Sim-to-real transfer in reinforcement learning-based, non-steady-state control for chemical plants

Shumpei Kubosawa, Takashi Onishii, and Yoshimasa Tsuruoka

1. Introduction

Chemical plants are complex and dynamical systems for manufacturing chemical products. Figure 1 shows a chemical plant for binary distillation (i.e. separating two components from a mixture). Generally, chemical plants leverage sensitive chemical phenomena such as the vapour-liquid equilibrium, reaction, and polymerization. To meticulously control complex and interdependent phenomena, chemical plants feature many sensors and manipulation points such as valves and switches. Additionally, to maintain continuous and stable production, modern industrial chemical plants are equipped with automatic controllers such as proportional-integral-derivative (PID) controllers and model predictive controllers (MPC) [1].

Although the extent of controllable situations in which conventional automatic controllers are usually applied is in the proximity of a few steady states (i.e. situations considered in the development of the controller), it is difficult for these controllers to maintain unexperienced steady states or to optimize non-steady situations, such as transition operations (e.g. changing the plant load and product grade). Transition operations are performed relatively frequently in industrial chemical plants. A number of studies [2–5] have been proposed to support transition operations. However, in most cases, manual manipulations by human plant operators are still required owing to the limitations of conventional approaches.

Industrial chemical plants are influenced by several disturbances, such as changes in the weather, feed composition, and main steam pressure. Time invariant disturbances (e.g. additive Gaussian noises of fixed parameters) are considered in conventional MPCs, whereas responding to time variant disturbances, such as the ramp shape noise or a sudden step change in certain situations (e.g. heavy rain), remains a contemporary challenge [6].

To predict unlikely but structurally possible situations, dynamic simulators can appropriately reproduce and predict detailed non-steady states because they can use nonlinear and precise dynamical models built on standard chemical engineering knowledge and the structure of the target plant, for example, piping and instrumentation diagrams (P&ID). The vinyl acetate monomer (VAM) plant model [7] is one such plant simulator.

Reinforcement learning (RL) [8] can generate optimized operation procedures by solely adopting the inputs and outputs of the environment, that is, the target dynamical systems (e.g. dynamic simulators), and the final desired situations; thus, RL is convenient for...
optimizing operation procedures on dynamic simulators. RL optimizes a policy to achieve the desired situation provided as a reward in a given environment. A policy is a parametrized mapping from the observation space (e.g., a set of sensor value vectors, process values (PV), or states) to the action space (e.g., a set of setpoint value (SV) vectors), and rewards are awarded by the environment depending on the given preference configuration of the current state. An agent has a policy and updates the policy parameters to enhance its performance on a trial-and-error basis for the environment. In the field of process control, RL has been employed to tune PID parameters [9], and recently, RL has attracted attention as a potential MPC replacement [10]. The computation cost of RL for the online control on the real plant is significantly lower than MPC. The optimization process generally requires iterative and time-consuming computations. In particular, solving nonlinear optimization problems commonly requires vast amounts of computation compared to linear problems. The optimization process in RL (i.e., training) has been done prior to the deployment, whereas MPC performs the optimization process during the actual operation on the real plant. Therefore, RL has potential to address complex nonlinear control problems online.

Recently, an RL-based guidance system for transition operations has been proposed by the authors of this study [11]; however, this system cannot guide transition operations in time-variant disturbance situations such as abrupt heavy rain. Battery limit conditions do not change (heavy rain) on a dynamic simulator because these situations are usually unmodelled and measurements to reproduce detailed situations (e.g., thermometer, rain gauge, aerovane, and heliograph) are generally unavailable. In such cases, the control methods typically try to identify the internal states and model parameters using measurements available for an actual plant. However, conventional nonlinear system identification methods including particle filters [12] often require an impractically large amount of computational resources continuously while controlling. Consequently, rapid and automatic counter-disturbance actions remain challenging.

Because RL is a method to optimize a parametrized controller (policy) based on the behaviour of the target system, the behaviour of systems for training (the simulator) and for actual control (the real plant) should be ideally identical. In practice, behavioural gaps exist between them owing to multiple issues, including modelling errors on the simulators, incorrect state identifications, and unpredicted disturbances on the real situations. Modelling errors are typically inevitable owing to simplified and idealized descriptions of physical phenomena leveraged by the plant, omission of detailed structures in equipment, temporal changes in model parameters and battery limit conditions, disturbance, etc. Figure 2 shows the schematic of the gap problem in chemical plant control. The problem is commonly referred to as the simulation-to-reality (sim-to-real) gap [13,14]. The most straightforward solution is to prepare simulators that exactly reproduce real situations via system identification methods as mentioned above. Domain randomization [15] is a simple and common method that adapts models to the real world by randomizing simulations (e.g., randomizing simulation parameters) during training, to utilize experiences in a variety of situations. Situations similar to the real

Figure 1. Chemical distillation plant for the experiment. Feeding the methanol and water mixture, this plant separates them. This binary distillation process is common in the chemical industry.

Figure 2. Simulation-to-reality gaps in chemical plant control.
settings are expected to be included in the randomized simulation situations.

Herein, we expand on a previous study [11] to propose a method to construct a control system for the transition operations of chemical plants addressing sim-to-real gaps even in disturbance situations. We propose a bidirectional adaptation approach to close the gap by both reproducing the real situation on the simulator and applying the simulated and predicted situation on the real plant. To implement the approach, the control problem is divided into three tasks, namely (1) identifying the model parameters and current state, (2) optimizing the non-steady operations, and (3) letting the real situations close to the simulated and predicted situations by adjusting the control inputs under the gapped situations including modelling errors and disturbances. Each task is assigned to a reinforcement learning agent and trained individually. After the training, these agents are integrated and collaborate on the original objective. The contribution of this study is twofold: (a) the proposed method can adapt to the actual plant situations of time-dependent disturbances, and (b) the method was evaluated on an actual chemical distillation plant Figure 1, and the plant operated successfully as scheduled, even under heavy rain disturbance.

