Machine Learning for Optimized Field-Oriented Control of PMSM Drives

Danial Ahmad, Muhammad Umer, Hassan Rizwan, Syeda Fatima Zahra
School of Electrical Engineering and Computer Science, NUST
CMS ID: 331388, 345834, 335753, 334379
Email: {dahmad, mumer, hrizwan, szahra}.bee20seecs@seecs.edu.pk

***Abstract*—The field-oriented control (FOC) strategy of a permanent magnet synchronous motor (PMSM) in a simplified form is based on PI-type controllers. Due to the nonlinear nature of the PMSM model's description equations under the typical conditions of a relatively wide variation in load torque and a high dynamic PMSM speed reference, these controllers offer limited performance in addition to their low complexity. A reinforcement learning process control algorithm is used to enhance the performance of the PMSM control system without using controllers with more complex mathematical descriptions. To improve the performance of PMSM control system without complicating the mathematical complexity of the model, a reinforcement learning process control technique is implemented. A Twin Delayed Deep Deterministic Policy Gradient (TD3) algorithm is used to perform correction signals to inner-control loop (current) of the speed controller. The performance gains are supported by numerical simulations carried out in the MATLAB/Simulink environment by implementing both the suggested agent and a conventional PI controller. We also draw the conclusion that the TD3 agent responds to sudden load variations more steadily than a conventional controller.**

***Index Terms*—machine learning, reinforcement learning, field oriented control, permanent magnet synchronous motor, space vector modulation**

I. Introduction

The PMSM's beneficial characteristics, such as its compact size, low harmonic distortion, high torque density, and simple cooling methods, have increased interest in the study of the PMSM and its applications over the past ten years. The fact that the PMSM can be applied to robotics, electric drives, computer peripherals, numerically controlled machines, the aerospace industry, and other fields also contributes to its growing popularity [1] [2] [3] [4] [5] [6] [7].

Naturally, researchers who are involved in the study and development of applications in the aforementioned disciplines have also been particularly interested in the study of PMSM control. The FOC-type control structure can therefore be described, in which, a PI-type controller is utilised for the outer PMSM rotor speed control loop and hysteresis ON/OFF-type controllers are used for the inner current control loop.

The intelligent control system plays a unique role in the advancement of control systems. Reinforcement learning agents, among other intelligent control systems, are distinguished by the fact that they are dependent upon the input signal containing information about the system's state, rather than the mathematical description of the controlled system. Control actions are provided to optimise a reward function made up of signals containing data on the controlled process.

The Twin-Delayed Deep Deterministic Policy Gradient (TD3) agent, an improvement and extension of the DDPG agent [8], is the type of agent used. Additionally, based on the fact that the FOC-type control structure offers the best performance of the control system, this report also shows how using an RL agent can improve the performance of this control structure.

For the control of an industrial process, RL is advised rather than other types of machine learning, even if they can estimate characteristics and parameters of the machine under study better. This is due to the fact that RL agents are updated based on states, rather than estimation and fitting of data. Additionally, the TD3 agent used in this report considered to be the most suitable agent for industrial process control due to a lower relative complexity, and a higher convergence rate.

Lastly, numerical simulations in the MATLAB/Simulink environment are used to validate the performance gains and demonstrate how TD3 agent improves the performance of the PMSM control system even when faced with parametric uncertainties.

II. Literature Review

Research has shown that PMSM is being widely used in industrial applications because of its high performance/cost ratio. Due to numerous uncertainties which decrease the performance of the motor driving system, motor control methods have been developed. The control techniques encompass speed, torque, and rotor position.

The control strategies studied for this paper are for speed control of PMSM. Numerous nonlinear control techniques, including linearization control, adaptive control, robust control, sliding mode control, disturbance observer-based control, finite time control, fractional order control, fuzzy control, and neural network control, have been developed for the PMSM system.

