The First International Conference on
Machine Learning and Knowledge Discovery (MLKD 2024)
Amirkabir University of Technology, December 18-19, 2024

PhIGT: Physics Informed Graph Transformer for Planar Cable-Driven Parallel Robot Forward Kinematics

Javad Koramdel* Ahmad Moori†

Abstract

Neural networks (NNs) have made significant progress in recent years, achieving impressive results in a wide range of applications. However, many of these models do not take advantage of the rich physical knowledge available from prior studies and domain expertise, especially in the field of robotics. To bridge this gap, Physics-Informed Neural Networks (PINNs) have emerged as a powerful framework, enabling the integration of physics information with various types of neural networks. This approach has gained popularity due to its ability to leverage physical laws, improve generalization, and enhance learning efficiency. In this paper, we propose a Physics-Informed Graph Transformer Network (PhIGT) for implementation on a cable-driven parallel robot to solve the forward kinematics problem. By incorporating physical constraints into the graph-based transformer architecture, our approach aims to achieve more accurate and physically consistent predictions for the cable robot's end-effector positions, offering a novel solution that effectively combines data-driven learning with domain-specific physics.

Keywords: Physics Informed Neural Network, Graph Neural Network, Transformer, Forward Kinematics, Planar Cable-Driven Parallel Robot

1 Introduction

Parallel robots, particularly cable-driven parallel robots (CDPRs), are well known for their straightforward inverse kinematics solutions [9]. However, solving the forward kinematics for these systems is often significantly more challenging, as analytical solutions rarely exist. This difficulty arises from the nonlinearities and complexities inherent in the robot's structure. As a result, obtaining accurate measurements of the end-effector's position and orientation within the workspace typically requires the use of expensive equipment, such as motion capture systems, which may not be feasible in many practical applications due to their high cost.

To address this challenge, we propose a novel approach: a physics-informed graph transformer network for solving the forward kinematics of a planar CDPRs. In this approach, the end-effector and the fixed anchors of the robot are represented as graph nodes, while the cables connecting them are treated as graph edges. The model leverages the measured joint variables (i.e., cable lengths) and incorporates physical constraints derived from the robot's structure to estimate the workspace variables, including the end-effector's position.

By embedding the robot's kinematic model into the graph structure, the proposed method effectively captures the physical relationships between the cables and the end-effector, enabling accurate estimation of the end-effector's position. This data-driven, physics-informed framework provides a cost-effective alternative to traditional sensor-based approaches, allowing for reliable forward kinematics estimation without the need for expensive external sensors. Although this study focuses on a specific planar CDPR, the proposed solution is easily generalizable to other types of robots.

The remainder of the paper is organized as follows: Section 2 reviews related works; Section 3 elaborates on the robot's structure and both the inverse and forward kinematics of the planar CDPR; Section 4 describes the network architecture and components; Section 5 explains the experiment details; and Section 6 discusses the results.

2 Related Works

Many works have focused on using deep learning algorithms to improve various aspects of parallel robots[21, 7, 22]. Cable-driven parallel robots (CDPRs), a type of parallel robot, are known for their high payload capacity and flexibility[2], making them suitable for applications such as camera motion control, rehabilitation, and material handling. Deep learning techniques have been used to enhance modeling, control, and trajectory planning for CDPRs[19, 8, 1], showing promising improvements over traditional methods.

Recently, Graph Neural Networks (GNNs)[5] have gained attention for their ability to model relational data, which is particularly useful for robotic systems like CDPRs where capturing the relationships between different components is crucial. GNNs have been applied in robotic control and dynamics prediction, showing promising results in handling complex system dependencies[11, 13]. Additionally, Transformers[16]

*Independent Scholar, j.khorramdel96@gmail.com
†Independent Scholar, ahmadmoori76@gmail.com

have also been adapted to robotics to process complex data effectively[6, 14]. Combining GNNs with transformers[17] has shown potential in representing complex systems more effectively, offering new opportunities for improving the control and efficiency of CDPRs[3].

