Nuclear Engineering and Technology 57 (2025) 103247

Contents lists available at ScienceDirect

Nuclear Engineering and Technology

journal homepage: www.elsevier.com/locate/net

Original Article

Small modular reactor reinforcement learning framework: Automating reactor core startup

Seong Jun Bae [a], Hong Hyun Son [a], Yongjae Lee [a], Jongin Yang [b,*]

[a] Korea Atomic Energy Research Institute, Yuseong-gu, Daejeon, 34057, Republic of Korea
[b] Kumoh National Institute of Technology, 61, Daehak-ro, Gumi-si, Gyeongsangbuk-do, 39177, Republic of Korea

ARTICLE INFO

Keywords:
Reinforcement learning
SMR simulation
Reactor core startup

ABSTRACT

A small modular reactor (SMR) has been considered a potential alternative for achieving carbon neutrality, and therefore, an increasing number of countries are performing extensive research and development. However, this is still in the development stage, and there are several technological or economical challenges that need to be overcome. Minimizing manual operations may be considered a wise approach to reduce the number of operators. Reactor core startup, which is a manual operation, is considered as an example. A method to automate the reactor core startup via the reinforcement learning (RL) algorithm is proposed in this paper. Further, an efficient SMR dynamic simulation model that performs simulations considering the action of the RL agent to achieve states and reward is developed. The suggested SMR dynamic simulation model is validated by the data available in the existing literature. The proposed method can perform automatic reactor core startup. The proposed framework that incorporates the SMR simulator to the RL algorithm is expected to be applied to various cases for reducing manual operations and contributing to realizing a higher level of SMR automation.

1. Introduction

The Net Zero Emissions by 2050 Scenario (Bouckaert et al., 2021) [1] aims to significantly reduce carbon emissions to limit the increase in the average global temperature to 1.5 °C. The global consensus on this goal has led to policy proposals within the framework of carbon emission regulations, which are expected to necessitate a significant reorganization of the current energy sectors and the development of advanced carbon-free technologies that will limit or eliminate carbon emissions.

Nuclear power has been proposed as a potential energy source to realize this goal. Similarly, the development of small modular reactors (SMRs) that can generate carbon-free energy and supply it to various industries including electricity (Bouckaert et al., 2021) [1] and shipping (Hirdaris et al., 2014) [2] has received significant research attention. SMRs have the potential to replace coal- and natural-gas-based power plants in the electricity generation sector and play a role in supplying distributed power across off-grid areas electrically isolated from the grid. Nuclear-propulsion SMRs have been adopted in the shipping sector instead of the conventional diesel-powered ships; they are small and have a lower power output (<100 MWe) than those used to generate electricity on the ground. Highly constrained maritime environments require complicated strategies to ensure flexible and safe operations.

These SMRs commonly seek operational flexibility and reduce the number of operators. To meet the general requirements, existing automated and manual controllers need to be more advanced in terms of decreasing or eliminating operator interventions. Most pressurized water-cooled reactor (PWR)-based NPPs and SMRs have standard operation modes such as core startup, plant heat-up, plant startup, power operation, and shutdown operation. Besides power operation, which is entirely automated, the other operations are partially or entirely manual. Reinforcement learning (RL), which is a machine learning (ML) method, can be used to replace manual controls with automated controllers, contributing to optimizing control actions and reducing human error by minimizing operator intervention.

DeepMind Technologies (Mnih et al., 2013) [3] first reported the remarkable performance of their own deep learning model using an RL method for playing Atari 2600 games, whose ability surpassed that of human experts in some cases. They showed that in an arcade learning environment that includes complex high-dimensional sensory inputs, agents learned control policies without any support from learning algorithms. Subsequently, several recent studies (Kim et al., 2023; Kim and Choi 2013; Lee et al., 2020; Lee et al., 2022 [4–7].) demonstrated

* Corresponding author.
E-mail address: jiyang@kumoh.ac.kr (J. Yang).

https://doi.org/10.1016/j.net.2024.10.009
Received 7 June 2024; Received in revised form 29 August 2024; Accepted 8 October 2024
Available online 15 October 2024

that RL methods have a high potential to replace traditional algorithms based on proportional-integral (PI), proportional-integral-differential (PID), lead-lag compensators, and manual controls. Lee et al. (2020) proposed an algorithm that couples an RL method with a rule-based system for a fully automated power-increase operation without operator intervention. Their simulation demonstrated that the proposed algorithm automatically followed a power increase from 2 % to 100 % without the support of traditional controllers, implying that the algorithm was autonomous. Recently, Lee et al. (2022) compared the control performance of three types of controllers in a cold shutdown operation: deep reinforcement learning (DRL)-based, Ziegler–Nichols (ZN)-tuned PID-based, and DRL-tuned PID-based controllers. The DRL-tuned PID--based controller achieved the best performance, with the smallest error and shortest time to attain a target value. Park et al. (2022) introduced a deep neural network structure and a compact nuclear simulator (CNS)-coupled DRL mechanism for automating the plant heat-up mode [8]. They demonstrated that the proposed method performed similarly to manual control when anticipating the pressurizer pressure and level. A higher disparity between rewards than that in the preceding stage improves the performance of DRL. Kim et al. (2023) proposed action-coordinating strategies that allow optimum control actions to replace manual control during startup operations. The strategies presented coordinated multiple agents when required to perform competing tasks that could result in unintentional outcomes. Comparing the average values of the monitored variables during reactor coolant system (RCS) heat-up yielded the highest ratings for optimal action among the three quantification methodologies. Thus, existing literature suggests that RL algorithms can help improve the automation of traditional controllers for operation modes in NPPs.

However, most previous studies introduced existing CNSs for transient simulation of large-scale NPPs which are not a type of SMR. Moreover, they combined existing CNSs with independent RL algorithms, necessitating external communication interfaces between them. This may require additional computation and memory resources for external coupling. On the contrary, the nuclear transient simulator considered in this study simulates a typical SMR that includes once-through steam generators (SG) instead of recirculation type SGs and build a learning environment capable of direct coupling with RL algorithms inside a source code without any external coupling.

A fully manual core startup operation with RL algorithms is yet to be investigated for the aforementioned NPP operation modes. The core startup operation has indeed been carried out manually because reaching criticality requires careful manipulation of the control rods; otherwise, it may result in core power runaway. Although manual control for the core startup operation has been received reasonable for SMRs that generate power on the ground, it would be inappropriate for nuclear propulsion SMRs, which are normally boron-free (Alam et al., 2019a; Alam et al., 2019b) [9,10], in the following ways.

(i) Boron-free SMRs have a greater control rod worth than that of soluble-boron SMRs, necessitating a more precise manipulation of the control rods, which can place an undue load on operators in time-critical scenarios.
(ii) Manual control of the core startup operation is longer than that of automated controls. This is contrary to the objectives of nuclear propulsion SMRs, and therefore, the core startup operation should be accelerated through automation.

This study aims to develop an efficient SMR dynamic simulation model that can be utilized as the environment in an RL algorithm and to automate the reactor core startup using the developed RL framework. The proposed SMR model is referred to as “S-SMRSim” in this study; the detailed methodology for embedding the S-SMRSim in the RL algorithm is provided. It is validated by comparing the simulation results to the numerical simulation and experimental data available in the literature. Further, this study provides an in-depth analysis of the learning environment, agents, actions, and states necessary for automating the reactor core startup process using the RL framework. The reward conditions essential for reliably reaching the target output are also presented. To demonstrate the effectiveness of our proposed approach, we select DQN and PPO as representative agents that cover both value-based and policy-based algorithm families.

2. Overview

Fig. 1 shows a schematic of the SMR system considered in this study, conceptually designed to have an approximately 100 MW thermal output based on SMART (Yang et al., 2023) [11]. An SMR consists of all major machines, including circulating pumps, steam generators, and a reactor core, in one reactor. The water is pressurized using compressed nitrogen gas to prevent boiling the reactor coolant. Fig. 1(a) shows that water is used as the reactor coolant. Circulating pumps recirculate water through a reactor core, circulating pump, and steam generator. The recirculation loop is considered the primary side. The heat generated by the fission reaction is transferred to the cold water in the reactor core. The hot water passing through the reactor core is cooled, when it flows on the primary side of the steam generator, by the feed water flowing on the secondary side of the steam generator.

Fig. 1(b) shows that the fission rate, which is proportional to the heat generation rate, can be adjusted by changing the position of the control rod. As shown in Fig. 1(b), the initial position of the control rod is 0 m, indicating that there is no heat generation if the residual heat is negligible. Control rods can be inserted to decrease the fission reaction rate or withdrawn to increase the heat generation rate. At the SMR reactor core startup, the critical state must be achieved by actuating the control rods; this critical state implies that nuclear chain reactions are sustained by maintaining a sufficient number of neutrons. In this study, a 0.01 % reactor core output is considered the critical state.

Automation of the reactor core startup can save time and reduce the workload of the operators. However, it is difficult to control the control rods using conventional feedback controllers (PID, Lead-lag compensator, etc.) because the reactor core output can vary nonlinearly, with a substantially wide range of scales, and an extremely fast or slow changing nature.

In this study, a novel approach for automatic reactor core startup is proposed via RL with the development of a simplified SMR simulator (S-SMRSim). RL aims to train an agent to accomplish a task in an environment. The agent interacts with the environment by receiving observations and states, and in return, it sends actions back to the environment. The reward indicates how effective an action is in achieving the task’s goal. The agent is composed of two key parts: a policy and a learning algorithm. The policy is a function that determines which actions to take based on the observations it receives. This policy is often represented by a function approximator with adjustable parameters, like a deep neural network. The learning algorithm’s role is to continually update the policy’s parameters using the data from the actions taken, the observations made, and the rewards received. The aim is to discover an optimal policy that maximizes the total reward throughout the task. In other words, reinforcement learning involves an agent improving its behavior through repeated interactions with the environment, learning through trial and error without direct human input. Fig. 2 shows the S-SMRSim, which is an efficient environment for providing the corresponding states and rewards for each action. The agent generates an action that indicates the movement of the control rod, such as withdrawal or insertion. The actions of an agent can be produced based on experience (trained neural network) or a random function. The action is applied as the input to S-SMRSim, and the position of the control rod is updated. Subsequently, numerical time integration is executed for each time step in S-SMRSim. The S-SMRSim simulates the transient thermal-hydraulic phenomena of the SMR system through nodes and paths, as shown in Fig. 3.

