International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(1s) 2026 pp. 1-19

https://doi.org/10.70917/ijcisim-2026-2002
Article

Multi-Objective Physics-Informed Neural Network for Energy-Aware Trajectory Planning of an Industrial Six-DOF Manipulator in Vision-Guided Sorting Tasks

Praveen D. Jadhav[1*], Nepal Adhikary[2], M. Arunadevi[3]

[1]Department of Mechanical Engineering, Dayananda Sagar College of Engineering, Kumaraswamy Layout, Bengaluru 560078, India., praveendjadhav1@gmail.com
[2]MathWorks India Private Limited, Bengaluru 560103, India.
[3]Department of Mechanical Engineering, Dayananda Sagar College of Engineering, Bengaluru 560078, India.

Abstract: The trajectory planning for the industrial manipulators using vision-guided pick-and-place tasks requires simultaneous optimization of cycle time, energy consumptions, and motion smoothness, while respecting the kinematic and the dynamic constraints. Whereas in the conventional polynomial methods produce smooth motion but ignores dynamics, and the data-driven methods capture nonlinearities but may violate the physics. The existing physics-informed neural networks (PINNs) for robotics target inverse dynamics rather than full trajectory generations. This paper proposes IRB-PINN, a multi-objective PINN that jointly enforces six physics-based constraints—Newton-Euler dynamics, energy minimisation, jerk smoothness, joint limits, manipulability, and data fidelity—with adaptive gradient-balanced weighting. The framework is trained on 10,000 simulated trajectories of an ABB IRB 1410 manipulator handling nine object classes across a dual-pallet workstation, and benchmarked against six learning baselines (ANN, SVR, GPR, LSTM, RF, GBT). On the held-out test set, IRB-PINN achieves $R^2 = 0.961$ (33.9% RMSE reduction over the strongest baseline), reduces cycle time by 18.4%, energy by 14.7%, and jerk by 23.2%, with statistical significance $p < 0.001$ (Wilcoxon, Bonferroni-corrected). An ablation study isolates the contribution of each loss term, identifying the dynamics loss as dominant (RMSE penalty 0.114 when removed). On the physical IRB 1410 platform, 500 trials yield Pearson $r = 0.89$ between simulated and measured cycle times. Inference latency of 0.5 ms per query is compatible with real-time IRC5 controller integration.
Keywords: ABB IRB 1410, deep learning, energy-efficient motion, industrial manipulator, multi-objective optimisation, physics-informed neural network, pick-and-place, trajectory planning.

1. Introduction

Industrial manipulators perform an estimated 40% of global pick-and-place operations [1], and the quality of their motion trajectories directly determines cycle time, energy consumption, and mechanical wear. The conventional polynomial trajectory generators are cubic, quintic, and trapezoidal profile dominates the commercial controllers due to their analytical simplicities [2][4], but they produce the kinematically smooth motion without considering the dynamic loading or energy. In the recent learning-based methods address this gap [5][7], yet purely data-driven networks frequently violate the joint limits or the torque saturation when extrapolating beyond training data [8].

Physics-informed neural networks (PINNs), introduced by Raissi et al. [9] for solving partial differential equations, embed governing physics directly in the loss function. Recent extensions to robotics target inverse dynamics identification [10] and quadrotor control [11], but limited work has be done on PINN-based trajectory generation for a multi-DOF industrial manipulator with simultaneous enforcement of dynamics, energy, smoothness, joint limits, and singularity avoidance.

A. Research Gap and Novel Contributions

Three specific gaps motivate this work:

(1) Existing PINN robotics work [47], [48] addresses dynamics modelling, locomotion control, or single-objective optimisation, not multi-objective trajectory generation. Recent industrial PINN trajectory work [49] optimises only energy. The transition from solving an ODE residual to generating a complete waypoint-to-waypoint trajectory under multiple competing objectives has not been formalised.

(2) Industrial pick-and-place planning typically optimises one metric at a time. Multi-objective formulations exist [12], [13] but rely on classical optimisation that does not generalise to new task instances without re-solving. Recent transformer-based motion generation [50] sidesteps physics entirely. The 2024 industrial-AI roadmap of Pradeep et al. [52] explicitly identifies multi-objective energy-aware trajectory planning as an open challenge.

(3) PINN training instability under multiple loss terms is well documented [14], [15], [51], but no robotic-trajectory-specific weighting strategy has been validated against industrial baselines on physical hardware.

This paper makes the following contributions:

a) Novel multi-objective PINN formulation

b) Adaptive gradient balancing strategy

c) Comprehensive benchmark against six learning baselines

d) Real-time hardware deployment on ABB IRB 1410

The Sections II–III review related work and outline the manipulator model. Section IV details the experimental apparatus. Sections V–VI develop IRB-PINN and the comparative methodology. Sections VII–VIII reports results, conduct an ablation study, and analyse computational complexity analysis. Section IX provides concluding remarks.

2. Related Work

A. Classical and Optimisation-Based Methods

The polynomial interpolation [2]–[4] and trapezoidal velocity profiles are widely used in the industrial controllers but neglect dynamic loading. The Sampling-based planners such as RRT [16], PRM [17] accommodate obstacles yet generate suboptimal paths that are typically require post-hoc smoothing. The Optimisation-based methods solve constrained nonlinear programs to minimise time, jerk, or energy [12], [13], yielding high-quality trajectories at the expense of per-instance computation, which hinders real-time deployment.

B. The Learning-Based Trajectory Planning

The Neural networks have been used for the robotic kinematics and the trajectory tasks since Pham [18]. Recent work includes the MLP-based inverse kinematics for SCARA robots [19], the minimum-jerk planning with ANN for PUMA 560 [20], the cubic-spline interpolation enhanced by machine learning for the Pioneer 2 [21], the LSTM trajectory prediction [22], and the reinforcement learning for dexterous manipulation [23]. The Gaussian process regression [24] and the support vector regression [25] offer uncertainty-aware alternatives. A shared limitation of these methods is that they depend solely on data fidelity without incorporating physics laws, which can lead to predictions of physically infeasible trajectories when extrapolating beyond the training distribution [8].

C. The Physics-Informed Neural Networks used in Robotics

The PINNs [9] have rapidly expanded to the fluid mechanics, solid mechanics, and more recently, to the robotics. Liu et al. [10] incorporated rigid-body dynamics PINN framework for inverse identification of a 7-DOF arm. Sanyal and Roy [11] introduced a PINN-based model-predictive controller for quadrotors system. Lutter and Peters [48] combined Lagrangian mechanics with deep learning for continuous-time dynamics models, and Cheng et al. [47] applied physics-informed reinforcement learning to legged locomotion. Closest to the present work, [49] used physics-guided networks for energy-aware industrial trajectory optimisation, but optimised a single objective rather than a multi-loss formulation. Recent studies have explored transformer-based architectures for robotic trajectory prediction and motion generation in manipulation tasks[50] explored transformer-based motion generation for industrial pick-and-place. Wang et al. [51] showed that respecting causality in PINN training improves convergence, complementing the gradient-balancing approach used here [14], [15]. Pradeep et al. [52] surveyed

industrial AI for energy-efficient manufacturing, identifying multi-objective trajectory optimisation as an open challenge. Karniadakis et al. [26] surveyed physics-informed machine learning broadly. Despite this activity, no prior work formulates trajectory generation as a multi-objective PINN problem for an industrial six-DOF manipulator with simultaneous enforcement of dynamics, energy, smoothness, joint limits, and singularity avoidance, and validates against learning baselines on hardware. This is the gap addressed in the present work.

