energies

*Article*

# Comparison of Deep Reinforcement Learning and PID Controllers for Automatic Cold Shutdown Operation

**Daeil Lee [1], Seoryong Koo [2], Inseok Jang [2] and Jonghyun Kim [1,*]**

[1] Department of Nuclear Engineering, Chosun University, Dong-gu, Gwangju 61452, Korea; dleodlf1004@chosun.kr
[2] Korea Atomic Energy Research Institute, Yuseong-gu, Daejeon 34057, Korea; srkoo@kaeri.re.kr (S.K.); isjang@kaeri.re.kr (I.J.)
* Correspondence: jonghyun.kim@chosun.ac.kr

**Abstract:** Many industries apply traditional controllers to automate manual control. In recent years, artificial intelligence controllers applied with deep-learning techniques have been suggested as advanced controllers that can achieve goals from many industrial domains, such as humans. Deep reinforcement learning (DRL) is a powerful method for these controllers to learn how to achieve their specific operational goals. As DRL controllers learn through sampling from a target system, they can overcome the limitations of traditional controllers, such as proportional-integral-derivative (PID) controllers. In nuclear power plants (NPPs), automatic systems can manage components during full-power operation. In contrast, startup and shutdown operations are less automated and are typically performed by operators. This study suggests DRL-based and PID-based controllers for cold shutdown operations, which are a part of startup operations. By comparing the suggested controllers, this study aims to verify that learning-based controllers can overcome the limitations of traditional controllers and achieve operational goals with minimal manipulation. First, to identify the required components, operational goals, and inputs/outputs of operations, this study analyzed the general operating procedures for cold shutdown operations. Then, PID- and DRL-based controllers are designed. The PID-based controller consists of PID controllers that are well-tuned using the Ziegler–Nichols rule. The DRL-based controller with long short-term memory (LSTM) is trained with a soft actor-critic algorithm that can reduce the training time by using distributed prioritized experience replay and distributed learning. The LSTM can process a plant time-series data to generate control signals. Subsequently, the suggested controllers were validated using an NPP simulator during the cold shutdown operation. Finally, this study discusses the operational performance by comparing PID- and DRL-based controllers.

**Keywords:** nuclear power plant; autonomous operation; artificial intelligence; deep reinforcement learning; soft actor-critic algorithm

check for updates

**Citation:** Lee, D.; Koo, S.; Jang, I.; Kim, J. Comparison of Deep Reinforcement Learning and PID Controllers for Automatic Cold Shutdown Operation. *Energies* **2022**, *15*, 2834. https://doi.org/10.3390/en15082834

Academic Editor: Hany Abdel-Khalik

Received: 10 March 2022
Accepted: 11 April 2022
Published: 13 April 2022

## 1. Introduction

Nuclear power plants (NPPs) use highly automated controllers to increase availability and reduce accident risk and operating costs [1,2]. In addition, digitalized controllers help process large amounts of data, improve system reliability, automate periodic tests, perform diagnosis, and increase operation capability [3].

Startup and shutdown operations in NPPs largely rely on the operator's manual controls, whereas the full power operation is highly automated. Thus, these operations are known to be error-prone for the following reasons [4,5]:

- There is a significantly increased need for decision-making, such as selecting the power operation target and determining the control strategy based on guidelines from the operating procedures.

- Many manual actions owing to extensive maintenance, tests, and monitoring of plant parameters.
- Manipulation of components for which the automatic system and safety functions may be disabled.

Therefore, automation of startup and shutdown operations is expected to reduce the operator's burden and errors.

Typical approaches to automatic controllers in current NPPs include proportional-integral-differential (PID) controllers, programmable logic controllers (PLCs), and field-programmable gate arrays (FPGAs) [6–9]. For safety systems, the PLC is generally used to automatically act as a fast and reliable response to prevent malfunctions from propagating into major accidents. For non-safety systems, PID controllers or controllers that combine two out of three types of controllers (e.g., proportional-integral controllers) are the most popular among the existing NPPs. These controllers generally aim to stabilize a system within a defined range.

To tune the PID controller, traditional tuning methods have been applied, such as Ziegler–Nichols [10], Cohen-Coon [11], and Astrom and Hagglund [12]. However, traditional methods still need re-tuning before being applied to industrial processes because the methods may cause high overshoots, large oscillations, and longer settling times for higher-order systems [13]. To enhance the capabilities of traditional PID parameter tuning techniques, intelligent tuning methods have been presented. Davut Izci et al. proposed a Harris hawks optimization (HHO) algorithm, which is a novel meta-heuristic algorithm, to achieve optimal parameters of a PID controller adopted for an aircraft pitch control system [14]. For DC motor control, Erdal Eker et al. improved an atom search optimization algorithm by using simulated annealing (SA) [15], and Mahmud Iwan Solihin et al. compared tuning algorithms between ZN and particle swarm optimization [16]. For a mobile robot, Ignacio Carlucho et al. proposed self-adaptive multiple PID controllers using the DRL [17].

Recently, controllers applying artificial intelligence (AI) techniques have been studied in several industrial fields [18]. Since the 2000s, deep-learning techniques have drawn attention for several reasons: increasing computing power, increasing data size, and advances in deep-learning research [19,20]. Among them, deep reinforcement learning (DRL) is a popular approach because it has a training mechanism very similar to that of humans. A DRL-based controller develops its own experiences through trial and error, similar to humans. In addition, this DRL-based controller can perform tasks that classical controllers cannot perform, such as selecting an operation strategy, operating systems, making decisions based on current conditions, and optimizing operations. For this reason, DRL-based controllers have been developed in robotics [6,21], autonomous vehicles [7–9,22,23], smart building [18,24], power management [25–28], railway industry [29], wind turbine [30], traffic signal [31], and nuclear power plants [32,33].

Although AI-based controllers have been developed in several studies, they are not applied to NPPs at a practical level. This is mainly because AI-based controllers do not sufficiently prove their performance to guarantee robustness and correctness and solve regulatory issues, such as the transparency of the algorithm. However, it is very likely that the AI-based controller implemented as part of autonomous reactor controls will be an important aspect of small modular reactors and microreactors that can be operated remotely by an offsite operations crew [34].

This study compares the performances of DRL and PID controllers in the cold shutdown operation of NPPs. First, this study analyzes the general operating procedures (GOPs) of the bubble creation operation, which is part of the cold shutdown operation. It identifies the operational goals and manual controls by operators, and defines the inputs and outputs for the automatic controllers. Subsequently, a DRL-based controller is developed by combining a rule-based system, long short-term memory (LSTM), and soft actor critic (SAC). Then, a PID controller was developed using the Ziegler–Nichols and

DRL-based tuning methods. Finally, the performances of both controllers were compared and discussed.

## 2. Task Analysis of Bubble Creation Operation in the Cold Shutdown Condition

To develop automatic algorithms for cold shutdown operations that are not automated in Korean NPPs, the current operating strategies were considered. First, this study analyzed GOPs and performed a task analysis to identify the operation goal and define the inputs and outputs of operational tasks during the cold shutdown operation. These are used as inputs and outputs in the automatic controllers.

### *2.1. Overview of Cold Shutdown Operation*

The cold shutdown operation is included in the GOPs, which provide instructions to start up the reactor and increase its power after refueling. A Westinghouse-900 MW pressurized water reactor (PWR) was used as the reference plant in the task analysis. The reference plant had six GOPs, as listed below [35,36]:

- Reactor coolant system filling and venting;
- Cold shutdown to hot shutdown;
- Hot shutdown to hot standby;
- Hot standby to 2% reactor power;
- Power operation at than 2% power;
- Secondary systems heatup and startup.

Figure 1 shows the trend of the important parameters in the startup operation along with the relevant procedures. These parameters provide milestones for operators to achieve successful startup operations. The cold shutdown operation, i.e., the focus of this study, is located in the gray area in Figure 1.

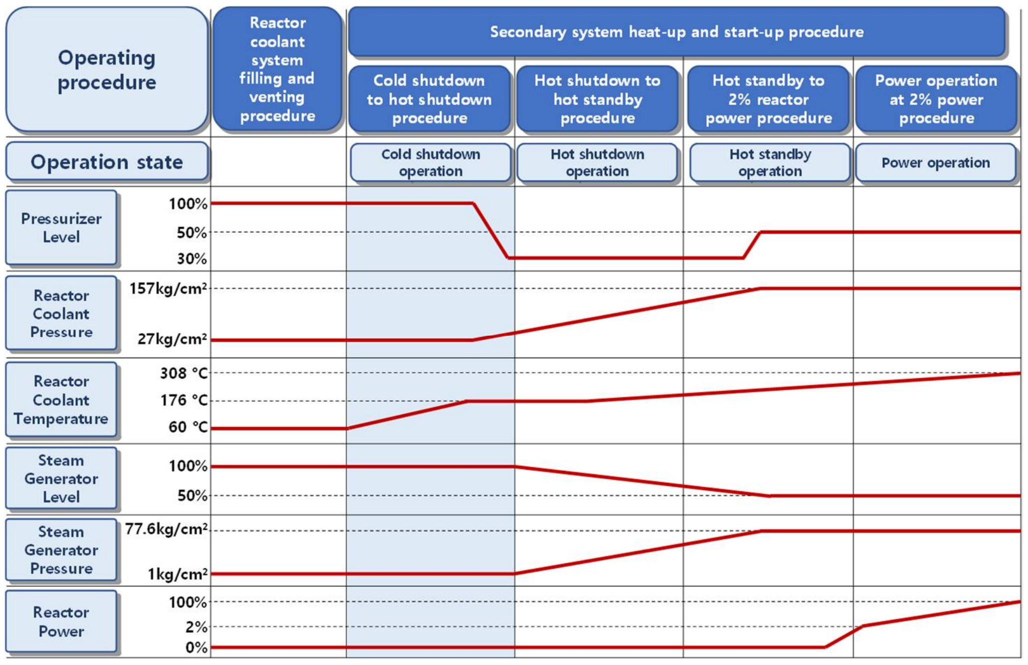

**Figure 1.** Significant parameters of the startup operation.

The cold shutdown procedure provides instructions for heating the plant from the cold shutdown condition to the hot shutdown condition (Tavg < 176.7 °C, Keff < 0.99). This operation allows the components to increase the temperature of the primary system by maintaining pressure in the pressurizer. The goal of the cold shutdown operation is to create bubbles in the pressurizer, that is, the bubble creation operation, and then to control the pressure and level of the pressurizer.

Figure 2 shows a simplified schematic of the components related to cold shutdown operation. The initial and final conditions of the operation of the plant variables are shown in Table 1.

**Table 1.** Initial and final conditions of the cold shutdown operation.

| Major Parameter | Initial Condition | Final Condition |
|---|---|---|
| Pressurizer pressure | 27 kg/cm$^2$ | 27 kg/cm$^2$ |
| Pressurizer temperature | 84 °C | 210 °C |
| Average temperature | 81 °C | 176 °C |
| Pressurizer level | 100% | 50% |
| Back-up heater | Off | On |
| Proportional heater | 0% | 100% |
| Letdown valve | 0% | ≈40% |
| Pressurizer spray valve | 0% | ≈30% |
| Charging valve | 0% | ≈60% |

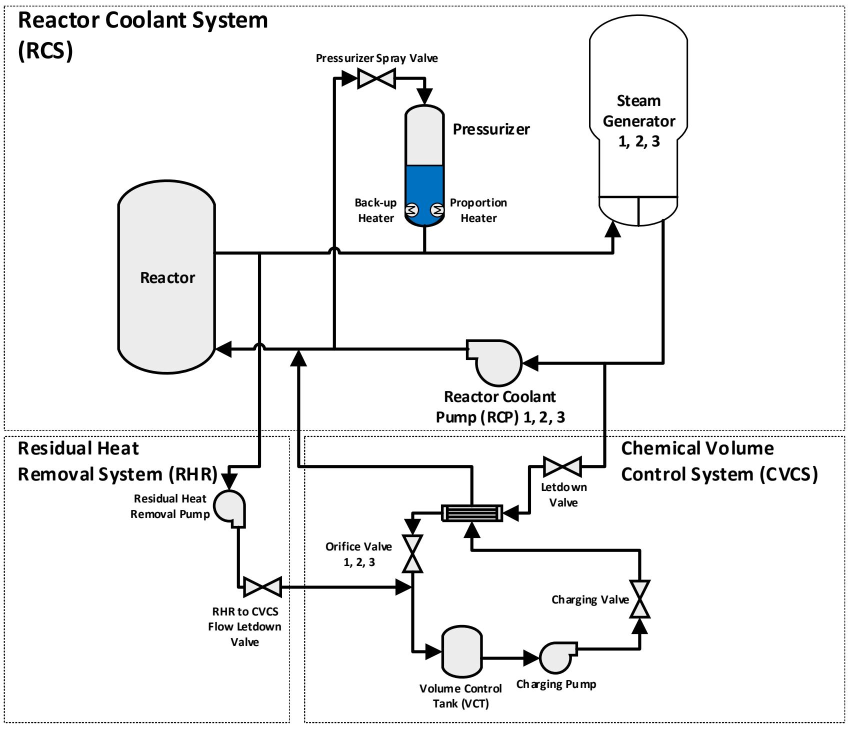
Reactor Coolant System (RCS)
Pressurizer Spray Valve
Pressurizer
Steam Generator 1, 2, 3
Back-up Heater
Proportion Heater
Reactor
Reactor Coolant Pump (RCP) 1, 2, 3
Residual Heat Removal System (RHR)
Residual Heat Removal Pump
RHR to CVCS Flow Letdown Valve
Chemical Volume Control System (CVCS)
Letdown Valve
Orifice Valve 1, 2, 3
Charging Valve
Volume Control Tank (VCT)
Charging Pump

**Figure 2.** Simplified schematic of related components.

The first step of the cold shutdown operation is to heat the coolant in the primary system by turning on all the pressurizer heaters (e.g., back-up and proportional heaters) and starting the reactor coolant pump. This leads to an increase in the temperature of the primary system and pressure inside the pressurizer. The pressure of the primary system, that is, the reactor coolant system (RCS), should be maintained between 25 kg/cm$^2$ and 29 kg/cm$^2$ despite the increase in the pressurizer temperature. Thus, the increase in pressure can be prevented by opening a letdown valve that handles the letdown flow rate from the RCS to a residual heat removal system (RHR). When the pressurizer temperature reached the saturation point of approximately 200 °C, its level decreased. A space filled with saturated steam was created on top of the pressurizer, allowing pressure to be controlled through the pressurizer spray. Subsequently, the level inside the pressurizer was maintained at 50% by adjusting the charging flow rate. It is known that this operation normally requires 8 h for actual NPPs.