Physics-informed learning has gained attention for its ability to integrate physical constraints and prior system knowledge directly into machine learning models[12], which helps achieve better robustness and generalization. This approach has shown notable success in applications involving dynamic systems and control[4, 18]. More recently, there have been efforts to combine physics-informed learning with graph transformers[20, 10], using physical laws in conjunction with graph-based relational modeling to improve model performance across various domains.

3 Physical Modeling

In the subsequent sections, the structure of the robot will be explained, and the equations describing the inverse and forward kinematics of the robot will be derived.

3.1 Robot Structure

The schematic of the robot is shown in Figure 1. The robot consists of a rectangular frame with a height h and width w. The end-effector, depicted at the center of the frame, is a circular object with a radius R_B. The global coordinate system is centered in the middle of the frame, with its principal axes aligned parallel to the edges of the frame.

The end-effector is connected to the frame via cables, with each cable attached to a fixed anchor point on the frame, denoted as A_i. The length of each cable can be controlled independently, allowing the end-effector to move within the workspace. The angle between each cable and the edges of the frame is denoted by α_i. The other end of each cable, denoted as L_i, is connected to the end-effector at attachment points B_i, positioned at an angle θ_i relative to the center of the end-effector.

3.2 Inverse Kinematics

The inverse kinematics for the studied cable-driven parallel robot aims to determine the set of cable lengths $L = [L_1, L_2, L_3, L_4]^T$ and their angles $\alpha = [\alpha_1, \alpha_2, \alpha_3, \alpha_4]^T$, given a desired configuration of the end-effector in the workspace $X = [x_{cg}, y_{cg}, \phi_{cg}]^T$. Here, x_{cg} and y_{cg} represent the position of the center of the end-effector in Cartesian coordinates, and ϕ_{cg} denotes its orientation in the global frame. For a better intuition, the end-effector is subjected to a virtual displacement in the workspace, and depicted in 2.

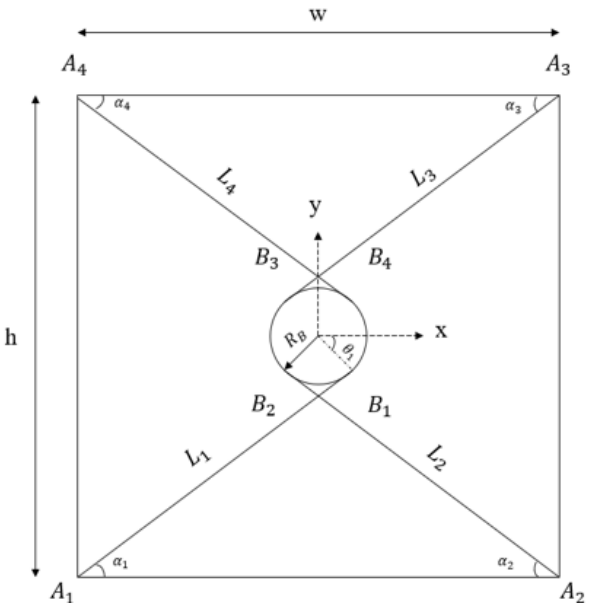

Figure 1: Schematic of the cable-driven parallel robot.

For each cable i, the corresponding length L_i is calculated based on the positional relationship between the anchor point A_i on the fixed frame and the attachment point B_i on the end-effector. The position of B_i is influenced by both the end-effector's global position and orientation.

The intermediate components of the ith cable length can be described by the following relations:

$$L_{x_i} = (x_g - A_{ix}) + R_B \cos(\phi_g - \theta_i), \tag{1}$$

$$L_{y_i} = (y_g - A_{iy}) + R_B \sin(\phi_g - \theta_i), \tag{2}$$

where (x_g, y_g) represents the global position of the center of the end-effector, R_B is the distance between the center of the end-effector and the attachment point B_i, and θ_i is the angular position of the attachment point B_i relative to the center of the end-effector. The constants A_{ix} and A_{iy} correspond to the fixed coordinates of the ith cable's anchor point in the base frame.