The permanent magnet motor control consists of three key factors:

1) Controller type (PID, fuzzy logic or neural networks)
2) Implemented algorithm (hardware, software, or both)
3) Availability of design tools

Based on the above factors, the control strategies can be broadly classified into the following different types:

A. *Fuzzy Logic Control*

To obtain high operating performance, intelligent control techniques like fuzzy control and adaptive fuzzy control have been developed. To counter the dynamic uncertainty and external load effect, an adaptive fuzzy control (AFC) is constructed using fuzzy basis function. Various studies have been conducted using fuzzy logic. Ranging from hardware [9], speed control of PMSM is also simulated using genetic-based fuzzy controller [10]. K Boby et al. [11] present a study on the performance of Fuzzy-PID controller using simulated results on MATLAB. The simulated results conclude that the fuzzy-PID controller is more robust as compared to traditional PI controller.

Kumar et al. [12] proposes a system that implements direct and quadrature current as well as the adaption mechanism. The adaption mechanism is used to force the system to behave like a model when add to the output signal of direct fuzzy controller. The designed controller allows for robust speed control under model parameter and load torque variations. To make the controller more efficient, DT Govindaraj et al. [13] present a Kalman filter for estimation of motor quantities.

B. *Neural Network Control*

The second under the category of intelligent controllers, Neural Networks can be directly employed for PMSM speed control. A Dinu et al. [14] detail over an algorithm for compact neural network hardware implementation. The algorithm is based on three steps: ANN mathematical model digitization, conversion from digitized model to logic-gate structure, and optimization of hardware. The proposed strategy combines ANN design software with hardware design packages.

CH Lin et al. [15] present a novel technique for speed control; a PMSM adaptive PID controller based on recurrent wavelet neural network (RWNN). Using WNN, this algorithm improves PWSM speed regulation performance due to the recurrent nose which increase network response speed and also give stable outputs. Another study [16] uses feed forward neural network in place of PU controllers for vector control of PMSM. The Neural Network based approach is used to enhance efficiency and ensure robustness against load and parameter variations.

C. *Sensorless Control*

Sensorless control uses the information of rotor position for efficient control of PMSM. As the sensor decreases robustness, it is employed to estimate rotor position using indirect techniques. The paper [[17] provides the design and FPGA implementation of a sensorless control using extended Kalman filter (EKF). The filter is used to find rotor flux angle and rotor speed and gives a good step response performance in case of low-speed control, inverse speed control and high speed control. Another paper studies nonlinear control with back-stepping technique [18] implements a back-stepping technique integrated with an adaptive pole placement control strategy. The simulations show good performance of this technique in comparison with nonlinear control.

An approach [18] using back-EMF space vector is also studied which proves to be advantageous because of integrator elimination. The algorithm is based on the estimation of rotor speed and angular position starting from back electromotive force space vector determination using reference voltages using current controllers. Hongryel et al. [19] proposed and verified that a new sliding-mode observer (SMO) has superior through simulations and experiments.

D. *Hybrid Control*

These techniques combine neural networks and fuzzy logic and are called Neuro-Fuzzy systems, whereby the fuzzy logic is integrated with sensorless control technique. HH Chou et al. [20] presented a neural fuzzy control (NFC) for speed loop of PMSM drives. The parameters are adjusted by a radial basis function neural network (RBF NN) to deal with the effect of system dynamic uncertainty.

C Elmas et al. [21] adjusted input and output parameters of membership functions in a fuzzy logic controller (FLC) using back propagation to train the model. J Shiny [22] proposes an integrated system using FPGA and fuzzy logic control technique to control rotor position angle and motor speed. For robustness and accuracy, fuzzy logic control is implemented. The simulation results prove that FPGA based speed control performance is better in terms of accuracy and response time as compared to other hardware base speed control techniques.

III. Principle of Vector Control

Vector control [23] is primarily used to maximize the motor's torque. Additionally, it reduces the ripples observed in the speed and torque response of the motor. The torque is maximized by making the rotor magnetic field orthogonal to the stator magnetic field. Vector control uses current and voltage measurements to control a motor. The measurements help to know the rotor position and calculate a new set of three-phase voltages applied to the motor. Figure 1 shows the block diagram logic of AC motor vector control.