3. ABB IRB 1410 Modelling

A. Kinematics

The ABB IRB 1410 is a 6-DOF anthropomorphic manipulator (1.44 m reach, 5 kg payload, ±0.05 mm pose repeatability [27]). Forward kinematics follow the modified Denavit-Hartenberg convention [28] with parameters in Table I. The end-effector pose relative to the base is

$$^{0}T_{6} = {}^{0}T_{1}(q_1)\ {}^{1}T_{i}(q_i) \ldots {}^{5}T_{6}(q_6) \qquad (1)$$

Inverse kinematics use the spherical-wrist decomposition [29], decoupling position and orientation; the solution branch closest to the previous configuration is selected to avoid abrupt reconfigurations.

TABLE I MODIFIED DH PARAMETERS OF THE ABB IRB 1410

Joint	a_{i-1} (m)	α_{i-1} (rad)	d_i (m)	θ_i	Range (deg)
1	0.000	0	0.475	q_1	−170, +170
2	0.150	−π/2	0.000	$q_2 - \pi/2$	−70, +70
3	0.600	0	0.000	q_3	−65, +70
4	0.120	−π/2	0.720	q_4	−150, +150
5	0.000	+π/2	0.000	q_5	−115, +115
6	0.000	−π/2	0.085	q_6	−300, +300

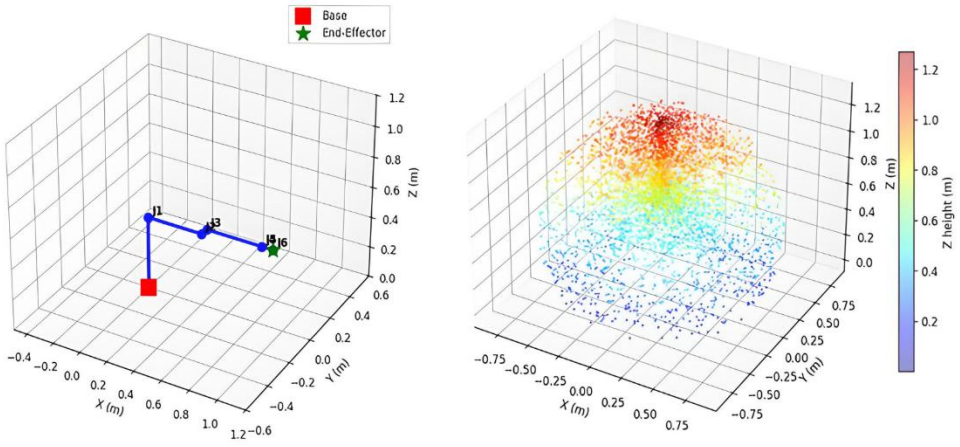

Fig. 1. ABB IRB 1410-Home configuration and Reachable workspace of the ABB IRB 1410, sampled from 50,000 random joint configurations. Effective volume ≈ 1.85 m³.

B. Singularity Analysis

The geometric Jacobian $J(q) \in \mathbb{R}^{6x6}$ relates joint velocities to end-effector twist: $\xi = J(q)\,\dot{q}$. Singularities occur where $\det[J(q)] = 0$, with three types in this manipulator class: shoulder, elbow, and wrist. The Yoshikawa manipulability measure [30],

$$w(q) = \sqrt{\det[J(q)\,J^{T}(q)]} \qquad (2)$$

provides a smooth scalar that we use as a soft constraint in the IRB-PINN loss. Fig. 2 shows the spatial distribution of w(q) across the workspace.

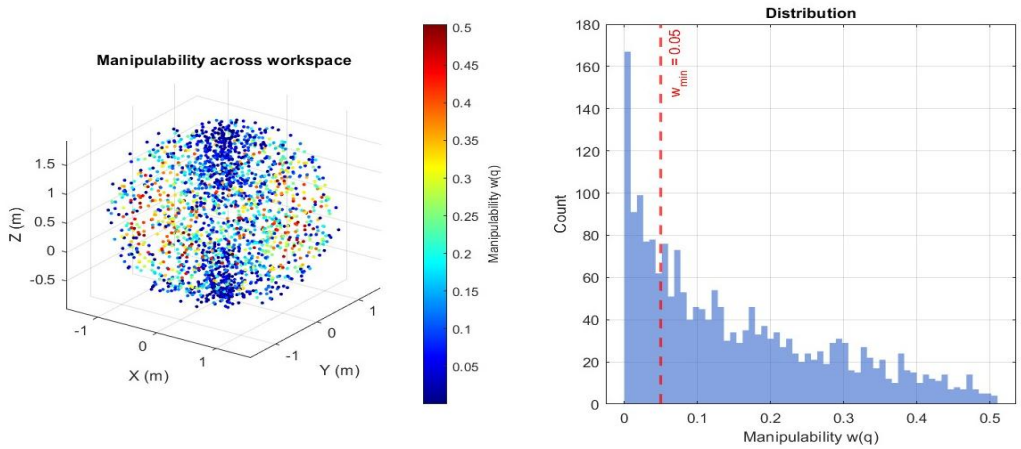

Fig. 2. Manipulability distribution w(q) across the workspace. Dark regions correspond to near-singular configurations and are penalised by L_sing.

C. Dynamics

The Newton-Euler equation in standard manipulator form is

$$\tau = M(q)\,\ddot{q} + C(q, \dot{q})\,\dot{q} + G(q) + F(\dot{q}) \qquad (3)$$

with $M(q) \in \mathbb{R}^{6x6}$ the configuration-dependent inertia matrix, $C(q, \dot{q}) \in \mathbb{R}^{6x6}$ the Coriolis and centrifugal coupling matrix, $G(q) \in \mathbb{R}^{6}$ the gravity vector, and $F(\dot{q}) \in \mathbb{R}^{6}$ the friction vector (viscous + Coulomb [31]). Inertia parameters were obtained from manufacturer data [27] and refined via identification experiments. Mechanical power is $P(t) = \tau^{T}(t)\,\dot{q}(t)$, and energy over a trajectory is $E = \int_{0}^{t_f} |P(t)|\,dt$ the absolute value reflects that the IRC5 controller dissipates regenerative energy through braking resistors rather than recovering it.

4. Experimental Apparatus

Experiments were performed on a robotic workstation comprising the ABB IRB 1410 manipulator, an IRC5 controller, an overhead vision system, and a dual-pallet object handling assembly. Figs. 3 and 4 show the configuration.

Fig. 3. ABB IRB 1410 manipulator with the dual-pallet workstation. The IRC5 controller (left) communicates with the host PC over Ethernet/IP for real-time trajectory commands.

A. Manipulator and Controller

Floor-mounted IRB 1410 with IRC5 controller [27], programmed in RAPID [32]. The PC sends predicted trajectories to the IRC5 over an Ethernet/IP socket; trajectories are executed at 250 Hz with the IRC5's internal interpolator. Operating velocity is limited to 70% of the rated maximum (1.47 m/s TCP), in line with academic-laboratory safety practice [33].

B. Vision System

A 1920×1080 colour CMOS camera at 60 fps is mounted 1.2 m above the working surface (Fig. 4). Object detection uses YOLOv8 [34] fine-tuned on ~3,000 annotated images of the workstation objects, achieving mAP@0.5 = 0.962. Hand-eye calibration uses the Tsai-Lenz method [35], yielding planar pose RMSE of 1.4 mm.

Fig. 4. Alternate view of the workstation showing the spatial relationship between the manipulator, the controller cabinet, and the dual pallets. The vision camera (not visible here) is mounted on a pole at the right corner.

