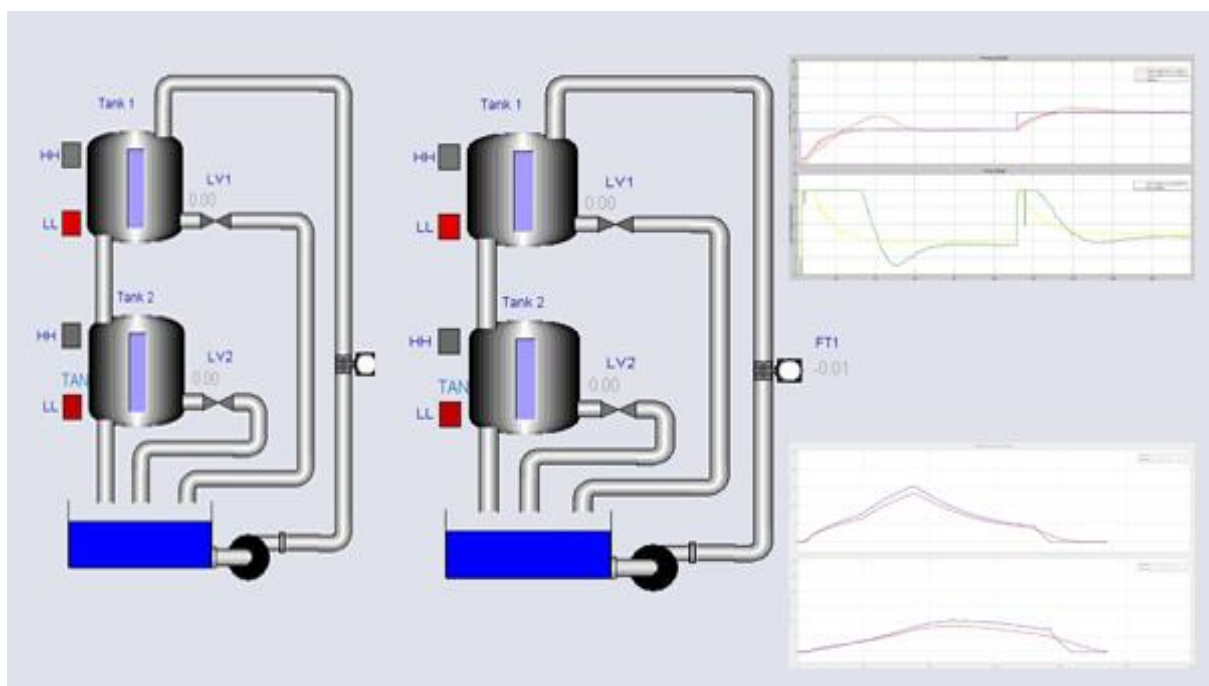


FMH606 Master's Thesis 2021
Industrial IT and Automation

Parallel configuration, control and mirror operation of twin processes using Emerson's DeltaV



Varjith Thilakasekaram

Course: FMH606 Master's Thesis, 2021

Title: Parallel configuration, control and mirror operation of twin processes using DeltaV

Number of pages: 85

Keywords: Parallel configuration, mirror processes, digital twin, synchronisation, process control, identical processes

Student: Varjith Thilakasekaram

Supervisor: Dr Carlos F. Pfeiffer

External partner: Emerson Porsgrunn (Rune Anderson)

Availability: Open

Summary:

The University of South-Eastern Norway has two identical Serial Two Tank systems, a first-order system with time delay. Identical twin processes are systems built with identical physical components. These identical systems were tested to configure in parallel and to mirror their performance identical to each other. As the first step, one process was taken as the primary process and tuned using the Skogestad PI tuning rule in MATLAB Simulink. Then the tuned controller parameters were used on both systems, and the outputs were compared. Moreover, to adapt both systems, the systems models were developed using the 1st principles theory. As some of the model parameters were not calculated experimentally, the models' parameters were tuned using an optimization algorithm.

The secondary process was configured using Emerson's DCS system. The configured system was simulated for a step-change in a closed-loop using the controller parameters gained for the primary system. The result showed that even though identical system dynamics will not be the same as each other. As for limited resources, the secondary system also connected with MATLAB Simulink for further experiments. Therefore, both primary and secondary systems were connected with MATLAB Simulink as the controller system. The simulation of both system with tuned parameters showed a need to optimize the secondary process controller parameters to mirror each other.

The secondary controller parameters were optimized using the LMAC (Learning Model Adaptive Control) and MRAC (Model Reference Adaptive Control) techniques. For the LMAC theory, the mathematical model of the secondary process was used to tune the controller parameter. For the MRAC, the controller parameter was tuned using the MIT rule of gradient descent. Moreover, the controller parameters also optimized using the error between the process input as the cost function.

The results showed that the mathematical model of the respective processes behaves in the same order but exists errors, even after the model parameters were tuned, which leads to conclude that the parameters' values even after tuning were not global minimum rather than a local minimum.

The optimization of the controller parameter using the model showed that this technique is possible only if the mathematical model accurately represents the real process to match the processes outputs to mirror each other. The MIT rule results showed that it is possible to reduce the error between the process outputs and follow one of the process outputs. It is concluded that using the MRAC technique, two identical models can be configured in parallel, and outputs can be mirrored.

Acknowledgements

I would be happy to say my sincere gratitude to Professor Dr Carlos F. Pfeiffer for selecting me as the project candidate. Moreover, I highly appreciate and thank him for his constant guidance, dedication, and effort in every step of this thesis. He played the most important job of mentoring me and supporting me in every hurdle during the thesis duration. On the other hand, I would like to thank Mr Rune Anderson for the guidance, pieces of advice, dedication, and commitment for helping me to conclude this project successfully. I am obliged to express my gratitude to both for spending their valuable time helping me during this project.

I also would like to convey my sincere appreciation to Mr Nima Janatianghadikolaei for the support and advice on some part of the project. The contribution and help received from him were very valuable and highly helpful to this project realization. I would like to appreciate for spending his time with me even in his tight schedule.

I truly owe Mr Rune Anderson for helping me in every step of configuring the DeltaV DCS system with the process and his constant advice when problems arise.

In addition, I would like to thank my colleges Mr Vasanthalingam and Miss. Kushila Jeyamanne, for their support. Last but not least, I cannot express enough thanks to my life partner Mrs Sagithiya Varjith for her moral support and guidance through this tough time and to my family members.

Varjith Thilakasekaram.

University of South-Eastern Norway – 2021.

Preface

This research was done as a part of the requirement to complete the master course Industrial IT and Automation offered by the University of South-Eastern Norway. This research introduces a novel idea for synchronizing similar processes in the industry in contrast with the present approach gaining popularity in the control engineering field called Digital Twin. In contrast to using the mathematical model to represent a process, in this research, one of the physical processes was used as the reference to manipulate the other process. The goal was to achieve a desired behaviour such that the second process mirrors the outputs of the first. This approach can be extended to several similar processes. For the testing and verification of the theory, two experimental systems for the level control of two tanks were used in the cascade. One of the experimental systems was controlled using a PID control unit with the Delta V systems of the Emerson, and the outputs from both processes were compared. A function of the errors between the outputs of the processes was used in an optimization algorithm to adjust the PID controller parameters of the second process to reduce the error between the process's outputs.

This report was written according to the guidelines of the USN. The report follows the structure of a standard template made available by the university for master's students. The optimization method used to test the theory and the procedures have been detailed in the chapter introduction and methodology and followed by the results of the experiments in a separate chapter. Subsequently, the report ends with the discussion, conclusion, recommendation, and Appendices.

Porsgrunn, 28.05.2021.

Varjith Thilakasekaram

Contents

Preface	5
Contents.....	6
Nomenclature	8
1 Introduction	9
1.1 DeltaV	9
1.2 Problem Description.....	10
1.3 Aim and Objectives.....	11
1.4 Structure of the thesis report.	11
2 Literature Review	12
2.1 The Two tank Process	12
2.2 Mathematical modelling of a tank system.....	13
2.3 PID Tuning of the controller	14
2.3.1 <i>The good gain method</i>	15
2.3.2 <i>Skogestad Method</i>	16
2.4 Optimisation Algorithms.....	18
2.4.1 <i>The MIT rule</i>	18
3 Mathematical Modelling of the Serial Two Tank Systems	20
3.1 The Serial Two-Tank Model	20
3.2 Model of the outputs flow rates (Fluid Resistance)	22
3.3 Mathematical modelling of the Serial Two Tank Model	24
3.3.1 <i>The transfer function of the model</i>	25
4 Methodology.....	28
4.1 Setup of the Serial Two Tank System with DeltaV	28
4.1.1 <i>Setting up Database</i>	28
4.1.2 <i>Setting the Control Module</i>	29
4.1.3 <i>Setting up I/O connections</i>	31
4.1.4 <i>Creating the HMI</i>	36
4.2 Setup of Serial Two Tank System with MATLAB.....	38
4.2.1 <i>Signal Configuration of the Inputs and Outputs</i>	40
4.2.2 <i>Bias transition of the Input signal for the pump</i>	41
4.2.3 <i>Setup for optimization of the Mathematical model</i>	41
4.2.4 <i>Connecting DeltaV with MATLAB</i>	42
4.3 Optimization Techniques	44
4.3.1 <i>Optimization of Input Signal</i>	44
4.3.2 <i>Optimization of Controller parameters using Model</i>	46
4.3.3 <i>The MIT Gradient Descent Rule</i>	48
5 Results	51
5.1 Tuning of the primary process' controller	51
5.1.1 <i>Simulating the Process with the same parameters for the DeltaV system</i>	53
5.1.2 <i>Simulation of both identical process in Simulink with Same control Parameters</i>	54
5.2 Optimization of the Model Parameters	55
5.3 Optimization Techniques	58
5.3.1 <i>Optimization of the control Inputs</i>	58
5.3.2 <i>Optimization of the secondary control parameters using the Model</i>	58
5.3.3 <i>Optimization of the controller parameters using the gradient descent rule</i>	59
5.3.4 <i>Optimization of the controller parameters using gradient descent rule and limit switch</i>	61

6 Discussion 64

 6.1 Tuning of the Primary process 64

 6.2 Optimization of the mathematical model..... 65

 6.3 Optimization of the secondary controller parameters 65

 6.3.1 Optimization of the input signal 65

 6.3.2 Optimization of the secondary control parameters using the Model 66

 6.3.3 Optimization of the controller parameters using the gradient descent rule..... 66

7 Conclusion 68

8 Recommendations 69

References 70

Appendices 72

Nomenclature

DCS – Distributed Control System

DIY – Do It Yourself

HIL – Hardware in a Loop

HMI – Human Machine Interface

I/O – Input or Output

LabVIEW – Laboratory Virtual Instrumentation Engineering Workbench

LMAC – Learning Model Adaptive Control

MATLAB – Matrix Laboratory

MIT – Massachusetts Institute of Technology

MPC – Model Predictive Control

MRAC – Model Reference Adaptive Control

NI-DAQ – National Instruments Data Acquisition

NTNU – Norwegian University of Science and Technology

OPC – Open Platform Communications

OPC UA – Open Platform Communications Unified Architecture

PID – Proportional, integral, and Derivative

PI – Proportional and Integral

SISO – Single Input Single Output

STTS – Serial Two Tank System

TM – Trademark

USN – University of South-Eastern Norway

1 Introduction

In the context of control theory and control engineering, process control is a big part. New inventions and technologies have made process control and production techniques to become more and more automated. Process automation has become inevitable nowadays because of its advantages over traditional process manufacturing, from product quality to cost reduction [5]. With the introduction of a mathematical model that mimics a process, optimisation and analysis of the process have become easy and unavoidable. This concept has been further streamlined in recent years, and a new name is given, known as a digital twin. A digital twin represents a physical system digitally, ranging from process to entire production line and even from building to cities [6].

The digital twin's journey starts with experts in the field of mathematics or data science, who research the dynamic physical parameters of the process and produce a mathematical model that mimics the characters of the process. Then this mathematical model can be used to generate data to compare with the real process. There are multiple applications for the digital twin, from maintenance, operation, fault diagnosis, and early error detection, such as HIL (Hardware in a Loop), which helps test new sensors, controllers, and actuators before implementing it real system.

In contrast, a twin process is a physical system that acts identical to the other system. In the production and manufacturing industry, production lines with the same kind of processes are commonly used for manufacturing using the same types of machinery and controls. A single company can have multiple production facilities worldwide at different places to achieve the same output with the same quality. Moreover, an emphasis can be given to achieve the same production rate in all the production lines. There are many advantages of using the physical twin method compared to digital twin in the context of optimising the production line like the development of a digital twin take more time and money, and it is tough to compensate for the dynamic noises of the physical process to represent it in a mathematical model.

1.1 DeltaV

DeltaV is a trademark for the DCS system offered by an American company called Emerson, which is a complete automation system.

According to the official website of Emerson Electric;

“The DeltaV™ Distributed Control System (DCS) is a flexible and easily usable automation control system with advantages of lowering project risks and operational complexities. The DeltaV consists of a state-of-the-art set of products and services that increase plants performance using intelligent control, which is easy to operate and maintain.

The DeltaV DCS system adapts for the customer's needs with easy scaling capability and avoids scaling complexity. The DeltaV system is inherently integrated with advanced control, change management, engineering tools and more.”[7]



Figure 1-1: S Series power controller and I/O modules of the Emerson DCS system (Emerson Co, 2021).

1.2 Problem Description

The Digital twin technology has gained momentum over the years between the control technology space because of its advantages over traditional control strategies and the control processes' maintenance and operation. Hardware in Loop (HIL) is a unique phenomenon in the digital twin context, which uses a mathematical model of a process control system to represent and behave nearly or precisely like the real physical process, which is then used in relation with a physical process to test and optimise the real controllers before implementing it into the original process.

This technology has advanced to modelling the entire process in recent years, including all the control systems, sensors, and actuators into a single simulation. This has led to this technology's use to predict the processes behaviour that can be utilized to fine-tune the processes parameters and maintenance. However, as the simulations do not consider the factors such as wear and tear, noises, and environmental effects, they cannot precisely give the same result as the real processes.

Nevertheless, there is a need to optimise identical production processes to achieve efficiency and quality end product, saving time, money, and resources. The present technique uses the same controller parameters obtained from an exemplary tuned controller to optimise a parallel production line. Even though using the same controller parameters helps optimise the process in reference to a more significant timeline such as a day or an hour. Since even the identical process and controllers cannot behave exactly like the other due to uncontrollable and unknown factors, achieving optimisation within the timeline of seconds and even minutes is very hard.

Therefore in this paper, a novel idea is presented to achieve this goal. Where one process line is fine-tuned according to the performance criteria set by the company and other identical processes control parameters are calculated according to the error between the two processes outputs, and to optimise the parameters in such a way the error is minimised.

1.3 Aim and Objectives

As this is an entirely new idea and no previous research have not been done, the objectives underlined here are set according to the authors understanding and knowledge of the subjects and the process. The process is a Serial Two Tank model (STTS), which is considered a SISO model designed and built by USN. Two identical processes exist in the lab, and only one relates to a DeltaV control system, so it is extended to study one with DeltaV and one with MATLAB. The mention headings are defined as the sub-objectives of this thesis.

