Robotica (2024), 1–16
doi:10.1017/S026357472400170X

RESEARCH ARTICLE

Hybrid learning-based visual path following for an industrial robot

Mustafa Can Bingol[1] and Omur Aydogmus[2]

[1]Department of Electrical-Electronics Engineering, Burdur Mehmet Akif Ersoy University, Burdur, Turkiye
[2]Department of Mechatronics Engineering, Faculty of Technology, Fırat University, Elazig, Turkiye
Corresponding author: Omur Aydogmus; Email: oaydogmus@firat.edu.tr

Received: 16 May 2024; **Revised:** 12 August 2024; **Accepted:** 9 September 2024

Keywords: hybrid learning; industrial robot; reinforcement learning; supervised learning

Abstract

This study proposes a novel hybrid learning approach for developing a visual path-following algorithm for industrial robots. The process involves three steps: data collection from a simulation environment, network training, and testing on a real robot. The actor network is trained using supervised learning for 500 epochs. A semitrained network is then obtained at the 250th epoch. This network is further trained for another 250 epochs using reinforcement learning methods within the simulation environment. Networks trained with supervised learning (500 epochs) and the proposed hybrid learning method (250 epochs each of supervised and reinforcement learning) are compared. The hybrid learning approach achieves a significantly lower average error (30.9 mm) compared with supervised learning (39.3 mm) on real-world images. Additionally, the hybrid approach exhibits faster processing times (31.7 s) compared with supervised learning (35.0 s). The proposed method is implemented on a KUKA Agilus KR6 R900 six-axis robot, demonstrating its effectiveness. Furthermore, the hybrid approach reduces the total power consumption of the robot's motors compared with the supervised learning method. These results suggest that the hybrid learning approach offers a more effective and efficient solution for visual path following in industrial robots compared with traditional supervised learning.

1. Introduction

Robots are reprogrammable machines used to perform repetitive or dangerous tasks instead of humans. They can be categorized into platform-dependent and mobile robots. According to World Robotics 2021 data, approximately 3 million industrial robots are in operation worldwide, with 384 thousand installed in 2020. Energy efficiency is crucial in industrial applications as it can lead to significant cost savings and environmental benefits. Many industries consume vast amounts of energy to run their processes, and energy efficiency can help reduce this energy consumption. Improving energy efficiency in industrial processes involves identifying areas where energy can be conserved and implementing measures to reduce energy waste. This can include using more energy-efficient equipment, optimizing processes to reduce energy consumption, and implementing energy management systems. Not only does energy efficiency lead to cost savings, but it also helps reduce greenhouse gas emissions and contributes to a cleaner environment. Moreover, energy efficiency can enhance the competitiveness of industries by improving productivity, reducing operational costs, and enhancing the reliability of energy supply.

Industrial robots have become increasingly popular in manufacturing and other industrial applications due to their ability to perform repetitive and complex tasks with precision and speed. However, these robots consume a significant amount of energy, which can lead to high operating costs and environmental impacts. Therefore, energy-saving measures are critical to improve the sustainability of industrial robot applications. There are various ways to reduce energy consumption in industrial robots, such as using more efficient motors, reducing idle time, and optimizing the robot's movements. By implementing

https://doi.org/10.1017/S026357472400170X Published online by Cambridge University Press

these measures, companies can save a considerable amount of energy and reduce their operating costs. Moreover, reducing energy consumption in industrial robots can help minimize their carbon footprint, contribute to a more sustainable manufacturing process, and align with environmental regulations. Therefore, energy-saving measures in industrial robots are essential for both economic and environmental reasons, and companies should prioritize them in their operations. In 2021, the number of robot installations reached a historic high of 517,385 units, marking a growth rate of 31% compared with 2020. This surge exceeded the previous record level of 423,321 units achieved in 2018 by 22%. Additionally, the operational stock of industrial robots was estimated to be 3,477,127 units in 2021. As a result, it is imperative for companies to prioritize the efficiency of their robots and invest in developing this feature to gain a competitive edge. All of these statistical contents have been taken from the International Federation of Robotics [1].

Numerous tasks have been proposed for robots in the literature. For a robot to perform a given task accurately, it must follow a predefined path or trajectory. Trajectory tracking involves following a path that changes over time and is used for tasks like welding and sealing. In contrast, path following involves adhering to a fixed path and is used in mobile robot navigation and industrial robot handling.

Several studies have explored model-based approaches for path following in robotics. For instance, Lin et al. [2] presented a data-driven method for online path correction in industrial robots, addressing joint position errors that impact path accuracy. Raikwar et al. [3] proposed a stable navigational algorithm for orchard robots designed to operate effectively in Global Navigation Satellite Systems (GNSS)-denied environments. Khaled et al. [4] introduced an active disturbance rejection control scheme to improve the path accuracy of Mecademic's Meca500 six-axis industrial robot. Model-based approaches are particularly suited for scenarios where accurate models of the robot and environment are available, offering predictability and stability but requiring complex modeling. Conversely, artificial intelligence (AI)-based approaches are advantageous in environments where creating precise models is challenging or where adaptability is essential, providing flexibility and learning capabilities but often requiring substantial data and computational resources.

There has been an extensive research on visual-based path following in robotics. Genk et al. [5] developed a novel algorithm for welding industrial robots that performs path extraction on complex surfaces by iteratively processing images from a 3D camera. However, due to differences in robots, processes, and cameras, the current work, when compared solely on methods, operates faster as it is noniterative. Other studies have developed structures for visual path tracking that are trained with data provided by operators using methods akin to end-to-end learning [6, 7]. This study eliminates the need for operator data. Instead, after updating the neural network weights with supervised learning, reinforcement learning is employed to enhance the robustness of the responses. Moreover, most reinforcement learning algorithms are typically applied in simulation environments to avoid potential damage to the robot or surroundings during the trial-and-error training phase [8, 9]. Consequently, networks trained in simulations often do not perform as expected in real-world environments. The hybrid learning method proposed in this study, incorporating reinforcement learning, demonstrates smooth operation in real-world settings.

A four-phase hybrid approach to solve the multiobjective trajectory planning problem for industrial robots has been presented in [10], combining robustness and flexibility. Zhang et al. [11] presented a trajectory planning method for manipulators that includes time and jerk optimization with obstacle avoidance. Yang et al. [12] proposed a collision avoidance trajectory planning method for dual-robot systems. Katiyar et al. [13] proposed an approach for a 14-DOF Rover Manipulator System's end-effector path tracking. Tang et al. [14] presented a path obstacle avoidance algorithm for a 6-DOF robotic arm.

