A Q-LEARNING BASED FAULT-TOLERANT CONTROLLER WITH APPLICATION TO CSTH SYSTEM

Seyed Ali Hosseini and Karim Salahshoor

Department of Automation and Instrumentation Engineering
Petroleum University of Technology Ahwaz, Iran

ABSTRACT

Systems are continually subjected to faults or malfunctions because of age or sudden events, which might degrade the operation performance and even result in operation failure that is a quite important issue in safety-critical systems. Thus, this important problem is the main reason to use the Fault-Tolerant strategy to improve the system's performance with the presence of faults. A fascinating property in Fault-Tolerant Controllers (FTCs) is adaptability to system changes as they evolve throughout system operations. In this paper, a Q-learning algorithm with a greedy policy was used to realize the FTC adaptability. Then, some fault scenarios are introduced in a Continuous Stirred Tank Heater (CSTH) to compare the closed-loop performance of the developed Q-learning-based FTC with concerning conventional PID controller and an RL-based FTC. The obtained results show the effectiveness of Q-learning-based FTC in different fault scenarios.

KEYWORDS

Reinforcement Learning, Q-learning Algorithm, Fault-Tolerant controller, Adaptive Controller.

1. INTRODUCTION

In today's world, the correct functioning of complex industrial systems is required to ensure efficient and high-quality production. This is particularly important to those safety-critical systems, such as power systems, aircraft, autonomous transportation, or chemical system processing hazardous materials where a small component fault/failure may cause catastrophic effects. To avoid production deteriorations and enhanced system reliability, measures must be taken to stop the propagation of fault and restore the system as much as possible to satisfactory performance when the fault occurs. This practical requirement gives rise to lots of studies in Fault-Tolerant Control (FTC) from both industry and academia [1].

Modern systems try to use historical data in their processes. As a result, data-driven identifications, diagnosis, and FTC have become a hot research topic. Previous information could be used for learning to extract the knowledge base, such as using Fuzzy predictions, Neural Network (NN) based methods, k-clustering, and support vector machines (SVM) [2].

Both sensors and control systems will target the plant to obtain the maximization benefits by optimizing the performance indicator within the scope of safety. FTC is a good alternative that has the capability of approaching performance indicators without any fault by adjusting the system variables. In [3], a gradient-based optimization method was given to optimizing the

David C. Wyld et al. (Eds): SOFEA, CTCM, BIBC, SIPR, NCWC, CSEN, EDTECH - 2021
pp. 57-66, 2021. CS & IT - CSCP 2021 DOI: 10.5121/csit.2021.111605

system performance using disturbance rejection. In [4], a recursive total principal component regression (R-TPCR)-based design and implementation approach was proposed for efficient data-driven FTC and optimization.

The performance indicator without any fault reflects the system's natural ability and is easy to be gained from obtained data. However, it becomes a challenge in the case of fault because the unexpected fault has changed the maps of system states, and there are not enough valid data to develop an FTC controller for an early fault. Reinforcement Learning (RL) inspires to solve the above problem. RL is about learning from the interaction how to behave to achieve a goal. The RL agent and its environment interact over a sequence of discrete-time steps and gain a series of optimal actions finally. If an unexpected fault is considered as the environment, and the system's performance under fault-free conditions is regarded as the desired goal, the controller can be designed by RL to achieve the optimal behavior. In this paper, we compare the use of Q-learning algorithm in FTC strategy with conventional PID-controller. Also, their performance was shown in the presence of varied fault scenarios.

2. BACKGROUND

2.1. Fault-Tolerant Control (FTC) strategy

Faults in automatic methods will frequently purpose undesired reactions and shut-down of a control plant to personnel or environment. Fault-tolerant control is the synonym for a set of recent techniques that were developed to increase plant availability and reduce the risk of safety hazards. The aim is to prevent that simple faults develop into severe failure [5].

