A Reinforcement Learning Approach for Simultaneous Generation, Design and Control of Reaction-Separation Process Flowsheets

Simone Reynoso-Donzelli, Luis A. Ricardez-Sandoval*

Department of Chemical Engineering, University of Waterloo, ON N2L 3G1, Canada

**Corresponding author e-mail: laricard@uwaterloo.ca*

Abstract: This work presents a methodology for the simultaneous generation, design, and control of chemical process flowsheets (CPF) using RL, starting from an inlet flowrate and a set of unit operations (UOs) involving reaction-separation systems, each equipped with an embedded decentralized control system. The key innovation lies in embedding neural network surrogate models, which approximate the dynamic behaviour of complex UOs within the RL environment. The proposed framework was validated through a case study focused on the reaction and separation of products at varying purities. Results demonstrate the agent's ability to generate economically attractive CPFs that can maintain the dynamic operation of the systems in closed-loop in the presence of external disturbances.

Keywords: Reinforcement Learning, integrated design and control, surrogate models, neural networks

1. INTRODUCTION

The integration of process design and control is a significant research area in chemical engineering, aimed at optimizing chemical process flowsheets (CPF) for both economic viability and dynamic operability. Traditional sequential approaches determine equipment sizing at steady state before evaluating controllability for dynamic operation, often overlooking transient conditions and resulting in suboptimal or infeasible designs. This underscores the need for integrated strategies that align design and control decisions, ensuring dynamic operability. However, integrating design and control poses challenges due to conflicting objectives—minimizing costs versus ensuring dynamic operability. For instance, smaller equipment lowers capital costs but can hinder dynamic response. CPF design further involves both continuous and discrete decisions, such as selecting reactor types or the number of distillation trays. Complexity rises when unit operations (UOs) are modeled with differential-algebraic equations (DAEs), leading to mixed-integer dynamic optimization (MIDO) problems

Model-based optimization methods have emerged as effective techniques for solving the integrated design and control problem (Rafiei and Ricardez-Sandoval, 2018; Burnak et al., 2019; Patilas and Kookos, 2021). Despite their benefits, those methods are often constrained by two major challenges: i) intensive computational efforts, and ii) predefined superstructures or fixed arrangements of UOs. The latter limits flexibility, restricting the exploration of novel flowsheet configurations and innovative solutions. Model-free optimization techniques, such as Reinforcement Learning (RL), show great potential to advance chemical process design given their ability to design flowsheets without the need of a process superstructure (Reynoso-Donzelli and Ricardez-Sandoval, 2024a). This shift toward RL-based approaches has expanded to the integration of process design and control, leveraging RL's ability to manage complex decision-making. Inspired by the successful outcomes reported in previous studies (Sachio et al., 2022; Mendiola-Rodriguez and Ricardez-Sandoval, 2022), this study seeks to explore the application of RL to integrate decisions involving process flowsheet design, unit operation (equipment) design, and process controllability for reaction-separation systems.

A Proximal Policy Optimization (PPO) agent is designed to interact with the RL environment to design and control a CPF that optimizes a user-defined objective function. The objective function incorporates dynamic process variability, such as disturbance rejection and tracking errors, while ensuring compliance with both process and equipment operational and logical constraints. The agent's actions are guided by a reward shaping strategy that enforces these objectives and constraints. A key idea in this work is that the environment is composed of neural networks (NNs), which serve as surrogate models to approximate the dynamic behavior of the UOs in closed-loop. These surrogate models are identified prior to the RL training phase. Using NNs as surrogates of the actual UO mechanistic models reduce the computational costs (Schweidtmann and Mitsos, 2019). The framework is validated through a case study with two scenarios, where the agent designs and controls a CPF involving reaction-separation systems for the production and purification of two products. In this case, the agent manages three UOs, dynamically adjusting their operation to achieve optimal performance while accounting for external process disturbances.

2. PROBLEM STATEMENT

In this section, a general definition of the integrated design and control problem is presented. As shown in Eq. (1), the integrated problem, formulated as a MIDO problem, aims primarily at minimizing an objective function (1a) that may represent a combination of process economics, environmental requirements, sustainability incentives, etc. MIDO problems are constrained by DAEs (1b-1c) and their corresponding

