Development of Deep Learning for Power Energy Optimization in the Industrial Robot System

Borihan Butsanlee [1,*], Watcharin Pongaen [1], Nuttapon Rothong [2], Supawan Ponpitakchai[3], and Songkran U-Thathong [4]

[1] Department of Teacher Training in Mechanical Engineering, Faculty of Technical Education, King Mongkut's University of Technology North Bangkok, Bangkok, Thailand
[2] College of Industrial Technology, King Mongkut's University of Technology North Bangkok, Bangkok, Thailand
[3] Department of Electrical and Computer Engineering, Faculty of Engineering, Naresuan University, Phitsanulok
[4] Autoflexible Advanced Engineering Co.,Ltd, Bangkok, Thailand
Email: s6102017910013@kmutnb.ac.th (B.B.); watcharin.p@fte.kmutnb.ac.th (W.P.); Nuttapon.r@cit.kmutnb.ac.th (N.R.); supawanpo@nu.ac.th (S.P.); frankuthathong.s@gmail.com (S.U.-T.)
*Corresponding author

***Abstract*—This paper established power consumption modeling and motion estimation optimization of industrial robots. We also studied factors affecting the use of electrical energy, such as friction, torque, and electric current. The energy consumption parameters of each coupling can be quantified through the Deep Learning (DL) technique, Scaled Conjugate Gradient (SCG) estimation, or Simulation and experimentation based on the movement posture of a given robot dynamic model to control the robot operation. The robot dynamic model parameters can be identified and expressed in mathematical equations. Electrical energy consumption estimates were analyzed using the SCG technique to compare with the Nonlinear Least Squares (NLS) method using a large dataset of approximately 60,000 samples. The results showed accurate parameter prediction and electrical energy consumption estimation of the robot locomotion pose. The maximum errors in the SCG and NLS methods were 0.89% and 1.54%, respectively. It indicated that the electric energy consumption model using the SCG estimation method is more efficient than the NLS method.**

***Keywords*—robot power consumption, Deep Learning (DL), Nonlinear Least Squares (NLS), Scaled Conjugate Gradient (SCG)**

I. Introduction

Various industries have continuously developed, starting with Industry 1.0 and progressing to Industry 4.0 [1]. These industries have developed technology capabilities to connect through the Internet system's structure, known as the Internet of Things (IoT) concept. Processes used in industrial plants are becoming more digital. As a result, factories are interested in developing strategies in the industry to be more efficient, such as improving production processes, using automation to replace manual systems, or using industrial robots to replace human workers. Industrial robots are more efficient and have specific abilities better than human workers, for example, heavy-lifting work, repetitive work [2], or work too dangerous for humans to do. Due to robots' higher efficiency and capabilities, industries have increasingly used robots in production. However, introducing robots into production results in higher power and energy consumption than before. Because the movement of robots in different poses and directions requires an additional amount of power energy, we developed an idea on how to improve the performance of industrial robots for optimal use to reduce unnecessary power energy use in the production processes by predicting movement [3, 4]. We utilized deep learning algorithms and robot dynamic models [5, 6] to analyze the use of power energy in the robot's movement. The method for estimating power energy used in the study was the Scale Conjugate Gradient (SCG) method to compare with the commonly-used traditional learning process, the nonlinear least squares (NLS) way. We used the root mean squared error (RMSE) method to measure the error in estimating power energy use.

II. Materials

A. The Industrial Robot Digital Twin System

The idea was to improve the efficiency of industrial robots to suit their use and reduce the problem of unnecessary power energy use in the production processes. The researchers, therefore, studied a model for estimating the energy consumption in robots using a digital twin system, as shown in Fig. 1.

To study the deep learning process of robots, we utilized the digital twin system [7, 8] of an industrial robot which covers three main functions as follows:

- Digital robot representation rendered in the 3D environment to exploit the capabilities provided by the robot operating system (ROS) framework [9].
- Semantic representation of the world through implementing a unified data model for the geometrical representation.

Manuscript received September 26, 2023; revised November 11, 2023; accepted December 7, 2023; published April 10, 2024.

- Dynamic updates of the digital twin based on the real-time sensor, such as torque and current resource data obtained from the robot's actual joint.

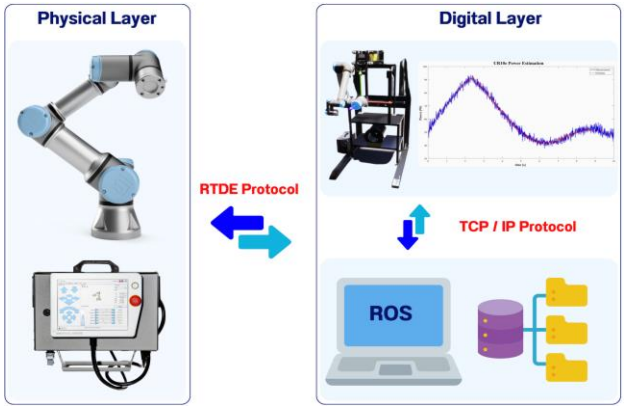


Fig. 1. The concept of estimating power energy in a digital twin system.

The digital twin shown in Fig. 2 is the communication and integration between physical and virtual agents in a Real-Time Data Exchange (RTDE) framework [10]. The RTDE interface offers a mechanism to synchronize external programs with the robot controller over a standard TCP/IP connection without affecting the real-time capabilities of the robot controller. This function can also control robot I/O, plot robot status (Robot trajectory), and communicate with fieldbus drivers (Ethernet/IP).

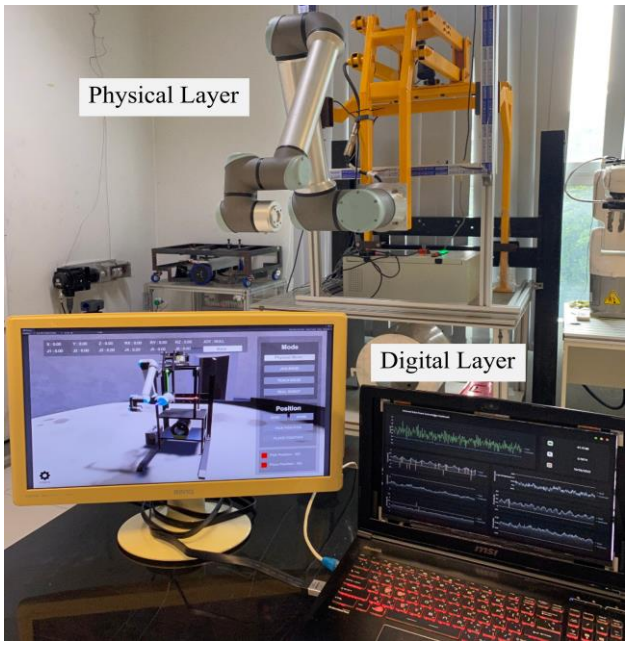


Fig. 2. The system of estimating power energy in a digital twin system.

The above digital twin system is sufficiently generic and exact to be used by most robots and allows for the retrieval of a precise parameter and energy consumption prediction. The utilized industrial robot model first calculates the power energy consumption from the robot joint trajectories [11, 12] (for example, after imposing an end-effector path within a robot simulation tool), and a dynamic model system generates trajectory data. The dynamic model of each joint robot component and motion planning is presented in the following sections.