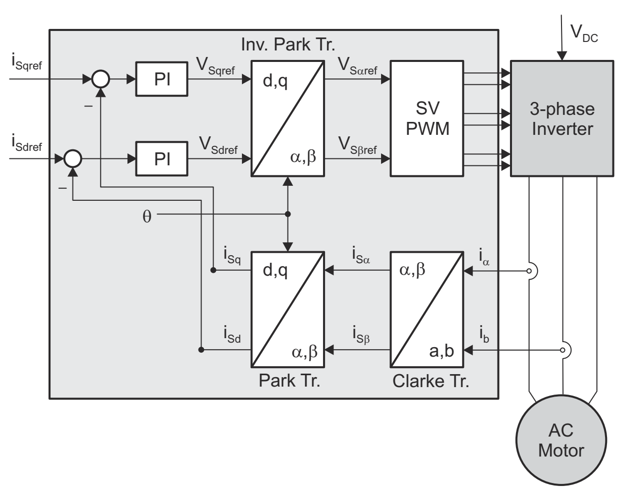

Fig. 1. Vector control block diagram

We maintain the three-phase currents to control the torque but do not control them directly. We use the DC signals equivalent to the three-phase current. This is done because the sinusoidal waveforms of three-phase AC currents are hard to operate by the PID controllers. Figure 2 shows the axial components of the magnetic field of the motor.

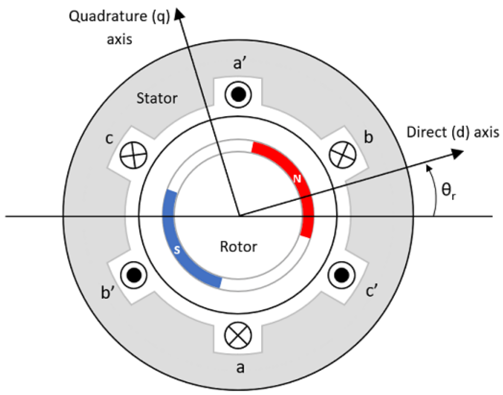

Fig. 2. Axial components of the motor

The stator's magnetic field is segregated into two components: a direct axis component and a quadrature axis component. The direct component is parallel to the rotor magnetic field and contributes nothing to torque generation. The quadrature component is orthogonal to the rotor field and generates all the torque. So, we maximize the quadrature component and minimize the direct component.

After applying Clark transform [24] on the three-phase currents, i_a, i_b, i_c, we obtain the quadrature and direct component of current, i_α and i_β. The Clark transform is implemented as follows:

$$\begin{bmatrix} \alpha \\ \beta \\ 0 \end{bmatrix} = \frac{2}{3} \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

, where a, b, and c are the components of the three-phase system in the abc reference frame, and α, β, and zero are the components of the two-axis system in the stationary reference frame.

These components are still not uniform and would not work efficiently with PI controller. So, we use Park transform which converts the two AC currents into DC currents. It does this by allowing the controller to move from the stationary reference frame to the rotating reference frame. The Park transform is implemented as follows:

$$\begin{bmatrix} d \\ q \\ 0 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \\ 0 \end{bmatrix}$$

, where d, and q are the components of the three-phase system in the rotating reference frame, and α, β, and zero are the components of the two-axis system in the stationary reference frame.

The values are sent to PID [25] controllers that output the rotating reference frame voltages: v_q and v_d. These are converted back to the three-phase voltages by performing inverse Clark and Park transform and the obtained three-phase voltages are sent to the motor.

IV. Proposed Model

The proposed approach mainly focuses on designing an optimized current control loop, leaving the rest of conventional models mostly untouched.

A. Inverter and Motor Plant Model

The frequency of power supplied to an AC motor is controlled by the inverter, in order to control the speed of rotation of the motor. The use of inverter prevents the AC motor to operate at full speed as soon as the power supply is turned ON. This allows us to control the speed and acceleration of the AC motor and allows us to use it for more applications.