The input of S-SMRSim, which involves the node and path structures,

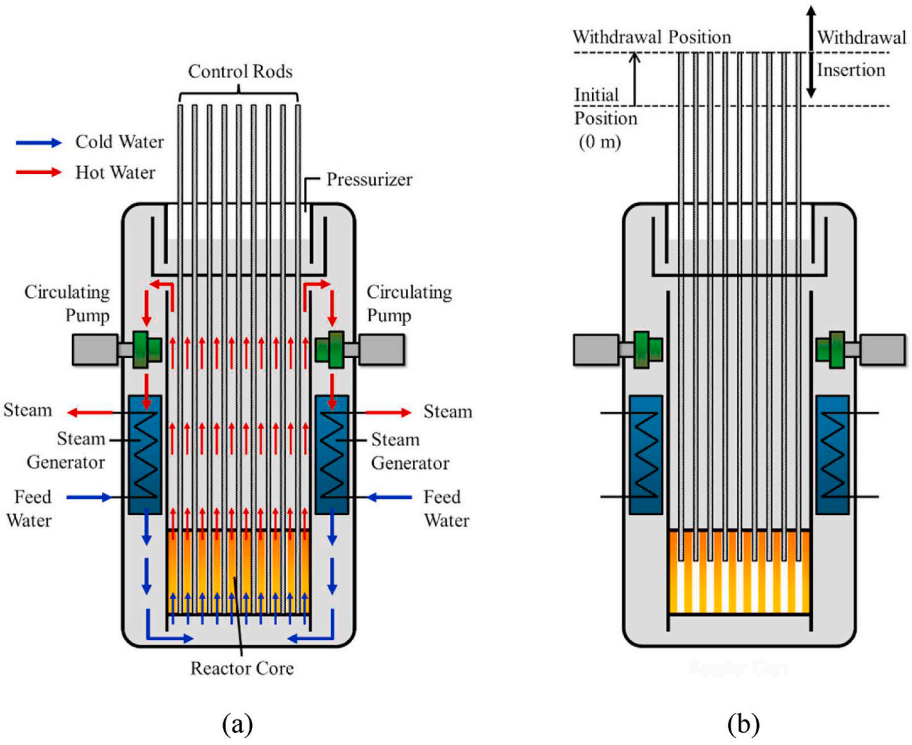

Fig. 1. Schematic of an SMR system: (a) Major machines and flow, (b) control rod's actions and withdrawal position.

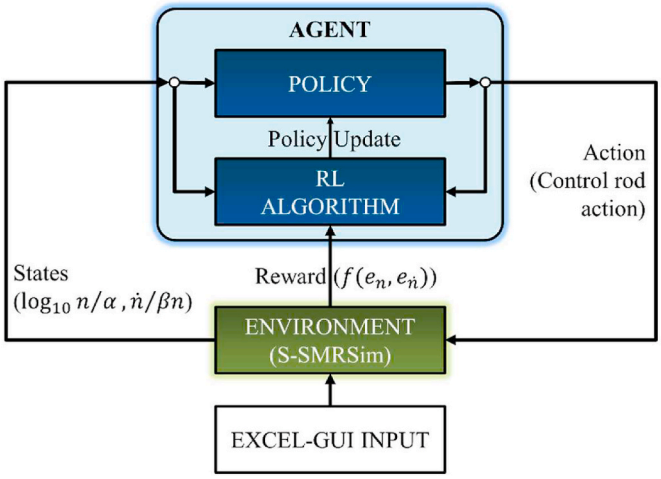

Fig. 2. Illustration of interaction between S-SMRSim and the agent for the automation of the SMR reactor core startup.

can be easily modified by the Excel GUI environment and incorporated into the RL algorithm. Thus, S-SMRSim can be applied to optimize various operational and control strategies. The computation of S-SMRSim for a time step yields solutions for time-dependent variables at the time. The defined states and rewards are evaluated using these solutions. During the training process, the states and rewards for an agent's action are saved as data, and the neural network of the agent's policy is trained using the data to produce the action to maximize the reward. Eventually, the agent can maximize the reward. The trained agent can control the control rods to achieve the critical state during reactor core startup, and thus, automation can be achieved.

3. Simplified-SMR simulator (S-SMRSim) for RL environment

3.1. Governing equations

Establishing a suitable RL environment to train an agent by transferring time-efficient and accurate information is challenging. S-SMRSim is proposed as a solution to this challenge. Fig. 4 shows the representative node and path expressions required to model the SMR system in Fig. 3. There are two node and path types for the fluid and the structure, respectively, and an additional heat transfer path can link the fluid and structure nodes to consider the heat transfer. As shown in Fig. 3, the feed-water flow rate is applied as the external flow (mass source) of the fluid node, and the heat generated in the reactor core is modeled as the heat source of the structure node.

From the governing laws with respect to mass, momentum, and energy conservation, the mass, momentum, and energy balance equations for each fluid and structure node can be written as follows:

Mass balance for a fluid node i is given by

$$\frac{dm_{fn,i}}{dt} = \sum_{(j)=1}^{n_{fe}} \dot{m}_{fe,(j)} + \dot{m}_{fn,ext,i}. \tag{1}$$

Momentum balance for a fluid path (j) given by

$$\frac{L_{fe,(j)}}{A_{fe,(j)}} \frac{d\dot{m}_{fe,(j)}}{dt} + \frac{\dot{m}^2_{fe,(j)}}{A^2_{fe,(j)}} \left(\frac{1}{\rho_{fn,i}} - \frac{1}{\rho_{fn,j}} \right) = \left(p_{fn,j} - p_{fn,i} \right) + \Delta p_{fe,(j),pmp} + \rho_{fe,(j)} g_{fe,(j)} \left(H_{fn,j} - H_{fn,i} \right) - K_{fe,(j)} \frac{\dot{m}_{fe,(j)} \left| \dot{m}_{fe,(j)} \right|}{2\rho_{fe,(j)} A^2_{fe,(j)}}. \tag{2}$$

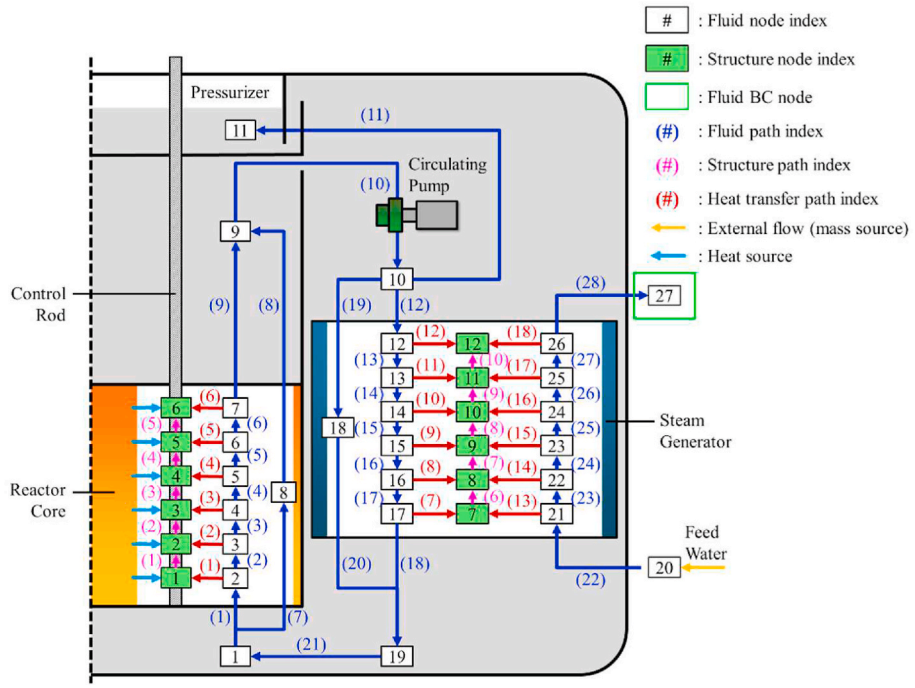

Fig. 3. Node and path diagram implemented in S-SMRSim.

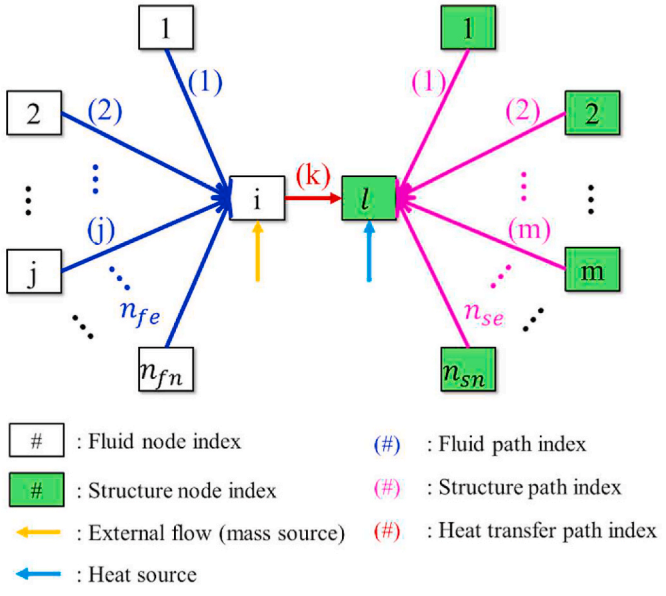

Fig. 4. Node and path diagram for fluid node i and structure node *l* connected by the heat transfer path (k).

Energy balance for a fluid node i is given by

$$\frac{dm_{fn,i}h_{fn,i}}{dt} = \sum_{(j)=1}^{n_{fe}} \dot{m}_{fe,(j)}h_{fe,(j)} + \frac{V_{fn,i}dp_{fn,i}}{dt} - \dot{Q}_{ht,(k)}. \tag{3}$$

Energy balance for a structure node *l* is given by

$$\frac{dm_{sn,l}C_{p,sn,l}T_{sn,l}}{dt} = \sum_{(m)=1}^{n_{se}} k_{se,(m)}A_{se,(m)}\frac{T_{sn,m} - T_{sn,l}}{\Delta x_{se,(m)}} + \dot{Q}_{ht,(k)} + \dot{Q}_{src,l}, \tag{4}$$

where $m_{fn,i}$, $\dot{m}_{fn,ext,i}$, h_{fn}, and $V_{fn,i}$ represent the mass, external mass flow rate, enthalpy, and volume, respectively, for fluid node i. $\dot{m}_{fe,(j)}$ represents the mass flow rate for fluid path (j). For the mass flow rate, positive and negative values indicate inflow and outflow, respectively. n_{fe} represents the number of fluid paths connected to fluid node i. $A_{fe,(j)}$, $L_{fe,(j)}$, and $K_{fe,(j)}$ represent the area, length, and loss coefficient of the path, respectively; ρ_{fn}, p_{fn}, and H_{fn} represent the fluid density, static pressure, and absolute height of the node, respectively; and j represents the node index connected to fluid node i. $\Delta p_{fe,(j),pmp}$, $\rho_{fe,(j)}$, $g_{fe,(j)}$, and $h_{fe,(j)}$ represent the pressure increase, density, gravity, and enthalpy of the pump for path (j), respectively.

The path enthalpy was determined by applying an upwind scheme to the nodal enthalpy. $m_{sn,l}$ and $C_{p,sn,l}$ represents the mass and specific heat for the structure node *l*, respectively; $k_{se,(m)}$, $A_{se,(m)}$, and $\Delta x_{se,(m)}$ represent the conductivity, area, and length for the structure path (m), respectively; and n_{se} represents the number of structural paths. $T_{sn,l}$ and $T_{sn,m}$ represent the temperatures for structure nodes *l* and m, respectively.