### *2.2. Task Analysis of Cold Shutdown Operation*

Task analysis identifies the objective of each operator action and defines the inputs and outputs of the actions to design the controllers. Table 2 presents the results of the task analysis of the operating procedure. The step numbers and tasks are the step numbers and instructions described in the procedure, respectively. Subsequently, task types are classified into control or check tasks. If the task type was "control", the task included action(s) on a component. The task type "check" is to check or monitor plant states without performing any action on components. The inputs and outputs of each task are then defined. The information necessary to design the controllers is then extracted by focusing on the task type of the control. Finally, four tasks were selected for implementation using the control algorithm, as listed in Table 3.

**Table 2.** Task analysis result for cold shutdown operation.

| Step Number | Task | Task Type |
|---|---|---|
| 1 | Cold shutdown operation should be completed within 8 h. | Check |
| 2 | The pressurizer pressure should be maintained between 25 kg/cm$^2$ and 29 kg/cm$^2$ during cold shutdown operation. | Check |
| 3 | The RHR system should be isolated from RCS before the pressurizer temperature reaches 200 °C or its pressure reaches 30 kg/cm$^2$. | Check |
| 4 | The reactor coolant loops and the pressurizer are filled and vented. | Check |
| 5 | The reactor coolant boron concentration is greater than or equal to that of the cold shutdown condition. | Check |
| 6 | The residual heat removal (RHR) system is served with all loop isolation valves open and one operational RHR. | Check |
| 7 | Close main steam isolation valves. | Check |
| 8 | Close steam generator (S/G) power operated relief valves. | Check |
| 9 | The makeup control system is in Auto mode. | Check |
| 10 | Letdown is established via the RHR letdown line, and three-letdown orifice isolation valves are open. | Check |
| 11 | Pressurizer spray control valve, power-operated relief valve (PORV), and PORV block valve are in manual and closed modes. | Check |
| 12 | The safety injection (SI) initiation signal is blocked. | Check |
| 13 | Maintain the RCS pressure between 25 kg/cm$^2$ and 29 kg/cm$^2$ by adjusting the letdown valve (RHR to CVCS flow). | Control |
| 14 | Maintain the flow through the bypass valve of the RHR heat exchanger. | Check |
| 15 | Check the PORV position, whether closed or open. If the PORV position is open, close PORV. | Check |
| 16 | If PORV block valves are in man mode, put the associated PORV block valves to auto mode. | Check |
| 17 | Energize all pressurizer heater groups and start increasing the pressurizer temperature. | Control |
| 18 | Open all three-letdown orifice isolation valves and establish letdown via RHR. | Check |
| 19 | If the pressurizer level goes down owing to reaching saturation point, maintain the pressure between 25 kg/cm$^2$ and 29 kg/cm$^2$ by adjusting pressurizer spray valve. | Control |
| 20 | Maintain the pressurizer level at 50% by adjusting charging valve. | Control |

**Table 3.** Simplified operational task for cold shutdown operation.

| Task | Input | Output | Control Type | Constraints |
|---|---|---|---|---|
| Put back-up heater from Off to On. | Back-up heater state | Back-up heater control | Discrete control | (1) Maintain RCS pressure between 26 kg/cm$^2$ and 28 kg/cm$^2$. (2) Maintain pressurizer level at 50%. |
| Increase the power of proportional heater from 0% to 100%. | Proportional heater state | Proportional heater control | | |
| Adjust letdown valve (RHR to CVCS flow) within the RCS pressure boundary. | RCS pressure, Letdown valve position | Letdown valve control | Continuous control | |
| Adjusting spray valve within the RCS pressure boundary. | RCS pressure, Spray valve position | Spray valve control | | |
| Adjust charging valve to maintain pressurizer level. | Pressurizer level, Charging valve position | Charging valve control | | |

Control tasks were also classified as discrete or continuous controls. Discrete control has two separate states: "on or off" or "fully open or closed". For example, the task "controlling the proportional heater" in Table 3 is an example of discrete control because the heater only has two states: on or off. In contrast, continuous control adjusts the state of the component to satisfy the specific value of the parameters. In the cold shutdown operation, controlling the charging valve, letdown valve (RHR to CVCS flow), and spray valve belong to the category of continuous control because the positions of those valves are adjusted between 0% and 100% to maintain the specified RCS pressure.

## 3. Development of a DRL-Based Controller

This section introduces the development of a DRL-based controller to create bubbles for the pressurizer and then controls the pressurizer pressure and level during the cold shutdown operation. The DRL-based controller comprises two DRL controllers and a rule-based controller, as shown in Figure 3. The rule-based controller performed the discrete controls listed in Table 4. As a result of the task analysis, if specific rules, that is, if–then logic, can be defined, the rule-based controller is applied, as shown in Table 4. Therefore, the back-up heaters and proportional heaters are controlled using a rule-based controller.

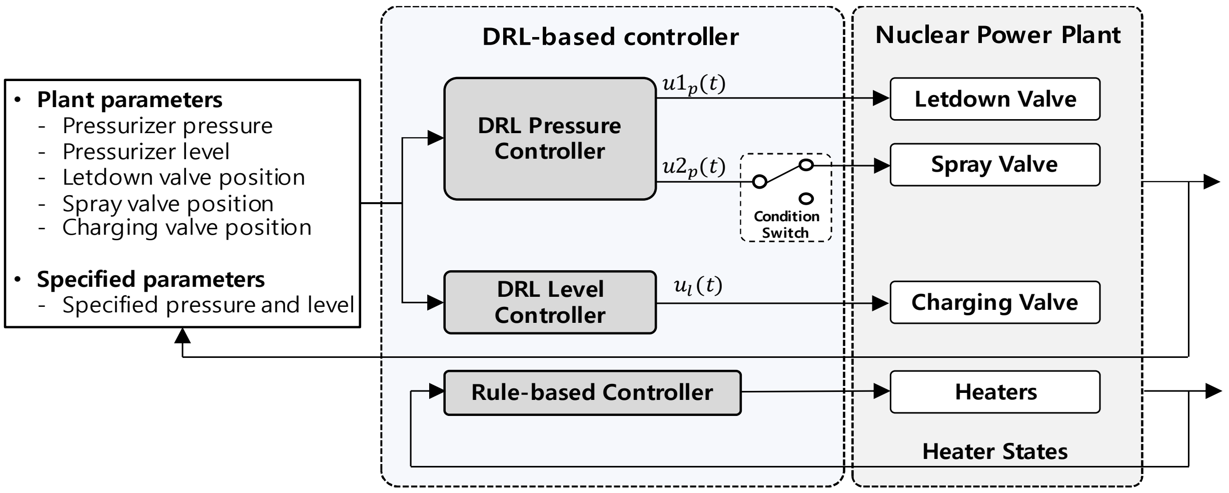

**Figure 3.** DRL-based controller block diagram.

**Table 4.** Controller tasks for cold shutdown operation.

| Task type | Controller | Action |
|---|---|---|
| Discrete control | Rule-based controller | If the back-up heater state is "Off", push "On" button. |
| | | If the proportional heater power is 0% or below 100%, increase the power to 100%. |
| Continuous control | Pressure DRL controller | Maintain the pressurizer pressure between 26 kg/cm$^2$ and 28 kg/cm$^2$ by adjusting letdown valve. |
| | | If the pressurizer temperature reaches the saturation point of about 200 °C, maintain the pressurizer pressure between 26 kg/cm$^2$ and 28 kg/cm$^2$ by adjusting the spray valve. |
| | Level DRL controller | Adjust the charging valve to maintain the pressurizer level at 50% |

DRL controllers perform continuous control, for which it is difficult to define specific rules, for example, how much a valve should be open to maintain the pressure. DRL controllers are divided into pressure and level controllers, as shown in Table 3. As shown in Table 4, the pressure DRL controller aims to maintain the pressure by adjusting the letdown and spray valve, whereas the level DRL controller adjusts the charging valve to maintain the pressure level at 50%. DRL controllers use an LSTM network trained using the SAC learning algorithm. The LSTM is known to show a good performance in handling time-related, dynamic data. In addition, the authors' previous studies also showed that the LSTM could support well the operation of nuclear systems and the diagnosis of events [36–39].

### 3.1. Soft Actor Critic with Distributed Prioritized Experience Replay

Reinforcement learning (RL) is a method for training an AI network through interaction with its environment [5,40]. Using a controller with RL provides the possibility of finding an optimal policy, which includes solving the given problem or achieving operational goals in the sequential decision-making of the current state collected from the environment. One of the challenges in RL is finding an optimized policy function to obtain the maximum reward for all given states. Determining an optimized policy function may take a long time. To resolve this issue, recent studies have suggested using a neural network as an optimized policy function of the RL owing to the increased computing power and an improved method called the deep neural network. Therefore, this study adopts an approach that uses deep reinforcement learning (DRL), which combines RL and deep neural network models, to find the optimal policy.

This study utilized SAC to improve the training stability of a DRL-based controller using an LSTM network model. The SAC was suggested to compensate for the deep Q-learning network (DQN), which is a basic model of the DRL. The drawback of the DQN is biased actions caused by predictions that rely on a single neural network. One network in the DQN predicts actions mixed with evaluations based on action probability and estimated reward. In contrast, SAC uses an actor-critic architecture with a separate value (Q-network) and policy network, as shown in Figure 4. Q-networks calculate the expected rewards for an action taken in the current state. Then, the policy network predicts each action probability based on the expected reward and the current state. For training stability, SAC uses two Q-networks consisting of an online network and a target network. The target network update is delayed when updating the online network parameters over many iterations [6]. While updating online network parameters over many iterations, the target network parameters perform delayed updates from the online network at regular intervals, which helps reduce biased training.

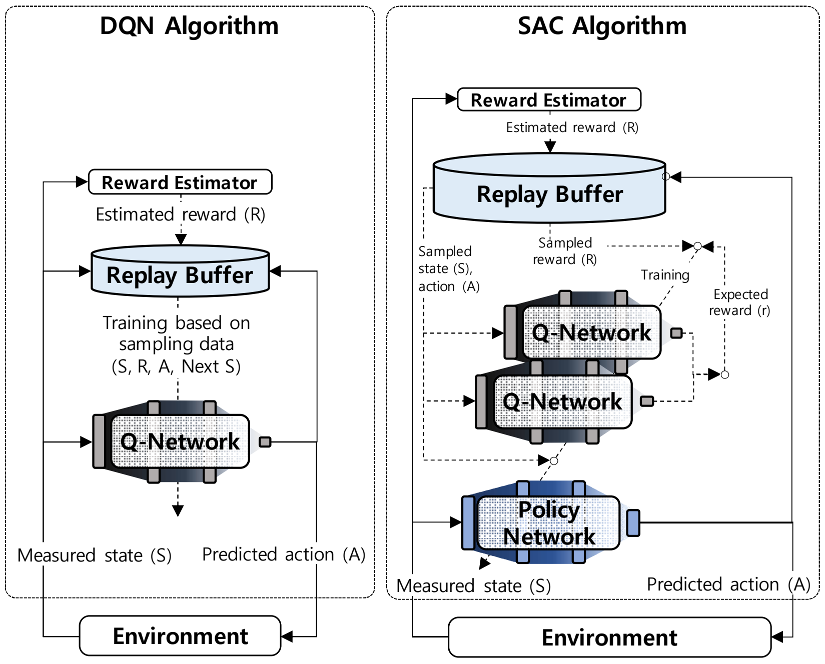

**Figure 4.** Training algorithm of DQN (left) and SAC (right).

Moreover, to reduce the training time, this study also adopted a distributed training architecture with distributed prioritized experience replay (DPER), a type of experience replay buffer, as shown in Figure 5. In this architecture, the main network is trained with data collected from multiple simulations using a local neural network. Each local network contains only a policy network that regularly distributes training from the main policy network.