Once the intermediate components L_{x_i} and L_{y_i} are determined, the length of the ith cable, L_i, is computed as:

$$L_i = \sqrt{L_{x_i}^2 + L_{y_i}^2}. \tag{3}$$

This expression yields the magnitude of the ith cable length, which ensures the attachment point B_i is positioned according to the desired end-effector configuration in the workspace.

Additionally, the angle α_i of the cable relative to the horizontal axis can be derived using the two-dimensional inverse tangent function:

$$\alpha_i = \text{atan2}(L_{y_i}, L_{x_i}), \tag{4}$$

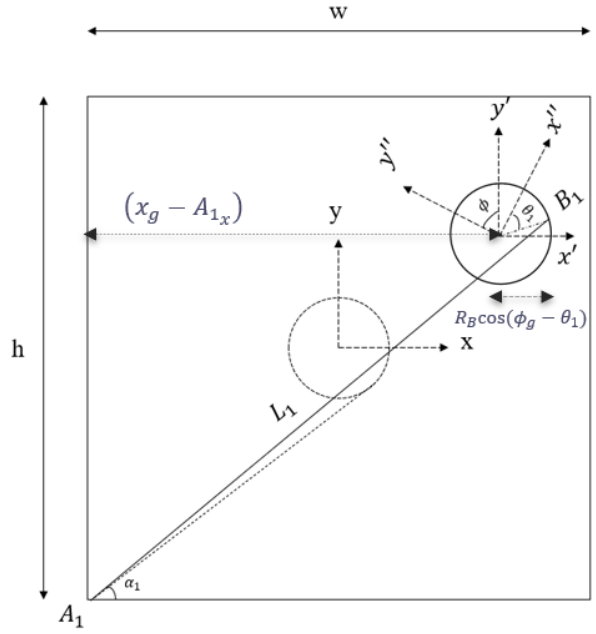

Figure 2: The illustration of virtually displaced end-effector

where α_i provides information regarding the direction of the cable relative to the fixed base frame.

By applying these equations for all four cables, the full set of cable lengths $L = [L_1, L_2, L_3, L_4]^T$ can be determined, allowing the system to achieve the desired pose of the end-effector, represented by $X = [x_{cg}, y_{cg}, \phi_{cg}]^T$. The inverse kinematics formulation presented here ensures that the physical constraints of the system are respected, as each cable length is uniquely defined based on the end-effector's position and orientation. This is noteworthy to mention although the equations for cable angles are derived, they are not easy to measure. Hence, only the cable lengths would be considered for further calculations.

3.3 Forward Kinematics

The forward kinematics of the cable-driven parallel robot involves determining the configuration of the end-effector, denoted as $X = [x_{cg}, y_{cg}, \phi_{cg}]^T$, from the given cable lengths $L = [L_1, L_2, L_3, L_4]^T$. Unlike inverse kinematics, the forward kinematics problem does not generally admit closed-form analytical solutions due to its nonlinear nature. However, a reformulation of the forward kinematics equations is suggested in [15], which leverages the physical relationship between the cable lengths, the end-effector position, and its orientation.

The squared length of each cable L_i^2 can be expressed in terms of the coordinates of the end-effector's center of gravity (x_{cg}, y_{cg}) and its orientation ϕ_{cg} as follows:

$$\begin{aligned} L_i^2 = &\left((x_{cg} - A_{ix}) + R_B\cos(\phi_{cg} - \theta_i)\right)^2 \\ &+ \left((y_{cg} - A_{iy}) + R_B\sin(\phi_{cg} - \theta_i)\right)^2 \end{aligned} \tag{5}$$

By expanding and rearranging terms, the equation can be expressed as:

$$L_i^2 - \left(A_{ix}^2 + A_{iy}^2 + R_B^2\right) = x_{cg}^2 + y_{cg}^2 + \beta_i x_{cg} + \gamma_i y_{cg} \tag{6}$$

where:

$$\begin{aligned} \beta_i &= 2\left(R_B\cos(\phi_{cg} - \theta_i) - A_{ix}\right) \\ \gamma_i &= 2\left(R_B\sin(\phi_{cg} - \theta_i) - A_{iy}\right) \end{aligned}$$