initial conditions (1d). These functions represent the equations that describe the dynamic behaviour of the system such as continuity, mass and energy balances, thermodynamics, reaction kinetics, as well as any DAE representing physical-chemical phenomena. The problem is also constrained by equality (1e) and inequality (1f) constrains, that may posses a differential-algebraic nature or a purely algebraic nature. The system states and their derivatives, represented as functions of time, are denoted by $\boldsymbol{x}(t), \dot{\boldsymbol{x}}(t) \in \mathbb{R}^{\boldsymbol{n_x}}$ while $\boldsymbol{y}(t) \in \mathbb{R}^{\boldsymbol{n_y}}$ represents the system's output variables. Logical and disjunctive constraints that delineate decisions like the existence or absence of a flow or UO in the CPF are also considered (1g). These inequalities give rise to Boolean variables; $\boldsymbol{D} \in \{True, False\}^{n_D}$ represents the Boolean variables that appear in disjunctions, determining whether a specific constraint vector is enforced or not. These constraints are related to the logical proposition $\Phi(\boldsymbol{D}) = True$ through the logical operators (e.g., AND, OR, XOR, negation, implication, equivalence), which are defined based on the specific disjunctive decisions considered in the problem. Vector $\boldsymbol{\eta} \in \mathbb{R}^{\boldsymbol{n_\eta}}$ includes time independent continuous design and control variables, while $\boldsymbol{\gamma} \in \mathbb{Z}^{\boldsymbol{n_\gamma}}$ contains the integer design variables. Vector $\boldsymbol{\kappa} \in \mathbb{R}^{\boldsymbol{n_\kappa}}$ represents continuous operating conditions, which can vary over time, while the controlled variables are denoted by $\boldsymbol{u}(t)$ ($\boldsymbol{u} \in \mathbb{R}^{\boldsymbol{n_u}}$). Furthermore, $\boldsymbol{d}(t)$ denotes the disturbances, which spans from an initial time t_0 to a final value t_f.

$$\min_{\boldsymbol{\eta}, \boldsymbol{\kappa}(t), \boldsymbol{\gamma}, \boldsymbol{u}(t)} \Psi(\boldsymbol{x}(t), \dot{\boldsymbol{x}}(t), \boldsymbol{y}(t), \boldsymbol{\eta}, \boldsymbol{\kappa}(t), \boldsymbol{\gamma}, \boldsymbol{u}(t), \boldsymbol{d}(t), t) \quad (1a)$$

$s.t.$

$$\boldsymbol{f_d}(\boldsymbol{x}(t), \dot{\boldsymbol{x}}(t), \boldsymbol{w}(t), \boldsymbol{\eta}, \boldsymbol{\kappa}(t), \boldsymbol{\gamma}, \boldsymbol{u}(t), \boldsymbol{d}(t), t) = 0 \quad (1b)$$

$$\boldsymbol{f_a}(\boldsymbol{x}(t), \boldsymbol{w}(t), \boldsymbol{\eta}, \boldsymbol{\kappa}(t), \boldsymbol{\gamma}, \boldsymbol{u}(t), \boldsymbol{d}(t), t) = 0 \quad (1c)$$

$$\boldsymbol{f_0}(\boldsymbol{x}(t_0), \dot{\boldsymbol{x}}(t_0), \boldsymbol{w}(t_0), \boldsymbol{\eta}, \boldsymbol{\kappa}(t_0), \boldsymbol{\gamma}, \boldsymbol{u}(t_0), \boldsymbol{d}(t_0), t_0) = 0 \quad (1d)$$

$$\boldsymbol{h}(\boldsymbol{x}(t), \dot{\boldsymbol{x}}(t), \boldsymbol{w}(t), \boldsymbol{\eta}, \boldsymbol{\kappa}(t), \boldsymbol{\gamma}, \boldsymbol{u}(t), \boldsymbol{d}(t), t) = 0 \quad (1e)$$

$$\boldsymbol{g}(\boldsymbol{x}(t), \dot{\boldsymbol{x}}(t), \boldsymbol{w}(t), \boldsymbol{\eta}, \boldsymbol{\kappa}(t), \boldsymbol{\gamma}, \boldsymbol{u}(t), \boldsymbol{d}(t), t) \leq 0 \quad (1f)$$

$$\Phi(\boldsymbol{D}) = True$$

$$\bigvee_{j \in P_q} \begin{bmatrix} D_{j,q} \\ g_{j,q}(\boldsymbol{\eta}, \boldsymbol{\kappa}(t), \boldsymbol{\gamma}, \boldsymbol{u}(t)) \leq 0 \end{bmatrix}, \forall q \in Q \quad (1g)$$

$$\boldsymbol{\eta} \in [\boldsymbol{\eta}^L, \boldsymbol{\eta}^U] \quad (1h)$$

$$\boldsymbol{\kappa} \in [\boldsymbol{\kappa}^L, \boldsymbol{\kappa}^U] \quad (1i)$$

$$\boldsymbol{u} \in [\boldsymbol{u}^L, \boldsymbol{u}^U] \quad (1j)$$

$$\boldsymbol{\gamma} \in \{\gamma_1, \dots, \gamma_N\} \quad (1k)$$

$$t \in (t_0, t_f] \quad (1l)$$

Model-based optimization methods are well-suited for addressing the integrated problem, but they present several difficulties. A major challenge is the inclusion of disjunctive and/or integer decisions alongside process dynamics, which significantly increases problem complexity; solving such problems often requires advanced mathematical and optimization techniques. Another challenge is the need to predefine a superstructure that includes all necessary UOs, a process that relies heavily on user experience. Moreover, these methods often struggle to scale to larger problems, even without Boolean or integer variables, and heavily depend on well-informed initial guesses for convergence. These limitations underscore the need for innovative new approaches.