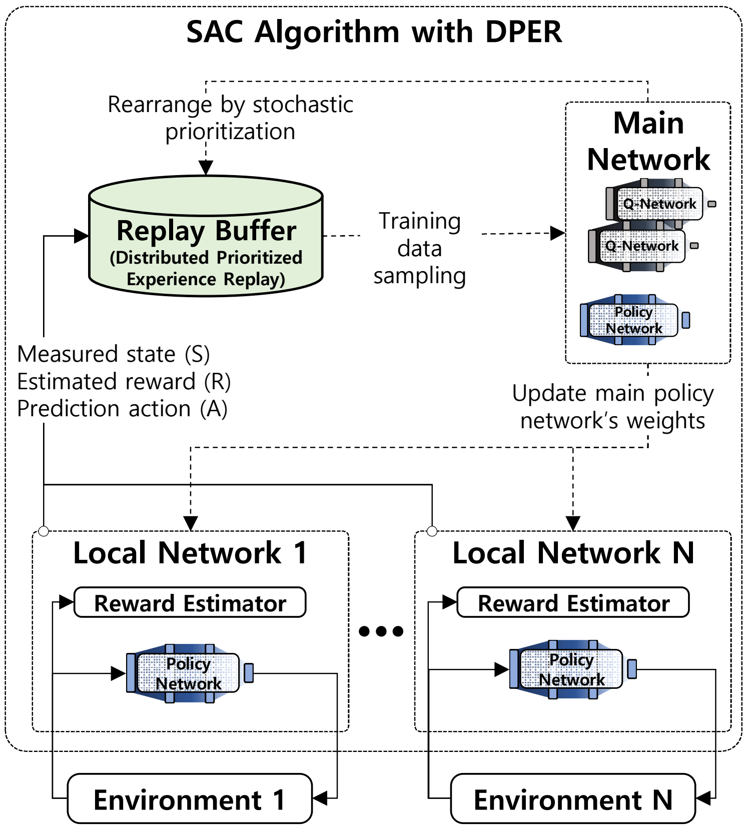

**Figure 5.** Training algorithm of SAC with DPER.

DPER was utilized to collect data simulated from local networks and improve the efficiency of the sampled data when the main network was trained [41]. DPER enables the DRL controller to remember and reuse experiences from the past, where observed transitions are stored for some time, usually in a queue, and sampled uniformly from this memory to update the network. For example, DQN training relies on randomly selected samples from the replay buffer. In contrast to the basic experience replay buffer, the DPER can sample data that are more frequently replayed transitions with high expected learning progress, as measured by the magnitude of their temporal difference (TD) error. TD error is the difference between the expected and actual rewards. Consequently, the main network learns by sampling data with higher stochastic priorities.

### *3.2. Design of the Reward Algorithm*

This section presents the reward algorithm for the DRL-based controllers. In DRLs, the reward is an essential element used to update the weights of the neural networks. A reward algorithm was used to evaluate the actions predicted by a network and provide guidance for updating the weights of the neural network [42,43]. DRL-based controllers obtain rewards by evaluating the actions performed within given states. Thus, the reward algorithm evaluates the performed action under the given state and creates training datasets that consist of pairs of states, actions, and rewards.

Two reward algorithms for the level and pressure controllers were suggested to reflect the operational constraints identified in Table 3. Reward algorithms aim to minimize the distance from the current state to the desired state, for example, the midpoint of the pressure boundary or the specified pressurizer level.

As the level controller does not operate until the saturation point is reached, the level-reward algorithm provides a reward when the level controller starts control. As shown in Figure 6, the reward value is defined as the difference between the current pressurizer level and desired level (50%), as shown in Equation (1). The pressure level in the pressurizer was varied between 0% and 100%. To provide a reward range between 0 and 1, the scaling value is defined as 50, which is the maximum distance from the desired pressurizer level to the limits of range (0% to 100%). For instance, the level reward is zero for the lowest reward when the current level is within the limits of the level range (Points A and B in Figure 6). As the current level increases from Point C to D, approaching the desired value, the level reward increases from 0.8 to 1. The level reward algorithm provides a maximum reward of one when the level controller is running successfully to maintain the current level at the desired pressure level:

$$\text{Level reward} = \left(1 - \frac{|\text{Current level} - \text{Specified pressurizer level}|}{\text{Scaling value}}\right) \tag{1}$$

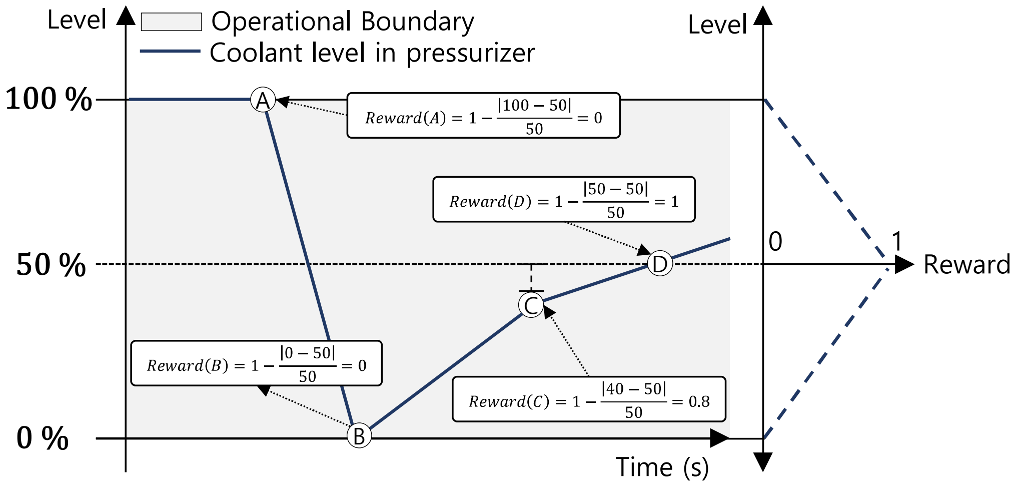
Level
▭ Operational Boundary
Coolant level in pressurizer
100 %
50 %
0 %
A
B
C
D
$Reward(A) = 1 - \frac{|100-50|}{50} = 0$
$Reward(D) = 1 - \frac{|50-50|}{50} = 1$
$Reward(B) = 1 - \frac{|0-50|}{50} = 0$
$Reward(C) = 1 - \frac{|40-50|}{50} = 0.8$
Time (s)
Level
0
1
Reward

**Figure 6.** Level reward algorithm for achieving operational goals.

The pressure-reward algorithm provides a reward for the pressure controller to maintain a pressure between 25 kg/cm$^2$ and 29 kg/cm$^2$. The reward value was calculated as the difference between the current RCS pressure and the desired condition, as shown in Equation (2). The scale value is defined as 2, which is half the desired range of pressure. Because the pressure changes slightly during the cold shutdown operation, the pressure reward uses the squared reward to consider pressure changes sensitively. For instance, in Figure 7, the reward value at Point C is the difference between the current RCS pressure and the lower desired pressure (25 kg/cm$^2$), which is 0.25 at the current pressure 26 kg/cm$^2$. In the case of point D, that is, at a current pressure of 27.5 kg/cm$^2$, the reward value is the difference between the current pressure and the upper desired pressure (29 kg/cm$^2$), which is 0.56. Therefore, the pressure reward increased as the pressure approached the middle of the pressure boundary, with a maximum of 1. When the current pressure exits the desired pressure boundary, e.g., Point E and F in Figure 7, the training is terminated.

$$\text{Pressure reward} = \left(1 - \frac{|\text{Current pressure} - \text{Middle of pressure boundary}|}{\text{Scaling value}}\right)^2 \tag{2}$$

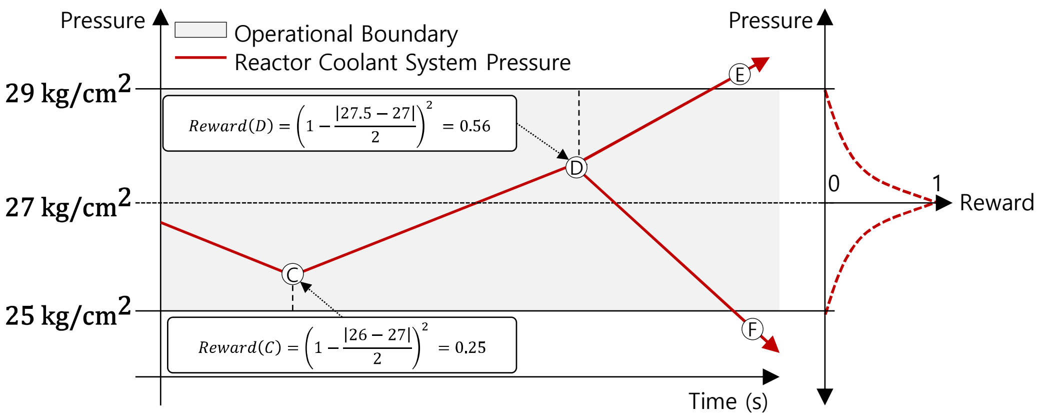
Pressure
Operational Boundary
Reactor Coolant System Pressure
29 kg/cm²
27 kg/cm²
25 kg/cm²
Reward(D) = (1 − |27.5 − 27|/2)² = 0.56
Reward(C) = (1 − |26 − 27|/2)² = 0.25
E
D
C
F
Pressure
0
1
Reward
Time (s)

**Figure 7.** Pressure reward algorithm for achieving operational goals.

In addition to this termination condition of pressure, another termination condition was defined during the training. If the operation time in the training reaches eight hours, the episode is terminated because the GOP instructs that the operation should be completed within 8 h.

## *3.3. Design of Long Short-Term Memory Network*

A neural network-based architecture, that is, a part of the DRL-based controller, was developed to perform continuous controls. To generate control actions from the DRL-based controller, this study used LSTM cells that can calculate time-series data [44]. LSTM cells were developed from recurrent neural networks (RNNs).

An RNN is a powerful network that can naturally represent dynamic systems and capture their behavior [45]. However, a problem that gradient values drastically vanish to zero may be observed when the network has many layers [46]. To complement this drawback, LSTM cells have been proposed.

The structure of a typical LSTM cell consists of cell state and three gates, i.e., cell state, input, output, and forget gates, as shown in Figure 8 [47]. The cell state retains the prior state information across time steps. The input, forget, and output gates determine which prior cell state will be saved, dumped, and sent out in the next cell state, respectively. The output of the input gate ($i^t$), forget gate ($f^t$), output gate ($o^t$), and cell state ($c^t$) are expressed as shown in Equations (3)–(6). Then, the hidden state ($h_t$) is updated with Equation (7) and sent to the next cell:

$$i_t = \sigma(x^t W_{xi} + h_{t-1} W_{hi} + b_i) \tag{3}$$

$$f_t = \sigma\left(x^t W_{xf} + h_{t-1} W_{hf} + b_f\right) \tag{4}$$

$$o_t = \sigma\left(x^t W_{xo} + h_{t-1} W_{ho} + b_o\right) \tag{5}$$

$$c_t = f_t \cdot c_{t-1} + i_t \cdot \tanh\left(x^t W_{xc} + h_{t-1} W_{hc} + b_c\right) \tag{6}$$

$$h_t = o_t \cdot \tanh(c_t) \tag{7}$$

where $W_{xi}$, $W_{xf}$, $W_{xo}$, and $W_{xc}$ denote weights between the input of the LSTM cell and units (input, forget, output, and cell state), while $W_{hi}$, $W_{hf}$, $W_{ho}$, and $W_{hc}$ denote weights between the prior hidden recurrent layer and units (input, forget, output, and cell state), $x^t$ is the input of the LSTM cell, and $b_I$, $b_f$, $b_o$, and $b_c$ represent the additive bias of units. This set of activation function include the sigmoid function, $\sigma$, and the hyperbolic activation function.

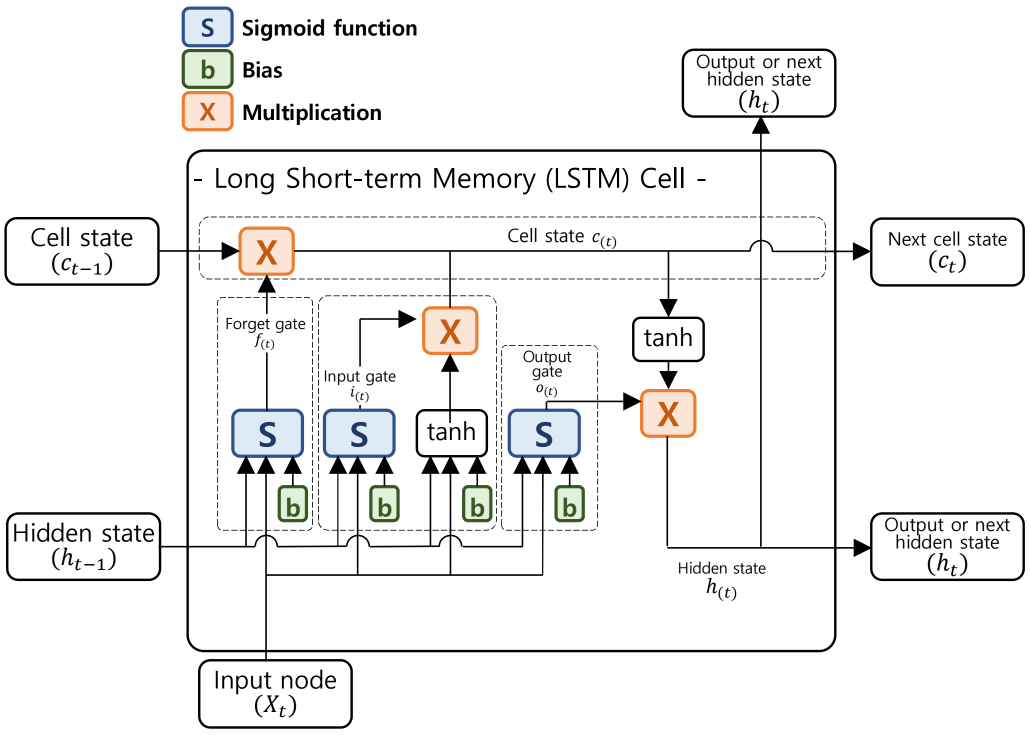

**Figure 8.** Structure of an LSTM cell.

LSTM cells allow the DRL controller to handle the NPP parameters and control the components with high performance because the NPP data exhibit the characteristics of non-linearity and time-series data. Figure 9 illustrates the structure of the LSTM network applied to the DRL controller policy. Value networks have the same structure as the policy network, except for the output layer that generates the expected reward. Generally, an LSTM network model consists of an input layer, an LSTM layer, and an output layer. The sizes of the input and output layers are defined according to the number of plant parameters. The number of LSTM cells is equal to the size of the time window.

