IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

Informed Federated Learning to Train a Robotic Arm Inverse Dynamic Model

Gabriel Jimenez-Perera, Brayan Valencia-Vidal, Niceto R. Luque, Eduardo Ros and Francisco Barranco

***Abstract*—Access to real-world data in robotics domains is often challenging due to restrictions on data sharing and limited availability. Although privacy and intellectual property concerns are the main barriers, ensuring data access is crucial for advancing data-driven models. Specifically, machine-learning-based inverse dynamic models show promising results for nonrigid robot identification, but the data used to train them are often kept private due to intellectual property protections. Federated learning proposes a methodology to access such data without centralizing them in a single repository, thus avoiding intellectual property limitations. We propose a solution that uses federated learning to train a model from distributed data to develop a robust robotic arm inverse dynamic model. Our approach demonstrates the feasibility of using a machine learning method in which local robots train on their own data while collaborating without sharing raw information. Furthermore, we propose a novel custom aggregation method that integrates locally learned solutions from different workspaces into a single global model without requiring raw data sharing. This method improves accuracy in our federated solution by approximately 20% for the learned inverse dynamic model.**

***Index Terms*—Deep learning methods, Data sets for robot learning**

I. Introduction

AUTOMATION enables robots to perform repetitive, precise, and non-ergonomic tasks in the industrial sector under controlled conditions, such as structured environments. However, tasks requiring dexterity and adaptability in less-controlled industrial scenarios are still carried out by humans. With ongoing industrial advancements and under the umbrella of Industry 4.0, the demand for increasingly flexible industrial environments (driven by dynamic product markets) has led to workspaces shared with humans. To meet this demand and to ensure that robots do not pose a risk to the human workforce, collaborative robots, or cobots, have emerged. Consequently, cobots shall ensure safe physical human-robot interaction (pHRI), leading manufacturers to adopt specific design elements, such as Series Elastic Actuators (SEAs) [1] that enable compliance. However, compliance comes at a cost: while torque control shall be implemented, SEAs also increase dynamics complexity, i.e., backlash and frictional torque [2]. Also, they make mathematical modeling intractable [3], [4], [5], thus invalidating traditional [3], [6], [7], [8] model-based torque control methods [9], [10].

An alternative to traditional analytic dynamics modeling in cobots has recently emerged, using machine learning (ML) methods to develop data-driven dynamic cobot models. These ML methods have demonstrated effectiveness in capturing complex nonlinear dynamics, making them well suited to the problem at hand [11], [12], [13]. However, a key challenge in applying ML methods is obtaining high-quality data. Particularly, datasets relating joint torques to the resulting joint movements are required for dynamics modeling in cobots. Ideally, these datasets should comprehensively represent the entire cobot workspace, typically covered by performing localized manipulative tasks. Furthermore, the dataset must be both diverse and representative, including a wide range of motions with variations in velocity and acceleration. However, restricted access to industrial processes and data privacy protections (both common in this domain) render the availability of such real-world datasets needed for machine learning virtually non-existent.

There are established collaborative learning techniques that support intellectual property protection, such as transfer learning. However, this is not a suitable solution in this case due to the unavailability of pre-trained models relevant to the target domain. Recent advances in federated learning (FL) [14] enable model training directly on distributed local repositories, eliminating the need to centralize raw data. This approach contrasts traditional ML methods, which rely on the aggregation of data in a single location for training. In the literature, FL has been applied to problems such as vision-based obstacle avoidance [15], grasp control [16], and autonomous navigation [17], showing great potential for improving model performance and adaptability across heterogeneous environments [18]. However, no works have studied its application to learning dynamics for cobots yet.

Ours is an innovative solution that applies FL to learn the inverse dynamic model (IDM) of a Baxter® robotic arm in a distributed manner, ensuring that raw data remain private,

Manuscript received: April 15, 2025; Revised June 8, 2025; Accepted August 30, 2025.

This paper was recommended for publication by Aleksandra Faust upon evaluation of the Associate Editor and Reviewers' comments.

This work was supported by the European Union's PROACTIF project, which has received funding from the CHIPS Joint Undertaking (JU) under Grant 101194239. The JU receives support from the European Union's Horizon Europe Research and Innovation Program and Austria, Belgium, Finland, France, Germany, Hungary, Israel, Italy, Latvia, the Netherlands, Poland, Spain, and Switzerland. This work was also supported by Grant PCI2025-163223, funded by the MICIU/AEI/10.13039/501100011033 and co-funded by the European Union and by the Spanish National Grant PID2022-141466OB-I00 funded by MICIU/AEI/10.13039/501100011033 and by ERDF/EU. Gabriel Jimenez-Perera's research has been supported by "Consejería de Transformación Económica, Industria, Conocimiento y Universidades de la Junta de Andalucía" (Grant Ref. PREDOC_00280). Brayan Valencia-Vidal's research has been supported by the Colombian Ministry of Science, Technology, and Innovation by the Grant (call 860).

