This article has been accepted for publication in IEEE Robotics and Automation Letters. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/LRA.2025.3645663

Physics-Informed Passive Motion Paradigm for Parallel Robots: A High-Precision Motor-Primitives Framework

Fuli Wang, Fazair Nizar Siraj, Windo Hutabarat, and Ashutosh Tiwari

***Abstract*—Complex embodied systems, whether biological or robotic, must continuously generate goal-directed behaviors while preserving coherence between motor intention and physical feasibility. In parallel robots, this link between intention and mechanics becomes particularly challenging due to their nonlinear, over-constrained kinematics and the absence of intuitive motor primitives. This letter introduces a passive motion paradigm for parallel robots using self-supervised physics-informed neural networks, which reformulates motion generation as the dynamic unfolding of motor primitives driven by attractor fields in actuator space. Unlike traditional forward or optimization-based formulations, the framework integrates analytical kinematics with neural fields to ensure both physical consistency and adaptive motion generation. The method estimates the Jacobian matrix as a physically constrained neural field, merging analytical structure with data-driven learning to achieve robust and interpretable behavior without relying on iterative numerical solvers. Theoretical analysis, simulations, and physical experiments demonstrate the framework's accuracy, stability, and adaptability across different parallel mechanisms.**

***Index Terms*—Passive motion paradigm, PINN, parallel robots.**

I. Introduction

In robotics, complex motor behavior can be understood as a sequence or combination of fundamental motor primitives. However, generating motor behaviors comparable to human capabilities remains a significant challenge [1]. Two prominent approaches have been developed to address this problem: Dynamic Movement Primitives (DMP) [2], and Elementary Dynamic Actions (EDA) [3]. These approaches have shown potential in various control tasks, particularly for humanoid robots, serial manipulators, and redundant robotic systems [4]–[6]. Expanding these frameworks to parallel robots is equally important, as parallel robots are widely used in manufacturing and assembly due to their high load capacity, stiffness, and excellent dynamic performance [7].

This letter builds upon the Passive Motion Paradigm (PMP) [8] to develop a control framework for parallel robots, such as Stewart platforms. From the EDA perspective, PMP leverages impedance control [9], kinematic networks [10], and the equilibrium point hypothesis [11] to guide robot motion through external forces and mechanical properties, steering the system toward desired equilibrium points or configurations. Unlike DMP, which emphasizes a formal mathematical framework for motion generation [12], EDA offers a modular control structure that also considers physical interaction with the environment [13]. This motivates applying PMP to admittance and impedance control in parallel robots, consistent with prior applications that concentrate on these areas [14], [15].

Although PMP has been applied in various contexts—such as whole-body reaching tasks [16], teaching humanoid robots to draw shapes [17], and enabling serial robots to harvest soft fruits [18]—it has not yet been implemented on parallel robots. Current control strategies for parallel robots primarily rely on combinations of numerical solutions, intelligent optimization algorithms, and sensor feedback [19]–[21]. However, numerical methods often become computationally intensive due to the highly nonlinear kinematic structures of parallel mechanisms. Even when combined with optimization algorithms that aim to refine approximate solutions, they may still yield configurations that are dynamically unfavorable or physically infeasible for real-world execution [22]. On the other hand, sensor-based control can estimate external forces and their positions, but mounting sensors may compromise the robot's stiffness and load-carrying capacity.

Additionally, learning-based approaches have emerged as an alternative for robot motion planning, especially Reinforcement Learning (RL), which has demonstrated impressive adaptability. However, most existing RL applications in parallel mechanisms are limited to cable-driven or low-stiffness structures, where compliant dynamics simplify policy learning and safety constraints [23]–[25]. Hybrid kinematic solvers and optimization-driven frameworks have also been developed to combine learning and analytical constraints [26], [27]. While such methods improve adaptability and accuracy, they remain computationally intensive and sensitive to initialization.

Hence, this letter addresses these challenges by adapting PMP to enable the control of parallel robots with closed-loop mechanisms. Instead of solving the highly nonlinear forward kinematics problem directly, this approach reformulates the cost function as a force field. The effectiveness of this paradigm depends heavily on accurately estimating the Jacobian matrix, which maps joint/actuator forces to end-effector/platform forces. However, the analytical calculation of

Manuscript received: October 30, 2025; Accepted December 7, 2025.

This paper was recommended for publication by Editor Clément Gosselin upon evaluation of the Associate Editor and Reviewers' comments. This work was supported in part by Innovate UK through the project under Grant 10002411, and in part by the Engineering and Physical Sciences Research Council (EPSRC) through the project under Grant EP/Y02270X/1.

All authors are with the School of Mechanical, Aerospace and Civil Engineering, The University of Sheffield, S1 4DT Sheffield, United Kingdom. E-Mail: [fuli.wang; f.nizarsiraj; w.hutabarat; a.tiwari] @sheffield.ac.uk

Digital Object Identifier (DOI): see top of this page.

Authorized licensed use limited to: BEIJING UNIVERSITY OF TECHNOLOGY. Downloaded on December 22,2025 at 15:18:29 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Robotics and Automation Letters. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/LRA.2025.3645663

the Jacobian matrix is prone to uncertainties and nonlinear perturbations. To mitigate this, the existing PMP framework typically employs Artificial Neural Networks (ANNs) to learn the relationship between joint angles and end-effector positions, using the chain rule to calculate the Jacobian matrix. While this approach simplifies implementation, its black-box nature introduces several limitations: the learned model lacks physical interpretability, is prone to overfitting, and often struggles to generalize under unseen poses or structural variations. As a result, ANN-based methods may produce kinematically inconsistent outputs or unstable Jacobians, ultimately constraining the robustness and reliability of the PMP framework.

To further address these limitations, this letter introduces a novel approach that integrates Physics-Informed Neural Networks (PINNs) [28] into the PMP framework using a self-supervised learning scheme. PINNs embed physical laws directly into the learning process by combining geometric constraints with neural network training, thereby improving the model's generalization, interpretability, and physical consistency.

Experimental results validate the effectiveness of the approach and demonstrate its potential as a reliable reference model for future robotic control strategies involving structured physical priors.