$\dot{Q}_{src,l}$ represents the heat generation rate in the reactor core, which can be expressed by multiplying the normalized neutron density (x_{rk}) by the nominal core output ($N_{rk,nm}$).

$$\dot{Q}_{src,l} = x_{rk}N_{rk,nm}. \tag{5}$$

Point kinetics equation with delayed neutrons is used to calculate the heat generation rate ($\dot{Q}_{src,l}$). The time derivative terms of the point kinetics model are discretized by Euler implicit which is consistent with the fluid model, and both models use the same time step for integration. Therefore, the time-dependent variables of the thermal hydraulic model and the neutron kinetics model are determined simultaneously through an iterative computation process at each time step. The theory for calculating heat generation in the reactor core is based on the normalized six-group reactor kinetic equations, which are

$$\frac{dx_{rk}}{dt} = \frac{R_{rk,t} - \beta_{rk}}{\Lambda_{rk}}x_{rk} + \frac{\sum_{i=1}^{6}\beta_{rk,i}y_{rk,i}}{\Lambda_{rk}} + S_{rk}. \tag{6}$$

$$\frac{dy_{rk,i}}{dt} = \lambda_{rk,i}\left(x_{rk} - y_{rk,i}\right), (\mathrm{i} = 1, \ldots, 6) \tag{7}$$

where Λ_{rk}, $\beta_{rk,i}$, and $\lambda_{rk,i}$ represent the average generation time, fraction

of delayed neutrons emitted by the precursor, and the decay constant of the delayed neutron concentration, respectively; S_{rk} represents reactivity; and β_{rk} represents the sum of $\beta_{rk,i}$. x_{rk} and $y_{rk,i}$ are normalized by their steady-state values, as expressed by

$$x_{rk} = \frac{N_{rk}}{N_{rk,s}}, \tag{8}$$

$$y_{rk,i} = \frac{C_{rk,i}}{C_{rk,s,i}}, \tag{9}$$

where N_{rk} and $C_{rk,i}$ represent the neutron density and delayed neutron concentration, respectively; the subscript s denotes steady-state values. The steady-state value of $C_{rk,i}$ can be expressed as

$$C_{rk,s,i} = \frac{\beta_{rk,i} N_{rk,s}}{\lambda_{rk,i} \Lambda_{rk}}, \tag{10}$$

where $R_{rk,t}$ represents the total reactivity, which considers the reactivity of the control rod ($R_{rk,cr}$), fuel ($R_{rk,fl}$), coolant ($R_{rk,cl}$), and external ($R_{rk,ext}$) effects.

$$R_{rk,t} = R_{rk,cr} + R_{rk,fl} + R_{rk,cl} + R_{rk,ext.} \tag{11}$$

$\dot{Q}_{ht,(k)}$ is the heat transfer rate for the heat transfer path (k), which indicates the heat transfer between the fluid and structure nodes. The heat transfer rate for the heat transfer path (k) can be expressed using the overall heat transfer coefficient ($U_{ht,(k)}$) and heat area ($A_{ht,(k)}$) as

$$\dot{Q}_{ht,(k)} = U_{ht,(k)} A_{ht,(k)} \left(T_{fn,i} - T_{sn,l}\right). \tag{12}$$

$T_{fn,i}$ represents the temperature of fluid node i. The overall heat transfer coefficient is a function of the heat transfer coefficient and the thermal resistance of the structure, and the heat transfer coefficient can be applied as a constant value or correlation.

3.2. Node and path modules

There are two types of pipe and tank node modules. A fluid node with a relatively small volume and fast response, such as a pipe or heat exchanger, is modeled as a pipe–node module. In contrast, a fluid node with a relatively large volume and slow response, such as a pressurizer, condenser, or steam separator, is regarded as a tank node module. For the pipe node module, the left-hand term of the mass-balance equation in Eq. (1) can be rewritten using the chain rule for thermodynamic properties

$$V_{fn,i}\left(\frac{\partial \rho_{fn,i}}{\partial p_{fn,i}}\frac{dp_{fn,i}}{dt} + \frac{\partial \rho_{fn,i}}{\partial h_{fn,i}}\frac{dp_{fn,i}}{dt}\right) = \sum_{(j)=1}^{n_{fe}} \dot{m}_{fe,(j)} + \dot{m}_{fn,ext,i.} \tag{13}$$

The momentum-balance equation in Eq. (2) can be rearranged into an explicit function of the dependent variable ($\dot{m}_{fe,(j)}$) with numerical differentiation, and it can be substituted into Eq. (13) to obtain the matrix equation with the static pressure ($p_{fn,i}$) as the dependent variable.

Unlike the pipe node module, the tank node module directly obtains the mass of the fluid node ($m_{fn,i}$) through the mass balance in Eq. (1) with numerical differentiation. Then, the static pressure can be calculated using the ideal gas law or thermodynamic property relation. For the pressurizer and condenser, a gas and liquid mixture exists inside; therefore, the ideal gas law is applied, as shown by

$$p_{fn,i} = \frac{m_{g,i}}{V_{fn,t,i} - \frac{m_{fn,i}}{\rho_{fn,i}}} R_{g,i} T_{g,i}, \tag{14}$$

where $m_{g,i}$, $R_{g,i}$, $T_{g,i}$, and $V_{fn,t,i}$ represent the gas mass, gas constant, gas temperature, and total volume of the fluid node i, respectively. The mass and energy balance equations for the gas phase were also considered. In addition, the static pressure of the steam separator was determined by the thermodynamic property relationship (Lemmon et al., 2007) [12] expressed by

$$p_{fn,i} = f\left(\rho_{fn,i} = \frac{m_{fn,i}}{V_{fn,i}}, h_{fn,i}\right). \tag{15}$$

The path module determined the mass flow rate with respect to the fluid path. The path module includes the pipe, turbine, and valve-path modules. The pipe path module applies Bernoulli's law, which can be utilized to model a general pipe (see Eq. (2)), heat exchanger, etc. It includes the pump pressure increase term such that the pump is considered a function of the mass flow rate ($\dot{m}_{fe,(j)}$), rotating speed ($\omega_{fe,(j),pmp}$), etc., as shown by

$$\Delta p_{fe,(j),pmp} = f\left(\dot{m}_{fe,(j)}, \omega_{fe,(j),pmp}, \ldots\right). \tag{16}$$

Further, a turbine is applied based on Flugel's formula (Liu et al., 2011) [13], which is expressed as

$$\dot{m}_{fe,(j)} = K_{fe,bld,(j)} \sqrt{\rho_{fe,(j)} \max\left[p_{fn,(i)}, p_{fn,(j)}\right] \left(1 - \frac{\min\left[p_{fn,(i)}, p_{fn,(j)}\right]}{\max\left[p_{fn,(i)}, p_{fn,(j)}\right]}\right)^2}, \tag{17}$$

where the $K_{fe,bld,(j)}$ represents the blade coefficient estimated at the design point. The valve path module is included in S-SMRSim, and the mass flow rate is expressed as a function of the valve coefficient (CV) with respect to the valve opening, as

$$\dot{m}_{fe,(j)} = f(CV, \ldots). \tag{18}$$

3.3. Computation procedure

S-SMRSim receives the action of the control rod and yields the states and rewards for a time step. The overall computational process for S-SMRSim for each time step is represented in Fig. 5. First, the position of the control rod is updated by the given action after loading all time-dependent variables saved at the previous time step. If the mass flow

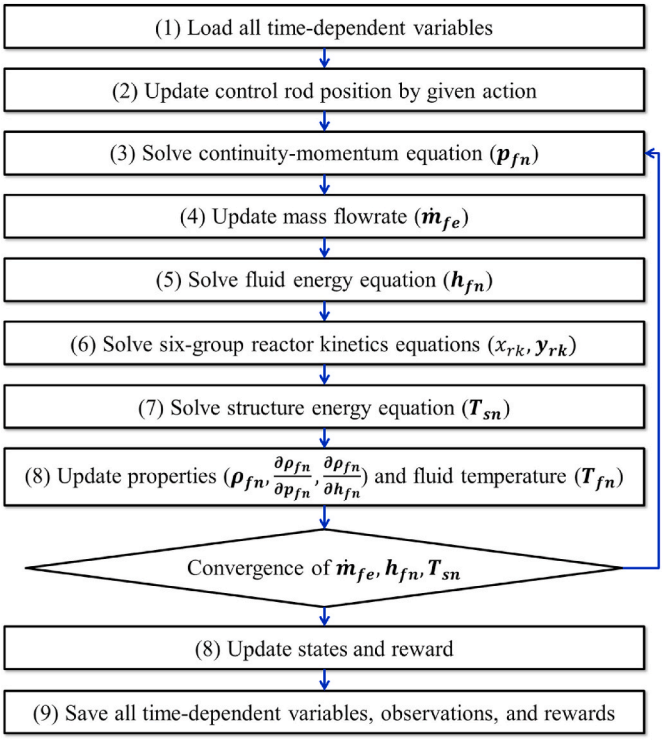

Fig. 5. S-SMRSim computation procedure for a time step during RL process.

rate equation for the path modules, such as Eqs. (2), (17) and (18), are substituted into the mass balance equation, the continuity-momentum matrix equation can be derived, which involves static pressure ($\boldsymbol{p_{fn}}$) as a dependent variable. Global matrices, which include the continuity-momentum and energy matrices, can be obtained by applying the FEM rule (Palazzolo, 2016) [14]. This rule assembles an elemental matrix for a path utilizing the connectivity matrix. From the static pressure solution obtained by solving the continuity-momentum matrix equation, the mass flow rate ($\dot{\boldsymbol{m}}_{\boldsymbol{fe}}$) can be updated using Eqs. (2), (17) and (18). In the updated static pressure and mass flow rate solution fields, enthalpy ($\boldsymbol{h_{fn}}$) is calculated by solving the fluid energy equation in Eq. (3). The heat generation rate in the reactor core can then be updated via the six-group reactor kinetics equations for x_{rk} and $\boldsymbol{y_{rk}}$. Then, the heat generation rate is applied to the structural node to solve the structural energy equation (Eq. (4)) to obtain the structural temperature solution ($\boldsymbol{T_{sn}}$). The fluid properties ($\boldsymbol{\rho_{fn}}, \frac{\partial \rho_{fn}}{\partial p_{fn}}, \frac{\partial \rho_{fn}}{\partial h_{fn}}$) and temperature ($\boldsymbol{T_{fn}}$) are updated by employing the embedded thermodynamic property relationships (Lemmon et al., 2007) [12]. In addition, the Euler implicit method is used for numerical differentiation with respect to time derivatives. Computation processes (3)–(8) shown in Fig. 5 continue until the relative errors of the dependent variables ($\dot{\boldsymbol{m}}_{\boldsymbol{fe}}, \boldsymbol{h_{fn}}, \boldsymbol{T_{sn}}$) converge. After the computation converges, the states and rewards are evaluated and saved using the action and all time-dependent variables.

4. Model validation

To verify the validity of the proposed S-SMRSim, the simulation results of S-SMRSim were compared with the numerical analysis and experimental cases available in (Hancox and Banerjee, 1977; Dutta and Doshi, 2008; Colombo et al., 2012; Takitani and Takemura, 1978) [15–18]. The three internal heated pipe flow cases used in the comparison are presented in Fig. 6; their input parameters are listed in Table 1. The pipe was modeled by connecting the fluid nodes and paths in a line, and heat was directly applied to the fluid node as the heat source term. In Takitani's study (Takitani and Takemura, 1978) [18], the thermal mass of the tube was considered in addition to the structure nodes and paths; therefore, heat was applied to the structure node (tube), and the heat transfer path connected the structure and fluid nodes. The node number and time step listed in Table 1 are adopted through sufficient test simulations. In all cases, the pipe loss coefficient in the momentum balance equation was estimated from the Blasius correlation (Vijayan et al., 2019) [19], and the fluid properties were modified based on mass fraction (Kim and Choi, 2013) [5]. For Case C, a correlation for the heat transfer coefficient was required, and several correlations (Kim and Choi, 2013; Shah, 1979; Chen, 1966; Biasi et al., 1967; Dittus and Boelter, 1985; Saha and Zuber, 1974) [5,20–24] were employed to account for the influence of the water, water-steam mixture, and steam states.