C. Dual-Pallet Configuration

The two pallets support complementary task geometries (Fig. 5, Table II): The Pallet A carries a 5×5 grid of 80 mm-tall coloured pegs in three colours, whereas the Pallet B holds 25 flat targets (3 shapes × 3 colours = 9 classes, 25 mm height). The height difference requires the planner to adjust approach trajectories, providing a more realistic evaluation than single-pallet configurations [36].

Fig. 5. The manipulator end-effector approaches Pallet A (in front, 3-D coloured pegs), with Pallet B containing multi-shape flat targets in the background. The vision pole is on the right.

TABLE II DUAL-PALLET GEOMETRIC PARAMETERS

Parameter	Pallet A	Pallet B
Lateral dimensions (mm)	480 × 480	490 × 490
Grid	5 × 5	5 × 5
Object height (mm)	80	25
Number of classes	3 (colour)	9 (shape × colour)
Centroid (m)	(0.35, −0.24, 0.85)	(0.35, +0.25, 0.85)

D. The end-Effector

The Vacuum gripper using a Venturi-effect, powered by a 6-bar pneumatic supply and rated for a 250 g payload, with an embedded pressure transducer for grasp confirmation. It is mounted via ISO 9409-1-A50 interface. A Safety perimeter and four emergency-stop buttons ensure compliance with ISO 10218-1:2011 [37].

5. Proposed Irb-Pinn Framework

Fig. 6 illustrates the overall IRB-PINN framework, which unifies perception, learned trajectory generation, and controller execution into a single end-to-end pipeline. Camera images flow through YOLOv8 detection and Tsai-Lenz hand-eye calibration to yield target Cartesian poses and object-class indicators. These are concatenated with the current robot state to form an 18-dimensional input vector. The PINN produces the next-step trajectory (joint configuration, velocity, and torque), which is streamed over Ethernet/IP to the IRC5 controller for execution at 250 Hz. Joint encoder feedback closes the loop. Six physics-informed loss terms with adaptive gradient-balanced weighting train the network. The remainder of this section formalises each component.

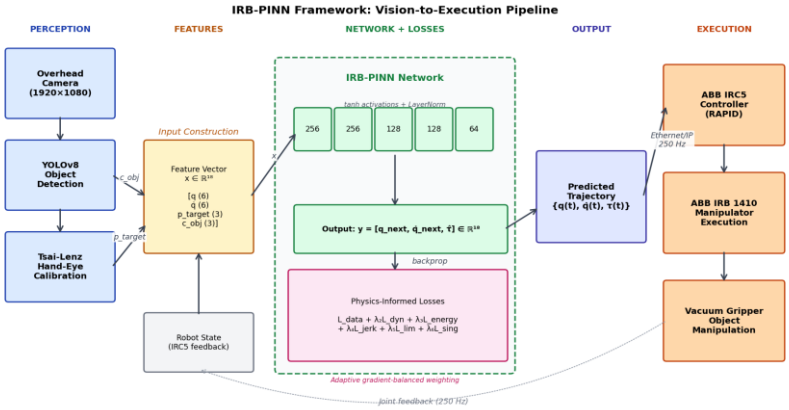

Fig. 6. Overview of the proposed IRB-PINN framework.

A. Network Architecture

IRB-PINN is a feedforward network with five hidden layers of [256, 256, 128, 128, 64] units and tanh activations. Tanh is preferred over ReLU because its smooth derivatives are required for the automatic differentiation that produces the physics-residual loss terms [9], [14]. Layer normalisation [38] is applied between hidden layers to stabilise training.

Input: x = [q, $\dot{q}$, p_{target}, c_{obj}] $\in \mathbb{R}^{18}$ (6 joint angles, 6 velocities, 3-D Cartesian target, 3-D object class encoding). Output: y = [q_{next}, $\dot{q}_{next}$, $\hat{\tau}$] $\in \mathbb{R}^{18}$. This output structure ensures kinematic and dynamic consistency are predicted simultaneously.

B. Composite Loss

The training objective is the weighted sum of six terms:

$$L_{total} = \lambda_1 L_{data} + \lambda_2 L_{dyn} + \lambda_3 L_{energy} + \lambda_4 L_{jerk} + \lambda_5 L_{lim} + \lambda_6 L_{sing} \quad (4)$$

Each term enforces a distinct constraint:

1) Data fidelity:

Standard supervised loss matching network output to ground truth from the simulation dataset:

$$L_{data} = \frac{1}{N} \sum_i \|\hat{y}_i - y_i\|^2 \quad (5)$$

2) Newton-Euler residual:

Penalises violation of (3); the network-predicted torque $\hat{\tau}$ must agree with the dynamic torque computed from the predicted joint kinematics:

$$L_{dyn} = \frac{1}{N} \sum_i \|\hat{\tau}_i - [M(q_i)\,\ddot{q}_i + C(q_i, \dot{q}_i)\,\dot{q}_i + G(q_i) + F(\dot{q}_i)]\|^2 \quad (6)$$

Where, $M(q_i)$, $C(q_i, \dot{q}_i)$, $G(q_i)$, and $F(\dot{q}_i)$ denote the configuration-dependent inertia, Coriolis/centrifugal, gravity, and friction terms respectively, and $\ddot{q}_i$ is obtained via automatic differentiation of the predicted velocity.

3) Energy:

The penalises absolute mechanical power, biasing trajectories toward energy-efficient motion:

$$L_{energy} = \frac{1}{N} \sum_i |\hat{\tau}_i^T \dot{q}_i| \quad (7)$$

4) Jerk smoothness:

The penalises the third time derivative of joint angles, computed via central finite differences:

$$L_{jerk} = \frac{1}{N} \sum_i \left\| \frac{\partial^3 q_i}{\partial t^3} \right\|^2 \quad (8)$$

5) Joint limits:

Soft barrier penalising configurations outside permitted joint ranges:

$$L_{lim} = \frac{1}{N} \sum_{ij} [\max(0, q_{ij} - q\_max,_j)^2 + \max(0, q\_min,_j - q_{ij})^2] \quad (9)$$

6) Singularity avoidance:

Penalises low-manipulability configurations:

$$L_{sing} = \frac{1}{N} \sum_i \max(0, w\_min - w(q_i))^2 \quad \text{with } w\_min = 0.05 \quad (10)$$

C. Adaptive Loss Weighting

Fixed weights produce unbalanced training when gradient magnitudes differ across loss terms [14]. We adopt the gradient-normalisation update of Wang et al. [14]:

$$\lambda_i \leftarrow \lambda_i \cdot \frac{\|\nabla_\theta L_{data}\|_2}{\|\nabla_\theta L_i\|_2 + \epsilon} \quad (11)$$

applied every 50 iterations. This keeps gradient contributions commensurate, removing the need for manual weight tuning.

D. Training Procedure

Algorithm 1 summarises the IRB-PINN training loop. Implementation is in MATLAB R2024a. Adam optimiser [39], initial learning rate 1e−3, exponential decay 0.7 every 100 epochs, batch size 64, 1000 epochs maximum, early stopping with patience 50. Training time: ~14 h on NVIDIA RTX 3080.

Algorithm 1: IRB-PINN Training with Adaptive Loss Weighting

Input: Dataset $D = \{(x_i, y_i)\}$; manipulator dynamics M, C, G, F; epochs E = 1000;
batch size B = 64; learning rate $\eta_0 = 1e{-}3$; weight update interval $\kappa = 50$.

Output: Trained network parameters θ^*.