B. Robot Dynamics Model

The dynamics model is a mathematical representation of the robot's physical behavior. It considers the robot's mass, inertia, joint friction, and other physical properties [13, 14] to predict how it will move in response to various inputs. The dynamics model is typically used in robot control algorithms to generate motion trajectories and ensure that the robot moves accurately and safely. The dynamic model can be represented by differential equations describing the robot's motion [15, 16]. These equations consider the robot's present position, velocity, and forces and torque operating on it.

Based on the dynamic model of industrial robots, it is possible to predict robot motion parameters and energy consumption. The robot's joint friction dynamic model equation describes the relationship between joint torque, position, velocity, and acceleration. Specifically, the dynamic model equation accounts for friction effects, which can arise from various sources such as viscous damping, coulomb friction [17], and static friction [18, 19]. The friction can significantly affect the behavior of a robot and, therefore, needs to be considered in robot control and trajectory planning [20, 21]. The general form of the dynamic model equation of a joint friction robot is shown in Eq. (1).

$$\tau = M(q)\times q'' + C(q,q')\times q' + G(q) \qquad (1)$$

where M(q) is the mass matrix describing the inertial properties of the robot and the relationship between joint acceleration and joint torque, q is a vector of joint positions, q' is a vector of joint velocity, q'' is a vector of joint acceleration, $C(q, q'')$ is the Coriolis and centrifugal term which accounts for the coupling between joint velocity and joint acceleration, $G(q)$ is the gravity term which describes the effect of gravity on the robot and τ is a vector of control torques applied to the joints. To summarize, models of the robot and its environment have information about its kinematics, workspace, and obstacles and constraints caused by the environment.

The covariant Hamiltonian Optimization for Motion Planning (CHOMP) technique presents possible robot motion instructions in this framework [22, 23]. CHOMP was developed to provide smooth, collision-free, high-quality trajectories of complicated robotic systems with six degrees of freedom. CHOMP, in combination with machine learning, can investigate robot motion with the amount of energy on the set of motion lines in the working area. All of these are limited to the robot kinetics, where each trajectory corresponds to the endpoint of the robot motion, and the objective function serves to push these workspace paths away from the obstruction. The machine learning approach predicted and optimized the joint friction effect on robot motion.

C. Power Estimation Method

This section explains how to estimate the power energy consumption of the robot in each movement [24]. The

energy consumption for each motor joint can be calculated from the torque generated by the robot moves according to Eqs. (2) and (3).

$$\tau(t) = K_t I(t) \tag{2}$$

$$V(t) = RI(t) + K_b q'(t) \tag{3}$$

where K_t is the diagonal matrix of motor torque constants, $I(t)$ is the current of the motor, K_b is the diagonal matrix of back-emf constants, R is the diagonal matrix of motor winding resistances, and $q'(t)$ is motor speed vectors.

The effect of induction is negligible, as it has been widely proven in the literature. The voltage-current product can represent the electrical power drawn by the robot as follows:

$$W_m(t) = V(t) \times I(t) \tag{4}$$

where W_m is the electrical power drawn from all six actuators, it is simple to calculate the total consumption of the robot as the sum of the six typical motor applications in Eq. (5).

$$W(t) = \sum_{i=1}^{6} W_{m,i}(t) \times W(t) \tag{5}$$

Using robot motion planning data, NLS and SCG with deep learning were applied in power estimation. The theory of NLS and SCG will be explained in the next section.

D. Deep Learning

Deep learning [25, 26] is a part of machine learning methods based on neural networks [27] and feature learning. The learning can be in the form of supervised learning, semi-supervised learning, or unsupervised learning. The meaning of the word "deep" here comes from having more efficient layers of networks, more accessible learning, and a clearer understanding of the structure.

The basis of deep learning is an algorithm that attempts to create a model to represent data at a higher level by creating a data architecture composed of many small structures, as shown in Fig. 3. Each design is obtained by non-linear transformation, and the variables can be described as follows.

Input Layer: The data fed to the model is loaded into the input layer from external sources. It is the only visible layer in the complete neural network architecture that obtains the full information from the outside world without any computation.

Output Layer: The output layer takes input from preceding hidden layers and makes a final prediction based on the model's learnings. It is the most critical layer where we get the result.

Hidden Layers: The hidden layers are what make deep learning what it is today. They are intermediate layers that do all the computations and extract the features from the data. Multiple interconnected hidden layers can account for searching different hidden features in the data.

Weights: Every interconnection between the neurons in the consecutive layers has an associated weight. It indicates the significance of the connection between the neurons in discovering some data pattern, which helps predict the neural network's outcome. The higher the weight values are, the higher the significance level is.

Bias: Bias helps shift the activation function to the left or right, which can be critical for better decision-making. Its role is analogous to the part of an intercept in the linear equation.

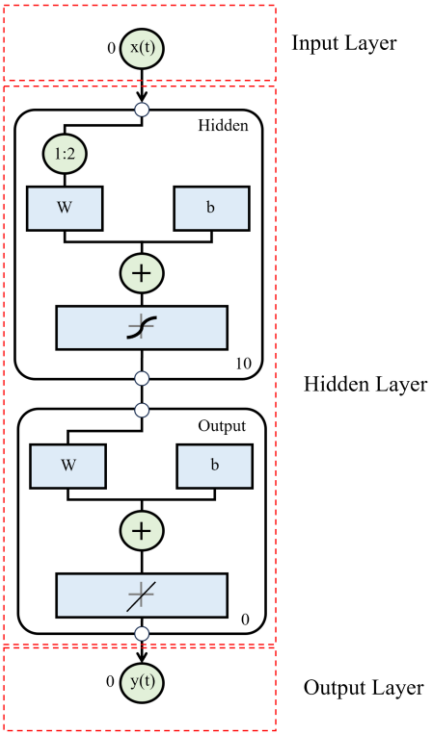


Fig. 3. Deep Learning architecture.

The Scaled Conjugate Gradient (SCG) algorithm [28, 29] is a common optimization technique used for training artificial neural networks chosen for parameter estimation in this project. It is an efficient algorithm that can quickly find a satisfying solution without requiring extensive computational resources.

In the SCG algorithm, the step size is a quadratic approximation of the error function to make it more robust and independent of user-defined parameters.

The step size is estimated by using different approaches. The second-order term is calculated as shown in Eq. (6).

$$\bar{s}_k = \frac{E'(\bar{w}_k + \sigma_k \bar{p}_k) - E'(\bar{w}_k)}{\sigma_k} + \lambda_k \bar{p}_k \tag{6}$$

where, λ_k is scalar and adjusted each time according to the sign of δ_k.

The step size formula is shown in Eq. (7).

$$\alpha_k = \frac{\mu_k}{\delta_k} = \frac{-\bar{p}_j^T E''_{qw}(\bar{y}_1)}{-\bar{p}_j^T E''_{qw}(\bar{w})\bar{p}_j} \tag{7}$$

where, $\bar{w}$ is the weight vector in space R^n, $E(\bar{w})$is the global error function, $E''(\bar{w})$ is the gradient of error, $E'_{qw}(\bar{y}_1)$ is the quadratic approximation of error function and $\bar{p}_1, \bar{p}_2, ..., \bar{p}_k$ are the set of non-zero weight vectors.

λ_k is to be updated such that,

$$\bar{\lambda}_k = 2(\lambda_k - \frac{\delta_k}{|p_k|^2}) \tag{8}$$

If $\Delta_k > 0.75$, then $\lambda_k = \lambda_k/4$

If $\Delta_k < 0.25$, then $\lambda_k = \lambda_k + \frac{\delta_k(1-\Delta_k)}{|p_k|^2}$

where, Δ_k is a comparison parameter given by,

$$\Delta_k = 2\delta_k[E(\bar{w}_k) - E(\bar{w}_k + \alpha_k \bar{p}_k)] / \mu_k^2 \tag{9}$$

Initial, the values are set as, $0 < \sigma \leqslant 10^{-4}$, $0 < \lambda_l \leq 10^{-6}$ and $\bar{\lambda} = 0$.