When stability and closed-loop performance are maintained despite faults, the system is said to be fault-tolerant, and the control scheme that ensures the fault tolerance is the fault-tolerant controller. FTC relates to recovery from weakness such that the system is controlled under actual constraints without replacing part(s) of the faulty system. In general, FTCs fall into two types: passive and active FTC systems. In passive FTCSs, controllers are only able to process faults that were considered during the controller's design stage. This system requires no controller self-repairing/reconfiguration and therefore has limited fault-tolerant capabilities. Unlike passive FTCSs, active FTCSs react actively to system fault by control action reconfiguration, stability, and good performance can then be recorded. In active FTCSs, the main focus is to design the controller. The existing design approaches can be generally classified into two categories: model-based and data-driven (model-free) methods. In the model-based method, a precise model of the existing system should be known a priori, and fault identification is required no construct a post-fault system model before active FTCS controller design. In the data-driven method, unlike in the model-based one, the system model is identified using available historical data [1].

2.2. Reinforcement Learning (RL) Method

RL is the learning of behavior by an agent, or a controller, from feedback through repeated interaction with its environment [6]. RL is learning what to do -mapping situations to actions- to maximize a numerical reward signal (R) (an agent's objective). The learner is not told which actions to take but instead must discover which efforts yield the most reward by trying them. An agent (controller) is connected to (interacts with) the environment through its actions and perceptions. The change is perceived as an RL signal from the environment and the agent's measurements of the new state. The standard RL problems can be represented as Markov Decision Process (MDP). An MDP consists of a set of states (S), a group of actions (A), a reward function R: $S \times A \times S \rightarrow R$, which reinforces after each state change. The history of prior states

does not affect the value or reward of following states and actions. This is known as the Markov property [7]. In the most exciting and challenging cases, actions may affect not only the immediate reward but also the following situation and, through that, all subsequent rewards. These two characteristics -trial-and-error search and delayed reward- are the two most important distinguishing features of RL.

The value of a state ($V(S)$) is the maximum reward that an agent can expect from that state in the future.

$$V(S) = max_{a\in A}(E[R(S.a.S_{t+1}) + \gamma V(S_{t+1})]) \quad (1)$$

Where γ is the discount factor that weighs delayed reward against immediate reinforcement and S_{t+1} is the next state.

Alternatively, an agent can learn the value of each action ($Q(S.a)$), the q-value, from a state:

$$Q(S.a) = E[R(S.a.S_{t+1}) + \gamma max_{a\in A}Q(S_{t+1}.a)] \quad (2)$$

An agent can exploit either value function to derive a policy that governs its behaviors during operation.

Three fundamental classes of methods for solving finite Markov decision problems are Dynamic Programming (DP), Monte Carlo (MC), and Temporal Difference-Learning (TD-Learning). Each class of methods has its strengths and weaknesses. DP methods are well developed mathematically but require a complete and accurate model of the environment. MC methods do not require a model and are conceptually simple but are not well for step-by-step incremental computation. Finally, TD methods require no model and are incremental but are more complex to analyze.

3. RL-BASED FTC

Supervision uses a priori knowledge of the system to choose optimal actions when a fault occurs in the system. Fault-tolerant supervision is explored in several fields, including probabilistic reasoning, Fuzzy logic, and Genetic algorithms, and one of the supervision methods for FTC is RL-based control.

TD-based RL methods iteratively derive a controller's behaviors by estimating the value of states and actions in a system through exploration. During operation, the controller exploits the knowledge gained through exploration by selecting actions with the highest value, to the fact that tolled above, a stationary system was supposed in this paper. The agent or controller can achieve optimal control by exploring over multiple-episode and iteratively converging on the value function. The conservative learning rate can be used such that the value approximation is representative of the agent's history. Dynamic of the system is changed by fault. The extant value function may not reflect the most rewarding actions in the environment model. The optimal actions should be taken with the agent. Therefore, the agent must estimate a new value function by exploration when a fault occurs.

