Journal of Marine Science and Technology

Volume 28 | Issue 5 Article 13

REINFORCEMENT LEARNING BASED MAXIMUM POWER POINT TRACKING CONTROL OF PARTIALLY SHADED PHOTOVOLTAIC SYSTEM

Kuan-Yu Chou
Institute of Electrical and Control Engineering, National Chiao Tung University, Hsinchu, Taiwan,
eric210092.ece01g@nctu.edu.tw

Chia-Shiou Yang
Institute of Electrical and Control Engineering, National Chiao Tung University, Hsinchu, Taiwan.

Yon-Ping Chen
Institute of Electrical and Control Engineering, National Chiao Tung University, Hsinchu, Taiwan.

Recommended Citation

Chou, Kuan-Yu; Yang, Chia-Shiou; and Chen, Yon-Ping (2020) "REINFORCEMENT LEARNING BASED MAXIMUM POWER POINT TRACKING CONTROL OF PARTIALLY SHADED PHOTOVOLTAIC SYSTEM," *Journal of Marine Science and Technology*: Vol. 28: Iss. 5, Article 13.
DOI: 10.6119/JMST.202010_28(5).0013
Available at: https://jmstt.ntou.edu.tw/journal/vol28/iss5/13

REINFORCEMENT LEARNING BASED MAXIMUM POWER POINT TRACKING CONTROL OF PARTIALLY SHADED PHOTOVOLTAIC SYSTEM

Acknowledgements

The authors would like to thank the Ministry of Science and Technology, Taiwan under Grant MOST 108-2221-E-009 -119 - for providing research funding.

Journal of Marine Science and Technology, Vol. 28, No. 5, pp. 433-443 (2020)
DOI: 10.6119/JMST.202010_28(5).0013

REINFORCEMENT LEARNING BASED MAXIMUM POWER POINT TRACKING CONTROL OF PARTIALLY SHADED PHOTOVOLTAIC SYSTEM

Kuan-Yu Chou, Chia-Shiou Yang, and Yon-Ping Chen

Key words: maximum power point tracking (MPPT), partially shaded, photovoltaic (PV) system, reinforcement Learning, Q-learning.

ABSTRACT

Under the sun insolation in the daytime, the Maximum Power Point Tracking (MPPT) technique is usually used to achieve the maximum power in the photovoltaic (PV) system and often implemented by the Perturbation and Observation (P&O) method. However, due to the use of fixed step size, the P&O method will generate undesired oscillation around the maximum power point (MPP) and thus reduce the tracking efficiency. Besides, the output power of PV modules highly depends on the environment factors such as irradiance and temperature, especially for a PV array, which is formed by PV modules connected in series and parallel. The partially shaded effect would easily happen in a PV array due to clouds, buildings, trees, etc. Due to the partially shaded effect, the characteristic P-V curve of a PV array may possess multi-peaks, which often results in tracking of a local maximum, not the expected global maximum. To deal with the partially shaded effect, this paper proposes a Reinforcement Learning based MPPT method, which is implemented by Q-learning method. Demonstrated by numerical simulation results, the proposed method indeed can track the global MPP faster and more precisely without oscillation.

I. INTRODUCTION

Since the rapid development of technology, the demand of energy has been increasing. However, because of the air pollution and the global warming problems, the sustainable energy has become a crucial issue recently. Therefore, solar energy is a great renewable energy source without producing greenhouse gases and air pollution. Under the improvement of semi-conductor research, the power conversion efficiency of photovoltaic (PV) system has been increasing. In order to achieve the maximum power under any environmental condition, the Maximum Power Point Tracking (MPPT) techniques based on the switching converter have been widely used.

On the other hand, the partially shaded condition usually happens when the area of PV array is large because some part of the PV array would be shaded by the clouds, trees, building, etc. In that case, the characteristic P-V curve of the PV array would change from single peak to multipeaks, which is difficult to track the MPP. The traditional MPPT techniques such as the perturbation and observation method (P&O), constant voltage method, incremental conductance method would let the operating point stagnate at the local MPP instead of global MPP, which result in unnecessary power loss. Thus, the partially shaded(Mohapatra et al., 2017) condition is one of the most important problems in MPPT techniques.

P&O method is the most popular model-free MPPT method, but the fixed size perturbation would cause the undesired oscillation or slow tracking speed, so many different adaptive P&O methods(Elgendy et al., 2011; Zainuri et al., 2012; Kollimalla et al., 2014) are developed to change the perturbation size according to the environment changes. On the other hand, many researchers also used sliding mode control(Bianconi et al., 2012; Levron and Shmilovitz, 2013; Pradhan and Subudhi, 2015), fuzzy logic control(Cheikh et al., 2007; Al Nabulsi and Dhaouadi, 2012; Algazar et al., 2012) in MPPT, but the selection of parameters and the design of controllers are too difficult to implement. In order to achieve the global MPP in partially shaded conditions, lots of online learning methods are proposed such as Particle Swarm Optimization (PSO) (Miyatake et al., 2011; Renaudineau et al., 2014; Yang et al., 2018), which randomly produces several particles to represent operating voltage, and update the particles after every iteration according to the difference of power and random coefficients. Moreover, lots of machine learning methods derived from PSO are used to improve the performance of PSO such as Whale Optimization(Kumar et al., 2017; Gupta and Saurabh, 2017) and Grey Wolf Optimization (Mohanty et al., 2015; Cherukuri and Rayapudi, 2017). However, these online

Paper submitted 01/19/20; revised 05/03/20; accepted 07/06/20. Corresponding Author: Kuan-Yu Chou (e-mail: eric210092.ece01g@nctu.edu.tw)
Institute of Electrical and Control Engineering, National Chiao Tung University, Hsinchu, Taiwan.

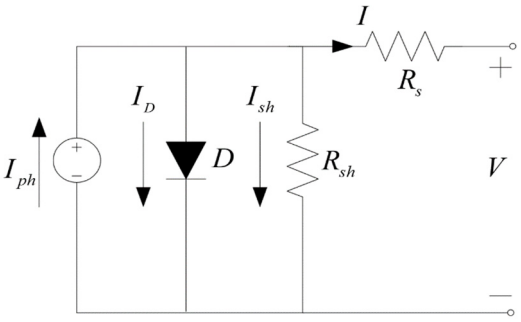

Fig. 1. Single diode solar cell model

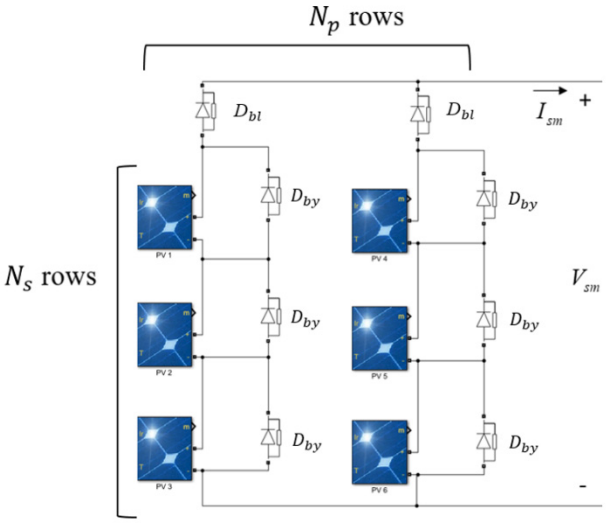

Fig. 2. The structure of the PV array

learning methods have to seek the MPP again whenever the environment changes, so the tracking speed is too slow, which would decrease the efficiency. On the other hand, some offline learning method like artificial neural network(Ramaprabha et al., 2009; Messalti et al., 2015; El-Helw et al., 2017) is proposed to increase the tracking speed, but it cannot track to the MPP precisely.