All authors are with the Department of Computer Engineering, Automation and Robotics, Research Centre for Information and Communication Technologies (CITIC-UGR), University of Granada, 18014 Granada, Spain gabrieljimenez@ugr.es

Brayan Valencia-Vidal is also with the Research Group Osiris & Bioaxis, Faculty of Engineering, El Bosque University, Bogotá, Colombia

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

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

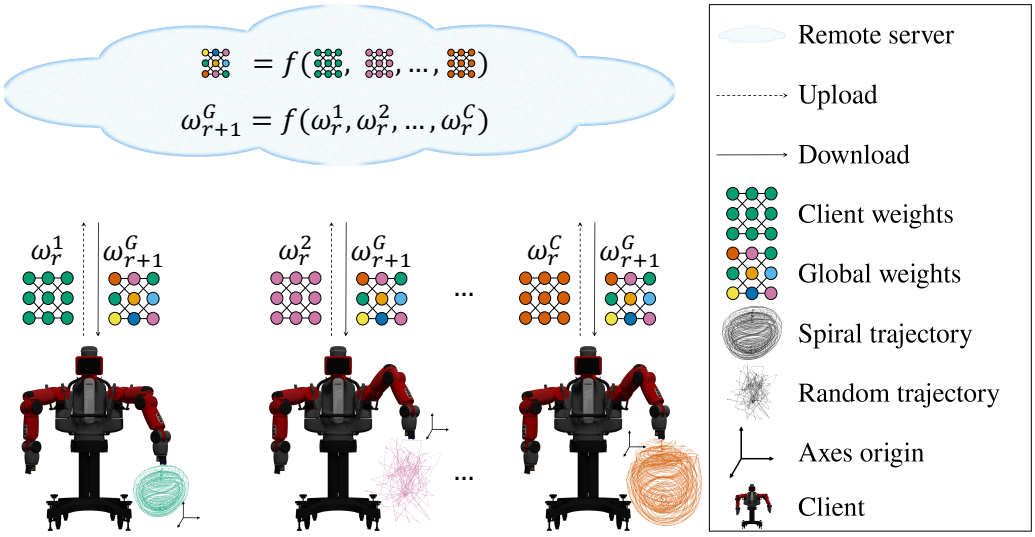

Fig. 1: Graphical abstract of our proposal. We represent different independent clients, each one of them following a trajectory in a specific workspace and recording torque data. Each client trains an IDM using its own data. They then upload their model parameters to the server, where the parameters of all selected clients are combined into the next global model parameters. These updated parameters are then redeployed to all clients to continue with the training process. This cycle repeats until convergence is reached.

i.e., that data are not shared among cobot clients. Each cobot client trains its local IDM using motion data only from its own workspace, optimizing its model parameters (also known as weights) for learning based on local experience. After each training cycle, only the parameters of the cobot client model are shared with a central server, aggregating them to refine a global IDM with better generalization capabilities. This updated model is then redeployed to the cobot clients for further local training. This cycle repeats iteratively, allowing the global IDM to improve accuracy while preserving data privacy and protecting intellectual property. After several iterations, the FL process reaches an optimized global IDM that generalizes across different workspaces and conditions, as illustrated in Figure 1.

To improve the FL learning process, we propose a variant called "informed federated learning", where cobot clients share additional metadata alongside model parameters while still protecting privacy and intellectual property. This metadata consists of the 3D coordinates of the bounding cube that encloses the local workspace from which motion data are used for learning, improving the convergence of the FL process and the accuracy of the global IDM.

The main contributions of this work are as follows:

- We gather a dataset using the methodology described in [19]. This dataset is publicly available and contains 4 different trajectories (two for training and two for testing) in six different workspace areas, and was collected on both arms of the same cobot.
- To the best of our knowledge, this is one of the first solutions in the literature that uses federated learning for modeling inverse dynamics. We demonstrate the learned IDM with experiments using a varying number of cobot clients, without sharing raw data.
- Finally, we propose "FedCubeNQ", a novel FL aggregation method that incorporates "bounding cubes", 3D volumes that enclose the local workspace of the cobot. This aggregation method reduces the accuracy errors in our federated IDM by approximately 20%, overcoming other state-of-the-art techniques.

II. Related Work

The state-of-the-art performance of ML methodologies in inverse dynamics modeling has led to the adoption of data-driven solutions. For instance, Rückert et al. [20] used a long short-term memory (LSTM) network, achieving better predictive performance and scalability with large datasets. Similarly, Liu et al. [21] reported that bidirectional recurrent neural networks (BRNNs) outperform nonrecurrent architectures for inverse dynamics identification, particularly in time series problems with a moderate number of inputs. Furthermore, Xiao et al. [22] proposed optimizing an LSTM network by particle swarm optimization to predict joint torques of industrial robots, indicating the growing impact of data-driven methodologies in dynamic modeling. Nevertheless, these modeling examples are primarily focused on identifying rigid-body dynamics, and do not account for the non-linear dynamics inherent in cobots. As a response, Valencia et al. [19] proposed a data-driven IDM for cobots, using a BRNN based on Gated Recurrent Units (GRUs) that effectively captured those non-linear dynamics.

Modeling a cobot inverse dynamics using a data-driven solution requires access to datasets that relate torque values to the resulting joint movements. However, as aforementioned, such purpose-driven datasets may not always be readily available. Nonetheless, cobots operating in industrial assembly lines are ubiquitous by definition; hence, why not use them for data

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

collection? FL [14] offers a solution to this challenge by allowing decentralized model training using multiple cobots. In FL, individual clients train a local model using their own data, while a central server orchestrates the training process and aggregates local model updates into a global model.

One of the main issues in FL is statistical heterogeneity, that is, data are not independent and identically distributed (non-IID data) across the network of clients. To address this, there are established non-IID FL methods, like FedProx [23]. In FedProx, minor modifications to the state-of-the-art method for FL (FedAvg, explained below) improve learning convergence. In robotics, FL has been successfully used in a variety of domains. For instance, in [17], FL has been applied in a vision-based autonomous robot navigation environment, using clustering to improve adaptability in different environments while significantly reducing communication bandwidth and matching the performance of centralized learning. Similarly, in [24], FL has also been applied to multi-drone systems for action recognition and control, allowing distributed decision making without requiring a central repository. Beyond autonomous control, FL has also been used in robotics to safeguard data privacy in human-robot collaboration. A FL-based robot navigation recommender system with humans in the loop was proposed in [25], in which privacy for both, agents' data and their decisions based on environmental perception was preserved. FL has also been applied in [26] to predict human intention in collaborative assembly tasks. This method protects sensitive information in assembly tasks, which means it also protects intellectual property. Finally, custom aggregations have been proposed for purpose-specific methods as in [27], where an FL-based controller for swarms with a custom aggregation function reduces the communication bandwidth while achieving high robustness and generalization.