3. REINFORCEMENT LEARNING FRAMEWORK

This section presents a RL approach to solve the integrated problem described in the previous section. In this work, process dynamics are represented through surrogate models. The way such models are identified is presented next, followed by rest of the RL scheme, i.e., action space, observation vector, reward function and agent. The methodology presented builds upon a previous work (Reynoso-Donzelli and Ricardez-Sandoval, 2024b). However, this study specifically focuses on reaction-separation systems, emphasizing the advantages of the proposed framework for those applications. More details on the implementation, reward function and hyperparameters considered in this framework can be found in our previous work (Reynoso-Donzelli and Ricardez-Sandoval, 2024b).

3.1 Surrogate models

The agent interacts with an environment composed of surrogate models that approximate the dynamic behavior of closed-loop UOs. These models condense dynamic response data into specific values for the agent's use. Three types of surrogate models were developed: endpoint regressors, dynamic performance metric regressors, and label classifiers. Each of these surrogate models is described at the end of this section. Data is needed to identify the surrogate models. In this work, the data generation process aims to create a matrix $\boldsymbol{M}_{UO}$ for each UO, as the input variables vary between different UOs. $\boldsymbol{M}_{UO} \in \mathbb{R}^{m \times n}$ is composed of m vectors ($\boldsymbol{z}^m$), each containing all the input variables necessary for that UO, i.e., $\boldsymbol{z}^m \in \mathbb{R}^n$. The input variables include feed stream conditions (e.g., temperature, concentration), UO-specific design variables (e.g., reactor diameter, number of distillation stages), and control system parameters (e.g., set-point values, controller tuning parameters). When designing $\boldsymbol{M}_{UO}$, operational bounds need to be provided, as the feed stream to the UO can be situated at any point within the CPF. These bounds ensure realistic input ranges for training the surrogate model. In this study, Latin Hypercube Sampling (LHS) was used to populate $\boldsymbol{M}_{UO}$ using the bounds defined *a priori* for each input variable. LHS is effective in covering the multidimensional input space while minimizing redundancy, enhancing the accuracy of the surrogate models. A preliminary data analysis needs to be performed to avoid input combinations that may be infeasible and could interfere with the learning process, ensuring logical consistency throughout the dataset, e.g., set-point concentrations higher than inlet concentrations in a reactive system are discarded in the design of $\boldsymbol{M}_{UO}$. The input variables $\boldsymbol{z}^m$ are used to simulate the mechanistic process models in closed-loop to generate time-dependent output variables ($\boldsymbol{y}(t)$). Since surrogate models cannot fully replicate the dynamic responses of mechanistic models, different key values are captured to approximate the dynamic behaviour. All responses generated when evaluating $\boldsymbol{z}^m$ were stored in a matrix $\boldsymbol{M}_{Resp}$, which, along with $\boldsymbol{M}_{UO}$, were used to train the surrogate models. Due to differences in output responses—both in magnitude and type (continuous vs. Boolean)—it was found that training the surrogate models individually yielded more accurate results. The Mean Squared

Error (MSE) served as the loss function for their training, and Adam's optimizer was used to compute gradients and update their network weights. As shown in Figure 1, three key metrics are used to represent the dynamic response of UOs, these are defined next. i) Endpoint value: this corresponds to the final value of an output variable, such as temperature or concentration, at t_f (i.e., $y(t_f)$). This time point is typically selected assuming the system has reached a steady state. Note that any other time points for which a variable is required to be within constraint could be considered. The endpoint is approximated by an endpoint regressor, represented as $\widehat{y(t_f)} = NN_{EP}(\boldsymbol{z}^m)$, where a neural network (NN_{EP}) uses $\boldsymbol{z}^m$ as input to predict the endpoint. ii) Dynamic performance metric: This value rates the control strategy of the proposed design using metrics such as the Integral of Squared Error (ISE) or Integral of Absolute Error (IAE). Dynamic performance metric regressors approximate these values, enhancing predictive accuracy by computing the natural logarithm of the metrics, particularly useful when values approach zero. This value is approximated using a surrogate dynamic performance regressor, expressed as $\ln\widehat{(DM)} = NN_{DP}(\boldsymbol{z}^m)$. iii) Label value: This is a Boolean value used to classify whether the proposed UO design violates design or operational constraints while maintaining target operation in closed-loop. The label is determined through an algorithm with various logical checks (Algorithm 1). A surrogate classifier model is tasked to predict the determined label, mathematically expressed as $\widehat{Label} = NN_C(\boldsymbol{z}^m)$.