The TD-RL approach is inherently adaptive as it frequently updates its policy from exploration. Its responsivity can be enhanced by increasing the learning rate α. In the extreme case, α = 1

replaces the last value of an action with the new estimate at that time step. However, large values of α may not converge the value estimate to the global optimum.

We used a reference model to detect the presence of a fault in the system. This model is obtained by Neural Network approximation and data in the state of no-fault.

In this paper, we consider the difference between the output of the system and the output of the reference model (Error) on time as our state in RL and consider reward according to our state that shows below:

$$R = \begin{cases} If\ State \leq 2 \rightarrow 1 \\ If\ State \geq 2 \rightarrow -1 \\ If\ State > 5 \rightarrow -100 \end{cases}$$

Figure 1 shows an overview of the system with an RL-based Fault-Tolerant controller. The following subsection discusses a practical approach of TD-learning in control theory.

3.1. Q-Learning: off-policy TD control

In the on-policy method, off-policy methods evaluate or improve a policy different from that was used to generate the data. The policy being learned about is called the target policy (π), and the policy used to generate behavior is called the behavior policy (b). One of the early breakthroughs in RL was the developing an off-policy TD control algorithm known as Q-Learning [8], defined by equation 3.

$$Q(S_t.A_t) = Q(S_t.A_t) + \alpha\,[R_{t+1} + \gamma\, max_a Q(S_{t+1}.a) - Q(S_t.A_t)] \qquad (3)$$

In this case, the learned action-value function Q directly approximates q_*, the optimal action-value function, independent of the policy being followed (b). This dramatically simplifies the analysis of the algorithm and enables an effect in that it determines which state-action pairs are visited and updated. The Q-Learning control algorithm is shown in Table 1 [9].

Note: in this algorithm, all that is required for correct convergence is that all pairs continue to be updated. Under this assumption and a variant of the usual stochastic approximation conditions on the sequence of step-size parameters, Q has been shown to converge with probability 1 to q_*0F[1].

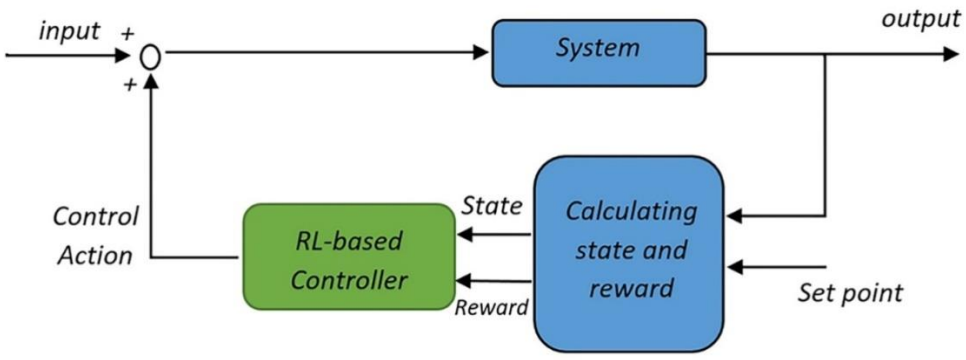

Figure 1. Block diagram of close loop system with RL-based FTC

[1] *: It is a symbol of optimality

4. CASE STUDY

CSTH is commonly used as a subsystem in heavy industrial processes [10]. A CSTH Process is depicted in Figure 2. We are interested in using a linear model of CSTH in Python application to be used in our closed-loop system. The total mass and energy balance equations for the process can be expressed as:

$$A_T \frac{dh_T}{dt} = q_i - q_o \quad (4)$$

$$A_j \frac{dh_j}{dt} = q_{ij} - q_{oj} \quad (5)$$

$$\frac{dT}{dt} = \frac{q_o}{V_T}(T_i - T) + \frac{UA(T_j - T)}{pCpV_T} \quad (6)$$

$$\frac{dT}{dt} = \frac{q_{oj}}{V_j}(T_{ij} - T_j) - \frac{UA(T_j - T)}{p_j Cp_j V_j} \quad (7)$$

