traditional rigid manipulators [16,17]. Modeling the dynamic behavior of vine robots presents significant mathematical complexity due to the inherent nonlinearities typically exhibited in soft robots. The air takes time to fill the pressure chamber, causing a delay in response to input pressure. In addition, the dynamic nature of the compressed air and the arbitrary geometry of the vine robot introduce further nonlinearities. As a result, first-principle models often fail to capture these dynamics accurately due to their simplifications. The piecewise kinematic models and kinematic-based controllers [18] for vine robots are suitable in scenarios where fast response times are not critical. However, these models do not explicitly capture the influence of forces, moments, and interactions with the external environment on the robot's motion, limiting their applicability in dynamic and reactive tasks. Alternative modeling approaches, such as the simulation open framework architecture (SOFA) [19], which is based on the finite element method (FEM), have also been explored. While FEM-based models offer a high degree of accuracy in capturing complex deformations, they suffer from significant computational costs and lack the capability to be easily transformed into mathematical models that are compatible with model-based controllers, thus limiting their use in real-time applications. These modelling challenges also affects the control strategies employed to control the vine robot.

Model-based control strategies, such as model predictive control (MPC), are employed to simultaneously predict and control the growth and steering behavior of the everting vine robot in free space, utilizing either kinematic or dynamic models [18,20]. MPC predicts the future behavior of the controlled system by solving a constrained optimization problem within a finite time horizon at each time step [21]. The widespread use of MPC in robotics is attributed to its ability to control complex nonlinear systems with constraints and uncertainty, a hallmark feature of soft robotic systems that often confounds classical linear control methods [21,22]. The initial challenge in applying MPC lies in developing a dynamic model that accurately and efficiently describes the system's behavior while being suitable for real-time control applications. This requires a thorough understanding of the system dynamics to create models that can predict future states and control actions effectively. In addition, the computational efficiency of the optimization process is crucial to ensure that the control actions can be computed within the time constraints required by real-time applications.

Developing a mathematical model for an everting vine robot that balances computational efficiency with sufficient accuracy is crucial for capturing the dynamics of this soft, flexible robot and for optimizing its design and control. The continuum nature, inherent compliance, and pneumatic actuation of the vine robot closely align with the core principles of soft robotic systems, which are defined by their flexibility and adaptability. State-of-the-art modeling techniques in soft robotics, such as the piecewise constant curvature approach, can be further enhanced by incorporating growth dynamics inspired by plant tip extension. This integration would enable a more comprehensive representation of the vine robot's motion, capturing both its unique growth-driven locomotion and its interaction with the environment, thereby broadening the applicability and flexibility of current soft robotics models. Only a limited number of studies have ventured into developing a dynamic model for everting vine robots to characterize their behavior in response to environmental interactions and actuator inputs, primarily due to the inherent complexity of such systems. In the study by Blumenschein *et al.* [23], a quasi-static force balance model for a pressure-driven everting vine robot was developed. This model focuses on the equilibrium of forces, disregarding system inertia, and links the driving force generated by internal pressure to the yield force and a term dependent on velocity. Similarly, Blumenschein *et al.* [24] presented a quasi-static model for a novel growing soft-continuum robot. This model integrates a kinematic framework with a quasi-static approach to precisely control the position of the end effector within the task space. In the study by Haggerty *et al.* [10], a detailed mathematical model was developed to describe the behavior of vine robots interacting with their environment in both unconstrained and constrained scenarios. This model accounts for axial and transverse buckling modes as the robot grows into obstacles, based on a quasi-static equilibrium framework for vine robots traveling at an average speed of less than 0.05 m/s and a path length of less than 5 m. Similarly, Naclerio *et al.* [13] modified a quasi-static model to consider the external opposing force of dry sand, where the soft burrowing device operates. This device utilizes tip-extension, inspired by plant root growth, and incorporates granular air-fluidization by channeling air through the backbone core.

Physics-based model for a miniature steerable soft growing robot, employing the SOFA to simulate growth, steering, external loading, and pressure stiffening was developed in the study by Wu *et al.* [19]. Despite its detailed simulations, this model is computationally intensive and unsuitable for real-time control. In contrast, Jitosho *et al.* [25] introduced a dynamic simulator for soft growing robots using pseudo-rigid-body kinematics with contact constraints to examine vine robot interactions with the environment. However, this model does not consider continuous length changes, it is restricted to planar motion, and its performance is limited by the number of discrete rigid bodies included. Despite significant efforts in developing dynamic models for pressure-driven everting vine robots to describe their behavior, it remains prohibitively challenging to create a parsimonious and computationally efficient model suitable for real-time control. The inherent complexities of these robots, including their nonlinear dynamics and interactions with the environment, pose substantial obstacles to achieving both accuracy and computational tractability in real-time applications.

Data-driven modeling, employing machine learning techniques, presents a novel approach to characterizing complex systems, including the use of deep learning [26]. Deep learning, a subset of artificial intelligence and machine learning, utilizes multilayer neural networks to identify patterns within data, effectively correlating input features with target outcomes. The confluence of big data, advanced algorithms in machine learning, and advancements in computational hardware are fueling a new paradigm in modeling and control of complex systems, resulting in watershed moment in the modern dynamical systems and transformative progress in data-driven dynamic systems of soft robots [27]. Recent works have demonstrated the effectiveness of deep neural networks (DNN) in modeling complex dynamic systems of soft robots. In the study by Cespedes *et al.* [28], a DNN was used to estimate the dynamic model of a soft robotic laparoscope using data sampled at fixed time steps for random inputs from the system. Similarly, in the study by Hyatt *et al.* [29], a neural network obtained from data-driven principles was used as a surrogate model in a MPC. A neural network-based dynamic model for a soft robot with symmetrical chambers, which was used to develop a MPC to control the motion and track the desired trajectory of the robot, was developed by Li *et al.* [11]. These studies demonstrate the potential of combining neural network-based dynamic models with MPC, offering improved computational efficiency and performance compared to traditional analytical models for soft robots.

In this work, we present a data-driven dynamic model for a pressure-driven everting vine robot, utilizing advanced deep learning techniques. A nonlinear first principles model of the everting vine robot was developed, and a DNN was employed to approximate the nonlinear discrete-time dynamics using data from nonlinear simulations of the proposed analytical model of the vine robot. The DNN-based dynamic model was then integrated into a MPC framework to enhance the data-driven control of the vine robot, improving computational efficiency for real-time applications. The deep learning model was rigorously validated against the first principles model by comparing the computation time required to determine optimal control actions. Furthermore, we performed k-fold validation to assess the model's generalization performance, ensuring robustness across different datasets.

The model was successfully incorporated into the MPC framework with additional constraints and bounds to ensure operation within the physical limitations of the actuators and the feasible region of the state space. Simulation results demonstrated significantly improved performance in trajectory tracking capabilities, point stabilization, and obstacle avoidance, highlighting the potential of combining deep learning with MPC for advanced control of soft robots. In addition, the computation time of the MPC optimizer reduced significantly with DNN-based dynamic model compared to the nonlinear first principles model.

The key contributions of this article can be summarized as follows:

- Development of a DNN architecture for dynamic modeling of a pressure-driven everting vine robot, utilizing data from a nonlinear first principles model.
- Designing a MPC that employs the neural network (NN)-based dynamic model, which is computationally efficient, to control the growth and direction of the vine robot.
- Formulation of a flexible first-principles model of the everting vine robot that accurately represents its dynamic behavior.
- Contribution to the literature by proposing a versatile model applicable to any pneumatic growing robot, irrespective of the steering mechanism used.
- Benchmarking and validation of the DNN-based dynamic model against the first principles model by evaluating the computation time required by the MPC optimizer to find the optimal control actions under given constraints, bounds, and uncertainties.

The final contribution represents the primary novelty of our article. We anticipate that the integration of a DNN-based dynamic model with MPC will significantly enhance the performance of data-driven control systems. This combination leverages the predictive capabilities of NNs to better capture the complex dynamics of the everting vine robot, thus optimizing control strategies and improving overall system efficiency.

The remainder of this article is structured as follows: Section 2 presents the basic structure of the everting vine robot and its kinematic analysis in Cartesian space. The dynamic model, encompassing both the growth and steering behaviors, is introduced in Section 3. Section 4 discusses the stresses experienced by the growing robot during operation. The data-driven dynamic model is detailed in Section 5, followed by the development of the MPC framework in Section 6. Simulation results are presented in Section 7, and finally, conclusions are drawn in Section 8.

2 Kinematics of vine robots

2.1 Structure description

The structural design of the vine-growing robot examined in this study is illustrated in Figure 1. The robot comprises several key components: a pressure chamber, a spool, a flexible tube, and a steering mechanism that enables control of the robot's tip local roll and yaw angles. The flexible tube, typically made from thin-walled low-density polyethylene, is sealed at one end and inverted, passing through the center of the existing tube before being rolled onto the spool. The open end is securely attached to the nozzle of the pressure chamber. When pressurized air from a compressor is introduced into the pressure chamber, the internal pressure propels the material through the robot's body, causing it to evert at the tip and extend in length. This process is referred to as growth.

The servomotor, attached to the spool, manages the feeding of material necessary for the robot's growth and retracts the material after task completion. While some studies use a servomotor to control the robot's elongation, in this research, the servomotor is employed solely for two purposes: to feed the material required for growth without causing tension in the "tail" – the material that travels within the robot's body – and to retract the material after the task is completed.

The steering system typically employs distributed strain actuation, using flexible actuators evenly spaced along the robot's backbone to achieve distributed strain. Alternatively, it may use concentrated strain actuation, where actuators are placed at specific intervals along the robot's length. Regardless of the approach, the primary purpose of the steering mechanism is to control the robot's growth direction by adjusting the local roll and yaw angles. This is accomplished through asymmetric growth, which creates a length differential on either side of the robot's body, mimicking how a vine bends due to uneven cell growth on its sides.

2.2 Kinematic analysis

To determine the tip's position and orientation of the soft growing robot, it is essential to establish its kinematics within the chosen configuration space. Various methods such as piecewise constant curvature [18,20,30] and the

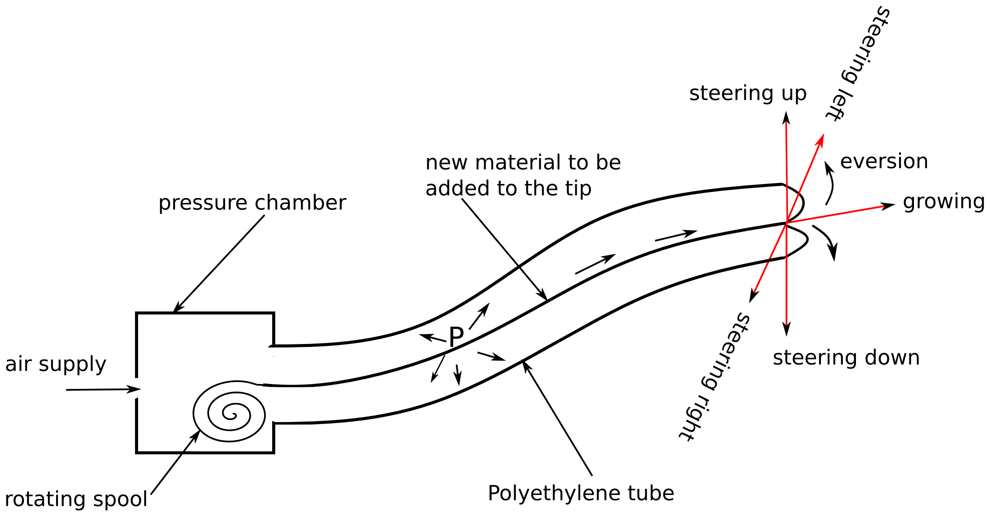

Figure 1: Schematic diagram of vine growing robot depicting its structure.

classical Denavit–Hartenberg (D-H) coordinate system [11] have been utilized to elucidate the kinematics of vine robots. The piecewise constant curvature approach is particularly favored for modeling a broad array of continuum robots due to its analytical elegance and the capability to decompose kinematics into two distinct mappings: from actuator space to configuration space and from configuration space to task space [16]. This method is especially accurate for soft robots with a single curvature and actuators distributed along their backbone, proving effective when gravitational effects and external loads are negligible [31]. However, it is less suitable for soft robots with minimal bending curvatures [32], and the constant curvature assumption becomes unreliable at longer lengths [33].