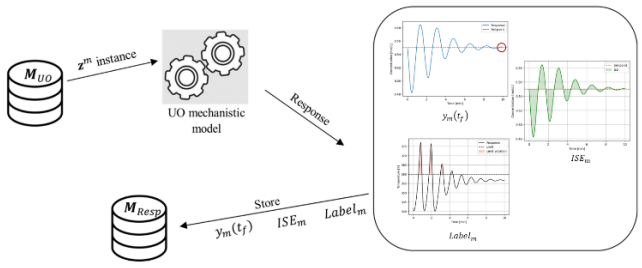

Figure 1. Database building

Algorithm 1 Pseudocode for label classification

```
if |y(t_f) − y_sp| ≤ ε_1 then
  if (1/r) Σ_{j=1}^{r} (dy/dt)_j ≤ ε_2 then
    if cons ≤ ε_3 then
      pass = 1
    else
      pass = 0
    end if
  else
    pass = 0
  end if
else
  pass = 0
end if
```

The first logical test in Algorithm 1 specifies if the distance between the mechanistic model's predicted endpoint $y(t_f)$ and the design set-point (y_{sp}) is within a threshold (ϵ_1). If this condition is not satisfied, the design is deemed unsatisfactory, regardless of the endpoint regressor's accuracy. The second test evaluates the oscillatory behavior by measuring the gradient of the model outputs ($\frac{dy}{dt}$) over the last r points. If the mean gradient exceeds a threshold (ϵ_2), the system's behavior is deemed non-ideal due to unwanted oscillations (i.e., not properly controlled). Once the two preceding logical conditions are satisfied, the next step is to verify compliance (i.e., $pass$) with design, logical, or operational constraints (i.e., $cons \leq \epsilon_3$). Estimations of these constraints vary by type, such as dynamic path or endpoint constraints, with a structure similar to the ISE used for dynamic path violations. Equation 2 presents a method for measuring the extent and magnitude of constraint violations. An auxiliary vector $\boldsymbol{\theta}(t)$, composed by the point difference between $g(t)$ and g_{cons}, i.e., maximum allowed input limit in a UO (e.g., temperature or liquid level), is integrated over time, outputting a punctual value denoted as $cons$. To address potential class imbalance, the Synthetic Minority Over-sampling Technique (SMOTE) is used to balance the dataset, enhancing classifier predictive power and preventing bias toward the overrepresented class.

$$g(\boldsymbol{x}(t), \dot{\boldsymbol{x}}(t), \boldsymbol{w}(t), \boldsymbol{\eta}, \boldsymbol{\kappa}(t), \boldsymbol{\gamma}, \boldsymbol{u}(t), \boldsymbol{d}(t), t) \leq g_{cons}$$
$$\boldsymbol{\theta}(t) = \begin{cases} g(t) - g_{cons}, & g(t) \geq g_{cons} \\ 0, & g(t) < g_{cons} \end{cases}$$
$$cons = \int_{t_0}^{t_f} \boldsymbol{\theta}(t)^2 dt \tag{2}$$

3.2 RL environment and agent

The environment serves as the external context for RL agent interactions and represents the part of the algorithm where Equation 1 is translated into RL terms to be amenable to the agent. At each step (i), the RL agent exchanges information with the environment through actions, observations and rewards. In this work, the action space $\boldsymbol{A}$ includes all design and control decisions for the UOs, represented as $\boldsymbol{A} = [\boldsymbol{\varrho}_d, \boldsymbol{\varrho}_c]$. The vector $\boldsymbol{\varrho}_d$ contains discrete variables, e.g., whether or not to include a UO in a flowsheet, and design decisions like the number of stages in a distillation column. Continuous variables in $\boldsymbol{\varrho}_c$ encompass design and control parameters, such as column diameter and PI controller tuning parameters. Note that other controllers can also be considered in the framework, e.g., model-based controllers. Discrete design variables are approximated using continuous distributions for computational efficiency, although this may introduce approximation errors. The environment includes functions that interpolate values from these probability distributions, guiding the agent's actions through sampling at each step (i). The step-observation vector o_i consists of key process variables, including operating conditions like temperature, pressure, and total flow, as well as a tracker for chemical components (e.g., reactants or products) and the current observation step (i). All elements are normalized to ensure balanced data processing within the neural network. In this approach, the reward shaping technique is used to guide the agent by composing the step reward r_i from multiple sub-rewards, each addressing different objectives. The step reward is expressed as: $r_i = \sum_{b \in B} r_b$ where b represents each sub-reward component. The three main sub-rewards are: 1) Capital cost r_{CC}, which quantifies the cost related to UO design variables ($r_{CC} = f\left(\boldsymbol{\gamma}, \boldsymbol{\eta}, \widehat{y(t_f)}\right)$). 2) Dynamic cost r_{DC}, based on dynamic performance metrics like the predicted ISE ($r_{DC} = f(\ln\widehat{(DM)})$). 3) Constraint violation cost r_{CV}, penalizes

constraint violations based on predicted label ($r_{CV} = f(\widehat{Label})$). At the final step, an additional penalty is applied if design objectives are not met. The agent used is a PPO agent (Schulman et al., 2017), with a modified activation function using tanh instead of ReLU. This change ensures the outputs are bounded, making it more suitable for real-life scenarios that involve working with data constrained within specific limits.