III. Proposed Approach

The inverse dynamics formulation determines the torque values required to produce a movement of the robot. In ML terms, this problem is formulated as a regression task [28], which we solve using a BRNN as proposed by [19].

To train our BRNN-based IDM with FL, we use a central server and data from different local cobot clients. In each FL round, a subset of cobot clients is selected and initialized with the parameters provided by the server. These clients train their local inverse dynamic models using only local data. The updated model parameters are then sent to the server, which aggregates them into a new global model that is redistributed. This iterative process is continuous until the global model converges, as formalized in (1)

$$\omega_{r+1}^{G} = f(\omega_r^1, \omega_r^2, ...\omega_r^C) \tag{1}$$

where r is the round index, ω_{r+1}^{G} are the global parameters at the end of the round r, these parameters are broadcast as initial parameters at the beginning of the round $r+1$ and will be the final parameters when the final round is reached; ω_r^i are the parameters of the local cobot client i generated in the round r, and C is the number of clients selected in each round.

The FedAvg aggregation function (implemented in FLWR [29]) is one of the most widely used methods in FL. It computes a weighted average, where each client contributes proportionally to its number of training samples, as shown in (2).

$$\omega_{r+1}^{G_{FedAvg}} = \frac{1}{Q}\sum_{c=1}^{C} Q^c \omega_r^c \tag{2}$$

where $\omega_{r+1}^{G_{FedAvg}}$ represents the global parameters generated after the round r, C is the number of cobot clients selected in the round r, Q^c is the number of samples from client c in the round r used in training, ω_r^c are the local parameters generated by client c in the round r, and Q is the number of training samples used in the round r ($Q = \sum_{c=1}^{C} Q^c$).

Our approach uses an alternative aggregation method, "FedCubeNQ", in which each trajectory is enclosed within a predefined workspace, which allows the central server to aggregate knowledge from different clients more effectively by incorporating this workspace spatial descriptor. FedCubeNQ allows each client to transmit both the updated model parameters and the bounding cube that encloses the trajectory. This bounding cube is the minimal axis-aligned cubic volume that fully contains a given trajectory within a cobot workspace. This does not pose a problem regarding intellectual property or data privacy as it does not describe the trajectory itself; it only states the region where it occurs. By transmitting this bounding cube along with the updated model parameters, FedCubeNQ enables informed aggregation using spatial constraints to improve distributed learning.

The bounding cube is computed by extracting the minimum and maximum Cartesian coordinates of all trajectory points, defining two diagonally opposed vertices: $(C_1(x_{min}, y_{min}, z_{min}), C_2(x_{max}, y_{max}, z_{max}))$. The bounding cube is an axis-aligned cubic volume defined by these two extreme corners. This cube fully encloses the trajectory, with its faces parallel to the coordinate axes. Its dimensions (width, height and depth) correspond to the differences between the maximum and minimum Cartesian coordinates along each axis, as illustrated in Figure 2. This allows for an easy computation of the overlapping cube. Given two cubes (A, B) described by descriptors (A_{C_1}, A_{C_2}) and (B_{C_1}, B_{C_2}) respectively, we can compute the resulting overlapping cube R with descriptors (R_{C_1}, R_{C_2}) in (3).

$$\begin{aligned} R_{C_1} &= max(A_{C_1}, B_{C_1}) \\ R_{C_2} &= min(A_{C_2}, B_{C_2}) \end{aligned} \tag{3}$$

where min and max are functions that return the min and max element-wise values respectively.

The intuition behind this idea is that the trajectories performed in different regions of the full robot workspace provide complementary information for the learning process. By using his bounding cube, the server identifies and prioritizes non-overlapping trajectories, ensuring a more diverse and spatially balanced model update. This reduces the impact of clients whose models may not generalize well due to working in spatially constrained regions of the workspace, thus mitigating overfitting while preserving the federated paradigm. Unlike FedAvg, FedCubeNQ does not weigh clients based on the number of training samples (hence the NQ suffix, no quantity)

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

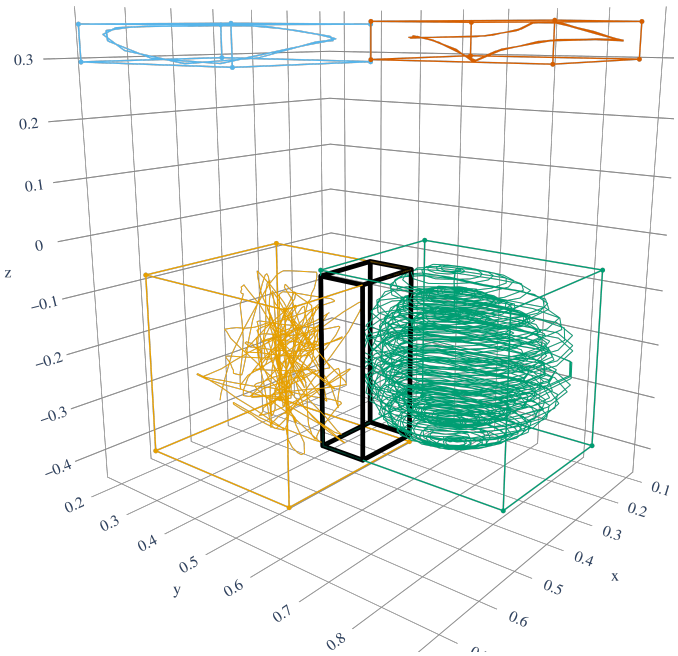

Fig. 2: Four trajectories and an intersection. In **blue** and **vermillion**, two trajectories and their bounding cubes without overlap. In **orange** and **green**, two trajectories and their intersecting bounding cubes. In **black**, the intersecting cube between the **orange** and the **green** bounding cubes.