Reinforcement Learning (RL) based MPPT (Hsu et al., 2015; Youssef et al., 2016; Kofinas et al., 2017; Chou et al., 2019) is a suitable model-free method to solve the problems which need precise control and faster tracking ability. Therefore, this paper first proposed a Reinforcement Learning based MPPT under partially shaded condition by selecting the voltage and current of each module as the state to distinguish the partially shaded conditions. At first, this paper would introduce the model of PV array and the description of partially shaded conditions in Section II. And then the Reinforcement Learning based MPPT method is described in Section III. In Section IV, the simulation and implementation results would show the comparison of traditional P&O method and the proposed method to prove the performance. Finally, Section V presents the conclusions.

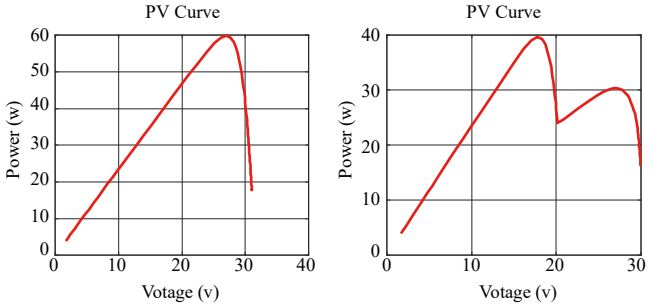

Fig. 3. (a) *P-V* curve in unshaded condition (b) *P-V* curve in partially shaded condition

II. MODELING OF THE PV ARRAY

1. The Mathematical Model of PV Module

A PV module is made up of solar cells, which are connected with each other in series or parallel, and the single-diode model(Humada et al., 2016) is often chosen for its simplicity and high efficiency in computation. Fig.1 illustrates the equivalent circuit of the single-diode model with PV current I_{ph}, diode current I_D, output current I, output voltage V, series resistor R_s and shunt resistor R_{sh}.

According to Kirchhoff's current law, the output current of a solar cell can be decomposed as

$$I = I_{ph} - I_D - I_{sh} \tag{1}$$

Where

$$I_{ph} = \frac{S}{1000}[I_{sc} + R(T - T_r)] \tag{2}$$

$$I_D = I_0\left[e^{\frac{V_D}{V_t}} - 1\right] = I_0\left[e^{\frac{q(V+IR_s)}{\eta KT}} - 1\right], V_t = \frac{\eta KT}{q} \tag{3}$$

Note that where I_0 is the diode saturation current, V_t is the thermal voltage, q is the charge of an electron, k is the Boltzmann constant, η is the diode ideality factor, T is the temperature of solar panel, S is the environment irradiance, I_{sc} is the short circuit current, R is the temperature coefficient and T_r is the reference temperature 25 °C.

2. The PV Arrays and Partially Shaded Conditions

The structure of a PV array is a series and parallel combination of several modules, and Fig.2 shows an example of PV array with $N_s \times N_p$ PV modules. Because of the shadows caused by buildings, trees or clouds, some parts of PV array may receive direct irradiance, while others may be shaded. Since the shaded modules generate less current than unshaded ones and the PV modules are connected in series, we have to add a bypass diode D_{by} to each PV module in parallel to prevent the shaded modules acting as resistive load and hotspot effect. On

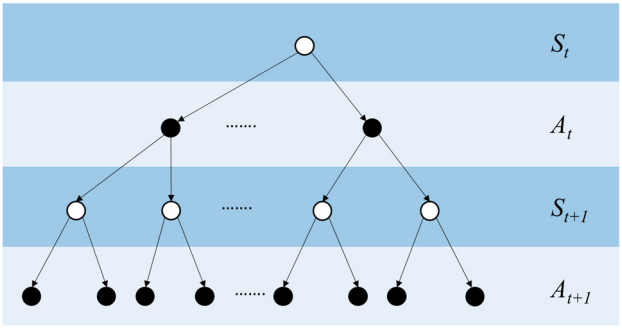

Fig. 4. The backup diagram in MDP

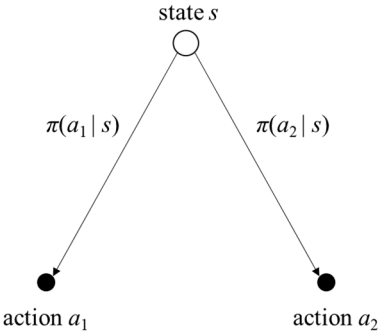

Fig. 5. The diagram of choosing action following the policy

the other hand, several strings are connected in parallel and a shaded string would provide less voltage than unshaded ones, so we have to connect a blocked diode D_{bl} to each string to prevent reverse current.

After adding the bypass diodes and blocked diodes, the characteristic *P-V* curve of the PV array is changed into a multi-peak curve that may have several local maximum and one global maximum. Fig.3 shows the same PV array in different conditions, where (a) shows the PV curve in the unshaded conditions and (b) shows the PV curve in the partially shaded condition.

III. THE PROPOSED METHOD

1. Introduction of Reinforcement Learning

Reinforcement Learning (RL) (Sutton and Barto, 2018) is a trial and error method, which can allow an innocent agent to learn a policy to approach the goal. The learning principles and processes are similar to human learning.

RL is a kind of unsupervised learning that only uses the reward signal to enable agents to learn the policy. First, the different conditions of the environment can be divided into many states. When facing an unknown environment, the agent can first interact with it randomly and then record the reward feedback signal. The reward signal provides a mechanism to determine whether the action taken under certain conditions is "good" or "bad". The actions leading to better outcomes have larger reward which would be reinforced in the future, while the actions that lead to worse outcomes may be weakened. After many times of learning, the agent can learn a policy of performing which action is the best in different states to achieve the goal. All RL problems can be described as the Markov Decision Process (MDP)(Bellman, 1957), which provides a framework model to solve problems systematically.