The input layer of the LSTM network has a time window of 10 s, which considers the trend of the plant parameters by exploiting the collected historical data. Therefore, the DRL controller uses states that include the current and previous states as a two-dimensional array. Thus, the number of LSTM cells is equal to the size of the time window.

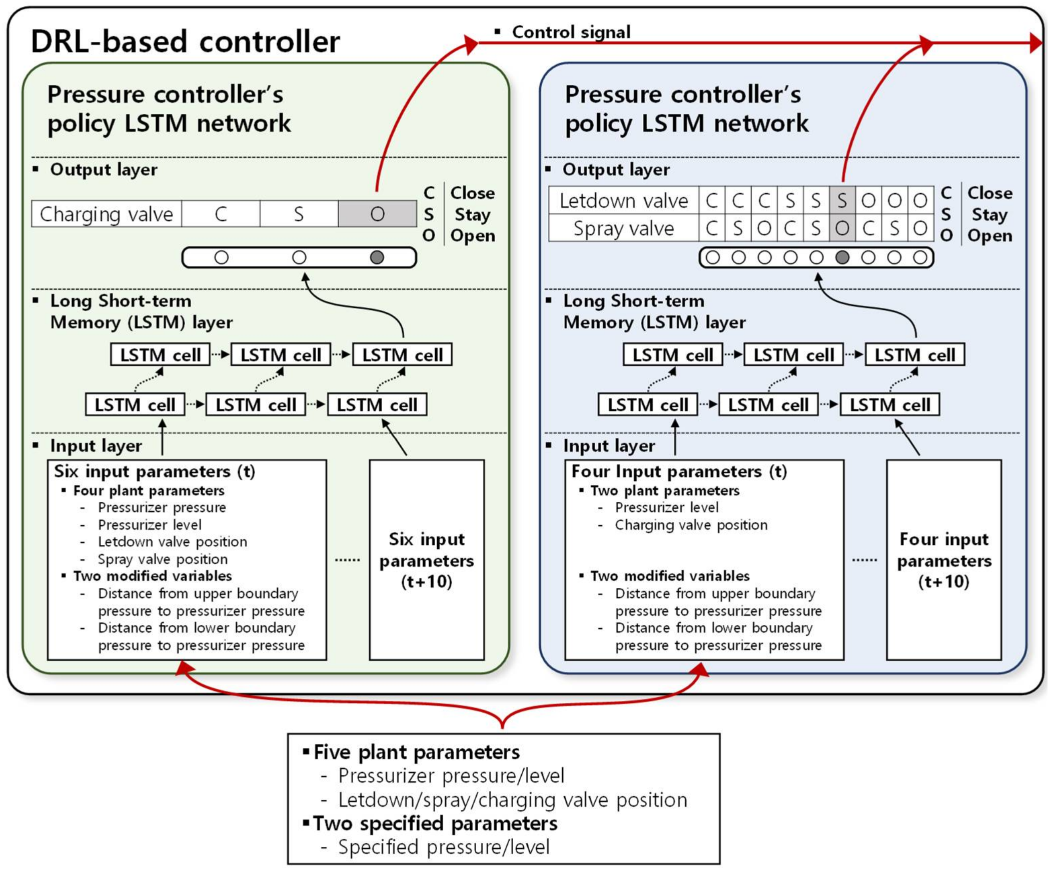
DRL-based controller
▪ Control signal
Pressure controller's policy LSTM network
▪ Output layer
Charging valve
C
S
O
C Close
S Stay
O Open
▪ Long Short-term Memory (LSTM) layer
LSTM cell
▪ Input layer
Six input parameters (t)
▪ Four plant parameters
- Pressurizer pressure
- Pressurizer level
- Letdown valve position
- Spray valve position
▪ Two modified variables
- Distance from upper boundary pressure to pressurizer pressure
- Distance from lower boundary pressure to pressurizer pressure
Six input parameters (t+10)
Pressure controller's policy LSTM network
Letdown valve
Spray valve
Four Input parameters (t)
▪ Two plant parameters
- Pressurizer level
- Charging valve position
▪ Two modified variables
- Distance from upper boundary pressure to pressurizer pressure
- Distance from lower boundary pressure to pressurizer pressure
Four input parameters (t+10)
▪ Five plant parameters
- Pressurizer pressure/level
- Letdown/spray/charging valve position
▪ Two specified parameters
- Specified pressure/level

**Figure 9.** Structure of LSTM network for the DRL controller policy network.

As shown in Figure 9, the proposed DRL controller includes two policy networks composed of LSTM networks to manage the pressurizer pressure and level. The DRL controller used five plant parameters and two specified pressures and levels. The plant parameters consisted of three component states (letdown/charging/spray valve positions) and two pressurizer states (pressure and level). The pressure policy network uses four plant parameters (pressurizer pressure, pressurizer level, letdown valve position, and spray valve position) and two modified variables that include the distance from the pressure boundary. The level-policy network uses two plant parameters (pressurizer level and charging valve position) and two distance values from the specified level.

The output layer consists of a set of actions for controlling the target components, such as the letdown, spray, and charging valves. The control strategies of one valve are threefold, that is, open, closed, or no control. If a control strategy selects "open valve", the valve position will increase. In the case of "no control", the valve maintains the current position. Therefore, the level policy network output is one of the three control strategies shown in Figure 9. However, because the policy network for pressurizer pressure aims to control the letdown and spray valves, the set of actions includes nine cases that combine the three control strategies of the two valves. To select a control strategy, the output size of the output layer should be equal to the number of control strategies. Therefore, in the case of the DRL pressure controller, nine control strategies for controlling the letdown and spray valves were mapped to the output valves for the output layer of the LSTM network.

To select one control strategy among the nine cases, the DRL-based controller for pressurizer pressure also calculates the expected reward acquisition probability for each action in the LSTM network. The softmax function was used to calculate the probability of each control strategy in the output layer. The softmax function can map the network output to a probability distribution between zero and one. Therefore, the sum of the values of the generated output is one. Therefore, the LSTM network can calculate the probability value for each control strategy.

### *3.4. Training of a DRL-Based Controller*

A compact nuclear simulator (CNS) was used as a real-time testbed to train and validate the developed DRL-based controller. The CNS was originally developed by the Korean Atomic Energy Research Institute (KAERI) with reference to a Westinghouse 930 MWe three-loop PWR [48]. Figure 10 shows the interface of the chemical and volume control system (CVCS) and reactor coolant system (RCS) in the CNS.

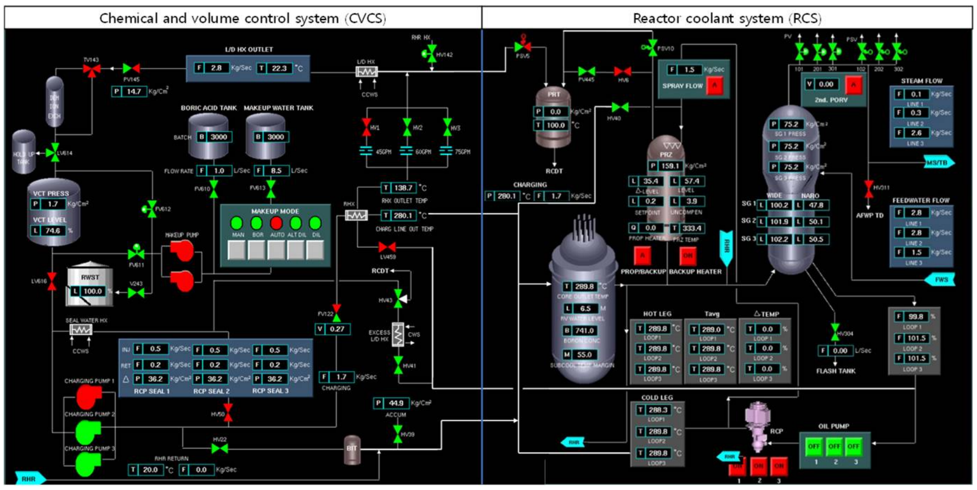
Chemical and volume control system (CVCS)
Reactor coolant system (RCS)

**Figure 10.** Chemical and volume control system (CVCS) and reactor coolant system (RCS) in the CNS.

Figure 11 shows the multi-CNS environment for training and validating the DRL-based controller. Two desktop computers were used to construct the multi-training environment. A DRL-based controller is installed on the main computer. CNSs were installed on a subcomputer with an Intel Core(TM) i7-8700K and 16 GB of memory. Twenty CNSs were simultaneously simulated. One of the local networks is connected to a CNS simulation through user datagram protocol (UDP) communication. The global network was trained on two Nvidia Geforce GTX1080Ti graphic cards, whereas the SAC training algorithm was trained using a 10 CPU core on Intel CoreX i7-7820X. The DRL-based controller was programmed using Python. PyTorch, which is a well-known Python machine-learning library, was used to develop a DRL-based controller.

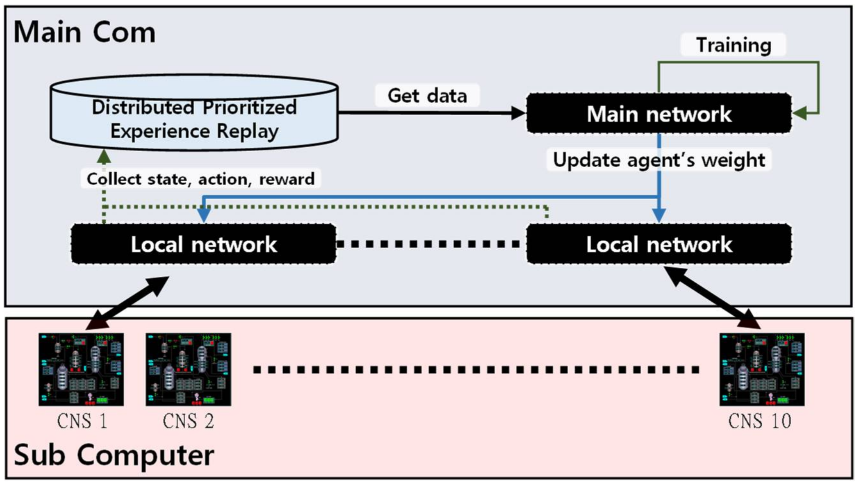

**Figure 11.** Multi-CNS environment for training and validating the developed DRL-based controller.

To achieve acceptable performance of the proposed DRL-based controller, it was trained until it reached a stable training state. DRL-based pressure and level controllers are trained through many episodes, each of which is completed if at least one controller reaches the termination condition. All controllers stop training when the average maximum probability converges to a certain value, or when the value stabilizes. The average maximum probability is the mean value of the probability of the actions selected by the DRL controller in one step. In one step, the DRL-based controller learns using 256 sample data from the DPER. In this study, the experimental results considering the entire operation time confirmed that operational goals could be reached when 256 data points were sampled. If there are more (512) or less (128) than this, learning fails. The average maximum probability refers to the degree to which the DRL controller completes the training. If the average maximum probability is higher than the previous step, it implies that the DRL controller selects actions that are more likely to succeed. Figure 12 shows the trend of the average maximum probability per step over time. Figure 13 shows the trend in the rewards per episode. The y-axis represents the total reward earned in each episode.