A key limitation of this approach lies in the use of surrogate models, which may result in the loss of transient information during the approximation process. Moreover, the accuracy of the surrogates depend on the availability of a comprehensive database or sufficient training data. Another challenge is the need for long simulation times, which can make the modeling and validation process more resource-intensive.

4. RESULTS AND DISCUSSION

The methodology described in the previous section was evaluated using a case study aimed to simultaneously generate, design and control a reactor-separator CPF capable of producing two high-purity products (B and C) from reactant A. The process operates under isothermal conditions and is subject to a +50% step disturbance in the inlet flow rate. For this case the RL agent has access to 3 UOs: a reactor where two first-order reactions ($A \rightarrow B$ and $A \rightarrow C$) occur in parallel, a binary distillation column (DC) and a flash tank (FT). The latter two units can be used by the agent to separate the binary mixture of products B and C. It is assumed that DC and FT operate with a constant relative volatility of 2.5 (i.e., ability of species to vaporize). The mechanistic models describing the dynamic behaviour for each UOs were adapted from the literature (Schweiger and Floudas, 1998; Bequette, 2002). Two scenarios were investigated, both requiring the agent to produce an outlet stream with a molar fraction of product B smaller or equal to: 0.1 (*Scenario A*) and 0.4 (*Scenario B*). In both scenarios, the agent was set to ensure a user-defined conversion of reactant A of at least 95% to enable an effective separation, while also rejecting disturbances in the inlet flow and keeping the liquid reaction volume within an 8-liter reactor design capacity limit. Note that different conversion values can be imposed to enhance or limit exploration. To simplify the analysis, the separation UOs are removed from the agent's action space until 95% of the reactant is converted. Once achieved, the separation units are activated, and the reactor deactivated. This action space masking is applied to prevent the agent from incorrectly using binary separation operations on a ternary stream.

A total of 11 surrogate models were identified: 5 endpoint regressors that predicted the outlet concentration of A from the reactor, the liquid reactor volume, the molar fraction of B in the bottoms for both the DC and FT and the liquid outflow of the FT. Moreover, 3 dynamic performance metric regressors that predicted the ISE for the concentration of A in the reactor, as well as the output liquid molar fraction of B for the FT and the DC were identified. Likewise, 3 classifiers that evaluate operational compliance with constraints for each unit are included. These operational constraints consist of maintaining the 8-liter design limit in the reactor, avoiding DAE violations (constraints enforced by the dynamic modeling suite), and preventing any stream in both the DC and FT from drying out (i.e., flowrate equal to zero). Note that the constraints for each UO are evaluated in closed-loop. Note that the agent can arrange multiple FTs in series, as their separation capability is lower than DC. To determine which of the two output streams will serve as the feed for the next FT, the stream with the greater mass is selected. For simplicity, if the vapor stream has a higher mass, it is assumed to be condensed before being fed into the subsequent FT. To train these models, three different matrices $\boldsymbol{M}_{UO}$ were generated, one for each UO. The variables considered in $\boldsymbol{M}_{UO}$ (i.e., $\boldsymbol{z}^m$) include design variables (sizing, set-points, control parameters) and input variables (concentration, molar flow, molar fraction) for each UO. The full set of variables used in the three matrices is not shown here for brevity but their definitions can be found elsewhere (Reynoso-Donzelli and Ricardez-Sandoval, 2024b). The action space for this problem was composed of all the design decisions related to the 3 UOs (Equation 3). For the reactor: set-point of the concentration of A $\left(C_A^{sp}\right)$, initial liquid reaction volume (V_r), and controller parameters (K_C^R, τ_I^R). For the DC: the number of stages (N), the set-point of the molar fraction of B at the bottoms (x_B^{sp}), column diameter (D_C), feed stage (f_n), reflux stage (r_n), and controller parameters at the tops (K_C^D, τ_I^D) and at the bottoms (K_C^B, τ_I^B). For the FT: molar fraction set-point of B at the liquid outlet $\left(x_L^{sp}\right)$ and controller parameters at the vapor outlet (K_C^{FT}, τ_I^{FT}). In addition to the design variables of the three UOs, $\boldsymbol{A}$ also includes a binary variable that indicates the choice between DC and FT, denoted as $\tilde{\mu}$.