Table 1
Input parameters for all validation cases.

Parameter	Case A: Hancox (Hancox and Banerjee, 1977) [15]	Case B: Solberg (Colombo et al., 2012) [17]	Case C: Takitani (Takitani and Takemura, 1978) [18]
Inner diameter [mm]	4.33	5.25	12.5
Outer diameter [mm]	–	–	15
Length	7.316	2.9	5.252
Roughness [μm]	–	25	50
Inlet loss coefficient	–	17.8	500–520
Outlet loss coefficient	–	17.8	10
Node number	152	122	202
Time step [s]	0.001	0.001	0.1
Total simulation time [s]	7	10	100

* HTC: Heat transfer coefficient.

Hancox and Banerjee (1997) [15] presented a numerical model and its simulation results for internal pipe flow (Fig. 6(a)), and Dutta and Doshi (2008) [16] harnessed the results for validation. The initial mass flow rate and enthalpy were 0.1065 kg/s and 1184.7 kJ/kg; the fluid inside the pipe was maintained in a liquid state. For the boundary condition, the static pressures ranged from 7.102 MPa (inlet) to 6.984

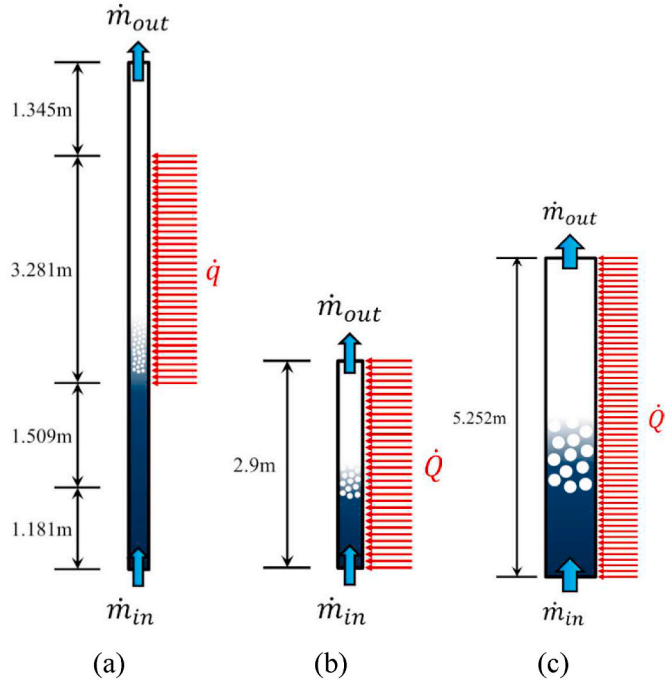

Fig. 6. Comparative case studies for S-SMRSim's validation: (a) Hancox (Hancox and Banerjee, 1977) [15], (b) Solberg (Colombo et al., 2012) [17], and (c) Takitani (Takitani and Takemura, 1978) [18].

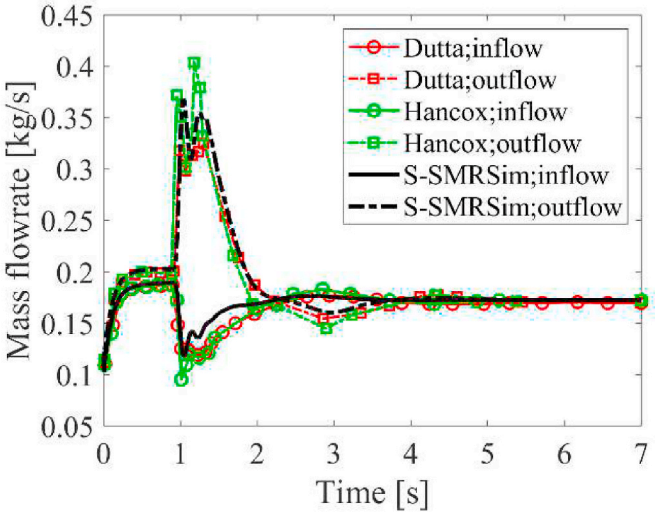

Fig. 7. Comparison of mass flowrates at the inlet and outlet between previous studies (Hancox (Hancox and Banerjee, 1977) [15] and Dutta (Dutta and Doshi, 2008) [16]) and S-SMRSim results.

MPa (outlet), and the initial static pressures were linearly distributed from the inlet to the outlet nodes. In addition, the inlet enthalpy was prescribed to be identical to the initial enthalpy (1184.7 kJ/kg). Fig. 7 shows the simulation results for the cases presented in Hancox and Banerjee's (1977) and Dutta and Doshi (2008) studies [15,16]. The mass flow rates at the pipe inlet and outlet were in contrast with the results produced by S-SMRSim. At the initial stage (from 0 to approximately 1 s), the mass flow rate is increased, and a phase change of the fluid occurs because of the heat applied to the fluid node. Then, the mass flow rate at the pipe outlet increases, while the amount of steam produced and the static pressure increase inside the pipe. In contrast, the mass flow rate decreased at the pipe inlet. The disparity in the response time between the mass flow rate, static pressure, and enthalpy oscillates the mass flow rate with a phase lag between the inflow and outflow until it reaches a steady-state value. Comparison between S-SMRSim and Hancox and Banerjee (1977) and Dutta and Doshi (2008) [15,16] helped confirm the validity of S-SMRSim, as shown in Fig. 7.

A two-phase flow implies an oscillating feature, and the oscillation may be stable or unstable depending on the flow conditions. An increase in heat flux (heat input) (or a decrease in mass flux, pressure, inlet pressure, etc.) worsens two-phase flow instability (dynamic instability). For example, as shown in Fig. 8(a), although there is an oscillation in the mass flow rate, it reaches the steady-state value. However, as the heat input increases from 24 to 25 kW, the oscillation amplitude does not decrease, and the oscillation is amplified under the 25 kW heating condition, as shown in Fig. 8(c). Fig. 8(b) shows that the heat input exhibiting flow instability is the instability onset threshold.

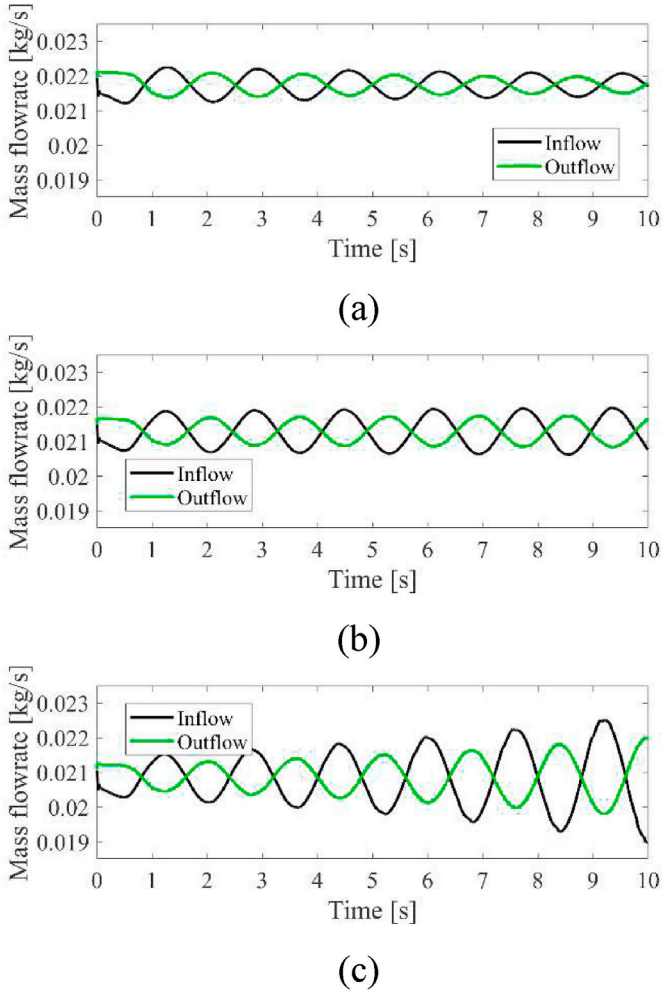

Fig. 8. Mass flowrates at the inlet and outlet (Case-B) according to heat input change, $\dot{Q}$ = (a) 24 kW, (b) 24.5 kW, (c) 25 kW.

Further details on the two-phase instability are not specified because they are beyond the scope of this study. The case is only utilized for validating S-SMRSim. As shown in Fig. 9, the instability onset threshold can be expressed by the nondimensional parameters N_{pch} and N_{sub} in the test conducted by Solberg (Colombo et al., 2012) [17] as

$$N_{pch} = \frac{\dot{Q} v_{fg}}{\dot{m} v_f (h_f - h_g)}, \tag{19}$$

$$N_{sub} = \frac{(h_f - h_{in}) v_{fg}}{h_{fg} v_f}, \tag{20}$$

where $\dot{Q}$, $\dot{m}$, and h_{in} represent the heat input, mass flow rate, and inlet enthalpy, respectively; v_f and v_{fg} are the specific volume of the saturated water and its difference from the specific volume of the saturated steam, respectively; and h_f and h_{fg} are the enthalpy of the saturated water and its difference from the saturated steam enthalpy, respectively.

For Case B listed in Table 1, static pressures are applied to the inlet and outlet as the boundary conditions. The outlet static pressure is 8.11 MPa, and the inlet static pressure and enthalpy are adjusted for generating six cases that can be contrasted to the experimental data, as shown in Fig. 9. The instability onset thresholds are searched while increasing or decreasing the heat input with 0.5 kW. The dynamic simulation was performed using steady-state solutions; the heat input was perturbed immediately after the simulation. S-SMRSim showed a superior prediction ability for the instability onset threshold observed in the Solberg (Colombo et al., 2012) [17] experiment. In addition, the S-SMRSim results contrast with the results obtained from Takitani's test (Takitani and Takemura, 1978) [18]. Fig. 10 shows that the heat input that produces instability is searched in S-SMRSim and compared with Takitani's experiment for eight cases (Takitani and Takemura, 1978) [18].

The transient analysis with an initial 1 % heat perturbation for 10 s was initiated from the steady-state solutions as initial conditions; the inlet conditions are listed in Table 2. The inlet static pressure and outlet external mass flow rate were imposed when performing calculations in the steady state. In the transient analysis, both the inlet and outlet were prescribed using a static pressure. The outlet static pressure obtained from the steady-state computation was used as the outlet boundary condition. In all comparison results presented in this section, including Takitani's test (Takitani and Takemura, 1978) [18], S-SMRSim shows good agreement with the numerical results and experimental data

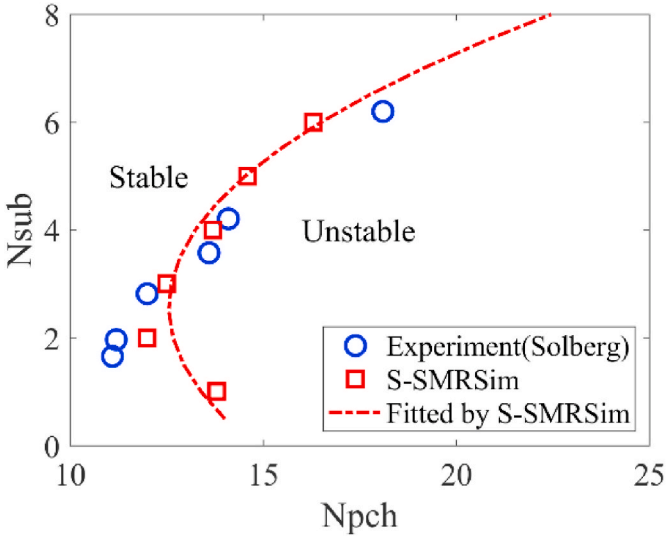

Fig. 9. Stability map comparison between experiment (Solberg (Colombo et al., 2012) [17]) and S-SMRSim.