The type of control we are using is called Pulse Width Modulation, or PWM. It is a method of reducing the average power delivered by an electrical signal, by turning the switch between the supply and load ON and OFF at a fast rate. The term "duty cycle" describes the proportion of "on" time to the regular interval or 'period' of time. For example, a low duty cycle corresponds to low power. And by doing this, we can reduce average power to our desired level and control amplitude of power to the load. It is important to note that PWM switching frequency has to be high enough to not effect the load, and the waveform perceived by the load is as smooth as possible.

B. Space Vector Pulse Width Modulation

The torque and stator current flux are controlled by a speed controllers, and these controllers determine the voltages, v_d and v_q, to be applied to the motor. After that, an inverse Park transform converts the two-axis rotating system (v_{sqref}, v_{sdref}) back to a two-axis stationary system ($v_{s\alpha ref}$, $v_{s\beta ref}$). These are the components of the stator voltage vector and are the inputs for the SVPWM, which generates the three-phase output voltage to the motor.

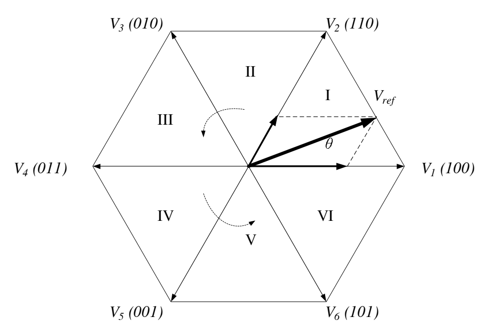

Fig. 3. Voltage space vectors

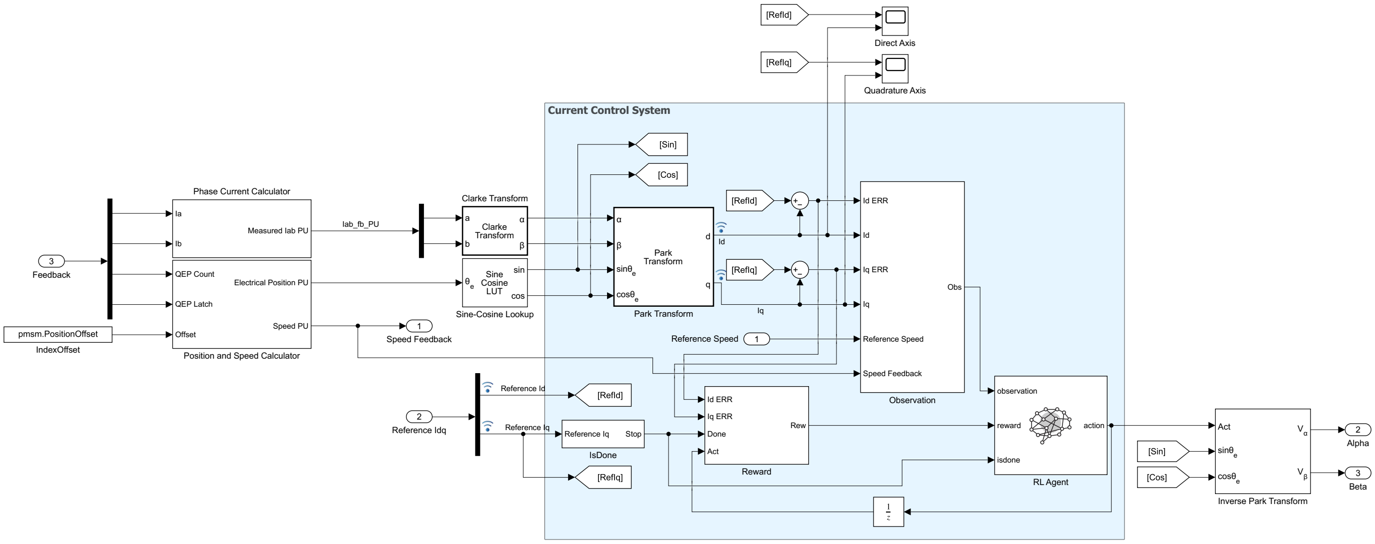

Fig. 4. TD3 agent controller

C. Speed Control Loop

The figure below shows the basic block diagram of a speed controller. The first step of the controller is simply to calculate the margin of error between the commanded and actual speed at that specific point of time. The error is directly sent to the current PI regulator. If the error has a very large value, the integrator will probably establish an excessive output.