The SCG algorithm calculates the learning rate of the error caused by the slope of the surface. It allows us to adjust the learning rate value to suit the current direction. This adaptive learning is a crucial feature of the SCG algorithm, making it highly efficient and effective for optimizing neural networks.

E. Nonlinear Least Squares Parameter Estimation

Nonlinear Least Squares (NLS) [30, 31] is an optimization technique used to simulate regression datasets with nonlinear properties in data fitting and parameter estimation to find the set for nonlinear joint friction that minimizes the difference between observed data and model predictions. The nonlinear least squares method aims to find the values of the dynamic friction parameters. Nonlinear functions can take many forms depending on the problem. For example, it could be a polynomial logistic function, sine function, or other nonlinear functions related to the independent and dependent variables. The optimization algorithms are typically Gauss-Newton or Levenberg-Marquardt algorithms [32]. Gauss-Newton algorithms were selected. They are equivalent to the reduced sum of the squared function values. It is an extension of Newton's method of minimizing nonlinear functions, as the sum of the squares must not be negative. From the algorithm, it can be explained that when given m functions $r = (r1, ..., rm)$ of n variables $\beta = (\beta 1, ..., \beta n)$, with $m \geq n$ Gauss–Newton algorithm iteratively finds the value of the variables that minimize the sum of squares shown in Eq. (10).

$$S(\beta) = \sum_{i=1}^{m} r_i(\beta)^2 \tag{10}$$

III. Methods and Results

Develop a system for estimating the electrical energy consumption of robots using deep learning to analyze the robot's movement with the minor electrical energy consumption. The robot dynamic model is used to determine the robot's movement. The Newton-Euler formulation is Newton's second law of motion, which describes dynamic systems in terms of force and momentum. The equations incorporate all the details and moments acting on the individual robot links, including the coulomb coefficient, friction force, robot inertia, and electrical parameters between the links.

In this paper, the estimation of electrical energy consumption is developed from the NLS conventional method to the SCG deep learning method. The SCG method learns friction, inertia, and various unknown parameters. However, the traditional way is to find multiple values from calculation methods by the researcher and use the weights to the equation.

The process of determining the robot's electrical energy usage starts with the robot dynamic model to know the behavior of the robot movement in each pose. Each pose can be explained in an example shown in Fig. 4. The starting point of the robot is at position S. The robot moves with the movement pose so that it can move to the end position as defined at position F.

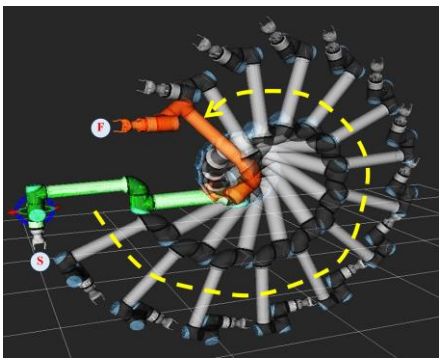
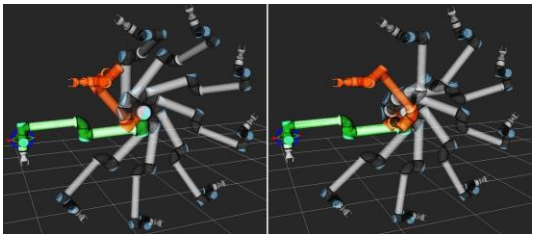


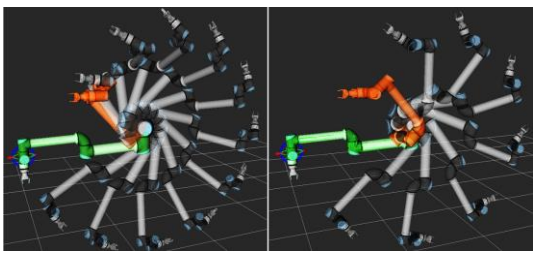
Fig. 4. The robot's movement in pose 1.



(a) (b)

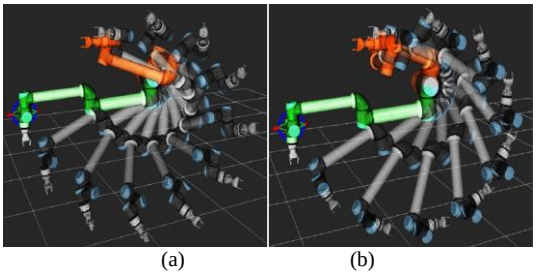
Fig. 5. The robot's movement in (a) poses 2 and (b) poses 3.

Figs. 5–7 show the robot's movement behaviour in each pose the robot can move.



(a) (b)

Fig. 6. The robot's movement in (a) poses 4 and (b) poses 5.



(a) (b)

Fig. 7. The robot's movement in (a) poses 6 and (b) poses 7.

When the robot is moving, the responding position of each joint in the robot could be presented in Fig. 8.

When the position changes, it will cause the angular velocity occurring to each joint in the robot as shown in Fig. 9.

The angular velocity occurrence in each joint causes the robot to experience angular acceleration, as shown in Fig. 10.

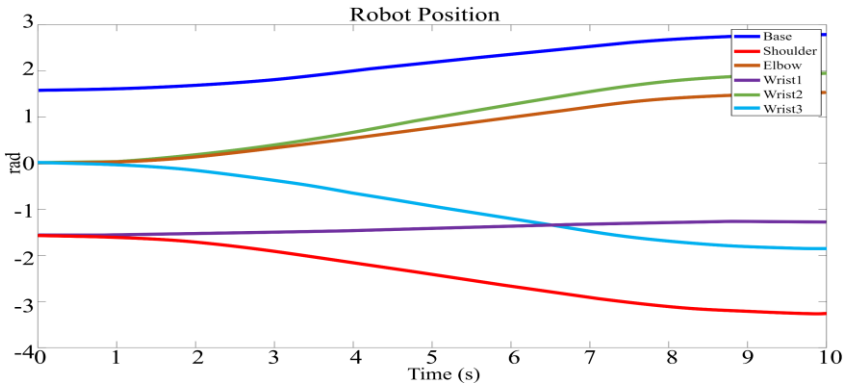


Fig. 8. Response position movement of the six joints of the robot.

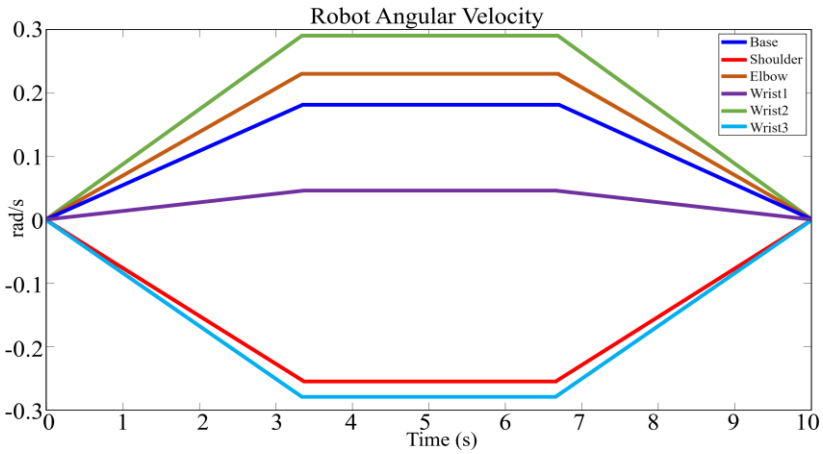


Fig. 9. Response angular velocity movement of the robot.