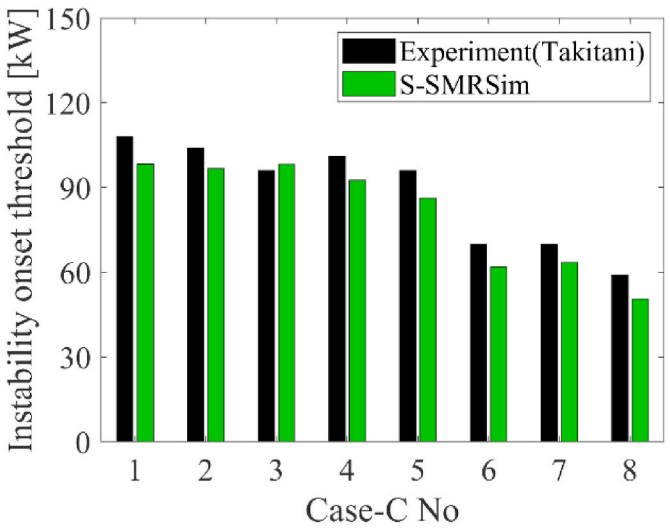

Fig. 10. Instability onset threshold comparison between the experiment (Takitani (Takitani and Takemura, 1978) [18]) and S-SMRSim for entire cases.

Table 2
Specific inlet conditions of Case C.

Case no.	Inlet pressure [MPa]	Inlet mass flux [kg/m^2]	Inlet enthalpy [kJ/kg]	Inlet loss coefficient
1	4.053	337	621.5	520
2	4.296	335	669.0	520
3	4.104	323	707.9	500
4	4.256	318	643.1	500
5	4.256	298	608.8	500
6	4.053	220	630.1	520
7	4.083	220	617.3	500
8	4.154	177	621.6	520

acquired by other researchers, thus verifying the validity of the proposed S-SMRSim.

5. Reinforcement learning

The RL methodology is an advanced approach for solving sequential decision problems using reward signals. In the RL paradigm, an agent interacts with its environment and identifies the optimal actions required to achieve a given goal. An behavior of the agent is dictated by the state information obtained from the environment, and the agent is rewarded according to its actions. Consequently, RL can be characterized as a process by which an agent develops a policy and systematic plan to guide its actions to maximize the accumulation of rewards.

The advantages of RL include its ability to tackle complex decision problems and independence from large amounts of labeled data because it relies on trial-and-error learning rather than supervised learning. Capitalizing on these advantages, recent research on deep reinforcement learning (DRL), which incorporates deep neural networks (DNNs), has been active and produced excellent results in various domains such as optimization, control, and design problems. However, RL also faces challenges, including the difficulty in designing suitable learning environments for agents, need for extensive simulations, complexity of creating appropriate reward functions, and exploration-exploitation dilemma. Constructing a suitable learning environment is necessary to apply RL to any problem; this is achieved using S-SMRSim.

A multitude of algorithms exist for training agents. Although accurate categorization is difficult, these algorithms can be divided into value-based and policy-based algorithms, with certain methods integrating aspects from both. Among the value-based algorithms, Q-learning is a quintessential example that determines optimal actions by estimating the state and action values. As an off-policy approach, Q-learning refines the current action value function $Q(s,a)$, referred to as the Q value, by employing the Bellman equation to approximate the optimal action value function $Q^*(s,a)$, signifying the maximum cumulative reward. The cumulative reward can be calculated using

$$Q^*(s,a) = E_{s'\sim\varepsilon}[r + \gamma \underset{a'}{max} Q^*(s',a')|s,a]. \quad (21)$$

If the optimal value $Q^*(s',a')$ for sequence s' at the subsequent time step is known across all potential actions a', the optimal strategy would involve selecting action a' that maximizes the expected value of $r+\gamma Q^*(s',a')$ () [3]. Representative algorithms that incorporate DNN include deep Q-networks (DQN) (Mnih et al., 2013; Mnih et al., 2015) [3,25] and double DQN (Van Hasselt et al., 2016) [26].

Policy-based algorithms represent policy optimization methods that maximize the cumulative rewards in a given environment without explicitly estimating the value function. The policy function is updated by increasing the gradient of the objective function for maximizing the expected cumulative reward in the current state. This optimization is executed in an on-policy manner, with notable algorithms including trust region policy optimization (TRPO) (Schulman et al., 2015) [27], proximal policy optimization (PPO) (Schulman et al., 2017) [28], and advantage actor–critic (A2C) (Mnih et al., 2016) [29].

In this study, we employed DQN and PPO because they are recognized for their robust performance within the respective categories of value- and policy-based methodologies.

5.1. DQN

A DQN (Mnih et al., 2013; Mnih et al., 2015) [3,25] is an advanced RL algorithm that effectively combines the strengths of DNNs with the value-iteration algorithm used in traditional Q-learning. Unlike conventional Q-learning algorithms that use tables to store Q values and are suitable for small discrete states and action spaces, DRL employs DNNs to estimate Q values in high-dimensional state spaces. To address the dimensional challenges frequently encountered in traditional Q-learning, a DQN utilizes a deep Q-network to approximate the Q values. This approach enables the efficient estimation of Q values for each action within the current state, significantly improving the storage and calculation efficiency when the actions are selected using the greedy strategy.

DQN is the cornerstone algorithm among Q-learning-based approaches, and although its fundamental design is suited for discrete and low-dimensional action spaces, it has significantly influenced a diverse array of fields, including video games, robot control, and HVAC system control (Mnih et al., 2013; Mnih et al., 2015 Fang et al., 2022; Gu et al., 2017; Lillicrap et al., 2015) [3,25,30–32].

5.2. PPO

PPO (Schulman et al., 2017) [28] is a popular on-policy policy gradient algorithm. PPO adopts the same surrogate loss function as its predecessor, TRPO (Schulman et al., 2015) [27], which has demonstrated notable performances. Furthermore, PPO is characterized by the inclusion of a clipping function, which imposes a penalty if an excessively large policy update occurs, circumventing the constraints of the existing objective function while maintaining impressive performance. This is referred to as the clipped surrogate objective, and it is expressed as

$$L^{CLIP}(\theta) = \widehat{\mathbb{E}}_t[\min(r_t(\theta)\widehat{A}_t, \mathrm{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\widehat{A}_t)], \quad (22)$$

where $\widehat{A}$ and $\widehat{E}$ represent the empirical estimates of the advantage function and expectation, respectively; $r(\theta)$ represents the probability ratio; and ϵ is a hyperparameter. Subscript t represents the values at time step t.

In the PPO framework, a policy surrogate and a value function are both incorporated into the loss function, resulting in a neural network architecture that considers both policy and value information. The PPO loss function also includes a loss term known as the entropy bonus, which fosters efficient exploration. The combined loss function is calculated for each iteration, and it is expressed as

$$L_t^{CLIP+VF+S}(\theta) = \widehat{\mathbb{E}}_t\left[L_t^{CLIP}(\theta) - c_1 L_t^{VF}(\theta) + c_2 S[\pi_\theta](s_t)\right]. \quad (23)$$

Where c_1 and c_2, L_t^{VF}, and S represent the coefficients of each loss term, value function error term, and entropy bonus, respectively. PPO accumulates trajectories equal to the horizon size and subsequently conducts training through stochastic gradient descent (SGD) on trajectories up to a specified epoch within each minibatch.

The PPO is a straightforward, pragmatic, and high-performance algorithm applicable to both continuous and discrete action spaces. Therefore, it has been extensively employed and implemented across numerous domains, including self-driving cars, drone control, robot control, and natural language processing with models such as ChatGPT (Guan et al., 2020; Bøhn et al., 2019; Zhao et al., 2023; Ouyang et al., 2022) [33–36].

5.3. RL training condition for automating the core startup

A nuclear power plant consists of a complex apparatus that generates electricity by maintaining the nuclear chain reaction in the core in a critical state and utilizing the resulting heat. The state of the nuclear chain reaction can be classified as subcritical, critical, or supercritical based on the multiplication factor (k), which is the ratio of the number of neutrons produced in fission generation to the number of neutrons present in the previous generation. Based on the operating state, the power plants are regulated to maintain a subcritical state with $k < 1$ during shutdown and a critical state with $k = 1$ during operation.

Core startup in a nuclear power plant refers to the initial process of activating a nuclear reactor by gradually increasing its power output. Excluding the decay heat, this process can be described as a transition from a zero-power subcritical state ($k < 1$) to an initial low-power critical state ($k = 1$). This process is essential for establishing a controlled and stable chain reaction within the reactor. In large nuclear power plants, operators monitor the status of the reactor during core startup and regulate the reactivity of the core by adjusting the control rods and boric acid concentration in accordance with relevant protocols.

Reactors are highly nonlinear and complex systems that require an equally nonlinear control strategy, making such control extremely challenging. Operations that transition the reactor from an initial, very low, subcritical state to a critical state, such as core startup, can cause potentially significant increases in the core power if the core reactivity is not controlled properly. Therefore, the core startup process in large commercial power plants is carried out over an extended period, with the operator managing the control rod adjustments and boric acid concentration changes.

In contrast, SMRs benefit from the relatively low operator intervention and a reduction in auxiliary equipment, such as boric acid water injection systems. This study aims to demonstrate that core startups can be achieved solely through control-rod management using RL.

In this core startup, the initial condition was assumed to be a subcritical state with 10^{-8} % power output, and the final condition was a 10^{-2} % critical state. The initial temperature of the reactor was constant in all sections, and the feed water to the steam generator was supplied continuously at the same initial temperature. Because core startup is a low-power process, the temperature change in the reactor during core startup is assumed to be nonsignificant. Thus, the main factor affecting core reactivity is the reactivity of the control rod position.

The reactivity of the control rod corresponding to the control-rod drive was incorporated into the reactor kinetic equations based on the control-rod position in the form of a lookup table. The control rod drive was assumed to be retracted, inserted, or stopped by 1 step every 0.2 s.

The RL structure is shown in Fig. 2. The SMR serves as the environment, whereas the agent controlling the control rod is represented by the DQN and PPO algorithms used in this study.

Environment: An environment is a system or framework to which RL is applied. In this study, a 100 MWth-class SMR, as shown in Fig. 1, is the target system. The goal is to perform the reactor core startup of this system, and therefore, we modeled the system in a simplified form (Fig. 3) and used it as the environment.

Action: Action refers to the decision made by an agent to withdraw, insert, or halt the control rod in the context of an RL environment.

State: States provide crucial information for an agent to determine the appropriate action within the reinforcement learning environment. Consequently, suitable system conditions must be selected as states for the agent to make informed decisions and achieve the desired objectives.