This system results in four equations (one for each cable) but with only two unknowns, x_{cg} and y_{cg}, as ϕ_{cg} is assumed to be known. To determine the position of the end-effector, we can construct an overdetermined system of linear equations from Equation 6:

$$\mu_i = x_{cg}^2 + y_{cg}^2 + \beta_i x_{cg} + \gamma_i y_{cg} \tag{7}$$

where:

$$\mu_i = L_i^2 - \left(A_{ix}^2 + A_{iy}^2 + R_B^2\right)$$

Next, by eliminating the nonlinear terms involving x_{cg}^2 and y_{cg}^2, we obtain a linear system as shown below:

$$\begin{bmatrix} \beta_2 - \beta_1 & \gamma_2 - \gamma_1 \\ \beta_3 - \beta_2 & \gamma_3 - \gamma_2 \\ \beta_4 - \beta_3 & \gamma_4 - \gamma_3 \\ \beta_4 - \beta_1 & \gamma_4 - \gamma_1 \end{bmatrix} \begin{bmatrix} x_{cg} \\ y_{cg} \end{bmatrix} = \begin{bmatrix} \mu_2 - \mu_1 \\ \mu_3 - \mu_2 \\ \mu_4 - \mu_3 \\ \mu_4 - \mu_1 \end{bmatrix} \tag{8}$$

This system represents four equations with two unknowns, x_{cg} and y_{cg}. Since this system is overdetermined, the solution can be obtained by solving it using a least-squares optimization approach. By minimizing the error between the left-hand and right-hand sides of Equation 8, the optimal values for x_{cg} and y_{cg} can be found. This approach provides an effective means of solving the forward kinematics problem in the absence of an analytical solution.

Thus, given the cable lengths $L = [L_1, L_2, L_3, L_4]^T$ and the orientation ϕ_{cg}, the position of the end-effector can be determined by solving this overdetermined system, yielding the values of x_{cg} and y_{cg}.

This intuition is used in our work, the network would estimate the ϕ_{cg} given the cable lengths such that the solving the this least square problem would result in least square error between the actual and estimated end effector position and orientation.

4 Neural Network Architecture

This study aims to integrate essential physical constraints into the network as an inductive bias to minimize the need for estimation. To achieve this, a network

is designed that directly takes cable lengths as input and outputs only the end-effector orientation, ϕ_{cg}. The estimated orientation, combined with cable lengths and equation 8, is used to calculate the end-effector position. The discrepancy between the predicted and actual positions is used as an additional physical constraint for refining orientation estimation. In essence, the predicted orientation should not only be accurate but also lead to a precise end-effector position prediction. The network architecture and its components will be detailed in the following sections.

4.1 Input Encoding

The input to the graph transformer network consists of the measured cable lengths and the robot's structural parameters. These are encoded as node and edge features within the graph. As previously described, the frame anchors and the end-effector are modeled as graph nodes. The node features represent the positions and orientations of the nodes. For the anchor nodes, these values are constant and directly derived from the robot's structure parameters. In contrast, the end-effector's position and orientation are unknown, so a learnable embedding is employed for these features.

The edge features represent the messages passed between the nodes. For the kinematic model, these messages are defined as the Euclidean distances between the nodes. While the distances between anchor nodes remain constant, the distances between the anchor nodes and the end-effector nodes are represented by the measured cable lengths.

It is important to highlight that while the node and edge features in this specific application are mostly static (due to the fixed anchor positions), the proposed framework is versatile enough to handle more complex cases, such as dynamic structures with moving anchors. This flexibility allows the method to generalize to a broader range of kinematic and structural problems.

4.2 Graph Transformer Block

The Graph Transformer Block is a crucial component of the proposed network architecture, designed to capture complex interactions within the graph representation of the robot's kinematic structure. This block leverages self-attention mechanisms to model the dependencies between nodes and edges, enabling the network to learn rich representations of the system's dynamics.

The operation of the Graph Transformer Block can be summarized as follows:

Graph Attention Layer The Graph Attention Layer computes attention scores to effectively aggregate information from both node and edge features.