- Complete a literature review of different approaches for comparing and controlling processes to “mirror” each other (operate in a similar form).
- Learn the bases for the installation and configuration of the DeltaV system and connect it to the two tanks system in the process hall at USN (depending on availability).
- Learn the necessary configuration skills to monitor a process remotely using DeltaV.
- Alternatively, if DeltaV is not available, configure the tanks available in the USN with the LabVIEW or MATLAB Simulink and monitor the process.
- Test the proposed strategies simulating the two tank systems experimental rigs (using MATLAB or Emerson’s MIMIC, depending on availability).
- Test the proposed strategies experimentally on the two tanks systems experimental rigs (using an Emerson DeltaV system), one physically at process hall at USN and the other at Emerson’s installation in Porsgrunn.
- Complete and deliver the master thesis report.
- Prepare and deliver a thesis presentation.

The Detailed task description is attached in appendix A of this thesis.

1.4 Structure of the thesis report.

This thesis is structured to explain the background, theory, and experimental method consistently and clearly. This thesis report is composed of eight chapters. Chapter 01 Introduction explains the background behind selecting this topic as a thesis and a short introduction about the DeltaV DCS system. Chapter 02 narrates the previous studies and experiments about the parts of the methods used in this thesis experiment. Chapter 03 explains the mathematical modelling of the STTS. Chapter 04 explains the materials and methodology. Chapter 05, 06, 07 and 08 are results, discussion conclusion and Recommendations, respectively.

2 Literature Review

The literature review chapter emphasises the previous works done by multiple scholars and researchers in correlation with the parts of the work that has been done in this thesis. This chapter has been subdivided into four main sub-topics: the two-tank process, the Mathematical modelling, PID tuning of the controller and the optimisation algorithm, where multiple studies have been interpreted related to this thesis.

2.1 The Two tank Process

The two-tank process is a combination of two tanks that are connected in serial one after the other. The process is a SISO type model that has been studied extensively by many institutions worldwide. A two-tank process has been studied in the control theory area, especially with multiple controllers and control hardware like PID controller, MPC controller and DeltaV DCS systems. The paper from NTNU, "Learning by DIY Exploring the Pedagogical Potential of the Serial Two-Tank System in a Control Theory Context"[1], describes the basic procedure and techniques needed to build a similar model and how to find a mathematical model of the experiment setup. Figure 2-1 shows the Serial Two Tank System developed by NTNU. In chapter 5 of this paper, a detailed explanation of designing and implementing a physical two-tank serial system has been explained.

There are two tanks in the system, and the upper tank, known as tank one, is filled with water, either hot or cold, using computer-controllable valves. The lower tank, also known as tank two, is filled using the water from the tank 01 output that exit at the tank's bottom. A steady level is maintained in tank two. Using this experiment setup, various level and control problems can be evaluated and studied [8]. In this paper, a similar tank was used to study the control and

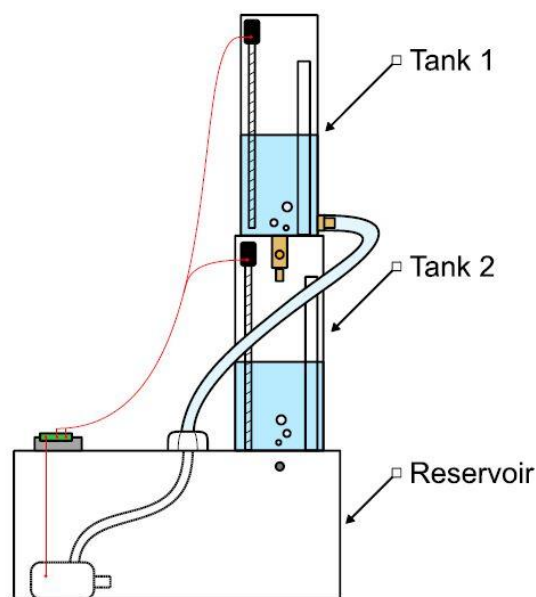


Figure 2-1: The Two tank model [1] .

system identification problem, and here the difference is that the level is controlled in tank two rather than tank one via a steady water supply for tank 02.

2.2 Mathematical modelling of a tank system

To create the mathematical formula of a serial two-tank process, the mass balance from the 1st principle of the theory is the ideal formula to start with. With the assumption, the tank's liquid is incompressible and has an exchange of mass and volume, which means the mass density does not vary across the whole process [1].

The standard expression for the mass balance is as follows.

$$\frac{dm(t)}{dt} = W_{in} - W_{out} + W_{generated} - W_{consumed} \quad (2-1)$$

The W denotes the mass flow rate with the unit in kg/s .

If there is no net loss or gain in the circulated waters' mass, the equation can be rewritten as

$$\frac{dm(t)}{dt} = W_{in} - W_{out} \quad (2-2)$$

Equation 2-2 can be rewritten using the relationship between the volumetric flow rate and mass.

$$W = \rho Q \quad (2-3)$$

Where Q is defined as the volume flow rate and calculated in m^3/s

The new equation becomes,

$$\dot{m} = \rho Q_{in} - \rho Q_{out} \quad (2-4)$$

Mass is a product of density and volume, and volume is a product of surface area and height so that the mass can be written as a product of density, surface area and height.

$$m = \rho * V = \rho * A[h(t)] * h(t) \quad (2-5)$$

By substituting equation 2-4 with equation 2.5, the following relationship between the water height of the tank and flow rate can be obtained.

$$\begin{aligned} A(h) * \dot{h} &= Q_{in} - Q_{out} \\ h(t) &\geq 0 \end{aligned} \quad (2-6)$$

For modelling a two-tank system, the dynamic behaviour of the 2nd tank in the series is influenced by the 1st tank [9]. To create the mathematical model of the system, the input from the pump is considered to be linear, and the dynamics can be represented by

$$Q_{pump} = C_p U \quad (2-7)$$

Where Q_{pump} is the volume flow rate from the pump in m^3/s

- C_p – the pump coefficient
- U – the input voltage to the pump

2.3 PID Tuning of the controller

Tuning is a method of finding the optimal controller parameters known as K_p , T_i , T_d . The tuning method can be used either in a real physical system or on a simulated system. The tuning methods can be used to tune processes with a time delay or a process with more than three dynamics order. The PID tuning cannot be done for a process where the time delay is much more significant than its constant time [10].

A good criterion for tuning the controller's parameters is that the control system or process must be fast and satisfactory, which are both contradictory in general. In other words, it can be said that if a controller is tuned for stability control, the controller becomes sluggish, and vice versa if a controller is tuned to fast controllability, the controller becomes unstable. Therefore, a controller must be tuned to compromise between these two criteria to get an acceptable controller parameter [10].

In this thesis, the controller used is a PI controller rather than a PID controller, and The PI controller function is the widely used controller for practical applications. A PI controller is merely a PID controller where the D (Derivative) term has been deactivated. The reason behind using a PI controller in favour of a PID controller is that the D – term amplifies random measurement noise leading to unexpected variations in the control signal [2].

In the PI controller, even though proportional gain gives the signal to the actuator to respond for the error elimination, the process may become sensitive to a higher gain causing the system to oscillate. The total error over the whole time is calculated to calculate the integral gain. Furthermore, to obtain the zero error, the controller output is adjusted [11].

The following equation gives a continuous-time PI controller.

$$u(t) = u_{man} + K_c e(t) + \frac{K_c}{T_i} \int_0^t e(\tau) d\tau \quad (2-8)$$

Where

- u = control signal
- u_{man} = Manual control signal
- $e = y_{sp} - y_m$ = control error
- y_{sp} = set-point

- y_m = process measurement
- K_c = controller gain
- T_i = integral time

The PI controller has been tuned empirically in the past years, for example, using the techniques described in [12]. This technique is very suitable for processes with very little information. However, unfortunately, this method has a significant disadvantage as it fundamentally provides poor damping. Hang et al. introduced the refined Ziegler – Nichols tuning method that helped to improve the performance of the PI controllers [13].

2.3.1 The good gain method

Another alternative method to Ziegler – Nichols ultimate gain method was proposed by Finn A Haugen, known as the good gain method [2]. In this method, unlike in the Ziegler- Nichols method, there is no need to introduce a sustained oscillation instance for a closed-loop system test.

The tuning process proposed by Finn A Haugen in his good gain method is as follows.

1. Set the controller in manual mode. Using the manual control (U_{man}), bring the process to the desired set-point or close to the specified set-point.
2. Turn the P controller and set the K_c value to 0 ($T_i = \infty$ and $T_d = 0$) and increase the value of gain until the control loop becomes stable under the controller set into automatic mode. The control loop is assumed to reach stability when the process variable corresponds to little overshoot and barely an observable undershoots, like shown in figure 2-2.

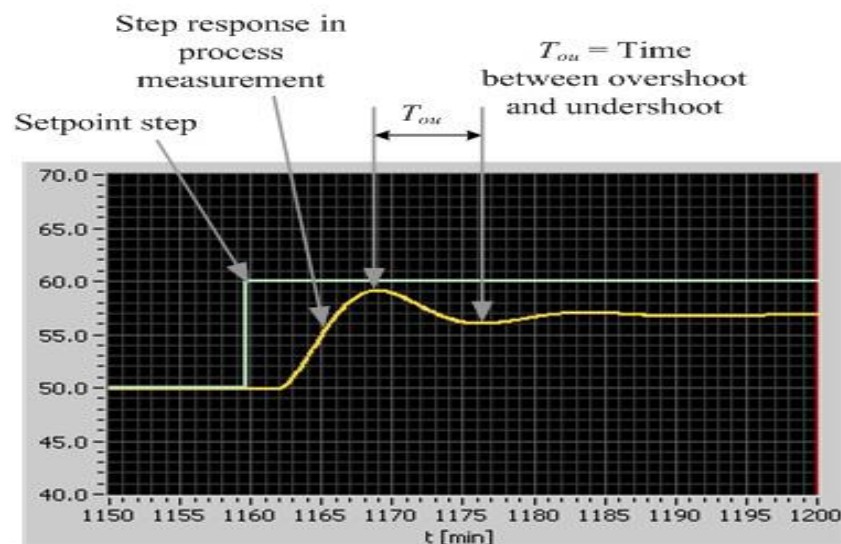


Figure 2-2: Reading off the time, T_{ou} , between the overshoot and the undershoot of the step response with P controller [2].

3. Measure the T_{ou} ; the time difference between the overshoot and undershoot and calculate the integral time T_i equals 1.5 times of T_{ou} .

$$T_i = 1.5T_{ou} \quad (2-9)$$

4. Turn on the I controller. As the I controller is turned on, the K_c value must be decreased to 80% of the previous value obtained from step 01.

$$K_c = 0.8K_{CGG} \quad (2-10)$$

5. Apply the value obtained for K_c and T_i calculated from the previous steps.
6. At last, verify the control system's stability by a step set-point change and verifying the stability is under the acceptable range.

There are multiple advantages of using the Good gain method over the ultimate gain method for tuning a PI controller in practical applications such as simplicity, acceptable performance, acceptable stability robustness, and this good gain method helps the PI controller to avoid severe process upset during the tuning process[2].

2.3.2 Skogestad Method

There are more than thousands of scientific papers related to the tuning of PID controllers. However, there must be a reason for a new publication. The primary justification is that PID controller is a widely used tuning algorithm in the industry, and the improvements in PID controller tuning will have a significant effect in the process industry. The second justification is that this method has presented simple guidelines and comprehensions to prove the understanding of a tuning method [3].

For the Skogestad method of tuning, multiple numbers of model parameters are determined either by using an open-loop step response execution of the process or from the transfer function of the model known as $K_c, T_{au}, T_c, T_{au2}$. Figure 2-3 shows the output of a first-order model with time delay and shows to find the model parameters.

Where

- K_c – the process gain
- T_{au} – the time delay of the model
- T_c – Time constant of the model
- T_{au2} – Second-order time constant (this is only applicable for a dominant second-order process where $Tau2 > Tc$)

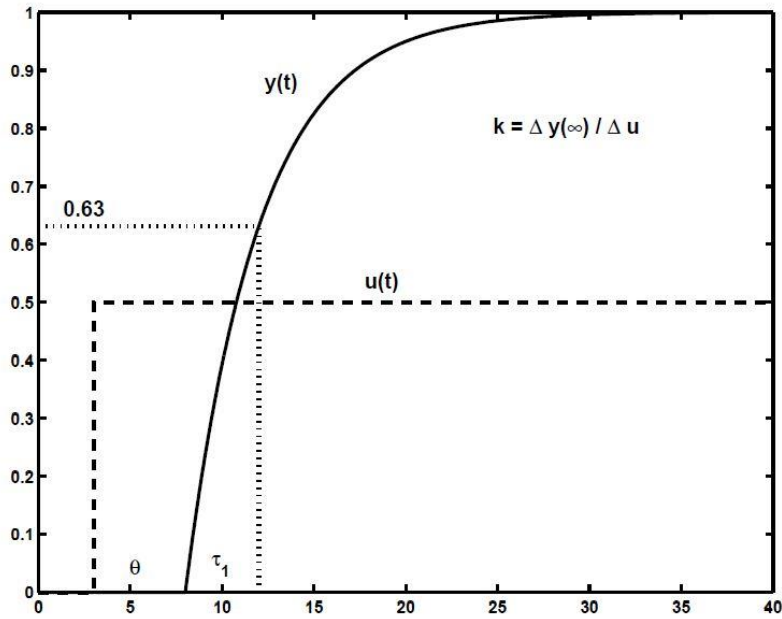


Figure 2-3 : Step response of a first-order with time delay system [3].

Figure:2-3 shows the step response of a first-order time-delay process and calculates the model parameters using the step response graph.

After finding the model parameters, using the tuning method equations proposed by Skogestad, the parameters of the PI controller can be calculated.

$$S = \frac{\Delta y}{\Delta x} \quad (2-11)$$

$$K_c = \frac{S}{\Delta U}$$

Where K_c is the process gain

S – Slope

Δy - change in Output

Δx – change in time

ΔU – change in the input signal

The Ultimate PI controller parameter values from the Skogestad setting are as follows.

$$K_p = \frac{1}{K_c(T_c + T_{au})} \quad (2-12)$$

$$T_i = 2(T_c + T_{au}) \quad (2-13)$$

$$K_i = \frac{K_p}{T_i}$$

$$T_d = 0 \quad (2-14)$$

2.4 Optimisation Algorithms

Industrial plants also become a subject of ageing, tear and wear, environmental disturbances, upgrades, and optimisation after its commissioning, leading to subtle changes in the plant's behaviour or processes [14]. Even identical systems that are commissioned can become variable in their behaviour. So, it will become less optimised when standard tuning parameters are used to control identical twins.

In the paper Synchronisation of industrial plant and digital twin presented by zipper et al., an optimisation algorithm has been proposed to synchronise a system's digital twin with its corresponding physical system. It states that the selection of the states that are needed to be taken into consideration depends on the formulation of the problem. Moreover, establish the general idea of using the responses from both systems digital twin and physical system to compare to an input from the controller.

In contrast, to the fixed control process where the controller parameters are fixed, the technique that can be used to test the theory of mirror process is known as adaptive control. The adaptive control mechanism measures process dynamics continuously and automatically compares them with the desired set-point, calculate the difference between the processes and adjust the controller variable until the optimal performance criteria are reached irrespective of the disturbances on the system [15].

There are two main approaches in the adaptive control theory; one is known as Learning Model Adaptive Control(LMAC), where a model of the process is tuned using online parameter estimation techniques, and the tuned model is used in the feedback control loop. This is a well-studied self-tuning control strategy. The second approach is known as Model reference Adaptive Control(MRAC). The control parameters are adjusted so that the system's closed-loop output matches the selected model according to a defined performance criterion [16].

