An Approach for DC Motor Speed Control with Off-Policy Reinforcement Learning Method

Sevilay Tufenkci, Gurkan Kavuran, Celaleddin Yeroglu

Abstract—Integration of self-learning mechanisms with control systems is frequently encountered in the literature due to the development of autonomous systems. This paper proposes a tuning method of PI controllers using a deep reinforcement learning algorithm, which is known as self-learning structure. The coefficients of the PI controller, which is used to control a DC motors, are determined. The proposed method aims to adjust the voltage value applied to the input of the DC motor to reach the desired speed with the tuned PI controller using the twin-delayed deep deterministic policy gradient (TD3) reinforcement learning algorithm. The Kp and Ki coefficients of the PI controller are taken as the absolute values of the neural network weights, which are driven by Gradient descent optimization to positive values with a fully connected layer. The proposed tuning method has been shown to provide a higher gain margin and a more optimal solution.

Index Terms— **Deep reinforcement learning, DC motor, PI controller, Twin-delayed deep deterministic policy gradient**

I. Introduction

IN RECENT YEARS, the interest in artificial intelligence and machine learning issues has increased with remarkable developments. Many thanks to the modern control techniques, which is one of the application areas of these studies, the system has revealed what is expected from it without any external influence [1, 2].

Reinforcement learning uses an agent that perceives its environment and acquires information as a result of its interaction with its environment and can act in line with this information. This agent helps the system choose the most appropriate action to achieve its purpose. [2] It is a method that enables learning using a series of reward and punishment systems in its background. The aim of the agent is to realize learning by trying to increase the reward value.

Reinforcement learning, a method derived from the science of psychology, is based on the famous psychologist Skinner's (1938) saying, "behaviors are affected by consequences".

Reinforcement learning is a method used to match actions and situations [3]. There are many studies in the literature using reinforcement learning. Some of them are used in many different areas, such as motor control [4, 5, 6], mobile robots [7, 8], and video games [9].

The Q-Learning method is a special type of reinforcement learning approach and is the most well-known among others [10, 11]. This method was first named in 1989 by using the letter Q for the Watkins value function. This method is based on the Markov theory of decision processes. The environment in which the agent is located is modelled as a probabilistic process with the Markov feature and while the present state is in the present state, future situations are evaluated independently from past situations. With the Q-learning algorithm, the agent can find the action that can get the greatest reward for the defined situations in the environment with the q-table.

The reinforcement learning method may find it difficult to find solutions in continuous action areas, high dimensional inputs, or complex actions [12, 13, 14]. Deep Q-Network algorithms have emerged to overcome this problem [15]. However, in these algorithms, the action that maximizes the action value is due to an outcome and is not sufficient in continuous domains.

Deterministic Policy Gradient (DPG) algorithms have emerged to solve the problem that arises with high dimensional and continuous domains [16]. This method, which uses a non-political actor-critic structure and performs deterministic learning, has performed well. Then, Deep Deterministic Policy Gradient (DDPG), which gives better results in large size and continuous actions, was developed based on the DPG algorithm [17]. With the advances made in this area, a new algorithm called Twin Delay Deep

SEVILAY TUFENKCI, is with Department of Computer Technology Department, Malatya Turgut Ozal University, Malatya, Turkey (e-mail: sevilay.tufenkci@gmail.com)

https://orcid.org/0000-0001-9815-7724

GURKAN KAVURAN, is with Department of Electrical-Electronics Engineering, Malatya Turgut Ozal University, Malatya, Turkey (e-mail: gurkan.kavuran@ozal.edu.tr).

https://orcid.org/ 0000-0003-2651-5005

CELALEDDIN YEROGLU, is with Department of Computer Engineering, Inonu University, Malatya, Turkey (e-mail: c.yeroglu@inonu.edu.tr).

https://orcid.org/0000-0002-6106-2374

Manuscript received May 10, 2022; accepted Aug 3, 2022.
DOI: 10.17694/bajece.1114868

Deterministic Policy Gradient (TD3) has been produced in a recent period [18]. It has been the best among the previously developed models with its performance in high dimension and continuous areas using a non-policy learning method.