While most of these robots are conventional, some researchers have aimed to modernize them. For example, Oh et al. [15] achieved fault detection in industrial robots using a recursive convolutional neural network (CNN). Chen et al. [16] performed fault prediction for heavy-duty robots with a CNN structure. Bingol et al. [17] used CNNs for classifying motor torque information to ensure safe human–robot interaction and employed speech for communication between humans and industrial robots [18]. AI has been used to model industrial robot energy consumption [19], and deep learning structures have been applied to solve inverse kinematics in parallel industrial robots [20]. Rath et al. [21] utilized a model predictive controller for path following in an autonomous submarine vehicle, and Zheng et al. [22] developed a soft actor–critic-based controller for unmanned surface vehicles. Mondal et al. [23] implemented a fuzzy logic controller for wheeled mobile robots, and other studies focused on optimal path following using least-squares methods [24], deep reinforcement learning [25], and 3D path-following control algorithms [26]. Learning involves using experiences gained from new situations. It can be categorized into supervised, unsupervised, and reinforcement learning. Supervised learning uses datasets of input–output pairs, unsupervised learning focuses on the relationships between input data, and reinforcement learning establishes relationships between state and reward pairs [27]. For example, trajectory planning for industrial robots has been performed using a region-based CNN trained through supervised learning [28], and supervised learning has enabled humanoid robots to determine hand location [29]. Position estimation for the KUKA KR240 R2900 robot [30] and motion planning optimization for assembly robots [31] have also used supervised learning. The current paper employs a CNN structure for visual path following, trained with both supervised and reinforcement learning algorithms. Reinforcement learning includes value and policy learning. The double deep Q-network (DQN) algorithm, a value learning method, has been used to guide mobile robots along paths [32] and navigate to targets [33]. The DQN structure has also been applied to detect odor/gas sources [34] and track weld seams [35]. Policy learning methods, such as the proximal policy optimization (PPO) algorithm, have been used for path planning in multirobot systems [36], industrial robots [37], and real-time motion planning [38]. The current study uses the PPO algorithm for reinforcement learning. Hybrid learning methods are also explored in the literature. For instance, fuzzy logic algorithms have been combined with genetic algorithms for controlling multiple mobile robot systems [39], and genetic fuzzy logic controllers have been used for mobile robot systems [40]. Fault detection in industrial robots has been addressed by combining artificial neural networks with support vector machines [41]. Hybrid learning often involves optimizing one method's parameters using another or connecting different structures in sequence.

As a distinct approach from previous literature, this study aims to combine the advantages of supervised and reinforcement learning algorithms while addressing their disadvantages. This study addresses the challenge of developing an efficient and accurate visual path-following algorithm for industrial robots. Traditional supervised learning methods, while somewhat effective, have limitations in accuracy and processing speed when applied to real-world scenarios. To overcome these limitations, a novel hybrid learning approach is proposed, combining supervised learning with reinforcement learning. The hybrid learning method significantly reduces the average error in the robot's path-following capabilities, achieving an average error of 30.9 mm compared with 39.3 mm with supervised learning alone. It also reduces the processing time required for path-following tasks, averaging 31.7 s compared with 35.0 s with supervised learning. Additionally, the hybrid learning method decreases the total power consumption of the robot's motors, enhancing energy efficiency. By addressing these multifaceted issues, the study demonstrates that the proposed hybrid learning approach offers a more effective and efficient solution for visual path following in industrial robots compared with traditional supervised learning methods. This approach not only improves the accuracy and speed of the robot's performance but also enhances energy efficiency, which is crucial for industrial applications.

Contribution of this work: A novel hybrid learning approach is introduced, combining supervised and reinforcement learning for visual path following in industrial robots. This method significantly improves accuracy, processing speed, and energy efficiency compared with traditional approaches, as demonstrated on the KUKA Agilus KR6 R900 six-axis robot. These contributions provide a more effective solution for industrial automation, particularly in energy-sensitive applications.

2. Method

2.1. Theoretical framework

A hybrid approach can improve performance compared with pure supervised or reinforcement learning techniques by combining the strengths of both methods. Supervised learning provides accurate and

https://doi.org/10.1017/S026357472400170X Published online by Cambridge University Press

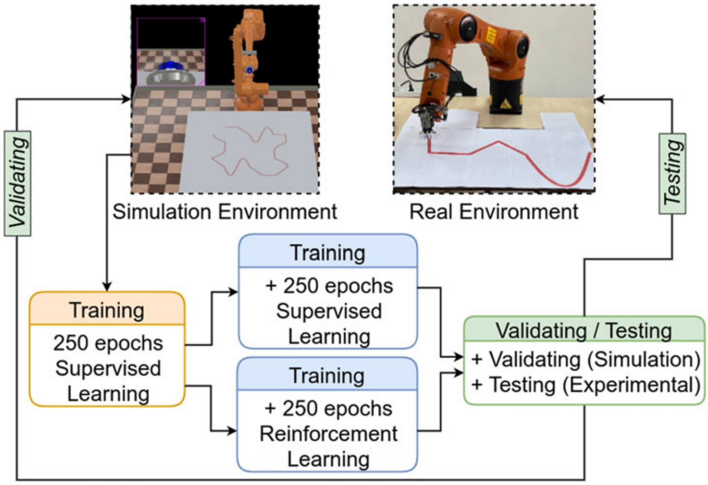

Figure 1. *Block diagram of system.*

efficient learning from labeled data, allowing the model to generalize well on seen examples, while reinforcement learning excels in decision-making and learning from interactions with the environment, helping the model adapt to new situations. By combining these strengths, hybrid approaches make better use of available data, such as using supervised learning to pretrain a model, reducing the exploration space, and speeding up convergence in reinforcement learning. This combination also improves generalization across different environments and scenarios. Moreover, incorporating supervised learning can guide the model in environments with sparse or delayed rewards, making the learning process more efficient. The initial foundation provided by supervised learning reduces the need for extensive exploration in reinforcement learning, balancing exploration and exploitation more effectively. Knowledge from supervised learning can be transferred to reinforcement learning in different domains, enhancing the model's performance across various tasks. By combining the strengths of both supervised and reinforcement learning, hybrid approaches achieve improved performance, robustness, and efficiency.

2.2. System design

Until recently, industrial robots were only used as machines that could perform certain preprogrammed tasks. However, for more effective use of robots, they must be able to make decisions like humans. AI-supported algorithms can enable standard robots to become smarter machines, allowing them to learn by themselves, much like humans. With this ability, a robot can respond optimally in situations it has not encountered before. The goal of this study was to create an AI-based algorithm for an industrial robot, eliminating the need for reprogramming and calibration for different tasks.

Simulation and experimental studies were carried out using the system presented in Figure 1. The artificial neural networks designed for the system were trained using an industrial robot (ABB IRB4600-40) in the Webots robot simulator program environment. Simulation parameters were chosen as default, with gravity of 9.81 m/s^2, basic time step of 32 ms, 3D display frame per second of 60, physics disable linear velocity threshold of 0.01 m/s, physics disable angular velocity threshold of 0.01 rad/s, drag force scale of 30, and drag torque scale of 5. As a next step, the networks trained in the simulation environment were experimentally tested on a real industrial robot (KUKA Agilus KR6 R900 sixx). In addition, the networks developed with a real image taken from the real system were tested in the simulation environment. The generated path was used for the real robot trajectory. A view of the created simulation environment is shown in Figure 2. The mass of the ABB robot shown in Figure 2 is about 435 kg, the payload of the robot is 40 kg, the reach distance is 2550 mm, and the repeatability of the robot is about 0.06 mm. A camera and a pen were mounted on the tool center point (TCP) of the manipulator. The

https://doi.org/10.1017/S026357472400170X Published online by Cambridge University Press

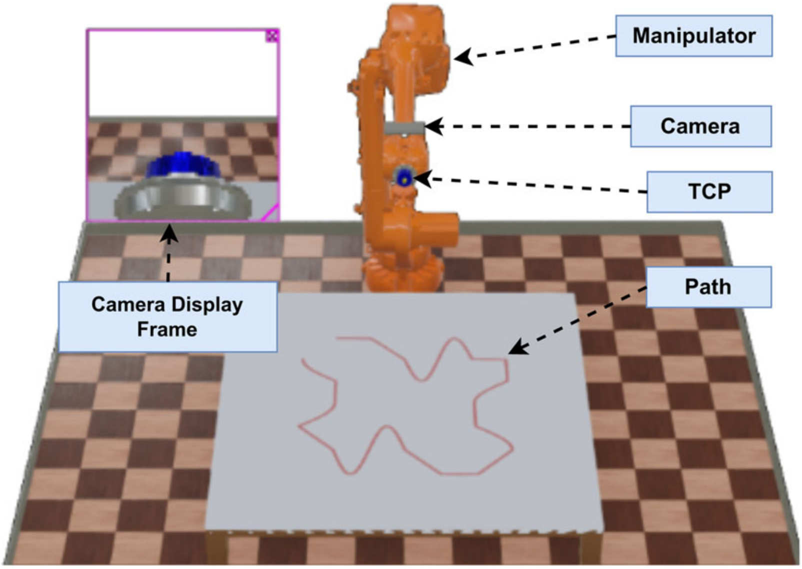

Figure 2. Simulation environment for training procedure.

resolution of the camera was adjusted as 64 × 64 with RGB video format. The pen on the TCP was used to simulate a dummy sealing operation on the determined trajectory on the workbench as labeled *path* on the figure.