2.4.1 The MIT rule

The MIT rule is mainly developed for the model reference adaptive control technique. This technique falls in the category of Non- dual adaptive control. The MIT rules were developed in the Massachusetts Institute of technology and can be applied in any practical system [17]. In the MIT rule, the cost function is defined as,

$$F(\theta) = \frac{e^2}{2} \quad (2-15)$$

Where

- e – error between the reference model and the process output.
- θ – the adjustable parameter.

The parameter Θ is tweaked until the cost function is minimized. To realize this, the parameter is changed in the direction of the negative gradient of the F . The adjustment rule is stated as

$$\begin{aligned}\frac{d\theta}{dt} &= -\gamma \frac{\partial F}{\partial \theta} \\ &= -\gamma e \frac{\partial e}{\partial \theta}\end{aligned}\tag{2-16}$$

Where

$\frac{\partial e}{\partial \theta}$ is called sensitivity derivative of the system and shows the dependence of the error in the adjustable parameter [15].

There are also multiple alternatives to choose the cost function of the MIT rule [17]. For industrial applications where the adaptation gain is crucial, and the value depends on the signal level. The MIT rule is modified as follows.

$$\frac{d\theta}{dt} = -\gamma \zeta e\tag{2-17}$$

where $\zeta = \frac{\partial e}{\partial \theta}$

also

$$\frac{dy}{dx} = \frac{\gamma \zeta e}{(\beta + \zeta^T \zeta)}\tag{2-18}$$

Where $\beta > 0$ is presented to prevent the equation for division of zero when $\zeta^T \zeta$ is small.

3 Mathematical Modelling of the Serial Two Tank Systems

Two Tank Systems

Mathematical modelling is the task of converting problems from a real application area to a controllable set of equations by utilizing the concepts of mathematics and languages [18]. This chapter explains building a mathematical model of the serial two-tank system shown in Figures 3-1, and figure 3-2 shows the P&ID diagram of the STTS. The mathematical model of both the systems is the same but the parameters may change.

3.1 The Serial Two-Tank Model



Figure 3-1: The serial two tank model setup.

3 Mathematical Modelling of the Serial Two Tank Systems

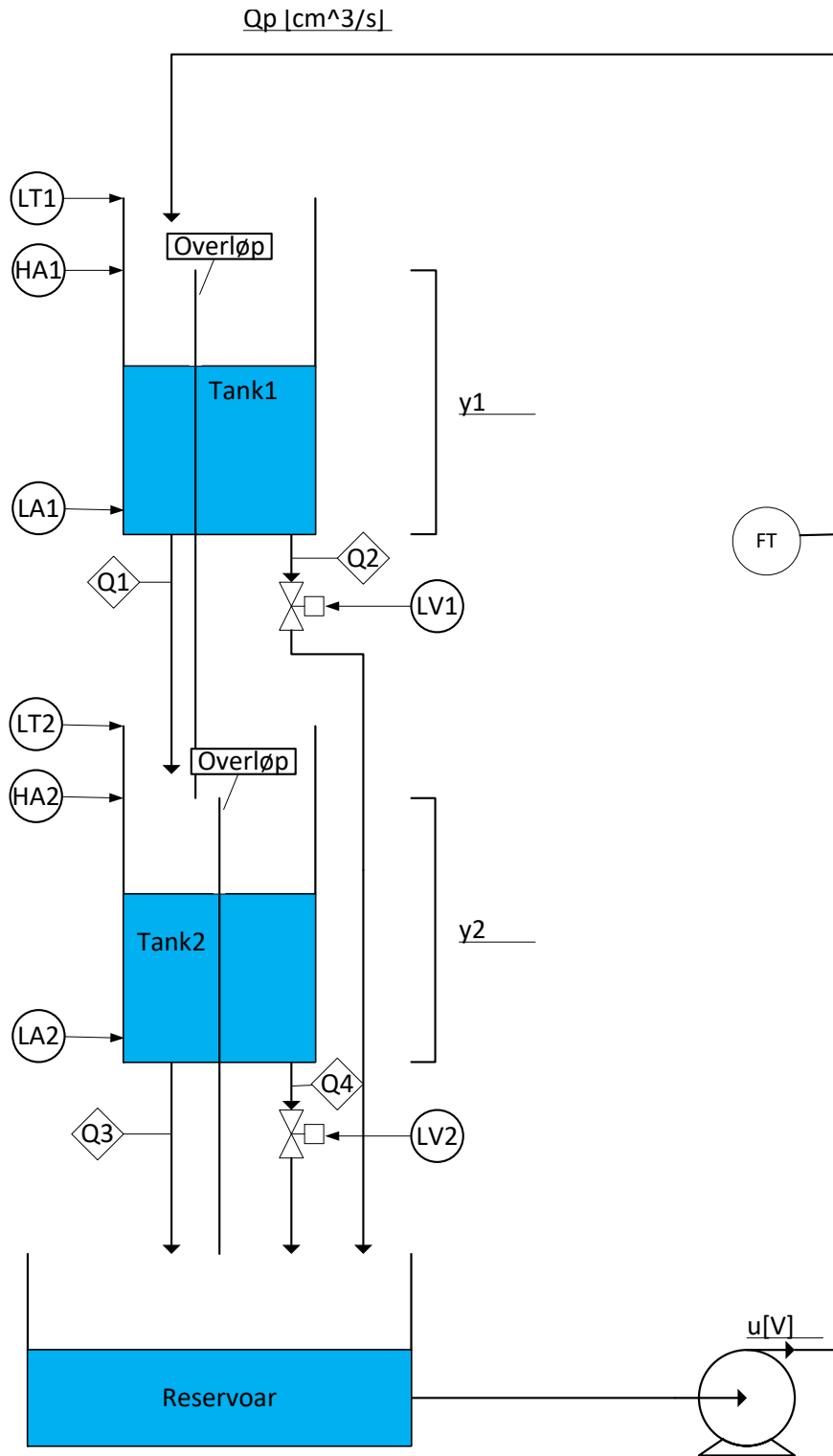


Figure 3-2: P&ID diagram of the Serial Two Tank system

3 Mathematical Modelling of the Serial Two Tank Systems

Developing a Standard tank model has been already discussed in the literature review part under the tank model; with the reference of the equation listed, the two tanks of the model is developed. The model consists of two tanks, so the tank model is applied to both tanks to model the entire system. The models for both of the tanks are as follows.

Model for the tank one

$$A_1 \frac{dy_1}{dt} = Q_p - Q_1 - Q_2 \quad (3-1)$$

Model of the tank two

$$A_2 \frac{dy_2}{dt} = Q_1 - Q_3 - Q_4 \quad (3-2)$$

Where

- A_1 [Area in tank1 = 113,1 cm²]
- y_1 [The level in tank1]
- Q_p [The flow to tank1]
- Q_1 [The flow out of tank1]
- Q_2 [The flow out of the solenoid valve]
- A_2 [Area in tank2 = 113,1cm²]
- y_2 [The level in tank2]
- Q_1 [The flow to tank2]
- Q_3 [The flow out of tank2]
- Q_4 [The flow out of the solenoid valve]

The Q_p flow of the tank can be represented with an Equation.

$$Q_p = C_p u \quad (3-3)$$

Where

- U - input voltage to the motor.
- C_p - estimated pump flow rate coefficient.

3.2 Model of the outputs flow rates (Fluid Resistance)

The W_{out} from tank one flows out from the bottom of the pipe the Q_1 and the output from the solenoid valve pipe Q_2 .

The flow rates from both of these outputs can be calculated from Torricelli's Principle. Fluids work against resistance when it flows through a valve or pipe opening. The mass flow rate in

3 Mathematical Modelling of the Serial Two Tank Systems

a resistive environment is happening due to the pressure difference between the cross-section of the resistance. The fluid resistance is calculated with the equation.

$$R = \frac{dp}{dW} \quad (3-4)$$

Figure 3-3 shows the dynamics of a valve opening. Torricelli's principle has been used to formulate an expression for the output flow rate. The potential energy of the water in the tank is converted into kinetic energy when the water flows out of the tank.

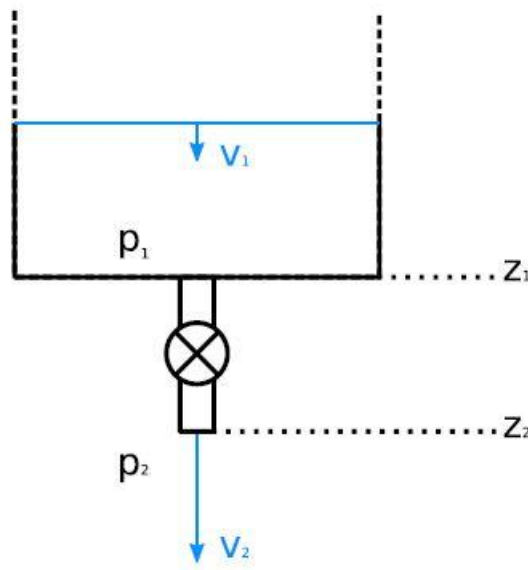


Figure 3-3: Dynamics of a valve parameters which are used to calculate the flow rate through an opening.

$$\frac{1}{2} (V_2^2 - V_1^2) + \frac{p_2 - p_1}{\rho} + (Z_2 - Z_1)g = 0 \quad (3-5)$$

Where

- V_i - liquid velocity in the reference points
- P_i - pressure in the reference points
- Z_i - Height of the reference points

It is assumed that the water velocity at the bottom of the tank is zero and the height difference between the reference points is negligible, that is $z_1 \approx z_2$. The equation (3-5) can be rewritten to calculate the output flow rate of the water from a valve or opening.

$$V = \sqrt{2 \frac{p_2 - p_1}{\rho}} = \sqrt{2 \frac{\Delta p}{\rho}} = \sqrt{2gh} \quad (3-6)$$

According to the volumetric flow rate definition, the equation (3-3) can be rewritten with respect to $Q = v * \alpha$, where α is the orifice area.

$$Q_{valve} = C\alpha \sqrt{2gh} \quad (3-7)$$

C represents the discharge coefficient factor where C lies between 0 and 1, 1 being the theoretical maximum and 0 being the theoretical minimum. The equation (3-7) is correct under the condition that the orifice area is small enough to neglect the pressure difference between the sides of the orifice.

3.3 Mathematical modelling of the Serial Two Tank Model

This subchapter focuses on modelling the mathematical model of the serial two-tank systems using the equations derived in chapter 3.1 and 3.2.

The rate of change of height in tank one and tank 02 is defined in equation 3-1 and 3-2. This can be further simplified as the electronically controlled valves are set off for the entire time of this experiment; the output from these valves are zero ($Q_2 = 0$, $Q_4 = 0$). So, the new equations are as follows.

Model of tank 01

$$A_1 \frac{dy_1}{dt} = Q_p - Q_1 \quad (3-8)$$

Model of tank 02

$$A_2 \frac{dy_2}{dt} = Q_1 - Q_3 \quad (3-9)$$

Using the equations (2-6), (2-7), (3.7), (3-8) and (3-9) and assuming the cross-sectional area of the tank are constant and equal, a differential equation is obtained that describes the dynamics of the STTS.

3 Mathematical Modelling of the Serial Two Tank Systems

$$\begin{aligned}
 \dot{\vec{x}} &= f(\vec{x}, u) \\
 &= \begin{pmatrix} \frac{1}{A_1}(C_p u - C_1 \alpha_1 \sqrt{2gx_1}) \\ \frac{1}{A_2}(C_1 \alpha_1 \sqrt{2gx_1} - C_2 \alpha_2 \sqrt{2gx_2}) \end{pmatrix} \\
 &= \begin{pmatrix} K_p u - K_1 x^{1/2} \\ K_2 x^{1/2} - K_3 x^{1/2} \end{pmatrix}
 \end{aligned} \tag{3-10}$$

Where $K_p = C_p/A_1$, $K_1 = C_1 \alpha_1 \sqrt{2g}/A_1$, $K_2 = C_2 \alpha_2 \sqrt{2g}/A_2$ and $K_3 = C_3 \alpha_3 \sqrt{2g}/A_2$ all are considered to be positive definite.

As the height of the water level in tank-01 and tank-02 can be measured directly, the output is as follows.

$$\vec{y} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \tag{3-11}$$

3.3.1 The transfer function of the model

From the equation 3-13, the transfer function of the model was derived.

The transfer function for the tank-01

$$H(s) = \frac{Y_1(s)}{U(s)} = \frac{\frac{b}{z_1}}{\frac{1}{z_1} \cdot s + 1} \tag{3-14}$$

The transfer function for tank-02

$$H(s) = \frac{Y_2(s)}{U(s)} = \frac{\frac{z_2}{z_3}}{\frac{1}{z_3} \cdot s + 1} \tag{3-15}$$

The open-loop transfer function of the entire model

$$H(s) = \frac{Y_2(s)}{U(s)} = \frac{\frac{bz_2}{z_1 z_3}}{\left(\frac{1}{z_3} \cdot s + 1\right) \left(\frac{1}{z_1} \cdot s + 1\right)} \tag{3-16}$$

Here the notations z_1, z_2, z_3 represents the same parameters as from the equation 3-13.

3 Mathematical Modelling of the Serial Two Tank Systems

Table 3-1: The model parameters of the primary process.

Parameters	Symbol	Value	Unit
General			
Gravitational acceleration	g	9.81	m/s^2
Pump parameters			
Estimated pump flow rate coefficient	$C_{p,p}$	1.98681e-05	$m^3/s/v$
Operating Voltage		0-5	v
Tank 01			
Height of tank	$h_{1,p,max}$	0.265	m
Diameter of tank	$dt_{1,p}$	0.2	m
Cross section of tank	$A_{1,p}$	0.01131	m^2
Diameter of the orifice of the valve opening		0.007	m
Cross-section of orifice	$\alpha_{1,p}$	3.8485e-05	m^2
Valve discharge coefficient	$C_{1,p}$	0.8311	--
Level transmitter output at ($h_1=h_{max}$)	$V_{max,p}$	5	v
Level transmitter output at ($h_1=0$)	$V_{min,p}$	1.25	v
Tank 02			
Height of tank	$h_{2,p,max}$	0.265	m
Diameter of tank	$dt_{2,p}$	0.2	m
Cross section of tank	$A_{2,p}$	0.01131	m^2
Diameter of the orifice of the valve opening		0.007	m
Cross-section of orifice	$\alpha_{2,p}$	3.8485e-05	m^2
Valve discharge coefficient	$C_{2,p}$	0.7740	--
Level transmitter output at ($h_1=h_{max}$)	$V_{max,p}$	5	v
Level transmitter output at ($h_1=0$)	$V_{min,p}$	1.25	v

3 Mathematical Modelling of the Serial Two Tank Systems

Table 3-2: The model parameters of the secondary process.