DC motors are ubiquitous in terms of their simplicity and ease of use. They are encountered in various industrial applications as well as small household appliances. However, it needs a control structure to operate as desired. The purpose of this control structure is to keep the output of the system, i.e. its speed, at a predetermined reference value for the system. There have been many studies on DC motor applications for decades [19, 20].

The motivation and novelty of the paper can be explained as follows;

- The training of the RL-based controller tuning approach benefits greatly from experience-based learning in a realistically structured simulation environment.
- The simulation environment may be used to train controller tuning algorithms in order to get the optimum practical performance against uncertainty, noise, and external disturbances.
- According to simulation environment results, RL approaches are intrinsically ideal for experience-based learning and performance improvement, which was a major motivator for the authors of the current work.
- We concentrated on creating an appropriate simulation environment for successful RL-based controller tuning.
- The Kp and Ki coefficients are the absolute values of the neural network weights, which are pushed to positive values using Gradient Descent Optimization with a fully connected layer.

Within the scope of this study, the TD3 method was used to achieve the desired operating speed for a DC motor and to maintain it at this speed. In order for the system to be at the desired speed, it is aimed to perform the desired behavior within the reward and penalty structure, and in line with this purpose, the voltage input value of the system that will provide this speed is adjusted as a result of learning.

II. MATERIALS AND METHODS

A. *Reinforcement Learning*

Reinforced learning is a learning method that enables an agent who can perceive the environment and act in his environment to choose the most appropriate actions to achieve his goal [2]. When the agent performs an action in the environment, the reward or punishment structure is used to show the suitability of this action for the system. With this structure, the agent performs the best action among the situations while trying to reach the highest reward value [21, 22].

This method, which is developed on the basis of behavioral psychology, interacts with its environment in this problem environment to provide a solution to a problem. It tries to find the most appropriate solution by trying different ways in the environment to reach a solution. Unlike consultancy and non-consultancy learning methods, it realizes learning without using any data set. The learning process is entirely dependent on the interaction with the environment.

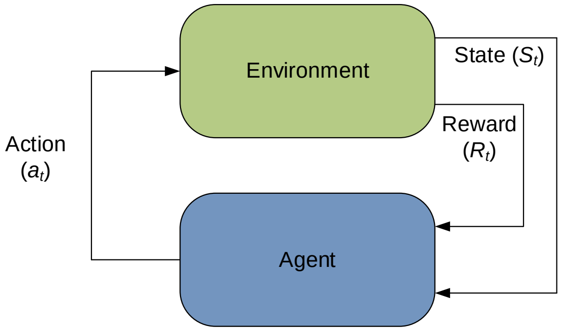

Fig.1. Block Diagram of Reinforced Learning System

Figure 1 presents the structure of the reinforcement learning system. Its essential components are action (A_t), agent, environment, reward (R_t), principle, and state function (S_t). Within this structure, the agent is the decision maker that performs the learning process. On the other hand, an agent takes actions in the environment and constantly interacts. The principle refers to the possibility of choosing each action that can be selected in its environment. The total reward to be obtained from the situation of the agent and other conditions that will occur accordingly is expressed as a state function [3].

B. *Markov Decision Process*

In the reinforcement learning method, the environment is modeled as a probabilistic process, and the future situations occur independently from the past situations.

In Figure 1, S_t and R_t represent the results after an action is performed and they are random variables. The general form of the probability density function including S_t and R_t.

$$p(s',r|s,a)=\mathrm{P}\left[S_t=s',R_t=r|S_{t-1}=s,A_{t-1}=a\right] \tag{1}$$

where $s,s'\in S, r\in R$ and $a\in A$.

Equation (1) expresses the basis of the working structure of the Markov process. When this equation is examined, it is revealed that the state and the form of the reward at time t depend only on the situation and action found a step earlier. Transition possibilities depend on these;

$$p(s'|s,a)=\mathrm{P}\left[S_t=s'|S_{t-1}=s,A_{t-1}=a\right]=\sum\nolimits_{r\in R}p(s',r|s,a) \tag{2}$$

$$r(s,a)=\mathrm{E}\left[R_t|S_{t-1}=s,A_{t-1}=a\right]=\sum\nolimits_{r\in R}r\sum\nolimits_{s'\in S}p(s',r|s,a) \tag{3}$$

Equation (3) uses the marginal distribution of R_t to obtain the expected reward.