We assume that the level of the fluid in the tank and the jacket is constant either by process design or the use of a high gain controller. So it could be written as:

$$q_i = q_o = q \quad (8)$$

$$q_{ij} = q_{oj} = q_j \quad (9)$$

$$\frac{dT}{dt} = \frac{q}{V_T}(T_i - T) + \frac{UA(T_j - T)}{pCpV_T} \quad (10)$$

$$\frac{dT}{dt} = \frac{q_j}{V_j}(T_{ij} - T_j) - \frac{UA(T_j - T)}{p_j Cp_j V_j} \quad (11)$$

Now we use steady-state analysis as follow,

$$\frac{dT}{dt} = \frac{q}{V_T}(T_i - T) + \frac{UA(T_j - T)}{pCpV_T} = 0 \quad (12)$$

$$\frac{dT}{dt} = \frac{q_j}{V_j}(T_{ij} - T_j) - \frac{UA(T_j - T)}{p_j Cp_j V_j} = 0 \quad (13)$$

The steady-state value of above variable could be found in Table 2.

Table 1. Q-Learning Algorithm

Q-Learning (off-policy TD control) for estimating by $\pi \approx \pi_*$

Initialize $Q(S.A)$, for all s ∈ S, a ∈ A(S), arbitrarily, and $Q(Terminal.\ all)$ = 0.
Repeat (for each episode):
  Initialize S
  Repeat (for each step of episode):
   Choose A from S using policy derived from Q(e.g. ε-greedy)
   Take action A, observe R, S_{t+1}

$$Q(S_t.A_t) = Q(S_t.A_t) + \alpha\,[R_{t+1} + \gamma\, max_a Q(S_{t+1}.a) - Q(S_t.A_t)]$$

 S = S_{t+1}
 Until S is terminal

Table 2. Steady-state value of variables

Variable	SS Value	Parameter	Value
T_{is}	*50*	UA	*183.9*
T_{ijs}	*200*	pCp	*61.3*
T_s	*125*	p_jCp_j	*61.3*
T_{js}	*150*	V_T	*10*
q_s	*1*	V_j	*1*
q_{js}	*1.5*	------	----

By using Taylor series expansion for linearizing the system, the following state-space equations are obtained:

$$\begin{bmatrix} T \\ T_j \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} T \\ T_j \end{bmatrix} \tag{14}$$

$$\begin{bmatrix} \dot{T} \\ \dot{T}_j \end{bmatrix} = \begin{bmatrix} -\frac{q}{V_T} - \frac{UA}{pCpV_T} & \frac{UA}{pCpV_T} \\ \frac{UA}{p_jCp_jV_j} & -\frac{q_j}{V_j} - \frac{UA}{p_jCp_jV_j} \end{bmatrix} \begin{bmatrix} T \\ T_j \end{bmatrix} + \begin{bmatrix} \frac{q}{V_T} & 0 & \frac{(T_i - T)}{V_T} & 0 \\ 0 & \frac{q_j}{V_j} & 0 & \frac{(T_{ij} - T_j)}{V_j} \end{bmatrix} \begin{bmatrix} T_i \\ T_{ji} \\ q \\ q_j \end{bmatrix} \tag{15}$$

Substitute given values:

$$\begin{bmatrix}\dot{T}\\ \dot{T}_j\end{bmatrix} = \begin{bmatrix}-0.4 & 0.3\\ 3 & -4.5\end{bmatrix}\begin{bmatrix}T\\ T_j\end{bmatrix} + \begin{bmatrix}0.1 & 0 & -7.5 & 0\\ 0 & 1.5 & 0 & 50\end{bmatrix}\begin{bmatrix}T_i\\ T_{ji}\\ q\\ q_j\end{bmatrix} \quad (16)$$

The control tank's output temperature (T) was studied by adjusting the jacket's output flow (Fj). Thus, the transfer function was shown in equation 17.

$$T = \frac{15}{s\text{^}2 \; + \; 4.9\, s \; + \; 0.9} \quad (17)$$

Finally, we use the Scipy library in Python to implement our system.