In this case the output results in an unwanted overshoot because of PI controller integral property.

$$e(t) = reference_{speed} - actual_{speed}$$

The error between reference and actual speeds is then fed into the control algorithm, which can be broken into its respective Proportional and Integral parts.

$$output = output_{bias} + K_c e(t) + \frac{K_c}{\tau_1} \int_0^t e(t)\,dt$$

The PMSM controller subsystem contains all three controllers. The outer-loop controller shown in figure regulates the speed of the motor. The two inner-loop controllers control the d-axis and q-axis currents separately. The command from the outer-loop controller directly feeds to the q-axis to control torque.

D. Current Control Loop

To have a basis to compare the TD3 agent controller to, we also implement conventional field-oriented control for direct-axis current i_d and quadrature-axis current i_q. As the mathmatical model of PMSM in Motor Control Blockset library is isotropic; the d and q axis have the same inductance, both PIs have the same following transfer function and tuning, as proposed by Ravindra et al [26].

$$H_d(s) = H_q(s) = \frac{1/R_s}{1 + s(L_d/R_s)} = \frac{K_1}{1 + sT_1}$$

a) Reinforcement Learning Agent: Reinforcement learning is employed for the successful completion of a desired task within an uncertain environment. After each sample time, the agent, which, in our model is a Twin-Delayed Deep Deterministic Policy Gradient (TD3) [27] algorithm. Agents are trained on reward basis; a reward is a measure of how successful the preceding action was with respect to the desired goal. Every reinforcement learning agent has two primary components: a policy and an algorithm.

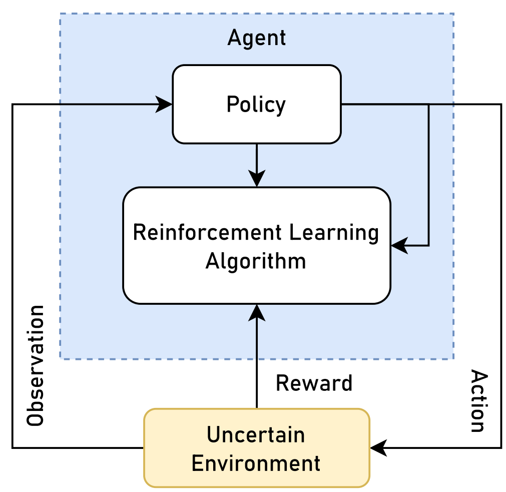

Fig. 5. Block diagram for RL workflow

A probability distribution of the actions to be taken is mapped from the current environment observation to the policy. A function approximator with adjustable parameters and a particular approximation model, like a deep neural network, implement the policy within an agent. Continually updated policy parameters are produced by the learning algorithm based on rewards, observations, and actions.

The learning algorithm seeks to identify the best course of action that maximizes the anticipated cumulative long-term reward during the activity. Figure 5 shows the workflow of a typical reinforcement learning agent.

In the proposed model, the input observation to the agent is a concatenated vector of direct current i_d, quadrature current i_q, the errors in their values with respect to the reference, the reference speed, the feedback state of the motor. The reward function at each step is expressed by the following relation:

$$r_t = -(Q_1 id_{error}^2 + Q_2 iq_{error}^2 + R\sum_j (u_{t-1}^j)^2) - 100d$$

, where $Q_1 = Q_2 = 5$, $R = 0.1$, id_{error} is the direct axis error, iq_{error} is the quadrature axis error, u_{t-1}^j are the previous time step actions, and d is a flag indication early termination of simulation.

b) TD3 Agent Controller: The twin-delayed deep deterministic policy gradient (TD3) algorithm is an off-policy reinforcement learning method. A TD3 agent looks for an optimal policy that maximizes the expected cumulative long-term reward in an actor-critic configuration.

The TD3 algorithm is a comprehensive extension of the DDPG [8] algorithm. DDPG agents can overestimate value functions, which can produce sub-optimal policies. To reduce value function overestimation, the TD3 algorithm learns two Q-value functions and uses the minimum value function estimate during policy updates. The policy and targets are updated less frequently than the Q functions in the TD3 agent and Gaussian-noise is also added to the target action, which makes the policy less likely to exploit actions with high Q-value estimates.