Node Feature Projections: The node features $\mathbf{N} \in \mathbb{R}^{N \times d_n}$ are projected into query, key, and value representations using linear transformations:

$$\mathbf{Q}^n = \mathbf{N}\mathbf{W}_q^n, \quad \mathbf{K}^n = \mathbf{N}\mathbf{W}_k^n, \quad \mathbf{V}^n = \mathbf{N}\mathbf{W}_v^n, \tag{9}$$

where $\mathbf{W}_q^n$, $\mathbf{W}_k^n$, $\mathbf{W}_v^n \in \mathbb{R}^{d_n \times d}$ are learned weight matrices, and d is the dimensionality of the transformed features.

Edge Feature Projections: Similarly, the edge features $\mathbf{E} \in \mathbb{R}^{N \times N \times d_e}$ are projected:

$$\mathbf{Q}^e = \mathbf{E}\mathbf{W}_q^e, \quad \mathbf{K}^e = \mathbf{E}\mathbf{W}_k^e, \quad \mathbf{V}^e = \mathbf{E}\mathbf{W}_v^e, \tag{10}$$

with $\mathbf{W}_q^e$, $\mathbf{W}_k^e$, $\mathbf{W}_v^e \in \mathbb{R}^{d_e \times d}$.

Attention Mechanism: The attention scores are computed using the scaled dot-product attention for both nodes and edges.

For node features:

$$\mathbf{A}^n = \text{softmax}\left(\frac{\mathbf{Q}^n(\mathbf{K}^n)^\top}{\sqrt{d}}\right), \tag{11}$$

and the updated node representations are:

$$\mathbf{N}' = \mathbf{A}^n\mathbf{V}^n. \tag{12}$$

For edge features:

$$\mathbf{A}^e = \text{softmax}\left(\frac{\mathbf{Q}^e \odot (\mathbf{K}^e)}{\sqrt{d}}\right), \tag{13}$$

where $\odot$ denotes element-wise multiplication, and the updated edge representations are:

$$\mathbf{E}' = \mathbf{A}^e\mathbf{V}^e. \tag{14}$$

Interaction between Nodes and Edges: The updated edge features $\mathbf{E}'$ are used to enhance the node features by incorporating information about the relationships between nodes:

$$\mathbf{N}'' = \sum_{j=1}^{N} \mathbf{A}_{ij}^e \mathbf{N}_j', \tag{15}$$

where $\mathbf{A}_{ij}^e$ represents the attention score between node i and node j via the edge connecting them.

Residual Connections and Layer Normalization To facilitate training and improve convergence, residual connections are employed along with layer normalization.

Node Features Update:

$$\mathbf{N} = \text{LayerNorm}\left(\mathbf{N} + \mathbf{N}''\right). \tag{16}$$

Edge Features Update:

$$\mathbf{E} = \text{LayerNorm}\left(\mathbf{E} + \mathbf{E}'\right). \tag{17}$$

Feedforward Networks Position-wise feedforward networks are applied to the node and edge features to introduce non-linearity and enhance the model's representational capacity.

Node Feedforward Network:

$$\mathbf{N}^{\text{final}} = \mathbf{N} + \text{FFN}(\mathbf{N}), \tag{18}$$

where FFN consists of two linear transformations with a non-linear activation function (e.g., LeakyReLU) in between:

$$\text{FFN}(\mathbf{N}) = \sigma\left(\mathbf{N}\mathbf{W}_1^n + \mathbf{b}_1^n\right)\mathbf{W}_2^n + \mathbf{b}_2^n. \tag{19}$$

Edge Feedforward Network:

$$\mathbf{E}^{\text{final}} = \mathbf{E} + \text{FFN}(\mathbf{E}), \tag{20}$$

with similar definitions for the feedforward network parameters. The Graph Transformer Block effectively models the interactions between the robot's components by attending over both node and edge features. By integrating self-attention mechanisms within a graph-based framework, the block captures the complex dependencies inherent in the robot's kinematics. The use of residual connections and layer normalization stabilizes the training process and enables the construction of deeper networks.

This approach enhances the network's ability to learn physics-informed representations, as it inherently respects the structural relationships and constraints of the robot's configuration, leading to more accurate and interpretable modeling of the system.