Parameters	Symbol	Value	Unit
General			
Gravitational acceleration	g	9.81	m/s^2
Pump parameters			
Estimated pump flow rate coefficient	$C_{p,s}$	1.40621e-05	$m^3/s/v$
Operating Voltage		0-5	v
Tank 01			
Height of tank	$h_{1max,s}$	0.265	m
Diameter of tank	$dt_{1,s}$	0.2	m
Cross section of tank	$A_{1,s}$	0.01131	m^2
Diameter of the orifice of the valve opening		0.007	m
Cross section of orifice	$\alpha_{1,s}$	3.8485e-05	m^2
Valve discharge coefficient	$C_{1,s}$	0.6673	--
Level transmitter output at ($h_1=h_{max}$)	$V_{max,s}$	5	v
Level transmitter output at ($h_1=0$)	$V_{min,s}$	1.25	v
Tank 02			
Height of tank	$h_{2max,s}$	0.265	m
Diameter of tank	$dt_{2,s}$	0.2	m
Cross section of tank	$A_{2,s}$	0.01131	m^2
Diameter of the orifice of the valve opening		0.007	m
Cross-section of the orifice	$\alpha_{2,s}$	3.8485e-05	m^2
Valve discharge coefficient	$C_{2,s}$	0.6591	--
Level transmitter output at ($h_1=h_{max}$)	$V_{max,s}$	5	v
Level transmitter output at ($h_1=0$)	$V_{min,s}$	1.25	v

4 Methodology

4.1 Setup of the Serial Two Tank System with DeltaV

4.1.1 Setting up Database

For the setup of STTS with DeltaV, all components were wired and ready. The only thing that has to be done for implementing the system with DeltaV DCS is connecting the wires with the DeltaV hardware component. As the first step, a database was created in the DeltaV database administration software, shown in figure 4-1. For the ease of the students, a template with all the hardware configuration that was already developed was available designed by the university. This created a shell program with no configuration.

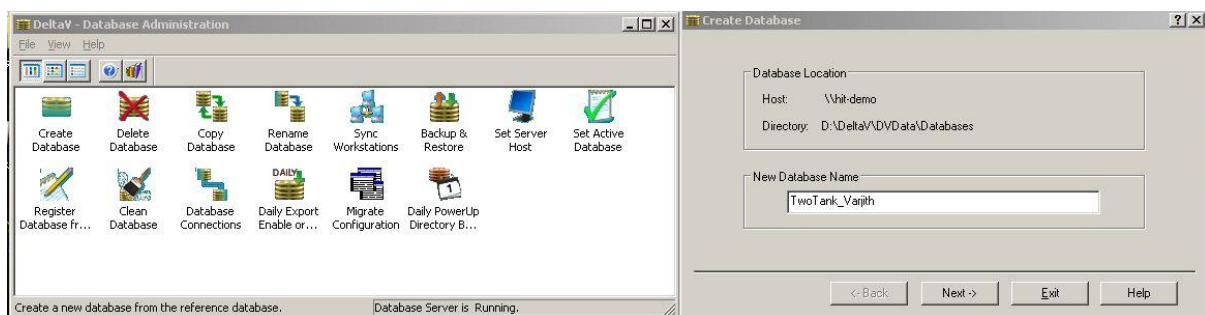


Figure 4-1: Procedure of creating a new database in the DeltaV Explorer software.

As shown in figure 4-2, after the database was established, it was selected as the active database to start the system configuration in the DeltaV Explorer software.

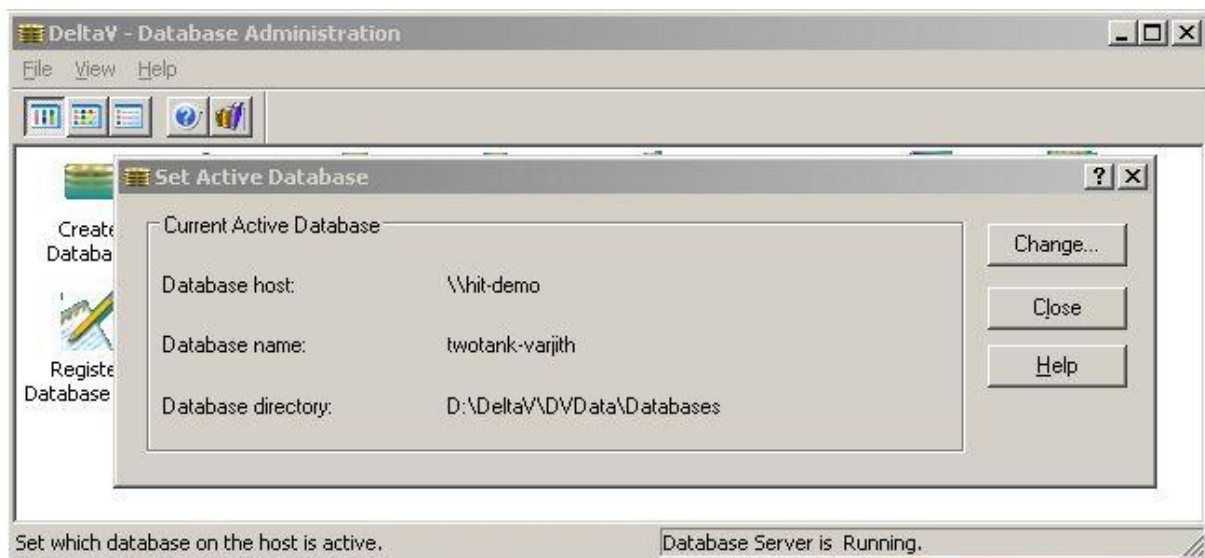


Figure 4-2: Activation of the created database as the current database.

4.1.2 Setting the Control Module

The DeltaV explorer has been equipped with inbuilt controllers like PID and MPC. So in order to control the process with a PI controller. A PID control module was copied into a newly created area under control strategies space, shown in figure 4-3.

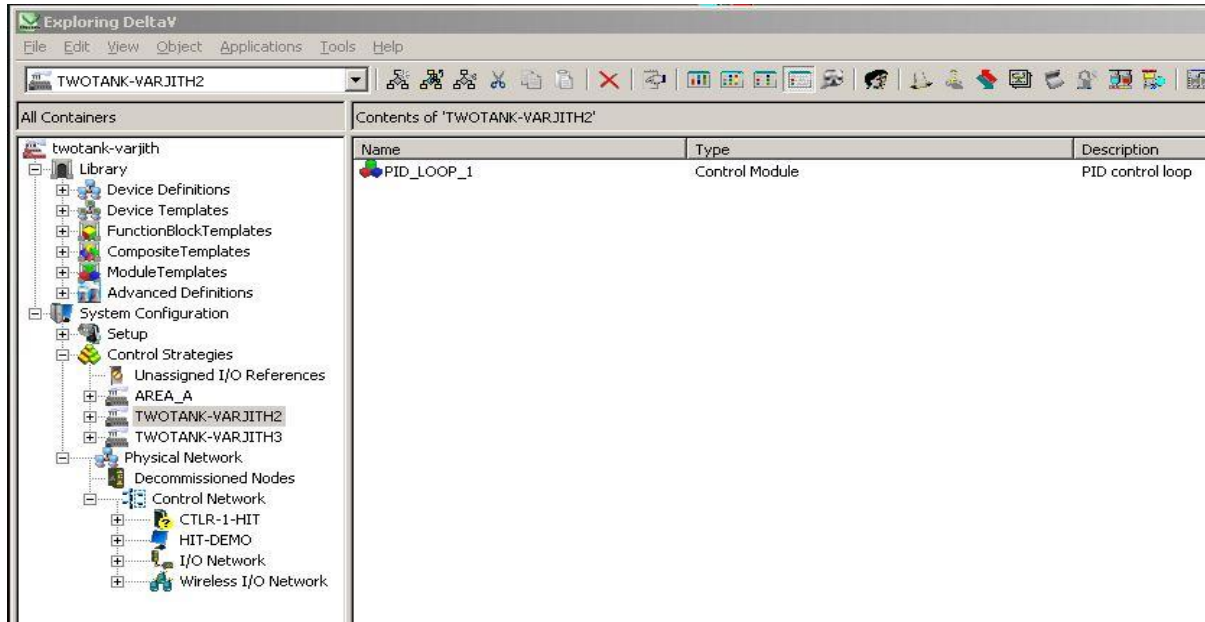


Figure 4-3: Implementation of the PID controller into the control strategies area.

After pasting the controller, the controller was edited using the control studio option. Two additional blocks were added to this module, shown in figure 4-4, namely the Analog Input block and scaler block. Figure 4-5 shows the procedure of scaling the level transmitter input signal by changing the configuration of the scaler block properties. This procedure is inevitable as the level transmitter gives 20mA and 4mA when the tank is empty and full, respectively.

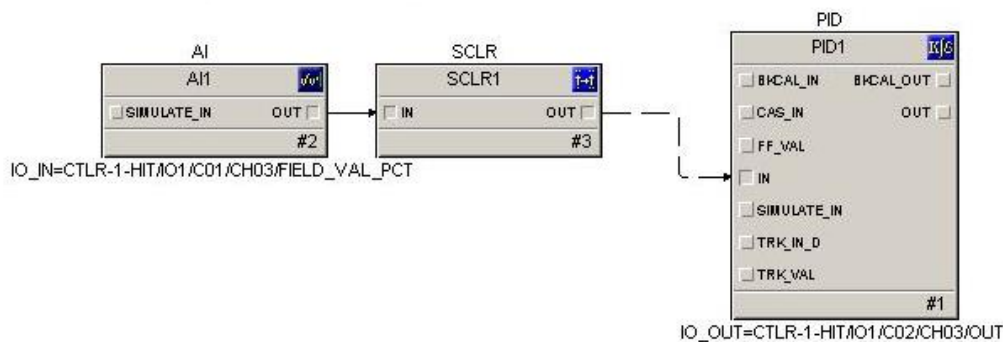


Figure 4-4: The diagram of the implemented PID control loop.

Figure 4-5: Configuration diagram of scale conversion to the PID controller input.

The controller's input and output were defined in the reference given to the Analog I/O signals of the process, as seen in figure 4-6.

Name	Value
IO_IN	
IO_OUT	TLR-1-HITC02CH03/OUT
IO_READBACK	

Figure 4-6: Assigning of input and output signal for the PID control loop.

After completion of the PID controller implementation, the control strategy was downloaded into the DeltaV module.

4.1.3 Setting up I/O connections

The process consists of several I/Os to set up. All the inputs and outputs are wired into the DCS system prior to the setup. The only thing left is activating it in the software, as seen in figure 4-8. This is done by selecting the I/O configuration option as shown in figure 4-7 under the application tab, selecting the desired channels, and enabling them.

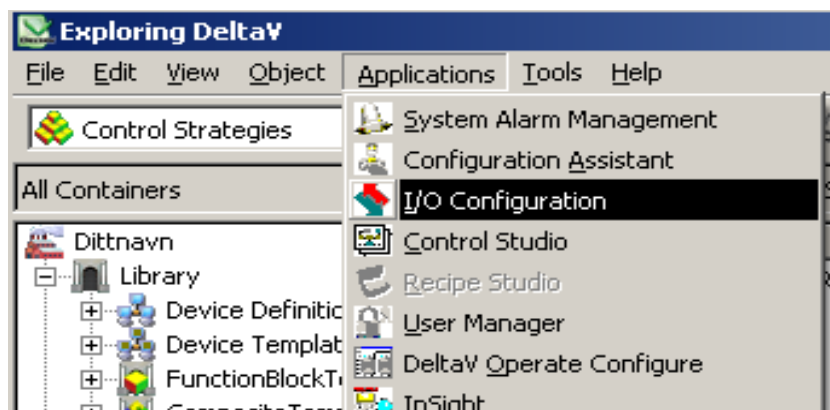


Figure 4-7: Pathway of I/O configuration option in the DeltaV explorer.

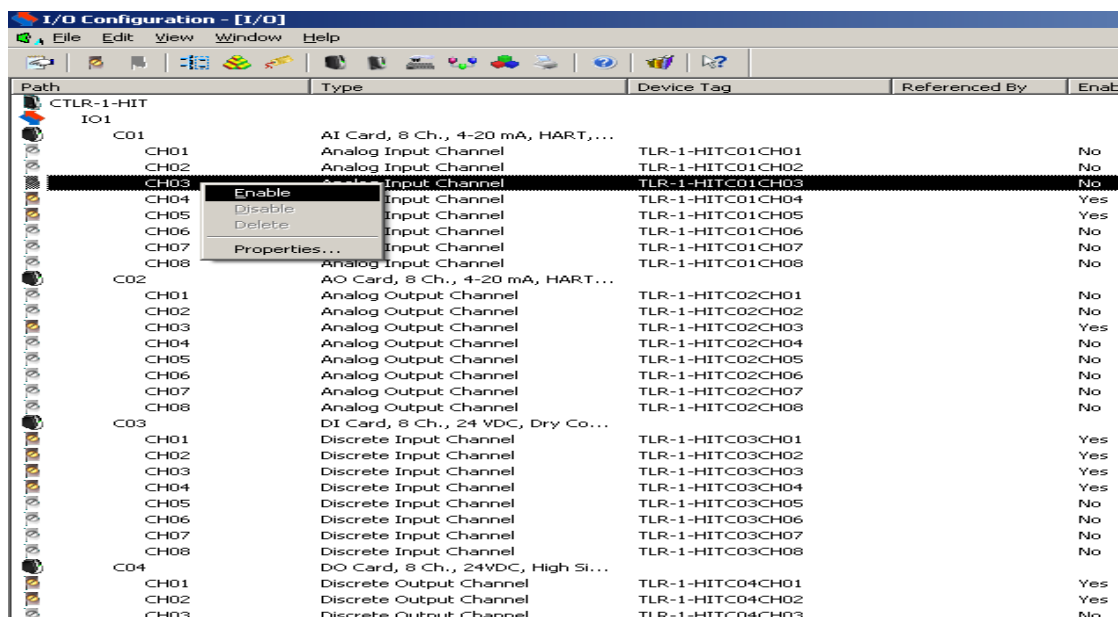


Figure 4-8: Shows the procedure of enabling the input and output ports of the I/O cards of the DCS system.

The Input and output channels configured are as follows.

Table 4-1: shows the description of the input and output signal configured in the DCS DeltaV system I/O cards.

IN	CARD/CHANEL	DESCRIPTION
LT1	C01CH03	LEVEL TANK1
LT2	C01CH04	LEVEL TANK2
FT1	C01CH05	FLOW
HA1	C03CH01	HIGH ALARM TANK1
LA1	C03CH02	LOW ALARM TANK1
HA2	C03CH03	HIGH ALARM TANK2
LA2	C03CH04	LOW ALARM TANK2
OUT		
P1	C02CH03	PUMP
LV1	C04CH01	SOLOINOID VALVE TANK1
LV2	C04CH02	SOLOINOID VALVE TANK2

Table 4-1 shows the configuration of the inputs and the outputs of the process with the Emerson’s DCS system. After enabling the nodes in the software DeltaV explorer, as shown in figure 4-8. A new area was configured under the control strategies section to implement all the I/O connections.