II. THEORETICAL FOUNDATIONS

In parallel robots, forward kinematics determines the platform pose $\mathbf{T}$, which includes both position $[x, y, z]$ and orientation $[\phi, \theta, \psi]$, based on the actuator or leg lengths $\mathbf{L}$=$[L_1, L_2, ..., L_n]$, where n is the number of legs. This problem is challenging due to the closed-chain nature of parallel robots, where multiple kinematic loops impose constraints on possible solutions. The PMP provides an alternative approach by simulating the robot's pose and actuator lengths through a dynamic system driven by specified forces. The core steps of this method are outlined as follows:

Step 1: Define a virtual spring–damper field between the robot's current and target actuator lengths:

$$\mathbf{F}_L = \mathbf{K}(\mathbf{L}_g - \mathbf{L}) - \mathbf{B}_L\dot{\mathbf{L}}, \tag{1}$$

where $\mathbf{K}$ and $\mathbf{B}_L$ are diagonal stiffness and damping matrices, respectively. This virtual impedance represents the interactive primitive that regulates the generalized displacement in actuator space and provides a compliant coupling toward the goal configuration.

Step 2: Map the force field from the actuator (leg) space to the platform space:

$$\mathbf{F_T} = \mathbf{J_{inv}^\top F_L} \tag{2}$$

Where, $\mathbf{J_{inv}^\top}$ is the transpose of the inverse Jacobian matrix, defined as:

$$\mathbf{J_{inv}} = \frac{\partial \mathbf{L}}{\partial \mathbf{T}} \tag{3}$$

Step 3: The platform pose evolves under an admittance law that translates task-space forces into pose-rate commands:

$$\dot{\mathbf{T}} = s(t)\,\mathbf{A}\mathbf{F_T} \tag{4}$$

where $\mathbf{A}$ is the admittance matrix and $s(t)$ is the normalized minimum-jerk scheduling function that ensures smooth temporal scaling. The damping component defined in Eq. (1) acts in the actuator space; in the implementation, its equivalent form $\mathbf{B_Q}\dot{\mathbf{T}}$ is applied directly in the pose space for numerical stability. This formulation preserves the physical effect of viscous damping while improving the convergence stability of the pose evolution.

Step 4: Explicit Euler Integration:

The platform pose is updated iteratively using an explicit Euler scheme:

$$\mathbf{T}_{k+1} = \mathbf{T}_k + \dot{\mathbf{T}}_k \Delta t. \tag{5}$$

The resulting discrete-time dynamic behaves as a first-order impedance element in task space

Step 5: Mapping Back to Actuator Space:

$$\dot{\mathbf{L}} = \mathbf{J}_{\text{inv}}\dot{\mathbf{T}}, \tag{6}$$

Integrating with the same Euler step, yields the evolving leg-length vector $\mathbf{L}_{k+1}$.

The temporal modulation $s(t)$ in Eq. (4) is defined by the minimum-jerk generator [29]:

$$s(t) = 10\left(\frac{t}{\tau}\right)^3 - 15\left(\frac{t}{\tau}\right)^4 + 6\left(\frac{t}{\tau}\right)^5, \quad 0 \le t \le \tau, \tag{7}$$

$$\dot{s}(t) = \frac{30}{\tau}\left(\frac{t}{\tau}\right)^2 - \frac{60}{\tau}\left(\frac{t}{\tau}\right)^3 + \frac{30}{\tau}\left(\frac{t}{\tau}\right)^4 \tag{8}$$

As shown in Fig. 1 (a), these five steps form a computational loop that steers the system toward the desired equilibrium configuration without the need to define an explicit cost function. More importantly, unlike numerical or direct-learning methods, the PMP-inspired formulation utilizes principles of physical compliance and closed-loop Jacobian feedback to intrinsically stabilize motion generation, even in highly redundant or nonlinear configurations.

Notably, the key component in this loop is the Jacobian inverse matrix. When the Jacobian is obtained from approximate models or numerical differentiation, small errors in its entries can be amplified by the repeated inversion required by PMP, leading to numerical instabilities and error accumulation. To address this, PMP typically employs a standard ANN to estimate the Jacobian. Specifically, to calculate the Jacobian matrix as defined in Eq. (3), a neural network with M layers is structured as follows:

- $\mathbf{T} \in \mathbb{R}^6$ representing platform poses.
- $\mathbf{L} \in \mathbb{R}^n$ representing actuator lengths.
- $W^{[i]}, b^{[i]}$ be the weights and biases for each layer i.
- $f^{[i]}$ be the activation function for each layer i.

The forward propagation at each layer i is expressed as:

$$h^{[i]} = W^{[i]}z^{[i-1]} + b^{[i]}, \quad z^{[i]} = f^{[i]}(h^{[i]}) \tag{9}$$

The final layer output is:

$$\mathbf{L} = W^{[M]}z^{[M-1]} + b^{[M]} \tag{10}$$

Authorized licensed use limited to: BEIJING UNIVERSITY OF TECHNOLOGY. Downloaded on December 22,2025 at 15:18:29 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Robotics and Automation Letters. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/LRA.2025.3645663

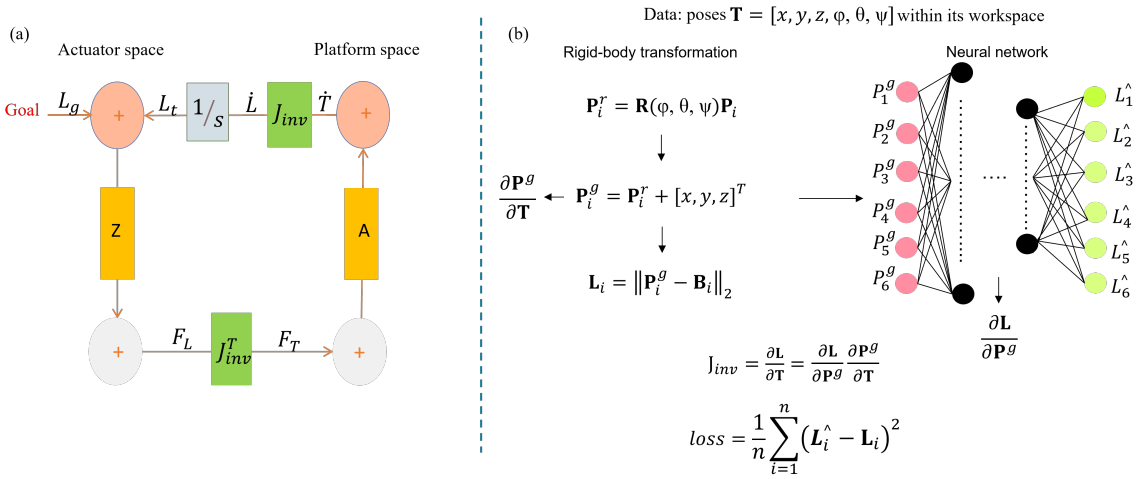

Fig. 1. (a) Basic kinematic network implements the passive motion paradigm. **Z**: Mechanical Impedance, including Stiffness and Damping; **A**: Admittance. (b) The framework of the self-supervised PINN uses platform poses as input data. The rigid-body transformation is embedded into the forward path, enabling real-time Jacobian computation via auto-differentiation for any arbitrary platform pose.