$$\boldsymbol{A} = \begin{bmatrix} C_A^{sp}, V_r, K_C^R, \tau_I^R, N, x_B^{sp}, D_C, f_n, r_n, \\ K_C^D, \tau_I^D, K_C^B, \tau_I^B, x_L^{sp}, K_C^{FT}, \tau_I^{FT}, \tilde{\mu} \end{bmatrix} \quad (3)$$

Note that the action space consists of individual probability distributions rather than a combination of different continuous and discrete actions. The selection of the action at each step (i.e., a_i) involves sampling values from predefined probability distributions. The observation vector returned at each step includes information on the outlet stream of the selected UO, such as the concentration of reactant A and products B and C, the molar fraction of B, and the current step, i.e., $o_i = [C_A, C_B, C_C, z_B, i]$. Table 1 outlines the general form of the reward function, which is composed of sub-rewards adapted for each UO. The mathematical formula and meaning of each sub-reward functions are also presented in Table 1.

In this context, ϑ represents a free design variable, meaning the user can assign any value depending on the sub-reward. For r_{CC}, ϑ is a design variable (e.g., volume, diameter, height) that characterizes the UO and is normalized using the maximum and minimum values the agent can take for that variable. For r_{DC}, the agent uses $\widehat{ISE}$, normalized by the smallest ISE value the surrogate model is capable to regress; and depending on the $\widehat{ISE}$ value, the penalty can be constant or variable. For r_{CV}, the constant value assigned to ϑ depends on the classifier's predicted label. Moreover, for the design requirement, ϑ receives a constant penalty if the agent fails to

meet the design constraint (e.g., final purity, conversion) within the maximum number of steps allowed per episode (I). The magnitude of the constant values assigned in the last two sub-rewards is user-defined. However, setting excessively high values – particularly for r_{CV} – can negatively impact the agent's ability to explore effectively.

Table 1: Sub-rewards

Sub-reward	Assigned	Mathematical formulation
r_{CC} (Capital cost)	Every step	$r_{CC} = \frac{\vartheta - \vartheta_{min}}{\vartheta_{max} - \vartheta_{min}}$
r_{DC} (Dynamic cost)	Every step	$r_{DC} = \begin{cases} -1, & \widehat{ISE} \geq 1 \\ -\left(1 - \frac{\ln(\widehat{ISE})}{\ln(ISE)_{min}}\right), & \widehat{ISE} < 1 \end{cases}$
r_{CV} (Constraint violation)	Every step	$r_{CV} = \begin{cases} 0, & \widehat{Label} = 1 \\ -\vartheta, & \widehat{Label} = 0 \end{cases}$
Design requirement	Final step	$\begin{cases} 0, & i < I \\ -\vartheta, & i \geq I \end{cases}$

For both scenarios the PPO agent was trained for 100,000 steps, taking approximately 50 minutes per training. Note that in RL, a step is a single interaction where the agent takes an action and receives feedback. An episode is a sequence of steps that ends when a termination condition is met. The actor network consisted of 2 hidden layers, each with 64 neurons, while the critic network had only 1 hidden layer with 64 neurons. The actor-critic networks were updated every 2048 steps with a discount factor of 0.99 and optimized using Adam optimizer with a starting learning rate of 2.5e-4, which was adjusted throughout the training.

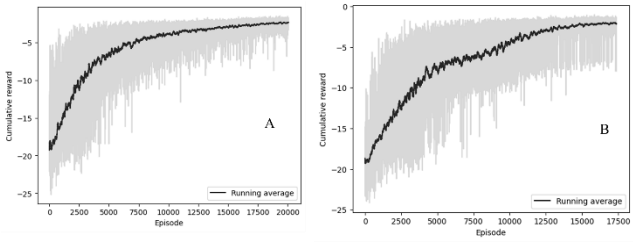

Figure 2: Learning curve: A) Scenario A; B) Scenario B

The learning curves shown in Figures 2.A and 2.B depict the cumulative rewards achieved by a PPO agent under two different scenarios. The black line represents the running average of the rewards over the episodes, while the shaded grey region illustrates the cumulative rewards obtained during each episode. Both scenarios display a similar trend: an initial phase of rapid, exponential growth followed by a plateau at a stable value. However, *Scenario B* (Figure 2.B) shows significantly more variability throughout the training process, as evidenced by the broader grey region around the running average. Despite these differences, both curves plateau at a similar cumulative reward value, around -2.5. The agent in *Scenario A* reaches this plateau much earlier, with variability decreasing quickly as training progresses, while the agent in *Scenario B* takes longer to stabilize and experiences greater fluctuations even after the plateau is reached. The higher variability in *Scenario B* is likely due to frequent operational failures during the simulations. The classifier flagged many designs as infeasible, resulting in extended exploration phases and slower convergence. When evaluating the best-performing agents for *Scenario A* while using the mechanistic process models, the resulting CPF is capable of producing product B to a molar fraction of up to 0.1, having previously converted reactant A to 95%; adhere to the design constraints (liquid reaction capacity smaller than 8 liters); and produce an attractive solution, i.e., a solution that maximizes the reward function, minimizing capital and dynamic costs. For this case, the agent required three reactors and a DC with the design specifications shown in Table 2 (*Scenario A*).