This paper is a revised version of the paper presented in the SICE 2021 annual conference [16]. We have added the background description of the sim-to-real gaps from the perspective of RL research in §1 and details of the method in §2. Additionally, in §4, we have discussed the proposed RL-based methods of system identification and disturbance rejection. In §4, we present an interpretation of the proposed RL-based system identification method as an offline optimization version of particle filtering (PF). PF is a common and powerful method for identifying the state of nonlinear systems online. A major drawback of PF is the vast amount of online computation required, as it searches the states using numerous predictions (i.e. simulations) of state candidates in parallel. Contrary, the proposed method optimizes (i.e. sharpens) the probability distributions of the current states conditioned on the current measurement and the previous state in the training phase of RL before the online estimation. Thus, the method requires significantly less online computation cost and is promising in estimating the states of nonlinear complex systems which require costly calculations for prediction in general. In §4, we also present an interpretation of the proposed disturbance rejection methods as classical control frameworks for disturbance rejection including Smith predictor control and internal model control. From these perspectives, the proposed disturbance rejection methods can be commonly employed for control problems of disturbance rejection other than chemical plant control.

2. Method

To address the simulation-to-reality gaps and realize non-steady control of chemical plants via RL, we propose a bidirectional adaptation approach implemented as a task dividing architecture. One direction of the approach is to copy the situation from the real plant to the simulator (real plant to simulator). To avoid unforeseen situations during operations on the real plant, the operation procedures of non-steady situations should be reviewed and authorized by human operators before performing them on the real plant; thus, the procedure should be generated and the resulting situations should be predicted on the simulator in advance. To generate realistic or reproducible procedures on the real plant, the initial situation on the simulator should be identical to the current real situation; thus, the precise estimation of the inner hidden states in the real plant and the identification of an optimal simulation model are necessary. The other direction is to copy the situation from the simulator to the real plant (simulator to real plant). To realize the simulated, predicted, and authorized situations on the real plant against modelling errors and abrupt disturbances, the previously generated procedures should be adjusted to minimize the current measured gaps during operation.

To implement the above bidirectional approach, we divided the control problem into three tasks, namely (1) system identification for improving simulated behaviour, (2) optimizing non-steady procedures, and (3) achieving simulated and predicted situation on the real plant by adjusting the procedures. Each task is assigned to an RL agent. Figure 3 shows the divided tasks and interactions between the real plant and three RL agents. Figure 4 illustrates the overall architecture of the method as a block diagram. The first agent for the system identification attempts to reproduce the real situation on the simulator by adjusting the simulation model parameters with time. The second agent for the procedure optimization generates an optimized procedure for the target non-steady operations initialized with the situation estimated by the first agent. After the human operators review and authorize the generated procedure, they begin by setting the proposed manipulation values for the actual plant one step at a time. The third agent for disturbance rejection attempts to bring the real situation to the simulated and predicted situation by adjusting the procedures generated by the second agent and authorized by the human operators. In this section, the function and task configuration of each RL agent are described individually.

2.1. System identification and state estimation

System identification is a family of methods for determining a simulation model that behaves exactly like the
Target system. The method typically employs parametric systems to determine the optimal parameters that can reproduce behaviour that is similar to that of the target system. Dynamical systems have internal hidden states, and the observable responses are calculated using the hidden states as observed in the state-space models; thus, to predict the system response, methods for estimating the internal states are required. In the field of the control theory, state observers, including Kalman filters [12,17], are conventionally employed for this purpose; however, this method requires the model to be linear and a provision of all the system parameters. Dynamic simulators are not limited to linear systems in general. Particle filters or sequential Monte Carlo methods [12] can estimate the states of nonlinear systems, including dynamic simulators with given distributions of state evolution and measurement noise, and can optimize system parameters because they utilize the system model to predict the next state and measurement. However, because particle filters calculate a number of simulation instances as “particles” in parallel, they require a large-scale computation environment for the entire period. The improvement of system identification is a common idea for addressing sim-to-real gaps, but requires the manual engineering of systems modelling [13] or involving challenging estimations of multiple parameters [14].

In this study, to estimate the optimal system parameters and simulation states that reflect the actual plant states, we employed a method inspired by tracking simulation [18]. Tracking simulation is a simple and practical method for estimating the actual-plant states on dynamic simulators. Figure 5 presents the schematic of tracking simulation. A single tracking simulator runs in parallel with the actual plant, that is, the control inputs set to the actual plant are also provided to the simulator simultaneously. If the simulator is modelled precisely, the response of the actual plant and simulation would be similar. The model parameters are updated with the simulation to improve reproducibility, that is, to minimize the residuals of the responses. Conventional methods employed traditional PID control to

---

**Figure 3.** Task division architecture for narrowing simulation-to-reality gaps for the bidirectional adaptation approach. The control problem is divided into three tasks.

**Figure 4.** Overall architecture of proposed system. Three RL agents (painted blocks) work together to optimize the transition procedure against disturbance.

**Figure 5.** Schematic of tracking simulation for estimating actual plant state and corresponding simulation parameters. Estimated states and parameters are employed as an initial condition of the transition procedure to be supported.
update the parameters, that is, the system parameters are adjusted based on the amount of residuals.

To update the parameters online, we adopt an RL agent for the parameter updater in Figure 5. The training setting is depicted in Figure 6. The agent employs the dynamic simulator

\[ f(\hat{s}_t, a_t; h_t) = \hat{s}_{t+1}, \]  

where \( \hat{s}_t, a_t, \) and \( h_t \) are the simulation states at time \( t \), control inputs (i.e. manipulation on the simulator), and simulation parameters (e.g. heat transmission parameters), respectively. The plant simulator \( f \) calculates the next state \( \hat{s}_{t+1} = f(\hat{s}_t, a_t; h_t) \) using the current situation. We first initialized the simulator by an appropriate state \( \hat{s}_0 \) prepared in advance and then the simulator calculated the time-series of the states \( \{\hat{s}_1, \hat{s}_2, \ldots\} \) using the control inputs at each time \( a_t \) according to the recoded trajectories (i.e. time series of control inputs and measured sensor values) of the actual plant.