Conversely, the classical D-H coordinate approach, while providing a homogeneous transformation matrix, relies on parameters that are difficult to measure directly in real-time settings. Therefore, this study adopts a comprehensive and general approach to model the kinematics of the growing robot, integrating the strengths of both methods to accurately capture the robot's kinematic behavior in various conditions.

Figure 2 illustrates the detailed kinematic modeling of a vine-growing robot in free space. The global frame $G(OXYZ)$ is anchored at the center of the robot's base, with the Y-axis tangent to the backbone's base, providing a global orientation reference. The body frame $B(Oxyz)$, attached to the robot's tip, ensures that the y-axis remains tangent to the end of the robot's backbone, regardless of environmental contacts or disturbances. The position of the origin of the body frame $B(Oxyz)$ relative to the global frame $G(OXYZ)$ is represented by the position vector $\boldsymbol{p} = [X\ Y\ Z]^T$, and the everted length of the robot is denoted by l. The angles α_k and β_k represent the bending angles around the local x- and z-axes, respectively, resulting from the asymmetrical growth due to the steering mechanism or environmental interactions and disturbances.

The spool, rotating at an angular velocity ω, controls the extension and retraction of the robot's body. At time step k, the local frame $B(Oxyz)$ undergoes a sequence of rotations: first, a rotation by angle α_k around the current x-axis, described by the rotation matrix $R_x(\alpha_k) \in SO(3)$, followed by a rotation by angle β_k around the current z-axis, described by the rotation matrix $R_z(\beta_k) \in SO(3)$. Consequently, the rotation matrix from the tip to the global coordinate system is expressed as follows:

$$ {}^{G}R_B = \prod_{i=0}^{\overset{\curvearrowright}{T}} R_x(\alpha_k) R_z(\beta_k), \tag{1}$$

where

$$R_x(\alpha_k) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha_k & -\sin\alpha_k \\ 0 & \sin\alpha_k & \cos\alpha_k \end{bmatrix},$$

$$R_z(\beta_k) = \begin{bmatrix} \cos\beta_k & -\sin\beta_k & 0 \\ \sin\beta_k & \cos\beta_k & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

and T represents the total number of discrete time steps. Therefore, after substitution, the transformation matrix ${}^{G}R_B$ is given as follows:

$$ {}^{G}R_B = \prod_{k=0}^{\overset{\curvearrowright}{T}} \begin{bmatrix} \cos\beta_k & -\sin\beta_k & 0 \\ \cos\alpha_k \sin\beta_k & \cos\alpha_k \cos\beta_k & -\sin\alpha_k \\ \sin\alpha_k \sin\beta_k & \sin\alpha_k \cos\beta_k & \cos\alpha_k \end{bmatrix}. \tag{2}$$

In this study, the final orientation of the body frame $B(Oxyz)$ relative to the global frame $G(OXYZ)$ after a series of finite rotations is represented using Euler angles φ, θ, and ψ. These angles correspond to the rotation around the global Z-axis, the rotation around the local x-axis, and the

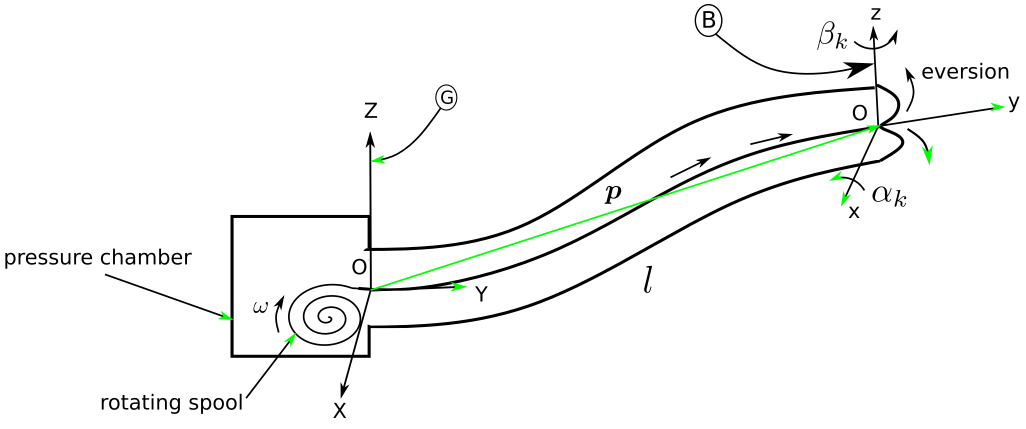

Figure 2: Schematic diagram of the kinematic model of the vine robot.

rotation around the local z-axis, respectively. The transformation matrix from $B(Oxyz)$ to $G(OXYZ)$ is given by:

$$
{}^{G}A_{B} = \begin{bmatrix} c\varphi\, c\psi - c\theta\, s\varphi\, s\psi & -\, c\varphi\, s\psi - c\theta\, c\psi\, s\varphi & s\theta\, s\varphi \\ c\psi\, s\varphi + c\theta\, c\varphi\, s\psi & -\, s\varphi\, s\psi + c\theta\, c\varphi\, c\psi & -\, c\varphi\, s\theta \\ s\theta\, s\psi & s\theta\, c\psi & c\theta \end{bmatrix}. \tag{3}
$$

Let r_{mn} denote the element in row m and column n of the rotation matrix in (2). The Euler angles at time step k can then be determined as follows:

$$
\theta_k = \arccos r_{33}. \tag{4}
$$

For $\sin\theta_k \neq 0$, the angles φ_k and ψ_k are given by:

$$
\varphi_k = \arctan\left(\frac{r_{13}}{-r_{23}}\right), \quad \psi_k = \arctan\left(\frac{r_{31}}{r_{32}}\right). \tag{5}
$$

However, if $\sin\theta_k = 0$, the angles simplify to:

$$
\varphi_k = \arctan\left(\frac{r_{21}}{r_{11}}\right), \quad \psi_k = 0. \tag{6}
$$

The rate at which the everted length l increases in free space must equal the velocity of the robot's tip, as the everted length directly translates to the linear displacement of the tip. This relationship ensures that the physical extension of the robot's body is accurately represented by its motion in space. Mathematically, this is expressed as follows:

$$
\dot{l} = \sqrt{\dot{X}^2 + \dot{Y}^2 + \dot{Z}^2}. \tag{7}
$$

To prevent tension in the tail, the rate at which the spool feeds the robot's body during growth must be at least equal to the rate at which material is added at the robot's tip. This ensures that the material supply keeps pace with the extension demands. This constraint is crucial to avoid buckling caused by the force applied at the tip due to tension in the tail and to simplify the dynamic model. Consequently, the velocity of the tip is controlled solely by internal air pressure. Maintaining this balance is essential for ensuring smooth and uninterrupted growth, as well as preventing mechanical failure. The mathematical relationship governing this balance is expressed as follows:

$$
\omega r_{sp} \geq \dot{l}, \tag{8}
$$

where r_{sp} is the radius of the spool and ω is the angular velocity at which the spool rotates. Ensuring this relationship holds true allows for consistent material feeding, thereby enhancing the robot's performance and reliability during operation.

3 Dynamic modeling

This section presents a dynamic model for the growing vine robot, designed to be both computationally efficient and suitable for real-time implementation. This model aids in the development and synthesis of closed-loop controllers. In this study, we employ a lumped parameter model, which simplifies the system by assuming a single-point mass at the tip of the soft growing robot. This approach is chosen for its effectiveness in capturing the dynamics of mixed systems across various domains and for its ability to describe system behavior through computationally efficient ordinary differential equations. Furthermore, the lumped parameter model facilitates seamless integration with control algorithms and accelerates the rapid testing and iteration of different models, enhancing the overall development process.

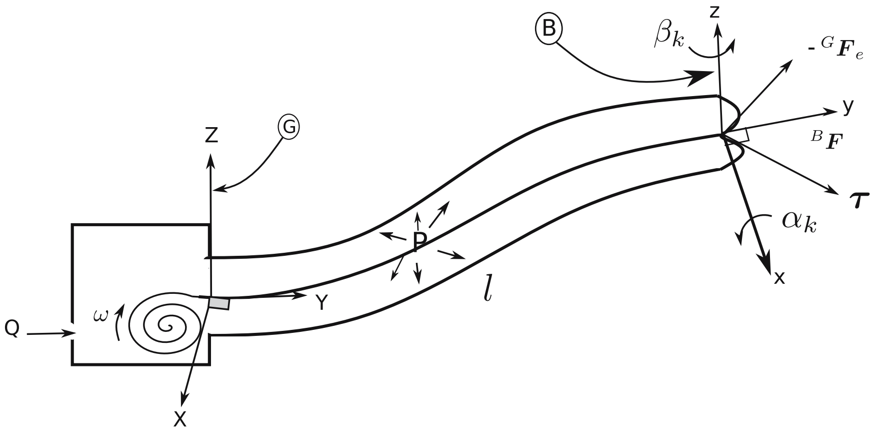

Figure 3: Dynamic model of the vine-growing robot, illustrating forces, actuation inputs, and environmental interactions.

3.1 The growth model

This section explores the modeling of the everting vine robot during its growth phase, focusing on the dynamics and interactions involved. It examines the forces exerted by both the environment and internal pressure. Figure 3 illustrates the free body diagram of the vine-growing robot, which has a concentrated fictitious mass m at its tip. The force acting on the tip, ${}^{B}\boldsymbol{F} = [0\ F\ 0]^{T}$ in the body frame, is generated by the internal absolute pressure P and acts along the y-axis of the body frame. In addition, the force exerted by the vine robot on the environment due to environmental interactions is represented by ${}^{G}\boldsymbol{F}_{e} = [F_{ex}\ F_{ey}\ F_{ez}]^{T}$ in the global frame. To model the bending behavior of the system caused by disturbances and environmental interactions, we introduce a torque $\boldsymbol{\tau}$ at the tip. Air with a volumetric flow rate Q is introduced into the pressure chamber by the compressor. The dynamics of the pressure-driven everting vine robot is expressed as follows:

$$ {}^{G}\boldsymbol{F} - {}^{G}\boldsymbol{F}_{e} = m\ddot{\boldsymbol{p}}, \tag{9}$$

where ${}^{G}\boldsymbol{F} = {}^{G}A_{B}{}^{B}\boldsymbol{F}$. By substituting this into the equation of motion, we obtain:

$$\begin{bmatrix} F(-\sin\psi\cos\varphi - \sin\varphi\cos\psi\cos\theta) - F_{ex} \\ F(-\sin\psi\sin\varphi + \cos\psi\cos\theta\cos\varphi) - F_{ey} \\ F\sin\theta\cos\psi - F_{ez} \end{bmatrix} = \begin{bmatrix} m\ddot{X} \\ m\ddot{Y} \\ m\ddot{Z} \end{bmatrix}. \tag{10}$$

The terms involving the angular variables φ, θ, and ψ in equation (10) represent the orientation of the vine robot in space and describe how the internal forces are projected along the global axes. These angles are derived from the robot's kinematic configuration and evolve with its motion during growth and steering.

The force F corresponds to the internal force generated by the pressure-driven eversion mechanism, and it is distributed across the global X, Y, and Z directions according to the robot's orientation. The terms $\sin\psi$, $\cos\varphi$, and $\cos\theta$ allow the projection of this internal force onto the respective global coordinate axes.