According to the Eq. (3), the Jacobian $\mathbf{J_{inv}} \in \mathbb{R}^{n\times 6}$ with respect to the input $\mathbf{x}$ is computed through the chain rule:

$$\mathbf{J_{inv}} = W^{[M]} \prod_{i=1}^{M-1} \text{diag}\left(f'^{[i]}(h^{[i]})\right) W^{[i]} \tag{11}$$

This method effectively maps input to output using activation functions and weights, with the chain rule accumulating the Jacobian during forward propagation. However, the accuracy of this data-driven approach depends heavily on the quality and quantity of training data. Poor convergence in the ANN may result in unstable Jacobian estimates, leading to large errors.

III. DESIGN OF THE SELF-SUPERVISED PINN

A. PINN-based Approximate Kinematic Transformations

In ANN-based Jacobian estimation, the input is the platform pose, and the output is the leg length, effectively modeling the inverse kinematics of parallel robots.

It is important to clarify that while the inverse kinematics of parallel robots is well-defined, using an ANN enables a differentiable input-output mapping through activation functions and weights, allowing automatic Jacobian computation. Therefore, to construct the PINN, it is essential to understand the inverse kinematics of parallel robots.

Consider a parallel manipulator with six adjustable legs. Given the platform pose $\mathbf{T}$=$[x, y, z, \phi, \theta, \psi]$ the task is to find the length of each leg. The transformed position of an attachment point on the moving platform in the global frame is given by:

$$\mathbf{P}_i^g = \mathbf{R}(\phi, \theta, \psi) \cdot \mathbf{P}_i + [x, y, z]^\top \tag{12}$$

where, $\mathbf{R}(\phi, \theta, \psi)$ is the rotation matrix. $\mathbf{P}_i \in \mathbb{R}^3$ is the position of the attachment point on the moving platform in the local frame. The transformation involves rotating the point by $\mathbf{R}$ to obtain $\mathbf{P}_i^r$, followed by translating it by $[x, y, z]^\top$, resulting in the global position $\mathbf{P}_i^g$.

The leg length L_i is computed as the Euclidean distance between the transformed point $\mathbf{P}_i^g$ and the corresponding fixed point $\mathbf{B}_i$ on the base platform:

$$\begin{aligned} L_i &= \|\mathbf{P}_i^g - \mathbf{B}_i\|_2 \\ &= \sqrt{(x_i - X_i)^2 + (y_i - Y_i)^2 + (z_i - Z_i)^2} \end{aligned} \tag{13}$$

where $\mathbf{B}_i \in \mathbb{R}^3$ is the structural position of the fixed point on the base platform. Once the platform's structural parameters are defined, both $\mathbf{B}_i$ and $\mathbf{P}_i$ are determinable.

To align with the inverse kinematics formulation of the parallel manipulator, the proposed PINN framework integrates a rigid-body transformation module and a neural network. As shown in Fig. 1 (b), given an input pose consisting of translation and Euler angles, the global coordinates of the platform attachment points $\mathbf{P}_i^g$ are computed explicitly via rotation and translation of fixed local points $\mathbf{P}_i$.

These transformed global points are then fed into a neural network, which predicts the corresponding leg lengths $\hat{\mathbf{L}} = [\hat{L_1}, \hat{L_2}, ..., \hat{L_n}]$.

In the proposed PINN framework, the inverse Jacobian matrix is computed using the chain rule:

$$\mathbf{J_{inv}} = \frac{\partial \mathbf{L}}{\partial \mathbf{P^g}} \frac{\partial \mathbf{P^g}}{\partial \mathbf{T}} \tag{14}$$

The term $\frac{\partial \mathbf{L}}{\partial \mathbf{P^g}}$ is obtained from the learned neural network, which handles the effect of global points on leg lengths.

Regarding $\frac{\partial \mathbf{P^g}}{\partial \mathbf{T}}$, the partial derivatives with respect to both translation and rotation parameters are required. Since the translation component $[x, y, z]^\top$ is added directly to the rotated point, its Jacobian is the identity matrix:

$$\frac{\partial \mathbf{P^g}}{\partial [x, y, z]} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \mathbf{I_3} \tag{15}$$

For the rotation component, the expression describing the effect of angle on the rotated point is as follows:

Authorized licensed use limited to: BEIJING UNIVERSITY OF TECHNOLOGY. Downloaded on December 22,2025 at 15:18:29 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Robotics and Automation Letters. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/LRA.2025.3645663

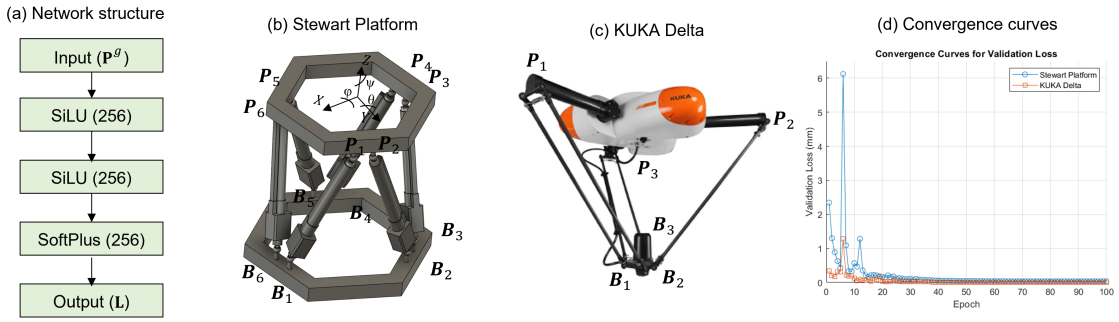

Fig. 2. (a) The details of the neural network structure of the PINN. (b) An illustration of a Stewart platform with diagonal cross-bracing. (c) An illustration of a KUKA Delta robot(KR-3-D1200). (d) The convergence curves of the PINN training process. Both convergences to approximately 0.01-0.03 mm.