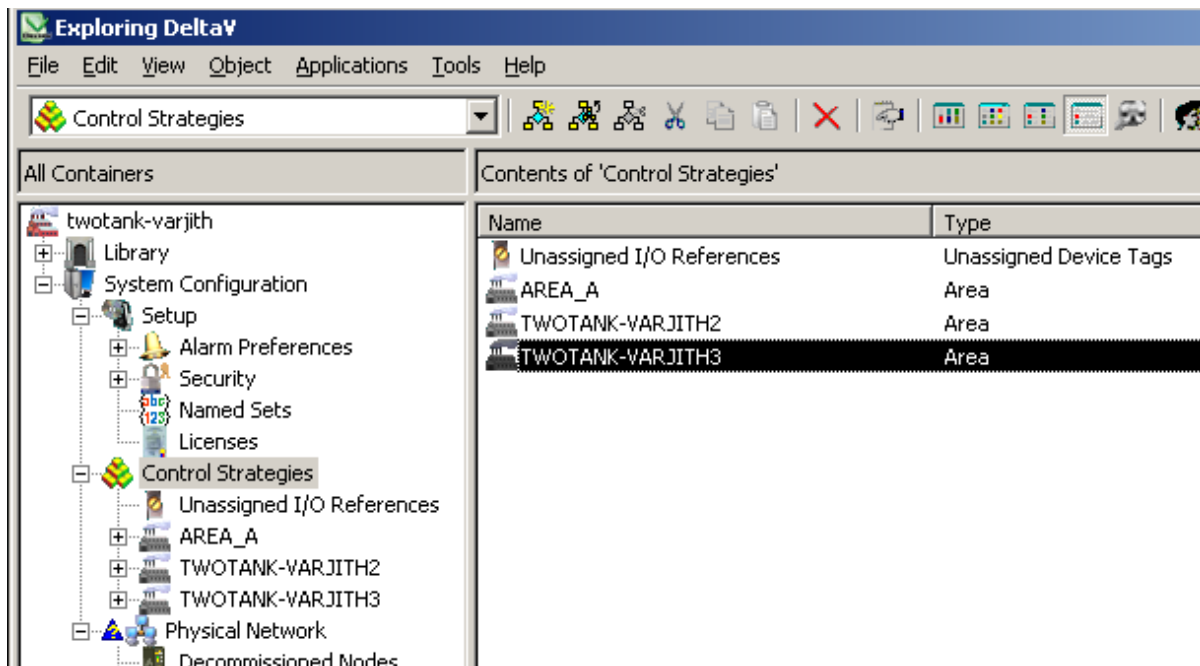


Figure 4-7: Creation of the second control strategies area for configuration of I/O signals.

After creating a new area called TWOTANK-VARJITH3, a new control module was created under this area to assign the I/O nodes to the controller. The procedure is shown in figure 4-10 and 4-11.

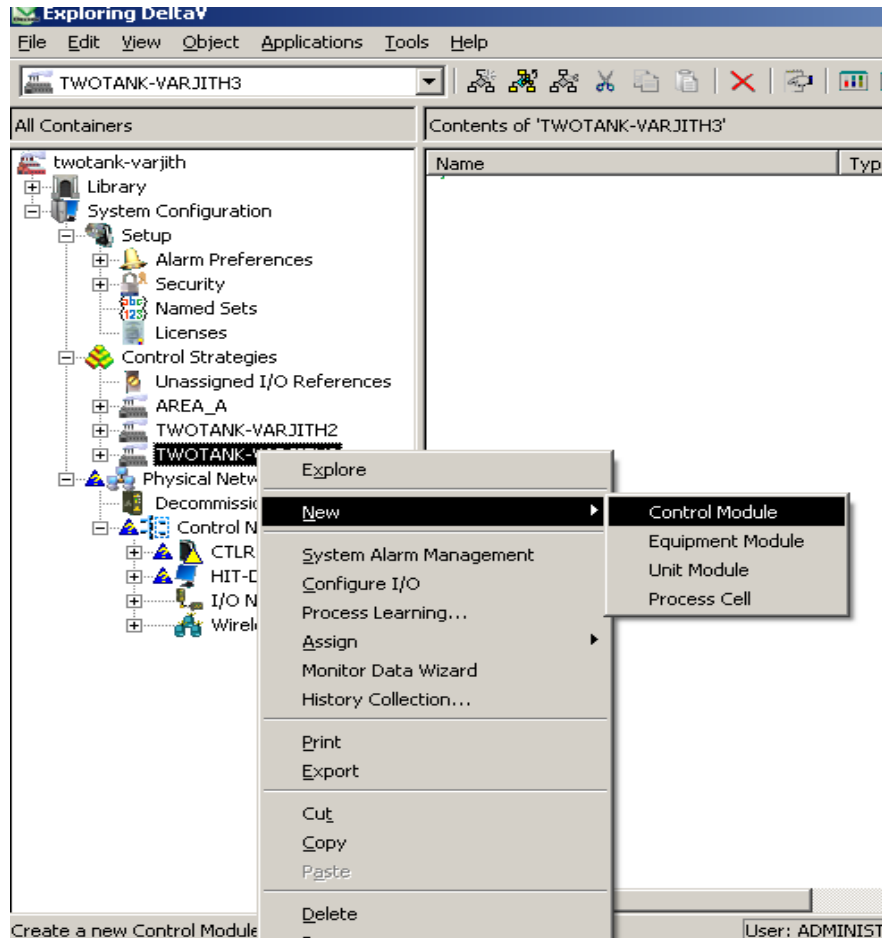


Figure 4-8: Shows the procedure of adding new control module into the second control strategies area.

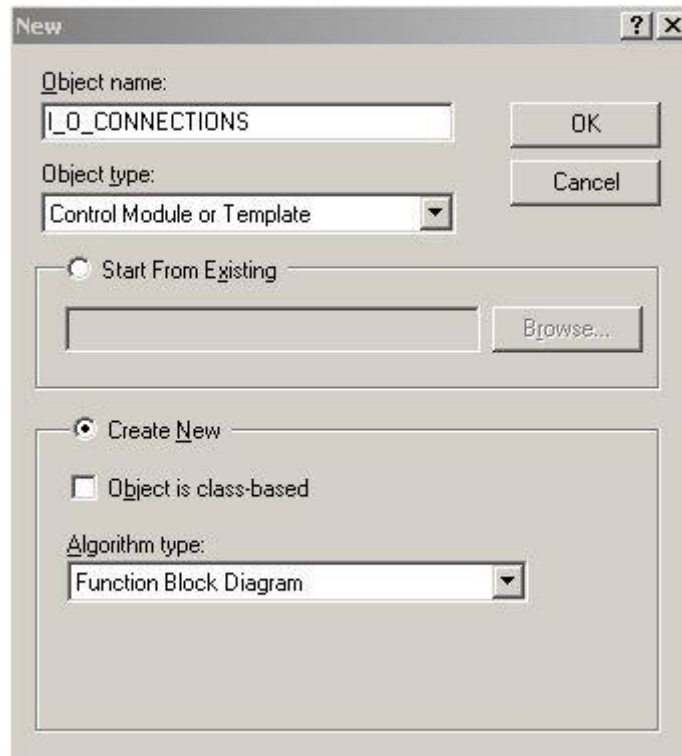


Figure 4-9: Configuration of a new control module for I/O connections.

The control module I_O_CONNECTIONS was edited using the control studio to configure all the inputs and outputs of the process. For each connection, a block is used, and the blocks were assigned to the appropriate signal tags

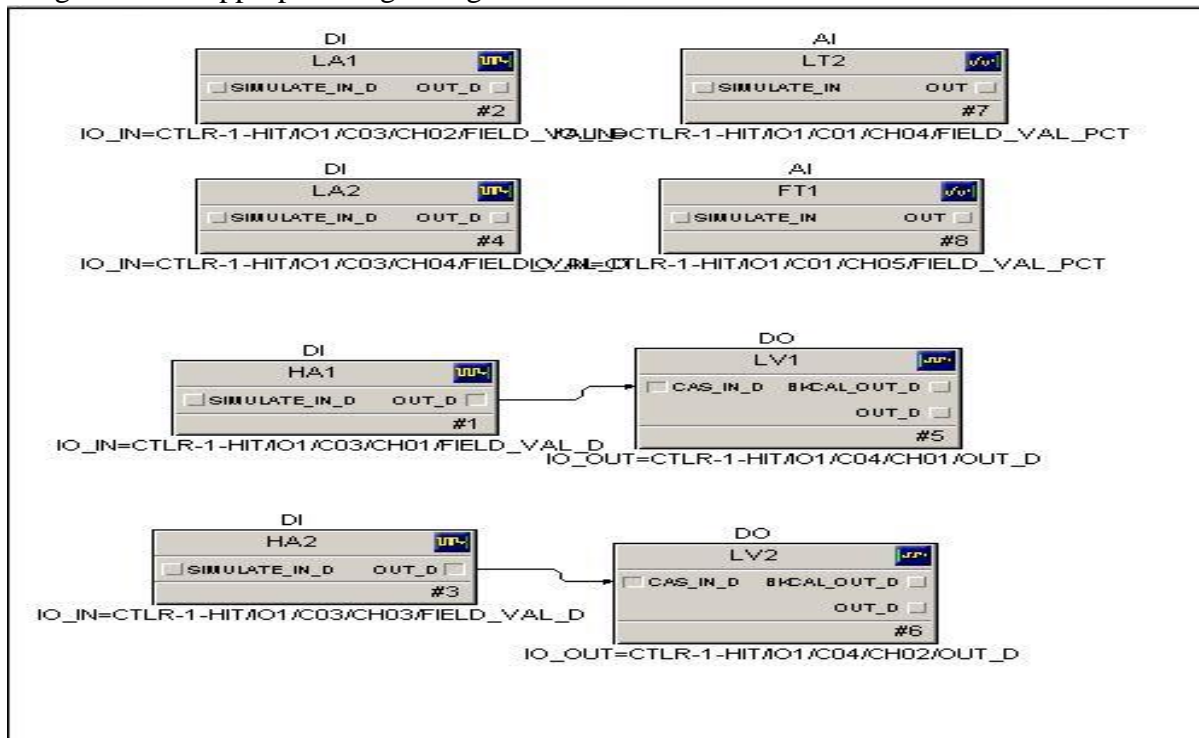


Figure 4-10: The configured control module with all the I/O blocks.

For the operation of the solenoid valves, their mode of operation must be changed from Cascade to Manual. This is done by selecting the mode of operation under the properties option for each valve. Figure 4-13 shows how the procedure was done.

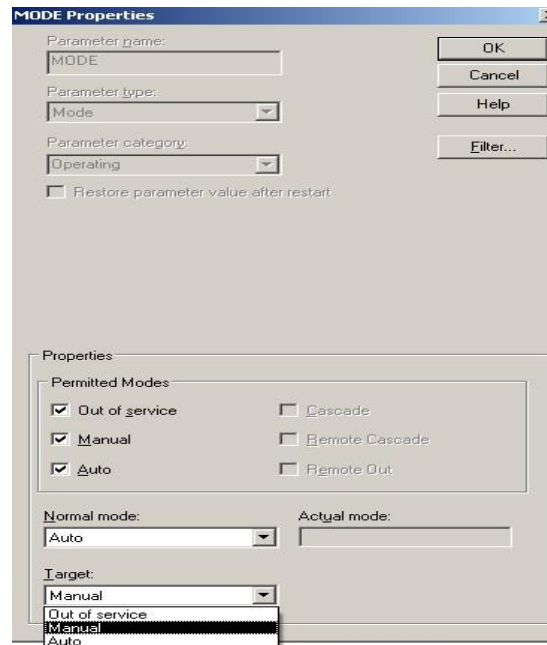


Figure 4-11: Shows the procedure of changing the valve operation mode.

After setup was done, the control modules were downloaded into the alarm and events section of the DCS system to activate the alarms. This was done by dragging the desired control modules into the alarm and events section under the HIT-DEMO node. After the configuration of alarm and events, the configured modules were downloaded into the physical network memory to activate the software with the hardware.

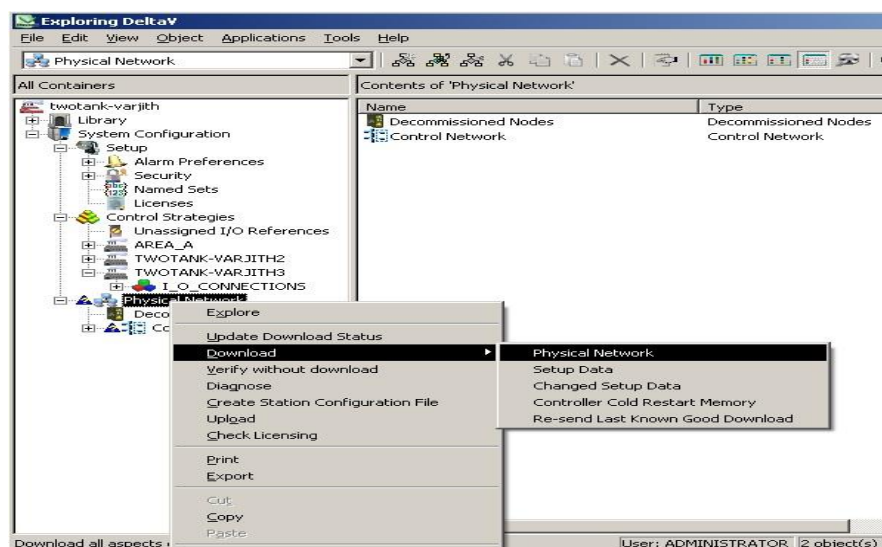


Figure 4-12: shows the procedure of downloading the configured control strategies into the physical network.

4.1.4 Creating the HMI

A Human Machine Interface must be created in order to interact with the module. HMI can be created using the DeltaV Operate Configure Software. Figure 4-15 shows the pathway of the “DeltaV Operate Configure” option in DeltaV Explorer.



Figure 4-13: shows the path of DeltaV operate configure panel in the DeltaV Explorer.

All the necessary components were added to the figure template, which was readily available from the templates. Figure 4-16 shows the HMI after all the configuration was done.

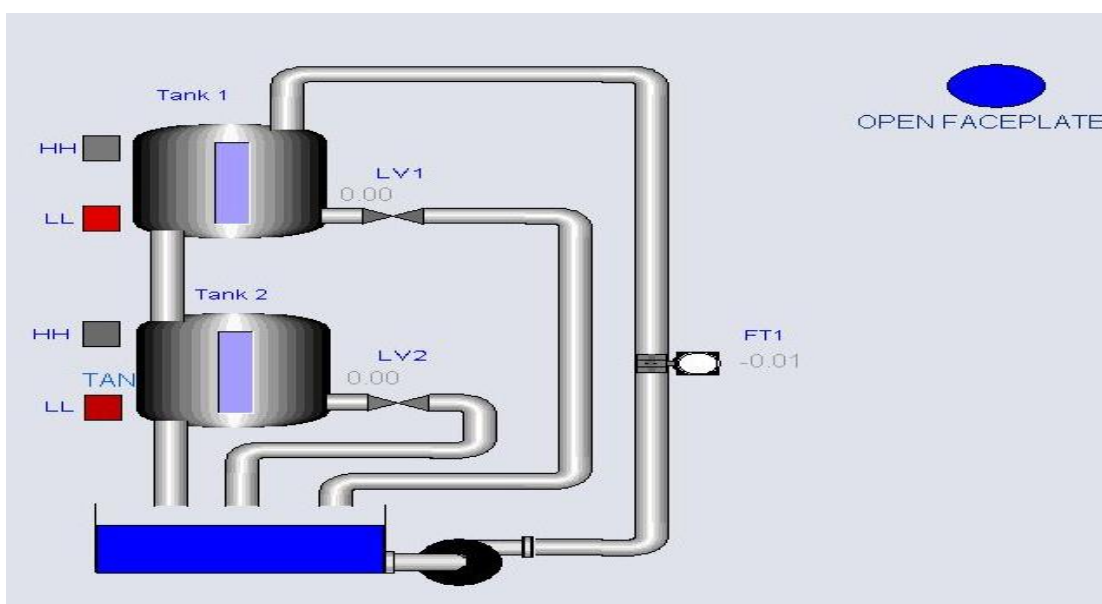


Figure 4-14: Configured HMI.

Figure 4-17 shows the configuration of the faceplate with the PID control. The faceplate acts like the dashboard for the controller, and this is where the set-point and process variable and other parameters can be viewed and changed. So, this faceplate must be connected with the PID loop to display and configure the parameters. This is done by assigning the faceplate to the PID control loop.



Figure 4-15: Setting up faceplate with the PID loop.