C. *Twin Delayed Deep Deterministic Policy Gradient Algorithm (TD3)*

TD3 algorithm was built on the basis of the Deep Determinative Policy Gradient algorithm that emerged in 2015 [17]. This method, aims to increase the stability and performance of the system by taking into account the approach error in the DDPG algorithm [18]. For this, it is based on the method applied to reduce over prediction in the Double-DQN algorithm [18] and is essentially a combination of three deep reinforcement learning techniques named Actor-Critic [23], Policy Gradient [16], and Double Deep Q-Learning [24].

The first technique applied in the TD3 algorithm is clipped double-Q learning. This technique consists of two Bellman equations, two-actor networks, and two critical network structures. Another applied technique is that it performs policy updates less frequently, unlike learning Q. It makes the process of updating the policy depending on the update of the value function of the model. This leads to a value estimation with a lower variance while providing a good policy that ensures the system's stability and reduces errors.

It realizes learning by adding noise to the policy with its Target Policy Smoothing. In order to obtain a satisfactory study result, it learns to evaluate all kinds of situations that may occur in terms of the system and its actions in a wide variety of environments. TD3 adds random noise to the target action and averaging over mini-batches.

$$y \equiv r + \gamma Q_w(s', \mu_\theta(s') + \varepsilon) \tag{4}$$

$$\varepsilon \approx clip(N(0,\sigma), -c, c) \tag{5}$$

where r is the reward value and γ is a discount factor expressing the effect of the reward value on reinforcement learning agents in the future, and this factor takes a value between 0 and 1 [25]. Since TD3 has two critical models that determine the loss function, it is expressed as $MSELoss\left(Q_1\left(s,a'\right),Q_t\right)+MSELoss\left(Q_2\left(s,a'\right),Q_t\right)$.

The critical loss is posted back, and the values are adjusted to determine the parameter values. As a result of these, the parameter of the actor model can be defined as;

$$\theta_i \leftarrow \min_{\theta_i} N^{-1}\sum_{i=1}^{N}\left(y - Q_{\theta_i}\left(s,a\right)\right)^2 \tag{6}$$

$$\nabla_\varphi J(\varphi) = N^{-1}\sum \nabla_a Q_{\theta_1}(s,a)\Big|_{a=\pi_\varphi(s)} \nabla_\varphi \pi_\varphi(s) \tag{7}$$

where N is from the repeat buffer $\left[s_t, a_t, r_t, s_{t+1}\right]$. We are updating φ_i, where φ_i and θ are the parameter actor and critic weights, respectively.

$$\theta'_i \leftarrow \tau\theta_i + (1-\tau)\theta'_i \tag{8}$$

$$\varphi'_i \leftarrow \tau\varphi\ + (1-\tau)\varphi'_i \tag{9}$$

Finally, we update the target networks as in Equations (8) and (9).

D. Brushed DC motor

Nowadays, the design and control of DC motors, which have a place in robotic applications, the military, and many other industrial areas, are among the subjects of interest. Among the many types of DC motors, brushed DC motors are widely used. The speed of brushed DC motors can be changed by varying the input voltage or the magnetic field. Depending on the field's connections to the power source, the speed and torque characteristics of a brushed motor can be varied to provide constant speed. The electrical circuit of the armature part has been shown with rotor structure in Figure 2.