2.3. Supervised learning

Supervised learning is a technique that establishes a relationship between the input and output of a system. The dataset comprises an input–output dataset ($D(x_i, y_i)$). The learning process involves the following steps: generating the dataset, standardizing the data, determining the network architecture, and training and testing the network.

In this study, the dataset was collected from a training path created as shown in Figure 3 for the robot's TCP traveling on the path. The path was created by combining a total of 16 different paths consisting of linear and sinusoidal functions, considering the possible paths that a real robot may experience. For the training, the background color was taken as gray [$ColorCode$: #8C99AA], and the line color was adjusted dark red [$ColorCode$: #782829], in accordance with the color tones from a real photo taken via a camera from the experimental environment. The distance between two consecutive points on the training path was constrained between 1 and 2 cm, and the path consists of a total of 362 waypoints. An image was taken with the camera mounted on the robot at each path step, and the position difference between the two steps (Δ_x and Δ_y) was recorded by moving the robot on the training path. The input data of the dataset consisted of recorded images taken from the camera on the TCP, and the output of the dataset was formed by the position differences that occurred during the TCP motion.

The output data were standardized with the equation given in Eq. (1) to make the learning process more stable:

$$D_{\mathcal{S}}(y_i) = \frac{D(y_i) - \overline{D(y_i)}}{\sigma\ (D(y_i))} \tag{1}$$

where σ is the standard deviation and the output of dataset is symbolized as $D(y_i)$ in Eq. (1). $\overline{D(y_i)}$ is the mean value of the data. The standard output data $D_{\mathcal{S}}(y_i)$ must be between -1 and 1 to be applied in the hyperbolic tangent (tan_h) activation function. tan_h function enables the explore process to work more stable in the reinforcement learning process. $D_{\mathcal{S}}(y_i)$ is normalized $D_{\mathcal{N}}(y_i)$ using Eq. (2). In addition, the

https://doi.org/10.1017/S026357472400170X Published online by Cambridge University Press

Table I. *CNN architecture.*

Layer type	Shape	Function
Input layer	$(64 \times 64 \times 3)$	-
Convolution layer	$(FS = 32, KS = 5 \times 5)$	*ReLU*
Convolution layer	$(FS = 128, KS = 5 \times 5)$	*ReLU*
Pooling layer	$(KS = 2 \times 2)$	*max*
Convolution layer	$(FS = 128, KS = 5 \times 5)$	*ReLU*
Pooling layer	$(KS = 2 \times 2)$	*max*
Convolution layer	$(FS = 32, KS = 5 \times 5)$	*ReLU*
Dense layer	$(NS = 256)$	*ReLU*
Dense layer	$(NS = 1024)$	*ReLU*
Dense layer	$(NS = 2)$	tan_h

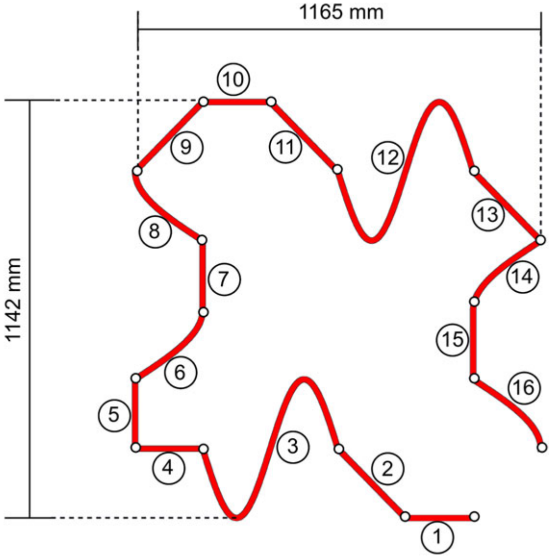

Figure 3. *Training trajectory path.*

brightness values of the input data $D(x_i)$ are randomly adjusted up to 10% during the training process to obtain a robust trained network working with the real image:

$$D_{\mathcal{N}}(y_i) = \frac{2\,(D_{\mathcal{S}}(y_i) - min\,(D_{\mathcal{S}}(y_i)))}{max\,(D_{\mathcal{S}}(y_i)) - min\,(D_{\mathcal{S}}(y_i))} - 1 \tag{2}$$

In this study, Table I presents the CNN architecture that is used as a supervised learning structure because regression will be applied to the input images.

The *Convolution Layer* is a customized neural network structure shown in Table I. This network produces output by convolutional processing of the input data. The shape column in the table presents the number of the filter (FS) and kernel size (KS). The last column shows the activation functions used in the related line. The pooling layer produces output based on the neighborhood relationship of the data input. The hyperparameters of this layer are kernel size and neighborhood function. In the study, maximum (*max*) was used as the pooling function. The dense layer was fully connected to the previous layer and produces the output after the necessary mathematical calculations. The neuron size was shown as NS. In this study, rectified linear units (*ReLU*) and tan_h were used as activation functions.

The training process was started after the creating the dataset ($D(x_i, y_i)$) and the CNN architecture given in Table I. The dataset was divided into 90% for training and 10% testing. Adam optimization method and mean absolute error loss function were used in the training process. The number of training epoch was taken as 500. The learning rate decay function for supervised learning is as follows: If the epoch is less than 100, the learning rate is set to 1e-4. If the epoch is between 100 and 200, the learning rate is reduced to 1e-5. If the epoch is between 200 and 400, the learning rate is further reduced to 1e-6. For epochs greater than or equal to 400, the learning rate is fixed at 5e-7. This decay function progressively reduces the learning rate as the training progresses, allowing for fine-tuning of the model parameters in the later stages of training. In this study, distance was measured in meters, and the learning rates were intentionally set lower than those in other studies in the literature. The training was carried out in two stages, the first 250 epochs were performed and recorded, and the last 250 epochs were performed using this record.

2.4. Reinforcement learning

Reinforcement learning is the process of enabling predetermined structures to produce optimal outputs based on instantaneous state information. There are two types of reinforcement learning: value learning and policy learning. Value learning is applied to systems with a fixed output space, while policy learning is applied to systems with a fixed output space width. In this study, the system output is the position difference (Δ_x and Δ_y) of the robot's TCP, and these position differences are not constant. Therefore, the policy learning method was chosen for this application.