V. Simulation Results

To validate the performance gains of the TD3 agent over a conventional PI controller, we calculate the mean relative error between the reference direct axis current, reference quadrature axis current, and reference speed with that measured after implementing the controllers. The mean relative error (MRE) is calculated as:

$$err_{mean} = mean(\frac{actual_{value} - measured_{value}}{measured_{value}})$$

Figure 6 shows the reference, PI, and RL agent direct axis current in a simulation conducted with a stop time of $T = 10s$. As the ideal direct axis current for an FOC-type control is zero, statistical analysis fails to converge. As such, the best conclusion can only be deduced by observing the figure. As it is apparent that the RL agent i_d stays much closer to zero, than the PI controller i_d, we conclude that our RL agent performs better in direct axis reference frame.

Figure 7 shows the reference, PI, and RL agent quadrature axis current in a simulation conducted with a stop time of $T = 10s$. The MRE between the reference quadrature and PI-controlled quadrature current is 7.82% whereas MRE between the reference quadrature and RL-controlled quadrature current is 6.88%, an absolute improvement of 0.94%.

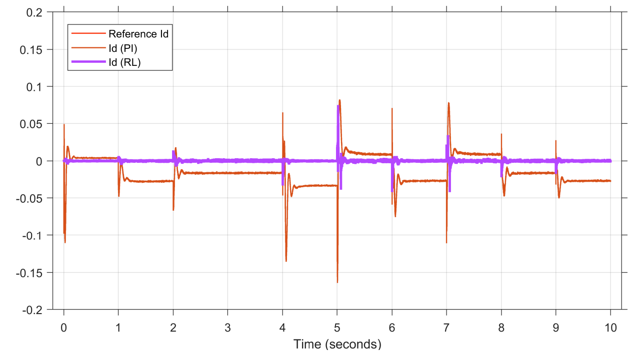

Fig. 6. Direct axis current

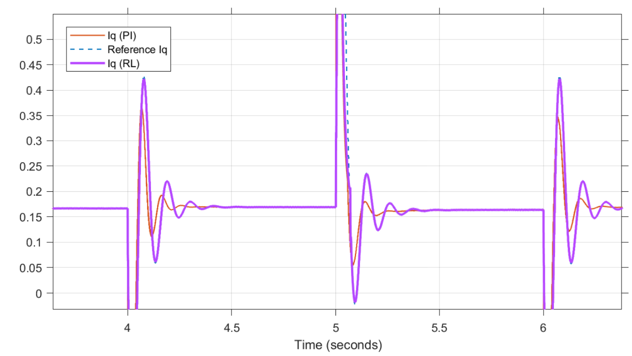

Fig. 7. Quadrature axis current

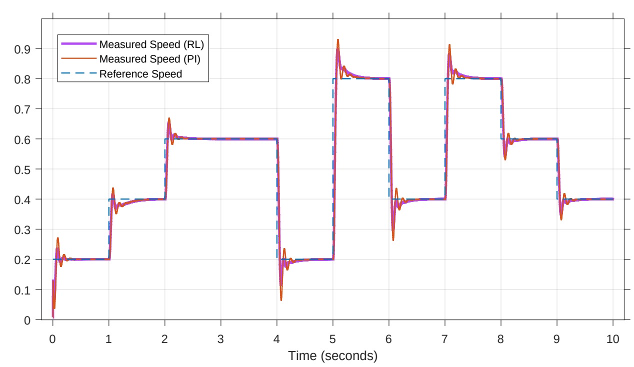

Fig. 8. Simulation speed results

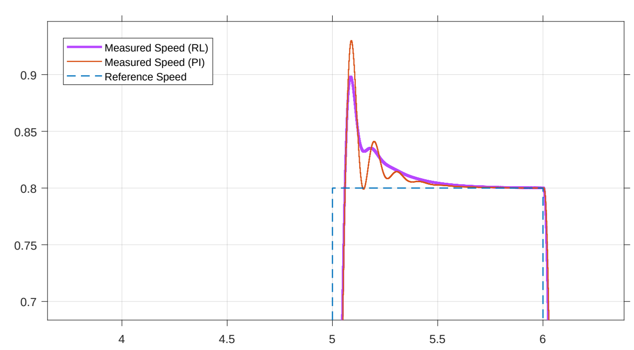

Fig. 9. Simulation speed results (zoomed)