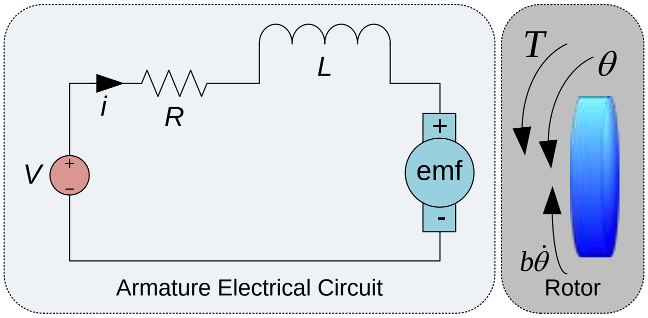

Fig.2. Equivalent Circuit for DC Motor

For the given model, it is assumed that the input of the system is the voltage source applied to the motor's armature and the rotation speed of the shaft θ at its output. Among the parameters presented in the model, J is the moment of inertia of the rotor, R is the electrical resistance, L is the electrical inductance, b is the motor viscous friction constant, K_e is the electromotive force constant, and K_t is the motor torque constant [26]. The general form of the motor torque equation for an armature controlled motor;

$$T = K_t i \tag{10}$$

The back emf, e, is proportional to the angular velocity of the shaft multiplied by a constant factor K_e.

$$e = K_e \theta \tag{11}$$

Rate of change integrals are taken for the rotational acceleration and the armature current. Then, by applying Newton's law and Kirchoff's law to the system, the equations are obtained.

$$J\frac{d^2\theta}{dt^2} = T - b\frac{d\theta}{dt} \Rightarrow \frac{d^2\theta}{dt^2} = \frac{1}{J}\left(K_t i - b\frac{d\theta}{dt}\right) \tag{12}$$

$$L\frac{di}{dt} = -Ri + V - e \Rightarrow \frac{di}{dt} = \frac{1}{L}\left(-Ri + V - K_e\frac{d\theta}{dt}\right) \tag{13}$$

The transfer function is used to express the relationship between the input and output of the system. The Laplace transform is applied to obtain the transfer function of the system and the system is expressed in the s domain.

$$s\left(Js + b\right)\theta\left(s\right) = KI\left(s\right) \tag{14}$$

$$\left(Ls + R\right)I\left(s\right) = V\left(s\right) - Ks\theta\left(s\right) \tag{15}$$

The open-loop transfer function of the system obtained via Equation (16) by using the above two equations;

$$P(s)=\frac{\theta(s)}{V(s)}=\frac{K}{(Js+b)(Ls+R)+K^2}\left|\frac{rad/\sec}{V}\right| \quad (16)$$

III. DESIGN OF TD3 BASED PI CONTROLLER

In this study, TD3 algorithm, which is one of the reinforcement learning methods, is used in the provision of speed control of a DC motor system. With this algorithm, a controller is used to perform the speed control of the DC motor, and the controller coefficients are adjusted. The open-loop transfer function form of the DC motor system to be controlled here;

$$P(s)=\frac{1.25}{(0.5s+0.01)(0.05s+0.4)+1.25^2}\left|\frac{rad/\sec}{V}\right| \quad (17)$$

The general form of the transfer function of the PI control system used in the study;

$$C(s)=k_p+\frac{k_i}{s} \quad (18)$$

An agent is used in the reinforcement learning environment to adjust the PI coefficients required for speed control. The environment required for the agent to provide control is designed using Matlab Simulink. To simulate the controller in this model, the simulation time is set as Tf =40 and the controller sampling time Ts = 1 seconds.

In the model, a reference signal consists of the observation vector for observing the results that will arise due to this signal, and the reward structure that forms the reward and punishment structure of the system in line with the action taken and the output signal. The observation vector used for this learning environment;

$$\left[\int edt\ e\right]^T \quad (19)$$

$$e=r-s \quad (20)$$

where s refers to the speed of the DC motor, and r represents a randomly determined reference speed between 0 and 155 as the input. Negative defined reward function that maximizes the reward that will guide the reinforcement learning agent to achieve the goal;

$$\text{Reward}=-\left((ref-s)^2(t)+0.01u^2(t)\right) \quad (21)$$

In order to decide which, action the TD3 agent will perform as an actor in line with the observations, a deep neural network structure is used as shown in Figure 3, which consists of an input and output based on observations.