Table 2: UOs designs and specifications for scenarios A and B

Scenario	Variable	CSTR 1	CSTR 2	CSTR 3	DC/FT
A	$V_r\ [L]$	3.8988	3.7787	3.8763	-
	$C_A^{sp}\ [mol/L]$	0.3352	0.1032	0.0480	-
	$K_C^R\ [L^2/mol\ min^{-1}]$	-164.6513	-172.4885	-173.0276	-
	$\tau_I^R\ [min^{-1}]$	5.2697	5.9563	6.3171	-
	N	-	-	-	13
	$D_C\ [m]$	-	-	-	0.3454
	x_B^{sp}	-	-	-	0.0984
	f_n	-	-	-	5
	r_n	-	-	-	12
	$K_C^D\ [mol/min]$	-	-	-	-13.7983
	$\tau_I^D\ [min^{-1}]$	-	-	-	0.1381
	$K_C^B\ [mol/min]$	-	-	-	14.6740
	$\tau_I^B\ [min^{-1}]$	-	-	-	0.0785
B	$V_r\ [L]$	4.0549	4.0239	4.1461	-
	$C_A^{sp}\ [mol/L]$	0.3482	0.1044	0.0498	-
	$K_C^R\ [L^2/mol\ min^{-1}]$	-148.5977	-162.3069	-171.0000	-
	$\tau_I^R\ [min^{-1}]$	4.9108	5.7208	6.1898	-
	x_L^{sp}	-	-	-	0.3635
	$K_C^{FT}\ [mol/min]$	-	-	-	7.7068
	$\tau_I^{FT}\ [min^{-1}]$	-	-	-	0.4628

As shown in Figure 3, the disturbance was rejected across all three reactors, maintaining a feasible operation near the specified set-point. Although the agent could have used only two reactors to achieve the desired conversion of A with fewer units, it opted for three reactors due to the reactor's maximum capacity limit constraint. Regarding the liquid reaction volume, the first two reactors experience significant fluctuations, reaching up to 6.5 liters before reaching a stable operation. Despite this variability, none of the reactors exceed the design limitations (not shown for brevity). As shown in Table 2, the three reactors require large K_C^R and τ_I^R values. This control scheme induces oscillatory behavior in the response variable, specially for the first reactor, before stabilizing at the set-point (Figure 3).

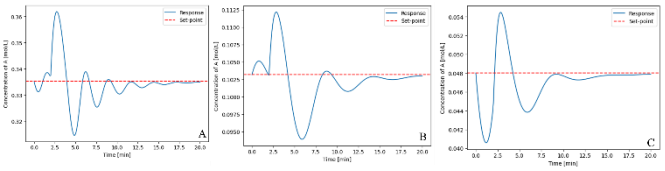

Figure 3: Scenario A: A) Reactor 1 B) Reactor 2 C) Reactor 3

Likewise, the DC was able to reject the disturbance while operating near the specified set-points (Figure 4A). The agent can achieve this operation using a DC with a small diameter and 13 trays. Also, the DC control strategy results in slow oscillations that settled after 15 minutes. Note that additional constraints on controller performance can be imposed while developing the proposed surrogate models.

A key advantage of the present RL framework is the ability to employ surrogate models across different problems, provided that the prediction ranges of these models align with the modifications required by the new problem. To illustrate this feature, the case study presented in this scenario (*Scenario A*) was modified by tasking the agent to achieve a target molar fraction of up to 0.4 for product B, while maintaining the 95% conversion of reactant A. The reactor's design volume constraint of 8 liters remained unchanged, and the agent was allowed to use a maximum of 10 steps to achieve the corresponding design goals.