The fundamental concept of Markov properties is that future states depend only on the current state. The MDP is composed of $\{S, A, P, R, \gamma\}$, where S is the representation of the finite set of states and A is the set of available actions that the agent can take. P is the state transition model that describes how the state changes to the next state after an action is executed. For example, $P_{ss'}^{a}$ is the probability of transitioning from state s to state s' after executing action a, and it also can be denoted as (4). The state transition model satisfies the Markov properties as shown in (5),

$$P_{ss'}^{a} = P[S_{t+1} = s' \mid S_t = s, A_t = a] \tag{4}$$

$$P[S_{t+1} \mid S_t, A_t, S_{t-1}, A_{t-1}, S_{t-2}, A_{t-2}, ...] = P[S_{t+1} \mid S_t, A_t] \tag{5}$$

R is the reward function that defines the rewards received by the agent. For example, R_s^a is the reward that the agent would receive when applying action a in state s as shown

$$R_s^a = E[R_{t+1} \mid S_t = s, A_t = a] \tag{6}$$

where γ is the discount factor $\gamma \in [0,1]$, which is used to calculate the discount reward.

The backup diagram in MDP is depicted in Fig.4, which shows the interaction between states and actions, where the top white dot is a state St and the black dots connected to it are all available actions A_t. The white dots connected to the black point are all possible state St after doing action A_t, and the black points connected to each white points above are all available actions in state S_{t+1}.

The policy π is the rule that regulates the agent to take actions. A policy can be seen as a mapping from state s to action a, so it is the distribution over the action a given the state s as written in (7). Thus, the sum of the probabilities for each feasible actions in any state will be equal to one as below

$$\begin{aligned} a &= \pi(a \mid s) \\ \sum_{n} \pi(a \mid s) &= 1 \end{aligned} \tag{7}$$

For example, as illustrated in Fig.5, in state s, there are two available actions, and according to the policy π, the probability of executing action a_1 and a_2 are $\pi(a_1 \mid s)$ and $\pi(a_2 \mid s)$, respectively. The goal of RL is to let the agent learn the optimal policy π to choose the best action in any state.

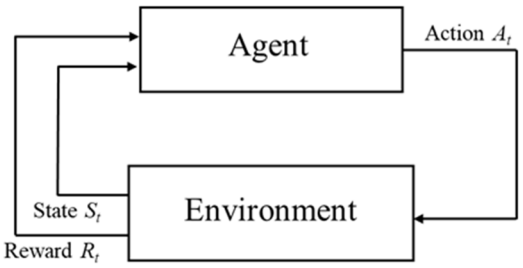

Fig. 6. The interaction between the agent and the environment

There are 5 main elements of RL, and they would be discussed in detail as below:

A. Reward

Reward $R(t)$ is a scalar feedback from the environment, which represents how well the agent did at time step t. Rewards may be delayed transmitting from the environment because sometimes we can not judge whether the results are good or bad right after taking an action. We may need more time to judge the results by calculating the cumulated reward. Therefore, the expected return Gt is defined as the cumulated discount reward that the agent expects to receive, as shown below

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \tag{8}$$

The discount factor γ is a scalar constant between 0 and 1 to prevent infinite accumulation of rewards. If the discount factor γ is closer to 1, the agent may be more far-sighted and more care about future rewards. While the discount factor is closer to zero, the agent may focus more on the instant rewards. Besides, the expected return can be written in an iterative form, as shown below:

$$\begin{aligned} G_t &= R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} \ldots \\ &= R_{t+1} + \gamma (R_{t+2} + \gamma R_{t+3} + \gamma^2 R_{t+4} \ldots) \\ &= R_{t+1} + \gamma G_{t+1} \end{aligned} \tag{9}$$

The goal of the agent is to maximize the expected return.

B. State

State $S(t)$ is the representation of the environment conditions and the basis of sending rewards $R(t)$ and observations $O(t)$ signal. Due to the difference of the observability, the environment can be divided into full observability and partial observability. Full observability means that the agent can directly observe the environment state, while partial observability means the agent only can observe some part of parameters of the environment. State $S(t)$ is the function of the history $H(t)$, which is the sequence of the observations O, actions A, rewards R as shown in (10). The history is very important because it can decide what happen next.

$$\begin{aligned} H(t) &= \{O_1, A_1, R_1, O_2, A_2, R_2, \cdots, A_{t-1}\} \\ S(t) &= f(H(t)) \end{aligned} \tag{10}$$

In many cases, the observation can be seen as the state for the agent.

We can define the state as Markov state if and only if the state satisfy the condition that the next state S_{t+1} only depends on the current state S_t as shown below

$$P[S_{t+1} \mid S_t] = P[S_{t+1} \mid S_1, S_2, \ldots\ldots S_t] \tag{11}$$

Markov state is the summary of the history because it has the sufficient information to determine the future.

C. Action

The agent can perform actions to the environment based on the state. According to the different methods to select action, the strategies can be classified into policy-based and value-based. The policy-based approach is choosing the action directly such as policy gradient method, while the value based method is selecting the action based on the value function such as Q-learning(Watkins, 1992).

D. Agent

The agent is a character to learn a policy to achieve the goal. It has two main functions. First, it can receive the state and reward signals transmitted from the environment. Second, it would follow the policy to take action based on the state and reward. The agent only can observe the state and reward passively, rather than changing the reward or state.

E. Environment

The environment is an unknown system that only can response the reward R_t and state S_t to the agent after the agent takes an action At. The interaction between the agent and the environment is illustrated in Fig.6.

The value functions are used to determine whether actions or states are good or bad, so there are action-value function and state-value function in MDP. The state-value function $v_\pi(s)$ is the expected return from the state s following the policy π, as shown below:

$$v_\pi(s) = \mathrm{E}_\pi[G_t \mid S_t = s] \tag{12}$$

If the state-value is higher, then the expected return of the state s is higher and better. When we want to calculate the expected return G_t, it does not need to wait after all timer stop. The Bellman equation is introduced to solve the value function. First, we combine (9) and (12), and then we can get an iterative form of $v_\pi(s)$

$$\begin{aligned} v_\pi(s) &= \mathrm{E}_\pi[R_t + \gamma G_{t+1} \mid S_t = s] \\ &= \mathrm{E}_\pi[R_t + \gamma v_\pi(s') \mid S_t = s] \end{aligned} \tag{13}$$

where R_t is the immediate reward at t and s' is the state at next

Table I example of Q-table

	a_1	a_2	a_3
s_1	$Q(s_1, a_1) = 3$	$Q(s_1, a_2) = 0$	$Q(s_1, a_3) = 1$
s_2	$Q(s_2, a_1) = 5$	$Q(s_2, a_2) = 7$	$Q(s_2, a_3) = 2$
s_3	$Q(s_3, a_1) = 4$	$Q(s_3, a_2) = 3$	$Q(s_3, a_3) = 6$

time step t+1. Consequently, we can use the immediate reward R_t and the state-value $v_\pi(s')$ at the next time step t+1 to calculate the state-value $v_\pi(s)$ at the current time t.