The action of the agent is to set values for the simulation model parameters \( h_t \). The policy distribution is represented by the policy function \( \pi \) as expressed by

\[ p(h_t|\hat{s}_t, s_t, a_{t-1}; \pi) = \pi(h_t, \hat{s}_t, s_t, a_{t-1}, h_{t-1}). \]  

The policy function is usually parametric and the parameters are updated through the training. The reward function is based on the residuals between the actual response \( s_t \) and the simulation response \( \hat{s}_t \) such as

\[ r(\hat{s}_t) = -||s_t - \hat{s}_t||_2. \]  

During the training, the agent attempts to minimize the residual by adjusting the simulation parameters \( h_t \) at each time step. In the case of the experiments presented in this paper, \( h_t \) is the vector of heat-balance parameters of the distillation tower (heat transfer coefficients from the tower to the atmosphere). Given the actual measurements of the tower temperature profile (a set of sensor readings of the thermometers attached to the real tower) and the simulated temperature profile, the agent tries to minimize the temperature gaps by manipulating \( h_t \). If the agent increases \( h_t \), the amount of heat removed from the simulated tower increases and its temperatures become lower than before given other manipulation points are untouched. During the training on the simulator, the agent is expected to capture the relationship between the parameter changes and plant state changes by trial manipulations, resulting situations and its evaluations. After the training, the agent can estimate the current optimal parameters and current plant state using the latest plant trajectories.

### 2.2. Optimized procedure generation

The procedure generation agent generates the optimized procedure for each desired plant state given by the plant operator. Figure 7 illustrates the training setup for the agent. The policy distribution is

\[ p(\hat{a}_t|\hat{s}_t; \pi) = \pi(\hat{a}_t, \hat{s}_t) \]  

where \( \hat{a}_t, \hat{s}_t \), and \( \pi \) are the action values (SVs), response values at time \( t \) on the simulator, and policy function, respectively. We prepare a reward function that returns a maximal value if the state satisfies the given operation objective, e.g. in the case of the objective “set the product purity to 95%,” we can configure the reward function as:

\[ r(\hat{s}_t) = -|0.95 - \hat{s}_t^{\text{(purity)}}| \]  

where \( \hat{s}_t^{\text{(purity)}} \) represents the product purity calculated on the simulator at time \( t \).

While training the agent, we employed a domain randomization technique to handle the agent in varied situations that may occur on the actual plant. Accordingly, the initial states and parameters for the simulator are randomly selected and applied during the training. The agent optimizes the procedure as well as the target detailed desired situation, for example, the agent uses the simulator to specify the desired tower temperatures depending on the desired production purity.
2.3. Trajectory tracking control

To realize the planned trajectories (i.e. the predicted simulation response based on the generated procedure beginning with the estimated current plant states) on the actual plant, we adopted two adaptation methods to correct the control inputs and cancel the residuals triggered by the disturbance and modelling errors. Both methods can follow the trajectories of multiple control target variables using multiple control inputs (i.e. they can control multiple-input-multiple-output systems).

2.3.1. Gradient-based adaptation

One method for realizing the reference trajectory is based on dynamic simulation. If we assume that the response of the real plant is still similar to that predicted on the simulator regardless of the existence of residuals, we can estimate the control input that can achieve the desired process value at each time. Figure 8 illustrates the correction concept using the simulation responses. The x and y-axes represent the setpoint (i.e. control input) and process (i.e. response) values, respectively. We assumed that the time interval between the continuous time steps is fixed and therefore omitted the time axis. The red cross-marked point indicates the current plant state, whereas the blue cross-marked point indicates the desired predicted process value. In this case, we tried to increase the process value. If we calculate the simulated process value of each time step based on the predicted control input (i.e. output of the generation agent) \( SV_{\text{Planned}} + \DeltaSV \) in addition to the response of the \( SV_{\text{Planned}} \) on the simulator, we can approximate the gradient of the response \( \DeltaPV/\DeltaSV \). Using this gradient and residual of the response, we can correct the control input as:

\[
SV_{\text{Corrected}} = SV_{\text{Planned}} + \frac{\DeltaPV}{\DeltaSV} (PV_{\text{Planned}} - PV_{\text{Actual}}).
\]

We can perform the correction on each time step such that the process values become closer than before. The concept of this method is simple; however, \( \DeltaSV \) needs to be manually tuned and the assumption does not always hold when the plant is in a non-steady state.

2.3.2. Learning-based adaptation

Another method for cancelling the residuals relies on the use of an RL agent. Figure 9 presents the configuration of the agent training. The agent observes the planned trajectories and simulation states. The action of the agent is to set the values \( \Delta a_t \) added to the generated control inputs \( \tilde{a}_t \) by the procedure generation agent in advance at time \( t \). The policy distribution is

\[
p(\Delta a_t | \tilde{s}_t, \tilde{a}_t, \hat{s}_t; \pi) = \pi(\Delta a_t, \tilde{s}_t, \tilde{a}_t, \hat{s}_t)
\]

where \( \tilde{s}_t, \hat{s}_t, \) and \( \pi \) are the actual measurements, predicted response by the procedure generation agent, and policy function, respectively. The action \( a_t + \Delta a_t \) would be set to the actual plant. The reward function is configured to output a maximal value if the simulation response is identical to the planned trajectories (i.e. minimize the residuals), for instance,

\[
r(\hat{s}_t) = -\|\hat{s}_t - \hat{s}_t\|_2,
\]

where \( \hat{s}_t \) is the planned response according to the trajectories generated by the procedure generation agent, and \( \hat{s}_t \) is the simulation response.

While training the agent, the simulation parameters are altered randomly to emulate the disturbance situations. In this training, the agent is expected to capture the relationship between the direction of adjustments and the resulting situation (e.g. if the SV of steam is increased more than the planned value, the tower temperature will be increased more than before). The randomized parameter values are not provided for the
agent (i.e., the randomized parameters are not included in the observable states); therefore, the agent attempts to reduce the residuals without considering the reason for the situation change. The advantages of this training setting is its ability to omit the process of investigating the cause and immediately responding to ongoing situations (i.e., the increase of residuals). After training, the agent is expected to have the ability to correct the control input in a variety of situations such as the plant state and disturbances.

When the agent is applied to the actual control on the plant site, the simulator in the training setting is replaced with the actual plant. The agent then calculates the corrected control input based on the actual plant state and planned trajectories provided by the procedure generation agent.

3. Experiments

3.1. Target plant

We evaluated our method on a real binary distillation plant built for training human operators. The plant is administrated by Mitsui Chemicals, Inc. and is illustrated in Figure 1. The target plant continuously operates to separate a methanol-water mixture. Figure 10 depicts the P&ID of the plant. This plant consists of one distillation tower, one reflux drum, one reboiler, one preheater, and two coolers, which is a standard design for a binary distillation process.