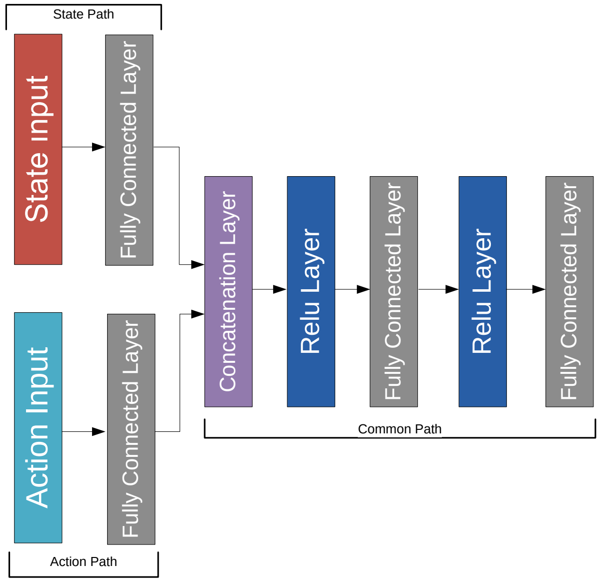

Fig.3. The critic network configuration

Figure 3 above, it refers to the fully connected layer with fc. The weights in the fully connected layer can take a negative value, and in this study, the $abs(weights)*X$ change in this layer ensures that the weights are only positive. Thus, the PI controller used within the scope of the study is expressed as a single layer fully connected neural network model consisting of error and integral of the error.

$$u=\left[\int edt\ e\right]*\left[K_i\ K_p\right]^T \quad (22)$$

The output of the neural network u expressed here, the controller coefficients K_p and K_i represent the absolute value of the neural network weights.

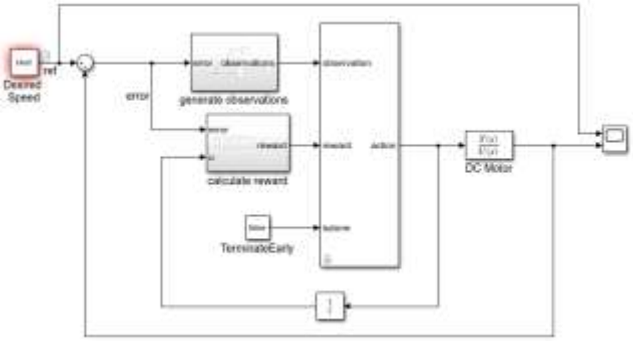

Fig.4. Simulink diagram of the proposed structure.

IV. SIMULATION RESULTS

This study carried out training to provide speed control of a DC motor system using the Twin Delay DDPG algorithm. The training-dependent learning process and learning process results are presented in Figure 5 and Table 1, respectively.

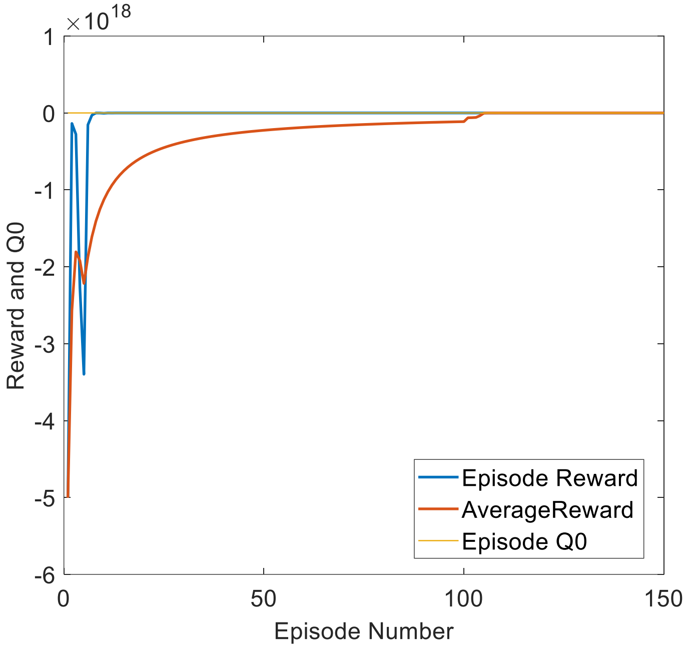

Fig.5. The representation of the learning process performance.