1) This study focuses on core startups, and therefore, it is essential for the agent to receive core power information. As discussed previously, there is a 10^6-fold difference between the two outputs because the initial state of the core startup is 10^{-8} % and the final state is approximately 10^{-2} %. Selecting the current output as the state results in the agent making decisions based on an excessively wide range of information during core startup; this is not conducive to effective decision making. Therefore, we transform the value into a dimensionless quantity given by $\log_{10}$(current output)/$\log_{10}$(initial output) and use it as the first element of the state. The current power is assumed to be the filtered value of the power calculated from the core physics model; a low-pass filter with a time constant of 50 s is assumed.
2) The second element is the rate of power change. During core startup, there is an approximate position of the control rod at which criticality is reached as the subcritical core transitions to criticality. Assuming that the initial core-start condition is subcritical when the control rod is fully inserted, the core power increases relatively slowly until it reaches criticality, even if the control rod is withdrawn. However, a rapid increase in core power can occur if the control rod is withdrawn too quickly and passes the critical position. At this point, it may be difficult to suppress the abrupt power increase, even with rapid insertion, owing to the limited speed of the control rod movement. Therefore, the time change rate of the logarithmic power was selected as the second-state element [37]; it is expressed as

$$\xi := \frac{1}{\overline{N}}\frac{d\overline{N}}{dt}. \quad (24)$$

Eq. (24) enables the agent to select an action based on both the current core power and the power change rate. For convenience, we refer to this as the power change rate. The power change rate divided by the empirical constant value was selected as the second element of the state.

Reward: A reward evaluates the previous actions of an agent; the choice is crucial in RL. When solving a control problem with RL, choosing a positive reward can cause the agent to learn in the wrong direction by accumulating this value during an episode. Therefore, in this study, the agent was assigned a negative reward to facilitate optimal control learning. In this RL framework, the reward was evaluated in two ways (called costs), as shown below. The learning process needs to be designed to minimize both costs simultaneously.

Cost 1 (reward associated with reactor power): In this RL training, the initial state of each episode was set to 10^{-8} % of the core power, and the final state was achieved by control maintenance at approximately 10^{-2} %. Therefore, the cost related to the reactor power was selected as the first element of the reward. The corresponding cost expression indicates that the cost is minimized as the target power is approached. Consequently, Cost 1 is designed to help the agent reach the target point with the most efficient possible decisions. The form of Cost 1, where P_{err}

represents the reactor power error term used in the calculations, is given as

$$P_{err} = \frac{1}{\log(N_{tgt})} - \frac{1}{\log(N_{crt})}.\ Cost\ 1 = 5 \bullet {P_{err}}^2. \tag{25}$$

Cost 2 (Reward related to the power change rate): The second cost is associated with the power change rate to prevent the rapid escalation of reactor power and to encourage a stable power increase. By maintaining the power change rate close to a certain value during the increase from the initial power to the target power, the cost is minimized by instructing the agent to increase the power while maintaining a certain power change rate. This cost takes different values for different ranges of current power, as shown in the expression below. A steady increase in power is induced by maintaining the power change rate until the power increases to a certain extent. Eventually, as the current power approaches the target power, the cost of achieving the target power is minimized. In scenario where the power is significantly greater than the target power, a significant cost is imposed to encourage learning to prevent such deviations. The form of cost 2, where PCR_{err} represents the power change-rate error term, is given by

$$PCR_{err} = \left|33.2\left(\xi_{tgt} - \xi_{crt}\right)\right|.\ Cost\ 2 = \begin{cases} PCR_{err}, N_{crt} < Y \times 10^{-5} \\ PCR_{err}\left|\frac{10}{3} - \frac{10^5}{3}N_{crt}\right|, Y \times 10^{-5} \le N_{crt} \le 1.15 \times 10^{-4} \\ 0.5, 1.15 \times 10^{-4} > N_{crt} \\ where, Y = 7\ (for\ DQN)\ or\ 9\ (for\ PPO) \end{cases}. \tag{26}$$

Cost 2 is designed to increase the reward as the power approaches the target value of $N_{crt} = 10^{-4}$. Therefore, the Y value is designed to range approximately between 5 and less than 10, meaning $(Y \times 10^{-5}) < N_{crt} <$ (over 10×10^{-5}) in Eq. (26). Both DQN and PPO models were trained by varying the Y value within this range, and it was observed that stable learning occurred at Y = 7 for DQN and Y = 9 for PPO. Therefore, the reward for this episode is represented by

$$Reward_{step} = \begin{cases} -500, if\ N_{crt} > 0.1, \\ -(Cost\ 1 + Cost\ 2), else \end{cases} \tag{27}$$

If the power output during this episode deviates significantly from the target value, a large penalty is applied and the episode ends. At the end of each episode, the rewards from each step are added to provide the total reward for that episode.

Table 3
List of hyperparameters for DQN training.

Hyperparameter	Value
Replay buffer size	187500
Mini-batch size	256
Learning rate	0.001
Target network update frequency	4
Discount factor	0.99
Epsilon (for ε-greedy policy)	0.0005
Network architecture	FC Neural Network [2 hidden layers with 24, 24 units]

Table 4
List of the hyperparameters of PPO training.

Hyperparameter	Value
Experience horizon	600
Mini-batch size	128
Learning rate	0.0001
Epochs	3
Discount factor	0.99
Clip factor	0.0005
Entropy coefficient	0.01
GAE factor	0.95
Network architecture of actor and critic	FC neural network [3 hidden layers with 128, 256, and 512 units]

6. Results

6.1. Training setup

The DQN and PPO algorithms were selected as agents, which necessitates determining the hyperparameters of each algorithm for training. The primary objective of this investigation was to determine the feasibility of achieving appropriate power control by applying RL during core startup; the hyperparameters were selected using empirical methods. The selected hyperparameters for DQN and PPO are described in Tables 3 and 4, respectively. Both algorithms use the Adam optimization method. For the neural network size, it is minimized to avoid impacting the agent's performance while saving computational time. For other hyperparameters, the default values provided by the commercial tool are generally used as a good starting point. However, some hyperparameters (such as relay buffer size, target update frequency, and clip factor) are adjusted from their default values to enhance agent performance.

6.2. Training results

6.2.1. Case 1: decide action every cycle

The agent influences the environment by deciding whether to withdraw, insert, or stop the control rod. In this study, the control period of the control-rod adjustment was assumed to be 0.2 s. Consequently, the

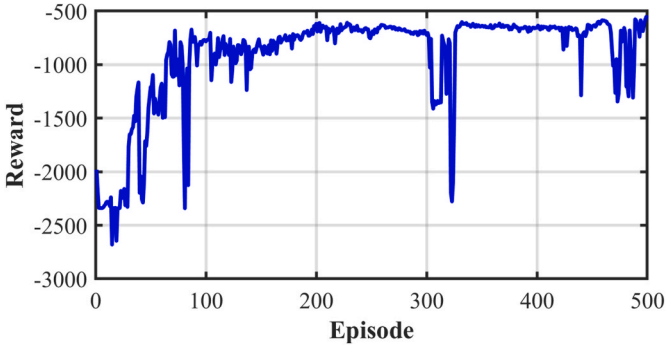

Fig. 11. Rewards per episode during DQN learning in Case 1.

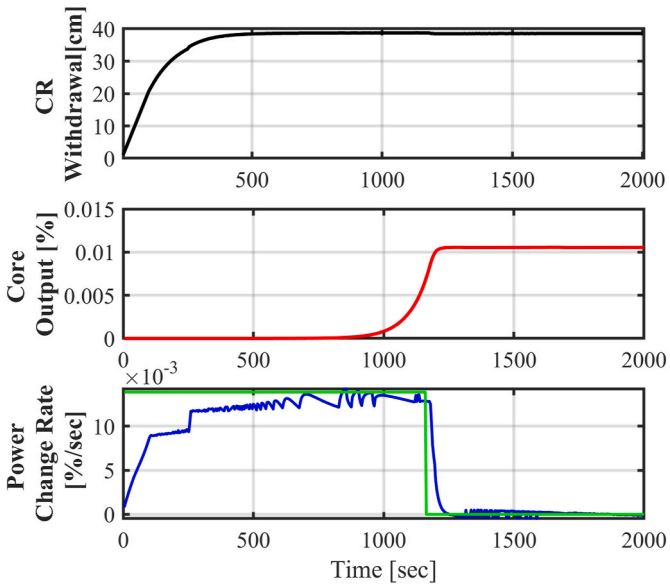

Fig. 12. DQN learning outcomes in Case 1: Control rod position (top), Core output (middle), and power change rate (bottom).

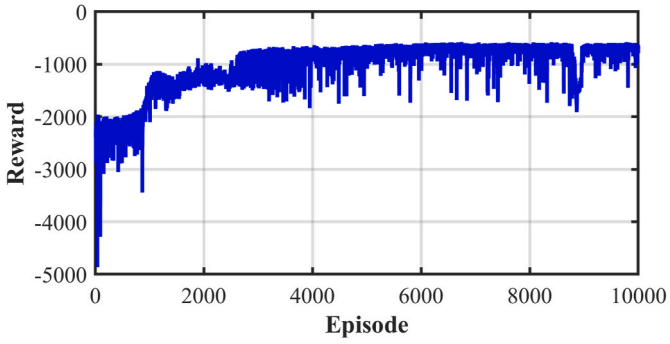

Fig. 13. Rewards per episode during PPO learning in Case 1.

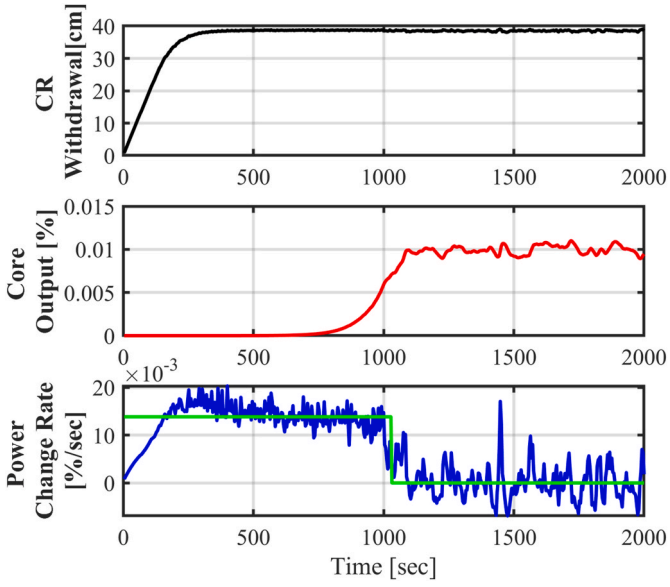

Fig. 14. PPO learning outcomes in Case 1: Control rod position (top), core output (middle), and power change rate (bottom).

agent was trained in a scenario where the agent updated the state information from the environment every 0.2 s and made three action decisions (withdrawal, insertion, and stop). This scenario was referred to as Case 1. Figs. 11–14 show the results for DQN and PPO for Case 1.

Figs. 11 and 13 show the rewards per episode during the learning process for the DQN and PPO, respectively, in Case 1. The reward figures show that the DQN achieved substantial rewards with relatively few training episodes, whereas the PPO required a comparatively long training period before achieving a degree of reward stability. Figs. 12 and 14 show the core startup results for the DQN and PPO at the end of training, respectively. The differences in the results are attributed to the type of agent although learning is performed using the same reward function. The results using the DQN as the agent show that high rewards can be achieved with less training time than that in the case of the PPO; the results after training are relatively stable.