$$\mathbf{J_r} = \frac{\partial \mathbf{P^r}}{\partial[\phi, \theta, \psi]} \tag{16}$$

Since the rotation transformation does not affect translation, the derivative of the translation term with respect to rotation is zero. Combining the partial derivatives of translation and rotation, the full Jacobian for $\mathbf{P}^g$ is:

$$\frac{\partial \mathbf{P^g}}{\partial \mathbf{T}} = \begin{bmatrix} \mathbf{I}_3 & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{J}_r \end{bmatrix} \tag{17}$$

The proposed PINN architecture does not rely on directly learning from raw platform poses, nor simply reduces the problem to training on global transformed points. Instead, it retains the pose parameters as the core inputs and explicitly embeds the rigid-body transformation process within the network's forward path, where the global coordinates of platform points become differentiable intermediate variables. This ensures end-to-end differentiability concerning the pose, enables accurate Jacobian estimation, eliminates the need for manual derivation or numerical approximation, and facilitates physically grounded and interpretable learning.

B. Loss Function

To incorporate physical constraints into the learning process in a self-supervised manner, we design a physics-informed loss function that enforces geometric consistency between the network's predictions and the robot's kinematic model:

$$\mathcal{L} = \frac{1}{n} \sum_{i=1}^{n} \left(\hat{L_i} - L_i \right)^2 \tag{18}$$

Given an input pose $\mathbf{T}$, the corresponding actuator lengths L_i are analytically computed using rigid-body geometric transformations derived from the manipulator's structural parameters, as described in Eqs. (13)–(14). Simultaneously, the neural network takes $\mathbf{P}_i^g$ as input and directly predicts the actuator lengths $\hat{L_i}$.

This purely physics-based loss guides the model to learn a mapping that is not only numerically consistent with the robot's inverse kinematics but also grounded in the physical configuration of the parallel mechanism. By eliminating dependence on precomputed labeled data, the model gains robustness, adaptability to parameter changes, and improved generalization in unseen workspaces.

IV. SIMULATION AND ANALYSIS

To verify the proposed framework, the first experimental setup is illustrated in Fig. 2, which includes the structure of the neural network within PINN, as well as schematic diagrams of a Stewart platform and a KUKA Delta robot.

The Stewart platform features a hexagonal base and a moving platform ($450\,\text{mm}$ and $350\,\text{mm}$ in diameter, respectively) with diagonally cross-braced legs. Its translational workspace covers approximately $\pm 60\,\text{mm}$ in X and Y and $406 - 506\,\text{mm}$ in Z, with angular ranges of $\pm 15^\circ$ in roll, pitch, and yaw.

For the KUKA KR 3 D1200 Delta robot, although the Delta architecture allows only translational motion, its forward kinematics still forms a nonlinear system of coupled quadratic equations, often leading to multiple solutions and singular configurations. Consequently, achieving a stable Jacobian estimation and maintaining high control accuracy remains challenging. By employing the proposed PINN-PMP framework, these issues can be mitigated. The training procedure is identical to that used for the Stewart platform, except that the rotational components are omitted, the network input and output dimensions are reduced from six to three, corresponding to the Cartesian coordinates and the three leg lengths, while the rest of the network structure remains unchanged.

A. PINN training

The self-supervised PINN was implemented in PyTorch with three hidden layers of 256 neurons each, using the sigmoid-weighted linear unit (SiLU) activation to ensure smooth gradient flow and stable convergence [30]. The output layer employed the SoftPlus activation, a smooth variant of ReLU that guarantees positive leg-length predictions and maintains differentiability for stable Jacobian computations. Optimization used the Adam algorithm with an initial learning rate of 1×10^{-3} (cosine decay to 1×10^{-5}), a batch size of 256, and L2 weight decay of 1×10^{-5}.

A total of 500,000 random platform poses were generated within the workspace limits, with 80% used for training and

Authorized licensed use limited to: BEIJING UNIVERSITY OF TECHNOLOGY. Downloaded on December 22,2025 at 15:18:29 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Robotics and Automation Letters. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/LRA.2025.3645663

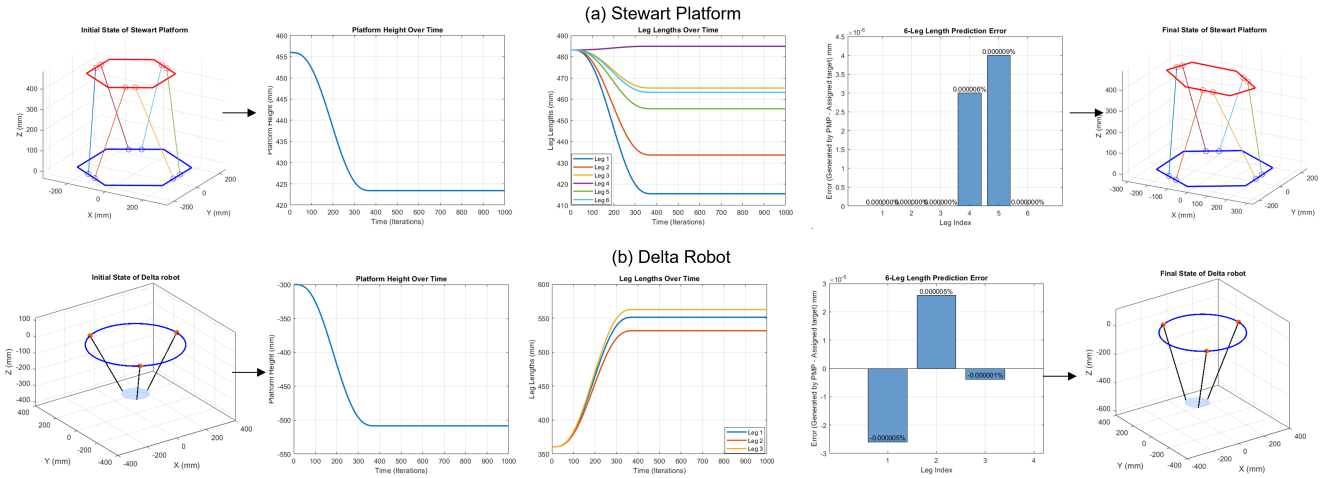

Fig. 3. The motion generation results using the proposed PMP framework. The trajectory plots illustrate the changes in platform height and leg lengths over time, while the bar chart summarizes the prediction errors for the legs (relative errors are expressed as percentages of the assigned target lengths).