TABLE I
VARIABLES OF TRAINING PROCESS

Maximum Number of Episodes	150
Maximum Steps per Episodes	40
Score Averaging Window Length	100
Training Stopped at Value	-355
Actor Learn Rate	1.00E-03

The reference input was chosen as a set of random values in the range of 0-150 rad/sec to train the system robustly and keep track of the desired reference input in a wide range of structurally acceptable. The following values for the controller coefficients were obtained for the system to perform the desired behavior at the end of 150 iterations.

$$C(s) = 0.1839 + \frac{0.4508}{s} \tag{23}$$

We obtained the system response when the variable speed value is given as the desired input according to time is shown in Figure 6. As can be seen in this figure, it is seen that the system has reached the desired output value for any value entered in a specific range. Figure 6. shows that the system follows the variable step input with high performance.

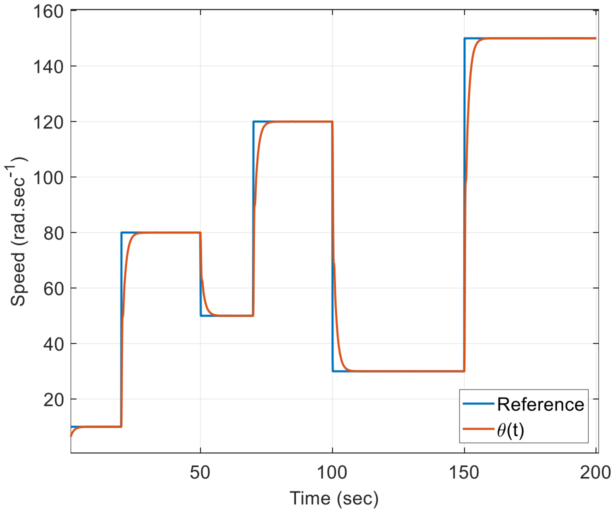

Fig.6. The step response of the DC motor system obtained with the TD3 algorithm

V. CONCLUSIONS

The speed control of DC motors, used in many areas, is carried out using the TD3 algorithm, one of the reinforcement learning algorithms. The learning process is carried out according to a randomly generated reference input in this predetermined value for the system to work in the desired range. As a result of the learning, it has been shown with the simulation results that the system reaches the desired output value. In this study, unlike other learning methods, the effectiveness of the reinforcement learning method, which does not require any prior data, is presented in the speed control of the DC motor.

References

[1] R.S. Sutton, "Reinforcement Learning: Past, Present and Future", Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), Vol. 1585, 1998, 195–197.

[2] L.P. Kaelbling, M.L. Littman, A.W. Moore, "Reinforcement Learning: A Survey", J. Artif. Intell. Res., Vol. 4, 1996, pp. 237–285.

[3] R.S. Sutton, A.G. Barto, "Reinforcement Learning: An Introduction", 1998.

[4] J. Xue, Q. Gao, W. Ju, "Reinforcement learning for engine idle speed control", 2010 Int. Conf. Meas. Technol. Mechatronics Autom. ICMTMA 2010, Vol. 2, 2010, pp. 1008–1011.

[5] A. Traue, G. Book, W. Kirchgässner, O. Wallscheid, "Toward a reinforcement learning environment toolbox for intelligent electric motor control", IEEE Transactions on Neural Networks and Learning Systems, 2020.

[6] Z. Song, J. Yang, X. Mei, T. Tao, M. Xu, "Deep reinforcement learning for permanent magnet synchronous motor speed control systems", Neural Computing and Applications, Vol. 33, 10, 2021, pp. 5409-5418.

[7] E. Uchibe, M. Asada, K. Hosoda, "Behavior coordination for a mobile robot using modular reinforcement learning", IEEE Int. Conf. Intell. Robot. Syst., Vol. 3, 1996, pp. 1329–1336.

[8] Z. Linan, Y. Peng, C. Lingling, Z. Xueping, T. Yantao, "Obstacle avoidance of multi mobile robots based on behavior decomposition reinforcement learning", 2007 IEEE Int. Conf. Robot. Biomimetics, ROBIO, 2007, pp. 1018–1023.

[9] N.J. Van Eck, M. Van Wezel, "Application of reinforcement learning to the game of Othello", Comput. Oper. Res., Vol. 35, 2008, pp. 1999–2017.