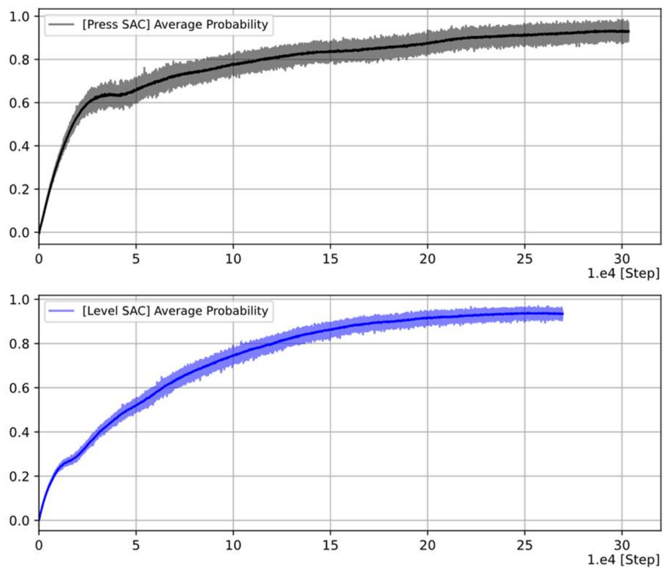

**Figure 12.** Average maximum probability per episode.

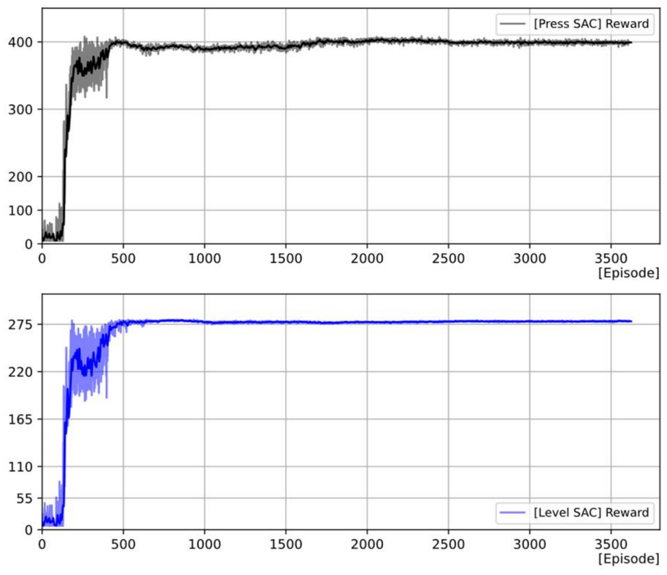

**Figure 13.** Reward per episode.

The pressure and level controllers reached a stable value after 2000 episodes. Approximately 84 h of training were required until the DRL-based controllers learned how to adjust the charging, letdown, and spray valves to achieve the operational goal. At approximately 500 episodes, the rewards reached 400 (pressure controller) and 275 (level controller).

## 4. Development of a PID-Based Controller

A PID-based controller was designed as shown in Figure 14. The PID-based controller should manage five components (charging, letdown, spray valves, and back-up and proportional heaters) to achieve two operational goals (pressurizer pressure and level). If-then logic was applied to control the two heaters. Therefore, three PID controllers are developed for the three valves.

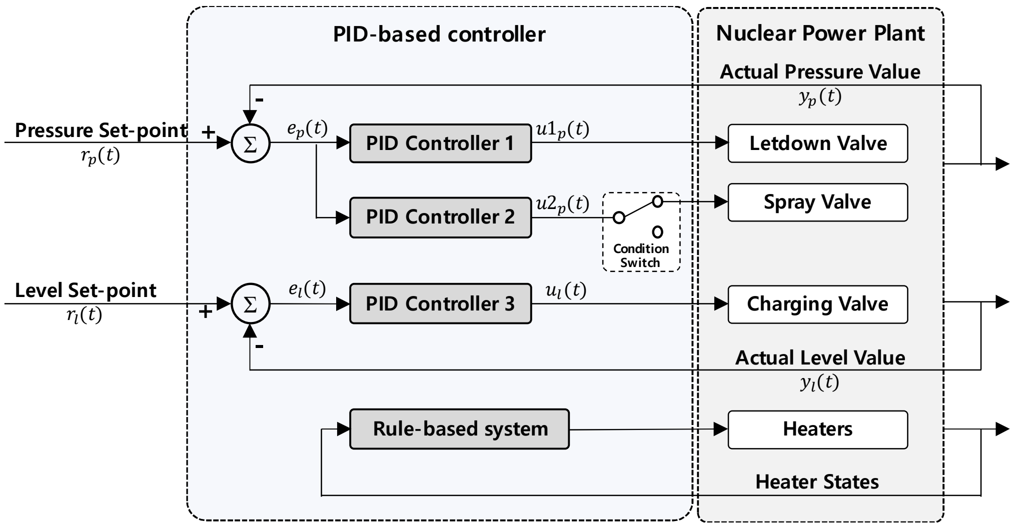

**Figure 14.** PID-based system block diagram.

In general, a PID controller is applied to a single-input, single-output system without considering the disturbance and nonlinearity of the system [49]. Thus, three PID controllers must be developed to adjust the charging, letdown, and spray valves. In Figure 14, PID Controllers 1 and 2 adjust the letdown and spray valves for pressurizer pressure, whereas PID Controller 3 manages the charging valve at the pressurizer level. Because the spray valve can be operated after pressurizer bubble creation, a condition switch was added to avoid unnecessary operations. The operational goal of PID controllers 1 and 2 was to regulate the pressurizer pressure within a specified pressure ($\mathrm{r_p(t)}$). The pressure deviation error ($\mathrm{e_p(t)}$) between the actual pressure value ($\mathrm{y_p(t)}$) and pressure set-point ($\mathrm{r_p(t)}$) is commonly used in PID controllers 1 and 2. PID controller 3 controls the charging valve to satisfy the pressurizer level.

### *4.1. Background of the PID Controller*

The PID controller is based on classical optimal control theory that uses a control loop feedback mechanism to control the process variables [50]. PID controllers are typically used in industrial control applications to regulate temperature, flow, pressure, speed, and other process variables. To increase plant performance and safety, a PID controller is also one of the most commonly used process controllers in NPPs [51].

As shown in Figure 15, the PID controller is designed to minimize the deviation error, $\mathrm{e(t)}$, between the set value, $\mathrm{r(t)}$, and the actual value $\mathrm{y(t)}$. With the deviation, the control output, $\mathrm{u(t)}$, is obtained by integrating the proportional, integral, and differential of errors. The proportional regulation is to increase the speed of control, the integral regulation is to minimize the steady-state error, and the differential regulation is to the stability of the system [52]. The regulation performance of classical PID controllers is described with the regulation time, overshoot, and system stability [53].

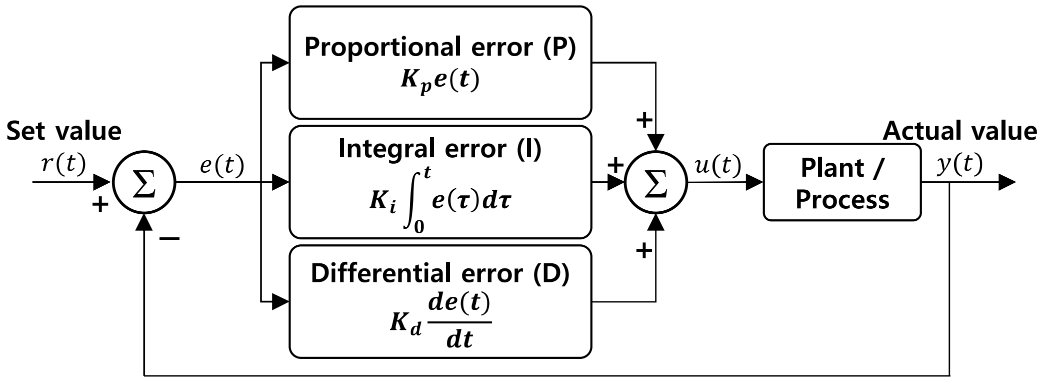

**Figure 15.** Block diagram of the PID controller principle.

## *4.2. PID-Based Controller Tuning Using the Ziegler–Nichols Rule and DRL Algorithm*

This study applies the Ziegler–Nichols closed-loop tuning method and DRL tuning method to achieve an acceptable performance of PID-based controllers. As a traditional tuning method, the Ziegler–Nichols method is well known as a suitable tool for nuclear power plants whose mathematical models are unknown or difficult to obtain [54]. Despite many design methods for PID controllers, the Ziegler–Nichols rule is one of the most widely used design methods in the literature [55,56]. In addition, the Ziegler–Nichols tuning method is used for automatic control in Korean NPPs [51].

According to the Ziegler–Nichols tuning method, a PID controller is tuned by first setting it to the P-only mode, which means that the integral gain ($K_i$) and derivative gain ($K_d$) are set to zero. The proportional gain ($K_p$) increases until the ultimate gain ($K_u$), where the system starts to oscillate, and an ultimate oscillation period ($T_u$), as shown in Figure 16. Then, $K_p$, $K_I$, and $K_d$ were approximated using Table 5 [10].

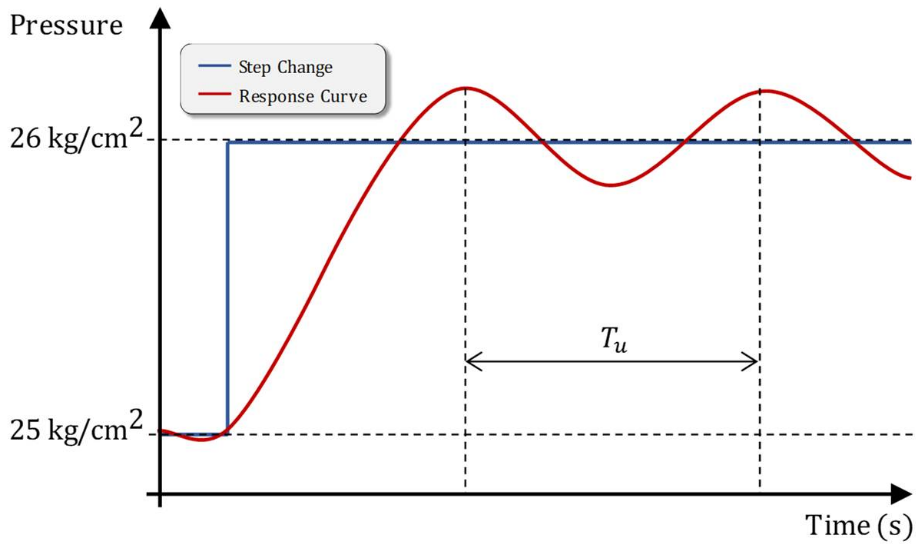

**Figure 16.** The change of pressure about step response.

**Table 5.** Ziegler–Nichols formula for PID controller tuning rules.

| Controller | $K_p$ | $K_i$ | $K_d$ |
|---|---|---|---|
| P | $0.50\ K_u$ [1] | 0 | 0 |
| PI | $0.45\ K_u$ [1] | $0.54\ K_u$ [1] $/T_u$ | 0 |
| PID | $0.60\ K_u$ [1] | $1.20\ K_u$ [1] $/T_u$ | $3\ K_u$ [1] $T_u/40$ |

[1] $K_u = K_p$.

As an alternative for the intelligent tuning methods, the DRL-based tuning was applied. This uses a DRL approach to obtaining the gains of controllers (Kp, Ki, and Kd). Figure 17 shows the process of the DRL-based tuning method. At the first step, the method initializes the gains as Kp = 0.1, Ki = 0, and Kd = 0. Then, in the second step, the policy network with simple DNN layers generates the gains, and the Q-network generates the expected reward by using initialized gains. The third step applies the gains to the PID controller and runs the CNS with the controller. In the fourth step, the cumulative rewards resulting from the simulation are calculated by using Equations (1) and (2). The fifth step calculates the loss value by the deviation between the cumulative rewards and the reward expected from the Q-network. Then, the policy and Q-network weights are updated by using the loss value in the sixth step. The seventh step evaluates whether the cumulative reward reaches a stable training state. If it is evaluated to be satisfactory, the generated gains are finally selected as the final gains of the controller.

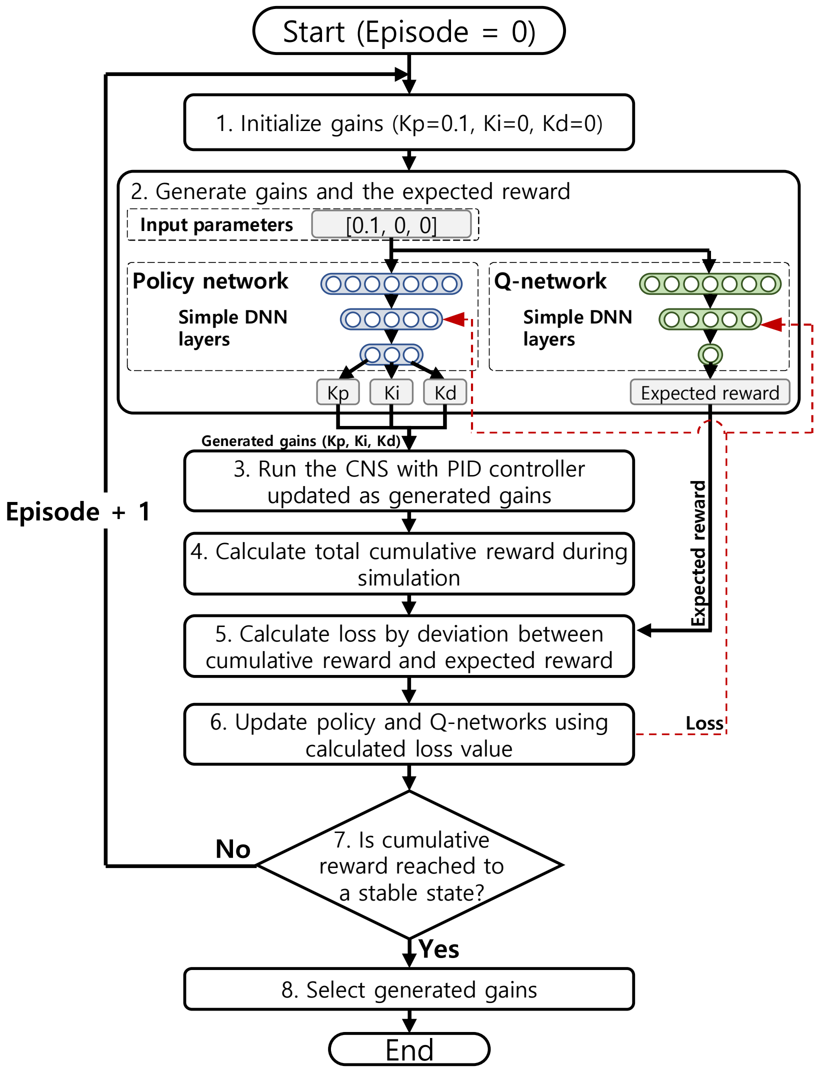
Start (Episode = 0)
1. Initialize gains (Kp=0.1, Ki=0, Kd=0)
2. Generate gains and the expected reward
Input parameters
[0.1, 0, 0]
Policy network
Simple DNN layers
Q-network
Simple DNN layers
Kp
Ki
Kd
Expected reward
Generated gains (Kp, Ki, Kd)
3. Run the CNS with PID controller updated as generated gains
4. Calculate total cumulative reward during simulation
5. Calculate loss by deviation between cumulative reward and expected reward
Expected reward
6. Update policy and Q-networks using calculated loss value
Loss
Episode + 1
No
7. Is cumulative reward reached to a stable state?
Yes
8. Select generated gains
End

**Figure 17.** Flowchart of the DRL tuning algorithm.

The PID controllers for the letdown, spray, and charging valves were tuned by using the DRL-based tuning algorithm. Figure 18 shows the history of cumulative reward per episode. The y-axis represents the total reward earned in each episode. Each PID controller is tuned until it reaches a stable training state.