20% for validation. The training objective combined a reconstruction loss with a physics-informed regularization term that aligns the auto-differentiated Jacobian with the analytical one. Training was performed on Google Colab using a single NVIDIA T4 GPU for approximately 100 epochs, requiring around 10 minutes per model. During each forward pass, the global coordinates of the platform points were computed through a rigid-body transformation based on the input pose and pre-defined geometric buffers.

The convergence curves for the training process are shown in Fig. 2 (d). The PINN demonstrated smooth and stable convergence, owing to the embedded physical priors that regularize the Jacobian field. Note that the model can be further optimized by using transfer learning and tuning the parameters layer by layer [31]. After training, the learned models were used to implement the PMP approach for each platform.

B. Results of the PINN-based PMP

The PMP computation loop functions as a convergence process, guiding the robot from its initial state to a designated target state. This process enables the full pose evolution before actual execution, aligning with concepts in motor neuroscience, where both real and imagined actions arise from an internal simulation process driven by passive goal-oriented animation of the body schema [32].

To evaluate the feasibility of the proposed framework, random target states were assigned to each of the two platforms, and the corresponding movements were generated. To visualize the results, all generated movements and dynamic variations were recorded and then plotted using MATLAB R2023a, as shown in Fig. 3. This figure illustrates the transition of platform height and leg lengths from the initial to the target state.

The evaluation was conducted on both the Stewart and Delta platforms, each tested over twenty configurations sampled within their respective feasible workspaces. For every configuration, three complementary consistency metrics were computed: the Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and Maximum Absolute Deviation (MAD), all defined as the discrepancy between the target leg lengths and those obtained from the inverse-kinematics reconstruction of the final converged pose. In all twenty test cases, the geometric setpoint consistency errors for both robots were below the numerical precision of 10^{-4} mm for the RMSE and MAE metrics, while the MAD occasionally reached 1×10^{-4} mm.

These results confirm the numerical stability and precision of the proposed control algorithm. It is also worth noting that the achieved performance depends on the controller's parameterization, particularly the stiffness matrix. In accordance with Eq. (1), the stiffness matrix $\mathbf{K}$ was set to $100\,\mathbf{I}_n$ in this experiment, representing an isotropic virtual spring with equal stiffness along all actuator directions. This parameter determines the coupling strength between the current and target actuator lengths. It has a direct influence on the convergence rate and steady-state precision, as discussed in the following section.

This modular design improves interpretability because each pipeline component links to a physically meaningful process. Since the rigid-body transformation is based on known geometric inputs rather than learned weights, the framework can be applied to different robot configurations. This study confirmed the framework's effectiveness on both the Stewart platform and the Delta robot, showing it can adapt to mechanisms with varying kinematic structures. Future work will extend this validation to irregular or large-scale geometries and to robots with different leg configurations or actuation redundancies, aiming to systematically examine how these structural differences impact the prediction accuracy and stability of the PINN-PMP framework.

V. EXPERIMENTAL RESULTS

To validate the effectiveness and practical applicability of the proposed PMP-based kinematics approach, we constructed a prototype of a 6-Spherical-Prismatic-Spherical (6-SPS) parallel manipulator, as shown in Fig. 4. The manipulator provides

Authorized licensed use limited to: BEIJING UNIVERSITY OF TECHNOLOGY. Downloaded on December 22,2025 at 15:18:29 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Robotics and Automation Letters. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/LRA.2025.3645663

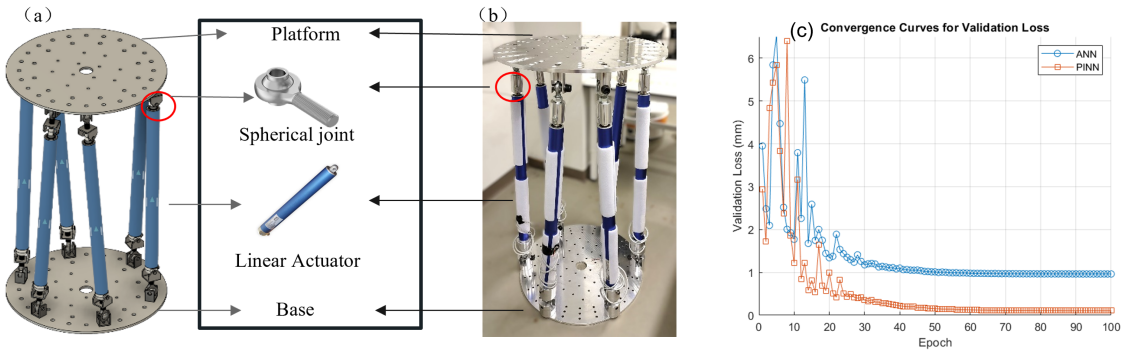

Fig. 4. Custom-built setup. (a) 6-SPS parallel manipulator CAD. (b) Prototype of the 6-SPS parallel manipulator. (c) Training convergence curves for the two models on the prototype. The standard ANN converges to 0.96 mm, the PINN reaches about 0.1 mm.

a translational workspace of approximately $\pm 100\,\mathrm{mm}$ along the X and Y axes and from $1000\,\mathrm{mm}$ to $1450\,\mathrm{mm}$ along the Z-axis, with rotational capabilities of $\pm 30^\circ$ in roll, pitch, and yaw. The top platform and base have diameters of $550\,\mathrm{mm}$ and $600\,\mathrm{mm}$, respectively.

To evaluate the performance of different learning frameworks, first, two models were trained on this new configuration. (1) A standard artificial neural network trained from scratch. The platform pose was directly used as the input, and the corresponding leg lengths, computed via inverse kinematics, served as the training labels. (2) The proposed physics-informed model was trained from scratch.