The external forces F_{ex}, F_{ey}, and F_{ez} represent interactions with the environment, such as contact forces from obstacles. These forces act in opposition to the robot's internal forces and are subtracted accordingly. In this model, the external forces are subject to the following assumptions: First, the contact point where the external force is applied must always lie outside the vine robot; no external forces can act within its structure. Second, the normal forces exerted by obstacles must be nonnegative, meaning that obstacles can only push the robot away and cannot pull it toward them. In addition, it is assumed that the material properties of the vine robot remain unchanged, with no degradation due to fatigue caused by repeated interactions with the environment. The angular variables also indirectly reflect external perturbations, as any environmental interaction (*e.g.*, obstacle contact) can alter the robot's configuration, which in turn affects the values of φ, θ, and ψ.

The force F exerted on the tip due to internal pressure is given by:

$$F = A_{v}(P - P_{\text{atm}} - P_{Y}), \tag{11}$$

where A_{v} represents the cross-sectional area of the tip, P_{atm} is the atmospheric pressure, and P_{Y} is the yield pressure, which is the minimum pressure required to initiate the eversion process. The yield pressure P_{Y} depends on several factors, including the material properties of the vine robot's membrane, such as stiffness and thickness. While the yield pressure is dynamically complex, it is typically treated as a constant in our model for computational simplicity and to align with prior research on vine robots [23].

In practice, P_{Y} could vary based on environmental conditions or material fatigue. However, in this study, we assume it remains constant throughout the simulation. The specific value used for P_{Y} in our model is based on experimental data from the literature [10], which provides validated pressure values for materials with similar characteristics.

To model the system's bending response, we incorporate torsion springs at the points of interaction with the environment. This approach can be mathematically expressed as follows:

$$\tau = K\boldsymbol{\theta}_{e} \tag{12}$$

where $K = \operatorname{diag}(k, k)$ is the stiffness matrix and $\boldsymbol{\theta}_{e}$ denotes the angles of rotation around the x and z axes of the body frame due to interaction with the environment. It is important to note that in this study, we assume a constant stiffness matrix for simplicity and computational efficiency, as validated in previous works [25]. However, we acknowledge that in real-world scenarios, the stiffness may vary along the robot's length.

For extended lengths of the vine-growing robot, the weight becomes significant, as it is proportional to the free length l in space (excluding the inverted tail and neglecting the weight of the air within the tube). This weight generates a restorative moment at the nearest support point. In addition, the robot may carry payloads at its tip, such as cameras or other scientific instruments, which further increase the lateral weight. The total weight acting on the robot, as illustrated in Figure 4, can be expressed as follows:

$$W = \mu l + m_t g, \tag{13}$$

where l is the unconstrained length, μ is the weight per unit length, m_t is the mass attached at the tip, and g is the acceleration due to gravity. This total weight W generates a moment at the nearest support point, which is counteracted by a restorative moment M, provided by the weight of the pressure chamber.

The deflection of the vine robot due to gravitational effects and additional weight is negligible, primarily because the stiffness induced by the internal pressure within the robot's membrane is significantly greater than the forces caused by these weights. For this assumption to hold, the internal pressure must be sufficiently high to ensure that deflections remain minimal, which justifies the exclusion of these effects in the development of the dynamic equation in (10). Specifically, the principal stress at any point along the backbone of the vine robot must be nonnegative to prevent any significant deformation [34].

To check this posteriori, the axial stress Σ_{XX} at the point $(X = 0, Y = 0, Z = -R)$ must satisfy:

$$\begin{aligned} \Sigma_{XX} = \sigma_L + \frac{M}{I}Z &= \frac{(P - P_{\text{atm}})R}{2t} - \frac{M}{\pi R^3 t}R \geq 0 \\ \Leftrightarrow \ M &\leq \frac{(P - P_{\text{atm}})\pi R^3}{2}, \end{aligned} \tag{14}$$

where R is the radius of the vine robot, P is the internal pressure, P_{atm} is atmospheric pressure, t is the membrane thickness, and M is the moment generated by the weight W. If the internal pressure P drops below a certain threshold governed by the inequality equation in (14), wrinkles may begin to form at the point $(X = 0, Y = 0, Z = -R)$, leading to bending. Such deformation would invalidate the assumptions made in the dynamic equation (10), as it does not account for these effects. This is why, under normal operating conditions, the pressure is maintained high enough to keep the robot's structure stiff and the deflections negligible.

The vine-growing robot can be modeled as a fluid capacitor with a capacitance C_f. When the robot is not fully extended, the volume flow rate, considering the vine robot as an ideal fluid capacitor, is given by:

$$Q = C_f \dot{P} + A_v \dot{l}, \tag{15}$$

where Q represents the volumetric flow rate of air supplied by the compressor, C_f is the fluid capacitance, $\dot{P}$ is the rate of pressure change, A_v is the cross-sectional area at the robot's tip, and $\dot{l}$ is the rate of change of the everted length.

The fluid capacitance for a given volume of fluid is typically defined as follows:

$$C_f = \frac{\Delta V}{\Delta P}, \tag{16}$$

where ΔV represents the change in the volume of the entrapped fluid and ΔP is the corresponding change in pressure.

Assuming the vine robot is in thermal equilibrium with the environment and experiences slow changes in pressure and volume, allowing it to maintain a constant temperature, the fluid capacitance of air under isothermal conditions is given by:

$$C_f = \frac{V}{P}, \tag{17}$$

where the volume of air V is defined as follows:

$$V = A_v l. \tag{18}$$

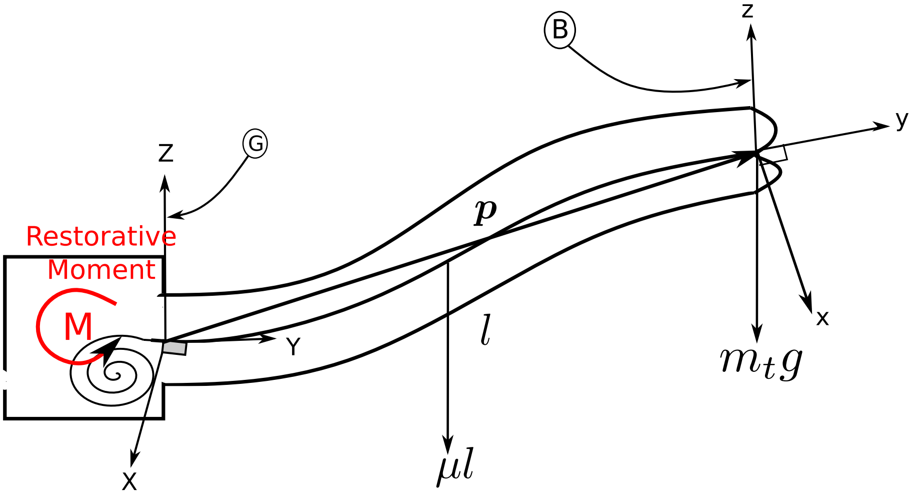

Figure 4: Dynamic force balance of the vine robot. The diagram illustrates the restorative moment M generated by the structure to counteract the gravitational forces and other external forces acting on the robot's extended length l.

In this context, l denotes the everted length of the vine robot and A_v is the cross-sectional area at the robot's tip.

When the vine-growing robot is fully extended, the volumetric flow rate into the robot can still be modeled as an ideal capacitor, described by:

$$Q = C_{\text{fl}}\dot{P}, \tag{19}$$

where C_{fl} represents the fluid capacitance, which consists of two components: one arising from the compressibility of air and the other from changes in volume due to the axial stiffness of the vine robot's membrane. The fluid capacitance C_{fl} is expressed as follows:

$$C_{\text{fl}} = C_s + \frac{V}{P}, \tag{20}$$

where C_s is defined as follows:

$$C_s = \frac{\Delta V}{\Delta P} = \frac{A_v \Delta l}{\frac{k_s \Delta l}{A_v}} = \frac{A_v^2}{k_s}. \tag{21}$$

In this context, k_s represents the spring stiffness of the vine robot's body. By substituting equation (21) into the equation for C_{fl}, the total fluid capacitance becomes:

$$C_{\text{fl}} = \frac{A_v^2}{k_s} + \frac{V}{P}. \tag{22}$$

Combining equations from (15) to (22), the rate at which the internal pressure P changes is given by:

$$\dot{P} = \begin{cases} \dfrac{(-A_v\dot{l} + Q)P}{A_v l} & \text{for } l < l_T, \\ \dfrac{Qk_sP}{A_v(A_vP + k_sl)} & \text{otherwise}, \end{cases} \tag{23}$$

where l_T is the length of the vine robot when it is fully extended. Furthermore, this length imposes physical constraints on the robot. When the robot is fully extended, the velocity and acceleration of the tip cease, as the materials predominantly used to construct the vine-growing robot, such as thin-walled low-density polyethylene, are flexible but inextensible with relatively high axial stiffness. This constraint can be mathematically represented as follows:

$$\dot{l} = \begin{cases} 0 & \text{for } l \geq l_T, \\ \dot{l} & \text{otherwise}, \end{cases} \tag{24}$$

$$\dot{\boldsymbol{p}} = \begin{cases} \boldsymbol{0} & \text{for } l \geq l_T, \\ \dot{\boldsymbol{p}} & \text{otherwise}, \end{cases} \tag{25}$$

$$\ddot{\boldsymbol{p}} = \begin{cases} \boldsymbol{0} & \text{for } l \geq l_T, \\ \ddot{\boldsymbol{p}} & \text{otherwise}. \end{cases} \tag{26}$$

These conditions indicate that the robot's tip cannot extend further and that the tip's velocity and acceleration are zero once the maximum length is reached.

3.2 The steering model

In contrast to passive environment steering, where the soft growing robot naturally bends around obstacles and follows existing pathways in cluttered and narrow spaces due to its inherent compliance, an active steering mechanism is necessary to control the growth direction for specific tasks. This is especially important in applications that require the creation of deployable and reconfigurable structures. Various steering mechanisms have been developed to control the growth direction of soft growing robots, all based on the principle of altering the relative length of the material on opposite sides of the flexible thin-walled tube. This method induce asymmetric growth by shortening one side to change the robot's shape and achieve turns, enabling it to curve in space. Recent innovative methods for steering the everting vine robot, which mimic how vine plants explore their surroundings in search of water and nutrients without moving already grown sections, are presented in [11,35].

This study considers a mutually independent, on-demand active steering mechanism as presented in [11]. This steering method uses four pairs of miniature electromagnets and circular plates, which are uniformly distributed at 90° intervals around the backbone's cross section to store parts of the robot. These electromagnet units are attached along the robot's body with double-sided tape. When energized, the electromagnets release the stored body sections between the electromagnets and circular plates, causing asymmetric growth. This process imitates how vines avoid obstacles and move toward their targets. The steering model that controls the motion trajectory of the vine-growing robot, as illustrated in Figure 5, relates the bending angle to the length of the stored section as follows:

$$\begin{gathered} L_m = \theta_m R_m, \\ \sin\frac{\theta_m}{2} = \frac{L_0/2}{R_m - D}, \quad m = 1, 2, 3, 4, \end{gathered} \tag{27}$$

where L_m is the length of the body stored between the electromagnet and circular plate, θ_m is the bending angle, R_m is the radius of curvature of each pair, L_0 is the thickness of the electromagnet unit after absorption (*i.e.*, when

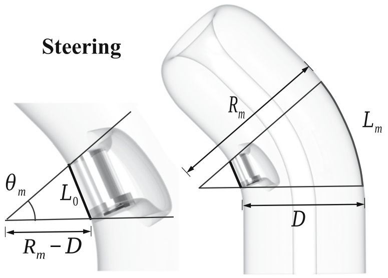

Figure 5: Schematic diagram of the steering kinematic model [11].

the electromagnet is in contact with the circular plate), and D is the diameter of the vine-growing robot. The turning mode utilized in this study ensures that at any given moment, only one electromagnet unit is activated at each joint to steer the robot, or none are activated, allowing the robot to continue moving in the same direction as mentioned earlier. When no electromagnet unit is activated ($L_m = 0$), the bending angle θ_m is zero, and the radius of curvature R_m is infinite, forming a straight line at each unit. The rotation angles α_k and β_k around the body frame's x and y axes, respectively, in terms of the bending angles θ_m are defined as follows:

$$\begin{aligned}\alpha_k &= \theta_1 - \theta_2,\\ \beta_k &= \theta_3 - \theta_4.\end{aligned} \tag{28}$$

The rotation angles α_k and β_k, necessary for the steering mechanism to control the growth direction, are computed as follows:

$${}^{G}A_B(\varphi_k, \theta_k, \psi_k) = {}^{G}A_B(\varphi_{k-1}, \theta_{k-1}, \psi_{k-1})R_x(\alpha_k)R_z(\beta_k). \tag{29}$$

The bending angles $\boldsymbol{\theta} = [\theta_1\ \theta_2\ \theta_3\ \theta_4]^T$ and the lengths of the body sections $\boldsymbol{L} = [L_1\ L_2\ L_3\ L_4]^T$, stored between the electromagnetic units of the steering mechanism, are derived from Eqs (28) and (27), respectively.