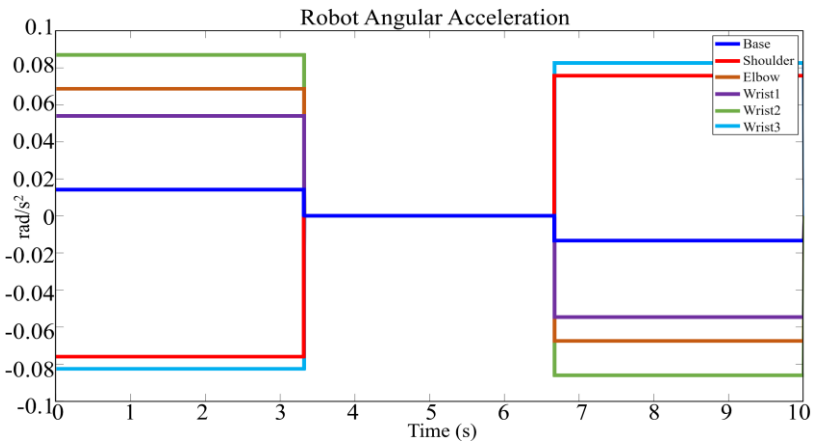


Fig. 10. Response angular acceleration movement of the robot.

The robot can move to different positions as specified. Data from the motion planning algorithm were analyzed to determine the torque applied to the robot's six-axis joints. The torque occurring in the robot joints can be found in Eq. (11).

$$\tau = (I \times \alpha) + (B \times \omega) + G \quad (11)$$

where τ is the joint torque required to move the joint, I is the moment of inertia of the joint, α is the angular acceleration of the joint, B is the coefficient of friction of the joint, ω is the angular velocity of the joint, and G is the gravitational force acting on the joint.

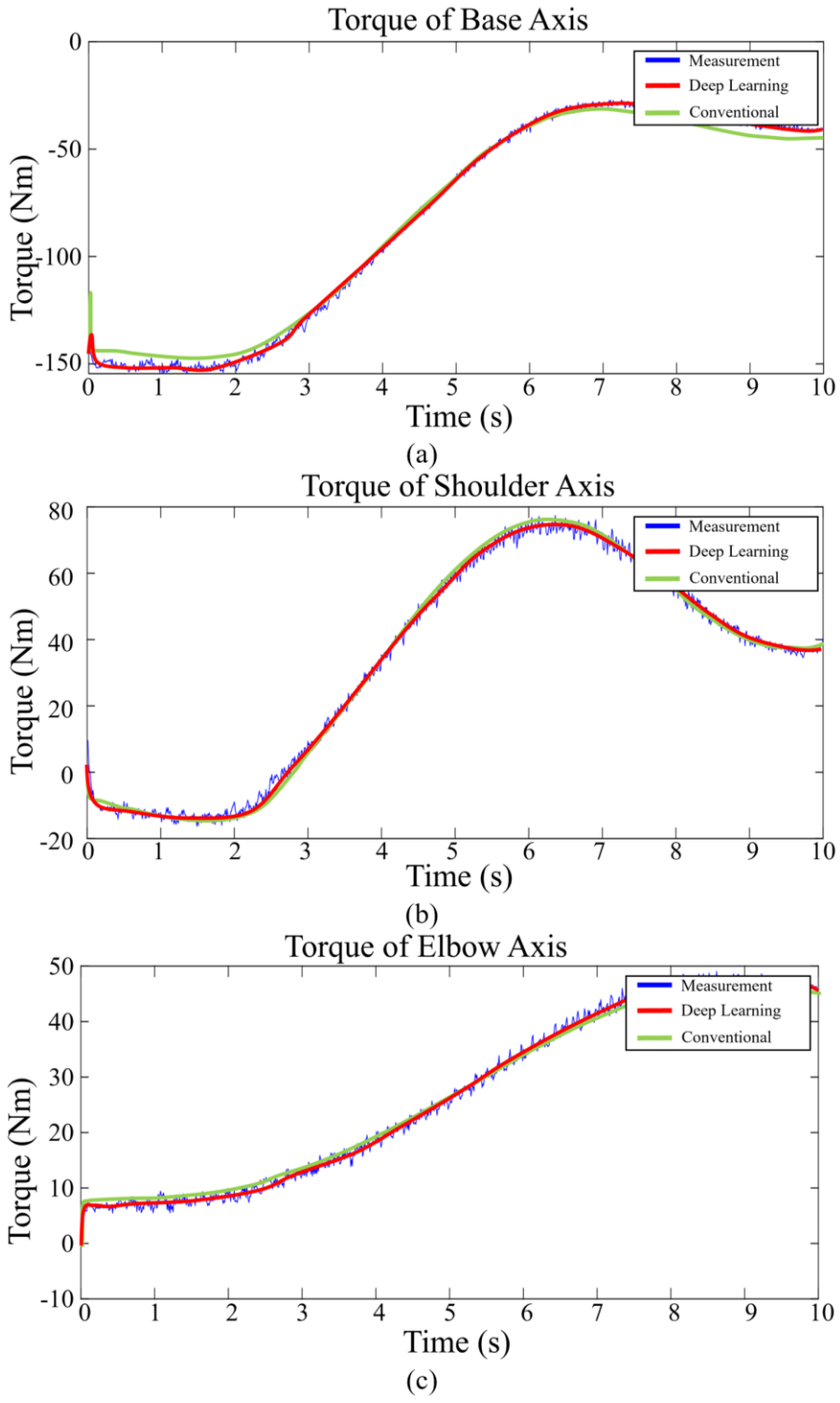


Fig. 11. Response torque movement of the (a) base axis, (b) shoulder axis, and (c) elbow axis.