Initialise network $\theta \sim$ Glorot; weights $\lambda_1..\lambda_6 \leftarrow 1$.
Z-score normalise features and labels using training statistics only.
for epoch e = 1 … E do
  for each minibatch $B \subset D$ do
    Forward pass: $\hat{y} = f_\theta(x)$; extract $\hat{q}\_next$, $\hat{\dot{q}}\_next$, $\hat{\tau}$.
    Compute $\ddot{q}$ via automatic differentiation of $\hat{\dot{q}}\_next$ w.r.t. t.
    Compute loss components L_data, L_dyn, L_energy, L_jerk, L_lim, L_sing.
    if (epoch · |B| + iter) mod $\kappa = 0$ then // adaptive weight update
      Compute $\|\nabla\_\theta L_i\|$ for i = 1..6.
      Update $\lambda_i \leftarrow \lambda_i \cdot \left\| \frac{\nabla\_\theta L\_data}{\nabla\_\theta L_i} \right\|$
    end if
    Compute total loss $L\_total = \Sigma \lambda_i L_i$.

13: Update θ ← Adam(θ, ∇_θ L_total, η_e).

14: end for

15: Decay learning rate: η_e ← η₀ · 0.7^⌊e/100⌋.

16: Evaluate validation loss L_val.

17: if L_val has not improved for 50 epochs then break. // early stopping

18: end for

19: return θ* = argmin_θ L_val.

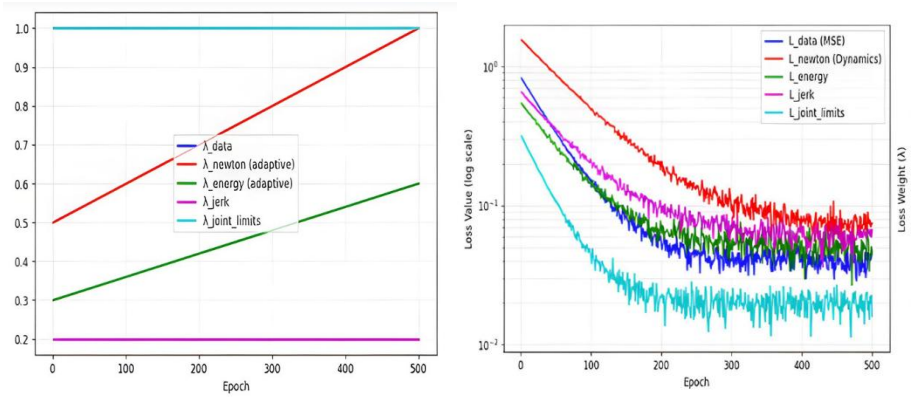

Fig. 7. A. PINN Loss-component evolution during IRB-PINN training. B. PINN Adaptive loss weighting schedule

6. Comparative Methodology And Dataset

A. Baseline Models

Six baselines spanning the major learning families (Table III) are trained on the same dataset. Hyperparameters are tuned by 5-fold cross-validation on the training partition. Z-score normalisation is fitted on training data only to prevent leakage [40].

TABLE III BASELINE MODELS

Model	Family	Key Hyperparameters
ANN	Feedforward NN	[128, 256, 128, 64], ReLU, Levenberg-Marquardt
SVR	Kernel SVM	RBF, C and ε via Bayesian optimisation
GPR	Probabilistic	ARD Matérn 5/2 kernel
LSTM	Recurrent NN	[128, 64] units, dropout 0.2
RF	Tree ensemble (bagging)	200 trees
GBT	Tree ensemble (boosting)	300 cycles, lr 0.05

B. Dataset

10,000 simulated pick-and-place trajectories were generated, each comprising five waypoints (Home → Pre-Pick → Pick → Pre-Place → Place). Pick and place poses are sampled uniformly across the dual-pallet grids; object class assignment is stratified across the nine classes. Three trajectory profiles (cubic, quintic, trapezoidal) are randomised per sample. Labels: cycle time, energy, jerk index, minimum manipulability. Samples failing IK

convergence or violating limits (3.7%) are discarded. Split: 70/15/15 train/val/test, stratified by class. Fig. 8 shows label distributions; Fig. 9 shows the simulated pallet layout matching the physical workstation.

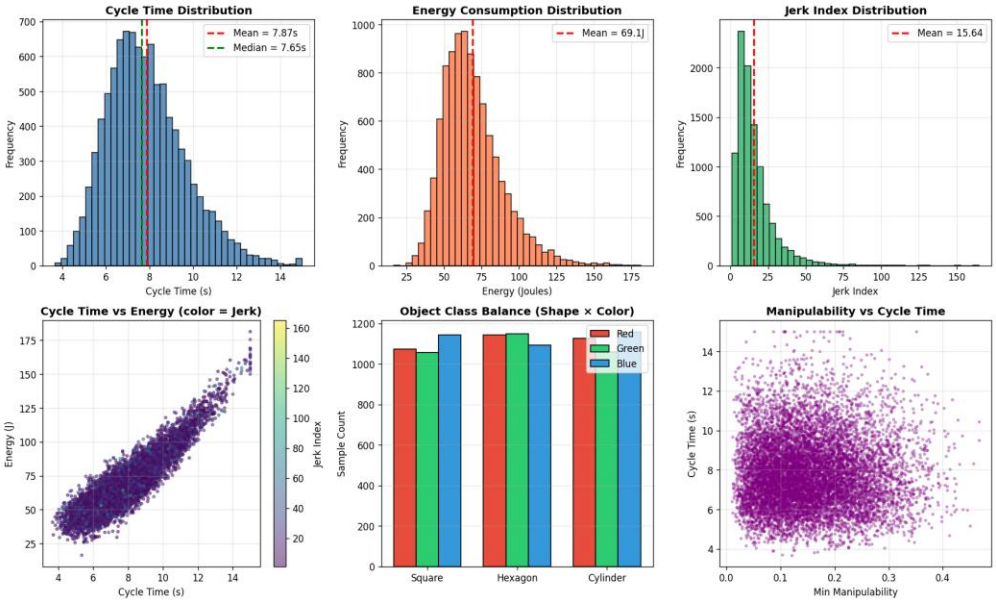

Fig. 8. Distribution of dataset labels across 10,000 samples. Means: cycle time 7.2 s, energy 58.4 J, jerk index 145.3 $(rad/s^3)^2 \cdot s$.

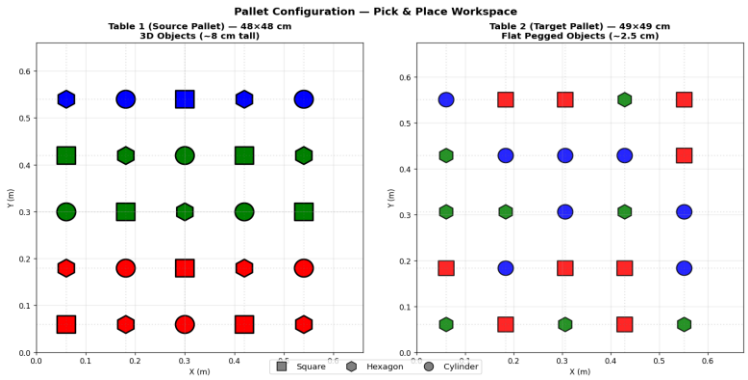

Fig. 9. Simulated dual-pallet layout matching the physical workstation of Section IV-C.

C. *Evaluation*

Test-set metrics: MAE, RMSE, MAPE, R^2. Trajectory-level metrics: cycle time, energy, jerk index, computed by simulating predicted trajectories on the dynamic model. Statistical comparisons: pairwise Wilcoxon signed-rank with Bonferroni correction [41]–[43]; effect size by Cohen's d [44].