since our experiments showed no benefits from doing so. The FedCubeNQ aggregation function is described in (4)

$$\omega_{r+1}^{G_{FedCubeNQ}} = \frac{1}{T_n} \sum_{c=1}^{C} T_n^c \omega_r^c \tag{4}$$

where

$$T_n^c = norm\left(\frac{-\sum_{i=0}^{I} v(X_i)}{v(T^c)}\right)$$

Here, v denotes the function that calculates the volume of a cube, X_i represents an overlapping cube, T^c is the trajectory descriptor from client c, T_n^c is the calculated weight for client c, T_n is the sum of all calculated weights ($T_n = \sum_{c=1}^{C} T_n^c$), and $norm$ is the function used to normalize the data from the original range $(-\infty, 0]$ to the interval $[0.1, 1]$. The minimal contribution provided by a robot client is set to 0.1, ensuring that each trajectory contributes even if its bounding cube is already occupied by another trajectory. Since the set of clients varies in each aggregation process, T_n^c shall be recalculated for every round.

IV. Our Setup

In this section, we present our experimental setup. We begin by explaining the process of creating our dataset for a Baxter® robotic arm. We use this dataset to conduct our experiments in a later section. We will also introduce the performance metrics used for our evaluation. These metrics are widely used in the literature, allowing direct comparison with existing approaches.

A. Our Dataset

A key requirement for training a data-driven IDM is to have high-quality data. As mentioned, no publicly available datasets exist for our specific problem. Therefore, we collected our own dataset following the methodology described in [19]. To ensure robust training, we require a sufficiently large dataset gathered from different cobot clients operating in partially overlapping workspaces. All data were collected using the same robot (right and left arms), with each limb maintained under consistent conditions to eliminate external variability, though maintenance procedures may differ between arms. The dataset consists of 4 different motion types performed in 6 different workspaces, where torque-to-joint position/velocity relationships were recorded at 500 Hz, resulting in approximately 1.8 million samples (80% for training and 20% for testing) for each limb.

To record our dataset, we have used the Baxter® cobot, which has two arms. Each arm has seven degrees of freedom and SEAs. This means that there is a spring between the motor gear and the actuator output. There is also a passive spring in one of the joints, which further increases the complexity of the robotic arm dynamics.

Each explored workspace (W_e) is a subset of the robot complete reachable workspace (W_T) ($W_e \subset W_T$). Specifically, each W_e is defined as a sphere with a radius $r_w = 0.18$ m and a center C_e. To simplify the definition of the subspaces, we use spherical coordinates with the origin aligned with the origin of the Cartesian coordinate system from the robot arm base. Therefore, the coordinates of the center C_e in the spherical system are $(r_c, \theta, \phi)_S$. Here, the value of r_c establishes the radial distance of the center C_e from the robot base origin. The angles θ and ϕ determine the angular position of the center C_e in 3D space. A change in θ (azimuthal angle) horizontally displaces the W_e subspace around the robot base Z-axis, varying its position in the XY-plane. Meanwhile, a change in ϕ (elevation angle) adjusts the height of the W_e subspace relative to the XY-plane, moving it vertically. This spherical definition of the subspace allows for flexible specification of regions for robot operation.

The specific explored subspaces were defined by the following parameters for the center of each sphere:

$$\begin{aligned} &\forall W_e : r_c = 0.535 \text{ m} \\ &\quad \wedge \\ &\{(\theta, \phi)\} = \quad \{(\text{-}15°, 105°), (15°, 70°), (15°, 140°), \\ &\qquad\qquad (60°, 70°), (60°, 140°), (90°, 105°)\} \end{aligned}$$

Each workspace includes two trajectories for training and two trajectories for testing. All trajectories are defined with a frame whose origin coincides with the center of the explored workspace W_e

- **Training data:**
  - **Random** — A motion type that generates random movements within the workspace W_e, shown in **orange** in Figure 2. This type of motion has a richness in velocity and acceleration variations.
  - **Spiral** — A motion type that follows a sphere-like movement, shown in **green** in Figure 2. It features

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

smooth transitions and predictable patterns, defined by the following parametric equations:

$$\phi_s = 8\pi f_s t$$
$$\theta_s = 1.6\pi t$$
$$r_s = r_w - 0.08 f_s t$$

with $f_s = [2.5 * 10^{-2}, 3.3 * 10^{-2}, 5 * 10^{-2}]$

- **Test data:**
  - **Squared** — A motion type that follows a square-like movement, shown in **vermillion** in Figure 2. It consists of continuous movements along the edges with abrupt velocity and acceleration changes at vertices. Each side of the square had a length of 0.2 m and the average speed to traverse each side was 0.27 m/s.
  - **Circle** — A motion type that follows a circular movement, shown in **blue** in Figure 2. Similarly to the spiral trajectory, it features smooth transitions and predictable movement patterns, defined in Cartesian coordinates as:

$$x = r_w \sin(\pi t)$$
$$y = r_w \cos(\pi t)$$
$$z = c_{ez} = r_w \cos(\phi)$$

For each recorded trajectory, the corresponding joint coordinates were computed using the robot inverse kinematics. A controller was tuned to ensure accurate tracking of these trajectories. During the data acquisition process, we record joint positions, velocities, and the commanded torque values. We downsampled these data recordings from 500 Hz to 100 Hz to provide our IDM with real-time capabilities. Each sample consists of 21 numerical values: 7 angular positions, 7 angular velocities, and 7 applied torques.