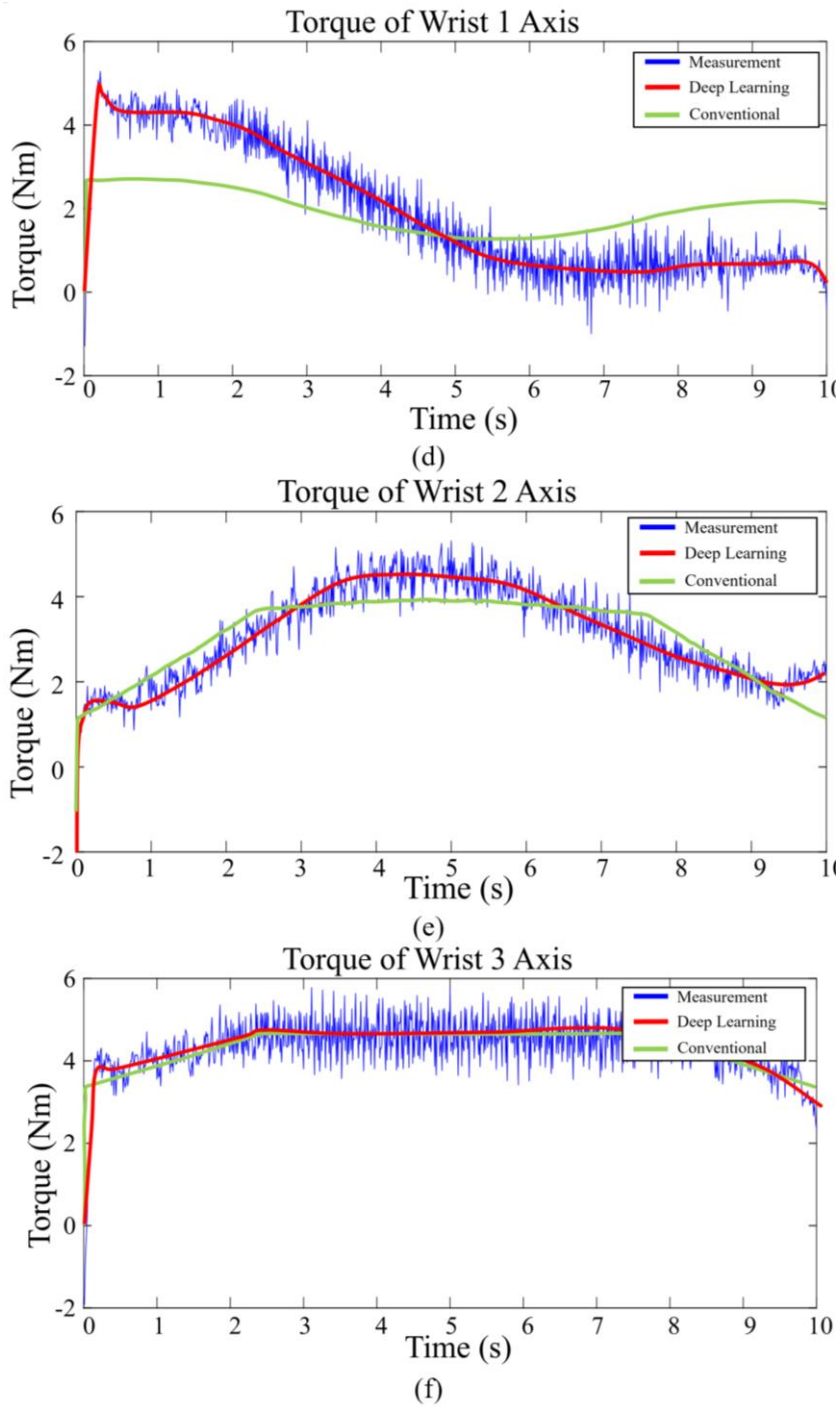


Fig. 12. Response torque movement of the (d) wrist1 axis, (e) wrist2 axis, and (f) wrist3 axis.

The torque applied to each robot joint is shown in Figs. 11–12. This graph shows a comparison of the torque produced between deep learning and conventional models. They can be calculated to yield the current applied to each joint. The robot's extension can be obtained from Eq. (12).

$$i = \frac{1}{k_m} \tau_{robot} \tag{12}$$

where i is the current of the motor joint, τ_{robot} is the constant torque of the motor, and k_m is the torque of the robot joints.

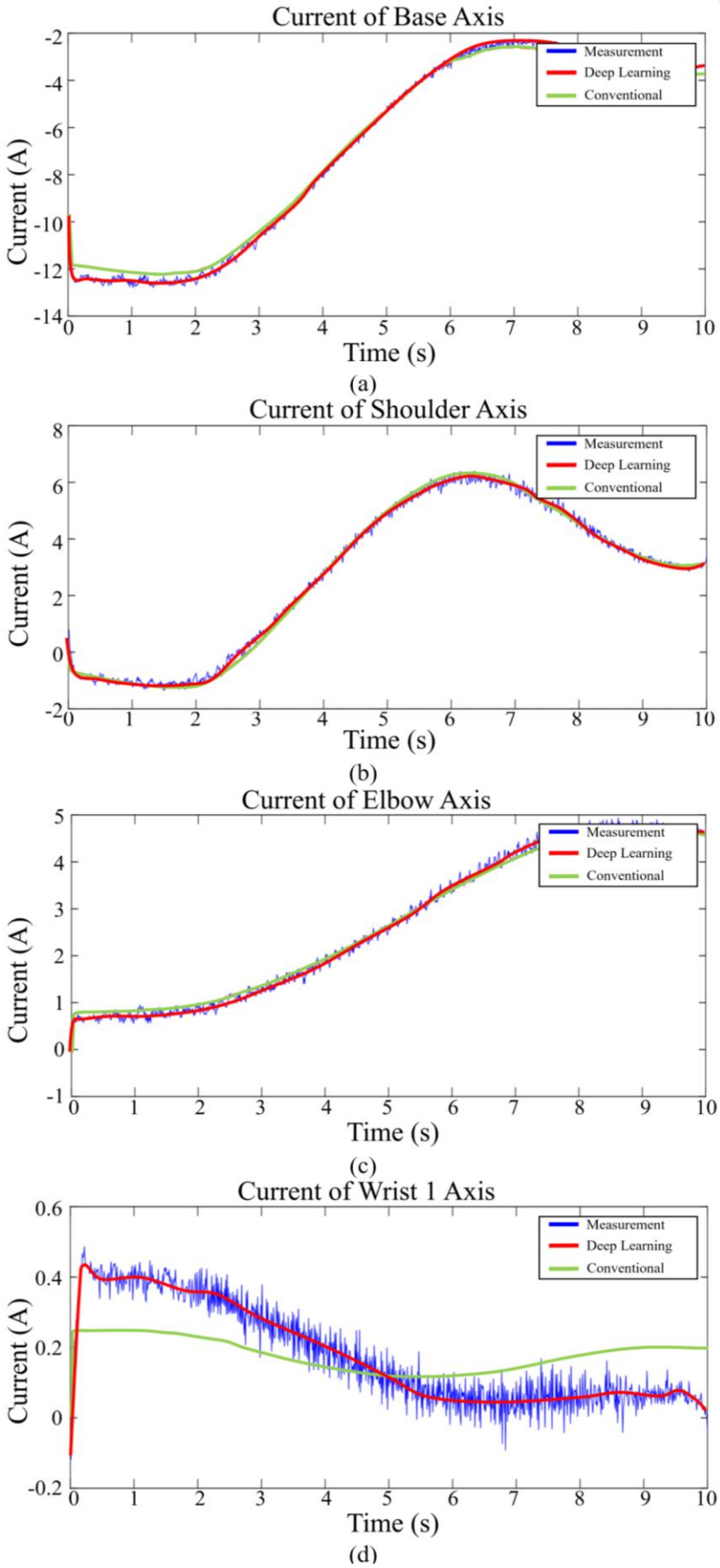


Fig. 13. Response current of the (a) base axis, (b) shoulder axis, (c) elbow axis, and (d) wrist1 axis.