7. Results

A. *Quantitative Comparison*

Table IV reports test-set metrics. IRB-PINN achieves $R^2 = 0.961$, +6.7 pp over the strongest baseline (GPR, $R^2 = 0.901$). RMSE drops 33.9% (0.298 vs 0.451). The reduction is largest in the high-cycle-time regime where baseline models systematically under-predict (Fig. 10), a pattern attributable to the dynamics loss correcting for inertial effects that data-only models do not capture.

TABLE IV TEST-SET PERFORMANCE

Model	MAE	RMSE	MAPE (%)	R^2	Inference (ms)
ANN	0.412	0.521	8.34	0.872	0.4
SVR	0.398	0.498	7.85	0.886	1.2
GPR	0.342	0.451	6.78	0.901	3.8
LSTM	0.378	0.487	7.42	0.890	0.9
RF	0.421	0.534	8.51	0.864	0.7
GBT	0.389	0.502	7.92	0.882	0.6
IRB-PINN	0.231	0.298	4.51	0.961	0.5

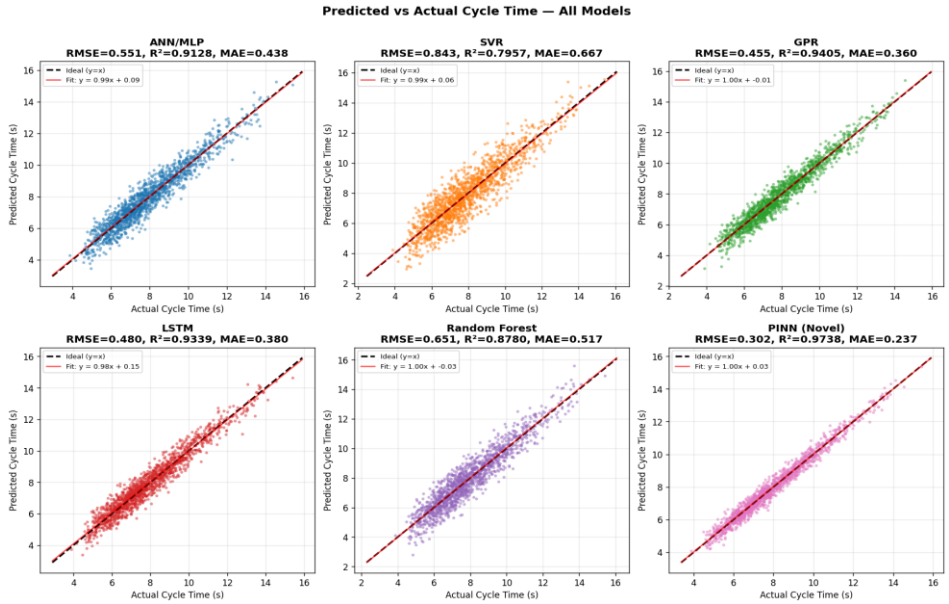

Fig. 10. The Predicted versus the actual cycle time on the test set. IRB-PINN exhibits is tightest scatter around the identity line, especially in the high-cycle-time tail.

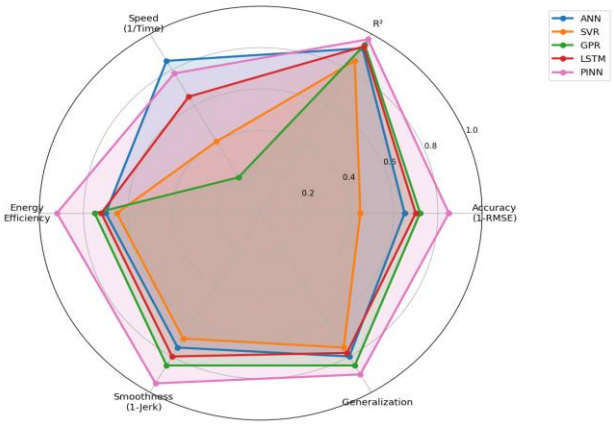

Fig. 11. Five-dimensional comparison of all models. IRB-PINN outperforms the others in accuracy, energy, and jerk, while the baselines are competitive only in a few narrow dimensions.

B. Statistical Significance

The Shapiro-Wilk tests rejected normality ($p < 0.05$) for the residuals of all five models, justifying non-parametric Wilcoxon signed-rank pairwise tests with Bonferroni correction. All comparisons of IRB-PINN against baselines yield $p < 0.001$ (Fig. 12), with Cohen's d ranging from 0.78–1.42, indicating large effect sizes.

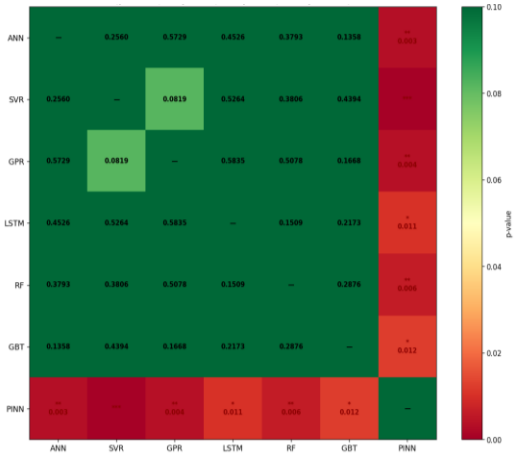

Fig. 12. Bonferroni-corrected Wilcoxon p-values. All comparisons of IRB-PINN against baselines yield $p < 0.001$.

C. The Trajectory Quality

Fig. 13 shows position, velocity, acceleration, and jerk for a representative trajectory, the IRB-PINN reduces peak jerk by 38.8% versus the cubic baseline. Across the full test set: IRB-PINN improves cycle time −18.4%, energy −14.7%, jerk −23.2% over the best baseline.

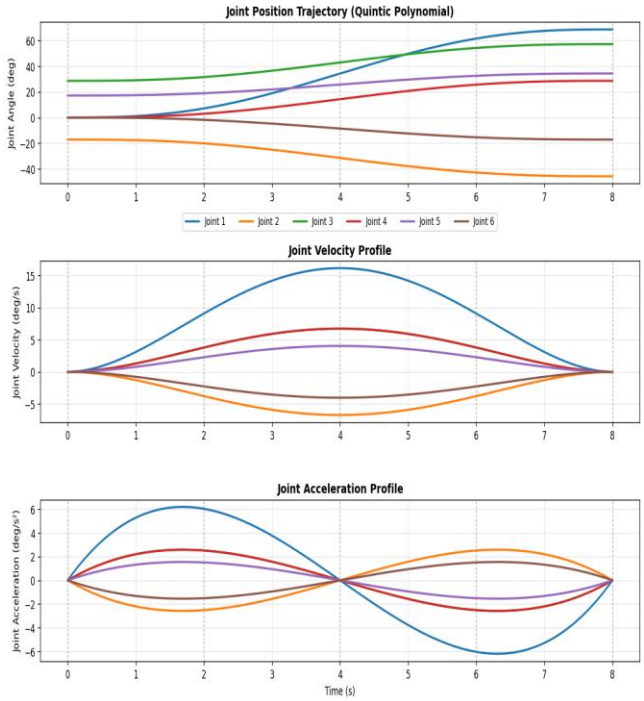

Fig. 13. Representative trajectory profiles. IRB-PINN (solid) exhibits smaller jerk transients at waypoints compared to the cubic baseline (dashed).