The action value function $q_\pi(s,a)$ is the expected return with choosing action a in the state s following the policy π as (14).

$$q_\pi(s,a) = \mathrm{E}_\pi[G_t \mid S_t = s, A_t = a] \tag{14}$$

Similarly, we can use the iterative form to solve the Bellman equation in (15).

$$\begin{aligned} q_\pi(s,a) &= \mathrm{E}_\pi[R_t + \gamma G_{t+1} \mid S_t = s, A_t = a] \\ &= \mathrm{E}_\pi[R_t + \gamma q_\pi(s',a') \mid S_t = s] \end{aligned} \tag{15}$$

After taking all possible states and actions into consideration, the state-value function can be rewritten as the sum of all possible action-values following the policy π in (16).

$$v_\pi(s) = \sum_{a \in \mathrm{A}} \pi(a \mid S_t = s) q_\pi(s,a) \tag{16}$$

Then the action-value function $q_\pi(s,a)$ can also be derived as (17).

$$q_\pi(s,a) = R_t + \gamma \sum_{s' \in S} P[S_{t+1} = s' \mid S_t = s, A_t = a] v_\pi(s') \tag{17}$$

After calculating the value function, the spirit of the RL is to find an optimal policy to maximize the expected return. The optimal policy π always exist when the policy π satisfies the condition that it is not worse than the other policies π', which can be written as $\pi \geq \pi$' if and only if $q_\pi(s,a) \geq q_{\pi'}(s,a)$. According to the optimal policy, we can get the optimal action-value function $q^*_\pi(s,a)$, which is defined as the maximum action-value in the action list of the state s. So the optimal action-value function can be described as below:

$$\begin{aligned} q^*_\pi(s,a) &= \max_\pi q_\pi(s,a) \\ &= \mathrm{E}_\pi[R_t + \gamma \max_\pi q^*_\pi(s',a') \mid S_t = s, A_t = a] \end{aligned} \tag{18}$$

where $q^*_\pi(s',a')$ is the optimal action-value in next state s' when the optimal action a' is performed. The backup diagram

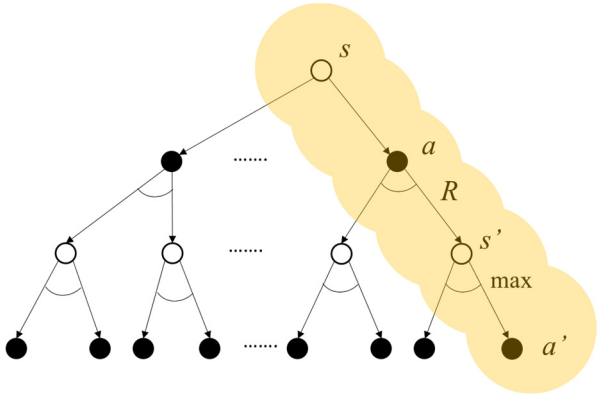

Fig. 7. The backup diagram of optimal policy

of the optimal policy is shown in yellow route in Fig.7, where the arc between the action a and a' are represented to choose the maximum value as the optimal action. The agent would choose the optimal action in any state to maximize the expected return.

2. Q-Learning

Q-learning (Minh et al., 2015) is a value-based RL method. First, the agent would construct a Q-table, which is a tabular form of the action-value function by storing individual action-values in each state. For example, Table I shows an example of Q-table with 3 states and 3 actions, where the columns (s_1~s_3) represent different states and the rows (a_1~a_3) are different actions. The values stored in the Q-table are the action-value of the action under the certain state. Suppose that when the environment is in state s_1, the optimal action a_1 is taken and the state transfers to s_2. In Q-learning, the optimal action-value is used to represent the state-value, also called state Q value in Q-learning, as shown below

$$Q(s) = \max_a Q(s,a) \tag{19}$$

The value $Q(s_1, a_1)$ is the estimate state Q value of the state s_2, but the target state Q value is the expected return in state s_2, which can be written as R+$\gamma Q(s_2)$. Thus, the purpose of Q-learning is to minimize the difference between the estimate state Q value and the target state Q value, so the update algorithm of the Q-table can be shown as

$$Q(s,a) \leftarrow Q(s,a) + \alpha * [R + \gamma \max Q(s',a') - Q(s,a)] \tag{20}$$

where α is the learning rate and $\max Q(s',a')$ is the optimal action value in state s'.

Reinforcement learning can balance the exploration and exploitation by using ε-greedy. There are ε probability to take the optimal action, and $1-\varepsilon$ probability to choose action randomly. By doing so, when taking the optimal action, the agent can reinforce the action, and when taking action randomly, the agent can explore other possible way to achieve the goal. The

Table II System element selection

RL Elements	MPPT System Corresponding Elements	
Environment	PV array and boost converter	
Agent	Controller	
State	(voltage, current)	
Action	$D = D \pm \Delta D$	$D = D_k$
Reward	$\Delta P_{rank} = P'_{rank} - P_{rank}$	

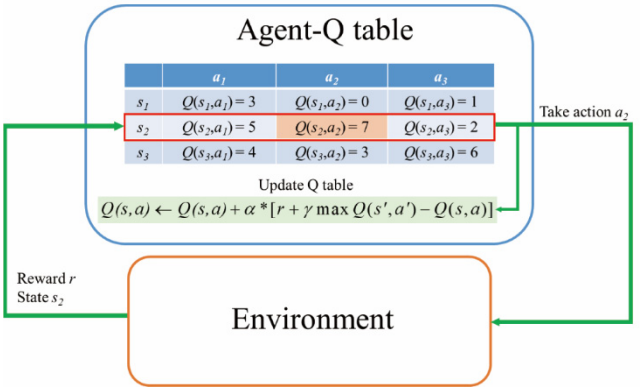

Fig. 8. The workflow of Q-learning

algorithm of Q-learning is shown in Algorithm 1 and the workflow of Q-learning is depicted in Fig.8.

Algorithm1: Q table $Q(s, a)$

Initialize Q table $Q(s, a)$

Repeat (for each iteration)

Initialize State ***s***

Repeat (for each iteration)

Choose action ***a*** from ***s*** using policy derived from Q(e.g. ε-greedy)

Take action ***a***, observe ***r***, ***s'***

$Q(s,a) \leftarrow Q(s,a) + \alpha * \left[r = \gamma \max Q(s',a') - Q(s,a) \right]$

$s \leftarrow s'$

Until ***s*** is terminal

3. The Proposed MPPT System Structure

The elements of Reinforcement Learning should be corresponded to the MPPT system as shown in Table II. It is intuitive that the PV array connected with the boost converter can be seen as the environment and the MPPT controller stands for the agent. Fig.9 illustrates the of proposed structure, which constructs 3×2 PV modules as a PV array to verify the Reinforcement Learning based MPPT using Q table (RL-QT MPPT) algorithm.