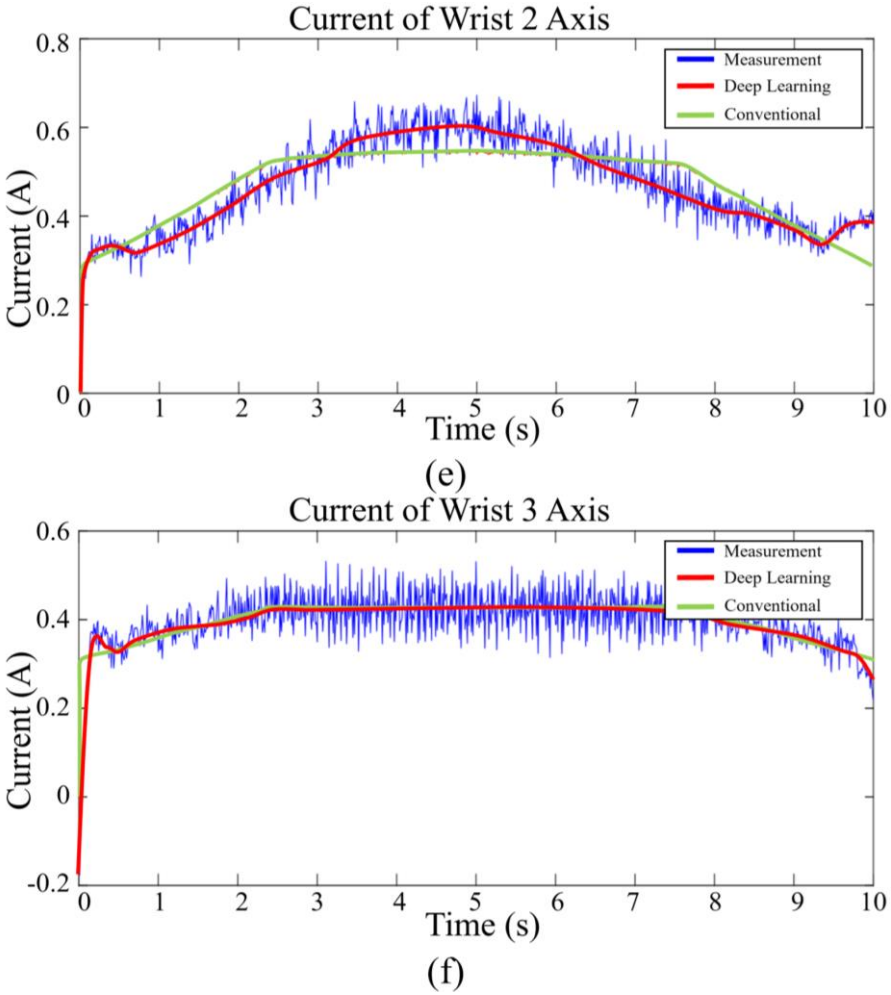


Fig. 14. Response current of the (e) wrist2 axis, and (f) wrist3 axis.

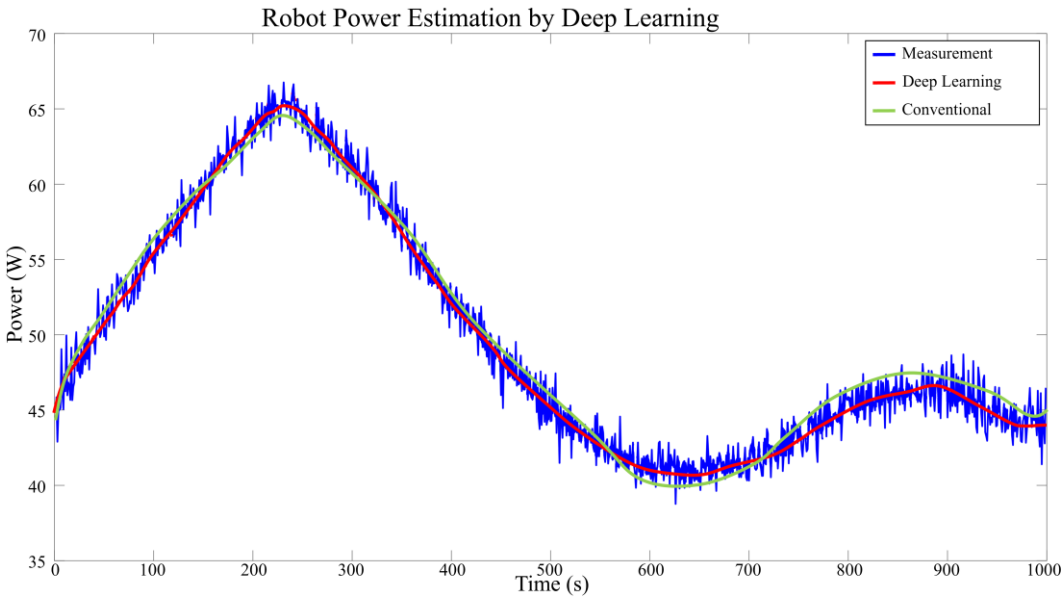


Fig. 15. Power energy consumption of the robot.

Figs. 13 and 14 show the response of the electric current used in each robot joint. This graph compares the torque produced between deep learning and conventional models. The current that occurs in each core will have different current values.

Once the electric current used in each robot joint is known, the resulting electric current value can be

calculated to yield the electrical energy used while the robot is moving, as shown in Eq. (13).

$$P_r = P_c + \sum_{i=1}^{6}\left(L_i i_i \frac{di_i}{dt} + R_i i_i^2 + k_m \dot{\theta}_i i_i + k_d \right) \quad (13)$$

where i_i is the motor current angular velocity, R_i is the motor winding resistance, and k_m is the motor speed constant. The unknown variables are as follows: P_c is the total electronic power constant, L_i is the inductance of the motor winding, and k_d is the drive motor constant.

The energy consumption is then the integral of the robot power (P) over time as shown in Eq. (14).

$$E_r = \int P_r dt \quad (14)$$

Fig. 15 shows the electrical energy consumption measured by the industrial robot and the electrical energy estimated from the industrial robot movement pose. This graph compares the torque produced between deep learning and conventional models.

Based on the method for estimating the electrical energy consumption in industrial robots, we designed 8 and 10 movement positions at the starting points [0, 0, 0, 0, 0, 0] and [0, –90, 0, –90, 0, 0] respectively.

Tables I and II show the results of experiments using deep learning methods to estimate electrical energy consumption in industrial robots based on the theory of scaled conjugate gradient, a new learning characteristic with the traditional nonlinear least squares theory. It estimated the different levels of electrical energy consumption in each movement pose.

TABLE I. POWER CONSUMPTIONS FOR ROBOT TARGET POSITION

Robot Instruction Poses	Power Consumptions for Robot Target Position (Start point [0,0,0,0,0,0])									
	Position 1		Position 2		Position 3		Position 4		Position 5	
	Error NLS (CV)**	Error SCG (DL)***	Error NLS (CV)**	Error SCG (DL)***	Error NLS (CV)**	Error SCG (DL)***	Error NLS (CV)**	Error SCG (DL)***	Error NLS (CV)**	Error SCG (DL)***
1	1.44%	0.48%	2.12%	1.33%	0.11%	0.45%	0.20%	0.36%	0.90%	0.78%
2	0.08%	0.019%	0.17%	0.33%	1.66%	0.89%	3.49%	1.56%	0.31%	0.12%
3	0.16%	0.18%	1.60%	0.24%	2.87%	1.42%	0.22%	0.12%	1.38%	0.58%
4	0.65%	0.07%	0.16%	0.21%	3.34%	1.23%	-*	-*	0.61%	0.72%
5	0.27%	0.31%	1.80%	1.03%	1.32%	0.89%	-*	-*	1.36%	1.08%
6	0.74%	0.44	-*	-*	1.64%	1.27%	1.31%	0.96%	1.63%	1.04%
7	1.39%	0.97%	2.44%	1.42%	0.38%	0.42%	1.64%	1.02%	*	-*
8	1.04%	0.85%	1.26%	0.98%	0.98%	0.56%	1.29%	1.08%	-*	-*
Average	**0.72**%	**0.41**%	**1.36**%	**0.79**%	**1.54**%	**0.89**%	**1.36**%	**0.85**%	**1.03**%	**0.72**%

Note * - is an impediment to the movement of the robot that cannot be measured, ** CV is conventional method, *** DL is deep learning method.