For the tanks, values from the level transmitter were assigned to show water level change and assign the colour of the variable change. In addition, the pump dynamo also set up to show the pump mode by the change in colour. The same procedure was done to show the operation of the valves and High or low level of alarms.

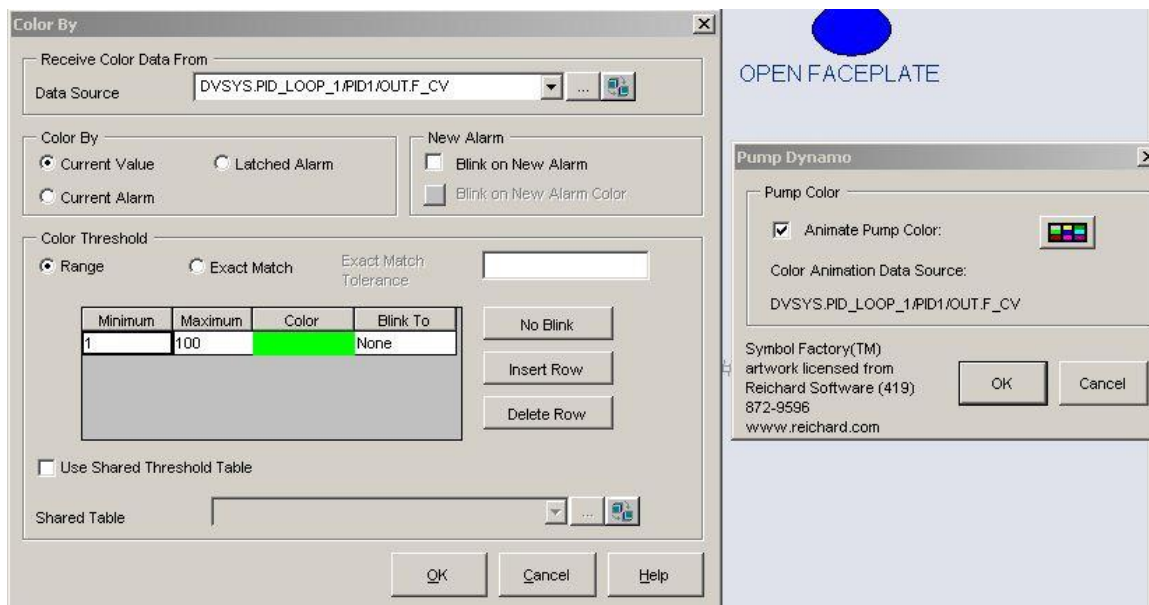


Figure 4-16: Configuration of the pump dynamics color change

Datalinks were created into the HMI to display the process variables numerically, like the flow rate of the pump, the states of the solenoid valves.

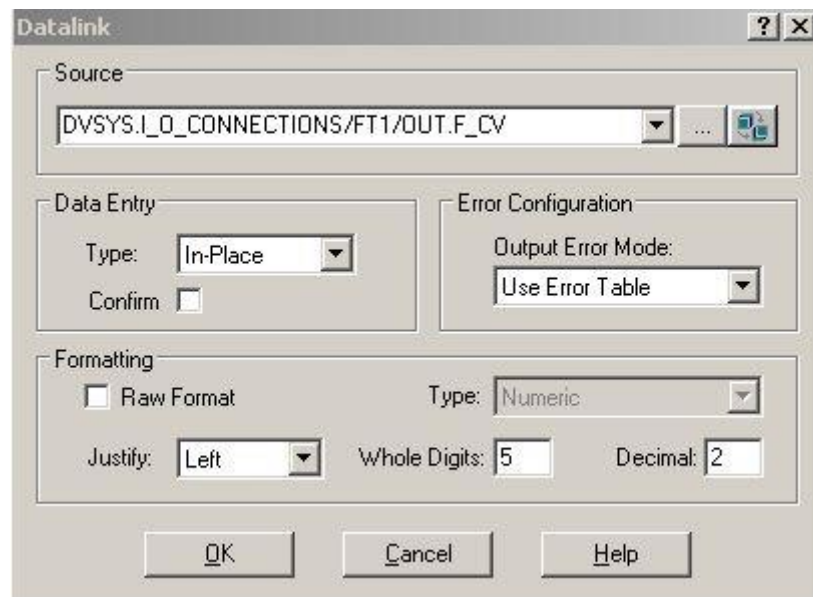


Figure 4-17: Configuration of the datalinks to show the status of the variables in the HMI.

After the completion of HMI, by pressing the control plus W button, the HMI was transferred from configuring state to operational state.

4.2 Setup of Serial Two Tank System with MATLAB

As for the present situation of pandemic prevails in the country and difficulties in acquiring an identical Emerson's DCS control system to configure with the second process, it was decided to use MATLAB as the control system for both processes and to evaluate the proposed theory. Therefore, the processes were connected with the MATLAB Simulink using a National Instruments data acquisition device called NI-DAQmx 60001. This device was configured with an operating voltage of 0 to 5 voltage.

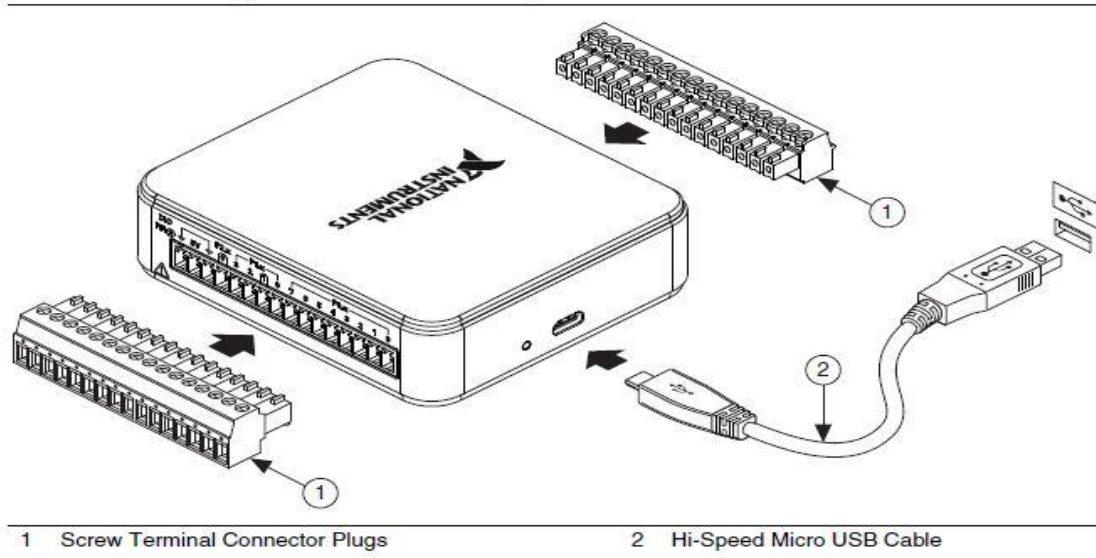


Figure 4-19: Components of the National Instruments data acquisition device NI-DAQmx 6001[4].

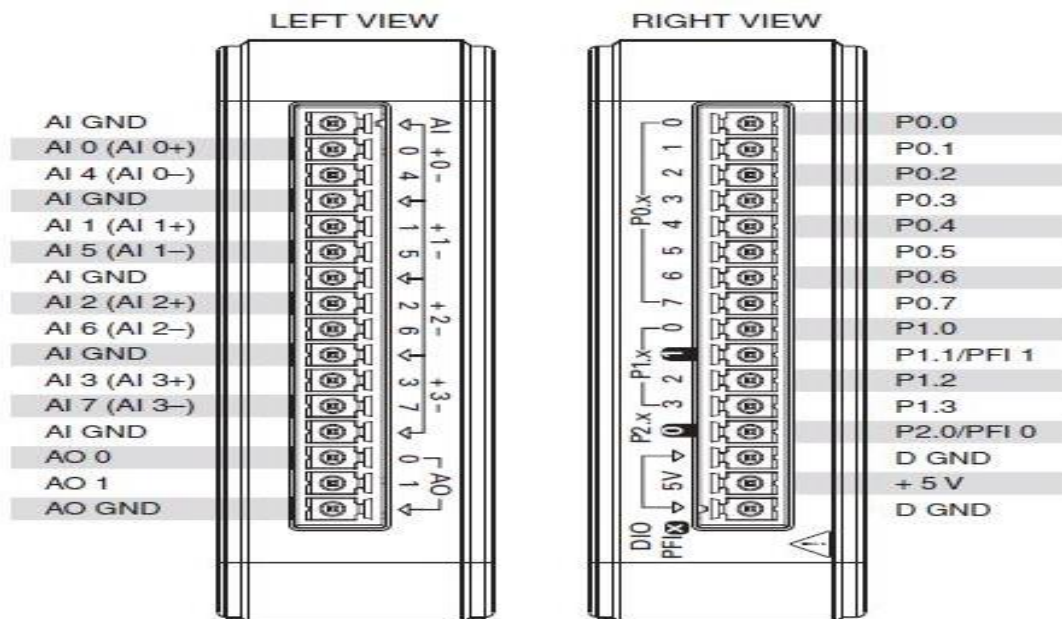


Figure 4-18: Pin configuration for the inputs and outputs ports of the NI-DAQmx 6001[4].

For this experimental setup, the solenoid valves were not used and left completely closed for the whole duration of the experiment. Therefore, the inputs and the outputs of the data acquisition device are as follows.

Table 4-2: Details of the input and output signal configuration for the NI-DAQmx 6001 device.

Input type	Name	Pin Configuration	Description
Analog Input	LT1	AI 0 (AI 0+) AI 4 (AI 0-)	Level Transmitter for tank 01 of the identical primary process
Analog Input	LT2	AI 1 (AI 1+) AI 5 (AI 1-)	Level Transmitter for tank 02 of the identical primary process
Analog Input	LT3	AI 2 (AI 2+) AI 6 (AI 2-)	Level Transmitter for tank 01 of identical secondary process
Analog Input	LT4	AI 3 (AI 3+) AI 7 (AI 3-)	Level Transmitter for tank 02 of identical secondary process
Analog Output	PMC1	AO 0 AO GND	The pump of the identical Primary process
Analog output	PMC2	AO 1 AO GND	The pump of the Secondary process

4.2.1 Signal Configuration of the Inputs and Outputs

Figure 4-22 shows the change of level transmitter output voltage with respect to the water level of tank 01. The input from the level transmitter was 5v when the tank was empty and 1.25v

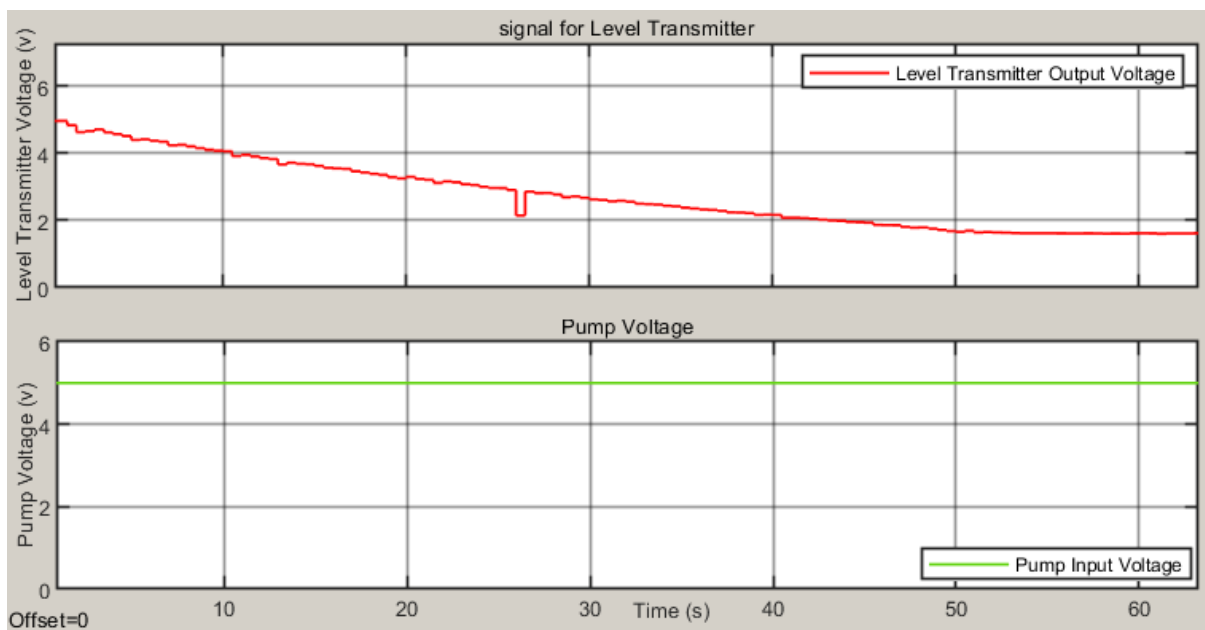


Figure 4-20: Output of level transmitter of tank one when the tank gets empty from full of the primary process.

when the tank was full. Therefore, a linear relationship was calculated to transfer the voltage into cm.

The linear relationship calculated are as follows.

$$Y = -7.64399147 * x + 38.1075907 \quad (4-1)$$

Where x is the voltage and y is the height of water in cm.

4.2.2 Bias transition of the Input signal for the pump

Even though the PI controller was set to calculate the output signal within the range of 0 to 5v range. The pump has a bias of 3.1 voltage, where the pump does not give any input flow into the tanks. So, in order to overcome this problem. A linear relationship was built in such a way that the pump works in the range of 0v to 5v.

The relationship is as follows.

$$y = 0.34 * x + 3.1 \quad (4-2)$$

where X is the output from the controller and y is the output from the offset block to the DAQ device.

4.2.3 Setup for optimization of the Mathematical model

Figure 4-23 shows a data logging Simulink application was developed to log the variables data to optimize the parameters of the models for both the system. A mathematical model was developed as a function block in the Simulink space using the equations 3-10 and 3-11.

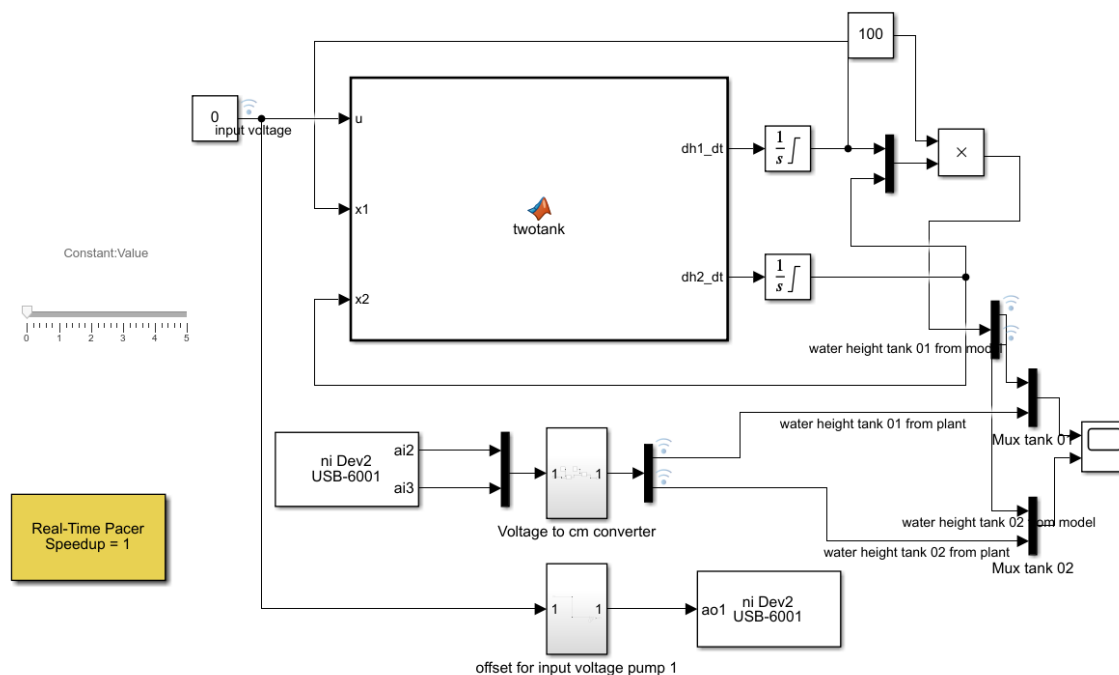


Figure 4-21: MATLAB Simulink setup for the simulation of model and the process simultaneously.