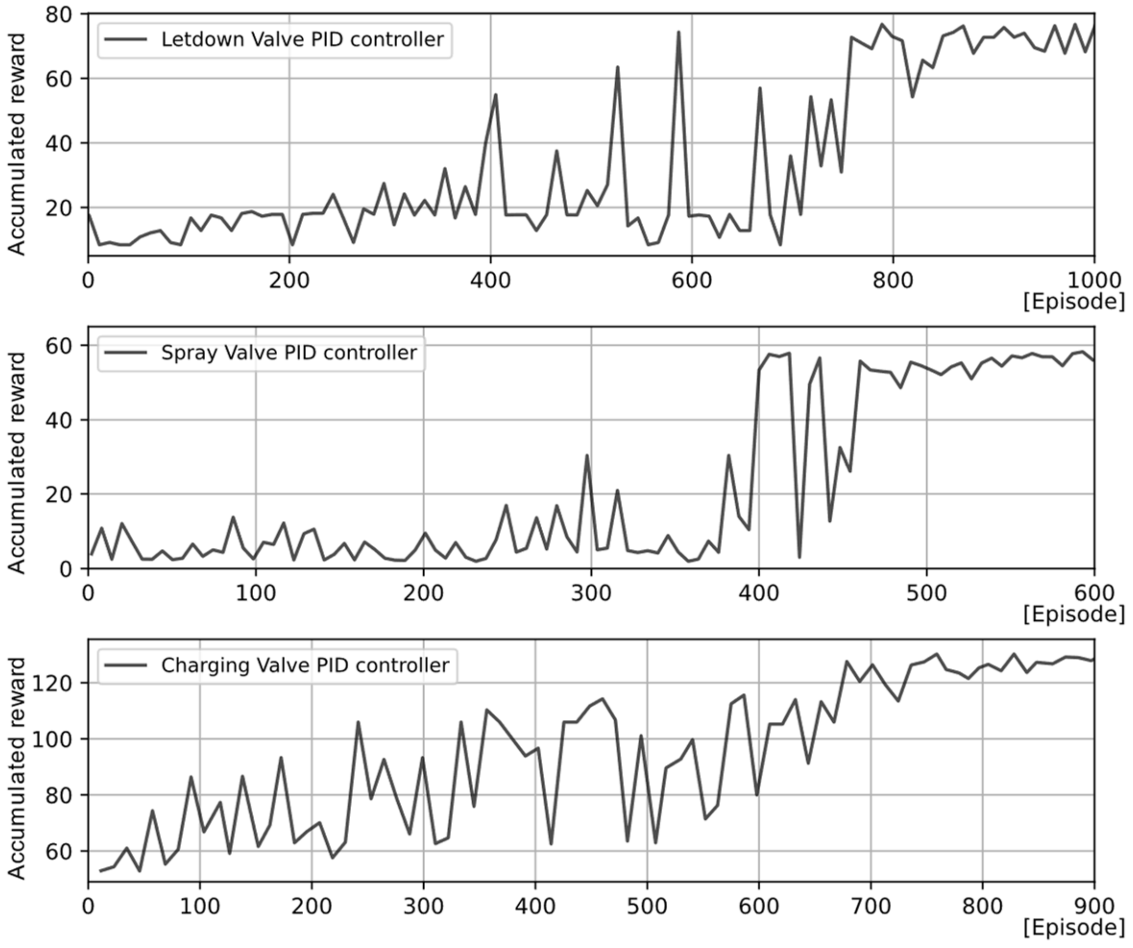

**Figure 18.** Reward per episode.

Table 6 shows the tuning results for the PID controllers by using the Ziegler–Nichols and DRD-based tuning method. Figure 19 also compares the performances of the different tuning results for the pressure and level of pressurizer. Because the pressure is managed by the letdown and spray valves, the letdown valve controller was tuned first and then the spray valve was tuned later. The comparison indicates that the DRL-based tuning shows better performances in time and accuracy than the Ziegler–Nichols method.

**Table 6.** Tuning results based on Ziegler–Nichols method and DRL tuning method.

| **Controller** | **Initial Parameter** | **Tuned Parameter (Ziegler–Nichols Method)** | **Tuned Parameter (DRL Method)** |
|---|---|---|---|
| Letdown valve | $K_p = 0.2$ $K_i = 0$ $K_d = 0$ | $T_u = 60$ s $K_p = 0.12$ $K_i = 0.004$ $K_d = 0.9$ | $K_p = 1.487$ $K_i = 1.155$ $K_d = 0.106$ |
| Spray valve | $K_p = 0.1$ $K_i = 0$ $K_d = 0$ | $T_u = 70$ s $K_p = 0.06$ $K_i = 0.001714$ $K_d = 0.525$ | $K_p = 1.657$ $K_i = 0.198$ $K_d = 0.078$ |
| Charging valve | $K_p = 0.1$ $K_i = 0$ $K_d = 0$ | $T_u = 300$ s $K_p = 0.06$ $K_i = 0.004$ $K_d = 2.25$ | $K_p = 0.833$ $K_i = 2.522$ $K_d = 0.105$ |

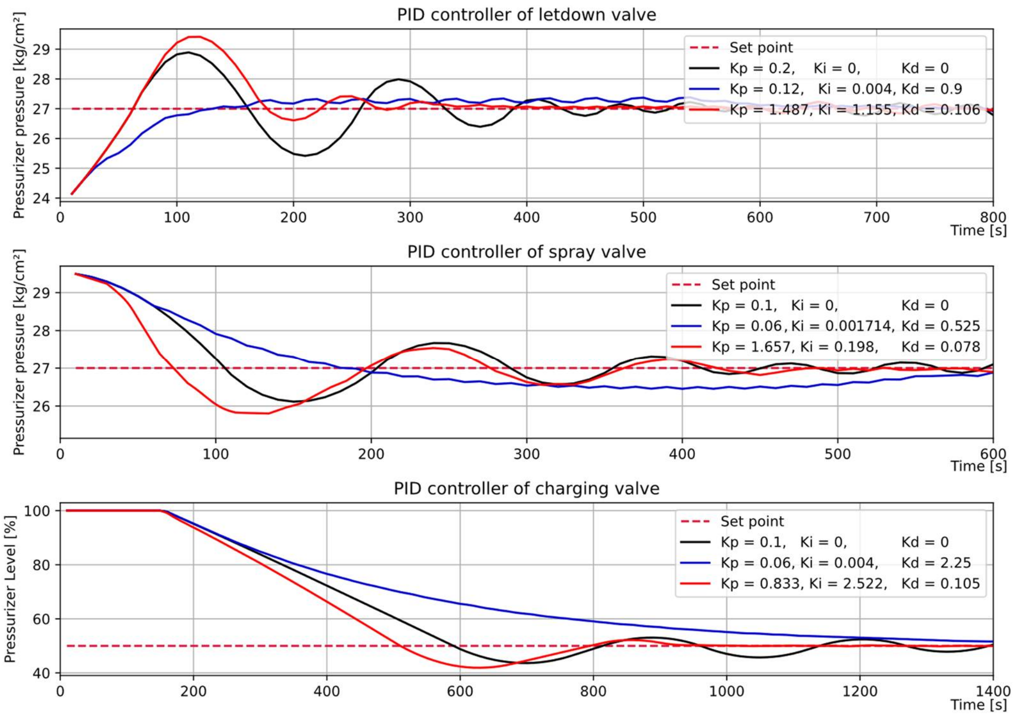

**Figure 19.** Comparison of performances for the different tuning methods.

## 5. Comparison of Performances of DRL-Based and PID-Based Controllers

A comparison of the performance of the developed DRL-based and PID-based controllers was conducted for the automation of the cold shutdown operation. The data were sampled from the simulator per second and the components could be manipulated every 10 s, which is considered enough time for the pressure and level of the pressurizer to change. The data sampling frequency was chosen by taking into account the computation time (0.5 s) of the simulator and the time transmitted to the controller (0.1 milliseconds).

Figure 20 shows the comparison of the performances in controlling the pressurizer pressure by the DRL-based and PID controllers. As shown in Figure 21, the DRL-tuned PID controller shows smaller accumulated error than the PID controllers tuned by the ZN- and DRL-based controller. The comparison for the pressurizer level also shows similar results, as shown in Figures 22 and 23. The DRL-tuned PID controller shows the smallest error in the level. For the time to reach the desired state, it appears that the DRL-tuned PID controller is faster than the other controllers.