4.3 Output Decoder

The network estimates the end-effector orientation $\hat{\phi}_{cg}$. To determine the end-effector position, given this orientation estimate and the cable lengths, a kinematic decoder layer is employed at the network output. This layer leverages the physical relationships between the cable lengths, the positions of the anchors, and the orientation of the end-effector, as formulated in Equation 8. Specifically, the decoder reconstructs the end-effector position by solving an overdetermined system of linear equations. This system is derived from geometric constraints, where the differences between the cable lengths and the end-effector position are mapped into a least-squares problem. The decoder computes the x_{cg} and y_{cg} coordinates using the anchor positions, cable lengths, and orientation.

By incorporating these physical relations directly into the model, the network becomes physics-informed, as it inherently respects the robot's geometric and kinematic constraints, improving the model's interpretability and alignment with the underlying physical system.

5 Experiment

To train the network, a set of configurations $X = [x_{cg}, y_{cg}, \phi_{cg}]^T$ is randomly sampled from a uniform distribution $X \sim$ Unif over the workspace, forming the target variable. The input to the network is the set of cable lengths L, obtained by solving the inverse kinematics of the robot.

To simulate real-world conditions, an additive zero-mean Gaussian noise with a standard deviation of 0.005 is applied to the cable lengths, introducing measurement noise. The network is optimized using a stochastic gradient descent (SGD) optimizer with a momentum factor of 0.996. The objective function to minimize is mean squared error. The learning rate is scheduled using a cosine decay strategy, starting from 0.001 and gradually decreasing to zero by the final iteration. The optimization process runs for 10,000 iterations.

6 Results

To evaluate the generalization capability of the trained model, a circular trajectory was designed and used as the test case. The results of the model's predictions are depicted in Figure 3. As seen in the figure, the predicted end-effector positions closely match the ground truth values across the entire trajectory, demonstrating strong alignment between the model's outputs and the target positions.

A small degree of fluctuation is observed in the predictions, which can be attributed to the additive measurement noise applied to the cable lengths during both training and testing. This noise was introduced to simulate realistic sensor inaccuracies and improve the robustness of the model to imperfect data.

Despite the presence of this noise, the model maintains high accuracy, with a mean squared error (MSE) of less than 0.002 millimeters over the entire trajectory. This low MSE indicates that the model has effectively learned the underlying kinematics of the cable-driven parallel robot and is capable of producing precise predictions, even in the presence of noise.

The results confirm that the proposed physics-informed graph transformer network generalizes well to unseen trajectories and demonstrates a high level of predictive accuracy for the inverse kinematics task.

References

[1] H. J. Asl and F. Janabi-Sharifi. Adaptive neural network control of cable-driven parallel robots with input saturation. *Engineering applications of artificial intelligence*, 65:252–260, 2017.

[2] R. Babaghasabha, M. A. Khosravi, and H. D. Taghirad. Adaptive robust control of fully-constrained cable driven parallel robots. *Mechatronics*, 25:27–36, 2015.