The action of the designated agent is called policy. Politics (π) is the establishment of a probabilistic distribution relationship between the state space ($\mathcal{S}$) and the action space ($\mathcal{A}$). It can be shown as $\pi:\mathcal{S}\rightarrow\mathcal{P}(\mathcal{A})$. This probabilistic relationship is instantaneous ($s_t=(x_1,a_1,\ldots,a_{t-1},x_t)$), instantaneous output (a_t), instantaneous reward ($r_t=(s_t,a_t)$), and the next state (s_{t+1}). x_t is a time-dependent observation of a stochastic environment (E). Accordingly, discounted future reward (R_t) can be defined as given in Eq. (3). Moreover, the detailed information about this equations are given in [42, 43]:

$$R_t=\sum_{i=t}^{T}\gamma^{i-t}r(s_i,a_i),\gamma\in[0,1] \tag{3}$$

where γ denotes reduction coefficient. The purpose of policy learning is to learn to maximize the initial distribution (J), which can be determined in Eq. (4):

$$J=\mathbb{E}_{r_i,s_i\sim E,a_i\sim\pi}[R_1] \tag{4}$$

The value R_1 represents a reward that is calculated using the initial distribution. The action-value function for output a_t in state s_t, as determined by the specified policy, is given by Eq. (5):

$$Q^{\pi}(s_t,a_t)=\mathbb{E}_{r_{i\geq t},s_i>t\sim E,a_i\sim\pi}[R_t|s_t,a_t] \tag{5}$$

Given that reinforcement learning is a dynamic process, Bellman's equation can be used, as given in Eq (6):

$$Q^{\pi}(s_t,a_t)=\mathbb{E}_{r_t,s_{t+1}\sim E}[r(s_t,a_t)+\gamma\mathbb{E}_{a_{t+1}\sim\pi}[Q^{\pi}(s_{t+1},a_{t+1})] \tag{6}$$

The target policy function can be defined as $\mu:\mathcal{S}\leftarrow\mathcal{A}$. In this study, the action-critic method was chosen, and these constructs were trained using the PPO algorithm presented in Table II. Probability ratio is given in Eq. (7):

$$\mathcal{R}_t(\theta)=\frac{\pi_\theta(a_t\ s_t)}{\pi_{\theta_{old}}(a_t\ s_t)} \tag{7}$$

The loss function used in the PPO algorithm is presented in Eq. (8). The detailed information about PPO algorithm has been presented in [42, 44]:

$$\mathcal{L}(\theta)=\hat{\mathbb{E}}_t\left[min\left(\mathcal{R}_t(\theta)\hat{A}_T,clip\left(\mathcal{R}_t(\theta),1\pm\epsilon\right)\hat{A}_T\right)\right] \tag{8}$$

https://doi.org/10.1017/S026357472400170X Published online by Cambridge University Press

Table II. *PPO algorithm.*

```
for episode: 1 → $M$
    Initialize $Batch$
    Reset environment and get $s_1$
    for iteration: 1 → $T$
        $a_t = \pi(s_t|\theta^\pi) + \mathcal{N}_t$
        $r_t, s_{t+1} \leftarrow E(a_t)$
        $Batch \leftarrow [s_t, a_t, r_t, s_{t+1}]$
    Calculate $\hat{A}_1, \ldots, \hat{A}_T$
    Calculate new weights by minimizing $\mathcal{L}$
    Update weights
```

Table III. *Critic CNN structure.*

Layer type	Shape	Function
Input layer	$(64 \times 64 \times 3)$	-
Convolution layer	$(FS = 32, KS = 3 \times 3)$	*ReLU*
Pooling layer	$(KS = 2 \times 2)$	*max*
Convolution layer	$(FS = 32, KS = 3 \times 3)$	*ReLU*
Pooling layer	$(KS = 2 \times 2)$	*max*
Convolution layer	$(FS = 32, KS = 3 \times 3)$	*ReLU*
Pooling layer	$(KS = 2 \times 2)$	*max*
Dense layer	$(NS = 128)$	*ReLU*
Dense layer	$(NS = 1)$	*Linear*

In Equation 8, θ, $\hat{A}_T$, *clip*, and ϵ represent the network weights, advantage estimation, clipping function, and clipping coefficient, respectively. The CNN structures used for the actor and critic in this study are presented in Tables I and III, respectively.

In this study, the image s_t was obtained from the camera mounted on the manipulator's TCP, and the position difference in the x and y axes was determined as a_t. The reward function r_t was determined according to Eq. 9. In this study, an acceptable following distance of 0.02 m was determined, and the reward function r_t was based on this threshold. If a similar study were to be conducted in the future, these values could be updated based on the requirements of that study:

$$r_t = \begin{cases} 10 & : \text{if } \Delta_t < 0.01 \\ 5 & : \text{if } \Delta_t < 0.02 \\ -1 & : \text{if } \Delta_t < 0.05 \\ -2 & : \text{if } \Delta_t < 0.1 \\ -5 & : \text{otherwise} \end{cases} \tag{9}$$

Δ_t represents the state of TCP relative to the path that is presented in Eq. (10):

$$\Delta_t = \sqrt{(x_{TCP_t} - x_{Path_t})^2 + (y_{TCP_t} - y_{Path_t})^2} \tag{10}$$

In this study, the Adam optimization method was used with the following parameters: $\gamma = 0.99$, $\epsilon = 0.2$, buffer size = 1024, and batch size = 64. The learning rate decay function for reinforcement learning is as follows: For epochs less than 150, the learning rate of the actor network is set to 1e-6, and the learning rate of the critic network is set to 2e-6. For epochs greater than or equal to 150, the learning rate of the actor network is reduced to 5e-7, and the learning rate of the critic network is reduced to 1e-6. This decay function adjusts the learning rates of both the actor and critic networks to

https://doi.org/10.1017/S026357472400170X Published online by Cambridge University Press

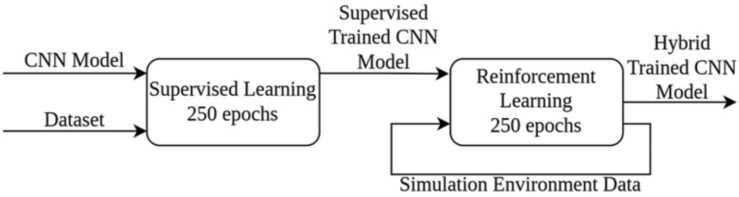

Figure 4. *Hybrid learning block diagram.*

optimize the training process for reinforcement learning. In addition, the training process was continued for 250 epochs.

2.5. Hybrid learning

In this study, the hybrid learning approach was developed by first training with supervised learning and then applying reinforcement learning algorithms.

A CNN model is trained using supervised learning and a dataset, as shown in Figure 4. The trained CNN is then further trained using reinforcement learning in a simulation environment. The motivation behind this proposed method was to combine the learning speed and accuracy of supervised learning with the robustness of reinforcement learning. Supervised learning provides two main advantages: optimizing weights in the neural network and predicting accurate results. The results were then made more robust by reinforcement learning, a learning method based on the principles of exploration and exploitation. This approach results in a learning method that was more robust than supervised learning and faster than reinforcement learning. However, this proposed method also had the disadvantage of requiring the design of an environment in which reinforcement learning can work, in contrast to supervised learning.

3. Experimental study and discussion

In the experimental setup, hyperparameters were configured as follows: loss clipping was set to 0.2, consistent with the original PPO paper [42]. The discount factor was set to 0.99, adhering to the recommendations outlined in the same foundational study. The buffer size was configured at 1024, with a batch size of 64 to facilitate efficient training. To introduce variability and enhance model robustness, noise was generated using a random normal distribution with a mean (loc) of 0 and a standard deviation (scale) of 0.02. Additionally, the dataset was partitioned with 90% of the images used for training and 10% for testing. The images were reshaped to a resolution of 64x64 pixels, and a batch size of 32 was employed during the training process. These hyperparameter choices and preprocessing steps align with established practices in reinforcement learning and image processing literature, ensuring both stability and performance in the training and evaluation phases.

An industrial robot consists of three parts: the manipulator, the cabinet, and the SmartPad. The manipulator is the moving part of the robot that performs the desired operations using motors and gears. The electronics and software of the robot, such as the motor driver circuits and operating system, are located inside the cabinet. The SmartPad provides communication between the robot and the operator (human, user). The KUKA Agilus KR6 R900 sixx robot and the experimental environment used in this study are presented in Figure 5. The robot has a weight of 52 kg and can carry a payload of up to 6 kg. It has a maximum reach of 901.5 mm. The photograph shown in Figure 5 was taken by the robot's camera in the experimental environment to test the developed method with real input.