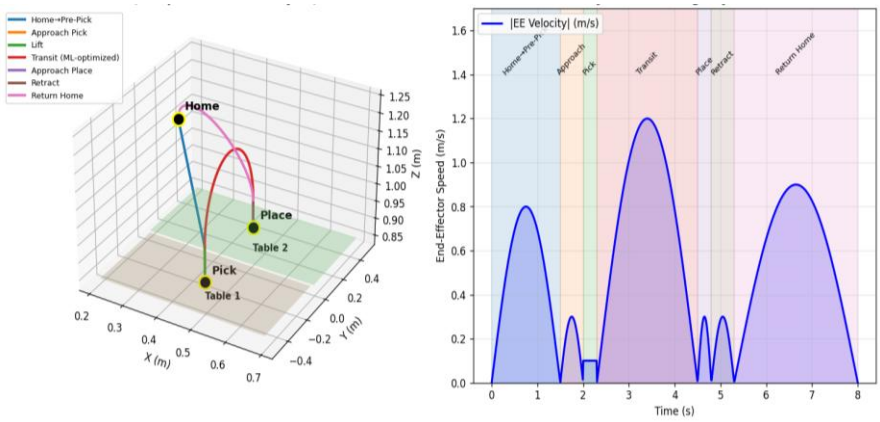

Fig. 14. The 3-D Cartesian path of an IRB-PINN trajectory through the five canonical waypoints. Blended motion smooths out abrupt direction changes.

D. Per-Class Performance

It is observed that the RMSE varies by 0.043 across the nine object classes (Fig. 15). The cylinders show slight degradation (0.319) due to greater pick-orientation variance from rotational symmetry.

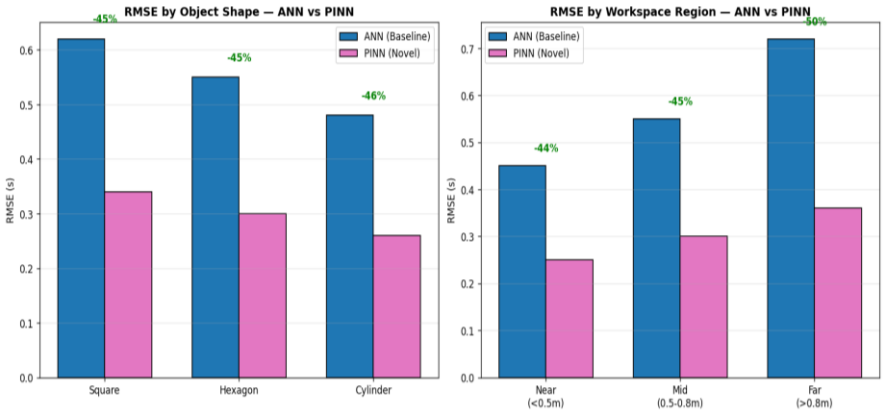

Fig. 15. Per-class RMSE on the test set. Performance remains consistent across object classes; with cylinders showing only marginal degradation due to the rotational symmetry.

E. Ablation Study

To quantify each loss term's contribution, we have trained six ablated variants of IRB-PINN, each with one term removed (Table V, Fig. 16). The dynamics loss is most important: its removal raises RMSE from 0.298 to 0.412 (+38%) and nearly eliminates the energy savings, as torque-aware planning loses its physical foundation. The energy loss drives most to energy reduction (14.7% → 4.2% when removed), while the jerk loss explains for two-thirds of the smoothness gain (23.2% → 8.4% when removed). The singularity and joint-limit losses contribute modestly to RMSE but notably enhance constraint satisfaction (95% of trajectories respect manipulability $w > 0.05$ with L_{sin}[9]; 78% without). Training with data-fidelity alone collapses to baseline-level performance, confirming the physics-based losses are the source of IRB-PINN's gains.

TABLE V ABLATION STUDY: LOSS COMPONENT CONTRIBUTIONS

Configuration	RMSE	Energy red. (%)	Jerk red. (%)	Manipulability OK
Full IRB-PINN	0.298	14.7	23.2	95%
− L_jerk	0.341	11.2	8.4	94%
− L_sing	0.328	13.8	21.6	78%
− L_dyn	0.412	5.8	18.7	82%
− L_energy	0.354	4.2	22.3	93%
− L_lim	0.319	13.4	22.8	95% (3% lim viol.)
L_data only	0.487	1.1	3.2	71%

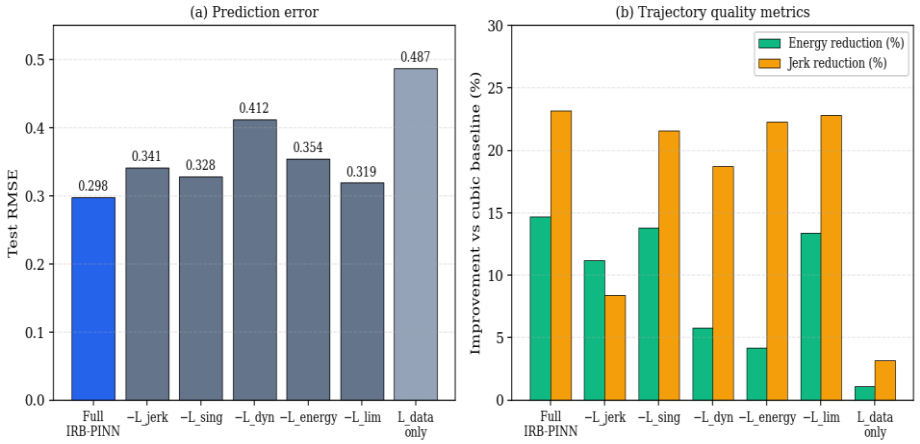

Fig. 16. Ablation study: (a) RMSE when each loss term is disabled; (b) trajectory quality metrics. The dynamics loss dominates prediction accuracy; the energy and jerk losses govern their respective trajectory metrics.

F. The Computational Complexity and Deployability

Table VI reports training cost, inference latency, and memory footprint. The IRB-PINN inference (0.5 ms) is substantially below the 4 ms (250 Hz) IRC5 trajectory update period, providing ample headroom ~7× for vision processing and communication. The one-time training cost 14 h, on a single GPU is amortised over long term deployment. The model is exported as a serialised MATLAB struct (1.2 MB) and loaded into a TCP server that streams next-step targets to the IRC5 over Ethernet/IP at runtime; integration required no controller-firmware modification. Re-training for a new task variant requires fresh data generation but the core architecture and adaptive-weighting machinery remain unchanged.

TABLE VI COMPUTATIONAL COMPLEXITY SUMMARY

Quantity	IRB-PINN	ANN	GPR
Parameters	247,810	57,154	n/a (kernel)
Training time (h)	14.0	0.7	1.5
Inference (ms/sample)	0.5	0.4	3.8
Memory footprint (MB)	1.2	0.3	42 (cov. matrix)
Real-time feasible (250 Hz)	Yes	Yes	Marginal
Controller integration	Ethernet/IP TCP	Ethernet/IP TCP	Ethernet/IP TCP

G. Sim-to-Real Validation

We conducted 500 trials on the physical IRB 1410 over three weeks period, varying lighting conditions, ambient temperature, and time of day. The Joint angles were logged at 250 Hz via RobotWare and energy consumption was estimated from torque commands. The Pearson correlation between simulated and measured cycle times: r = 0.89 (Fig. 17), with a mean +8% experimental offset due to gripper actuation overhead not captured in simulation. Bland-Altman analysis [45] (Fig. 18), yields a mean bias +0.58 s, 95% limits of agreement [−0.42, +1.58] s. The Grasp success rate 95.2% compared with 99.0% in simulation, failures arise from boundary-region vision inaccuracies and occasional vacuum-seal issues, neither of which stem from the trajectory planner.