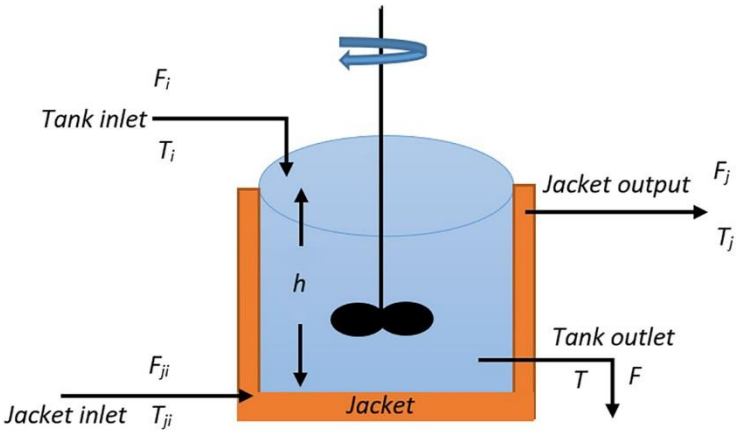

Figure 2. CSTH system

5. RESULT AND DISCUSSION

The RL-based fault-tolerant controller was evaluated with the presence of three fault scenarios, which were discussed in the following subsections. In addition, we run the program for 100 loops, and results were presented.

5.1. Sensor Fault Scenario

A time incipient fault with an amplitude of 5 and slope of 0.01 was added to the control system that is shown in Figure 3 [11]. The result was shown in Figure 3, and the total MSE was written in Table 3. According to
Figure *4* and Table 3, the performance of the system was evaluated with two controllers, conventional PID-controller and Q-Learning-based FTC, in the presence of the sensor fault. The performance of a closed-loop system with the proposed method is quite efficient than another controller.

5.2. Actuator fault scenario

According to Figure 3, a constant fault with an amplitude of 0.4 was added to the actuator, moreover, the result was picketed in
Figure *5*, and the total MSE could be found in Table 3. As shown in

Figure 5 and Table 3, the Q-learning controller performed flawlessly, and the output remained at the optimal point.