The developed model was simulated along with the actual process, and the outputs were logged into a file, which was later used to optimize the model's parameters.

4.2.4 Connecting DeltaV with MATLAB

The only option available to transfer the variables from the DeltaV into MATLAB is through an OPC UA connection. An OPC connection is a set of rules and standards designed to structure the communication protocols between control devices from different manufacturers. OPC UA is the newest product from the OPC Foundation and consists of all the features in one framework. The DCS system with windows OS is already equipped with the Matrikon OPC software as a standard. The only step needed is to configure the tags in the OPC explorer to extract the desired variables into the OPC server.

Figure 2-14 shows the configuration dialogue of adding tags from the DCS system. Under the available items in Server. It can be seen that the control strategies modules, I/O connections and the diagnostics parameters are available to select from.

Figure 4-25 shows, the PID controller variables, the water height from the tank and the control input for the pump are tagged in the "OPC.DeltaV" server. Like this, the necessary variables can be transferred into the OPC server. Moreover, using the Matrikon OPC tunneller configuration option shown in figure 4-26, the configured variables in another node can be transferred into the OPC explorer for further analysis.

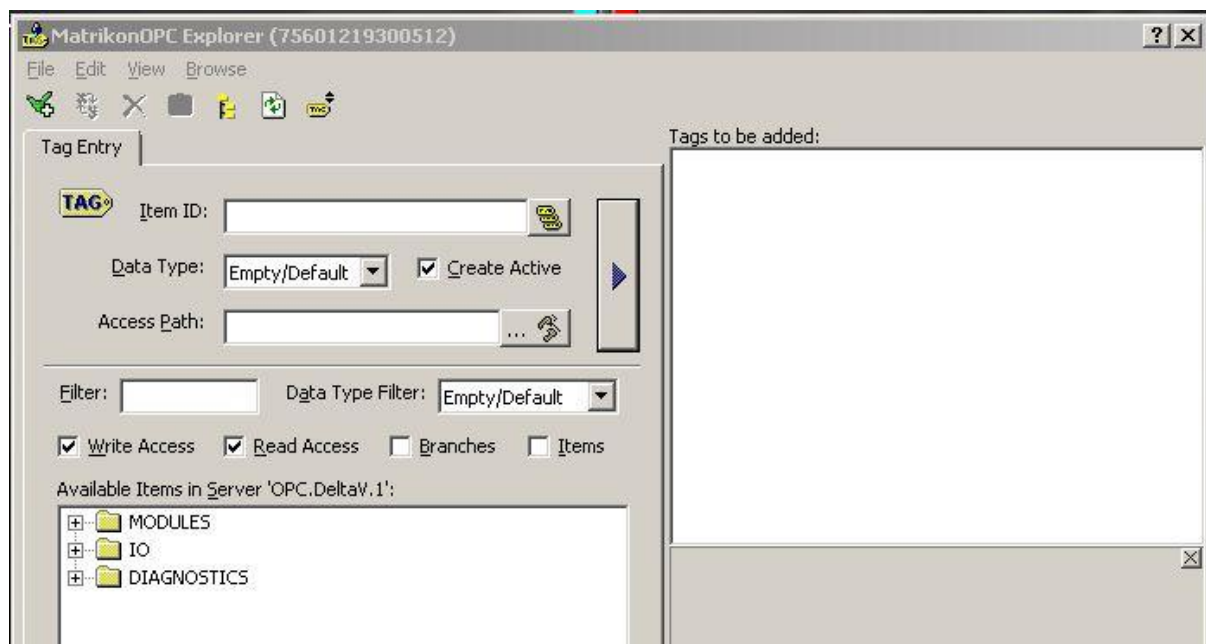


Figure 4-24: Configuration box for adding tags of DeltaV variables into Matrikon OPC server.

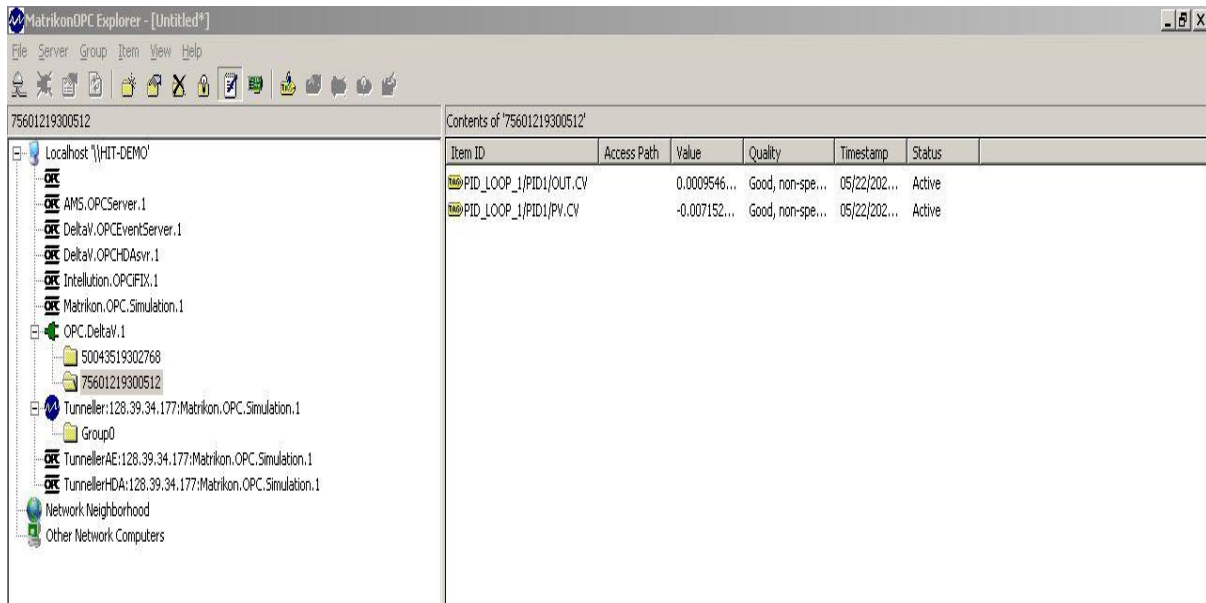


Figure 4-25: Shows the PID controller variables that are tagged into the Matrikon OPC Server from the DeltaV server.

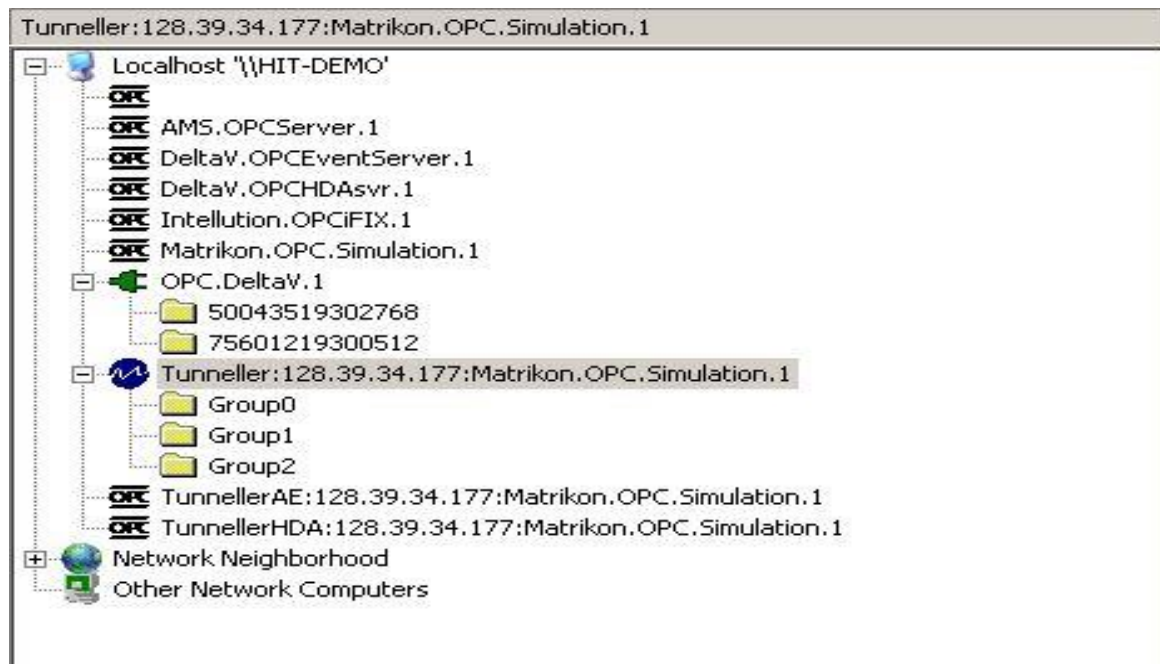


Figure 4-22: This shows the Tunneller option available in the Matrikon OPC server.

After the configurations of the variables into the Matrikon OPC server, using a set of standard codes given in appendix H, the tags can be transferred into the MATLAB software.

4.3 Optimization Techniques

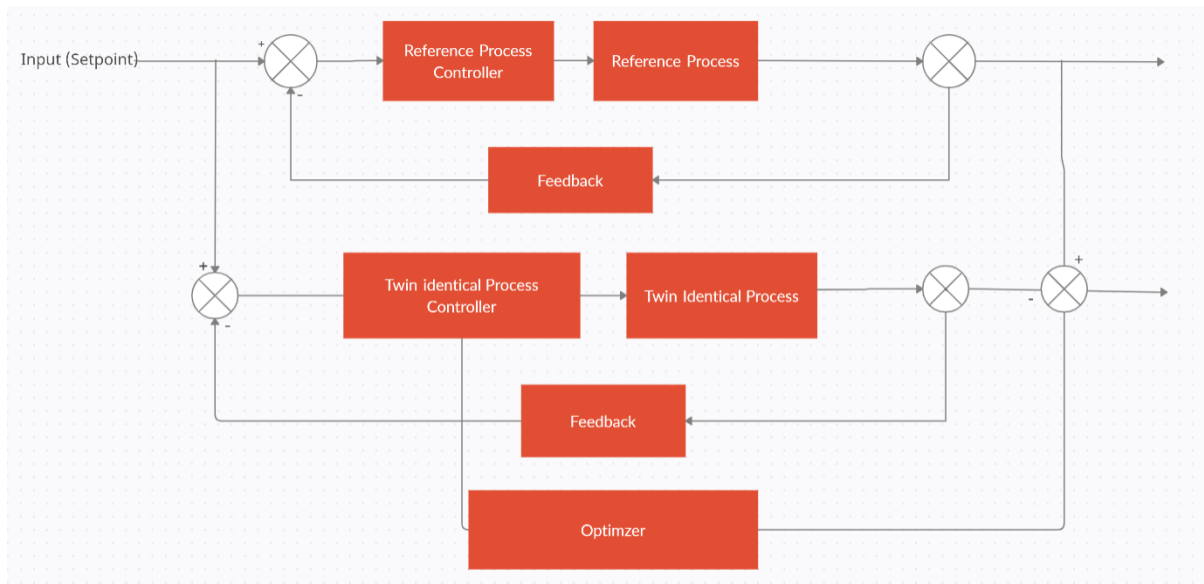


Figure 4-23: Architecture of the real time optimization.

This chapter explains the optimization techniques that have been used to realize the objective of this research. The aim is to match or minimize the outputs of the two models as far as possible. In order to succeed in the objective, there can be two techniques; one is to take the output of the primary process and to set it as the reference for the secondary process' secondary controller. This technique is called cascade control. It is widely used in the industry. However, it has its own limitation. The second technique is auto-tuning the controller parameters such that inputs are optimized to minimize the processes outputs. This technique is known as adaptive control. Model reference adaptive controllers are efficient than cascade controller for removing dynamic uncertainties and modelling errors.[19]. In MRAC, the process model is used; in this experiment, a real plant is used as the reference model.

4.3.1 Optimization of Input Signal

If an assumption is made, the models are entirely identical, and no external disturbances act on the models. Then we can say that the only parameter that will influence the output of the process in an ideal environment is the input signal.

Primary process model

$$\begin{aligned} \dot{x}_p &= A_p x_p + B_p u_p \\ Y_p &= C_p x \end{aligned} \quad (4-3)$$

Secondary process model

$$\begin{aligned} \dot{x}_s &= A_s x_s + B_s u_s \\ Y_s &= C_s x \end{aligned} \quad (4-4)$$

For the assumption made, it can be stated that $A_p = A_s, B_p = B_s, C_p = C_s$.

and if $U_p = U_s$ then the states of the process models become equal $\dot{x}_1 = \dot{x}_2$.

Therefore, it is assumed that the process will behave identically by giving the same input to the model. In order to check this theory, the parameters of the secondary controller are optimized in every time step to calculate the output same as the primary controller output. The optimization function is as follows.

$$\begin{aligned} \min E_i &= \frac{(U_p^i - U_s^i)^2}{2} \\ \min E_i &= \frac{\left(U_p^i - \left(K_{p,s}^{i-1} e_i + K_{i,s}^{i-1} \int_0^i e \right) \right)^2}{2} \end{aligned} \quad (4-5)$$

With respect to

$$K_{p,s}, K_{i,s} > 0$$

Furthermore, the calculated $K_{p,s}$ and $K_{i,s}$ will be fed back again into the controller to calculate the $K_{p,s}$ and $K_{i,s}$ in the next time step. To experiment with the theory in equation 4-5, a MATLAB function block was added into the Simulink. A function was defined into the block that resembles the equation 4-5, and necessary inputs were configured as shown in figure 4-28. For the optimization, the MATLAB non-linear algorithm “fmincon” was used by defining the necessary parameters needed to execute the non-linear algorithm. The primary controller parameters values were given as the initial values of the optimization problem. The code for the MATLAB function is given in Appendix B.