For each spiral and random trajectory, a randomly selected segment equivalent to 20% of the total data points was designated for validation. The remaining 80% of the trajectory was used for training. Temporal windows with a duration of 250 ms were employed for data processing. A data sample S_i is defined as $[x_{i-T/2}, x_{i-T/2+k\Delta t}, \ldots, x_{i+T/2}]$, where $k = 0, 1, \ldots, \lfloor T/\Delta t \rfloor$, Δt is the sampling interval, and T is the window duration. The feature vector for each sample comprises the joint positions and velocities within the window, and the target output is the vector of commanded torques at the center of the temporal window. This dataset can be found at https://doi.org/10.5281/zenodo.17035193.

B. Performance Metrics

We use the mean absolute error (MAE) metric in our experiments. The MAE is computed as the average of the absolute difference between the actual and predicted torque values at each joint, as shown in (5), where N denotes the number of evaluated time instants.

$$MAE_j = \frac{1}{N} \sum_{t=1}^{T} |\tau_{t,j} - \hat{\tau}_{t,j}| \tag{5}$$

We also used the coefficient of determination, r^2, which is defined as

$$r^2 = 1 - \frac{\sum_{t=1}^{T} \sum_{j=1}^{J} (\tau_{t,j} - \hat{\tau}_{t,j})^2}{\sum_{t=1}^{T} \sum_{j=1}^{J} (\tau_{t,j} - \bar{\tau})^2} \tag{6}$$

where

$$\bar{\tau} = \frac{1}{TJ} \sum_{t=1}^{T} \sum_{j=1}^{J} \tau_{t,j}$$

r^2 is a single value $\in [0, 1]$, where 1 means that model predictions are perfectly aligned with the ground truth. It is used to evaluate and compare the overall performance of a certain model.

V. Experimental Evaluation

In this section, we present the process of modeling an IDM for a Baxter® robotic arm, our robotic platform demonstrator. We begin by exploring the applicability of our method in industrial scenarios where robots are specialized in particular tasks. These experiments are designed to showcase the potential of FL, and our aggregation strategy in particular, in such contexts. In the spatial-aware experiments V-B, we begin by explaining how we use our dataset, followed by an explanation of our experiments and their results, highlighting the importance of FL in this context. We continue by introducing our aggregation method and demonstrate its robustness. We compare our proposed method against two FL strategies: FedAvg, a general-purpose aggregation baseline method, and FedProx (with $\mu = 0.25$ the value that yielded the best results in our tests), which is designed to address data heterogeneity in non-IID scenarios. Additionally, we include centralized learning results as baseline references. These results are not intended for direct comparison with FL methods, but rather to provide context on the maximum achievable performance under full data sharing, an assumption often unrealistic in practical robotic applications. In order to simulate a more realistic scenario with a larger number of clients, we divided the training trajectories where $\theta = 70$. Each random trajectory was split into two segments, and each spiral trajectory into three. These segments are treated as independent trajectories and replace the original full-length ones. Finally, we validate the robustness of our approach through additional experiments simulating real-world deployment challenges. These include using the Baxter's right limb, changing the client pool size and online learning. All results are the mean MAE over 30 runs of each experiment. The code used for these experiments can be found at https://github.com/GabriJP/ifl_baxter.git. All data about our experiments can be found on W&B at https://api.wandb.ai/links/gabijp/1sjqlvl2.

A. FL for Robotic Tasks

In realistic scenarios, robots are often specialized in specific tasks. To address this, we conducted two experiments comparing our method to centralized learning in previously unseen task scenarios. These scenarios were inspired by [26].

In the first scenario, a robot trained to perform circular welding motions is evaluated on square welding motions.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

TABLE I: MAE results of our proposal in different challenging tasks

Challenging tasks	Centralized	FL
Square welding	0.983	0.424
Spot welding	0.381	0.31

In the second, a robot trained on continuous circular and square welding motions is evaluated on spot welding. For the centralized baseline solution, we trained an IDM using circular trajectories and then evaluated it on square trajectories for the first scenario. For the second scenario, we trained our IDM using both circular and square trajectories while evaluating on random trajectories. In the FL setup, clients receive the aggregated knowledge gathered by other clients using different trajectories. Test data is shared across both versions. We do not segment the data by workspace in these experiments; all motion types are used across all available workspaces. The results are reported in Table I. Our method consistently outperforms the centralized baseline in these challenging generalization tasks, despite not using workspace-based partitioning.

B. Spatial-Aware Experiments

As a step further, our research focuses on the observation that IDMs trained on data from a specific subregion of the workspace do not generalize well outside that region and may even produce physically inconsistent predictions [30].

First, we divide our data into a training split, i.e., including training and validation sets, and a test split. The trajectory motions used in training (spiral and random) differ from those used in testing (square and circle). The test split is replicated across all cobot clients and is used consistently in all our experiments. In centralized learning, all trajectories would be stored in the same data silo, allowing the training loop to access all training data. In contrast, in the federated setup, each cobot client trains exclusively on a single trajectory from a single workspace. Since the test split is replicated across clients, the server requests the test results solely from a single client. For each experiment, we extract a validation set (20%) from the training split.

Regarding the data, in the *all* condition, we use training data from all workspaces but only test data from the workspace with $\theta = 70$. This experiment represents ideal conditions where the data are representative of the environment (the training split includes trajectories from the workspace with $\theta = 70$). To summarize, the cobot client pool used in this experiment contains 18 clients (4 full-length and 4 segmented random trajectories; 4 full-length and 6 segmented spiral trajectories). Every round, 4 randomly sampled clients are selected for training.