TABLE II. POWER CONSUMPTIONS FOR ROBOT TARGET POSITION

Robot Instruction Poses	Power Consumptions for Robot Target Position (Start point [0,-90,0,-90,0,0])									
	Position 6		Position 7		Position 8		Position 9		Position 10	
	Error NLS (CV)**	Error SCG (DL)***	Error NLS (CV)**	Error SCG (DL)***	Error NLS (CV)**	Error SCG (DL)***	Error NLS (CV)**	Error SCG (DL)***	Error NLS (CV)**	Error SCG (DL)***
1	0.84%	0.41%	1.38%	0.92%	1.12%	0.84%	0.16%	0.26%	0.92%	0.89
2	1.50%	1.05%	0.14%	0.26%	0.76%	0.27%	0.63%	0.45%	0.12%	0.14
3	0.28%	0.45%	1.00%	0.74%	0.86%	0.43%	1.26%	0.89%	1.73%	1.05
4	1.12%	0.97%	0.19%	0.03%	1.09%	0.98%	-*	-*	0.70%	0.56
5	0.01%	0.08%	0.13%	0.08%	0.06%	0.13%	-*	-*	0.27%	0.34%
6	2.97%	1.27%	-*	-*	1.11	0.96%	0.72%	0.14%	1.39%	1.08%
7	1.82%	1.32%	1.38%	1.26%	2.00%	1.29%	0.45%	0.31%	-*	-*
8	0.19%	0.47%	2.73	1.08%	0.40%	0.62%	1.57%	1.05%	-*	-*
Average	**1.09**%	**0.85**%	**0.99**%	**0.62**%	**0.93**%	**0.69**%	**0.80**%	**0.52**%	**0.86**%	**0.68**%

Note * - is an impediment to the movement of the robot that cannot be measured, ** CV is conventional method, *** DL is deep learning method.

The method used to calculate the error was the root mean squared error method (RMSE) [33, 34]. The RMSE method computes the square root of the mean square of the total error. It is a primary method used to estimate the cost of electrical energy usage. The value can be calculated from the Eq. (15).

$$RSME = \sqrt{\frac{1}{n}\sum_{i=1}^{n} S_i - O_i^2} \quad (15)$$

where O_i is the observations, S_i is the predicted values of a variable, and n is the number of observations available for analysis.

The estimation of electrical energy consumption was evaluated by calculation from the training and testing data sets, as shown in Fig. 16. For the training dataset, the maximum and minimum percentage SCG errors were 0.89% and 0.41%, and for the training data set, the maximum and minimum rate NLS errors were 1.54% and 0.72%, respectively.

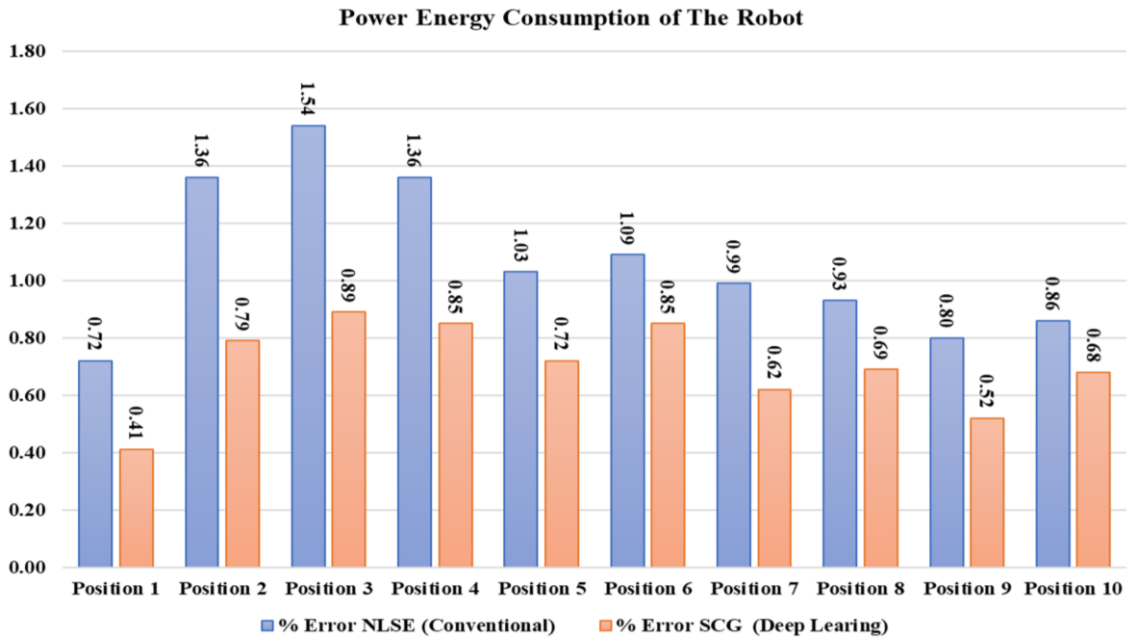


Fig. 16. Power energy consumption of the robot in each position.

IV. Discussions

The study used the SCG technique as part of the deep learning method to estimate the electric power consumption of robots and compare them with results obtained from the NLS technique in the traditional learning method. The model used in the study analyzed the robot's electrical energy consumption to generate the robot's locomotion model by performing integrated calculations of the energy profile. The Newton-Euler method used in the robot dynamics model is a recursive algorithm designed for implementing computations. Using a large dataset of approximately 60,000 samples, motor current could be predicted with a maximum error of 9.88%.

The system to estimate the electrical energy consumption of robots using the SCG deep learning method performed accurately with a maximum error of 0.89%, considerably less than an error of 1.54% obtained from the NLS. Based on a previous study, the reasons for model errors were due to sensor noise and temperature variations during robot operation with an average viscous friction coefficient of approximately 20% [35].

V. Conclusion

Like other intelligent machines, industrial robots are essential to industrial robots and necessary for many industrial applications. Energy consumption has, therefore, become a primary focus for robot manufacturers. The solution to machine expansion is to optimize electrical energy use. The deep learning evaluation method called scaled conjugate gradient was shown to predict the best pose for robot locomotion. Reducing the cost of electrical energy consumption may lead to the optimal performance of the robot in the production processes.

The proposed method aimed to predict industrial robots' electrical energy consumption, and the six joints' design motion profiles were used as inputs. The joint parameters were included to examine the relative movement between the positions and the eight possible orders. The predictions were evaluated based on the forecast for the accuracy of the parameters and the fault prediction. This method was shown to predict the development of industrial robot energy consumption in advance. The study's results can also be applied to analyze and determine the proper duration of robot maintenance and the wear and tear of parts inside the robot.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Borihan Butsanlee conducted the research and wrote the paper; Borihan Butsanlee, Watcharin Pongaen, and

Supawan Ponpitakchai analyzed the data; Nuttapon Rothong and Songkran U-Thathong designed the hardware; all authors had approved the final version.

REFERENCES

[1] A. G. Frank, L. S. Dalenogare, and N. F. Ayala, "Industry 4.0 technologies: Implementation patterns in manufacturing companies," *International Journal of Production Economics*, vol. 210, pp. 15–26, Apr. 2019.

[2] K. T. Andrzejewski, M. P. Cooper, C. A. Griffiths, and C. Giannetti, "Optimisation process for robotic assembly of electronic components," *The International Journal of Advanced Manufacturing Technology*, vol. 99, no. 9–12, pp. 2523–2535, Sep. 2018.