The goal was to assess the convergence behavior and final accuracy of each method, with emphasis on the reusability and adaptability of the proposed PINN framework. The convergence curves of the two models are plotted in Fig 4 (c).

A. Steady-state accuracy and comparison

To assess the influence of physical priors on learning-based kinematics, we implemented an ANN-based PMP and a PINN-PMP. Both frameworks share an identical control and trajectory generation structure, differing only in how the Jacobian is computed.

For all experiments, the reference trajectory was generated using a normalized minimum-jerk profile over a duration of $T = 1.5\,\mathrm{s}$, and the controller operated with a time step of $\Delta t = 0.004\,\mathrm{s}$. To investigate the effect of mechanical compliance, the diagonal stiffness matrix $\mathbf{K} = \mathrm{diag}(K_i)$ was varied across eight levels. In contrast, the actuator-side damping matrix $\mathbf{B}_L$ was set to zero. Only a pose-side diagonal damping term $\mathbf{B}_Q$ was applied in the Cartesian domain to ensure numerical stability and smooth convergence. The admittance scaling factor $s(t)$ followed the same temporal law for both frameworks, and all other control gains were kept identical.

Although the final setpoint accuracy does not explicitly distinguish between the ANN- and PINN-based PMP frameworks, it remains a crucial indicator of the overall precision and consistency of the proposed method. As shown in Table I, both frameworks achieved sub-micrometer precision across most stiffness levels, with the final leg-length error on the order of $10^{-4}\,\mathrm{mm}$. This remarkable accuracy confirms that the PMP can achieve numerically stable convergence to the desired pose without relying on static analytical mappings or iterative numerical inversion.

While rigorous numerical methods can incorporate additional optimization techniques to obtain an exact solution in theory, they typically do not encode motor-level compliance or dynamic consistency. In contrast, the PMP approach interprets motion generation as the evolution of a dynamical system shaped by virtual springs, damping, and admittance modulation. This formulation provides a physically meaningful and temporally smooth convergence process, effectively bridging the gap between purely geometric kinematics and motor control.

Note that a slight reduction in final setpoint accuracy occurs at the highest stiffness, which can be attributed to the underdamped behavior due to the lack of actuator-side damping and the finite integration horizon. This sensitivity at high stiffness levels aligns with the discrete-time implementation of the admittance loop, where increased virtual stiffness amplifies numerical errors and small model mismatches. Nevertheless, this issue can be alleviated by adjusting damping or admittance, which will be investigated in future work to create different motion profiles suited for specific tasks.

B. Transient and tracking performance comparison

The tracking metrics report the command-following residual. This residual captures both the imposed reference offset (for impedance-driven correction) and the closed-loop response error; it is therefore larger during the transient and contracts to the steady-state accuracy after settling.

As illustrated in Fig. 5 and Table I, both frameworks exhibit smooth leg-space trajectories that closely follow the reference lengths; however, the PINN-PMP consistently achieves lower command-following residuals and faster convergence across a wide range of stiffness values. By embedding physical priors into the Jacobian learning process, the PINN yields a better-conditioned mapping between actuator and platform spaces, resulting in improved transient stability and smaller $\Delta\mathrm{RMSE}$ values. This improvement is also reflected in shorter settling times and smoother recovery after transient peaks, demonstrating the effectiveness of the physics-informed regularization in suppressing oscillations during motion execution.

Authorized licensed use limited to: BEIJING UNIVERSITY OF TECHNOLOGY. Downloaded on December 22,2025 at 15:18:29 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Robotics and Automation Letters. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/LRA.2025.3645663

TABLE I
COMPARISON OF SETPOINT CONSISTENCY AND TRACKING METRICS BETWEEN TWO LEARNING FRAMEWORKS WITH DIFFERENT STIFFNESS [UNIT: mm, 4-DECIMAL PRECISION]

K	ANN-PMP				PINN-PMP			
	MAE (Final)	RMSE (Final)	MAE (Tracking)	RMSE (Tracking)	MAE (Final)	RMSE (Final)	MAE (Tracking)	RMSE (Tracking)
$5\,\mathbf{I}_6$	0.0032	0.0032	11.2399	16.5196	0.0031	0.0031	11.2140	16.5029
$20\,\mathbf{I}_6$	0.0007	0.0007	2.8293	4.4609	0.0006	0.0006	2.8206	4.4536
$40\,\mathbf{I}_6$	0.0003	0.0003	1.4147	2.2423	0.0001	0.0001	1.4103	2.2384
$100\,\mathbf{I}_6$	0.0001	0.0001	0.5659	0.8982	0	0	0.5641	0.8966
$200\,\mathbf{I}_6$	0.0001	0.0001	0.2829	0.4492	0.0001	0.0001	0.2820	0.4484
$300\,\mathbf{I}_6$	0.0001	0.0001	0.1886	0.2995	0.0001	0.0001	0.1880	0.2989
$400\,\mathbf{I}_6$	0.0001	0.0001	0.1415	0.2246	0.0001	0.0001	0.1410	0.2242
$500\,\mathbf{I}_6$	0.0025	0.0032	0.1259	0.1870	0.0009	0.0009	0.1134	0.1807

To further evaluate the generalization capability of both learning frameworks, tracking experiments were conducted on multiple target poses under a fixed stiffness setting. Each target was first defined in the Cartesian space as a desired platform pose, then converted into the corresponding leg lengths through the analytical geometry of the parallel mechanism. During testing, these target leg lengths served as goal states for the controller, which progressively generated the platform motion through its motor primitives governed by the learned Jacobian mapping.

The first two targets were selected within the training workspace, while the remaining five corresponded to out-of-distribution poses whose Cartesian displacements or orientations exceeded the training range by up to $\pm 25\,\mathrm{mm}$ in x/y and $\pm 5^\circ$ in roll, pitch, or yaw. As summarized in Table II, the PINN-PMP consistently maintained comparable or lower RMSE and MAE values across both in-distribution and out-of-distribution targets, confirming its ability to generalize beyond the sampled training data. In contrast, the ANN-PMP became numerically unstable (NaN values in the leg-space control loop due to Jacobian ill-conditioning) for the most extreme target (Target 7: $[-111.5,\ 124.6,\ 1085.2,\ 10.3^\circ,\ 5.1^\circ,\ 10.2^\circ]$), where the extrapolation error in the learned Jacobian led to divergence in the leg-space control loop. This experiment demonstrates that embedding physical priors within the PINN formulation significantly enhances stability and adaptability when extrapolating to unseen or out-of-range configurations.