The plant is equipped with seven PID controllers (indicated by the circles in Figure 10) on the distributed control system (DCS) to maintain the feed flow, steam flow of the reboiler, steam flow of the preheater, reflux drum level, reflux flow, top product flow, and tower bottom level. The top product flow is usually controlled to maintain the reflux drum level (i.e., cascaded to the level controller), whereas the bottom product flow is controlled to maintain the tower bottom level. If we do not change the plant load, the feed flow is fixed, and the temperature of the preheater outlet is fixed. Therefore, the manipulation points for normal operation during production are the remaining two controllers of the reboiler steam and reflux flows. We employed the two SVs of the PID controllers as the control inputs in these experiments.

Furthermore, we utilized the corresponding dynamic simulator, which modelled the entire plant and was developed on Visual Modeller, a product of Omega Simulations Co., Ltd., as well as the VAM plant simulator [7]. The plant simulator was originally installed in the plant for training human operators. We upgraded the simulation model before employing it in this study. We manually selected the two simulation parameters to be updated online by the identification agent, which are the heat-balance of the distillation tower as a whole and the bottom part of it. We adopted proximal policy optimization (PPO) [19] for the three RL agents implemented on the ChainerRL [20] library. To connect the agent to the simulator, we developed interface modules in compliance with the OpenAI Gym [21] interface. The three RL agents were trained before the plant experiments. The agents manipulated the simulators every 5 min.

3.2. Operation scenario

We evaluated our method on the two operation scenarios required to maintain a steady state and change the product grade (purity). Both operations were performed under disturbance situations. Maintaining a steady-state despite heavy disturbances is difficult but possible during operation, and changing the product grade is a relatively common transition operation. For these experiments, we set the disturbance situation as heavy rain. To reproduce the situation with the real plant, we sprayed water from the roof of the distillation tower at a constant flow rate over a given short period during operation. Notably, almost all the components of the chemical plants are surrounded by thermal insulation to reduce the effect of external situations, whereas the target plant has a relatively small-scale and is sensitive to weather changes. Generally, product purity is highly dependent on the tower temperatures of the outlet stages (i.e., the top and bottom stages on the standard binary distillation case); therefore, we employed the target control variables as the tower temperatures. We imposed constraints for both operations to maintain the bottom product purity as lower than 1% to ensure that the material is economically beneficial. We set the agent-proposed SVs on the DCS every 5 min.

3.3. Steady-state disturbance

We commenced the experiment at a steady state of 100% load production, and the product purity was higher than 99%, which is the normal plant state. The objective of this experiment was to maintain the steady state, that is, to maintain a product purity higher than 99% despite the rain disturbance. Subsequently, to

![Figure 10. P&ID of the target distillation plant.](image-url)
emulate the heavy rain disturbance, we sprayed water for 30 min. The system automatically corrected the setpoint values to minimize the residuals between the original temperature profiles and current actual temperatures during the entire period of the experiment. We employed the gradient-based adaptation method (see § 2.3.1) in this experiment.

Figure 11 presents the time-series of the tower temperature profiles. The solid lines indicate the actual measured values, whereas the dashed lines represent the simulation values estimated by the system identification agent. The maximum value of the residuals are lower than 5 °C even in the fluctuating non-steady situations due to the rain disturbance. Figure 12 shows the time-series of the optimal values of heat-balance parameters during the operation. Both values increase after the spraying starts. The system identification agent does not know when to spray; however, the agent adjusts the parameters adaptively.

Figure 13 presents the setpoint values of the two PID controllers. The blue and yellow lines indicate setpoint values of the reboiler and reflux, respectively. The dashed and solid lines indicate the fixed original setpoint values and corrected values, respectively. A short period after the onset of spraying, the system increases the setpoint values of the reboiler steam. As the temperatures recover to their original states, the system decreases the reboiler setpoint value to a point less than the original value. Figure 14 shows the time-series of the top and bottom product purity. The blue dots indicate the sampled and measured purities. The system successfully recovered the original steady state and maintained a bottom product purity lower than 1% and a top product purity greater than or equal to 99%. Therefore, the agent satisfied the purity constraints.

Figure 12. Estimated system parameters of heat-balances, or heat transfer to outside the system (i.e. air) of total tower (blue) and tower bottom (red) by the proposed system identification method. After beginning of the heavy rain disturbance, both the values increase (i.e. cooling down the tower on the simulator) corresponding to the drop of the actual tower temperatures. As the actual heat transfer area of the tower is not rigorously identical to that of the simulator, these parameters shall be considered as relative values (i.e. they indicate dynamical changes in heat balance rather than precise absolute values).

Figure 13. Original setpoint (dashed lines) and corrected (i.e. actually performed) (solid lines) values. Reboiler steam (blue lines) and reflux (yellow lines) flow. After a while of the heavy rain disturbance, the proposed disturbance rejection method increased reboiler steam rapidly in responding to the temperature drop. The adjusted manipulation of steam (solid blue line) achieved the ceiling of the normal operation range since the amount of temperature drop triggered by the heavy rain disturbance was enormous. After a while of terminating the disturbance, the adjusted steam flow returned back to around the original value since the tower temperatures recovered. In this experimental case, the proposed method maintained reflux flow because the temperature drop occurred mostly on the bottom side of the tower and changes in reflux tend to affect top side temperatures of the tower.

This result indicates that the system maintained the top and bottom temperatures.

3.4. Grade change under disturbance

The objective of the grade change operation was to change the product purity to the desired value. We began the experiment at a weak steady situation where the purity was approximately 95% (see Figure 17 at 13:00), with slightly increasing tower temperatures (see Figure 15 at 13:00). We set the target purity as 99% and above. The system estimated the plant state and proposed the optimized procedure for the grade initialized
by the estimated plant state. After starting the procedure on the real plant, the rain disturbance occurred for 10 min. The system corrected the proposed control inputs in response to the tower temperature drop triggered by the disturbance. We employed the learning-based adaptation method (see §2.3.2) in this experiment.