This formulation allows for the comprehensive analysis and control of the vine robot's behavior by incorporating both the growth and steering dynamics within a unified state-space framework.

3.3 Model description

The development of a precise model is crucial for the implementation of model-based control. In this study, we formulate the model using continuous ordinary differential equations and represent it in state-space form. The state vector is defined as $\boldsymbol{x} = [\boldsymbol{p}^T\ \dot{\boldsymbol{p}}^T\ l\ P]^T$, encapsulating the system's state. The input vector to the system is given by $\boldsymbol{u} = [Q\ \boldsymbol{L}^T]^T$. Consequently, the nonlinear model describing the dynamics of the everting vine robot in state-space form is expressed as follows:

$$\begin{aligned}\dot{\boldsymbol{x}} &= f(\boldsymbol{x}, \boldsymbol{u}),\\ \boldsymbol{y} &= \boldsymbol{x}.\end{aligned} \tag{30}$$

This model effectively captures the complex interplay between the growth and steering mechanisms, enabling precise control of the vine robot's movements and interactions with its environment.

4 Stress assessment

The internal pressure within the vine robot generates axial stress that propagates along the backbone's wall to the base. Due to the very small ratio of membrane thickness to diameter, the vine robot primarily experiences circumferential (hoop) stress and longitudinal (axial) stress when pressurized [36]. If the internal pressure causes the stress to exceed the membrane's maximum capacity, the robot will burst. For simplicity, the tip surface is approximated as a perfect circle with the same diameter as the backbone. The longitudinal stress, assumed to be uniform across the cross-section of the backbone, is given by:

$$\sigma_L = \frac{(P - P_{\text{atm}})D}{4t}, \tag{31}$$

where t is the thickness of the membrane.

Assuming the circumferential stress is uniformly distributed along the length of the vine robot, the circumferential (hoop) stress is expressed as follows:

$$\sigma_\theta = \frac{(P - P_{\text{atm}})D}{2t}. \tag{32}$$

The circumferential and longitudinal stresses must always be nonnegative and below the yield (burst) stress of the membrane during the robot's operation.

5 Data-driven dynamic model

In this work, we leverage the power of deep learning techniques due to their flexibility and general representation capabilities to develop a data-driven dynamic model of the everting vine robot using a DNN, as depicted in Figure 6. This DNN serves as a discrete-time dynamic model that can be integrated with model-based controllers. The primary objective of this DNN is to learn the nonlinear mapping from the system state $\boldsymbol{x}_k$ and input $\boldsymbol{u}_k$ at time step k to the

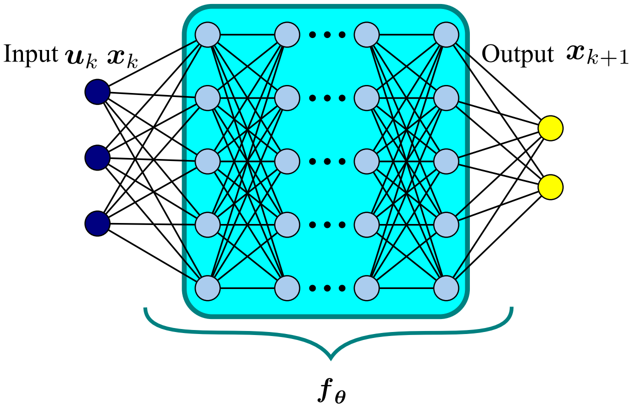
Input $\boldsymbol{u}_k$ $\boldsymbol{x}_k$
Output $\boldsymbol{x}_{k+1}$
$\boldsymbol{f}_\theta$

Figure 6: DNN architecture for the everting vine robot.

future state $\boldsymbol{x}_{k+1}$. The nonlinear discrete-time dynamic model is formulated as follows:

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}_\theta(\boldsymbol{x}_k, \boldsymbol{u}_k), \tag{33}$$

where $\boldsymbol{f}_\theta$ represents the dynamics that map the state of the system forward in time and θ denotes the network weights. This approach enables the modeling of complex, nonlinear behaviors inherent in the vine robot's dynamics, facilitating more accurate and efficient control.

The DNN employed in this research utilizes fully connected layers with hyperbolic tangent activation functions. It comprises three hidden layers: the first with 32 neurons, the second with 16 neurons, and the third with 8 neurons. The architecture is completed with an output layer that uses a linear activation function to predict the future state $\boldsymbol{x}_{k+1}$. The choice of activation type, the number of layers, and the number of neurons in each layer is somewhat arbitrary. It is important to note that widening or deepening of the network is straightforward but has a minimal impact on the network's performance and can complicate the tuning process.

The training data for the DNN are generated from nonlinear simulations of the first-principles model described in Eq. (30). These simulations are conducted using the state-space model with the do-mpc simulator [37] at a sampling time of 0.08 s. The model is driven by sinusoidal forcing functions: $(0.0332\sin t^*)^2$ for the volumetric flow rate Q and $2\pi\sin t^*$ for the Euler angles φ, θ, and ψ, where t^* represents time. Prior to training, the data are normalized and then split into 80% for training and 20% for validation. This ensures that the DNN is trained on a comprehensive set of scenarios, enhancing its ability to accurately predict future states under various conditions.

The DNN model is designed and trained using Keras with TensorFlow as the backend engine due to its simple application programming interface and flexibility. After training, normalization layers are incorporated before the input layer and after the output layer to facilitate seamless integration with model-based controllers, ensuring that the input and output data are normalized and denormalized accordingly. Subsequently, the open neural network exchange (ONNX) [38] framework is employed to convert the model from the TensorFlow computation graph to the ONNX format using the `tf2onnx` package. This conversion enhances interoperability with other Python tools, enabling broader application and integration of the trained model across various platforms.

6 MPC

In this section, we present MPC as a method to simultaneously control the growth and steering of the everting vine robot in free space. MPC, schematically depicted in Figure 7, functions through real-time, iterative optimization using a mathematical surrogate model of the system. The surrogate model employed in this research is a DNN-based discrete-time dynamic model defined in Eq. (33). This approach allows for predicting the future behavior of the robot by solving a potentially constrained optimization problem to determine the optimal trajectory of the manipulated variable $\boldsymbol{u}$ [21]. At each time step, only the initial value of the optimized output trajectory is applied to the system. The optimization and prediction processes are repeated continuously, ensuring that the output $\boldsymbol{y}$ closely follows the desired reference trajectory $\boldsymbol{r}$ over the prediction horizon N_2 (length of the prediction window), as illustrated in Figure 8. For computational efficiency, the lower prediction horizon is set to $N_1 = 1$ (starting point for predictions) to accommodate the time required for implementing the optimal control input $\boldsymbol{u}$ in the subsequent time step. Furthermore, the control horizon N_u (number of future steps for control actions) is chosen to be equal to

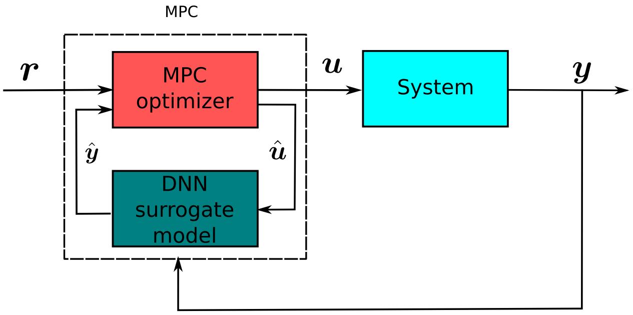

Figure 7: Schematic diagram of model-based predictive control (MPC) for the everting vine robot.

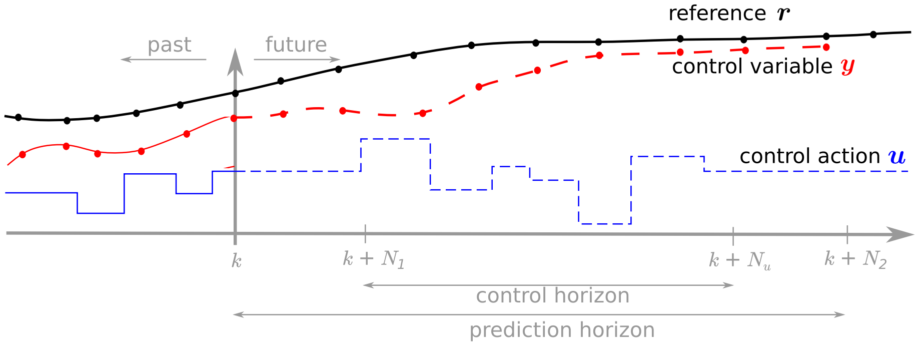

Figure 8: Illustration of the functional principle of MPC for tracking the reference trajectory, with prediction horizons N_1 and N_2, and control horizon N_u (in accordance to the study by Schwenzer *et al.* [21]). The diagram demonstrates how MPC optimizes and predicts at each time step to ensure the output $\boldsymbol{y}$ follows the desired reference $\boldsymbol{r}$.

the prediction horizon N_2, simplifying the tuning of the optimizer.

MPC's widespread adoption in robotics for controlling real systems is attributed to its ability to incorporate constraints directly into the optimization process and its effectiveness in handling nonlinear models, which often exceed the capabilities of classical controllers. This flexibility makes MPC an invaluable tool for achieving precise and reliable control in complex robotic applications.

6.1 Cost function

To achieve precise tracking of the desired reference trajectory while ensuring smooth control behavior, a cost function must be defined to penalize deviations in both the manipulated variable and the trajectory. This cost function is minimized by the optimizer and accounts for deviations from the reference trajectory $\boldsymbol{r}$ as well as changes in the manipulated variable over the prediction horizon N_2. The cost function to be minimized at each time step k is defined as follows:

$$\begin{aligned} J = &\sum_{i=N_1}^{N_2} \|\boldsymbol{r}(k+i|k) - \boldsymbol{y}(k+i|k)\|^2_{W_r} \\ &+ \sum_{j=1}^{N_u-1} \|\Delta\boldsymbol{u}(k+j|k)\|^2_{W_u} \\ &+ \sum_{i=N_1}^{N_2} \|\boldsymbol{\xi}(k+i|k)\|^2_{W_\xi} \end{aligned} \tag{34}$$

s.t.

$$\begin{aligned} \boldsymbol{u}_{lb} \le \boldsymbol{u}(k+j|k) &\le \boldsymbol{u}_{ub}, \\ \boldsymbol{y}_{lb} - \boldsymbol{\xi}(k+i|k) \le \boldsymbol{y}(k+i|k) &\le \boldsymbol{y}_{ub} + \boldsymbol{y}(k+i|k), \\ \text{where} \quad \boldsymbol{\xi} &\ge 0, \end{aligned}$$

where $\boldsymbol{y}(k+i|k)$ represents the predicted control variable at time $k+i$ given the system state at time step k, and $\Delta\boldsymbol{u}_k = \boldsymbol{u}_k - \boldsymbol{u}_{k-1}$ denotes the change in the manipulated variable. The slack variable $\boldsymbol{\xi}$ quantifies the permissible violation of output constraints, allowing for the softening of constraints within the cost function. The weight matrix W_ξ controls the extent of this violation. In addition, the weight matrices $W_r \ge 0$ and $W_u > 0$ are positive semi-definite and positive definite, respectively, controlling the trade-off between accurate trajectory tracking and smooth control behavior.