TABLE II
COMPARISON OF TRACKING METRICS BETWEEN TWO LEARNING FRAMEWORKS WITH DIFFERENT TARGETS [UNIT: mm, K=$300\,\mathbf{I}_6$]

Target	ANN-PMP		PINN-PMP	
	MAE	RMSE	MAE	RMSE
1	0.1303	0.2234	0.1302	0.2224
2	0.1040	0.1780	0.1038	0.1780
3	0.2641	0.3795	0.2619	0.3795
4	0.1575	0.2186	0.1555	0.2169
5	0.1865	0.2777	0.1850	0.2764
6	0.1831	0.2695	0.1818	0.2684
7	Unstable	Unstable	0.3574	0.4189

C. Physical demonstration

Finally, a physical demonstration of the proposed PINN-PMP framework is conducted on the custom-built 6-SPS parallel manipulator (Fig. 6), commanding the platform to move from an initial configuration to two desired poses. This experiment validates the deployability of the framework for executing precomputed, safety-aware motion plans on hardware. It also highlights a key practical benefit of using a PINN-based Jacobian: in real deployments, actuator noise, assembly tolerances, backlash, and compliance can make analytical kinematic models inaccurate or difficult to maintain. Trained on data collected from the physical prototype, PINN-PMP learns a differentiable, structure-consistent mapping and delivers transparent, compliant behaviours that remain effective under such mechanical uncertainties.

VI. CONCLUSION

This letter proposes a physics-informed neural network-based Passive Motion Paradigm tailored for parallel robots, backed by both theoretical formulation and experimental validation. The core contribution lies in the development of a self-supervised, PINN-based PMP framework for generating physically consistent motor behaviors in parallel robots. The proposed approach offers a potential alternative to conventional numerical solvers, especially in scenarios requiring modularity, physical plausibility, and interaction-aware motion generation.

Future work will compare PINN-PMP with state-of-the-art learning-based controllers under matched sensing and safety constraints to highlight its advantages in transparency and physical consistency. We will also explore scalability across various robot geometries and higher-DoF mechanisms by using structure-aware architectures such as graph neural networks or attention-based models. Finally, we will expand real-world validation in parallel-mechanism assembly and contact-rich manipulation, including robustness evaluation under controlled uncertainty injections.

REFERENCES

[1] A. Billard and D. Kragic, "Trends and challenges in robot manipulation," *Science*, vol. 364, no. 6446, p. eaat8414, 2019.
[2] M. Saveriano, F. J. Abu-Dakka, A. Kramberger, and L. Peternel, "Dynamic movement primitives in robotics: A tutorial survey," *The International Journal of Robotics Research*, vol. 42, no. 13, pp. 1133–1184, 2023.
[3] M. C. Nah, J. Lachner, and N. Hogan, "Robot control based on motor primitives: A comparison of two approaches," *The International Journal of Robotics Research*, vol. 43, no. 12, pp. 1959–1991, 2024.
[4] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: Learning attractor models for motor behaviors," *Neural Computation*, vol. 25, no. 2, pp. 328–373, 02 2013.
[5] N. Hogan, *Physical Interaction via Dynamic Primitives*. Springer International Publishing, 2017, pp. 269–299.

Authorized licensed use limited to: BEIJING UNIVERSITY OF TECHNOLOGY. Downloaded on December 22,2025 at 15:18:29 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Robotics and Automation Letters. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/LRA.2025.3645663

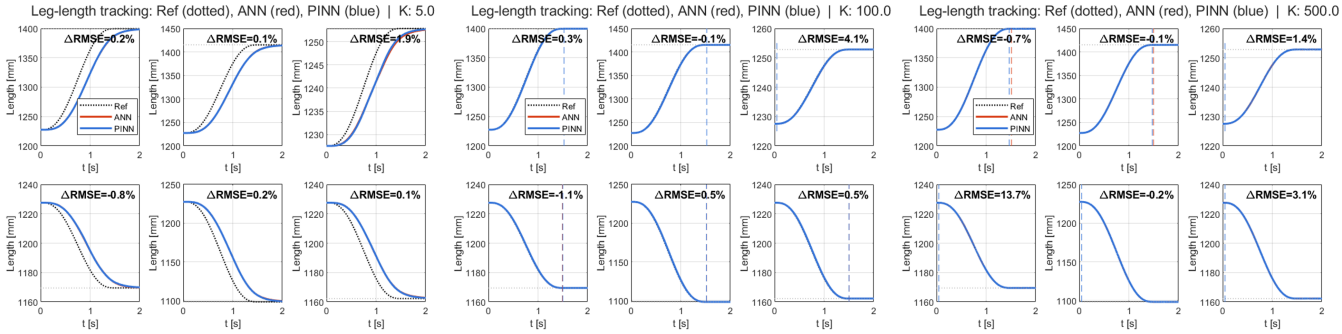

Fig. 5. Comparison of leg-space position tracking performance between ANN-PMP and PINN-PMP under different stiffness values K. Black dotted lines represent the reference leg lengths, red solid lines correspond to ANN-PMP, and blue solid lines correspond to PINN-PMP. The gray dotted horizontals denote the $\pm 0.02\ mm$ settling band around the final reference, and the dashed vertical lines indicate the settling times for ANN (red) and PINN (blue). The label in each panel reports $\Delta\mathrm{RMSE}$, where positive values indicate that PINN-PMP achieves lower RMSE.

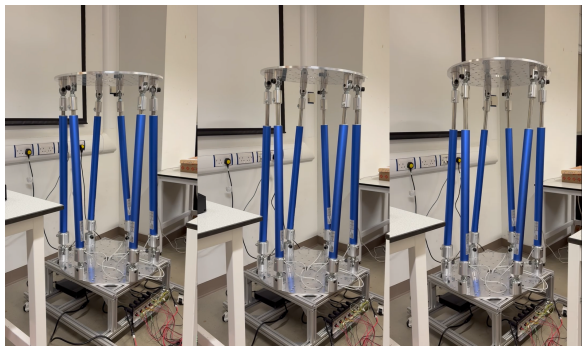

Fig. 6. Physical demonstration of the proposed PINN-PMP framework on the 6-SPS parallel manipulator.