The test path consisted of a total of 145 steps from five different subfunctions. The path is 920 mm long on the x-axis and 760 mm wide on the y-axis. In addition, the photograph shown in Figure 5 was used as input for the developed system without any preprocessing that would alter the pixel values.

https://doi.org/10.1017/S026357472400170X Published online by Cambridge University Press

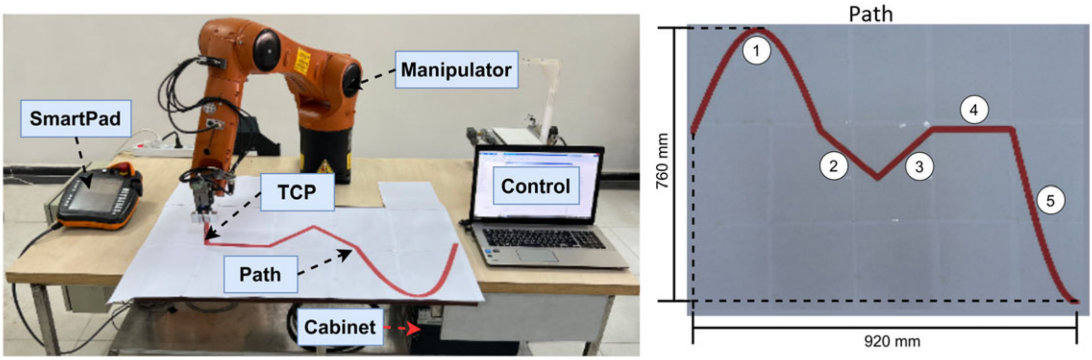

Figure 5. *Experimental setup and real photo of test path.*

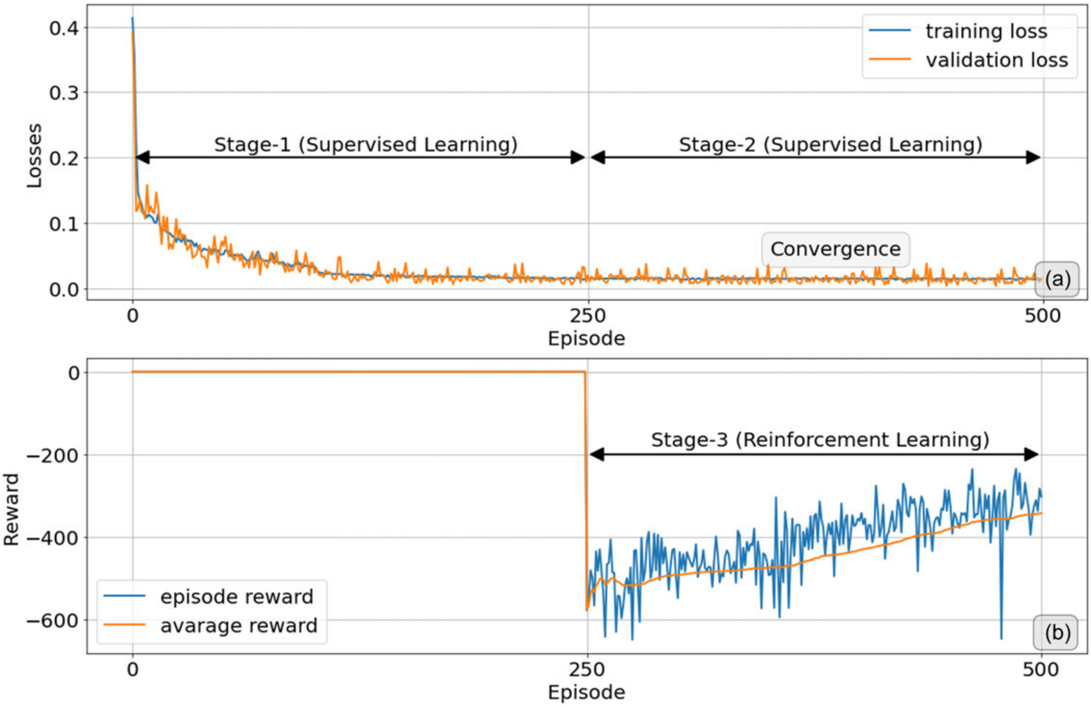

Figure 6. *(a) Training performance of supervised learning, (b) training performance of reinforcement learning.*

In this study, a hybrid learning method combining supervised learning and reinforcement learning was trained for the same action CNN structure. First, the supervised learning method was trained using the training dataset for 500 epochs. As shown in Figure 6(a), Stage 1 progressed rapidly with supervised Learning training, while in Stage 2, progress stalled after convergence at the 250*th* epoch. Stages 2 and 3 continued training together for up to 250 epochs. To obtain a valid comparison for this study, both trainings were terminated at the 500*th* epoch.

The actor network was saved after the completion of the 250*th* epoch during supervised learning. This saved actor network was then used for reinforcement learning for the next 250 epochs. The reward and average reward graphs created during reinforcement learning are shown in Figure 6(b). If the episode count is above 100, the average reward is calculated based on the last 100 rewards. Otherwise, the average reward is calculated up to the current episode count. The networks trained in the simulation environment

https://doi.org/10.1017/S026357472400170X Published online by Cambridge University Press

Table IV. *Descriptive analysis of test environment results.*

	Supervised learning		Hybrid learning	
	Error (mm)	**Duration (s)**	**Error (mm)**	**Duration (s)**
Mean±SD	39.3±1.64	35.0±0.25	30.9±1.27	31.7±0.07
Min-max	37.00–41.50	34.69–35.33	29.00–32.50	31.62–31.74
Median	39.00	35.07	30.50	31.74
IQR	2.88	0.45	2.63	0.13
p	**0.000***	**0.000***	**0.000***	**0.000***

SD: standard deviation, IQR: interquartile range, $^*p < 0.05$

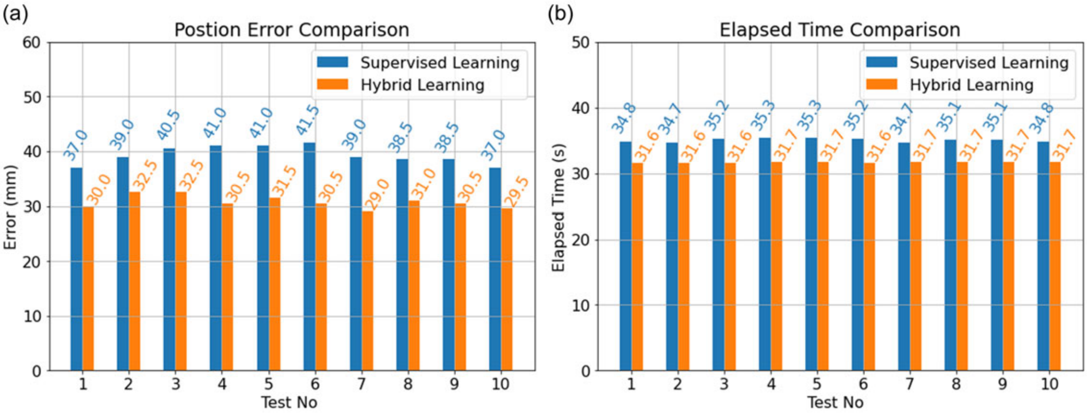

Figure 7. *Performance metrics of supervised and hybrid learning for 10 tests: (a) position error, (b) elapsed time comparison.*

were tested using the photograph taken from the experimental environment. In the simulation environment, the robot was tested 10 times by moving it in positive and negative directions along the x-axis. The average errors and elapsed times of the tests are shown in Figure 7. The purpose of this study is visual path tracking. For robotic systems, a path is a set of defined position information that is independent of time. In this work, the error symbolizes the difference between the robot's position and the expected path position. Descriptive analysis of the two methods used in the test environment is presented in Table IV. It can be observed that a statistically significant difference was found between the learning types in terms of error and duration ($p < 0.05$).