Figure 8 shows the reference, PI, and RL agent speed in a simulation conducted with a stop time of $T = 10s$. The MRE between the reference quadrature and PI-controlled speed is 5.87% whereas MRE between the reference speed and RL-controlled speed is 4.60%, an absolute improvement of 0.94%.

We also observe from Figure 9 that the PU peak deviations are lower in TD3 agent controller than the conventional PI controller. Although the settling time increases by a bit, the speed-curve is comparatively much smoother than the conventional PI controller. We deduce that the proposed TD3 agent providing correction signals to the inner current loop observed performance gains against a conventional PI controller in all fronts, with negligible added mathematical complexity and convergence times.

VI. Conclusion

This project report intends to underline that a machine learning algorithm-based controller attached to a PMSM allows the motor to perform better than a conventional PI controller attached to it. All the PI controllers that are used frequently were explored in this report. Field-oriented control was used with a PMSM, with nearly all other conventional controller models unaltered.

We used a Twin-Delayed Deep Deterministic Policy Gradient (TD3) machine learning algorithm to correct the signals sent as input to the controller's inner control (current). The inputs to the algorithm were quadrature current i_q, direct current i_d, errors in their values with respect to the reference frame, reference speed, and feedback state of the monitor.

The TD3 algorithm-based and a conventional controller were implemented on MATLAB/Simulink, and their results were compared. Mean relative error was the metric of comparison between the two controllers. The results of mean relative error declared the TD3-algorithm-based controller to be better by 0.94% in quadrature current following as well as 0.94% in motor speed following as well.

References

[1] S. Eriksson, "Design of permanent-magnet linear generators with constant-torque-angle control for wave power," *Energies*, vol. 12, no. 7, p. 1312, 2019.

[2] P. Ouyang, V. Pano, J. Tang, and W. Yue, "Position domain nonlinear pd control for contour tracking of robotic manipulator," *Robotics and Computer-Integrated Manufacturing*, vol. 51, pp. 14–24, 2018.

[3] S.-W. Baek and S. W. Lee, "Design optimization and experimental verification of permanent magnet synchronous motor used in electric compressors in electric vehicles," *Applied Sciences*, vol. 10, no. 9, p. 3235, 2020.

[4] F. Amin, E. B. Sulaiman, W. M. Utomo, H. A. Soomro, M. Jenal, and R. Kumar, "Modelling and simulation of field oriented control based permanent magnet synchronous motor drive system," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 6, no. 2, pp. 387–395, 2017.

[5] F. Mohd Zaihidee, S. Mekhilef, and M. Mubin, "Robust speed control of pmsm using sliding mode control (smc)—a review," *Energies*, vol. 12, no. 9, p. 1669, 2019.

[6] N. Maleki, M. R. A. Pahlavani, and I. Soltani, "A detailed comparison between foc and dtc methods of a permanent magnet synchronous motor drive," *Journal of Electrical and Electronic Engineering*, vol. 3, no. 2-1, pp. 92–100, 2015.

[7] S. Sakunthala, R. Kiranmayi, and P. N. Mandadi, "A review on speed control of permanent magnet synchronous motor drive using different control techniques," in *2018 International Conference on Power, Energy, Control and Transmission Systems (ICPECTS)*. IEEE, 2018, pp. 97–102.

[8] 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. [Online]. Available: https://arxiv.org/abs/1509.02971

[9] Y.-S. Kung, M.-H. Tsai, and C.-S. Chen, "Fpga-based servo control ic for pmlsm drives with adaptive fuzzy control," in *2006 1ST IEEE Conference on Industrial Electronics and Applications*. IEEE, 2006, pp. 1–6.