[6] M. C. Nah, J. Lachner, F. Tessari, and N. Hogan, "On the modularity of elementary dynamic actions," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024, pp. 1398–1405.

[7] F. Du, K. Wen, and H. Yu, "A self-adaptive alignment strategy for large components based on dynamic compliance center," *Assembly automation*, vol. 39, no. 2, pp. 345–355, 2019.

[8] V. Mohan and P. Morasso, "Passive motion paradigm: an alternative to optimal control," *Frontiers in Neurorobotics*, vol. 5, p. 4, 2011.

[9] N. Hogan, "Impedance control: An approach to manipulation: Part ii—implementation," *Journal of Dynamic Systems, Measurement, and Control*, vol. 107, no. 1, pp. 8–16, 03 1985.

[10] F. M. Ivaldi, P. Morasso, and R. Zaccaria, "Kinematic networks: A distributed model for representing and regularizing motor redundancy," *Biological cybernetics*, vol. 60, pp. 1–16, 1988.

[11] E. Bizzi, N. Hogan, F. A. Mussa-Ivaldi, and S. Giszter, "Does the nervous system use equilibrium-point control to guide single and multiple joint movements?" *Behavioral and Brain Sciences*, vol. 15, no. 4, p. 603–613, 1992.

[12] P. Morasso, "Spatial control of arm movements," *Experimental brain research*, vol. 42, no. 2, pp. 223–227, 1981.

[13] N. Hogan, "Contact and physical interaction," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 5, no. Volume 5, 2022, pp. 179–203, 2022.

[14] Z. Chen, G. Zhan, Z. Jiang, W. Zhang, Z. Rao, H. Wang, and J. Li, "Adaptive impedance control for docking robot via stewart parallel mechanism," *ISA Transactions*, vol. 155, pp. 361–372, 2024.

[15] T. Sun, J. Sun, B. Lian, and Q. Li, "Sensorless admittance control of 6-dof parallel robot in human-robot collaborative assembly," *Robotics and Computer-Integrated Manufacturing*, vol. 88, p. 102742, 2024.

[16] P. Morasso, M. Casadio, V. Mohan, and J. Zenzeri, "A neural mechanism of synergy formation for whole body reaching," *Biological Cybernetics*, vol. 102, pp. 45–55, 2010.

[17] V. Mohan, P. Morasso, J. Zenzeri, G. Metta, V. S. Chakravarthy, and G. Sandini, "Teaching a humanoid robot to draw 'shapes'," *Autonomous Robots*, vol. 31, pp. 21–53, 2011.

[18] F. Wang, R. C. Urquizo, P. Roberts, V. Mohan, C. Newenham, A. Ivanov, and R. Dowling, "Biologically inspired robotic perception-action for soft fruit harvesting in vertical growing environments," *Precision Agriculture*, vol. 24, no. 3, pp. 1072–1096, 2023.

[19] Q. Zhu and Z. Zhang, "An efficient numerical method for forward kinematics of parallel robots," *IEEE Access*, vol. 7, pp. 128 758–128 766, 2019.

[20] D. Gan, Q. Liao, J. S. Dai, S. Wei, and L. Seneviratne, "Forward displacement analysis of the general 6–6 stewart mechanism using gröbner bases," *Mechanism and Machine Theory*, vol. 44, no. 9, pp. 1640–1647, 2009.

[21] J. Xia, Z. Lin, X. Ai, G. Yu, and A. Gao, "External interaction estimation of 6-pss parallel robots with embodied mechanical intelligence," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024, pp. 1376–1381.

[22] H. Zhu, W. Xu, B. Yu, F. Ding, L. Cheng, and J. Huang, "A novel hybrid algorithm for the forward kinematics problem of 6 dof based on neural networks," *Sensors*, vol. 22, no. 14, 2022.

[23] H. Xiong, T. Ma, L. Zhang, and X. Diao, "Comparison of end-to-end and hybrid deep reinforcement learning strategies for controlling cable-driven parallel robots," *Neurocomputing*, vol. 377, pp. 73–84, 2020.

[24] Y. Lu, C. Wu, W. Yao, G. Sun, J. Liu, and L. Wu, "Deep reinforcement learning control of fully-constrained cable-driven parallel robots," *IEEE Transactions on Industrial Electronics*, vol. 70, no. 7, pp. 7194–7204, 2022.

[25] C. Sancak, F. Yamac, and M. Itik, "Position control of a planar cable-driven parallel robot using reinforcement learning," *Robotica*, vol. 40, no. 10, p. 3378–3395, 2022.

[26] I. Kardan and A. Akbarzadeh, "An improved hybrid method for forward kinematics analysis of parallel robots," *Advanced Robotics*, vol. 29, no. 6, pp. 401–411, 2015.

[27] R. S. Zarrin, R. Jitosho, and K. Yamane, "Hybrid learning- and model-based planning and control of in-hand manipulation," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023, pp. 8720–8726.

[28] M. Raissi, P. Perdikaris, and G. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *Journal of Computational Physics*, vol. 378, pp. 686–707, 2019.

[29] N. Hogan and T. Flash, "Moving gracefully: quantitative theories of motor coordination," *Trends in neurosciences*, vol. 10, no. 4, pp. 170–174, 1987.

[30] S. Elfwing, E. Uchibe, and K. Doya, "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning," *Neural Networks*, vol. 107, pp. 3–11, 2018, special issue on deep reinforcement learning.

[31] F. Wang, V. Mohan, and A. Tiwari, "Passive motion paradigm implementation via deep neural networks: analysis and verification," *Robotica*, vol. 43, no. 5, p. 1766–1784, 2025.

[32] V. Mohan, A. Bhat, and P. Morasso, "Muscleless motor synergies and actions without movements: From motor neuroscience to cognitive robotics," *Physics of Life Reviews*, vol. 30, pp. 89–111, 2019.

Authorized licensed use limited to: BEIJING UNIVERSITY OF TECHNOLOGY. Downloaded on December 22,2025 at 15:18:29 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.