One of the paths was chosen randomly and used for the experimental study with the KUKA Agilus KR6 R900 sixx robot. The real robot's TCP movements of the process are shown in Figure 8(a). P1 denotes the start point of the movements, and P2 is the last point of the trajectory path. The robot first moved from 0 mm to 935 mm in the positive x-axis direction and then moved in the opposite direction. In addition, the errors of the predictions are shown in Figure 8(b). The mean values of the supervised learning and hybrid learning were obtained as 37.36 mm and 34.18 mm, respectively.

The robot was moved using the linear movement (LIN) command with a speed of $2m/s$ between points on the path. Hybrid learning completed the process approximately 716 ms earlier than the supervised learning method. The motor power of the robot's axes was measured using the robot trend feature of the KUKA, as shown in Figure 9. The red markers indicate the end of the hybrid learning process, while the blue markers signify the end of the supervised learning process. The power consumption of the motors was found to be almost the same for both methods. However, the hybrid learning method completed the process in a shorter time than the supervised learning method, resulting in reduced total power consumption.

Figure 10 shows two videos that were recorded during the experimental study. Photos of the experimental process were created by cropping the recorded videos at a rate of 0.5 fps. The first row of photos

https://doi.org/10.1017/S026357472400170X Published online by Cambridge University Press

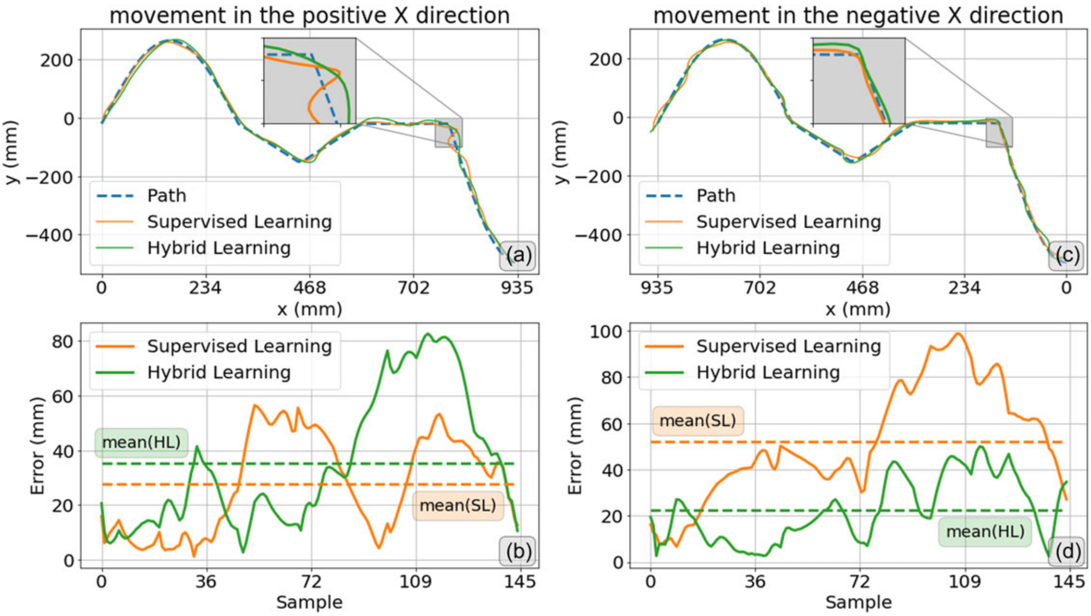

Figure 8. *(a) Trajectory with TCP movement along positive x-axis, (b) errors of positive travel, (c) trajectory with TCP movement along the negative x-axis, (d) errors of negative travel.*

depicts the process using the supervised learning algorithm, while the second row shows the process based on hybrid learning. Photos numbered 1 and 11 mark the start point of the processes, and the middle of the trajectory path is marked as 6/7. As shown, the method based on hybrid learning completed the process faster than the other method.

There are several studies in the literature that are similar to the present study. In [25], an industrial KUKA KR16HW robot was visually guided to follow a specified path using a CNN structure, similar to our work. However, the CNN structure in that study was trained using only reinforcement learning algorithms, and the training process took approximately 10^6 steps. One advantage of that study was that it included a real production process. In contrast, our proposed hybrid learning method reduced the training process to just 500 steps. Another study with a similar approach to hybrid learning added a support vector machine to the output of an artificial neural network for fault detection in an industrial robot [41]. Several studies have combined these approaches to create hybrid systems [39, 40]. Although these studies did not specify any differences from the present work, the same structure was trained using different learning methods, and their performance was compared with the present work.

The study has some constraints, which can be divided into algorithmic, hardware, and software limitations. The proposed method can only work on labeled data because it combines supervised and reinforcement learning methods. This is an algorithmic limitation of the study. In the performed application, the hardware limitation is that data collected from the real environment had to be processed by an external computer before the robot could follow the created path. This prevented the robot from performing the process online. The software constraint is that the developed software works only on the specified plane. The *z*-axis was not taken into account in the presented study, but a depth camera could be added to measure the distance between the TCP and the table surface if needed. Other software constraints include ground texture, path color, and path thickness. However, these constraints can be overcome by training a CNN architecture using the proposed method with training routes that have different ground textures, path colors, and path thicknesses.

Path following is a method that can be used in industrial robots for manufacturing process such as sealing and grinding. In the current study, the aim was to develop an effective path following method

https://doi.org/10.1017/S026357472400170X Published online by Cambridge University Press