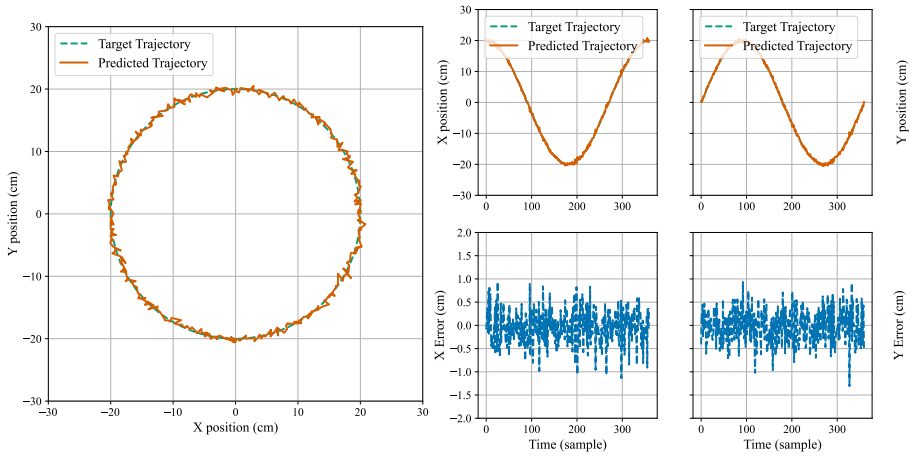

Figure 3: Estimated and actual end-effector positions on test trajectory

[3] C. Chen, Y. Wu, Q. Dai, H.-Y. Zhou, M. Xu, S. Yang, X. Han, and Y. Yu. A survey on graph neural networks and graph transformers in computer vision: A task-oriented perspective. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[4] S. Cuomo, V. S. Di Cola, F. Giampaolo, G. Rozza, M. Raissi, and F. Piccialli. Scientific machine learning through physics–informed neural networks: Where we are and what's next. *Journal of Scientific Computing*, 92(3):88, 2022.

[5] M. Gori, G. Monfardini, and F. Scarselli. A new model for learning in graph domains. In *Proceedings. 2005 IEEE international joint conference on neural networks, 2005.*, volume 2, pages 729–734. IEEE, 2005.

[6] J. Hatori, Y. Kikuchi, S. Kobayashi, K. Takahashi, Y. Tsuboi, Y. Unno, W. Ko, and J. Tan. Interactively picking real-world objects with unconstrained spoken language instructions. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3774–3781. IEEE, 2018.

[7] J. Ma, X. Duan, and D. Zhang. Kernel extreme learning machine-based general solution to forward kinematics of parallel robots. *CAAI Transactions on Intelligence Technology*, 8(3):1002–1013, 2023.

[8] T. Ma, H. Xiong, L. Zhang, and X. Diao. Control of a cable-driven parallel robot via deep reinforcement learning. In *2019 IEEE International Conference on Advanced Robotics and its Social Impacts (ARSO)*, pages 275–280. IEEE, 2019.

[9] A. Moori, J. Khoramdel, and S. A. A. Moosavian. Deep learning approach for object tracking of roboeye. In *2019 7th International Conference on Robotics and Mechatronics (ICRoM)*, pages 386–391. IEEE, 2019.

[10] C. Peng, F. Xia, V. Saikrishna, and H. Liu. Physics-informed graph learning. In *2022 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 732–739. IEEE, 2022.

[11] F. Pistilli and G. Averta. Graph learning in robotics: a survey. *IEEE Access*, 2023.

[12] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations. *arXiv preprint arXiv:1711.10561*, 2017.

[13] A. Sanchez-Gonzalez, J. Godwin, T. Pfaff, R. Ying, J. Leskovec, and P. Battaglia. Learning to simulate complex physics with graph networks. In *International conference on machine learning*, pages 8459–8468. PMLR, 2020.

[14] J. Sun, L. Yu, P. Dong, B. Lu, and B. Zhou. Adversarial inverse reinforcement learning with self-attention dynamics model. *IEEE Robotics and Automation Letters*, 6(2):1880–1886, 2021.

[15] H. D. Taghirad. *Parallel robots: mechanics and control*. CRC press, 2013.

[16] A. Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

[17] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, et al. Graph attention networks. *stat*, 1050(20):10–48550, 2017.

[18] R. Wang and R. Yu. Physics-guided deep learning for dynamical systems: A survey. *arXiv preprint arXiv:2107.01272*, 2021.

[19] 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*, 377:73–84, 2020.

[20] Q. Xu, Y. Pang, X. Zhou, and Y. Liu. Pigat: Physics-informed graph attention transformer for air traffic state prediction. *IEEE Transactions on Intelligent Transportation Systems*, 2024.

[21] D.-Y. Yu. Parallel robots pose accuracy compensation using artificial neural networks. In *2008 IEEE International Conference on Mechatronics and Automation*, pages 750–754. IEEE, 2008.

[22] Y. Zhang, Z. Wang, M. Li, and P. Gao. Rp-yolox-dl: a deep learning hybrid method for parallel robots target positioning. *Measurement Science and Technology*, 34(10):105010, 2023.