To deal with partially shaded conditions, the voltage of six PV modules (V_A, V_B, V_C, V_D, V_E, V_F) and two current(I_1, I_2) of

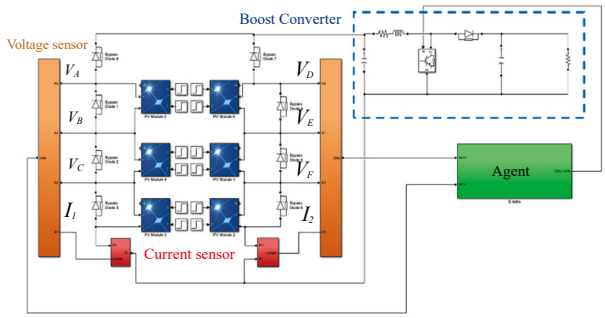

Fig. 9. The Proposed Structure

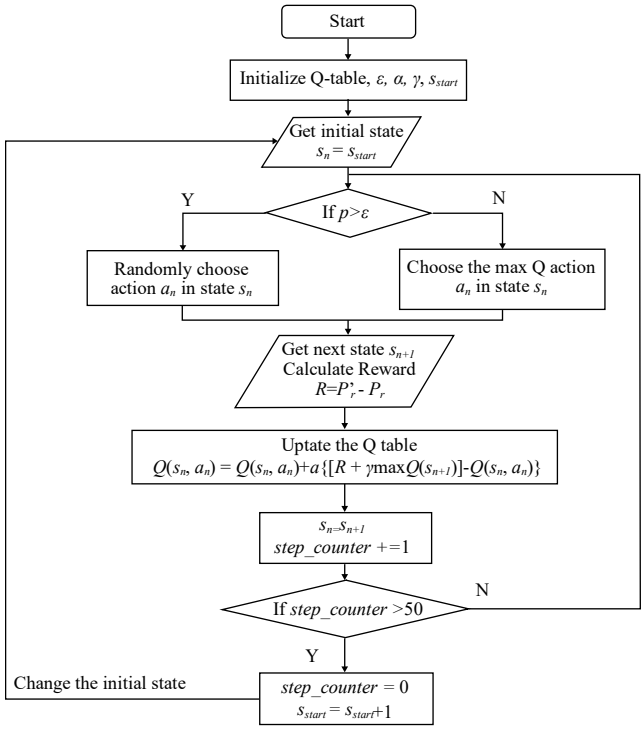

Fig. 10. The flowchart of Q-learning in training phase

each string are used as the state parameters. It is sufficient to use the voltage and current as the state parameters because they imply the information of shaded and weather condition.

The agent can receive the voltage and current of the PV array and use them as the state parameters to the algorithm. The output of the agent is an action to change the duty cycle. There are adjusting action and jumping action in the action list to change the duty cycle. The adjusting action ($D = D \pm \Delta D$) is designed to track the MPP more precisely, while the jumping action ($D = D_k$) is designed to track the MPP faster and avoids being stagnated in the local maximum.

Originally, the difference of power ΔP is used as the reward, but the ΔP around the MPP is much smaller than it in other region, which would fail in tracking the MPP or cause undesired steady state oscillation. As a result, this paper proposed

Table III Hardware configuration

	Hardware Configuration		
PV array (3×2 modules)	Each PV module parameters (Test in STC : I=1000W/s, T=25°C)	P_{max}	10 W
		V_{mpp}	9.00 V
		I_{mpp}	1.12 A
		V_{oc}	10.8 V
		I_{sc}	1.23 A
	D_{by}	1N5408	
	D_{bl}	1N5408	
Agent	Controller	Raspberry Pi 3 Mode B +	
	MOSFET Driver	TLP250	
Voltage sensor	ADS1115 16-bit ADC		
Current sensor	ACS723		
Resistive divider	R_1	9 $M\Omega$	
	R_2	1 $M\Omega$	
	R_3	5 $M\Omega$	
	R_4	1 $M\Omega$	
	R_5	2 $M\Omega$	
	R_6	1 $M\Omega$	
Boost converter	L	3.99 mH	
	D_1	1N5408	
	MOSFET	IRF840	
	C_{in}	2.09 μF	
	C_o	60.0 μF	
	R_o	100 Ω	

a method to convert P to a new parameter P_r to calculate reward. First, rearrange P by the magnitude in each condition and assign the order of each P to P_r respectively. Therefore, the reward is defined as $\Delta P_r = P_r - P_{r,old}$.

The simulation data set including voltage and current of each PV array and the duty cycle using different duty cycle under different irradiance and shaded conditions are prepared before training process. In the proposed method, the Q-learning can be divided into training phase and tracking phase.

In training phase, the initial state s_{start}, learning rate α, ε-greedy, discount factor γ would be initialized and the Q-table would be set to 0. Using the ε-greedy technique, with the probability ε, the agent would take the optimal action with maximum action value in the Q-table. Otherwise the agent would randomly take action. After taking an action, the agent can get the next state s' from the simulation data and calculate the reward from the difference of P_r' and P_r. According to the next state s' and the reward R, the Q-table can be updated by (20), and then the state s would also be transferred to the next state s'. In order to fill the Q table, each state would be set as the initial state s_{start} to train the Q-table for an episode (50 steps), so after 50 steps the agent would change to another initial state s_{start}. Fig.10 show the flowchart of Q-learning in training phase.

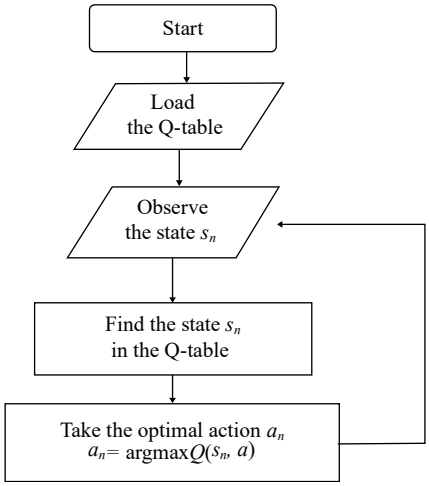

Fig. 11. The flowchart of Q-learning in tracking phase

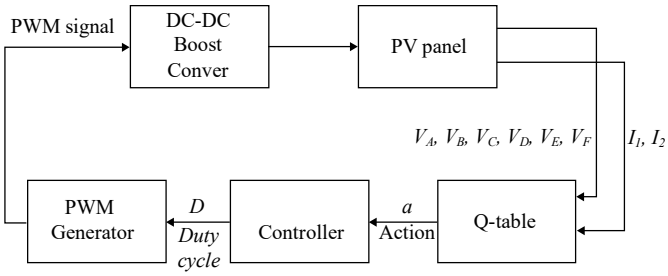

Fig. 12. The control block diagram of the proposed MPPT system

In tracking phase, the Q-table is supposed to be trained properly. Therefore, first sense the state parameters to find the state in the Q-table and take the optimal action to change the duty cycle. The agent only takes the optimal action without updating the Q table in tracking phase. After several steps, the controller can drive the operating point to the MPP successfully. Fig.11 show the flowchart of Q-learning in tracking phase.

The control block diagram of the proposed MPPT system is shown in Fig.12. First, the Q table could calculate the optimal action a based on the voltage of six PV modules (V_A, V_B, V_C, V_D, V_E, V_F) and two current (I_1, I_2) measured from the PV array. Second, the controller decide the duty cycle D of the PWM generator based on the action a. Finally, the PWM signal is generated to control the DC-DC boost converter to track the global MPP of the PV array.