This formulation ensures that the system not only tracks the reference trajectory accurately but also does so with minimal abrupt changes in the control inputs, leading to more stable and efficient operation.

6.2 Constraints and bounds

The system imposes physical constraints and limits on both the actuators and the output $\boldsymbol{y}$ to ensure the robot operates within safe and feasible regions, thereby preventing potential damage to the actuators. To achieve this, these constraints and bounds must be incorporated into the optimization loop, ensuring that the optimal trajectory of the manipulated variable $\boldsymbol{u}$ and the system states remain within acceptable ranges. The specific constraints and bounds considered in this study are as follows:

$$\begin{aligned} \dot{l} - \omega r_{sp} &\le 0, \\ P_{\text{atm}} + P_Y \le P &\le P_{\text{max}}, \\ 0 \le Q &\le Q_{\text{max}}, \\ -2\pi \le \varphi, \theta, \psi &\le 2\pi, \\ 0 \le \boldsymbol{L} &\le L_{\text{max}}, \end{aligned} \tag{35}$$

where $\dot{l}$ represents the rate of change of the everted length, P_{max} denotes the maximum allowable pressure, Q_{max}

specifies the maximum volumetric flow rate that can be supplied by the compressor, and $L_{\max}$ denotes the maximum stored length between the electromagnet units of the steering mechanism. These constraints are crucial for ensuring that the system operates both safely and efficiently, staying well within its physical and operational limits.

6.3 Implementation of MPC

The MPC approach proposed in this study for controlling the vine-growing robot is implemented using the do-mpc framework [37], a Python-based platform for nonlinear and robust MPC. This framework leverages CasADi [39] for symbolic computation and Ipopt [40] for solving nonlinear optimization problems, making it well suited for evaluating the performance and robustness of the proposed controller.

To integrate the DNN-based discrete-time dynamic model of the vine-growing robot, as detailed in Section 5, the model is first converted into a CasADi symbolic expression using the do-mpc ONNXConversion class, which facilitates the conversion from ONNX format. This conversion enables the model to operate as a discrete-time dynamic model within the MPC framework. The nonlinear optimization problem is addressed using Ipopt, which leverages the MA27 linear solver, while the dynamic model's simulation is carried out using the cvodes tool. This approach ensures precise simulation and optimization, thereby supporting the robust real-time control of the vine-growing robot.

7 Results and discussion

To evaluate the effectiveness of the DNN-based discrete-time dynamic model of the vine-growing robot and assess the performance of the nonlinear robust MPC, simulations are conducted in two distinct settings: open-loop prediction and closed-loop control.

First, an open-loop prediction is performed using the proposed DNN model to compare the predicted system states with those obtained from the nonlinear first-principles model. This comparison allows us to assess the accuracy of the DNN in estimating the system's dynamics.

Second, to evaluate the effectiveness and performance of the proposed MPC in controlling both the growth and steering of the vine-growing robot, a closed-loop simulation is conducted. This simulation ensures that the system states achieve the desired setpoints or follow the specified trajectory under the control of the MPC. The closed-loop simulations are executed using the do-mpc framework.

Table 1: Simulation parameters for vine-growing robot

Parameter	Description	Value
m	Fictitious mass	10^4 kg
l_T	Fully extended length of the robot	1.2 m
P_{atm}	Atmospheric pressure	101325 Pa
P_Y	Yield pressure	3.210×10^3 Pa
r_{sp}	Radius of the spool	0.025 m
D	Diameter of the membrane	0.0798 m
t	Thickness of the membrane	0.0001 m
L_0	Thickness of the electromagnet unit	0.012 m
k	Torsional stiffness	22 Nm/rad
μ	Weight per unit length	0.008 N/m
k_s	Stiffness of the membrane	20×10^6 N/m
A_v	Cross-sectional area of the tip	5×10^{-3} m^2
g	Gravitational acceleration constant	9.81 m/s^2
R	Radius of the membrane	0.0399 m

The parameters used for simulating the vine-growing robot are presented in Table 1. Notably, the magnitude of the fictitious mass m attached to the tip of the vine robot has been carefully selected to accurately represent the robot's slow and deliberate movement, which is crucial for applications like exploration. A larger fictitious mass is essential to model the slow acceleration and precise movements characteristic of the vine robot. Conversely, choosing a smaller mass would result in much higher acceleration at the tip, akin to a rocket's behavior, which is contrary to the intended operation of the vine robot. This choice of mass enables a more realistic simulation of the robot's behavior within the computational model. The remaining parameters used in this simulation were obtained from existing literature [10,35]. Open-loop simulations of the nonlinear first-principles model, as presented in the literature [41], were conducted to validate the dynamic model of the vine robot.

7.1 Performance of the DNN dynamic model of the vine robot

In this section, we evaluate the performance of the DNN-based discrete-time dynamic model for the pressure-driven everting vine robot, which is intended for integration into the MPC as a surrogate model. The DNN model is trained using Keras with TensorFlow as the backend engine, running on an Intel Core i5-4300M CPU @ 2.60 GHz with 7.4 GB of RAM. This setup allows us to assess the model's accuracy and computational efficiency, ensuring it is suitable for real-time control applications within the MPC framework.

To evaluate the performance of the proposed DNN-based dynamic model for the vine robot, the training and validation loss were monitored over 200 epochs on normalized data. The Adam optimizer was employed with a learning rate of 0.001. Figure 9 illustrates the training and validation loss curves, both of which show the mean squared error (MSE) steadily converging. The rapid decrease in MSE during the initial epochs indicates effective learning, and the subsequent stabilization suggests that the model is not overfitting, maintaining consistent performance across both training and validation datasets.

Figure 10 illustrates the open-loop predictions of the system states using the DNN-based model, compared to the actual states derived from the first-principles model. The comparison shows that the differences between the DNN-predicted states and the first-principles model's states are minimal, indicating a high level of accuracy in the DNN model's predictions. This close alignment validates the use of the proposed DNN model as an effective surrogate for the more complex first-principles model, demonstrating its potential for integration into the MPC framework.

To assess the generalization performance of the proposed DNN for the pressure-driven everting vine robot, a K-fold cross-validation approach was employed, as recommended by Marcot and Hanea [42]. This method is vital for identifying a robust architecture that exhibits minimal sensitivity to variations in training data partitions. In this study, the dataset was divided into five folds, and five distinct feed-forward DNN models were evaluated for performance. These models, varying in complexity, each comprise fully connected hidden layers as detailed in Table 2. The layers are denoted by “*F*” with subscripts indicating the number of neurons in each layer. All hidden layers utilize hyperbolic tangent activation functions, while the output layer employs a linear activation function. This systematic evaluation ensures the selection of a DNN

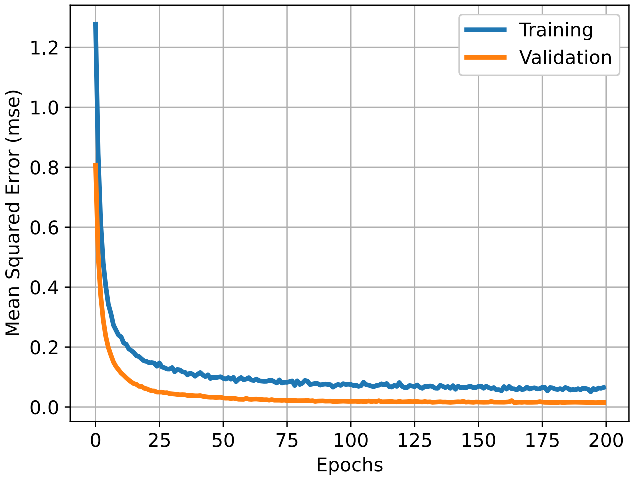

Figure 9: Training performance of the DNN model.

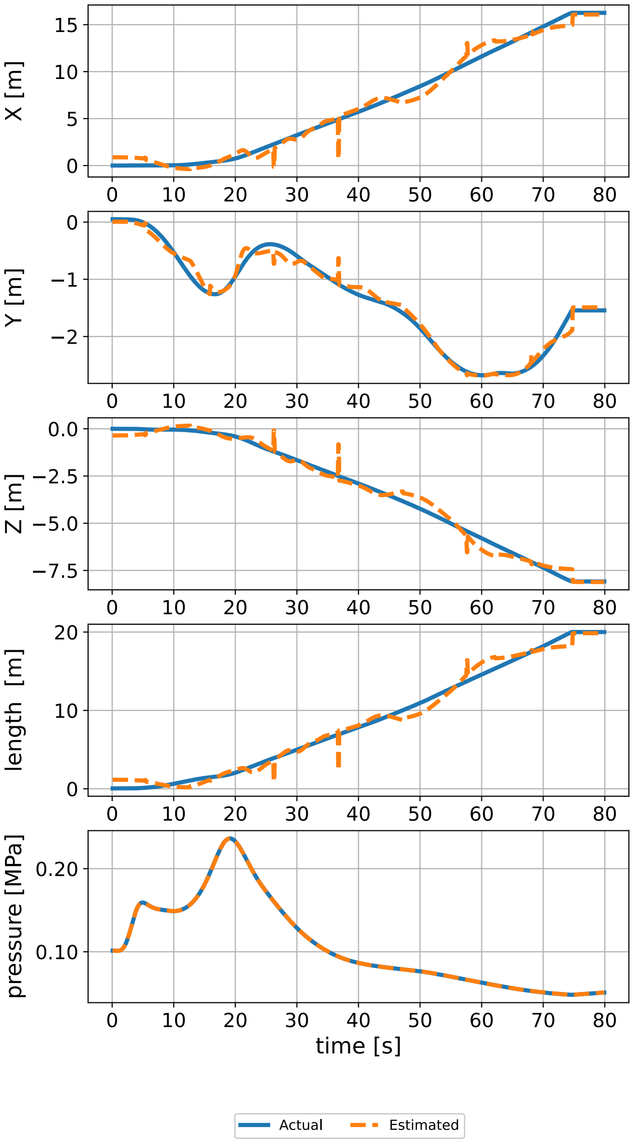

Figure 10: Comparison of the time evolution of system states between the nonlinear first-principles model and the DNN-predicted model.

Table 2: Comparison of models with different hidden layer configurations and their corresponding average MSE with standard deviation

Model	hidden layers	Avg. MSE (×10^{-2})
1	F_{128}, F_{32}, F_{16}	7.78 ± 0.95
2	F_{256}, F_{128}, F_{32}	5.14 ± 0.47
3	F_{512}, F_{256}, F_{32}	4.99 ± 0.28
4	F_{512}, F_{512}, F_{128}, F_{64}	4.31 ± 0.85
5	F_{512}, F_{512}, F_{128}, F_{64}	3.68 ± 0.48

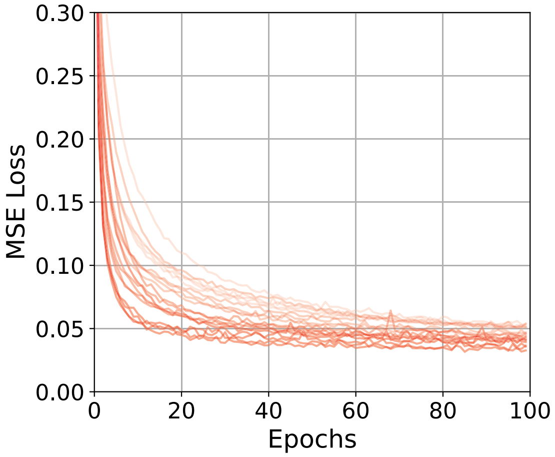

Figure 11: Training losses for five DNN models during five-fold cross-validation.

architecture that balances complexity and performance, optimizing the model's ability to generalize to unseen data.