[10] N. Öztürk and E. Çelik, "Speed control of permanent magnet synchronous motors using fuzzy controller based on genetic algorithms," *International Journal of Electrical Power & Energy Systems*, vol. 43, no. 1, pp. 889–898, 2012.

[11] K. Boby, A. M. Kottalil, and N. Ananthamoorthy, "Simulation of pmsm vector control system with fuzzy self-adjusting pid controller using matlab," *International Journal of Scientific and Research Publications*, vol. 3, no. 3, pp. 1–4, 2013.

[12] R. V. Kumar and D. T. Govindaraj, "Adaptive fuzzy logic based speed control of permanent magnet synchronous motor," in *MNRE Sponsored National Conference AITHRE*, 2013, pp. 108–113.

[13] D. T. Govindaraj and R. V. Kumar, "Aflc based speed control of pmsm drive," *International Journal Of Advanced and Innovative Research*, pp. 433–439, 2012.

[14] A. Dinu, M. N. Cirstea, and S. E. Cirstea, "Direct neural-network hardware-implementation algorithm," *IEEE Transactions on Industrial Electronics*, vol. 57, no. 5, pp. 1845–1848, 2009.

[15] C.-H. Lin, "Hybrid recurrent wavelet neural network control of pmsm servo-drive system for electric scooter," *International Journal of Control, Automation and Systems*, vol. 12, no. 1, pp. 177–187, 2014.

[16] S. Mishra and A. Tomer, "Speed control of pmsm drives by using neural network controller," *Advance in Electronic and Electric Engineering*, vol. 4, no. 4, pp. 353–360, 2014.

[17] Y.-S. Kung, N. Vu Quynh, N. Trung Hieu, and J.-M. Lin, "Fpga realization of sensorless pmsm speed controller based on extended kalman filter," *Mathematical Problems in Engineering*, vol. 2013, 2013.

[18] D. U. A. N. A. BACKSTEPPING, "Low-speed sensorless control of pmsm motor drive using a nonlinear approach backstepping control: Fpga-based implementation," *Journal of Theoretical and Applied Information Technology*, vol. 36, no. 2, 2012.

[19] H. Kim, J. Son, and J. Lee, "A high-speed sliding-mode observer for the sensorless speed control of a pmsm," *IEEE Transactions on Industrial Electronics*, vol. 58, no. 9, pp. 4069–4077, 2010.

[20] H.-H. Chou, Y.-S. Kung, N. V. Quynh, and S. Cheng, "Optimized fpga design, verification and implementation of a neuro-fuzzy controller for pmsm drives," *Mathematics and Computers in Simulation*, vol. 90, pp. 28–44, 2013.

[21] C. Elmas, O. Ustun, and H. H. Sayan, "A neuro-fuzzy controller for speed control of a permanent magnet synchronous motor drive," *Expert Systems with Applications*, vol. 34, no. 1, pp. 657–664, 2008.

[22] J. Shiny, M. Ragulkumar, and R. Govindarajulu, "Adaptive fuzzy logic based speed control of permanent magnet synchronous motor using fpga," *International Journal of Emerging Trends in Science and Technology*, vol. 1, no. 3, pp. 393–398, 2014.

[23] D. W. Novotny and T. A. Lipo, *Vector control and dynamics of AC drives*. Oxford university press, 1996, vol. 41.

[24] S. Chattopadhyay, M. Mitra, and S. Sengupta, "Clarke and park transform," in *Electric Power Quality*. Springer, 2011, pp. 89–96.

[25] S. Bennett, "Development of the pid controller," *IEEE Control Systems Magazine*, vol. 13, no. 6, pp. 58–62, 1993.

[26] R. K. Sharma, V. Sanadhya, L. Behera, and S. Bhattacharya, "Vector control of a permanent magnet synchronous motor," in *2008 Annual IEEE India Conference*, vol. 1, 2008, pp. 81–86.

[27] S. Fujimoto, H. van Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," *CoRR*, vol. abs/1802.09477, 2018. [Online]. Available: http://arxiv.org/abs/1802.09477