Before starting the transition operation, the system identification agent estimated the plant state online. The procedure generation agent generated the optimized grade change procedure based on the estimated initial state and system parameters and proposed the procedure to the human plant operators with the predicted plant response. After the operators reviewed and authorized the procedure, we began the procedure on the actual plant. During this procedure, the tower temperature suddenly dropped, owing to the spraying disturbance. To eliminate the disturbance and achieve the predicted temperatures at each instance, the adjusting agent corrected the initial plan at each step.

Figure 15 presents the time-series of the tower temperature profile. At the beginning of the operation, the estimated temperatures are lower than the actual ones. After the sprinkle, the temperature dropped for a while; however, the temperature recovered quickly after the drop.

Figure 16 presents the setpoint values of the two PID controllers. The dashed and solid lines represent the originally proposed setpoint values and the corrected and actually applied setpoint values, respectively. Shortly after the spraying, the system increased the setpoint values of the reboiler steam (to heat up) and significantly decreased the reflux (to decrease coolant); consequently, the tower was effectively heated. As the temperatures recovered to their original states, the adjusting agent reduced the reboiler steam, while maintaining a higher reflux flow than the originally planned value. Figure 17 shows the time-series of top product purity. A top purity of 99.0% was finally achieved,
whereas the bottom purity remained undetected. The system successfully responded to the disturbance and realized a top product purity of 99%, as scheduled.

4. Discussion

4.1. System identification

In this study, an RL-based state and model parameter identification method has been introduced (see §2.1). From the perspective of particle filters [12], this RL-based method can be regarded as a pre-trained version of it. In this section, we first briefly review the particle filtering and then describe the interpretation of the method as a version of it.

Particle filtering is a stochastic and recursive method for estimating unmeasured states of a nonlinear dynamical system from the measurements. A dynamical system is considered a stochastic model involving a couple of probability distributions, namely the state evolution distribution $p(x_k|x_{k-1})$ and the measurement distribution $p(z_k|x_k)$, where $x_k$ and $z_k$ are the unmeasured state and measurement of time $k$, respectively. The task of the particle filtering is to estimate the unmeasured state $x_t$ from the given sequence of measurement $\{z_1, z_2, \ldots, z_t\}$. In the presented experiments, the state $x_t$ and measurements $z_t$ correspond to the heat-balance parameters and observable states including measured temperatures of the real plant, respectively.

A description of a generic particle filter algorithm is presented in Algorithm 1, where $q(x_k|x_{k-1}, z_k)$, $L(z_k, x_{k}^{l}, x_{k-1}^{l})$, $N_s$, and $N_T$ are the importance distribution, the likelihood function, given the number of candidates at each time step, and given the minimum number of reliable candidate states, respectively. In the particle filter, $N_s$ states $\{x_{k}^{l}\}_{l=1}^{N_s}$ are randomly sampled as particles, or candidates of the estimated states at time $k$.

The importance distribution $q$ is used for the sampling. The optimal importance distribution $q$ is the target distribution

$$p(x_k|x_{k-1}, z_k) = \frac{p(x_k|x_{k-1})p(x_{k-1}|z_k)}{\int p(x_k|x_{k-1})p(x_{k-1}|z_k)dx_k}, \quad (9)$$

However, its computation is generally impractical owing to the integral; thus, $q$ is commonly approximated by the state evolution distribution $p(x_k|x_{k-1})$ and sampling is replaced by the calculation of the next state $x_k$ using the state-space model and the assumed noise distribution. Each particle is then evaluated and assigned a likelihood by the likelihood function $L(z_k, x_{k}^{l}, x_{k-1}^{l})$ according to the measurement $z_k$. The likelihood function typically consists of the state evolution distribution, the measurement distribution, and the importance distribution as follows:

$$L(z_k, x_{k}, x_{k-1}) = \frac{p(z_k|x_k)p(x_k|x_{k-1})}{q(x_k|x_{k-1}, z_k)}, \quad (10)$$

Based on the evaluation, the particles assigned low likelihood are considered ineffective for state prediction; thus, they are filtered out from the candidates by resampling and proceed to the next time step $k + 1$. The resampling procedure is employed to avoid performance degeneracy owing to scattering of the particles (i.e., increase in variance) and concentration of the weight on the single most likely particle, which results in losing the opportunity to find better states proximal to the estimated most likely state. Similarly, as the number of particles considered in each time step $N_s$ significantly increases, the estimation performance improves. However, the increased particles require a vast amount of parallel prediction of state evolution i.e. simulation; thus, the particle filter is considered impractical for online system identification if the simulation itself involves costly computations.

Algorithm 1 Generic particle filter

**Input:** Sequence of measurement $\{z_1, z_2, \cdots, z_t\}$

**Output:** Estimated current state $\hat{x}_t$

1. for $k = 1$ to $t$ do $\triangleright$ Recursively estimate states
2. for $i = 1$ to $N_s$ do $\triangleright$ Predict and evaluate candidate states in parallel
3. Sample a state $x_{k}^{i} \sim q(x_{k}^{i}|x_{k-1}^{l}, z_k)$ as a particle
4. Predict and evaluate the state $w_k^i \leftarrow L(z_k, x_{k}^{i}, x_{k-1}^{l})$
5. Select the current best estimation for output
6. Normalize each weight $w_k^i \leftarrow w_k^i / \sum_i w_k^i$
7. Compute the number of the effective samples $N_e = 1 / \sum_i (w_k^i)^2$
8. if $N_e < N_T$ then $\triangleright$ Resample if the effective particles become too few
9. Sample a set $\hat{X}_k$ of $N_s$ states from $\{x_{k}^{i}\}_{i=1}^{N_s}$ proportional to each $w_k^i$
10. Replace $\{x_{k}^{i}\}_{i=1}^{N_s}$ with $\hat{X}_k$ $\triangleright$ Filter out unlikely particles

The generic particle filter algorithm (Algorithm 1) can be regarded as an online optimization method of the states for maximizing the likelihood, or minimizing the residual between the measurement $z_k$ and estimated measurement $\hat{z}_k$. Meanwhile, the proposed RL-based system identification method can be regarded as an offline optimization method of importance distribution $q(x_k|x_{k-1}, z_k)$ for the sampling states. The trained policy of the proposed method in Equation (2) represents the optimized distribution $\hat{q}(x_k|x_{k-1}, z_k) \equiv p(x_k|x_{k-1}, z_k)$ and the agent can select the most likely state $\hat{x}_t = \arg\max_i \hat{q}(x_{t}|x_{t-1}, z_t)$ in online use. The common approach in the particle filtering and proposed RL-based method is to approximate posterior
distribution \( p(x_k | x_{k-1}, z_k) \) by sampling; however, their outputs are different.