IV. RESULTS

1. System configuration

The system configuration of simulation and implementation are the same, and the overall hardware structure of the system configuration is shown in Fig.13, which contains a PV array, two current sensors, two voltage sensors, resistive divider of R_1~R_6, the boost converter and the agent. The specific circuit

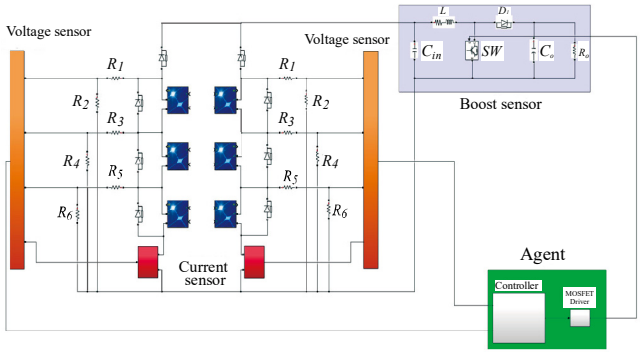

Fig. 13. The hardware structure of the MPPT system

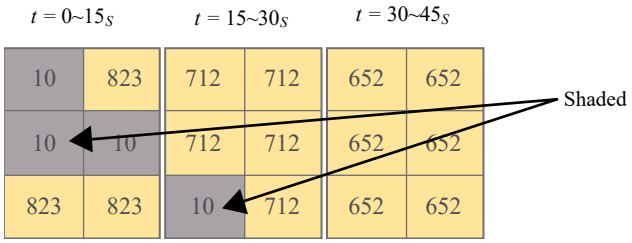

Fig. 14. The dynamic shaded condition in simulation, and the gray area represent that the PV panel is shaded.

selection of the system is shown in Table III. The PV array is constructed by connecting 3 PV modules in series as a string and connecting 2 strings in parallel, where the specification of each module is also shown in Table III. The Raspberry Pi 3 Model B + is used as the controller because of the great arithmetic capability, but the PWM signal of the GPIO port is not sufficient enough to drive the MOSFET switch of the boost converter. Thus, the MOSFET driver TLP250 is used to enhance the current driving capability. However, there is no built-in analog-to-digital converter (ADC) in the Raspberry Pi 3 Mode B +, so the additional ADS1115 16-bit ADC is required to measure the analog voltage of the PV array. In addition, the current sensor converts the current to an analog voltage output, so the value should be converted by the ADC. Since the ADS1115's input range is -4.096 V to +4.096 V, the resistive divider is used to reduce the input voltage of the ADC. The actual voltage value can be obtained by multiplying the quotient of the two series resistors. Therefore, the resistive divider is designed to be large enough to reduce its impact on the system.

Before the training process, the training data should be prepared. The training data contain the state parameters (V_A, V_B, V_C, I_1, V_D, V_E, V_F, I_2), the duty cycle D and the output power of the PV array under any environment condition, which is obtained by sweeping the duty cycle from 0.25 to 0.95 and the level of D is discretized to 0.01 using MATLAB 2017b Simulink. The effect of temperature is ignored and set all temperature to 25°C. The shaded condition is divided into 64 cases, depending on the module that is "shaded" or "unshaded". The

Table IV Agent configuration

	RL-QT MPPT	
D range	0.25~0.95 (71 points in total)	
Irradiance Condition	{50,100,150,……,1000}	
Shaded Condition	2^6(shaded or unshaded)	
Sampling time	1s	
State	(V_A, V_B, V_C, V_D, V_E, V_F, I_1, I_2)	
Action list	Adjusting actions	$D=D+\Delta D$ $\Delta D=\{0,\pm0.01,\pm0.05\}$
	Jumping actions	$D=D_k$ $D_k=\{0.35,0.55,0.75\}$
Reward	ΔP_{rank}	
ε	0.7	
γ	0.8	
Q value storing type	Q-table 102240*8 (71*20*2^6)	
α	0.01	

irradiance of the shaded module is set to 10 W/m^2, while the irradiance of the unshaded module is divided to 20 cases, which is simulated from 50 W/m^2 to 1000 W/m^2 and the level of irradiance is discretized to 50 W/m^2. Therefore, there are 1280(64*20) cases of the environment conditions, and 90880 (71*20*26) states in the training data. The training data should be preprocessed first by transforming the actual power P to P_r for calculating the reward R. In the training phase of Q-learning, after the agent takes an action a, the next state s' can be obtained by searching the next state duty cycle D in the training data, and the reward R is also calculated by P_r and $P_{r,old}$.

The experience replay period is set to 10 so that it would be performed every 10 steps to sample 50 pieces of data in the memory whose size is 5000. There are 8 actions in the action list. The adjusting actions contain five different ΔD (0, ±0.01, ±0.05), and the jumping actions include three different D_k (0.35, 0.55, 0.75). In addition, the parameters such as ε, γ and α are selected by trial and error methods. Lastly, the agent configuration of the Q-learning is shown in Table IV.

2. Simulation results

The performance of the P&O and RL-QT MPPT methods are simulated by MATLAB 2017b and Simulink with Intel i7-8750H, 2.2GHz processor, 8GB RAM and windows 10 operating system. The circuit parameters are given in Table V. Due to the partially shaded condition, the multi-peaks of the P-V curve would make it difficult to track the MPP by using the P&O method. Therefore, the simulation results of the P&O method and RL-QT method under dynamic environment conditions are shown below.

In dynamic shaded condition, the shaded condition would change at t=15s and t=30s as shown in Fig.14. To verify the

Table V Circuit Parameters

	PV array and boost converter
PV array	3 modules in series and 2 strings in parallel
Maximum power (each modules) P_{max}	10W
Open circuit voltage V_{oc}	9.8V
Short circuit current I_{sc}	1.23A
Input capacitance	20.8μF
Input inductance	3.9μH
Output capacitance	6μF
Output payload resistance	100Ω