[3] L. C. Brito, G. A. Susto, J. N. Brito, and M. A. V. Duarte, "An explainable artificial intelligence approach for unsupervised fault detection and diagnosis in rotating machinery," *Mechanical Systems and Signal Processing*, vol. 163, 108105, Jan. 2022.

[4] M. Pellicciari, G. Berselli, F. Leali, and A. Vergnano, "A method for reducing the energy consumption of pick-and-place industrial robots," *Mechatronics*, vol. 23, no. 3, pp. 326–334, Apr. 2013.

[5] Y. Han, J. Wu, C. Liu, and Z. Xiong, "An iterative approach for accurate dynamic model identification of industrial robots," *IEEE Transactions on Robotics*, vol. 36, no. 5, pp. 1577–1594, Oct. 2020.

[6] J. Dong, J.-X. Xu, Q. Zhou, J.-W. Zhu, and L. Yu, "Dynamic identification of industrial robot based on nonlinear friction model and LS-SOS algorithm," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–12, Jan. 2021.

[7] R. Rolle, V. Martucci, and E. Godoy, "Architecture for digital twin implementation focusing on industry 4.0," *IEEE Latin America Transactions*, vol. 18, no. 5, pp. 889–898, May 2020.

[8] M. Schluse, M. Priggemeyer, L. Atorf, and J. Rossmann, "Experimentable digital twins—Streamlining simulation-based systems engineering for industry 4.0," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 4, pp. 1722–1731, Apr. 2018.

[9] C. Hu, W. Dong, Y. Yang, H. Shi, and G. Zhou, "Runtime verification on hierarchical properties of ROS-based robot swarms," *IEEE Transactions on Reliability*, vol. 69, no. 2, pp. 674–689, Jun. 2020.

[10] V. Maru, S. Nannapaneni, K. Krishnan, and A. Arishi, "Deep-learning-based cyber-physical system framework for real-time industrial operations," *Machines*, vol. 10, no. 11, 1001, Oct. 2022.

[11] K. Paes, W. Dewulf, K. V. Elst, K. Kellens, and P. Slaets, "Energy efficient trajectories for an industrial ABB robot," *Procedia CIRP*, vol. 15, pp. 105–110, 2014.

[12] L. Tian and C. Collins, "An effective robot trajectory planning method using a genetic algorithm," *Mechatronics*, vol. 14, no. 5, pp. 455–470, Jun. 2004.

[13] M. Gautier. (1990). Numerical calculation of the base inertial parameters of robots. [Online]. Available: https://ieeexplore.ieee.org/document/126126

[14] J. Wu, J. Wang, and Z. You, "An overview of dynamic parameter identification of robots," *Robotics and Computer-Integrated Manufacturing*, vol. 26, no. 5, pp. 414–419, Oct. 2010.

[15] A. D. Dragan, K. Lee, and S. S. Srinivasa, "Legibility and predictability of robot motion," in *Proc. 2013 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, Mar. 2013.

[16] M. Bennewitz, W. Burgard, G. Cielniak, and S. Thrun, "Learning motion patterns of people for compliant robot motion," *The International Journal of Robotics Research*, vol. 24, no. 1, pp. 31–48, Jan. 2005.

[17] N. Farhat, V. Mata, Á. Page, and M. D. Rodríguez, "Dynamic simulation of a parallel robot: Coulomb friction and stick–slip in robot joints," *Robotica*, vol. 28, no. 1, pp. 35–45, Apr. 2009.

[18] A. C. Bittencourt and S. Gunnarsson, "Static friction in a robot joint—Modeling and identification of load and temperature effects," *Journal of Dynamic Systems, Measurement, and Control*, vol. 134, no. 5, Jul. 2012.

[19] X. Tu, Y. Zhou, P. Zhao, and X. Cheng, "Modeling the static friction in a robot joint by genetically optimized BP neural network," *Journal of Intelligent & Robotic Systems*, vol. 94, no. 1, pp. 29–41, Mar. 2018.

[20] A. Gasparetto, P. Boscariol, A. Lanzutti, and R. Vidoni, "Trajectory planning in robotics," *Mathematics in Computer Science*, vol. 6, no. 3, pp. 269–279, Aug. 2012.

[21] K. G. Shin and N. D. McKay, "Minimum cost trajectory planning for industrial robots," *Control and Dynamic Systems*, pp. 345–403, Jan. 1991.

[22] M. Zucker *et al*., "CHOMP: Covariant Hamiltonian optimization for motion planning," *The International Journal of Robotics Research*, vol. 32, no. 9–10, pp. 1164–1193, Aug. 2013.

[23] B. You, Z. Li, L. Ding, H. Gao, and J. Xu, "A new local path planning approach based on improved dual covariant Hamiltonian optimization for motion planning method," *Advances in Mechanical Engineering*, vol. 11, no. 5, May 2019. https://doi.org/10.1177/168781401985100

[24] I. Palomba, E. Wehrle, G. Carabin, and R. Vidoni, "Minimization of the energy consumption in industrial robots through regenerative Drives and optimally designed compliant elements," *Applied Sciences*, vol. 10, no. 21, 7475, Oct. 2020.

[25] A. L. Karn *et al*., "ICACIA: An Intelligent Context-Aware framework for COBOT in defense industry using ontological and deep learning models," *Robotics and Autonomous Systems*, vol. 157, 104234, 2022.

[26] R. J. Mammone and Y. Y. Zeevi, *Rutgers University, Center for Computer Aids for Industrial Productivity, Neural Networks: Theory and Applications*, Boston: Academic Press, 1991.

[27] H. Mohsen *et al*., "Classification using deep learning neural networks for brain tumors," *Future Computing and Informatics Journal*, vol. 3, no. 1, pp. 68–71, 2018.

[28] B. Rana, Y. Singh, and P. K. Singh, "A systematic survey on internet of things: Energy efficiency and interoperability perspective," *Transactions on Emerging Telecommunications Technologies*, Dec. 2020.

[29] L. Caballero, Á. Perafan, M. Rinaldy, and W. Percybrooks, "Predicting the energy consumption of a robot in an exploration task using optimized neural networks," *Electronics*, vol. 10, no. 8, p. 920, Apr. 2021.

[30] G. Simon, R. Pintelon, L. Sujbert, and J. Schoukens, "An efficient nonlinear least square multisine fitting algorithm," *IEEE Transactions on Instrumentation and Measurement*, vol. 51, no. 4, pp. 750–755, Aug. 2002.

[31] F. J. Lin, H. C. Chiang, J. K. Chang, and Y. R. Chang, "Intelligent wind power smoothing control with BESS," *IET Renewable Power Generation*, vol. 11, no. 2, pp. 398–407, Feb. 2017.

[32] W. R. Esposito and C. A. Floudas, "Gauss–newton method: Least squares, relation to newton's method," *Springer eBooks*, pp. 1129–1134, Jan. 2008.

[33] Z. Lin *et al*., "Developing a novel gaussian process model predictive controller to improve the energy efficiency and tracking accuracy of the pressure servo control system," *Journal of Cleaner Production*, pp. 138057–138057, Jul. 2023.

[34] C. J. Willmott and K. Matsuura, "On the use of dimensioned measures of error to evaluate the performance of spatial interpolators," *International Journal of Geographical Information Science*, vol. 20, no. 1, pp. 89–102, Jan. 2006.

[35] A. Raviola *et al*., "Effects of temperature and mounting configuration on the dynamic parameters identification of industrial robots," *Robotics*, vol. 10, no. 3, p. 83, 2021.