6.2.2. *Case 2: decide action every two cycles*

The results of Case 1 demonstrate the potential of RL for accomplishing this task; however, there is room for improvement. In Case 1, the agent is trained to obtain a state and perform an action every 0.2 s. However, this condition is unlikely to lead to higher rewards even with further learning. Therefore, the characteristics of the physical model need to be integrated more strongly into the agent training process. Although the agent can perform an action every 0.2 s, this interval is insufficient for the action to manifest itself in the environment and for the state information to be provided to the agent to make appropriate decisions. The state includes information on the core power and power change rate derived from the calculations of the SMR physical model, including the reactor kinetics model. Therefore, the effect of a previous action on the state of the environment depends on the responsiveness of the system; 0.2 s is potentially insufficient to provide sufficient state information to the agent.

Therefore, based on the knowledge gained from Case 1, training was carried out with the setting that the state is transmitted to the agent every 0.4 s, with the agent deciding the action to adjust the control rod for the following two steps. Thus, considering the responsiveness of the environment, the agent decides the action for two steps every 0.4 s, and if the insertion of the control rod is −1, the stop is 0 and the withdrawal is 1. The agent chooses one of the following decisions: [−1, −1], [−1, 0], [0, 0], [1, 0], [1, 1]. This scenario is referred to as Case 2. The training results for DQN and PPO are presented below.

Figs. 15 and 17 show the rewards during the learning process for DQN and PPO, respectively, in Case 2. Compared with the previous reward episode results for Case 1, there is a tendency to converge to a relatively higher reward range in Case 2.

Fig. 16 shows the simulated core startup with the trained DQN agent, indicating that the reward is relatively more optimized compared to results of Case 1. The power change rate follows the target value better from approximately 0 to 1100 s with an increase in power. For the PPO results shown in Fig. 18, the rewards in Case 2 converge to a higher

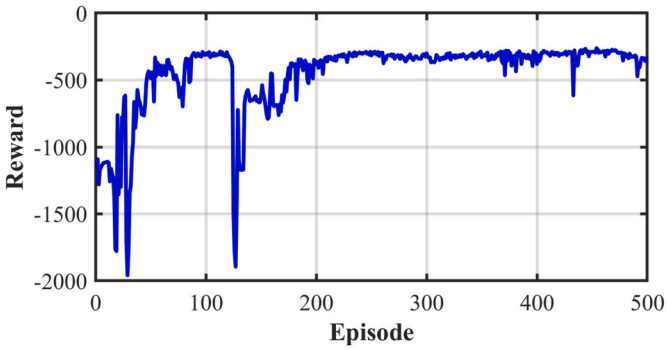

Fig. 15. Rewards per episode during DQN learning in Case 2.

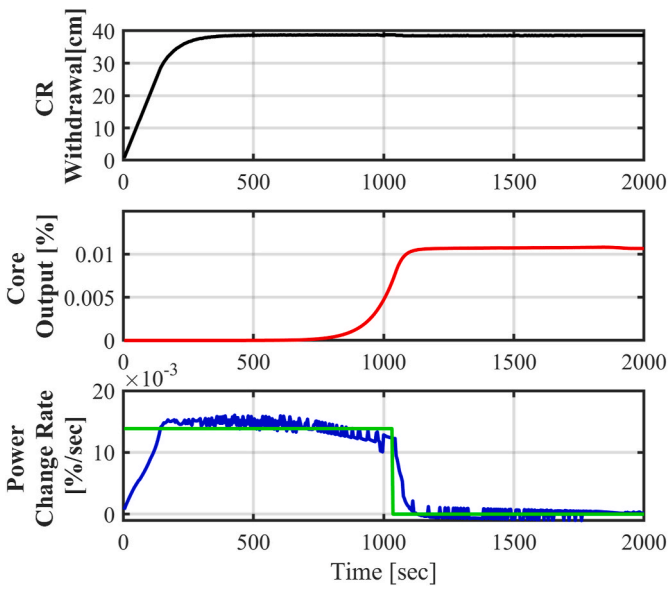

Fig. 16. DQN learning outcomes in Case 2: Control rod position(top), Core output(middle), Power change rate(bottom).

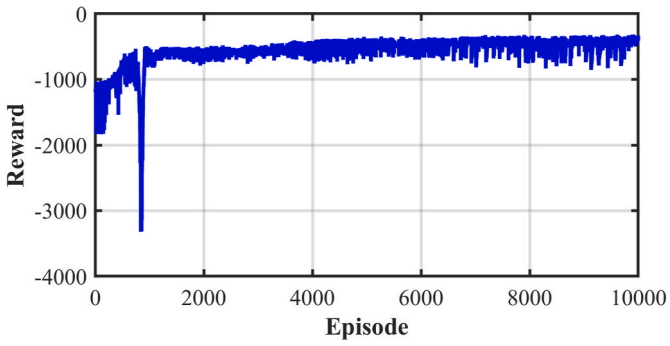

Fig. 17. Rewards per episode during PPO learning in Case 2.

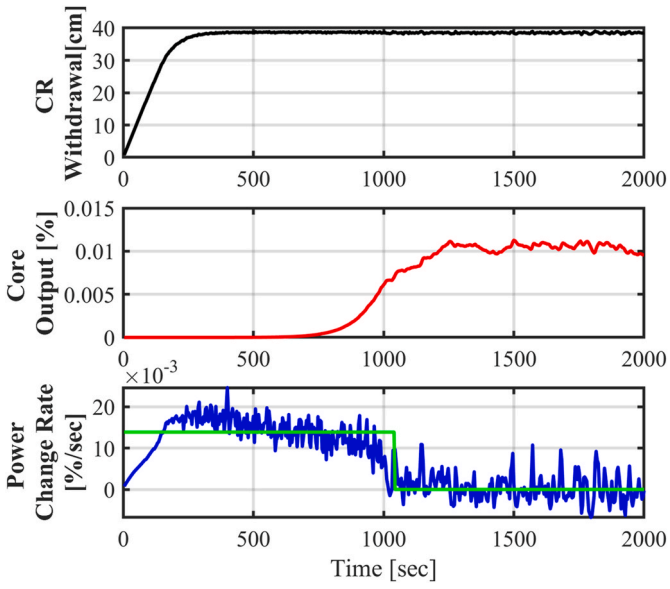

Fig. 18. PPO learning outcomes in Case 2: Control rod position(top), Core output(middle), Power change rate(bottom).

region compared to that for Case 1; however, they still appear relatively less stable compared to the DQN results.

The DQN minimizes the movement of the control rod near the target power and maintains a relatively stable power; however, it does not fully converge to the target power of 0.01 %. The PPO result attempts to achieve the 0.01 % target power with a constant control rod movement around the target power. This can be caused by the conceptual difference between the two RL algorithms; however, it requires further investigation because this can be caused by many other factors (e.g., hyperparameters and network architecture).

7. Conclusion

This study introduced an efficient SMR dynamic simulation model as an RL environment to automate the operation of the reactor core startup. The developed SMR model was referred to as "S-SMRSim" in this study for its first disclosure and straightforward use in the present and future publications. S-SMRSim was embedded in the RL algorithm such that S-SMRSim received the action of the agent and produced the states and reward. The agent actions in the RL algorithm included withdrawal, insertion, or stay motion of the control rod. The states and rewards were appropriately designed by utilizing the reactor core output and its change rate. The excel-GUI input, which can easily be revised, was created so that the application can be extended to various cases, helping the transition from manual SMR to automatic operations.

S-SMRSim models a dynamic system through simplified nodalization to reduce computational load and to conduct an uncomplicated dynamic system analysis. The mass, momentum, and energy balances for the node or path govern the transient thermal-hydraulic phenomena. S-SMRSim was validated by comparing it with theoretical and experimental results acquired in three previous studies (Hancox and Banerjee, 1977; Colombo et al., 2012; Takitani and Takemura, 1978) [15,17,18]. All compared cases were for internal heated pipe flow, which implied a two-phase oscillating flow. From the comparative study, S-SMRSim demonstrated acceptable accuracy when compared to the numerical or experimental data. The results confirmed that S-SMRSim is a reliable code for use in an RL environment.

An agent equipped with an adequate understanding of the underlying physics, together with the ability to make informed decisions based on the provided state and reward structure, is required to facilitate the successful implementation of RL for core startup control. We selected the appropriate state and reward to reliably achieve the target power output while maintaining the power change rate during core startup. To this end, the agent used the DQN and PPO decision-making algorithms. The training process involved iteratively refining the agent's policy to minimize the total costs associated with the reactor power and power change rate. During the learning phase, the agent explored different control rod movements and continuously updated its policy based on the rewards and state transitions. The training process converges when the agent's policy stabilizes and consistently provides appropriate control actions to ensure a safe and stable core startup process.

The results indicated that the DQN algorithm achieved better targeted learning than that with the PPO algorithm because the final target state is consistent with the same initial state during learning, making it easier to optimize the value function for each step. The performance differences between PPO and DQN can be attributed to their distinct update mechanisms. DQN updates its parameters at each timestep, allowing for more frequent learning from discrete decision points, whereas PPO updates its parameters after collecting a batch of experiences over multiple timesteps, which might result in less responsiveness to sparse but critical decision moments. To address this, the number of episodes for PPO training was increased by 20 times compared to DQN, as illustrated in Figs. 13 and 17. Despite this adjustment, the results suggest that the discrete nature of the decision period in this example made it more suitable for DQN. PPO performs similarly to DQN in compensating for the power change rate as the reactor power increases;

however, its performance is relatively low when the core power is close to the target power. Although the hyperparameters of each algorithm were not optimized, the results suggested that both algorithms were likely to perform adequately in targeted learning. Thus, selecting an appropriate state and reward has a dominant effect on the corresponding RL. Based on the state and reward presented, the agent demonstrated potential to perform targeted output control as the learning progressed. The core startup scenario assumed in this study was in the range of 10^{-8} to 10^{-2} % output. Considering that there is a power change of approximately 10^{6} times between the initial and final conditions, the results showed that the proposed reward, state, and agent configurations enabled RL for a successful core startup. Thus, by carefully designing the state and reward structures and using advanced RL algorithms, such as DQN and PPO, this study demonstrated the potential for developing intelligent control strategies that can understand and manage the complexity of reactor physics.

There may be potential challenges related to real-world implementation, sensitivity to initial conditions, and the computational resources required for training. Experimental verification with the control hardware system will be necessary. Additionally, the proposed method needs to account for various initial conditions during agent training and demonstrate robustness in such cases. Efforts should further be made to accelerate the simulation time of the theoretical model to reduce overall training time.

CRediT authorship contribution statement

Seong Jun Bae: Writing – review & editing, Writing – original draft, Software, Methodology. **Hong Hyun Son:** Writing – review & editing, Writing – original draft. **Yongjae Lee:** Investigation. **Jongin Yang:** Writing – review & editing, Writing – original draft, Validation, Software, Methodology, Investigation, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

Dr. Yang developed S-SMRSim by modifying and improving the author's previous work (Yang et al., 2023) [11], and Bae and Yang implemented it in the RL algorithm. This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government (MSIT) (RS-2024-00451800).