The models were trained for 100 epochs using the Adam optimizer, with MSE as the primary loss metric. The convergence of the learning curves during five-fold cross-validation for all network configurations is depicted in Figure 11. The learning curves demonstrate that models with higher complexity, indicated by darker shades of red, tend to converge faster and stabilize more effectively, reflecting the models' capacity to capture the underlying dynamics of the system more accurately.

In addition, Figure 12 illustrates the average mean squared error and their standard deviations across different model architectures. The results show a clear trend where increased model complexity leads to lower MSE values, indicating improved predictive performance. However, this comes with the trade-off of increased computational cost and potential overfitting, as evidenced by the greater variability in some of the more complex models. These findings highlight the critical need to balance model complexity and generalization capability when selecting the optimal architecture for the vine robot's dynamic model, ensuring both accuracy and robustness in real-world applications.

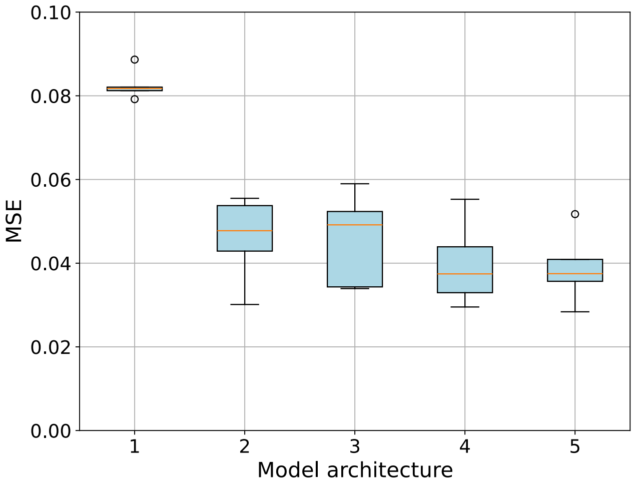

Figure 12: Average MSE and standard deviation.

7.2 Closed-loop simulation

To evaluate the performance and capabilities of integrating the DNN-based discrete-time dynamic model with the MPC framework, we conducted a comparative analysis between the DNN dynamic model and the first-principles model across three tracking control scenarios: set-point stabilization, obstacle avoidance, and trajectory tracking. The MPC was designed using both the first-principles model and the DNN model, allowing us to assess the performance improvements offered by a data-driven approach in controlling a nonlinear dynamic system in real-time applications. This comparison highlights the potential advantages of utilizing DNN-based models for enhanced accuracy and efficiency in complex control tasks.

7.2.1 Set-point stabilization

The performance of the proposed MPC, utilizing both DNN-based and first-principles dynamic models, is evaluated based on its ability to precisely track a desired setpoint in space. This capability is particularly vital for applications such as medical procedures, where the vine robot may be required to deliver drugs accurately to a specific organ from its base. In this study, the control objective is to guide the robot to a specific position in Cartesian space, ensuring that it comes to rest with the desired force. The target state is set as $\boldsymbol{r} = [0.3\ \ -0.8\ \ -0.1\ 0\ 0\ 0\ 2 \times 10^5]^T$, which includes the desired position, velocity, and pressure.

The weighting matrices used in the MPC formulation are $W_r = I_7$ and $W_u = 0.01I_5$, where I_n denotes an $n \times n$ identity matrix, reflecting a balanced emphasis on state tracking and control effort. These matrices were carefully tuned to achieve optimal tracking performance, ensuring a balance between accurate trajectory tracking and smooth control input adjustments. The simulation is conducted with a time step of $T_s = 0.08$ s, and the prediction horizon is set to $N_2 = 20$ steps. To ensure safe operation, the maximum allowable pressure $P_{\max}$ and air volume flow rate $Q_{\max}$ are constrained to 241,325 Pa and 0.01 m^3/s, respectively.

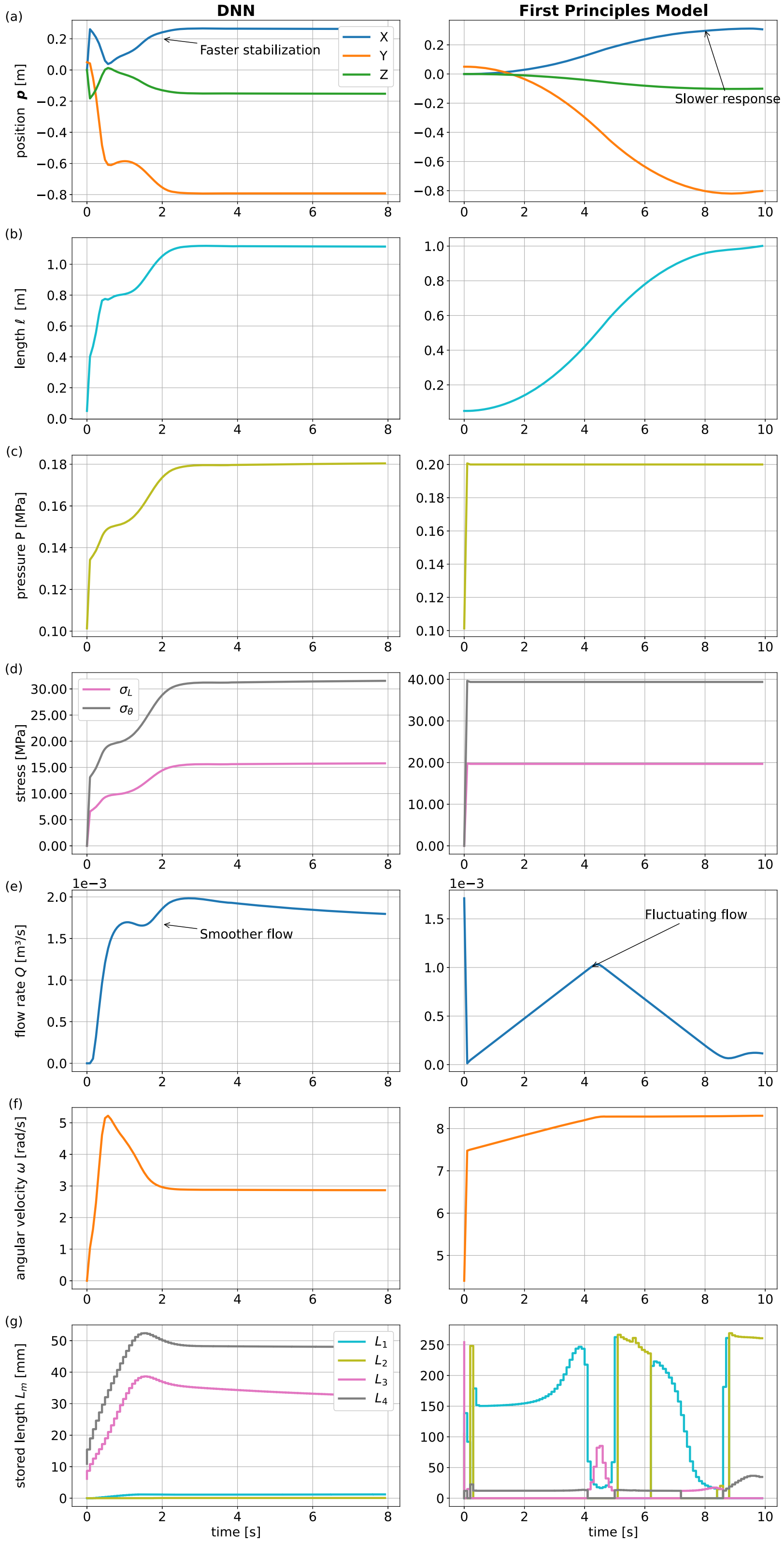

Figure 13: Comparison of the closed-loop control performance of the DNN-based dynamic model and the first-principles model for set-point stabilization simulation. (a) Position tracking $\boldsymbol{p}$. (b) Length l of the robot. (c) Pressure P of the system. (d) Stress response in longitudinal (σ_L) and circumferential (σ_θ) directions. (e) Volumetric flow rate (Q). (f) Angular velocity (ω). (g) Stored lengths (L_1, L_2, L_3, and L_4) during the actuation process.

In addition to the system constraints outlined in Section 6, the robot's position is further constrained within the following bounds:

$$\begin{bmatrix} -4 \\ -4 \\ -4 \end{bmatrix} \le \boldsymbol{p} \le \begin{bmatrix} 4 \\ 4 \\ 4 \end{bmatrix}.$$

These constraints ensure that the robot operates within a safe and feasible region while achieving the control objectives. This scenario highlights the effectiveness of the proposed MPC in maintaining precise control of the vine robot under the specified conditions.

The dynamic model is initialized with initial state of

$$\boldsymbol{x}_0 = [0\ 0.05\ 0\ 0\ 0\ 0\ 0.05\ 101{,}325]^T + 1 \times 10^{-6}.$$

Figure 13 compares the performance of the DNN-based dynamic model with the first-principles model during closed-loop control, illustrating the evolution of system states, stresses, and actuation inputs over time.

In this comparison, both the DNN-based model and the first-principles model achieve effective set-point stabilization, with the DNN model showing slightly faster convergence in the position states (X, Y, Z). The system length also stabilizes more rapidly in the DNN model. In addition, the figure highlights the efficiency of the actuation inputs required for control. In the DNN model, the modulation of the volumetric flow rate (Q) and angular velocity (ω) is smoother, resulting in more efficient transitions compared to the first-principles model. The stored lengths (L_1, L_2, L_3, and L_4) exhibit a more controlled response under the DNN model, particularly in minimizing the overall material usage required to complete the task.

The results indicate that the average time taken by the MPC solver to find the optimal control with the DNN dynamic model is 2.2 times faster than using the first-principles model. This improvement in computational efficiency is particularly useful in real-time applications, where the prediction horizon can be five times shorter compared to that of the analytical model.

Overall, these figures highlight the advantages of the DNN-based model in achieving quicker and more stable control responses while requiring less aggressive actuation. This suggests that the DNN model can offer superior performance in real-time control scenarios, providing a viable alternative to more computationally intensive first-principles models.

7.2.2 Obstacle avoidance

In this section, the effectiveness of the proposed MPC is evaluated in the context of obstacle avoidance, a crucial capability for navigating cluttered environments such as archaeological excavation sites. The obstacle is positioned at $\boldsymbol{x}_{\text{obs}} = [0.2\ \ -0.5\ \ -0.1]^T$ along the robot's path to its desired setpoint. The initial state $\boldsymbol{x}_0$, prediction horizon N_2, desired state $\boldsymbol{r}$, and sampling time T_s are consistent with those utilized in Section 7.2.1.

To ensure the robot successfully avoids the obstacle, a nonlinear inequality constraint is introduced into the optimization problem. This constraint enforces that the Euclidean distance between the tip of the vine robot and the center of the obstacle remains above a specified safety threshold:

$$\|\boldsymbol{p} - \boldsymbol{x}_{\text{obs}}\| \ge r_o + r_v, \tag{36}$$

where $r_o = 0.2$ m and $r_v = 0.0399$ m represent the radii of the robot's tip and the obstacle, respectively. This constraint is crucial for ensuring that the robot maintains a safe distance from the obstacle, thereby preventing collisions and ensuring smooth navigation toward the desired setpoint.