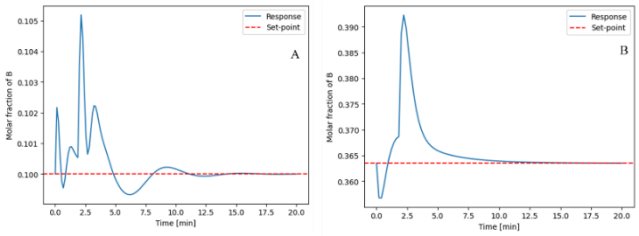

Figure 4: A) Scenario A, DC operation; B) Scenario B, FT operation

When evaluating the best-performing agents, the resulting CPF is capable of distilling product B to a molar fraction of 0.4, having previously converted reactant A to 95%; adhere to the design constraints (reaction's volume smaller than 8 liters); and produce an attractive solution. For this case, the agent required three CSTRs and a FT with the design specifications shown in Table 2 (*Scenario B*). The three reactors designed by the agent rejected the disturbance in the inlet flow rate while maintaining a feasible operation at the established set point (image not shown for brevity and close similarity to Figure 3). Compared to *Scenario A*, the reactor capacities are larger, with V_r around 4 liters. Despite this increase in reactor's capacity, none of the reactors violate the 8-liter capacity constraint. The second reactor comes closest to the limit, exhibiting oscillations that peak at approximately 7.9 liters (not shown for brevity). Also, the 3 reactors used relatively large values for both K_C^R and τ_I^R, allowing process oscillations. As shown in Figure 4B, the FT unit was able to reject the disturbance. The agent adopts a relatively aggressive strategy for FT, allowing a swift response of the controlled variable with minimal oscillations. In this scenario, the agent achieved a molar fraction of component B using only one FT. Furthermore, since x_L^{sp} is a decision variable that the agent must select, it is unlikely that the agent will precisely choose 0.4 unless the problem formulation explicitly sets this as an equality constraint. In both case studies, the most time-consuming task is data generation for training the surrogate models, taking up to 8 hours of CPU time. In contrast, training the surrogate models and the RL agent required ~55 minutes.

5. CONCLUSIONS AND FUTURE WORK

This study presented a novel framework for the simultaneous generation, design, and control of CPFs starting from an inlet stream and a set of UOs. The innovative aspect of this approach is highlighted by the implementation of surrogate models embedded within the RL environment, which approximate the mechanistic models of UOs in a dynamic state. Additionally, a tailored reward system was designed to guide the agent's design decisions penalizing economic and dynamic effects, as well as constraint violations. A case study with multiple UOs was addressed with this framework showing the benefits of using surrogate models by simplifying a complex mechanistic model, like the DC, into multiple surrogate models. Future work will consider the addition of uncertainty and the use of external advanced dynamic simulation suites to improve the quality of the predictions.

REFERENCES

Bequette, B., 2002. Process control: modeling, design, and simulation, First. ed. Prentice Hall Press, USA.

Burnak, B., Diangelakis, N.A., Katz, J., Pistikopoulos, E.N., 2019. Integrated process design, scheduling, and control using multiparametric programming. Comput. Chem. Eng. 125, 164–184. https://doi.org/10.1016/j.compchemeng.2019.03.004

Mendiola-Rodriguez, T.A., Ricardez-Sandoval, L.A., 2022. Robust control for anaerobic digestion systems of Tequila vinasses under uncertainty: A Deep Deterministic Policy Gradient Algorithm. Digit. Chem. Eng. 3, 100023. https://doi.org/10.1016/j.dche.2022.100023

Patilas, C.S., Kookos, I.K., 2021. Algorithmic Approach to the Simultaneous Design and Control Problem. Ind. Eng. Chem. Res. 60, 14271–14281. https://doi.org/10.1021/acs.iecr.1c01855

Rafiei, M., Ricardez-Sandoval, L.A., 2018. Stochastic Back-Off Approach for Integration of Design and Control Under Uncertainty. Ind. Eng. Chem. Res. 57, 4351–4365. https://doi.org/10.1021/acs.iecr.7b03935

Reynoso-Donzelli, S., Ricardez-Sandoval, L.A., 2024a. A reinforcement learning approach with masked agents for chemical process flowsheet design. AIChE J. n/a, 16. https://doi.org/10.1002/aic.18584

Reynoso-Donzelli, S., Ricardez-Sandoval, L.A., 2024b. An integrated reinforcement learning framework for simultaneous generation, design, and control of chemical process flowsheets. Comput. Chem. Eng. 194,108988. https://doi.org/10.1016/j.compchemeng.2024.108988

Sachio, S., Mowbray, M., Papathanasiou, M.M., del Rio-Chanona, E.A., Petsagkourakis, P., 2022. Integrating process design and control using reinforcement learning. Chem. Eng. Res. Des. 183, 160–169. https://doi.org/10.1016/j.cherd.2021.10.032

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O., 2017. Proximal Policy Optimization Algorithms.

Schweidtmann, A.M., Mitsos, A., 2019. Deterministic Global Optimization with Artificial Neural Networks Embedded. J. Optim. Theory Appl. 180, 925–948. https://doi.org/10.1007/s10957-018-1396-0

Schweiger, C.A., Floudas, C.A., 1998. Interaction of Design and Control: Optimization with Dynamic Models, in: Hager, W.H., Pardalos, P.M. (Eds.), Optimal Control: Theory, Algorithms, and Applications, Applied Optimization. Springer US, Boston, MA, pp. 388–435. https://doi.org/10.1007/978-1-4757-6095-8_19