Particle filters output the sequence of estimated states, which is generated using locally sampled states around the sequence. However, the proposed method outputs the estimated posterior distribution as policy, which is generated using globally sampled states by searching on the state space, and outputs the sequence of estimated states using the estimated posterior distribution in use.

The proposed RL-based optimization algorithm for the state estimation policy leveraging PPO in actor-critic style [19] is presented in Algorithm 2. The algorithm optimizes the policy distribution \( p(x_k^i | x_{k-1}^i, z_k^i; \pi_\theta) \) represented by the policy function \( \pi_\theta \) using a training environment (i.e. simulator) \( \hat{f} \) and a reward function \( r \) as well as a dataset of the actual measurements \( \{(z_k^i|_{k=1}^{Nd})_{i=1}^{Ns}\} \), where \( \theta, Nd, Nr, Ns, T, \) and \( K \) are the parameter of the policy function, the number of training sequences, iterations, simulators available in parallel, time steps in a sequence, and epochs for policy update using the same samples, respectively. The environment predicts the next measurement \( \hat{z}_{k+1} \) using the current measurement \( z_k \) and estimated current state \( \hat{x}_k \). A reward function (e.g. Equation (3)) is used for the evaluation of the current state similar to the likelihood function in the particle filter. The objective of RL is to maximize the cumulative reward during the total sequence including future states; thus, the trained policy can be robust in temporary measurement errors. After training, the optimized policy distribution \( p(x_k^i | x_{k-1}^i, z_k^i; \pi_\theta) \) is used for the online estimation of the current state \( \hat{x}_k^i \) using the single simulator, that is, the hyperparameters for the training is set to \( N_r = N_s = 1 \) in Algorithm 2. Notably, if the size \( N_r \) of the dataset is insufficiently small, artificially generated sequences of measurements leveraging the simulator and the domain randomization can be adopted as the dataset.

Similar to the degeneracy problem in the particle filter (i.e. dependence on the most likely single particle), the high-variance problem (i.e. overfitting) often becomes an issue in RL methods. Both of them share the situation that a small number of sample instances mostly affect the optimization result. To address the problem, PPO leveraged the advantage estimator \( \hat{A} \) and introduced the clipped surrogate objective to reduce the influence of each particular instance in optimizing the policy.

The sampled particles are unshared among the different sequences in particle filters, whereas the proposed method reuses samples for multiple state estimations as the training data. Therefore, the proposed method has potential to achieve better sample efficiency than the particle filtering.

### Algorithm 2: Training of the state estimation policy using PPO [19]

| Input: Training environment \( \hat{f} \), reward function \( r \), and a dataset of sequences of measurement \( \{\{z_k^i|_{k=1}^{Nd}\}_{i=1}^{Ns}\} \) |
| Output: Optimized policy function \( \pi_\theta \) |
| 1: for \( j = 1 \) to \( N_r \) do \( \triangleright \) Train the policy repeatedly |
| 2: for \( i = 1 \) to \( N_s \) do \( \triangleright \) Collect \( T \) time steps of \( z_k^i \)
| 3: Randomly select sequence index \( d \) from \( \{1, 2, ..., N_d\} \)
| 4: for \( k = 1, 2, ..., T \) do \( \triangleright \) Collect \( T \) time steps of instances
| 5: \( z_k^i \leftarrow s_k^d \)
| 6: Sample a state \( x_k^i \) from \( p(x_k^i | x_{k-1}^i, z_k^i; \pi_\theta) \)
| 7: Predict the next measurement \( \hat{z}_{k+1}^i \) using \( \hat{f}(\hat{z}_{k}^i, x_{k}^i) \)
| 8: Evaluate the state \( \hat{r}^{i}_{k+1} = \hat{r}(\hat{z}_{k+1}^i, \hat{x}_{k+1}^i) \)
| 9: Compute advantage estimates \( \hat{A}_t^i, \ldots, \hat{A}_T^i \) using collected states and rewards
| 10: \( \theta_{old} \leftarrow \theta \)
| 11: Optimize policy loss \( L \) wrt \( \theta \) using collected instances and \( \theta_{old} \) with \( K \) epochs \( \triangleright \) Update policy parameter |

As regards the aspect of required models, the proposed method uses less information on the model than the particle filter. The proposed method only uses the simulated and given actual measurement, whereas the particle filter uses the model distributions or the state-space models besides the given actual measurement. Therefore, particle filters cannot be adopted on a system if its internal models are inaccessible, but the proposed method can be adopted.

Compared to the particle filters, the proposed method has the following advantages:

1. It can significantly reduce online computation because single simulation is sufficient.
2. It requires only measurements, not the internal models of the system.
3. Future changes in states can be considered on the estimation.

However, there are also some drawbacks including

1. Training can be time-consuming.
2. Adjusting hyperparameters are required.
3. Developing interface module to connect RL agents and simulators is necessary.

In spite of the drawbacks, the RL-based identification method is promising because it pioneers practical online system identification of complex nonlinear dynamical systems, to the best of our knowledge.
In light of the identification ability, the proposed method can address “regular” models as well as PF. In regular models, a model prediction corresponds to a single set of parameters, that is, multiple different sets of parameters do not result in an identical model prediction. In the presented experiments, we selected two parameters to be identified, namely, the heat-balance parameter of the total tower and that of the bottom part. In this case, the agent can adjust the heat transfer of the bottom and top parts of the tower distinctively; thus, a heat loss situation is triggered by only one parameter set. By contrast, if we add air temperature to be identified, multiple sets of parameters could trigger similar predictions. Increase of the air temperature and decrease of the total tower parameter (heat loss) may result in similar predictions. In the latter case, the identified values are unreliable because there are multiple optimal values and we have no criterion to choose between them. Additionally, in that case, the search space (the number of degrees of freedom) during the training becomes unnecessarily large and the training speed would be slowed down.