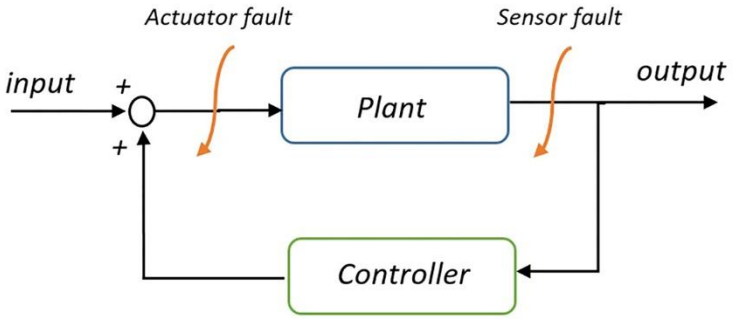

Figure 3. Location of actuator and sensor fault

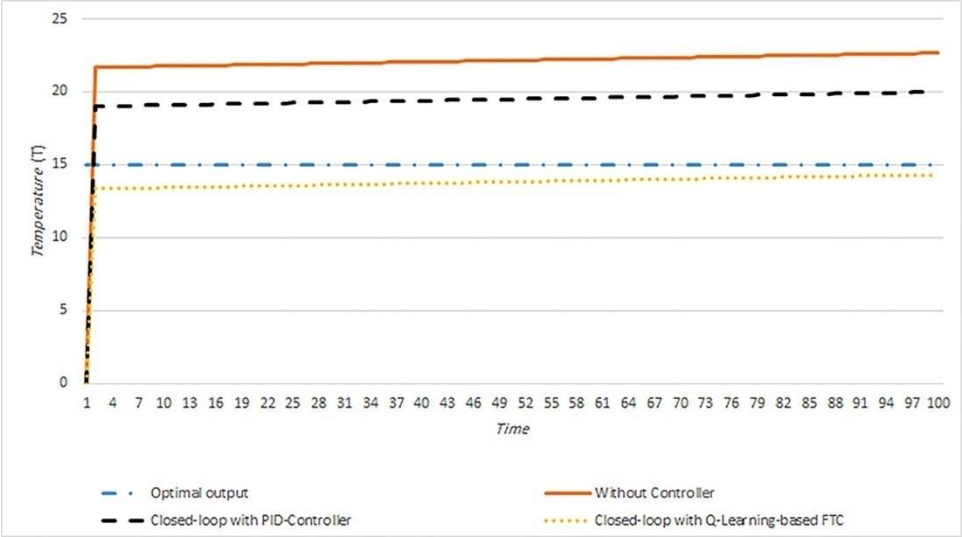

Figure 4. Sensor fault scenario

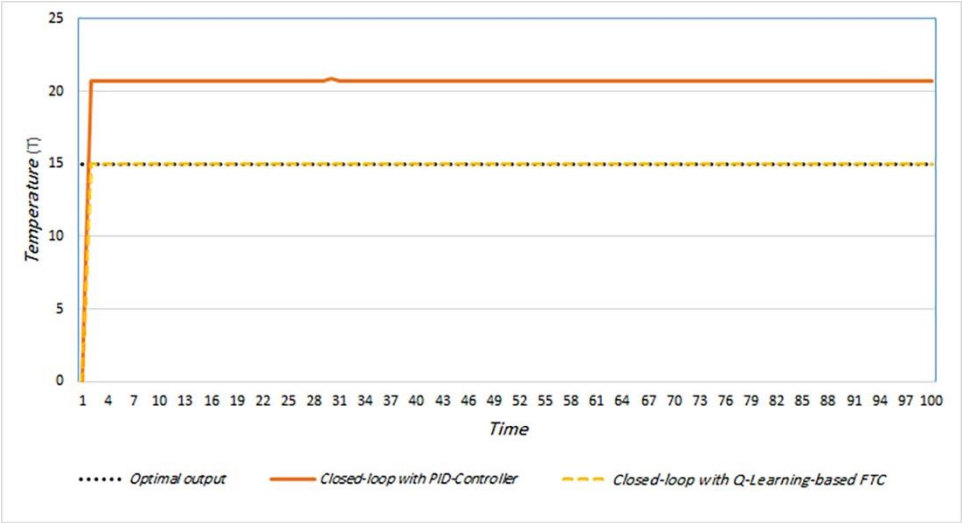

Figure 5. Actuator fault scenario

5.3. Actuator and Sensor Fault Scenario

In this section, a constant actuator fault with an amplitude of 0.1 plus a time incipient sensor fault with an amplitude of 4 and slope of 0.01 were added to the closed-loop system. Their location could be found in Figure 3. In addition,
Figure *6* shows the performance of closed-loop with conventional PID-controller and closed-loop system with Q-Learning-based FTC. Also, the total MSE was shown in Table 3. The results show that the Q-learning controller could find optimal actions to be stable the closed system as well as possible, and its performance is much better than PID-controller.