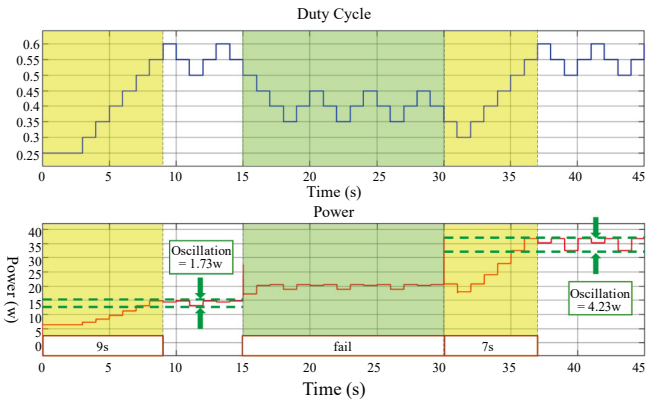

Fig. 15. The simulation results of P&O method in dynamic condition

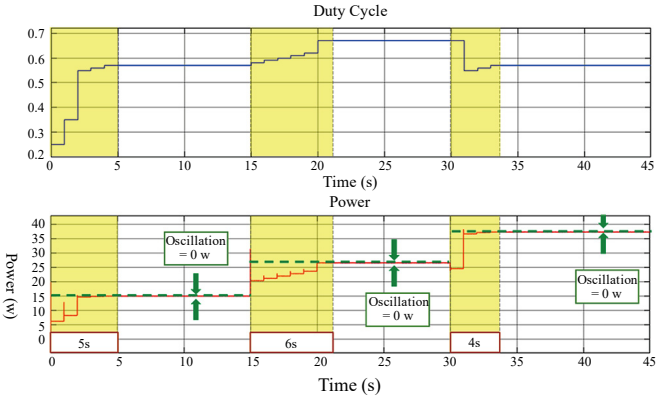

Fig. 16. The simulation results of RL-QT method in dynamic condition

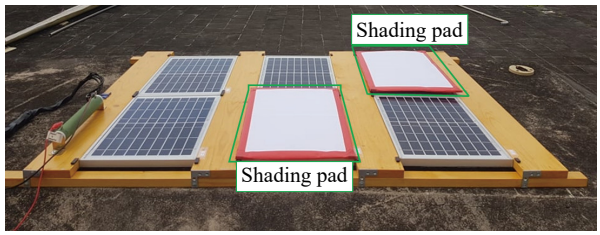

Fig. 17 The shaded condition of implementation

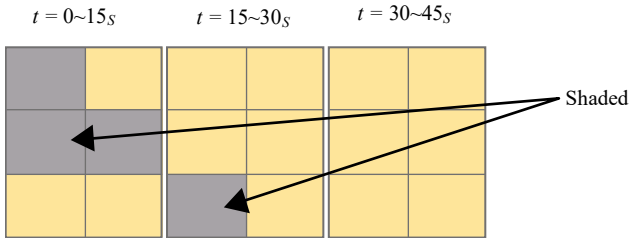

Fig. 18. The dynamic shaded condition in implementation, and the gray area represent that the PV panel is shaded.

efficiency of the proposed RL-QT method in dynamic variation of weather and shaded conditions, three different shaded conditions are used which have global MPPs 15.08W, 26.62 W and 37.4 W respectively. Fig.15 shows the simulation results of the P&O method with ΔD=0.05, and the yellow area indicates the tracking time and the green area indicates that the algorithm fails on tracking the MPP. It is obvious that during t=15~30s the P&O method only tracks the local MPP 20.5W, instead of the global MPP 26.65W. Besides, during t=0~15s and t=30~45s, the undesired oscillation around the MPP is 1.73W and 4.73W. However, the simulation result of the proposed RL-QT method can track the MPP successfully without any oscillation in Fig.16.

3. Implementation results

Partially shaded conditions are simulated by covering a shading pad as shown in Fig.17, where the shading pad is made of the thick cardboard. The implementation results of the P&O method and RL-QT methods are given respectively. However, there is still a little inevitable changing of the irradiance during data acquisition, so there is still a little oscillation of power in the implementation.

This implementation case is similar to the simulation case, but the irradiance varies with the weather. The dynamic condition in Fig.18 is used to test the performance of the algorithm when the shaded condition changes, and the shaded condition would change at t=15s and t=30s.

Fig.19 and Fig.20 show the implementation results of the P&O and RL-QT methods, where the green areas indicate that the algorithm failed to track the global MPP and the yellow areas represent the tracking stage. The oscillations of power and average power are denoted as ΔP and P_{avg}, respectively. However, since it is actually difficult to control the irradiance during a long period of experiment, after the algorithm succeeds to track the MPP, the operating point still changes with the weather condition. Besides, due to the noise of the ADC, there is a little oscillation in the implementation results. Therefore, these implementation results are generated under similar but different environment conditions.

Fig.19 is the implementation result of the P&O method,

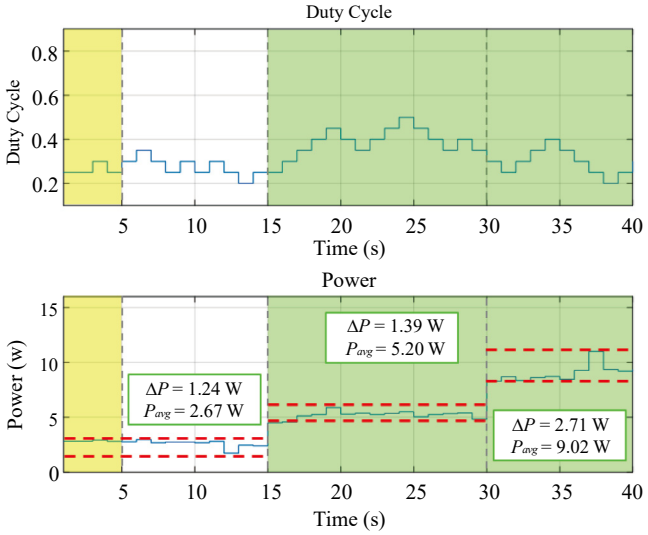

Fig. 19. The implementation results of P&O method in dynamic condition

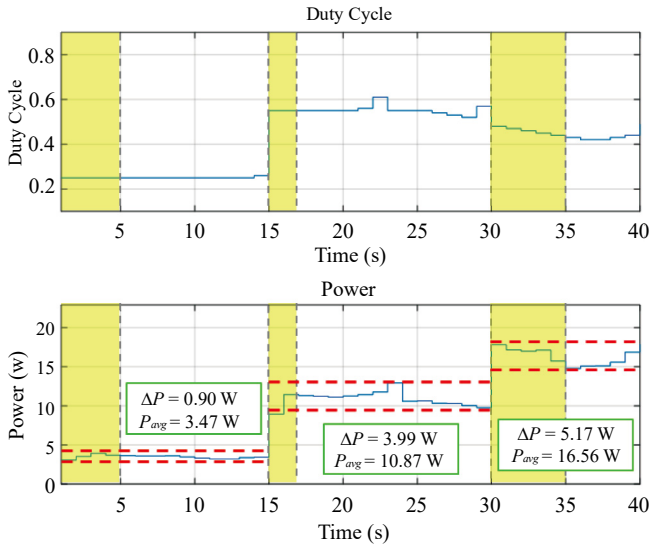

Fig. 20. The implementation results of RL-QT method in dynamic condition.

which shows that the algorithm fails to track the global MPP at t=15s and t=30s, because the duty cycle and P_{avg} are far from the results of RL-QT method. In addition, the oscillation of duty cycle is much larger than the RL-QT method. Fig.20 is the implementation result of RL-QT method, which shows that the algorithm can track the global MPP successfully all the time. The oscillation during t=15~40s and the rising of power at t=37s are caused by the changing of irradiance.