The proposed method is expected to predict parameters effectively if the target parameters are orthogonal (i.e. mutually independent). In the field of chemical engineering, for instance, energy (e.g. heat) balance and material balance are the basic concepts. In the presented experiments, we focused on heat-balance, that is, the fact that the amounts of the input and output heat of a system are identical (balanced); thus, the tower temperature can be adjusted by manipulation of the tower output heat. Heat balance of other pieces of equipment (e.g. preheater, cooler, and drums) would also be identifiable. Similarly, material-related parameters such as raw material feed composition (e.g. methanol purity of the raw material feed) would also be identifiable. Other mutually independent parameters are also expected to be effectively identified. We plan to investigate the identifiability of the method in various situations and domains other than chemical engineering.

4.2. Disturbance rejection

In conventional MPCs, the existence of a time invariant disturbance is considered; however, if the disturbance depends on time, the predictions of the controlled variable will not be optimal. Stochastic MPC (SMPC) [22] is a method that addresses this disturbance challenge. SMPC extends MPC by introducing stochastic optimal control. It considers the prediction errors (i.e. the disturbances and state variables) as stochastic variables and maintains the probability of satisfying the constraints at levels greater than the given threshold level during optimization.

![Figure 18. Block diagrams of (a) Smith predictor control (upper) and (b) internal model control (lower) commonly adopted for disturbance rejection in the field of classical control.](image)

The practical or conventional methods of both MPC and SMPC are based on linear models and solving the optimization problem for each state; therefore, they can ensure the optimality of the output procedure. Regarding RL, the optimality of the procedure depends on the “sufficiency” of the training, whereas RL is capable of optimising the procedures for nonlinear systems as performed in our experiment.

In the field of classical control, Smith predictor control Figure 18 (a) and internal model control (IMC) Figure 18 (b) have been adopted for disturbance rejection [23]. In Figure 18, r, u, d, and s are the reference input, control input, disturbance, and output, respectively. Both methods leverage the nominal plant $\hat{P}$ (i.e. the simulator) which is the plant model without considering the uncertainties including the disturbances. They both attempt to minimize the residual between the responses of the nominal and actual plant. IMC explicitly uses the inverse model of the plant $C_{IMC}$, which identifies the appropriate control input corresponding to the current desired output. These methods can address the undesired residual owing to both the disturbances and modelling errors to some extent. To construct these control systems, the manual design of the nominal model and controller considering the uncertainties is necessary.

In contrast, the proposed gradient-based method (see § 2.3.1) can be regarded as a simple approximation of the inverse model of the IMC in the time domain. The proposed RL-based method (see § 2.3.2) can be regarded as a unified feedback controller consisted of the controller and nominal model in the Smith predictor control. The simulator is unincorporated in the RL-based controller actually; however, the behaviour of the controller (i.e. policy) deeply depends on the dynamics of the simulator used for the training. The predicted response of the procedure generation agent can be seen as the output of the nominal plant $\hat{P}$. Additionally, the RL-based method only requires the simulator and distribution of disturbances.
for domain randomization in the training and can construct the controller automatically. Notably, both proposed methods can address multiple-input-multiple-output systems, whereas the classical control methods are limited to control single-input-single-output systems.

The proposed method can learn and leverage basic dynamics such as qualitative or directional relationships between the manipulations and responses (e.g., opening the valve on a water pipe results in an increase of the water flow); thus, it can address various situations if the target systems are working normally. Additionally, if the method cannot reject the disturbance in a reasonable period of time, it would imply the occurrence of an abnormal situation (e.g., leakage); therefore, the method could also be employed for monitoring the soundness of the system.

### 4.3. Prescribable disturbance and failure

In the presented experiments, we employed heavy rain as the disturbance, but the target disturbance of the method is not limited to heat-related phenomena. The VAM plant model [7] provides disturbances including feed composition change, feed pressure change, main steam pressure change, and main steam temperature change. If the target measurable states to be achieved and variable parameters are provided, the method should be effective. For instance, in the case of a methanol plant, abrupt changes in feed composition would also be addressed by our method. The changes in feed methanol purity would affect the tower temperature profile and product flows; however, the product purities can be maintained if the temperatures are maintained. The relationship between feed composition, tower temperatures, and product composition are rather common in binary chemical distillation; therefore, maintaining the tower temperature against multiple kinds of disturbances can be considered as a generally required functionality. Disturbances on other equipment or processes can also be handled by the proposed method if the disturbed situations can be reproduced by the simulator and there exist some certain appropriate manipulations (potential solutions).

In light of the application field of the proposed RL-based method, it can contribute to control problems in domains other than chemical plants facing external disturbances involving changes in both natural and social situations including the control of robots, vehicles, and social infrastructure systems. The remaining issue involves addressing undesirable situations that can be simulated but are not considered in advance. Additionally, the handling of difficult-to-simulate situations such as process leakage requires further investigations.

### 4.4. Scalability to manufacturing plant

Chemical plants that include several processes involve a large number of manipulation points and measurements. In RL training, increasing the number of manipulation points significantly deteriorates the performance owing to the exponential growth of the candidate combinations of manipulations. To overcome the dimensionality problem, a previous study [11] introduced qualitative reasoning utilizing the plant structure to specify the effective manipulation points. Additionally, standard operation procedure (SOP) manuals are defined for each plant; therefore, specifying manipulation points that are compliant with the SOP is another practical and acceptable method for the human operators.

### 5. Conclusion

We developed a method for the control of a chemical plant with non-steady operations (e.g., product purity change) under disturbance situations. The method addresses the sim-to-real gaps, which are common in RL-based methods, by introducing the bidirectional narrowing approach. Based on the desired situations (e.g., a product purity of 99% and above) by the plant operators, our system proposes the optimized trajectories for achieving these situations, starting from identifying the current plant state to realizing the trajectory on an actual plant subject to the disturbances in the external environment by correcting the planned control inputs.

Considering an actual chemical distillation plant, we experimentally demonstrated that our system successfully realized the operation objective of changing the product grade, by cancelling the effects of the emulated disturbances of the heavy rain.

In future work, we plan to develop methodologies to support human plant operators responsible for non-steady operations. We reckon that it would be more practical to develop assistance systems that propose appropriate procedures in response to inquiries from human operators, rather than develop fully automatic control systems.