[10] C.J.C.H. Watkins, "Learning from delayed rewards", 1989.

[11] C.J.C.H. Watkins, P. Dayan, "Q-learning", Mach. Learn. 1992, Vol. 83, 8, 1992, pp. 279–292,

[12] Y. Liu, H. Wang, T. Wu, Y. Lun, J. Fan, J. Wu, “Attitude control for hypersonic reentry vehicles: An efficient deep reinforcement learning method”, Applied Soft Computing, Vol. 123, 2022, 108865.

[13] S. Zhang, Y. Li, Q. Dong, “Autonomous navigation of UAV in multi-obstacle environments based on a Deep Reinforcement Learning approach”, Applied Soft Computing, Vol. 115, 2022, 108194.

[14] K. M. Zielinski, L. V. Hendges, J. B. Florindo, Y. K. Lopes, R. Ribeiro, M. Teixeira, D. Casanova, “Flexible control of Discrete Event Systems using environment simulation and Reinforcement Learning”, Applied Soft Computing, Vol. 111, 2021, 107714.

[15] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, M. Riedmiller, " Playing Atari with Deep Reinforcement Learning", 2013.

[16] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, M. Riedmiller, "Deterministic Policy Gradient Algorithms".

[17] T.P. Lillicrap, J.J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, D. Wierstra, " Continuous control with deep reinforcement learning", 4th Int. Conf. Learn. Represent. ICLR 2016 - Conf. Track Proc., 2015.

[18] S. Fujimoto, H. Hoof, D. Meger, "Addressing Function Approximation Error in Actor-Critic Methods", http://proceedings.mlr.press/v80/fujimoto18a.html, 2018.

[19] F. Harashima, S. Kondo, "Design Method For Digital Speed Control System Of Motor Drives", PESC Rec. - IEEE Annu. Power Electron. Spec. Conf., 1982, pp. 289–297.

[20] D. Germanton, M. Lehr, "Variable speed DC motor controller apparatus particularly adapted for control of portable-power tools", 1989.

[21] Y. Hoshino, "A proposal of Reinforcement Learning System to Use Knowledge effectively", 2003, pp. 1582–1585.

[22] S.J. Russell, P. Norvig, "Artificial Intelligence A Modern Approach", 2003.

[23] R.S. Sutton, D. Mcallester, S. Singh, Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation", Adv. NEURAL Inf. Process. Syst. 12, Vol. 12, 2000, pp. 1057--1063.

[24] H. van Hasselt, A. Guez, D. Silver, "Deep Reinforcement Learning with Double Q-Learning", Proc. AAAI Conf. Artif. Intell. 30, 2016.

[25] W.B. Knox, P. Stone, "Reinforcement learning from human reward: Discounting in episodic tasks", Proc. - IEEE Int. Work. Robot Hum. Interact. Commun., 2012, pp. 878–885.

[26] University of Michigan: Control Tutorials for MATLAB and Simulink - Motor Speed: System Modeling, https://ctms.engin.umich.edu/CTMS/index.php?example=MotorSpeed&section=SystemModeling.

BIOGRAPHIES

SEVILAY TUFENKCI graduated from Selçuk University Computer Engineering Department in 2017. She graduated her MSc program in Inonu University, department of Computer Engineering in 2019 and is continuing her PhD program in the Computer Engineering Department of Inonu University. Her research interests are control systems, metaheuristic optimization and reinforcement learning methods.

GURKAN KAVURAN received his B.Sc. degree in Electrical and Electronics Engineering from Firat University in 2008. He received PhD degree in Electrical and Electronics Engineering from Firat University in 2017. His research interests include robotics, fractional calculus, control theory and applications, modeling and simulation, signal processing.

CELALEDDIN YEROGLU, received the BSc degree in electronics engineering from Hacettepe University, in 1991, and the Master and PhD degree from the Department of Computer Engineering, Trakya University, in 2000. He received second PhD degree in 2011 from the Department of Electrical and Electronics Engineering, Inonu University. Since 2009, he has been working with the Computer Engineering Department, Inonu University. His research interests include intelligent systems, control theory and applications, simulation and modeling of networks.