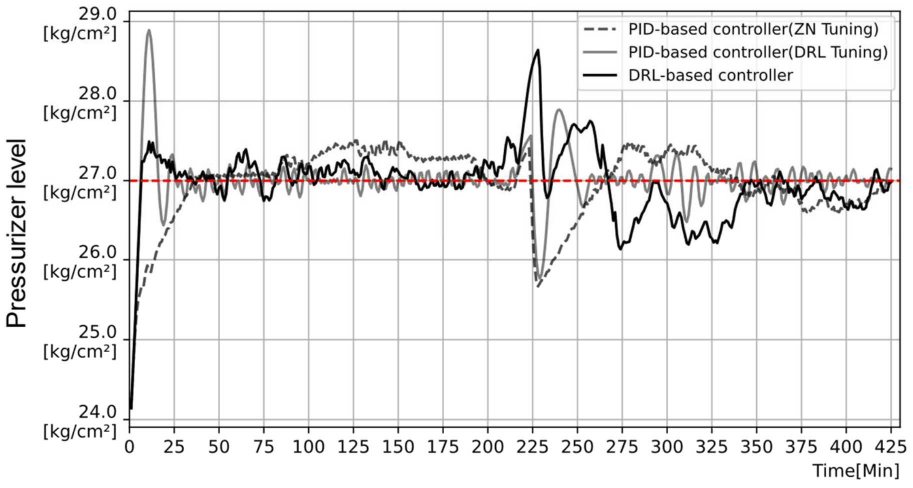

**Figure 20.** Comparisons of controllers for the pressurizer pressure.

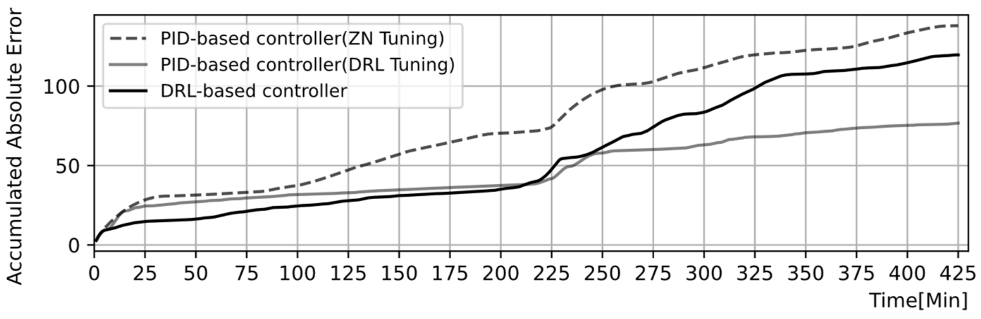

**Figure 21.** Accumulated error for the pressure with a reference of 27 kg/cm$^2$.

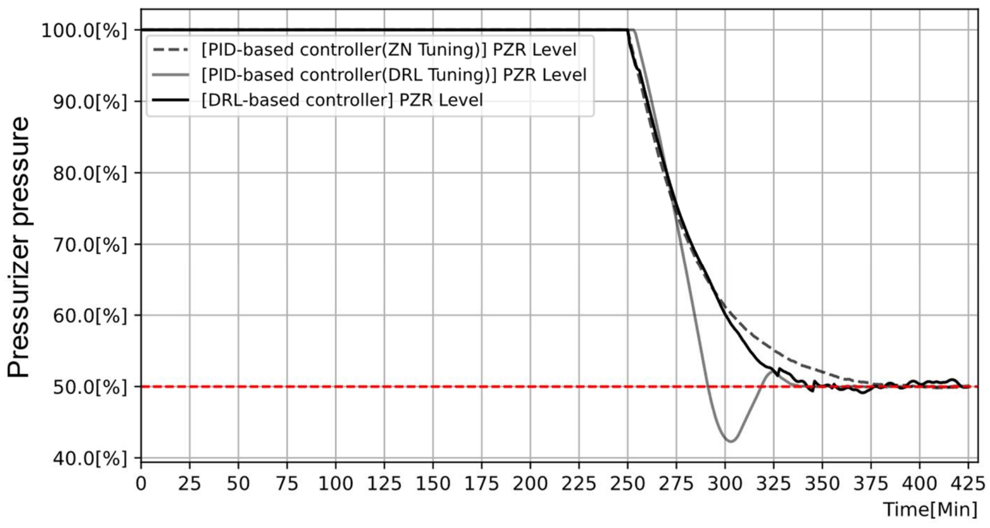

**Figure 22.** Comparisons of controllers for the pressurizer level.

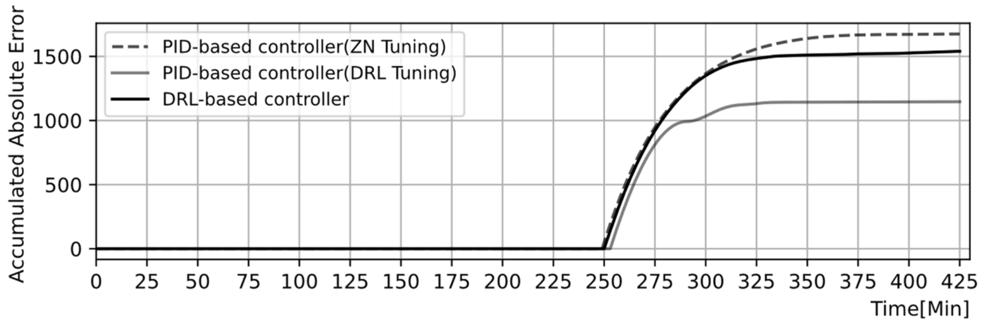

**Figure 23.** Accumulated error for the level with a set-point of 50%.

## 6. Discussion

This section discusses some interesting findings from this comparison study.

1. The DRL-tuned PID controller exhibited best performances in terms of error and time.

Table 7 compares the DRL- and PID-based controllers in terms of the average deviation error from the target value of the parameters and the time taken to reach the target value. For the pressurizer pressure and level, the DRL-tuned PID controller generally exhibited the smallest error and fastest reaching time than both the ZN-tuned PID controller and the DRL-based controller.

**Table 7.** Comparison result of operational performances.

| | Performance | PID-Based Controller | DRL-Based Controller |
|---|---|---|---|
| Pressurizer Pressure | Average deviation error from 27 kg/cm$^2$ | ±0.3248 kg/cm$^2$ (ZN)<br>±0.1805 kg/cm$^2$ (DRL) | ±0.2816 kg/cm$^2$ |
| | Reaching time to 27 kg/cm$^2$ | 32 min (ZN)<br>10 min (DRL) | 10 min |
| Pressurizer Level | Average deviation error from 50% | ±9.56% (ZN)<br>±6.55% (DRL) | ±8.79% |
| | Reaching time to 50% | +144 min (ZN)<br>+38 min (DRL) | +93 min |

2. Although PID-based controllers are dedicated to one component, DRL-based controllers manage the parameters and control multiple components simultaneously.

To control the pressurizer pressure to the desired value, three PID-based controllers were designed for three components: the charging valve, letdown valve, and spray. The controllers opened these values and the spray simultaneously to reduce pressure, as shown in Figure 24. On the other hand, two DRL-based controllers were developed for the control of pressure, not the control of components. Thus, the DRL-based controllers manipulate the three components in an interactive manner. For instance, as shown in Figure 24, the DRL-based controllers closed the letdown value at approximately 260 min and instead maintained the charging valve closed, while the PID-based controllers consistently opened these valves at the same time.

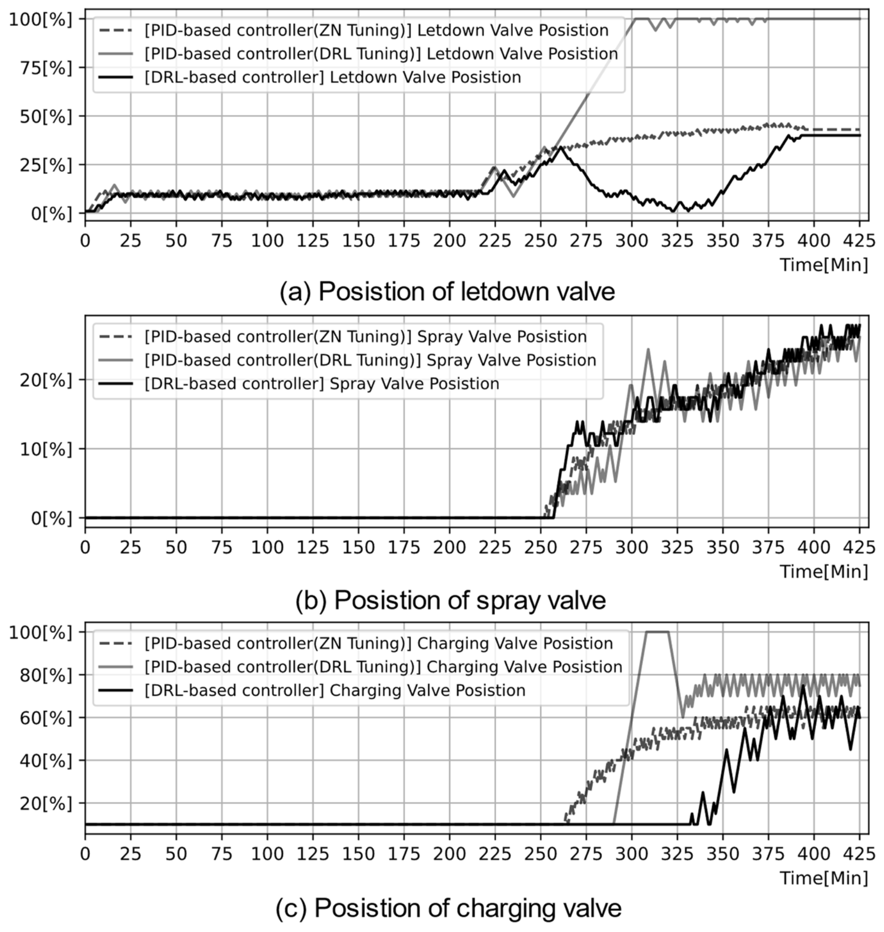

**Figure 24.** Positions of (**a**) letdown; (**b**) spray; (**c**) charging valve.

3. PID-based controllers manipulate components more frequently than DRL-based controllers.

Figure 25 shows a comparison of the number of manipulations for the three components. As shown in the figure, PID-based controllers control the components more frequently than DRL-based controllers. This may be related to the second finding described above. DRL-based controllers work interactively and can satisfy the operational goal with fewer manipulations.

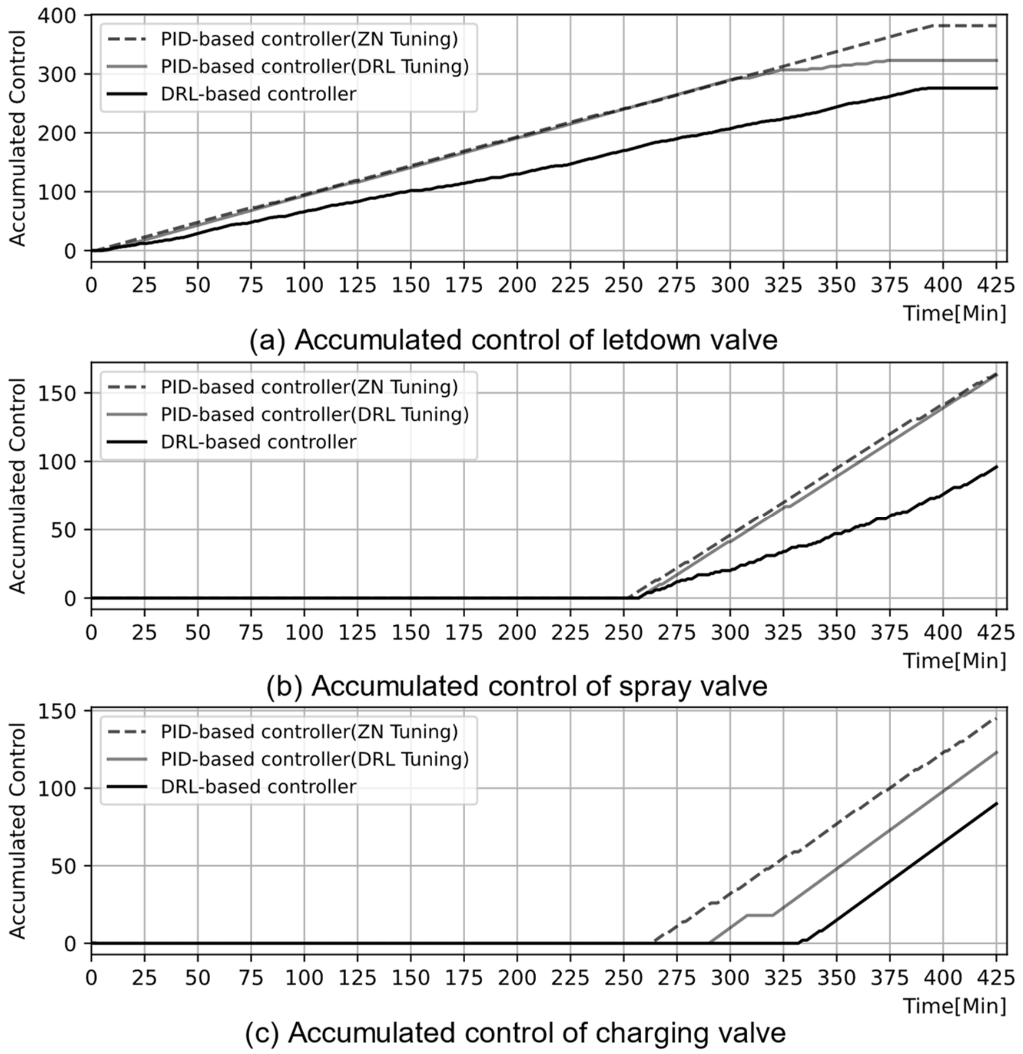

**Figure 25.** Total manipulation of (**a**) letdown; (**b**) spray; (**c**) charging valve during cold shutdown operation.

Less frequent manipulation is desirable in NPPs. First, frequent manipulation is likely to lead to component failures. From the perspective of probabilistic safety assessment, once a component starts to work (e.g., open/close or start/stop)., the probability of failing to work increases. Second, frequent manipulation accelerates the aging or fatigue of components. Thus, the replacement period is shortened because of aging.

4. Licensing is one of the unsolved issues for the DRL-based controllers.

The application of NPPs requires proven technologies. In particular, for safety-critical systems, controllers need to be approved by regulations. PID-based controllers have been acknowledged as a proven technology, because they have been used in NPPs for decades. However, it is common knowledge that AI technologies have not been sufficiently proven. Therefore, solving the licensing issue is the largest problem for applying DRL-based controllers to NPPs.

Even though the licensing issue is beyond the scope of this study, it is worth investigating some approaches to proving AI. The first is the use of an explainable AI called XAI. XAI can show how the AI produces the result and makes the AI closer to a whitebox. The second is the application of the software development process. The software used in the safety-critical system of NPPs should follow a very strict development process recommended by various standards, such as IEEE Standards 1012 [57] and 7-4.3.2 [58]. Because AI-based controllers can also be regarded as software, they are considered to apply the software development process to them.

## 7. Conclusions

This study compares the performance of DRL- and PID-based controllers in the cold shutdown operation of NPPs. This study conducted a task analysis for the bubble creation operation based on the operating procedures. Subsequently, PID- and DRL-based controllers were developed to satisfy the operational goal of the operation. The PID-based controllers were tuned by using the Ziegler–Nichols and DRD-based tuning method. This study compared the performances of the controllers. In general, the DRL-tuned PID controller exhibited the smallest error and fastest reaching time than both the ZN-tuned PID controller and the DRL-based controller. Finally, we presented some interesting findings.

**Author Contributions:** Conceptualization, D.L. and J.K.; Methodology, D.L.; Validation, D.L.; Writing—original draft preparation, D.L. and J.K.; Writing—review and editing, D.L., S.K., I.J. and J.K. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by a National Research Foundation of Korea (NRF) grant funded by the Korean government(Ministry of Science, ICT & Future Planning) (NRF-2016R1A5A1013919) and Korea Institute of Energy Technology Evaluation and Planning (KETEP) and the Ministry of Trade, Industry, & Energy (MOTIE) of the Republic of Korea (No. 20211510100020).