However, in real-world conditions, cobot dynamics is heavily influenced by the environment. This means that a robot client is likely to encounter unfamiliar environmental conditions. Furthermore, accessing the necessary data in this environment is challenging due to intellectual property protections, as it could reveal underlying production processes. This is another key reason for the use of federated learning.

To demonstrate the benefit of the federated setup, we performed two additional experiments, namely, *o70* and *n70*. In our experiments, the test data remained consistent with the initial case, testing with squares and circles performed exclusively in areas where $\theta = 70$. Regarding training:

- In *o70*, we trained exclusively with data from workspaces where $\theta = 70$, simulating an ideal scenario where a cobot client is trained in a specific region and operates solely within those boundaries. The cobot client pool contains a total of 10 clients (4 segmented random trajectories and 6 segmented spiral trajectories), of which 4 randomly sampled clients are selected for training every round.
- In *n70*, we train with data from workspaces where $\theta \neq 70$, representing a cobot operating under entirely new conditions. Under the assumption that data are not shared and remain in the local original silo, this experiment demonstrates that FL successfully generalizes. The cobot client pool contains a total of 8 clients (4 full-length random trajectories and 4 full-length spiral trajectories), of which 4 randomly sampled clients are selected for training every round.

As shown in Table II, the errors for *o70* are the lowest, representing the best-case scenario where the cobot client is trained with data from the same workspaces as the test data. For the FL approach (using the FedAvg aggregation method), the error is approximately 0.06 Nm higher because learning is performed by aggregating models from different cobot clients that locally learn in these workspaces where $\theta = 70$. In contrast, the centralized approach learns from a single silo containing all the data from these workspaces.

Regarding *n70*, this experiment is designed to demonstrate the contribution of federated knowledge to the learning process in scenarios where data silos are independent and no training data from test workspaces are available. As expected, our experiments confirm that the premise from [30] that we use for this experiment also applies to the FL setting. However, we can expect the FL to learn from more spaces than the centralized version, potentially even from the most challenging regions for this test. In this setup, the federated approach accesses the learned models from various workspaces (*all*), which reduces the MAE by approximately 15%. This demonstrates the effectiveness of FL in enhancing IDM performance without the need of sharing raw data (Centralized *n70* vs. FedAvg *all*).

Nevertheless, in the FL setup, prediction errors are nearly twice as high as those estimated in our optimal scenario. Given the importance of learning from diverse spaces, we propose a new FL aggregation method that uses a simple spatial descriptor, the bounding cube, which encloses the workspace area of each trajectory. When comparing FedAvg and our proposed method, FedCubeNQ, each FedCubeNQ training round assigns a higher weight to the parameters of robot clients whose workspaces have minimal overlap. This approach enhances model generalization. We also observed that in the most general scenario (*all*), our FedCubeNQ method achieves nearly 20% lower error compared to FedAvg. Although FedAvg uses the number of samples per client to compute a weighted average during aggregation, we obtained better results by

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

TABLE II: MAE and r^2 mean values over all joints and runs. The best FL values are highlighted in **bold**.

	MAE (lower is better)				r^2 (higher is better)			
	Centralized	FedAvg	FedProx	FedCubeNQ	Centralized	FedAvg	FedProx	FedCubeNQ
all	0.264	0.528	0.530	**0.428**	0.996	0.986	0.986	**0.991**
n70	0.628	0.780	0.839	**0.765**	0.979	0.966	0.963	**0.970**
o70	0.292	0.355	0.360	**0.353**	0.995	0.993	0.993	**0.994**

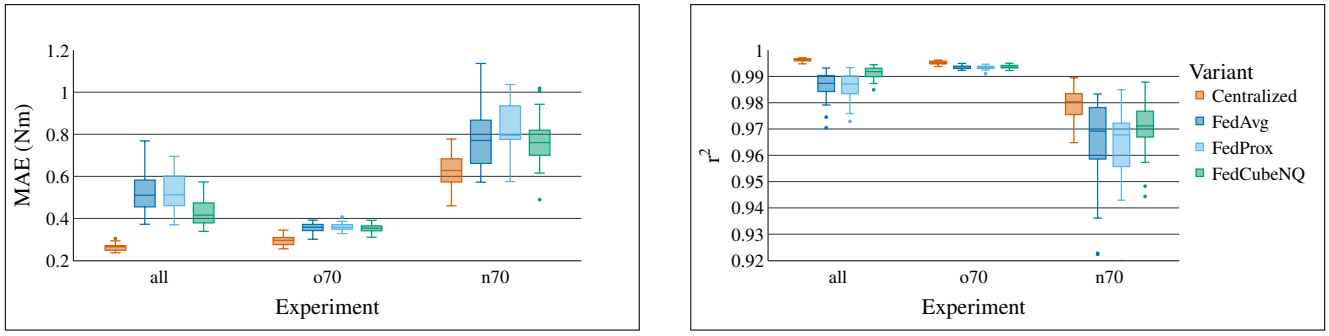

Fig. 3: Box plots comparing MAE (left) and r^2 (right) values between our experiments and their variants over 30 trials. Our proposal achieves the lowest error among the FL alternatives. Centralized learning is included as a reference, representing the lower bound for the learning process.

TABLE III: MAE results of robustness and online learning

Limb	FedCubeNQ
Left	0.428
Right	0.451

FedCubeNQ number of clients			
4	6	9	18
0.398	0.366	0.454	0.428

Online learning	
FedAvg	FedCubeNQ
0.526	0.422

disregarding this criterion and instead weighing cobot client parameters based only on the spatial overlap of their working areas.