The simulation results using both the DNN-based discrete-time dynamic model and the first-principles model, depicted in Figure 14, demonstrate that the proposed MPC successfully navigated the vine robot around the obstacle and achieved the desired setpoint. The corresponding actuation inputs and system states during the obstacle avoidance maneuver are shown in Figure 15.

The results confirm that the proposed MPC effectively guides the vine robot in avoiding the obstacle while maintaining the desired trajectory. Notably, the DNN-MPC

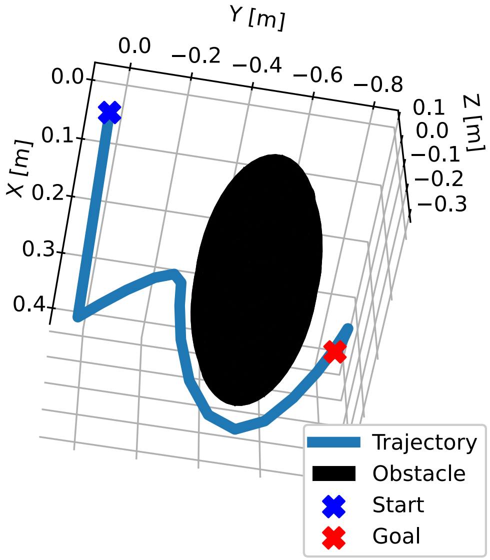

Figure 14: Simulation results of vine robot avoiding an obstacle in Cartesian space.

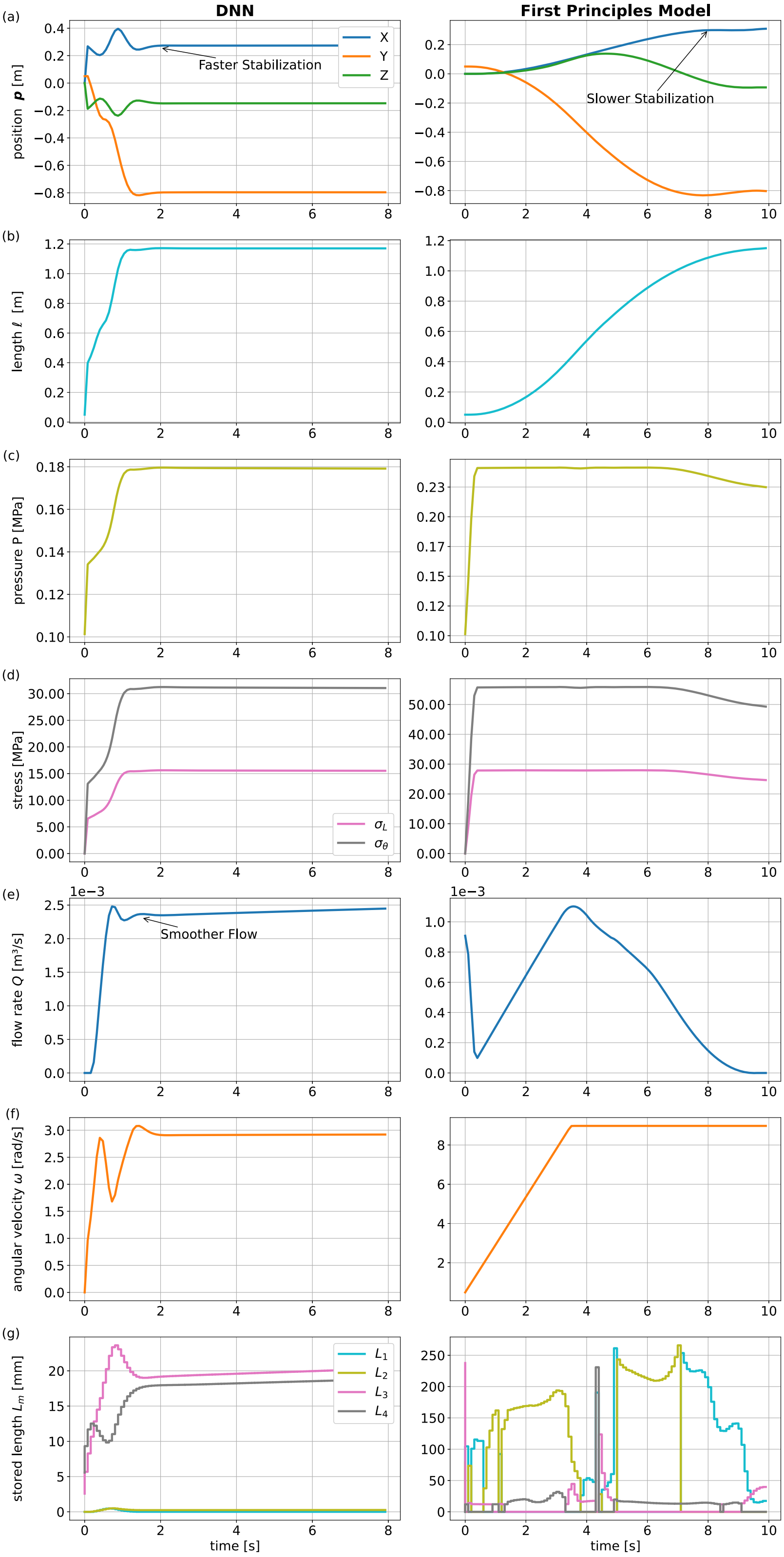

Figure 15: Closed-loop control performance comparison between the DNN-based dynamic model and the first-principles model during obstacle avoidance. (a) Position $\boldsymbol{p}$. (b) Length l. (c) Internal pressure P. (d) Stress distribution. (e) Flow rate Q. (f) Angular velocity ω. (g) Stored lengths for each of the four electromagnet units (L_1, L_2, L_3, L_4).

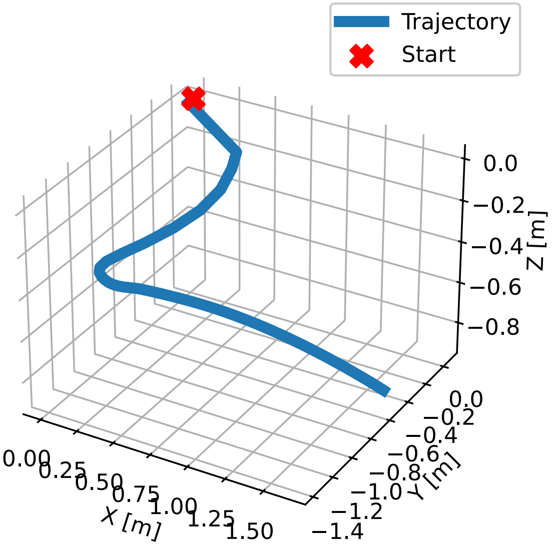

Figure 16: 3D motion of the vine robot while tracking a time-varying trajectory.

architecture significantly outperformed the MPC framework based on the first-principles model in terms of computation time, with the DNN-MPC being 7.2 times faster than its counterpart. This substantial improvement in computational efficiency highlights the DNN-based approach's potential for real-time applications, where quick decision-making is crucial for effective obstacle avoidance and trajectory control.

7.2.3 Trajectory tracking

In this final evaluation, a time-varying trajectory from the dataset is used to assess the effectiveness of the proposed MPC in tracking dynamic paths with the surrogate models. This scenario is particularly relevant for applications requiring a soft, reconfigurable, and deployable antenna, such as those that need to assume various shapes like a helix. The ability to accurately follow such trajectories is crucial for ensuring the antenna's functionality in diverse operational environments.

All tuning parameters for the MPC, including the prediction horizon, weighting matrices, and constraints, are consistent with those presented in Section 7.2.1. This consistency allows for a direct comparison of the controller's performance in both set-point stabilization and more complex trajectory tracking tasks, demonstrating the robustness and versatility of the proposed MPC framework.

Figure 16 illustrates the motion of the vine robot in space while tracking a time-varying trajectory. The performance of the proposed MPC, utilizing both the DNN-based dynamic model and the first-principles model, is presented in Figure 17, along with the corresponding actuation inputs to the system. The results show that the DNN-MPC architecture tracks the trajectory more effectively, achieving faster convergence and more precise alignment with the reference trajectory compared to the first-principles model. Furthermore, the DNN-MPC architecture demonstrates smoother and more efficient actuation, resulting in better overall control performance compared to the first-principles model.

Significantly, the DNN-MPC architecture outperformed the first-principles model-based MPC framework in terms of computation time, with the DNN-MPC being 11 times faster. This substantial improvement highlights the DNN model's suitability for real-time control applications, where computational efficiency is critical for responsive and accurate trajectory tracking.

8 Conclusions

This study proposed a data-driven dynamic model of a pressure-driven everting vine robot using a DNN architecture as a discrete-time dynamic model to control its growth and direction. The DNN model was trained using data generated by the nonlinear simulation of the first-principles model of the everting vine robot. A K-fold cross-validation approach was employed to assess the general performance of the proposed DNN model for the vine robot, which subsequently informed the selection of the optimal architecture based on average MSE. The data-driven model was then integrated into a MPC framework to evaluate its computational tractability and effectiveness in controlling the nonlinear dynamics of the vine robot.

The comparative analysis between the DNN-MPC and the traditional MPC based on the nonlinear first-principles model revealed that the DNN-MPC framework provided superior control performance while significantly reducing computation time. This indicates that the DNN-MPC framework is a promising approach for efficient and effective data-driven nonlinear control of soft robotic systems.

This work opens up several avenues for future research and exploration. Although this study assumed full state measurement, in practice, some states may be difficult to measure directly. In such cases, state estimators like moving horizon estimator can be employed to estimate unmeasured states from the available measurements. In addition, other data-driven discovery techniques, such as sparse identification of nonlinear dynamics (SINDy), could be explored to develop more parsimonious and interpretable models, offering