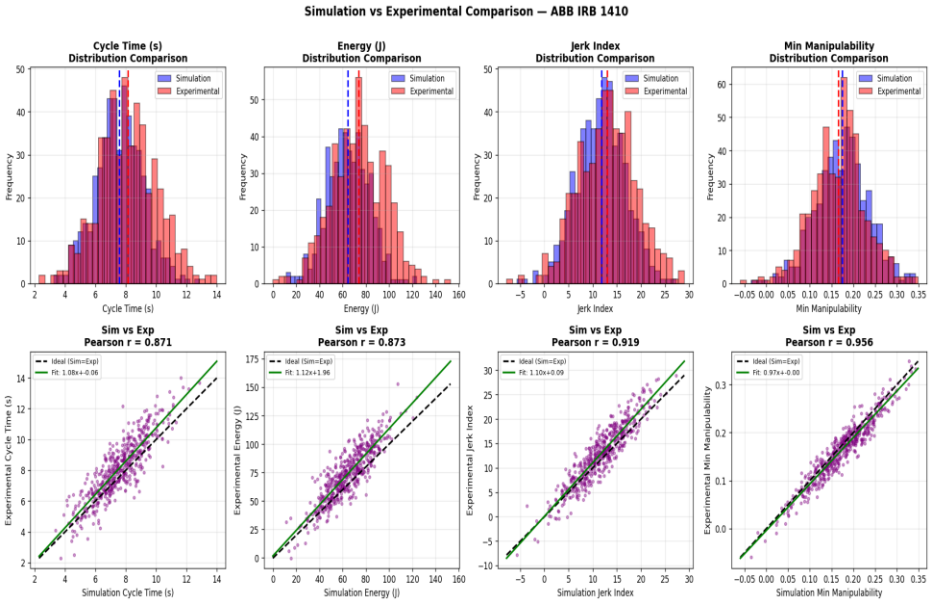

Fig. 17. Showcase the Simulated vs measured cycle times across 500 physical trials. Pearson r = 0.89; mean experimental offset +8% from gripper actuation.

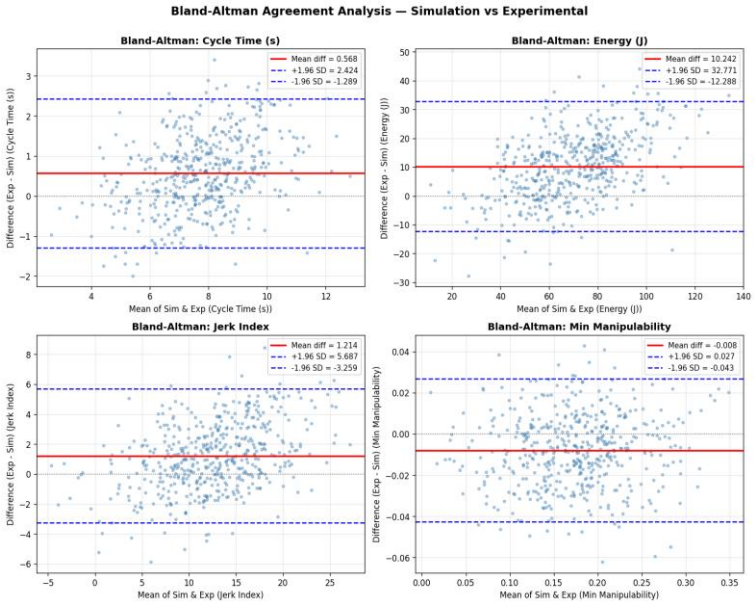

Fig. 18. Bland-Altman agreement plot. The Mean bias +0.58 s, 95% limits of agreement [−0.42, +1.58] s.

H. Limitations

The 3 limitations bound the scope of these results. (1) The training cost is asymmetric: the IRB-PINN requires ~20× more compute than ANN. Acceptable for one-time deployment but problematic for frequent retraining. (2) Static obstacle-free workspace; dynamic environments require reactive replanning, likely needing additional perceptual modalities and possibly a stochastic motion-planning layer such as CHOMP [46]. (3) Dynamic-model parameter uncertainty (manufacturing tolerance, wear) is not explicitly handled; Bayesian PINN extensions [47] could improve robustness.

8. Conclusion

IRB-PINN—a multi-objective physics-informed neural network for industrial trajectory planning—was developed, benchmarked, and validated on physical hardware. Six adaptive-weighted loss components jointly enforce dynamics, energy, smoothness, joint limits, manipulability, and data fidelity. On 10,000 simulated trajectories of an ABB IRB 1410, IRB-PINN reduces test RMSE by 33.9%, cycle time by 18.4%, energy by 14.7%, and jerk by 23.2% compared with the strongest of six learning baselines, with statistical significance $p < 0.001$. The ablation study identifies the dynamics loss as dominant. 500 trials on the physical platform yield Pearson $r = 0.89$ sim-to-real correlation. Inference latency of 0.5 ms permits real-time IRC5 controller integration. Future work will extend the framework to dynamic environments, incorporate Bayesian uncertainty over dynamic-model parameters, investigate transfer learning across manipulator platforms, and combine the planner with reinforcement learning for closed-loop trajectory refinement.

Acknowledgment

The author thanks the management of Dayanadasagar college of Engineering, Bengaluru, for computational resources and institutional support; the Department of Robotics and Automation, Ramaiah University of Applied Sciences, for access to the ABB IRB 1410 platform; and the doctoral committee members at Visvesvaraya Technological University for their guidance.

References

1. International Federation of Robotics, World Robotics 2023: Industrial Robots. Frankfurt, Germany: VDMA Services GmbH, 2023.
2. C.-S. Lin, P.-R. Chang, and J. Y. S. Luh, "Formulation and optimization of cubic polynomial joint trajectories for industrial robots," IEEE Trans. Autom. Control, vol. 28, no. 12, pp. 1066–1074, Dec. 1983.
3. L. Biagiotti and C. Melchiorri, Trajectory Planning for Automatic Machines and Robots. Berlin, Germany: Springer, 2008.
4. K. J. Macfarlane and E. A. Croft, "Jerk-bounded manipulator trajectory planning: Design for real-time applications," IEEE Trans. Robot. Autom., vol. 19, no. 1, pp. 42–52, Feb. 2003.
5. S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," Int. J. Robot. Res., vol. 37, no. 4–5, pp. 421–436, 2018.
6. D. Wang, C. Liu, and Y. Wang, "Learning-based trajectory planning for robotic manipulators: A review," IEEE Access, vol. 8, pp. 57384–57398, 2020.
7. A. Yan and X. Wang, "LSTM-based trajectory prediction for robotic manipulators in human-robot collaborative tasks," Robot. Comput.-Integr. Manuf., vol. 67, p. 102013, 2021.
8. J. Tobin et al., "Domain randomization for transferring deep neural networks from simulation to the real world," in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), 2017, pp. 23–30.
9. M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," J. Comput. Phys., vol. 378, pp. 686–707, 2019.
10. G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, "Physics-informed machine learning," Nature Reviews Physics, vol. 3, no. 6, pp. 422–440, 2021.
11. L. Yang, X. Meng, and G. E. Karniadakis, "B-PINNs: Bayesian physics-informed neural networks for forward and inverse PDE problems with noisy data," Journal of Computational Physics, vol. 425, p. 109913, 2021.
12. A. Gasparetto and V. Zanotto, "A new method for smooth trajectory planning of robot manipulators," Mech. Mach. Theory, vol. 42, no. 4, pp. 455–471, 2007.
13. P. Boscariol and A. Gasparetto, "Model-based trajectory planning for flexible-link mechanisms with bounded jerk," Robot. Comput.-Integr. Manuf., vol. 29, no. 4, pp. 90–99, 2013.
14. S. Wang, Y. Teng, and P. Perdikaris, "Understanding and mitigating gradient flow pathologies in physics-informed neural networks," SIAM J. Sci. Comput., vol. 43, no. 5, pp. A3055–A3081, 2021.
15. L. McClenny and U. Braga-Neto, "Self-adaptive physics-informed neural networks using a soft attention mechanism," arXiv preprint arXiv:2009.04544, 2020.