V. CONCLUSIONS

In this paper, a Reinforcement learning using Q table based maximum power point tracking(RL-QT MPPT) is proposed for the PV array under partially shaded condition. The RL-QT MPPT method applies the voltage and current of each PV module are used to stand for the states to distinguish the partially shaded conditions. The actions of tracking process are the different ways to control the duty cycle including adjusting actions and jumping actions. The jumping actions can move the operating point away from the local MPP, while adjusting actions can track the global MPP more precisely. The improved P_r is used to calculate the reward in Reinforcement learning. The Q learning algorithm is implemented on the MPPT system by constructing a Q table to store the state and action value. In the tracking phase, the agent choose the action with optimal action value to achieve the global MPP. Further, the numerical simulation results obvious show that the proposed RL-QT MPPT method can track the MPP faster and more accurate without undesired oscillation compared to the traditional P&O method under various environmental factors.

ACKNOWLEDGEMENT

The authors would like to thank the Ministry of Science and Technology, Taiwan under Grant MOST 108-2221-E-009 -119 - for providing research funding.

REFERENCES

Al Nabulsi, A. and R. Dhaouadi (2012). Efficiency optimization of a DSP-based standalone PV system using fuzzy logic and dual-MPPT control. IEEE Transactions on Industrial Informatics 8(3), 573-584.

Algazar, M. M., H. A. El-Halim and M. E. E. K. Salem (2012). Maximum power point tracking using fuzzy logic control. International Journal of Electrical Power & Energy Systems 39(1), 21-28.

Bellman, R. (1957). A Markovian decision process. Journal of mathematics and mechanics, 679-684.

Bianconi, E., J. Calvente, R. Giral, E. Mamarelis, G. Petrone, C. A. Ramos-Paja, G. Spagnuolo and M. Vitelli (2012). A fast current-based MPPT technique employing sliding mode control. IEEE Transactions on Industrial Electronics 60(3), 1168-1178.

Chou, K. Y., S. T. Yang and Y. P. Chen (2019). Maximum power point tracking of photovoltaic system based on reinforcement learning. Sensors 19(22), 5054.

Cheikh, M. A., C. Larbes, G. T. Kebir and A. Zerguerras, (2007). Maximum power point tracking using a fuzzy logic control scheme. Revue des energies Renouvelables 10(3), 387-395.

Cherukuri, S. K. and S. R. Rayapudi (2017). Enhanced Grey Wolf optimizer based MPPT algorithm of PV system under partial shaded condition. International Journal of Renewable Energy Development 6(3), 203.

Elgendy, M. A., B. Zahawi, and D. J. Atkinson (2011). Assessment of perturb and observe MPPT algorithm implementation techniques for PV pumping applications. IEEE transactions on sustainable energy 3(1), 21-33.

El-Helw, H. M., A. Magdy, and M. I. Marei (2017). A hybrid maximum power point tracking technique for partially shaded photovoltaic arrays. IEEE access 5, 11900-11908.

Gupta, S. and K. Saurabh (2017). Modified artificial killer whale optimization algorithm for maximum power point tracking under partial shading condition. In 2017 International Conference on Recent Trends in Electrical, Electronics and Computing Technologies (ICRTEECT) (pp. 87-92). IEEE.

Hsu, R. C., C. T. Liu, W. Y. Chen, H. I. Hsieh, and H. L. Wang (2015). A reinforcement learning-based maximum power point tracking method for photovoltaic array. International Journal of Photoenergy, 2015.

Humada, A. M., M. Hojabri, S. Mekhilef and H. M. Hamada (2016). Solar cell parameters extraction based on single and double-diode models: A review. Renewable and Sustainable Energy Reviews, 56, 494-509.

Kofinas, P., S. Doltsinis, A. I. Dounis and G. A. Vouros (2017). A reinforcement learning approach for MPPT control method of photovoltaic sources. Renewable Energy 108, 461-473.

Kollimalla, S. K. and M. K. Mishra (2014). A novel adaptive P&O MPPT algorithm considering sudden changes in the irradiance. IEEE Transactions on Energy conversion, 29(3), 602-610.

Kumar, N., I. Hussain, B. Singh and B. K. Panigrahi (2017). MPPT in dynamic condition of partially shaded PV system by using WODE technique. IEEE Transactions on Sustainable Energy, 8(3), 1204-1214.

Li, H., D. Yang, W. Su, J. Lü and X. Yu (2018). An overall distribution particle swarm optimization MPPT algorithm for photovoltaic system under partial shading. IEEE Transactions on Industrial Electronics, 66(1), 265-275.

Levron, Y. and D. Shmilovitz (2013). Maximum power point tracking employing sliding mode control. IEEE Transactions on Circuits and Systems I: Regular Papers, 60(3), 724-732.

Messalti, S., A. G. Harrag and A. E. Loukriz (2015). A new neural networks MPPT controller for PV systems. In IREC2015 The Sixth International Renewable Energy Congress (pp. 1-6). IEEE.

Miyatake, M., M. Veerachary, F. Toriumi, N. Fujii and H. Ko (2011). Maximum power point tracking of multiple photovoltaic arrays: A PSO approach. IEEE Transactions on Aerospace and Electronic Systems, 47(1), 367-380.

Mnih, V., K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg and D. Hassabis (2015). Human-level control through deep reinforcement learning. Nature 518(7540), 529.

Mohanty, S., B. Subudhi and P. K. Ray (2015). A new MPPT design using grey wolf optimization technique for photovoltaic system under partial shading conditions. IEEE Transactions on Sustainable Energy 7(1), 181-188.

Mohapatra, A., B. Nayak, P. Das and K. B. Mohanty (2017). A review on MPPT techniques of PV system under partial shading condition. Renewable and Sustainable Energy Reviews 80, 854-867.

Pradhan, R. and B. Subudhi (2015). Double integral sliding mode MPPT control of a photovoltaic system. IEEE Transactions on Control Systems Technology 24(1), 285-292.

Ramaprabha, R., B. L. Mathur and M. Sharanya (2009). Solar array modeling and simulation of MPPT using neural network. In 2009 International Conference on Control, Automation, Communication and Energy Conservation (pp. 1-5). IEEE.

Renaudineau, H., F., Donatantonio, J. Fontchastagner, G. Petrone, G. Spagnuolo, J. P. Martin and S. Pierfederici (2014). A PSO-based global MPPT technique for distributed PV power generation. IEEE Transactions on Industrial Electronics 62(2), 1047-1058.

Sutton, R. S. and A. G. Barto (2018). Reinforcement learning: An introduction. MIT press.

Watkins, C. J. and P. Dayan (1992). Q-learning. Machine learning 8(3-4), 279-292.

Youssef, A., M. El-Telbany and A. Zekry (2016). Reinforcement learning for online maximum power point tracking control. J. Clean Energy Technol 4, 245-248.

Zainuri, M. M., M. M. Radzi, A. C. Soh and N. A. Rahim (2012). Adaptive P&O-fuzzy control MPPT for PV boost dc-dc converter. In 2012 IEEE International Conference on Power and Energy (PECon), 524-529.