To ensure statistical significance, we conduct 30 independent runs for each experiment and its variants with all reported values in the tables representing the average over these 30 runs. We summarized all the results in Figure 3 using the box-and-whisker plot for every experiment and its variants, demonstrating that, in the more realistic scenario (*all*), our aggregation method achieves significantly better results. For the sake of completeness, we show all our results for these experiments in Table II.

Finally, we also conducted a one-way ANOVA test to ensure that there is a statistically significant difference in the obtained MAE. The test was conducted between the FedAvg and FedCubeNQ aggregation methods (the two FL methods with best results) in the experiment *all*, obtaining a result of $(F(1, 58) = 21.198, p = 2.3 * 10^{-5})$.

C. Experiments for Robustness and Deployment

We believe that the rationale for using Federated Learning (FL) and the benefits of our proposed method have been demonstrated in the previous subsection. Nevertheless, further experimentation in more challenging environments is necessary to fully validate the robustness of our approach.

1) Cross-Platform Evaluation: All data used in this study were initially collected from the left arm of the Baxter robotic platform. To evaluate the generalizability of our method across platforms, we repeated data collection using its right arm. As discussed in Section I, using the right instead of the left arm effectively results in a different robotic platform due to differences in calibration, hardware aging, or mirrored kinematic structure that affect the resulting dynamics. As shown in Table III (left), our method maintains consistent performance for both arms, further supporting its robustness. In this case, we have not segmented any trajectory.

2) Scalability Test: To evaluate the scalability of our method across different client pool sizes, we conducted a dedicated experiment. Specifically, we replicated the *all* experiment while varying the client pool size to values of $4, 6, 9$ respectively. Due to the smaller pool sizes available, we selected only 3 clients to participate in each training round. In Table III, (center) we present the results of this experiment and those from the spatial experiment (with a pool size of 18 and selecting 4 clients per round) for comparison. The results show that our FL method maintains competitive and robust performance across varying client availability.

3) Online Learning for Real-World Deployment: Unlike our offline, simulated setup, real-world online learning results in local datasets that grow over time as more data becomes available. To evaluate the adaptability of our method in such settings, we designed an experiment comparing our strategy to the standard FedAvg approach. From the spatial experiment V-B, we observed that each client was selected for training at least 474 times. In this new experiment, we divided each client's dataset into 475 sequential segments. Each time a client was selected, it trained on all previously used segments plus one additional data segment. For example, during the first training request, the client uses only the first segment; in the second, it uses the first two segments, and so on. Once all segments have been used, the entire dataset is employed in subsequent rounds (this occurs only in the final rounds, given

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

our setup and hyperparameters). Table III (right) presents the results using this progressive data exposure. As the results indicate, the performance remains stable, showing minimal variability compared to the spatial experiment.

VI. Conclusion

The development of new machine learning methods relies on access to sufficient high-quality data. In our case, we emphasize the importance of covering the entire workspace of our robot. Furthermore, sensitive data, which must be protected due to privacy or intellectual property concerns, is even more difficult to find freely available. This motivated the creation of our own dataset and the adoption of FL.

FL enables training ML models without centralizing data, preserving data ownership and industrial confidentiality. In this work, we have shown that it is possible to train an inverse dynamic model from independent data silos using different trajectories from mostly non-overlapping workspaces, with a collaborative FL setup. Our proposal, FedCubeNQ, uses spatial descriptors, the bounding cubes that enclose the workspace area of each trajectory, to guide the aggregation process. This simple yet effective metadata has led to a 20% improvement in the performance of the IDM compared to baseline FL methods, while protecting intellectual property.

To validate the robustness of our approach, we further conducted three complementary experiments: a cross-platform evaluation using the opposite arm of the robot, confirming generalization to different hardware; a scalability test with varying number of cobot clients, showing consistent performance across settings; and an online learning scenario, simulating real-world incremental data collection, in which our method showed stable performance as well. These results reinforce the feasibility of our solution for real deployment and collaborative training in robotics.

References

[1] J. Hurst and K. Green, "Series Elastic Actuation," in *Encyclopedia of Robotics*. Berlin, Germany: Springer, 2020, pp. 1–12.

[2] E. Madsen, O. Rosenlund, D. Brandt, and X. Zhang, "Comprehensive modeling and identification of nonlinear joint dynamics for collaborative industrial robot manipulators," *Control Engineering Practice*, vol. 101, 2020.

[3] C. Lee, S. Kwak, J. Kwak, and S. Oh, "Generalization of Series Elastic Actuator configurations and dynamic behavior comparison," *Actuators*, vol. 6, no. 3, 2017.

[4] I. Abadía, F. Naveros, E. Ros, R. R. Carrillo, and N. R. Luque, "A cerebellar-based solution to the nondeterministic time delay problem in robotic control," *Sci. Robot.*, vol. 6, no. 58, Sep. 2021.

[5] I. Abadía, A. Bruel, G. Courtine, A. J. Ijspeert, E. Ros, and N. R. Luque, "A neuromechanics solution for adjustable robot compliance and accuracy," *Sci. Robot.*, vol. 10, no. 98, Jan. 2025.

[6] J. Luh, M. Walker, and R. Paul, "On-line computational scheme for mechanical manipulators," *J. Dynamic Syst., Meas. Control, Trans.*, vol. 102, no. 2, pp. 69–76, 1980.

[7] A. Ata, S. Elkhoga, M. Shalaby, and S. Asfour, "Causal inverse dynamics of a flexible hub-arm system through Liapunov's second method," *Robotica*, vol. 14, no. 4, pp. 381–389, 1996.

[8] A. Calanca, L. Capisani, A. Ferrara, and L. Magnani, "MIMO closed loop identification of an industrial robot," *IEEE Trans. Control Syst. Technol.*, vol. 19, no. 5, pp. 1214–1224, Sep. 2011.