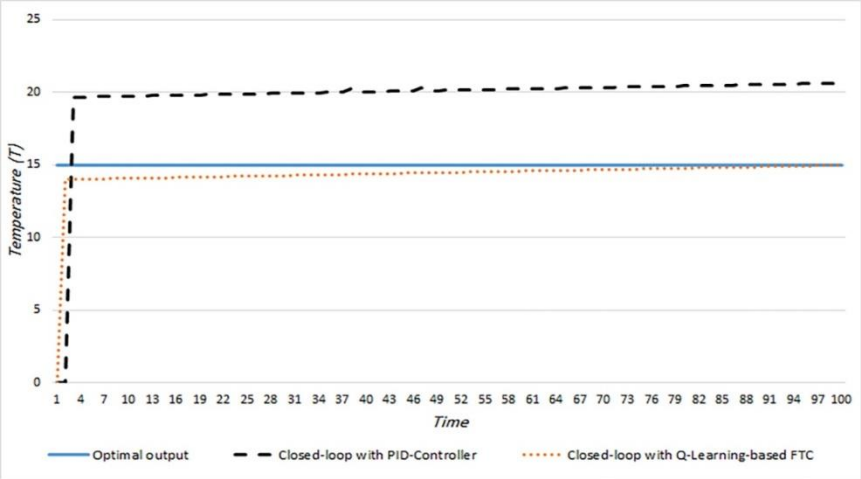

Figure 6. Sensor and actuator fault

6. CONCLUSION AND FUTURE WORK

A Q-learning algorithm with a greedy policy is used in the FTC framework. The result is a data-driven online method, which directs to the goal without knowing any system dynamic characteristics that are difficult to understand at the beginning when a fault occurs. A CSTH process was used to test the control systems. The comparison results showed that Q-learning algorithm-based FTC presents an efficient methodology to stabilize the CSTR performance when sensor and actuator faults occur in a closed-loop system. In addition, we compared its performance with conventional PID-controller, and the superiority of the proposed method was shown in the result section. In this paper, we used the greedy policy for the Q-learning algorithm. For further work, it would be interesting to search for another policy to obtain more efficiency and get a better result with respect to the RL-based FTC.

Table 3. Total MSE

Controller / **Scenario**	**PID**	**Q-Learning**
A	4.4938	1.2223

B	5.6672	0.0845
C	0.5691	0.051648

REFERENCES

[1] c. Hua, S. X.ding and Y. A.W.Shardt, "A New Method for Fault-Tolerant Control through Q-Learning," IFAC(International Federation of Automatic Control) Hosting by Elsevier Ltd., 2018.

[2] D. Zhang and Z. Gao, "Reinforcement learning-based fault-tolerant control with application to flux-cored wire system.," Measurement and Control, 2018.

[3] Hao Luo, Steven X. Ding, Kai Zhang, and Shen Yin, "A data-driven fault detection approach for static processes with deterministic disturbances," 2014 IEEE 23rd International Symposium on Industrial Electronics (ISIE).

[4] Macgregor J and Cinar A, "Monitoring, fault diagnosis, fault-tolerant control and optimization: data-driven methods," Comput Chem Eng, 2012.

[5] M. Blanke, C. Frie, F. Kraus, R. J. Patton and M. Straoswiecki, "What if Fault-tolerant Control?," IFAC Proceedings Volumes, 2000.

[6] S. A. a. S. K. Hosseini, "Comparison between SARSA and Q-Learning algorithms based Fault-Tolerant Control," in 1st International Conference on Mechanical, Electrical engineering and Engineering Sciences, Brussels-Belgium, 2021.

[7] I. Ahmed, H. Khorasgani and G. Biswas, "Comparison of Model Predictive and Reinforcement Learning Methods for Fault-Tolerant Control," IFAC, 2018.

[8] C. J. C. H. Watkins, learning from delayed reward, King's College, 1989.

[9] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction, 2017.

[10] Z. Chen, Data-Driven Fault Detection for Industrial Processes, Duisburg, German: Dissertation, University of Duisburg-Essen, 2016.

[11] Y. Zhang and J. Jiang, "Bibliographical review on reconfigurable fault-tolerant control systems," Annual Reviews in Control, 2008.