To propose the procedures to be performed by the operators in the actual plant, the reliability and acceptability of the procedures to human operators are becoming more important than in the case of fully automatic systems. Therefore, we aim to further the empirical studies to improve the interfaces between the control systems and operators.

### Acknowledgments

This work is a result of the collaborative research between AIST, Mitsui Chemicals Inc., Omega Simulation Co., Ltd., and NEC Corporation. The authors are grateful for the helpful discussion on the needs to support human plant operators.
and the experiments from Yasuo Fujisawa, Toshihide Kihara, Masahiko Tatsumi, Masanori Endo, Atsushi Uchimura, and Norio Esaki (Mitsui Chemicals), and for the useful advice about the design of the system architecture for the experiments and the modeling of the chemical process from Gentaro Fukano, Tsutomu Kimura, Akihiko Imagawa, Takayasu Ikeda, Yasuhiro Kamada, and Naoki Ura (Omega Simulation).

**Disclosure statement**

No potential conflict of interest was reported by the author(s).

**Notes on contributors**

Shumpei Kubosawa received his BA degree in Social Sciences from Hitotsubashi University in 2011, and ME from Tokyo Institute of Technology in 2013. Currently, he is a Special Researcher at Data Science Research Laboratories in NEC Corporation, and a Specialist for Specific Subject (Research) at NEC-AIST AI Cooperative Research Laboratory, AIST. His current research interests include artificial intelligence, natural language processing, and reinforcement learning. He is a member of SICE, SCE, etc.

Takashi Onishi received the B.E. and M.E. degrees from the University of Tokyo in 2004 and 2006, respectively. Currently, he is a Research Manager at Data Science Research Laboratories in NEC Corporation, and a Project Manager at NEC-AIST AI Cooperative Research Laboratory in AIST. His research interests include applications of reinforcement learning.

Yoshimasa Tsuruoka received his Doctor of Engineering degree from the University of Tokyo in 2002. He worked as a research associate at the University of Tokyo and at the University of Manchester. He was an associate professor at the Japan Advanced Institute of Science and Technology from 2004 to 2006, respectively. Currently, he is a Project Manager at NEC-AIST AI Cooperative Research Laboratory in AIST. His research interests include applications of reinforcement learning.

**ORCID**

Shumpei Kubosawa @ http://orcid.org/0000-0002-5729-800X

**References**

[1] Garcia CE, Prett DM, Morari M. Model predictive control: theory and practice – a survey. Automatica. 1989;25(3):335–348.

[2] Prata A, Oldenburg J, Kroll A. Integrated scheduling and dynamic optimization of grade transitions for a continuous polymerization reactor. Comput Chem Eng. 2008;32(3):463–476.

[3] McAuley B, MacGregor J. Optimal grade transitions in a gas phase polyethylene reactor. AIChE J. 1992;38(10):1564–1576.

[4] Mahadevan R, Doyle III F, Allcock A. Control-relevant scheduling of polymer grade transitions. AIChE J. 2002;48(8):1754–1764.

[5] Tousain L, Bosgra O. Efficient dynamic optimization for nonlinear model predictive control – application to a high-density poly-ethylene grade change problem; Proc. of the 39th IEEE Conf. on Decision and Control; 2000.p. 1.

[6] Qin J, Badgwell T. A survey of industrial model predictive control technology. Control Eng Pract. 2003;11(7):733–764.

[7] Machida Y, Ootakara S, Seki H, et al. Vinyl acetate monomer (VAM) plant model: A new benchmark problem for control and operation study. IFAC Pap Line. 2016;49:533–538.

[8] Sutton R, Barto A. Reinforcement learning: an introduction. Cambridge, MA: MIT Press; 1998.

[9] Howell M, Best M. On-line PID tuning for engine idle-speed control using continuous action reinforcement learning automata. Control Eng Pract. 2000;8(2):147–154.

[10] Badgwell T, Lee J, Liu KH. Reinforcement learning – overview of recent progress and implications for process control. Comput Aided Chem Eng. 2018;44:71–85.

[11] Kubosawa S, Onishi T, Tsuruoka Y. Computing operation procedures for chemical plants using whole-plant simulation models. Control Eng Pract. 2021;114:104878.

[12] Arulampalam M, Maskell S, Gordon N, et al. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. IEEE Trans Signal Process. 2002;50(2):174–188.

[13] Tan J, Zhang T, Coumans E, et al. Sim-to-real: learning agile locomotion for quadruped robots. Robotics: Science and Systems (RSS). 2018.

[14] Zhao W, Queralt J, Westerlund T. Sim-to-real transfer in deep reinforcement learning for robotics: a survey. 2020 IEEE Symposium Series on Computational Intelligence (SSCI); 2020. p. 737–744.

[15] Tobin J, Fong R, Ray A, et al. Domain randomization for transferring deep neural networks from simulation to the real world. 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS); 2017. p. 23–30.

[16] Kubosawa S, Onishi T, Tsuruoka Y. Non-steady state control under disturbances: navigating plant operation via simulation-based reinforcement learning. SICE 2021: 60th Annual Conference of the Society of Instrument and Control Engineers of Japan, Tokyo, 8–10 Sept; 2021. p. 796–803.

[17] Bishop G, Greg W. An introduction to the Kalman filter. Proc SIGGRAPH, Course. 2001;41(8):27599–23175.

[18] Friman M, Airikka P. Tracking simulation based on PI controllers and autotuning. IFAC Proc. 2012;45(3):548–553.

[19] Schulman J, Wolski F, Dhariwal P, et al. Proximal policy optimization algorithms. Preprint 2017. Available from: arXiv: 1707.06347.

[20] Fujita Y, Nagarajan P, Kataoka T, et al. ChainerRL: a deep reinforcement learning library. Workshop on Deep Reinforcement Learning at the 33rd Conference on Neural Information Processing Systems; 2019.

[21] Brockman G, Cheung V, Pettersson L, et al. OpenAI Gym. Preprint 2016. Available from: arXiv:1606.01540. 2016.

[22] Moshar A. Stochastic model predictive control: an overview and perspectives for future research. IEEE Control Syst. 2016;36(6):30–44.

[23] Abe N, Yamanaka K. Smith predictor control and internal model control – a tutorial. In SICE 2003 proceedings SICE annual conference 2003 in Fukui: 4–6 Aug, 2003; Vol. 2. p. 1383–1387.