[9] D. Mukherjee, K. Gupta, L. H. Chang, and H. Najjaran, "A Survey of Robot Learning Strategies for Human-Robot Collaboration in Industrial Settings," *Robot. Comput.- Integr. Manuf.*, vol. 73, p. 102231, Feb. 2022.

[10] Z. M. Bi, C. Luo, Z. Miao, B. Zhang, W. J. Zhang, and L. Wang, "Safety assurance mechanisms of collaborative robotic systems in manufacturing," *Robot. Comput. Integr. Manuf.*, vol. 67, 2021.

[11] B. Xu, Z. Shi, C. Yang, and F. Sun, "Composite neural dynamic surface control of a class of uncertain nonlinear systems in strict-feedback form," *IEEE Trans. Cybern.*, vol. 44, no. 12, pp. 2626–2634, Dec. 2014.

[12] C. Liu, G. Wen, Z. Zhao, and R. Sedaghati, "Neural-Network-Based Sliding-Mode Control of an Uncertain Robot Using Dynamic Model Approximated Switching Gain," *IEEE Trans. Cybern.*, vol. 51, no. 5, pp. 2339–2346, May 2021.

[13] Z. Li and S. Li, "Neural Network Model-Based Control for Manipulator: An Autoencoder Perspective," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 6, pp. 2854–2868, Jun. 2023.

[14] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Artif. Intell. Statist.*, 2017, pp. 1273–1282.

[15] X. Yu, J. P. Queralta, and T. Westerlund, "Federated Learning for Vision-based Obstacle Avoidance in the Internet of Robotic Things," in *2022 Seventh International Conference on Fog and Mobile Edge Computing (FMEC)*, 2022, pp. 49–54.

[16] Y. Wang, S. Zhong, and T. Yuan, "Grasp Control Method for Robotic Manipulator Based on Federated Reinforcement Learning," in *Proc. 7th Int. Conf. Adv. Algorithms Control Eng.*, 2024, pp. 1513–1519.

[17] S. Gummadi, M. V. Gasparino, D. Vasisht, and G. Chowdhary, "Fed-EC: Bandwidth-Efficient Clustering-Based Federated Learning for Autonomous Visual Robot Navigation," *IEEE Robot. Automat. Lett.*, vol. 9, no. 12, pp. 11 841–11 848, Dec. 2024.

[18] J. Weber, M. Gurtner, A. Lobe, A. Trachte, and A. Kugi, "Combining Federated Learning and Control: A Survey," *IET Control Theory Appl.*, vol. 18, no. 18, pp. 2503–2523, Dec. 2024.

[19] B. Valencia-Vidal, E. Ros, I. Abadía, and N. R. Luque, "Bidirectional recurrent learning of inverse dynamic models for robots with elastic joints: a real-time real-world implementation," *Front. Neurorobot.*, vol. 17, Jun. 2023.

[20] E. Rueckert, M. Nakatenus, S. Tosatto, and J. Peters, "Learning inverse dynamics models in O(n) time with LSTM networks," *IEEE-RAS 17th Int. Conf. Humanoid Robot.*, pp. 811–816, 2017.

[21] N. Liu, L. Li, B. Hao, L. Yang, T. Hu, T. Xue, and S. Wang, "Modeling and Simulation of Robot Inverse Dynamics Using LSTM-Based Deep Learning Algorithm for Smart Cities and Factories," *IEEE Access*, vol. 7, pp. 173 989–173 998, 2019.

[22] W. Xiao, Z. Fu, S. Wang, and X. Chen, "Joint torque prediction of industrial robots based on PSO-LSTM deep learning," *Ind. Robot*, vol. 51, pp. 501–510, 2024.

[23] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated Optimization in Heterogeneous Networks," 2020, arXiv:1812.06127.

[24] L. N. Nobrega, E. de Oliveira, M. Saska, and T. Nascimento, "Proximal Control of UAVs With Federated Learning for Human-Robot Collaborative Domains," *IEEE Robot. Automat. Lett.*, vol. 9, no. 12, pp. 11 305–11 312, Dec. 2024.

[25] M. Moradi, M. Moradi, and D. C. Guastella, "Experience Sharing and Human-in-the-Loop Optimization for Federated Robot Navigation Recommendation," in *Proc. Image Anal. Process. - ICIAP 2023 Workshops, PT II*, vol. 14366, 2024, pp. 179–188.

[26] J. Cai, Z. Gao, Y. Guo, B. Wibranek, and S. Li, "FedHIP: Federated learning for privacy-preserving human intention prediction in human-robot collaborative assembly tasks," *Adv. Eng. Inform.*, vol. 60, Apr. 2024.

[27] S. Na, T. Roucek, J. Ulrich, J. Pikman, T. Krajnik, B. Lennox, and F. Arvin, "Federated Reinforcement Learning for Collective Navigation of Robotic Swarms," *IEEE Trans. Cogn. Develop. Syst.*, vol. 15, no. 4, pp. 2122–2131, Dec. 2023.

[28] D. Nguyen-Tuong and J. Peters, "Local Gaussian process regression for real-time model-based robot control," in *Proc. 2008 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2008, pp. 380–385.

[29] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, J. Fernandez-Marques, Y. Gao, L. Sani, K. H. Li, T. Parcollet, P. P. B. d. Gusmão, and N. D. Lane, "Flower: A Friendly Federated Learning Research Framework," Mar. 2022, arXiv:2007.14390.

[30] N. Pedrocchi, E. Villagrossi, F. Vicentini, and L. M. Tosatti, "On robot dynamic model identification through sub-workspace evolved trajectories for optimal torque estimation," in *Proc. 2013 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2013, pp. 2370–2376.