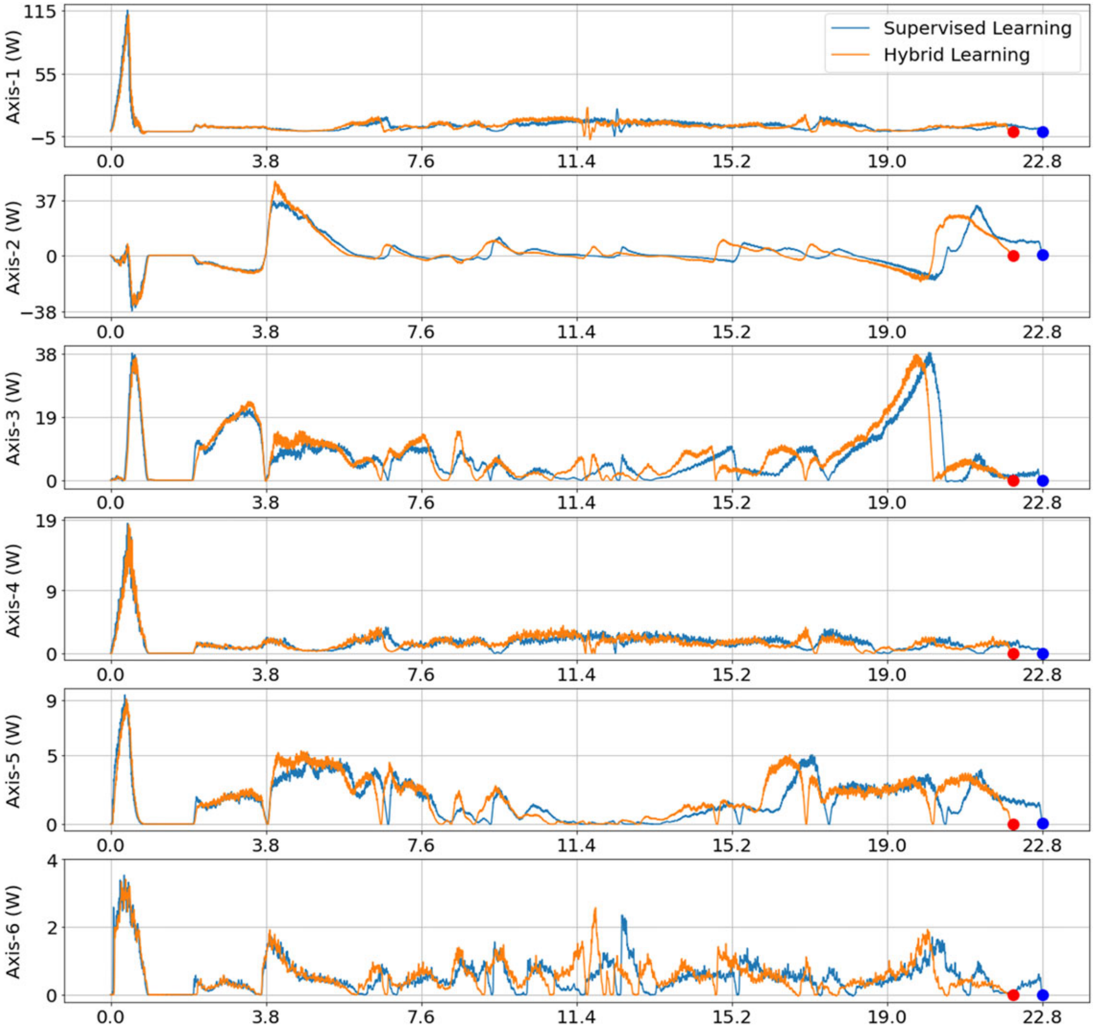

Figure 9. *Power consumption of robot axis motors during movement.*

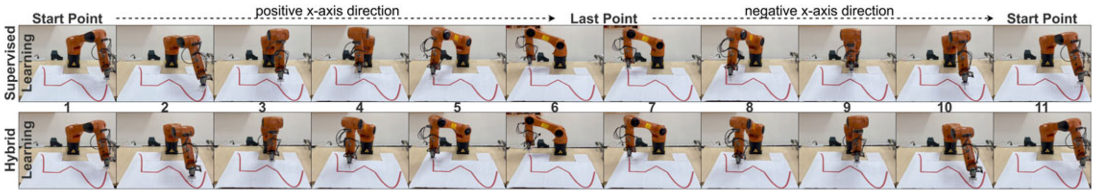

Figure 10. *Cropped video files with 0.5 fps for two proposed methods.*

for industrial robots. As the first step of the study, a system similar to the experimental environment was modeled using the Webots simulation environment. A dataset consisting of image and position differences was collected from the simulation environment. The actor networks were trained using this dataset with both supervised and hybrid learning methods. The trained networks were tested with a real image taken from the experimental environment. The results of the tests show that the actor network trained with supervised learning followed the path with an average error of 39.30 mm, while the network trained

https://doi.org/10.1017/S026357472400170X Published online by Cambridge University Press

with hybrid learning had an average error of 30.75 mm. The actor networks trained with supervised and hybrid learning had average elapsed times of 35.02 and 31.69 s, respectively, and there was a statistically significant difference in both the errors and elapsed times between the two methods ($p < 0.05$). As the last step of the study, the method was tested by selecting one of the generated paths at random and tested with the KUKA Agilus KR6 R900 sixx robot.

The motor powers of the robot's axes were measured during the test process. The results showed that two learning methods offered nearly the same performance for the time axis. However, the hybrid-based method consumed less power than the supervised method because it completed the path faster. In this way, the robot controlled with the neural network trained with the hybrid learning method consumed less energy for the same task than the robot controlled with the neural network trained with the supervised learning method. This difference is small because there is a certain speed constraint in a specified short-term process. This energy difference is expected to increase in tasks with longer task duration and faster movement. As a result, the proposed hybrid learning model exhibited fewer errors and faster path following than the supervised learning method. Additionally, the proposed hybrid learning method required less elapsed time to train than traditional reinforcement learning methods. In future studies, various real industrial applications will be implemented using networks trained with the proposed learning method.

4. Conclusion

This study presents a novel hybrid learning approach that synergizes the advantages of both supervised and reinforcement learning to enhance the performance of visual path-following algorithms for industrial robots. The proposed hybrid method effectively combines the precision of supervised learning with the adaptability of reinforcement learning, addressing the limitations inherent in each technique when used in isolation. The hybrid approach demonstrated significant improvements over traditional supervised learning methods. Utilizing supervised learning for initial model training, the approach reduced the exploration space required for reinforcement learning, resulting in a notable decrease in path-following error from 39.3 mm to 30.9 mm and a reduction in processing time from 35.0 s to 31.7 s. Additionally, the hybrid method decreased the total power consumption of the robot's motors, enhancing energy efficiency, which is crucial for industrial applications where energy costs are a concern.

The increasing global prevalence of industrial robots underscores the importance of adopting energy-efficient solutions. With over 3 million robots in operation and half a million new additions annually, optimizing energy usage in robotic systems is imperative. This study illustrates that AI-assisted algorithms can achieve substantial gains in efficiency, even on a small scale. Future research should focus on extending these findings to real-world industrial processes with extended duty cycles to further validate the energy efficiency and overall effectiveness of the hybrid approach. In conclusion, the integration of supervised and reinforcement learning methods offers a promising avenue for enhancing the accuracy, speed, and energy efficiency of robotic systems. The hybrid approach not only improves performance in visual path-following tasks but also addresses critical challenges related to energy consumption, paving the way for more sustainable and efficient industrial automation solutions.

Author contributions. MCB conceived of the presented idea, developed the theory and performed the computations, discussed the results, and contributed to the final article. OA developed the theory and performed the computations, discussed the results, and contributed to the final article.

Financial support. The authors would like to thank the FIRAT University Scientific Research Projects Unit (FUBAP) for their financial support for the current study (Project No:TEKF.21.31).

Competing interests. The authors declare no conflicts of interest exist.

Ethical standard. Not applicable.

References

[1] International Federation of Robotics (IFR), Homepage, https://ifr.org, accessed October 05, 2024.

[2] Y. Lin, H. Zhao and H. Ding, "A data-driven approach for online path correction of industrial robots using modified flexible dynamics model and disturbance state observer," *IEEE/ASME Trans. Mechatron.* **28**(6), 3410–3421 (2023).

[3] S. Raikwar, J. Fehrmann and T. Herlitzius, "Navigation and control development for a four-wheel-steered mobile orchard robot using model-based design," *Comput. Electron. Agric.* **202**, 107410 (2022).

[4] T. A. Khaled, O. Akhrif and I. A. Bonev, "Dynamic path correction of an industrial robot using a distance sensor and an ADRC controller," *IEEE/ASME Trans. Mechatron.* **26**(3), 1646–1656 (2020).

[5] Y. Geng, Y. Zhang, X. Tian and L. Zhou, "A novel 3D vision-based robotic welding path extraction method for complex intersection curves," *Robot. Comput.-Integr. Manuf.* **87**, 102702 (2024).

[6] Z. Chen and S. T. Birchfield, "Qualitative vision-based path following," *IEEE Trans. Robot.* **25**(3), 749–754 (2009).

[7] K. Okamoto, L. Itti and P. Tsiotras, "Vision-based autonomous path following using a human driver control model with reliable input-feature value estimation," *IEEE Trans. Intell. Veh.* **4**(3), 497–506 (2019).

[8] K. Duan, C. W. K. Suen and Z. Zou, "Robot morphology evolution for automated HVAC system inspections using graph heuristic search and reinforcement learning," *Automat. Constr.* **153**, 104956 (2023).

[9] J. Baltes, G. Christmann and S. Saeedvand, "A deep reinforcement learning algorithm to control a two-wheeled scooter with a humanoid robot," *Eng. Appl. Artif. Intell.* **126**, 106941 (2023).