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The data are not publicly available.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Wood, R.T.; Neal, J.S.; Brittain, C.R.; Mullens, J.A. Autonomous control capabilities for space reactor power systems. In *AIP Conference Proceedings*; American Institute of Physics: College Park, MD, USA, 2004; pp. 631–638.
2. Lu, C.; Lyu, J.; Zhang, L.; Gong, A.; Fan, Y.; Yan, J.; Li, X. Nuclear power plants with artificial intelligence in industry 4.0 era: Top-level design and current applications—A systemic review. *IEEE Access* **2020**, *8*, 194315–194332. [CrossRef]
3. Choi, S.S.; Park, J.K.; Hong, J.H.; Kim, H.G.; Chang, S.H.; Kang, K.S. Development strategies of an intelligent human-machine interface for next generation nuclear power plants. *IEEE Trans. Nucl. Sci.* **1996**, *43*, 2096–2114. [CrossRef]
4. Lee, D.; Kim, J. Autonomous algorithm for start-up operation of nuclear power plants by using LSTM. In Proceedings of the International Conference on Applied Human Factors and Ergonomics, Orlando, FL, USA, 21–25 July 2018; Springer: Orlando, FL, USA, 2018; pp. 465–475.
5. Li, Y. Deep reinforcement learning: An overview. *arXiv* **2017**, arXiv:1701.07274.
6. Haarnoja, T.; Zhou, A.; Hartikainen, K.; Tucker, G.; Ha, S.; Tan, J.; Kumar, V.; Zhu, H.; Gupta, A.; Abbeel, P. Soft actor-critic algorithms and applications. *arXiv* **2018**, arXiv:1812.05905.
7. Viitala, A.; Boney, R.; Kannala, J. Learning to Drive Small Scale Cars from Scratch. *arXiv* **2020**, arXiv:2008.00715.
8. Bhalla, S.; Subramanian, S.G.; Crowley, M. Deep multi agent reinforcement learning for autonomous driving. In *Canadian Conference on Artificial Intelligence*; Springer: Cham, Germany, 2020; pp. 67–78.
9. Yu, C.; Wang, X.; Xu, X.; Zhang, M.; Ge, H.; Ren, J.; Sun, L.; Chen, B.; Tan, G. Distributed multiagent coordinated learning for autonomous driving in highways based on dynamic coordination graphs. *IEEE Trans. Intell. Transp. Syst.* **2019**, *21*, 735–748. [CrossRef]
10. Ziegler, J.G.; Nichols, N.B. Optimum settings for automatic controllers. *Trans. ASME* **1942**, *64*, 759–765. [CrossRef]
11. Cohen, G. Theoretical consideration of retarded control. *Trans. ASME* **1953**, *75*, 827–834.

12. Åström, K.J.; Hägglund, T. Automatic tuning of simple regulators with specifications on phase and amplitude margins. *Automatica* **1984**, *20*, 645–651. [CrossRef]
13. Malwatkar, G.; Sonawane, S.; Waghmare, L. Tuning PID controllers for higher-order oscillatory systems with improved performance. *ISA Trans.* **2009**, *48*, 347–353. [CrossRef]
14. Izci, D.; Ekinci, S.; Demirören, A.; Hedley, J. HHO algorithm based PID controller design for aircraft pitch angle control system. In Proceedings of the 2020 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA), Ankara, Turkey, 26–28 June 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–6.
15. Eker, E.; Kayri, M.; Ekinci, S.; Izci, D. A new fusion of ASO with SA algorithm and its applications to MLP training and DC motor speed control. *Arab. J. Sci. Eng.* **2021**, *46*, 3889–3911. [CrossRef]
16. Solihin, M.I.; Tack, L.F.; Kean, M.L. Tuning of PID controller using particle swarm optimization (PSO). In Proceedings of the International Conference on Advanced Science, Engineering and Information Technology, Kuala Lumpur, Malaysia, 14–15 January 2011; pp. 458–461.
17. Carlucho, I.; De Paula, M.; Acosta, G.G. An adaptive deep reinforcement learning approach for MIMO PID control of mobile robots. *ISA Trans.* **2020**, *102*, 280–294. [CrossRef] [PubMed]
18. Wei, T.; Wang, Y.; Zhu, Q. Deep reinforcement learning for building HVAC control. In Proceedings of the 54th Annual Design Automation Conference, Austin, TX, USA, 18–22 June 2017; pp. 1–6.
19. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.
20. Schmidhuber, J. Deep learning in neural networks: An overview. *Neural Netw.* **2015**, *61*, 85–117. [CrossRef] [PubMed]
21. Vecerik, M.; Sushkov, O.; Barker, D.; Rothörl, T.; Hester, T.; Scholz, J. A practical approach to insertion with variable socket position using deep reinforcement learning. In Proceedings of the International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 754–760.
22. Palanisamy, P. Multi-agent connected autonomous driving using deep reinforcement learning. In Proceedings of the International Joint Conference on Neural Networks (IJCNN), Glasgow, UK, 19–24 July 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–7.
23. Kiran, B.R.; Sobh, I.; Talpaert, V.; Mannion, P.; Al Sallab, A.A.; Yogamani, S.; Pérez, P. Deep reinforcement learning for autonomous driving: A survey. In *IEEE Transactions on Intelligent Transportation Systems*; IEEE: Piscataway, NJ, USA, 2021; pp. 1–18.
24. Kazmi, H.; Mehmood, F.; Lodeweyckx, S.; Driesen, J. Gigawatt-hour scale savings on a budget of zero: Deep reinforcement learning based optimal control of hot water systems. *Energy* **2018**, *144*, 159–168. [CrossRef]
25. Kang, C.; Huang, J.; Zhang, Z.; Liu, Q.; Xiang, W.; Zhao, Z.; Liu, X.; Chong, L. An automatic algorithm of identifying vulnerable spots of internet data center power systems based on reinforcement learning. *Int. J. Electr. Power Energy Syst.* **2020**, *121*, 106145. [CrossRef]
26. Rocchetta, R.; Bellani, L.; Compare, M.; Zio, E.; Patelli, E. A reinforcement learning framework for optimal operation and maintenance of power grids. *Appl. Energy* **2019**, *241*, 291–301. [CrossRef]
27. Du, G.; Zou, Y.; Zhang, X.; Liu, T.; Wu, J.; He, D. Deep reinforcement learning based energy management for a hybrid electric vehicle. *Energy* **2020**, *201*, 117591. [CrossRef]
28. Zhou, S.; Hu, Z.; Gu, W.; Jiang, M.; Chen, M.; Hong, Q.; Booth, C. Combined heat and power system intelligent economic dispatch: A deep reinforcement learning approach. *Int. J. Electr. Power Energy Syst.* **2020**, *120*, 106016. [CrossRef]
29. Yang, Z.; Zhu, F.; Lin, F. Deep-Reinforcement-Learning-Based Energy Management Strategy for Supercapacitor Energy Storage Systems in Urban Rail Transit. In *IEEE Transactions on Intelligent Transportation Systems*; IEEE: Piscataway, NJ, USA, 2020; pp. 1150–1160.
30. Saenz-Aguirre, A.; Zulueta, E.; Fernandez-Gamiz, U.; Lozano, J.; Lopez-Guede, J.M. Artificial neural network based reinforcement learning for wind turbine yaw control. *Energies* **2019**, *12*, 436. [CrossRef]
31. Genders, W.; Razavi, S. Policy Analysis of Adaptive Traffic Signal Control Using Reinforcement Learning. *J. Comput. Civ. Eng.* **2020**, *34*, 04019046. [CrossRef]
32. Dong, Z.; Huang, X.; Dong, Y.; Zhang, Z. Multilayer perception based reinforcement learning supervisory control of energy systems with application to a nuclear steam supply system. *Appl. Energy* **2020**, *259*, 114193. [CrossRef]
33. Park, J.; Kim, T.; Seong, S. Providing support to operators for monitoring safety functions using reinforcement learning. *Prog. Nucl. Energy* **2020**, *118*, 103123. [CrossRef]
34. Belles, R.; Muhlheim, M.D. *Licensing Challenges Associated with Autonomous Control*; Oak Ridge National Lab. (ORNL): Oak Ridge, TN, USA, 2018.
35. Kim, J.-T.; Kwon, K.-C.; Hwang, I.-K.; Lee, D.-Y.; Park, W.-M.; Kim, J.-S.; Lee, S.-J. Development of advanced I&C in nuclear power plants: ADIOS and ASICS. *Nucl. Eng. Des.* **2001**, *207*, 105–119.
36. Lee, D.; Arigi, A.M.; Kim, J. Algorithm for Autonomous Power-Increase Operation Using Deep Reinforcement Learning and a Rule-Based System. *IEEE Access* **2020**, *8*, 196727–196746. [CrossRef]
37. Kim, H.; Arigi, A.M.; Kim, J. Development of a diagnostic algorithm for abnormal situations using long short-term memory and variational autoencoder. *Ann. Nucl. Energy* **2021**, *153*, 108077. [CrossRef]
38. Yang, J.; Kim, J. Accident diagnosis algorithm with untrained accident identification during power-increasing operation. *Reliab. Eng. Syst. Saf.* **2020**, *202*, 107032. [CrossRef]
39. Lee, D.; Seong, P.H.; Kim, J. Autonomous operation algorithm for safety systems of nuclear power plants by using long-short term memory and function-based hierarchical framework. *Ann. Nucl. Energy* **2018**, *119*, 287–299. [CrossRef]

40. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*; MIT Press: Cambridge, MA, USA, 2018.
41. Horgan, D.; Quan, J.; Budden, D.; Barth-Maron, G.; Hessel, M.; Van Hasselt, H.; Silver, D. Distributed prioritized experience replay. *arXiv* **2018**, arXiv:1803.00933.
42. Guo, X.; Singh, S.; Lewis, R.; Lee, H. Deep learning for reward design to improve monte carlo tree search in atari games. *arXiv* **2016**, arXiv:1604.07095.
43. Aggarwal, M.; Arora, A.; Sodhani, S.; Krishnamurthy, B. Improving search through a3c reinforcement learning based conversational agent. In *International Conference on Computational Science*; Springer: Berlin, Germany, 2018; pp. 273–286.
44. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
45. Şeker, S.; Ayaz, E.; Türkcan, E. Elman's recurrent neural network applications to condition monitoring in nuclear power plant and rotating machinery. *Eng. Appl. Artif. Intell.* **2003**, *16*, 647–656. [CrossRef]
46. Gers, F.A.; Schmidhuber, J.; Cummins, F. Learning to forget: Continual prediction with LSTM. *Neural Comput.* **2000**, *12*, 2451–2471. [CrossRef] [PubMed]
47. Trinh, T.T.; Yoshihashi, R.; Kawakami, R.; Iida, M.; Naemura, T. Bird detection near wind turbines from high-resolution video using lstm networks. In Proceedings of the World Wind Energy Conference (WWEC), Tokyo, Japan, 30 October–1 November 2016; p. 6.
48. KAERI. *Advanced Compact Nuclear Simulator Textbook*; Nuclear Training Center in Korea Atomic Energy Research: Daejeon, Korea, 1990.
49. Pongfai, J.; Angeli, C.; Shi, P.; Su, X.; Assawinchaichote, W. Optimal PID controller autotuning design for MIMO nonlinear systems based on the adaptive SLP algorithm. *Int. J. Control. Autom. Syst.* **2021**, *19*, 392–403. [CrossRef]
50. Willis, M. *Proportional-Integral-Derivative Control*; Dept. of Chemical and Process Engineering University of Newcastle: Newcastle, Australia, 1999.
51. Sung, C.H.; Min, M.G. A Method of Tuning Optimization for PID Controller in Nuclear Power Plants. *Trans. Korean Soc. Press. Vessel. Pip.* **2014**, *10*, 1–6.
52. Zeng, W.; Jiang, Q.; Xie, J.; Yu, T. A fuzzy-PID composite controller for core power control of liquid molten salt reactor. *Ann. Nucl. Energy* **2020**, *139*, 107234. [CrossRef]
53. Zeng, W.; Jiang, Q.; Xie, J.; Yu, T. A functional variable universe fuzzy PID controller for load following operation of PWR with the multiple model. *Ann. Nucl. Energy* **2020**, *140*, 107174. [CrossRef]
54. Korkmaz, M.; Aydoğdu, Ö.; Doğan, H. Design and performance comparison of variable parameter nonlinear PID controller and genetic algorithm based PID controller. In Proceedings of the 2012 International Symposium on Innovations in Intelligent Systems and Applications, Trabzon, Turkey, 2–4 July 2012; IEEE: Piscataway, NJ, USA, 2012; pp. 1–5.
55. Mutlag, A.H.; Salim, O.N.M.; Mahdi, S.Q. Optimum PID controller for airplane wing tires based on gravitational search algorithm. *Bull. Electr. Eng. Inform.* **2021**, *10*, 1905–1913. [CrossRef]
56. Ogata, K. *Modern Control Engineering*; Prentice Hall: Hoboken, NJ, USA, 2010.
57. *IEEE Std 1012-2012*; IEEE Standard for System, Software, and Hardware Verification and Validation. IEEE Computer Society: Washington, DC, USA, 2017; pp. 1–206.
58. *IEEE Std 577-2012*; IEEE Standard Criteria for Programmable Digital Devices in Safety Systems of Nuclear Power Generating Stations. IEEE Power and Energy Society: Washington, DC, USA, 2016; pp. 1–86.