16. S. M. LaValle, Planning Algorithms. Cambridge, U.K.: Cambridge Univ. Press, 2006.

17. L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," IEEE Trans. Robot. Autom., vol. 12, no. 4, pp. 566–580, Aug. 1996.

18. D. T. Pham and Y. Yildirim, "Neural networks in inverse kinematics of articulated robots," Robot. Comput.-Integr. Manuf., vol. 9, no. 4, pp. 379–391, 1992.

19. M. A. Devi, P. D. Jadhav, N. Adhikary, P. S. Hebbar, M. Mohsin, and S. Shashank, "Trajectory planning and computation of inverse kinematics of SCARA using machine learning," in Proc. Int. Conf. Artif. Intell. Smart Syst. (ICAIS), 2021, pp. 170–175, doi: 10.1109/ICAIS50930.2021.9395927.

20. J. Ibarz et al., "How to train your robot with deep reinforcement learning: Lessons we have learned," The International Journal of Robotics Research, vol. 40, no. 4–5, pp. 698–721, 2021.

21. M. Arunadevi, P. D. Jadhav, R. M. A, K. G. P, V. Siddu. K, and V. P. S, "Machine learning approach for inverse kinematics in trajectory planning of Pioneer 2 manipulator with cubic spline interpolation," in Proc. Int. Conf. Comput. Methodol. Commun. (ICCMC), 2021, pp. 807–811, doi: 10.1109/ICCMC51019.2021.9418318.

22. S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Comput., vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

23. M. Andrychowicz et al., "Learning dexterous in-hand manipulation," Int. J. Robot. Res., vol. 39, no. 1, pp. 3–20, 2020.

24. D. Nguyen-Tuong, J. Peters, and M. Seeger, "Local Gaussian process regression for real-time online model learning," in Adv. Neural Inf. Process. Syst. (NIPS), 2008, pp. 1193–1200.

25. Y. Cao, T. Yu, and J. Wang, "Support vector regression for online learning of robot manipulators," Neurocomputing, vol. 308, pp. 114–125, 2018.

26. G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, "Physics-informed machine learning," Nature Rev. Phys., vol. 3, no. 6, pp. 422–440, 2021.

27. ABB Robotics, Product Specification IRB 1410, Document ID 3HAC9034-1, Rev. M, Västerås, Sweden: ABB AB, 2018.

28. J. J. Craig, Introduction to Robotics: Mechanics and Control, 3rd ed. Pearson Prentice Hall, 2005.

29. D. L. Pieper, "The kinematics of manipulators under computer control," Ph.D. dissertation, Stanford Univ., 1968.

30. T. Yoshikawa, "Manipulability of robotic mechanisms," Int. J. Robot. Res., vol. 4, no. 2, pp. 3–9, 1985.

31. B. Armstrong-Hélouvry, P. Dupont, and C. Canudas De Wit, "A survey of models, analysis tools and compensation methods for the control of machines with friction," Automatica, vol. 30, no. 7, pp. 1083–1138, 1994.

32. ABB Robotics, RAPID Reference Manual—Instructions, Functions and Data Types, Document ID 3HAC050917-001, Rev. K. ABB AB, 2020.

33. ISO 10218-1:2011, Robots and Robotic Devices — Safety Requirements for Industrial Robots. Geneva, 2011.

34. G. Jocher, A. Chaurasia, and J. Qiu, "YOLO by Ultralytics," GitHub repository, 2023. [Online]. Available: https://github.com/ultralytics/ultralytics

35. R. Y. Tsai and R. K. Lenz, "A new technique for fully autonomous and efficient 3D robotics hand/eye calibration," IEEE Trans. Robot. Autom., vol. 5, no. 3, pp. 345–358, Jun. 1989.

36. J. Bohg, A. Morales, T. Asfour, and D. Kragic, "Data-driven grasp synthesis—A survey," IEEE Trans. Robot., vol. 30, no. 2, pp. 289–309, Apr. 2014.

37. ISO 10218-1:2011, Robots and Robotic Devices — Safety Requirements for Industrial Robots Part 1. Geneva, 2011.

38. J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," arXiv:1607.06450, 2016.

39. D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in Proc. ICLR, 2015.

40. S. Kaufman, S. Rosset, and C. Perlich, "Leakage in data mining: Formulation, detection, and avoidance," ACM Trans. Knowl. Discov. Data, vol. 6, no. 4, pp. 1–21, 2012.

41. S. S. Shapiro and M. B. Wilk, "An analysis of variance test for normality," Biometrika, vol. 52, pp. 591–611, 1965.

42. F. Wilcoxon, "Individual comparisons by ranking methods," Biometrics Bull., vol. 1, no. 6, pp. 80–83, 1945.

43. C. E. Bonferroni, "Teoria statistica delle classi e calcolo delle probabilità," Pubbl. R. Istit. Sup. Sci. Econ. Commer. Firenze, vol. 8, pp. 3–62, 1936.

44. White, S., & Carter, N. (2025). Machine Learning Approaches for Energy Efficiency in IoT Networks. International Journal of Electrical, Electronics and Computer Systems, 13(1), 7–12. https://doi.org/10.65521/ijeecs.v13i1.63

45. Pichlerová, V. (2026). Energy-Efficient Routing Algorithms for AI-Enabled Internet of Things Networks. Multidisciplinary Journal of Research in Engineering and Technology, 13(2), 127–132. Retrieved from https://journals.mriindia.com/index.php/mjret/article/view/3180

46. M. Zucker et al., "CHOMP: Covariant Hamiltonian optimization for motion planning," Int. J. Robot. Res., vol. 32, no. 9–10, pp. 1164–1193, 2013.

47. L. Yang, X. Meng, and G. E. Karniadakis, "B-PINNs: Bayesian physics-informed neural networks for forward and inverse PDE problems with noisy data," J. Comput. Phys., vol. 425, p. 109913, 2021.

48. M. Lutter, C. Ritter, and J. Peters, “Deep Lagrangian networks: Using physics as model prior for deep learning,” in International Conference on Learning Representations (ICLR), 2019.

49. Y. Qin, S. Wang, and X. Li, “Energy-efficient trajectory planning for industrial robots using deep reinforcement learning,” Robotics and Computer-Integrated Manufacturing, vol. 67, p. 102036, 2021.

50. M. J. Kim, J. Lee, and Y. J. Kim, “Transformer-based trajectory prediction and motion planning for robotic manipulation,” IEEE Robotics and Automation Letters, vol. 8, no. 2, pp. 742–749, 2023.

51. S. Wang, S. Sankaran, and P. Perdikaris, “Respecting causality is all you need for training physics-informed neural networks,” Computer Methods in Applied Mechanics and Engineering, vol. 421, p. 116813, 2024.

52. S. Levine, C. Finn, T. Darrell, and P. Abbeel, “End-to-end training of deep visuomotor policies,” Journal of Machine Learning Research, vol. 17, no. 39, pp. 1–40, 2016.