[10] T. Chettibi, "Multi-objective trajectory planning for industrial robots using a hybrid optimization approach," *Robotica* **42**(6), 2026–2045 (2024).

[11] X. Zhang and G. Shi, "Multi-objective optimal trajectory planning for manipulators in the presence of obstacles," *Robotica* **40**(4), 888–906 (2022).

[12] D. Yang, L. Dong and J. K. Dai, "Collision avoidance trajectory planning for a dual-robot system: Using a modified APF method," *Robotica* **42**(3), 846–863 (2024).

[13] S. Katiyar and A. Dutta, "End-effector path tracking of a 14 DOF rover manipulator system in CG-space framework," *Robotica* **42**(1), 265–301 (2024).

[14] X. Tang, H. Zhou and T. Xu, "Obstacle avoidance path planning of 6-DOF robotic arm based on improved A∗ algorithm and artificial potential field method," *Robotica* **42**(2), 457–481 (2024).

[15] Y. Oh, Y. Kim, K. Na and B. D. Youn, "A deep transferable motion-adaptive fault detection method for industrial robots using a residual–convolutional neural network," *ISA Transactions* **128**, 521–534 (2021).

[16] L. Chen, J. Cao, K. Wu and Z. Zhang, "Application of generalized frequency response functions and improved convolutional neural network to fault diagnosis of heavy-duty industrial robot," *Robotics and Computer-Integrated Manufacturing* **73**, 102228 (2022).

[17] M. C. Bingol and O. Aydogmus, "Practical application of a safe human-robot interaction software," *Industrial Robot: The International Journal of Robotics Research and Application* **47**(3), 359–368 (2020).

[18] M. C. Bingol and O. Aydogmus, "Performing predefined tasks using the human–robot interaction on speech recognition for an industrial robot," *Engineering Applications of Artificial Intelligence* **95**, 103903 (2020).

[19] J. Yan and M. Zhang, "A transfer-learning based energy consumption modeling method for industrial robots," *Journal of Cleaner Production* **325**, 29299 (2021).

[20] J. S. Toquica, P. S. Oliveira, W. S. R. Souza, J. M. S. T. Motta and D. L. Borges, "An analytical and a deep learning model for solving the inverse kinematic problem of an industrial parallel robot," *Computers & Industrial Engineering* **151**, 106682 (2021).

[21] B. N. Rath and B. Subudhi, "A robust model predictive path following controller for an autonomous underwater vehicle," *Ocean Engineering* **244**, 110265 (2022).

[22] Y. Zheng, J. Tao, Q. Sun, H. Sun, Z. Chen, M. Sun and G. Xie, "Soft actor–critic based active disturbance rejection path following control for unmanned surface vessel under wind and wave disturbances," *Ocean Engineering* **247**, 110631 (2022).

[23] S. Mondal, R. Ray, S. Reddy and S. Nandy, "Intelligent controller for nonholonomic wheeled mobile robot: A fuzzy path following combination," *Mathematics and Computers in Simulation* **193**, 533–555 (2022).

[24] J. Handler, M. Harker and G. Rath, "Multidimensional path tracking with global least squares solution," *IFAC-PapersOnLine* **53**, 6189–6194 (2020).

[25] A. Maldonado-Ramirez, R. Rios-Cabrera and I. Lopez-Juarez, "A visual path-following learning approach for industrial robots using drl," *Robotics and Computer-Integrated Manufacturing* **71**, 102130 (2021).

[26] Y. Wen and P. R. Pagilla, "A novel 3d path following control framework for robots performing surface finishing tasks," *Mechatronics* **76**, 102540 (2021).

[27] M. C. Bingol, *Development of Artificial Intelligence-Based Self-Programmable Robot Software Compatible with Industry 4.0 using Human-Robot Interaction PhD thesis* (Firat University, Institute of science, 2020).

[28] Y. Yang, Z. Li, X. Yu, Z. Li and H. Gao, "A trajectory planning method for robot scanning system uuuusing mask r-cnn for scanning objects with unknown model," *Neurocomputing* **404**, 329–339 (2020).

[29] A. Almeida, P. Vicente and A. Bernardino, "Where is my hand? deep hand segmentation for visual self-recognition in humanoid robots," *Robotics and Autonomous Systems* **145**, 103857 (2021).

[30] D. K. Bilal, M. Unel, L. T. Tunc and B. Gonul, "Development of a vision based pose estimation system for robotic machining and improving its accuracy using lstm neural networks and sparse regression," *Robotics and Computer-Integrated Manufacturing* **74**, 102262 (2022).

[31] K.-C. Ying, P. Pourhejazy, C.-Y. Cheng and Z.-Y. Cai, "Deep learning-based optimization for motion planning of dual-arm assembly robots," *Computers & Industrial Engineering* **160**, 107603 (2021).

https://doi.org/10.1017/S026357472400170X Published online by Cambridge University Press

[32] W. Zhang, J. Gai, Z. Zhang, L. Tang, Q. Liao and Y. Ding, "Double-dqn based path smoothing and tracking control method for robotic vehicle navigation," *Computers and Electronics in Agriculture* **166**, 104985 (2019).

[33] J. Wang, S. Elfwing and E. Uchibe, "Modular deep reinforcement learning from reward and punishment for robot navigation," *Neural Networks* **135**, 115–126 (2021).

[34] X. Chen, C. Fu and J. Huang, "A deep q-network for robotic odor/gas source localization: Modeling, measurement and comparative study," *Measurement* **183**, 109725 (2021).

[35] Y. Zou, T. Chen, X. Chen and J. Li, "Robotic seam tracking system combining convolution filter and deep reinforcement learning," *Mechanical Systems and Signal Processing* **165**, 108372 (2022).

[36] S. Wen, Z. Wen, Z. Di, H. Zhang and T. Wang, "A multi-robot path-planning algorithm for autonomous navigation using meta-reinforcement learning based on transfer learning," *Applied Software Computing* **110**, 107605 (2021).

[37] M. C. Bingol, "Proximal policy optimization-based path generation of an industrial robot," *International Conference on Engineering, Natural and Applied Science 2021 (ICENAS'21)* (2021) pp. 322–328.

[38] K. Kamali, I. A. Bonev and C. Desrosiers, "Real-Time Motion Planning for Robotic Teleoperation Using Dynamic-Goal Deep Reinforcement Learning," In 2020 17th Conference on Computer and Robot Vision (CRV) (2020) pp. 182–189.

[39] Z. Liu and N. Kubota, "Hybrid Learning Approach Based on Multi-Objective BehaviorCoordination for Multiple Robots," In 2007 International Conference on Mechatronics and Automation (2007) pp. 204–209.

[40] S. F. Desouky and H. M. Schwartz, "A Novel Hybrid Learning Technique Applied to a Self-Learning Multi-Robot System," In 2009 IEEE International Conference on Systems, Man and Cybernetics (2009) pp. 2616–2623.

[41] J. Long, J. Mou, L. Zhang, S. Zhang and C. Li, "Attitude data-based deep hybrid learning architecture for intelligent fault diagnosis of multi-joint industrial robots," *Journal of Manufacturing Systems* **61**, 736–745 (2021).

[42] J. Schulman, F. Wolski, P. Dhariwal, A. Radford and O. Klimov, Proximal policy optimization algorithms (2017).

[43] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver and D. Wierstra, Continuous control with deep reinforcement learning (2015).

[44] J. Schulman, S. Levine, P. Abbeel, M. Jordan and P. Moritz, "Trust region policy optimization," International Conference on Machine Learning (2015) pp. 1889–1897.

Cite this article: M. C. Bingol and O. Aydogmus, "Hybrid learning-based visual path following for an industrial robot", Robotica. https://doi.org/10.1017/S026357472400170X