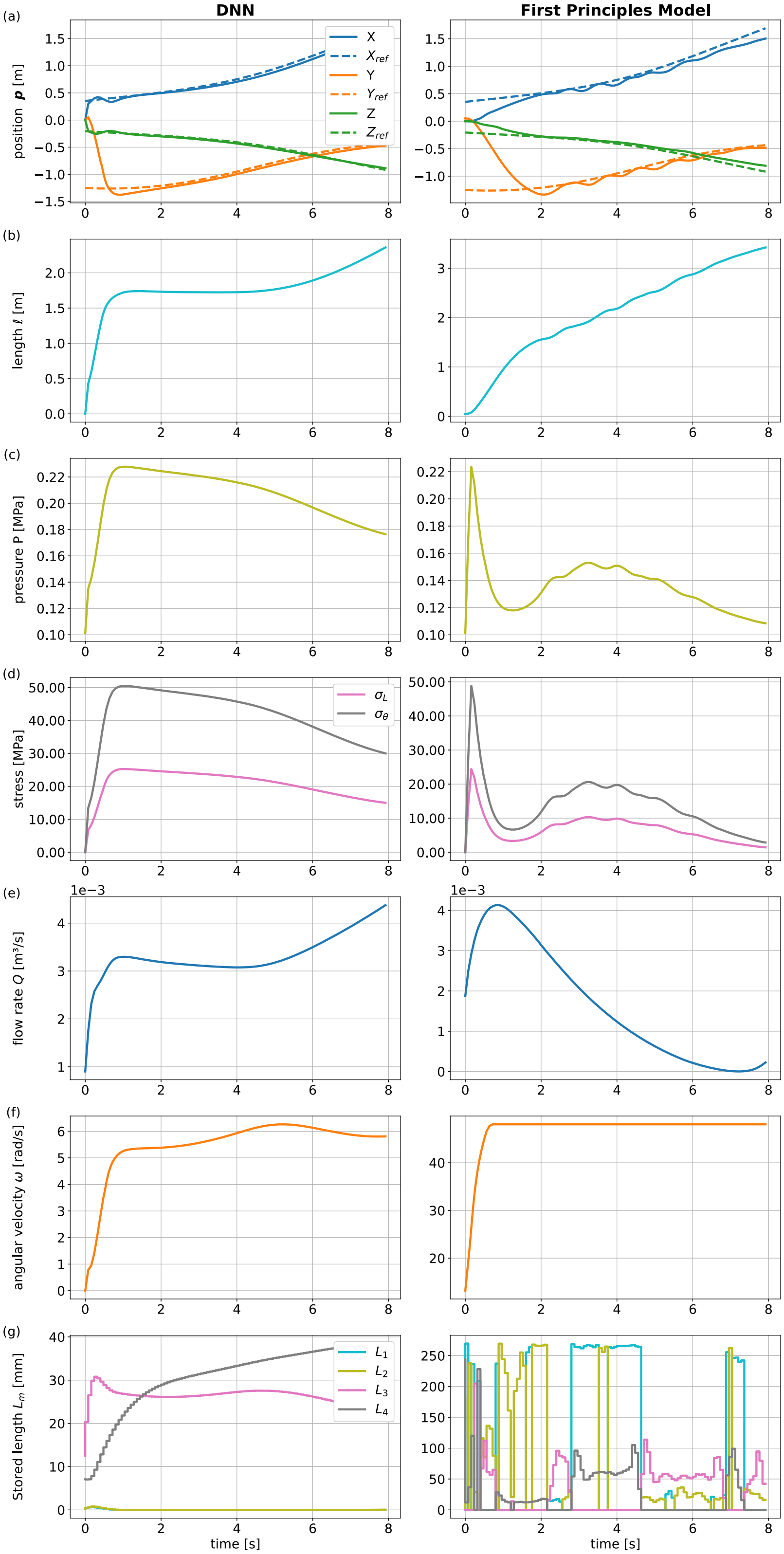

Figure 17: Closed-loop control performance comparison between the DNN-based dynamic model and the first-principles model during for the trajectory tracking task of the vine growing robot. (a) Position $\boldsymbol{p}$. (b) Length l. (c) Internal pressure P. (d) Stress distribution. (e) Flow rate Q. (f) Angular velocity ω. (g) Stored lengths for each of the four electromagnet units (L_1, L_2, L_3, L_4).

greater insights compared to DNN models, which are often considered black boxes and do not easily incorporate known constraints.

Furthermore, DNN models typically require large quantities of significant data to train effectively, especially when dealing with the complex and nonlinear dynamics of soft robots. This can be challenging to obtain and may limit the model's ability to generalize beyond the training data. To address this, physics-informed neural networks (PINNs), which embed known physical laws governing the system into the NNs, could be utilized. PINNs require less training data compared to traditional NNs, thus providing a more robust DNN model that converges faster and generalizes well to unseen data.

In applications involving human–robot interactions, such as minimally invasive surgery, it is not sufficient to control only the position and velocity of the vine robot. Equally important is regulating how the robot responds to forces exerted by its environment. To address this, an impedance controller based on a DNN-discrete model is essential for regulating the interaction between the robot's tip and the human body, where compliance and safety are critical. Such approaches have the potential to significantly enhance the performance, adaptability, and applicability of DNN-based control systems across a wide range of soft robotic applications.

Acknowledgments: This research was supported by EJUST-TICAD8 scholarship through JICA and the government of Japan and Egypt for the first author.

Funding information: The authors state no funding involved.

Author contributions: All authors have accepted responsibility for the entire content of this manuscript and consented to its submission to the journal, reviewed all the results, and approved the final version of the manuscript.

Conflict of interest: The authors state no conflict of interest.

Data availability statement: All data that support the findings of this study are included within the article.

References

[1] Meder F, Baytekin B, Del Dottore E, Meroz Y, Tauber F, Walker I, et al. A perspective on plant robotics: from bioinspiration to hybrid systems. Bioinspirat Biomimetics. 2023 Jan;18(1):015006.

[2] Gährs C, Vidal-Gadea A. Locomotion. In: Encyclopedia of animal cognition and behavior. Cham: Springer International Publishing; 2022. p. 3986–4001.

[3] Mano H, Hasebe M. Rapid movements in plants. J Plant Res. 2021 Jan;134(1):3–17.

[4] Fiorello I, Del Dottore E, Tramacere F, Mazzolai B. Taking inspiration from climbing plants: methodologies and benchmarks-a review. Bioinspirat Biomimetics. 2020 May;15(3):031001.

[5] Dent EW, Gertler FB. Cytoskeletal dynamics and transport in growth cone motility and axon guidance. Neuron. 2003 Oct;40(2):209–27.

[6] Palanivelu R, Preuss D. Pollen tube targeting and axon guidance: parallels in tip growth mechanisms. Trends Cell Biol. 2000 Dec;10(12):517–24.

[7] Vidoni R, Mimmo T, Pandolfi C. Tendril-based climbing plants to model, simulate and create bio-inspired robotic systems. J Bionic Eng. 2015 June;12(2):250–62.

[8] Mazzolai B, Tramacere F, Fiorello I, Margheri L. The bio-engineering approach for plant investigations and growing robots. A Mini-Review. Front Robotics AI. 2020 Sep;7:573014.

[9] Blumenschein LH, Coad MM, Haggerty DA, Okamura AM, Hawkes EW. Design, modeling, control, and application of everting vine robots. Front Robot AI. 2020 Nov;7(November):1–24.

[10] Haggerty DA, Naclerio ND, Hawkes EW. Characterizing environmental interactions for soft growing robots. In: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE; 2019. p. 3335–42.

[11] Li P, Zhang Y, Zhang G, Zhou D, Li L. A bioinspired soft robot combining the growth adaptability of vine plants with a coordinated control system. Research. 2021 Jan;2021:9843859.

[12] Hawkes EW, Blumenschein LH, Greer JD, Okamura AM. A soft robot that navigates its environment through growth. Sci Robot. 2017 Jul;2(8):1–8.

[13] Naclerio ND, Hubicki CM, Aydin YO, Goldman DI, Hawkes EW. Soft robotic burrowing device with tip-extension and granular fluidization. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE; 2018. p. 5918–23.

[14] Putzu F, Abrar T, Althoefer K. Plant-inspired soft pneumatic eversion robot. In: 2018 7th IEEE International Conference on Biomedical Robotics and Biomechatronics (Biorob). vol. 2018-Augus. IEEE; 2018. p. 1327–32.

[15] Blumenschein LH, Gan LT, Fan JA, Okamura AM, Hawkes EW. A tip-extending soft robot enables reconfigurable and deployable antennas. IEEE Robot Automat Lett. 2018 Apr;3(2):949–56.

[16] Webster RJ, Jones BA. Design and kinematic modeling of constant curvature continuum robots: a review. Int J Robot Res. 2010 Nov;29(13):1661–83.

[17] Gravagne IA, Rahn CD, Walker ID. Large deflection dynamics and control for planar continuum robots. IEEE/ASME Trans Mechatronics. 2003 June;8(2):299–307.

[18] El-Hussieny H, Hameed IA, Ryu JH. Nonlinear model predictive growth control of a class of plant-inspired soft growing robots. IEEE Access. 2020;8:214495–503.

[19] Wu Z, Reyzabal MDI, Sadati SMH, Liu H, Ourselin S, Leff D, et al. Towards a physics-based model for steerable eversion growing robots. IEEE Robot Automat Letters. 2023 Feb;8(2):1005–12.

[20] Talas SK, Baydere BA, Altinsoy T, Tutcu C, Samur E. Design and development of a growing pneumatic soft robot. Soft Robotics. 2020 Aug;7(4):521–33.

[21] Schwenzer M, Ay M, Bergs T, Abel D. Review on model predictive control: an engineering perspective. Int J Adv Manufact Tech. 2021 Nov;117(5-6):1327–49.

[22] Kaiser E, Kutz JN, Brunton SL. Sparse identification of nonlinear dynamics for model predictive control in the low-data limit. Proc R Soc A Math Phys Eng Sci. 2018 Nov;474(2219):20180335.

[23] Blumenschein LH, Okamura AM, Hawkes EW. Modeling of bioinspired apical extension in a soft robot. vol. 1; 2017. p. 522–31.

[24] Tutcu C, Baydere BA, Talas SK, Samur E. Quasi-static modeling of a novel growing soft-continuum robot. Int J Robot Res. 2021 Jan;40(1):86–98.

[25] Jitosho R, Agharese N, Okamura A, Manchester Z. A dynamics simulator for soft growing robots. Proceedings - IEEE International Conference on Robotics and Automation. 2021;2021-May (Icra):11775–11781.

[26] Lusch B, Kutz JN, Brunton SL. Deep learning for universal linear embeddings of nonlinear dynamics. Nature Commun. 2018 Nov;9(1):4950.

[27] Sapai S, Loo JY, Ding ZY, Tan CP, Baskaran VM, Nurzaman SG. A deep learning framework for soft robots with synthetic data. Soft Robot. 2023 Dec;10(6):1224–40.

[28] Cespedes A, Terreros R, Morales S, Huamani A, Canahuire R. Dynamic modeling of a soft laparoscope: A deep neural network approach. In: 2022 Latin American Robotics Symposium (LARS), 2022 Brazilian Symposium on Robotics (SBR), and 2022 Workshop on Robotics in Education (WRE). IEEE; 2022. p. 1–6.

[29] Hyatt P, Wingate D, Killpack MD. Model-based control of soft actuators using learned non-linear discrete-time models. Front Robot AI. 2019 Apr;6(APR):1–11.

[30] Greer JD, Morimoto TK, Okamura AM, Hawkes EW. A soft, steerable continuum robot that grows via tip extension. Soft Robot. 2019 Feb;6(1):95–108.

[31] Neppalli S, Jones BA. Design, construction, and analysis of a continuum robot. In: 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE; 2007. p. 1503–7.

[32] Gan LT, Blumenschein LH, Huang Z, Okamura AM, Hawkes EW, Fan JA. 3D electromagnetic reconfiguration enabled by soft continuum robots. IEEE Robot Automat Lett. 2020 Apr;5(2):1704–11.

[33] Coad MM, Blumenschein LH, Cutler S, Zepeda JAR, Naclerio ND, El-Hussieny H, et al. Vine robots. IEEE Robot Automat Magazine. 2020 Sep;27(3):120–32.

[34] Levan A, Wielgosz C. Bending and buckling of inflatable beams: Some new theoretical results. Thin-Walled Struct. 2005 Aug;43(8):1166–87.

[35] Bianchi G, Agoni A, Cinquemani S. A bioinspired robot growing like plant roots. J Bionic Eng. 2023;20(5):2044–58.

[36] Godaba H, Putzu F, Abrar T, Konstantinova J, Althoefer K. Payload capabilities and operational limits of eversion robots. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). vol. 11650 LNAI. Springer International Publishing; 2019. p. 383–94.

[37] Fiedler F, Karg B, Lüken L, Brandner D, Heinlein M, Brabender F, et al. do-mpc: Towards FAIR nonlinear and robust model predictive control. Control Eng Practice. 2023 Nov;140(September):105676.

[38] ONNX. Open Neural Network Exchange (ONNX); 2020.

[39] Andersson JAE, Gillis J, Horn G, Rawlings JB, Diehl M. CasADi: a software framework for nonlinear optimization and optimal control. Math Program Comput. 2019 Mar;11(1):1–36.

[40] Wächter A, Biegler LT. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Math Program. 2006 Mar;106(1):25–57.

[41] Kalibala A, Nada AA, Ishii H, El-Hussieny H. Dynamic modelling and predictive position/force control of a plant-inspired growing robot. Bioinspirat Biomimetics. 2025 Jan;20:016005.

[42] Marcot BG, Hanea AM. What is an optimal value of k in k-fold cross-validation in discrete Bayesian network analysis? Comput Stat. 2021 Sep;36(3):2009–31.