NOMENCLATURE

$m_{fn,i}$	Mass for a fluid node i, kg
$\dot{m}_{fn,ext,i}$	External mass flowrate for a fluid node i, kg/s
$h_{fn,i}$	Enthalpy for a fluid node i, kJ/kg
$V_{fn,i}$	Volume for a fluid node i, m^3
$\rho_{fn,i}$	Density for a fluid node i, kg/m^3
$p_{fn,i}$	Static pressure for a fluid node i, Pa
$H_{fn,i}$	Absolute height for a fluid node i, m
$\dot{m}_{fe,(j)}$	Mass flowrate for a path (j), kg/s
$\rho_{fe,(j)}$	Density for a path (j), kg/m^3
$g_{fe,(j)}$	Gravity acceleration for a path (j), m/s^2
$h_{fe,(j)}$	Enthalpy for a path (j), kJ/kg
n_{fn}	Number of fluid nodes
n_{sn}	Number of structure nodes
n_{fe}	Number of fluid paths
n_{se}	Number of structure paths
$A_{fe,(j)}$	Area of a path (j), m^2
$L_{fe,(j)}$	Length of a path (j), m
$K_{fe,(j)}$	Loss coefficient of a path (j)
$\Delta p_{fe,(j),pmp}$	Pressure increase by pump for a path (j), Pa
$m_{sn,l}$	Mass for a structure node *l*, kg
$C_{p,sn,l}$	Specific heat for a structure node *l*, J/kg•K
$k_{se,(m)}$	Conductivity for a structure path (m), W/m•K
$A_{se,(m)}$	Area for a structure path (m), m^2
$\Delta x_{se,(m)}$	Length for a structure path (m), m
$T_{sn,l}$	Temperature for a structure node *l*, K
$\dot{Q}_{src,l}$	Heat generation rate in reactor core for a structure node l, W
x_{rk}	Normalized value of neutron density (N_{rk})
$y_{rk,i}$	Normalized value of ith delayed neutron concentration ($C_{rk,i}$)
Λ_{rk}	Average generation time, s
$\beta_{rk,i}$	Fraction of ith delayed neuron emitted by the precursor
$\lambda_{rk,i}$	Decay constant of ith delayed neutron concentration
β_{rk}	Summation of $\beta_{rk,i}$
S_{rk}	Reactivity source
$U_{ht,(k)}$	Overall heat transfer coefficient for a heat transfer path (k), W/m^2•K
$A_{ht,(k)}$	Heat area for a heat transfer path (k), m^2
$T_{fn,i}$	Temperature for a fluid node i, K
$m_{g,i}$	Mass of gas phase for a fluid node i, kg
$R_{g,i}$	Gas constant for a fluid node i, J/kg•K
$T_{g,i}$	Temperature for a fluid node i, K
$V_{fn,t,i}$	Total volume for a fluid node i, m^3
s	State
a	Action

(*continued on next page*)

(*continued*)

r	Reward
ε	Emulator
π_θ	Stochastic policy
γ	Discount factor
θ	Weights
k	Neutron multiplication factor
ξ	Power change rate, s^{-1}
N	Neutron density
P_{err}	Reactor power error term
PCR_{err}	Power change rate error term
Subscripts	
fn	Fluid node
fe	Fluid path
sn	Structure node
se	Structure path
rk	Reactor kinetics
pmp	Pump
ext	External
nm	Nominal
s	Steady-state
cr	Control rod
fl	Fuel
cl	Coolant
err	Error term
tgt	Target value
crt	Current value
Acronyms	
SMR	Small modular reactor
RL	Reinforcement learning
DRL	Deep reinforcement learning
DNN	Deep neural network
DQN	Deep Q-network
TPRO	Trust region policy optimization
PPO	Proximal policy optimization
A2C	Advantage actor-critic
SGD	Stochastic gradient descent

References

[1] S. Bouckaert, A. Pales, C. McGlade, U. Remme, B. Wanner, L. Varro, D. D'Ambrosio, T. Spencer, Net Zero by 2050 : A Roadmap for the Global Energy Sector, 2021, p. 224.

[2] S. Hirdaris, Y. Cheng, P. Shallcross, J. Bonafoux, D. Carlson, B. Prince, G. Sarris, Considerations on the Potential Use of Nuclear Small Modular Reactor (SMR) Technology for Merchant Marine Propulsion, 2014, pp. 101–130. OE, vol. 79.

[3] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, M. Riedmiller, Playing Atari with Deep Reinforcement Learning, 2013 *arXiv preprint arXiv:1312.5602*.

[4] J. Kim, J. Bae, S. Lee, Strategy to coordinate actions through a plant parameter prediction model during startup operation of a nuclear power plant, Nucl. Eng. Technol. 55 (2023) 839–849.

[5] J. Kim, S. Choi, The influence of tube and membrane structure on dynamic instability in furnace wall tubes of a fossil fired once-through boiler in a simulated furnace environment, J. Mech. Sci. Technol. 27 (2013) 3553–3560.

[6] D. Lee, A. Arigi, J. Kim, Algorithm for autonomous power-increase operation using deep reinforcement learning and a rule-based system, IEEE (2020) 196727–196746.

[7] D. Lee, S. Koo, I. Jang, J. Kim, Comparison of deep reinforcement learning and PID controllers for automatic cold shutdown operation, Energies (2022) 1–25.

[8] J. Park, T. Kim, S. Seong, S. Koo, Control automation in the heat-up mode of a nuclear power plant using reinforcement learning, Prog. Nucl. Energy 145 (2022) 104107.

[9] S. Alam, C. Goodwin, G. Parks, Parametric neutronics analyses of lattice geometry and coolant candidates for a soluble-boron-free civil marine SMR core using micro-heterogeneous duplex fuel, Ann. Nucl. Energy 129 (2019) 1–12.

[10] S. Alam, B. Almutairi, T. Ridwan, D. Kumar, C. Goodwin, K. Atkinson, G. Parks, Neutronic investigation of alternative & composite burnable poisons for the soluble-boron-free and long life civil marine small modular reactor cores, Sci. Rep. 9 (2019) 19591.

[11] J. Yang, H.H. Son, Y.J. Lee, D. Shin, T. Kim, S.S. Choi, Boundary condition coupling methods and its application to BOP-integrated transient simulation of SMART, Nucl. Eng. Technol. (2023).

[12] E. Lemmon, M. Huber, M. McLinden, NIST Standard Reference Database 23: Reference Fluid Thermodynamic and Transport Properties-REFPROP, National Institute of Standards and Technology, Gaithersburg, Maryland, USA, 2007, Version 8.0.

[13] J. Liu, S. Yan, D. Zeng, A new measurement model for main steam flow of power plants, Proc. Environ. Sci. 11 (2011) 18–24.

[14] A. Palazzolo, Vibration Theory and Applications with Finite Elements and Active Vibration Control, Wiley, Chichester, UK, 2016.

[15] W. Hancox, S. Banerjee, Numerical standard for flow boiling analysis, Nucl. Sci. Eng. 64 (1977) 106–123.

[16] G. Dutta, J. Doshi, A characteristic-based implicit finite-difference scheme for the analysis of instability in water cooled reactors, Nucl. Eng. Technol. 40 (6) (2008) 477–488.

[17] M. Colombo, A. Cammi, D. Papini, M. Ricotti, RELAP5/MOD3.3 study on density wave instabilities in single channel and two parallel channels, Prog. Nucl. Energy 56 (C) (2012) 15–23.

[18] K. Takitani, T. Takemura, Density wave instability in once-through boiling flow system, (I) experiment, J. Nucl. Sci. Technol. 15 (5) (1978) 355–364.

[19] P.K. Vijayan, A.K. Nayak, N. Kumar, Single-phase. Two-Phase and Supercritical Natural Circulation Systems, Woodhead Publishing, 2019.

[20] M. Shah, A general correlation for heat transfer during film condensation inside pipes, Journal of Heat and Mass Transfer 22 (4) (1979) 547–556.

[21] J. Chen, A correlation for boiling heat transfer to saturated fluid in convective flow, in: 6th National Heat Transfer Conference, 1963. Boston, MA, USA.

[22] L. Biasi, e. al, Studies on burnout, Part 3, Energ. Nucl. 14 (9) (1967) 530–536.

[23] F.W. Dittus, L.M.K. Boelter, Heat transfer in automobile radiators of the tubular type, Reprinted in Int. Commun. Heat Mass 12 (1985) 3–22.

[24] P. Saha, N. Zuber, Point of Net vapor generation and vapor void fraction in subcooled boiling, in: Proceedings of the 5th International Heat Transfer Conference, 1974. Tokyo, Japan.

[25] V. Mnih, 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, D. Hassabis, Human-level control through deep reinforcement learning, Nature 518 (7540) (2015) 529–533.

[26] H. van Hasselt, A. Guez, D. Silver, Deep reinforcement learning with double Q-learning, arXiv preprint arXiv:1509.06461 (2015).

[27] J. Schulman, S. Levine, P. Abbeel, M. Jordan, P. Moritz, Trust region policy optimization, in: In Proceedings of the 32nd International Conference on Machine Learning, 2015, pp. 1889–1897.

[28] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal Policy Optimization Algorithms, 2017 *arXiv preprint arXiv:1707.06347*.

[29] V. Mnih, A.P. Badia, M. Mirza, A. Graves, T. Harley, T.P. Lillicrap, D. Silver, K. Kavukcuoglu, Asynchronous methods for deep reinforcement learning, in:

Proceedings of the 33nd International Conference on Machine Learning (ICML), 2016, pp. 1928–1937.
[30] X. Fang, G. Gong, L. Chun, P. Peng, W. Li, X. Shi, X. Chen, Deep reinforcement learning optimal control strategy for temperature setpoint real-time reset in multi-zone building HVAC system, Appl. Therm. Eng. 212 (2022) 118552.
[31] S. Gu, E. Holly, T. Lillicrap, S. Levine, Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. 2017 IEEE International Conference on Robotics and Automation (ICRA), 2017, pp. 3389–3396.
[32] T.P. Lillicrap, J.J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, D. Wierstra, CONTINUOUS CONTROL WITH DEEP REINFORCEMENT LEARNING, 2015 *arXiv preprint arXiv:1509.02971*.
[33] Y. Guan, Y. Ren, S.E. Li, Q. Sun, L. Luo, K. Li, Centralized cooperation for connected and automated vehicles at intersections by proximal policy optimization, IEEE Trans. Veh. Technol. 69 (11) (2020) 12597–12608.
[34] E. Bøhn, E.M. Coates, S. Moe, T.A. Johansen, Deep reinforcement learning attitude control of fixed-wing UAVs using proximal policy optimization. 2019 International Conference on Unmanned Aircraft Systems (ICUAS), 2019, pp. 523–533.
[35] L. Zhao, T. Chang, J. Zhang, L. Zhang, K. Chu, L. Guo, D. Kong, A policy optimization algorithm based on sample adaptive reuse and dual-clipping for robotic action control, Appl. Soft Comput. 134 (2023) 109967.
[36] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C.L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, J. Schulman, J. Hilton, F. Kelton, L. Miller, M. Simens, A. Askell, P. Welinder, P. Christiano, J. Leike, R. Lowe, Training language models to follow instructions with human feedback, arXiv preprint arXiv: 2203.02155 (2022).
[37] M. Eom, D. Chwa, D. Baang, Robust disturbance observer-based feedback linearization control for a research reactor considering a power change rate constraint, IEEE Trans. Nucl. Sci. 62 (3) (2015) 1301–1312.