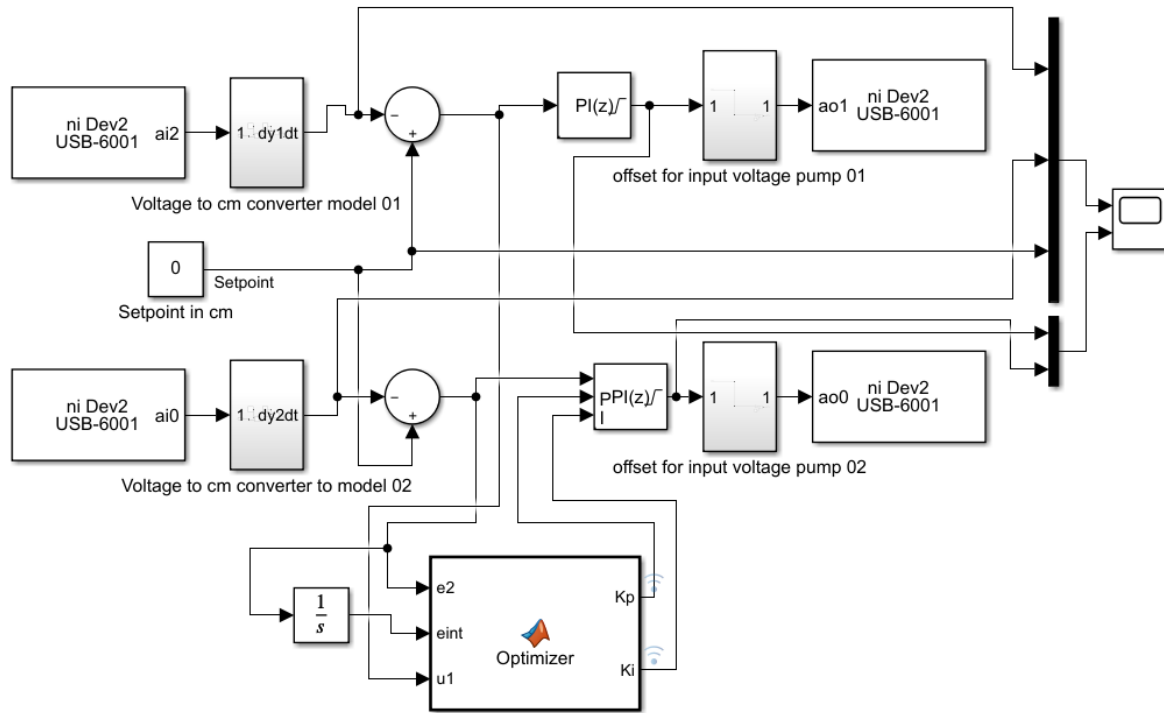


Figure 4-24: MATLAB Simulink setup for the optimization of the controller parameter using the controller outputs error.

4.3.2 Optimization of Controller parameters using Model

The second method evaluated is to find the parameters that reduce the error between the outputs of the processes. A new objective function was designed to run the optimization algorithm. From the equation (3-1), the states of tank 01 for both models can be derived.

The mathematical model of tank 01 in the primary process;

$$\frac{dy_{1,p}}{dt} = \frac{1}{A_{1,p}} \left(C_{p,p} u_p - C_{1,p} \alpha_{1,p} \sqrt{2gy_{1,p}} \right) \quad (4-6)$$

The mathematical model of tank 01 of the secondary process.

$$\frac{dy_{1,s}}{dt} = \frac{1}{A_{1,s}} \left(C_{p,s} u_s - C_{1,s} \alpha_{1,s} \sqrt{2gy_{1,s}} \right) \quad (4-7)$$

In both equation (4-6) and (4-7), the symbols represent the same parameters that are mentioned in equation (3-3) and (3-7), except the subscript symbol p, s represents the primary and secondary processes, respectively.

The optimization function was developed to test this theory.

$$\min E_i = \frac{\left(\frac{dy_{1,p}^i}{dt} - \frac{dy_{1,s}^i}{dt}\right)^2}{2} \tag{4-5}$$

$$\min E_i = \frac{\left(\frac{dy_{1,p}^i}{dt} - \left(C_p \left(K_{p,s}^{i-1} e_i + K_{i,s}^{i-1} \int_0^i e\right) - C_{1,s} \alpha_{1,s} \sqrt{2gy_{1,s}}\right)\right)^2}{2}$$

With respect to

$$K_{p,s}, K_{i,s} > 0$$

Furthermore, the calculated $K_{p,s}$ and $K_{i,s}$ will be fed back to the controller to calculate the $K_{p,s}$ and $K_{i,s}$ in the next iteration. Figure 4-29 shows the MATLAB Simulink setup for the optimization theory mentioned above. A MATLAB function block was created to code the mathematical equation (4-5). The setup is made to optimize the controller parameters for every iteration until the execution is aborted by the user.

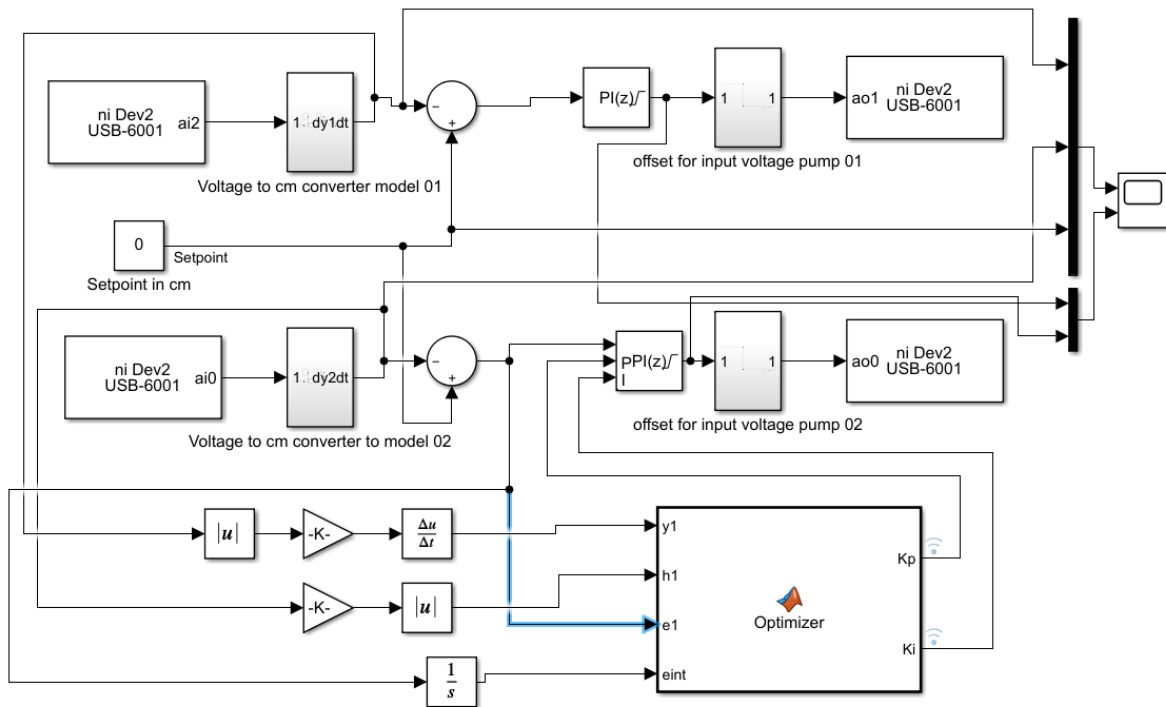


Figure 4-25: MATLAB Simulink setup for controller parameter optimization using the sum of output error as the performance criteria and the model.

4.3.3 The MIT Gradient Descent Rule

The optimization for a controller or process is frequently done using the mathematical model of a process in the industry. Because when a good mathematical model is available, it is easy to do calculations using the model to predict the behaviour of the process and using the data, a good optimization value can be obtained numerically. This gradient descendant rule method does not use the mathematical model of the process or the controller to calculate the variables needed to optimize the error between the outputs of the processes. Instead, it follows an iterative process of increasing or decreasing the parameter values according to a set of rules defined until the error is optimized.

The performance criterion that needs to be achieved is as follows.

$$\sum_i^{i+t} \frac{((y_p - y_s)^2)}{2} \quad (4-6)$$

Where

- i – the initial time of the simulation
- t – Step time of the simulation
- y_p – the primary process output
- y_s – Secondary process output

The gradient descent rules to calculate the $K_{p,s}$ Parameter is as follows.

$$K_{p,s_{n+1}} = K_{p,s_n} - \frac{\alpha(e_{s_n}^2 - e_{s_{n-1}}^2)}{(K_{p,s_n} - K_{p,s_{n-1}})} \quad (4-7)$$

The $K_{i,s}$ parameter will be calculated with the equation (2-13) using the value obtained for the $K_{p,s}$ from the equation of the primary controller. Or using the same gradient descent rule

The gradient descent rule for calculating the K_i parameter is as follows.

$$K_{i,s_{n+1}} = K_{i,s_n} - \frac{\alpha(e_{s_n}^2 - e_{s_{n-1}}^2)}{(K_{i,s_n} - K_{i,s_{n-1}})} \quad (4-8)$$

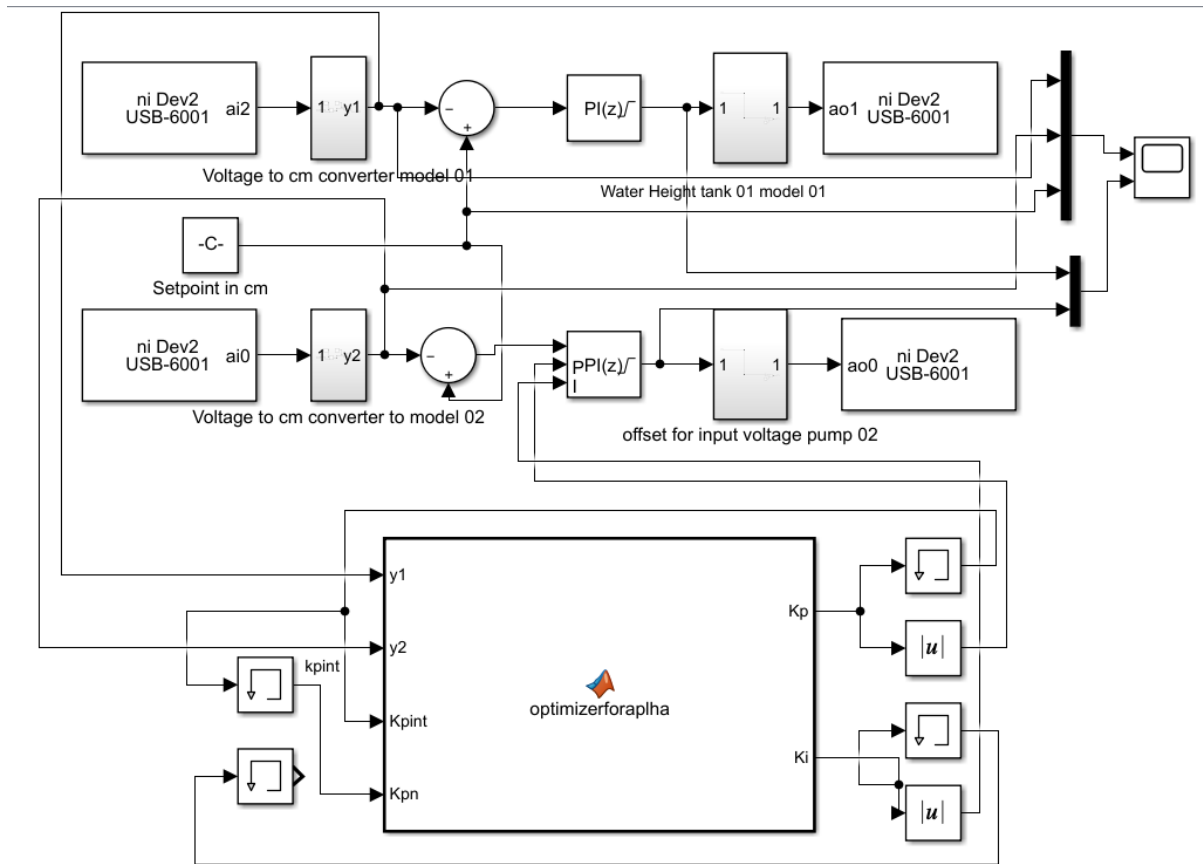


Figure 4-26: MATLAB Simulink setup for optimizing the controller parameter using the sum of output error as the performance criteria and using the gradient descent rule to calculate the new optimized value.

Figure 4-31 shows the setup of the optimizer with conditional switches at the output to limit the output once the switch condition becomes active. This setup was done to limit the frequent update of the controller parameters once the condition is fulfilled. Furthermore, the values of the previous optimization will be given as the input to the controller parameters once the condition is fulfilled.

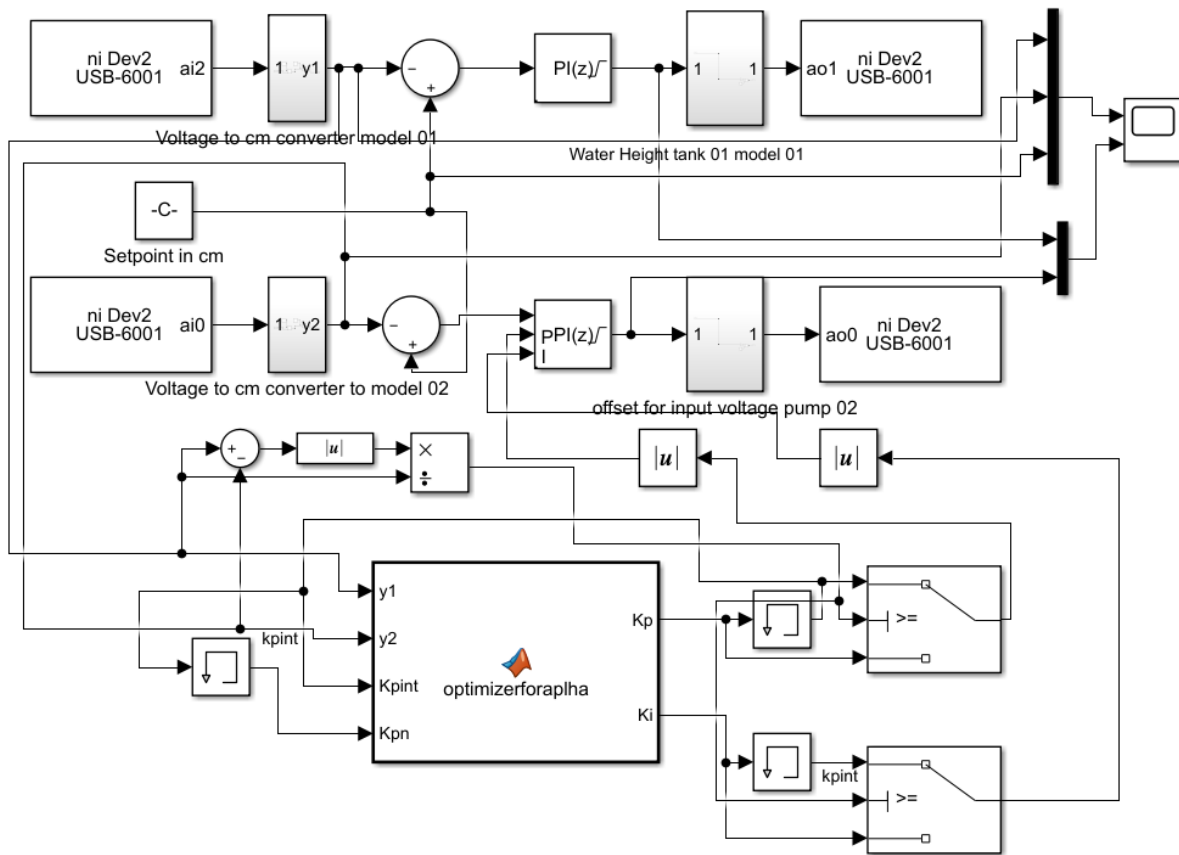


Figure 4-27: MATLAB Simulink setup for optimizing the controller parameter using the sum of output error as the performance criteria and using the gradient descent rule to calculate the new optimized value.

5 Results

5.1 Tuning of the primary process' controller

The primary model was tuned with the Skogestad method for the PI controller. To tune the controller according to the Skogestad method, an open-loop simulation was performed.

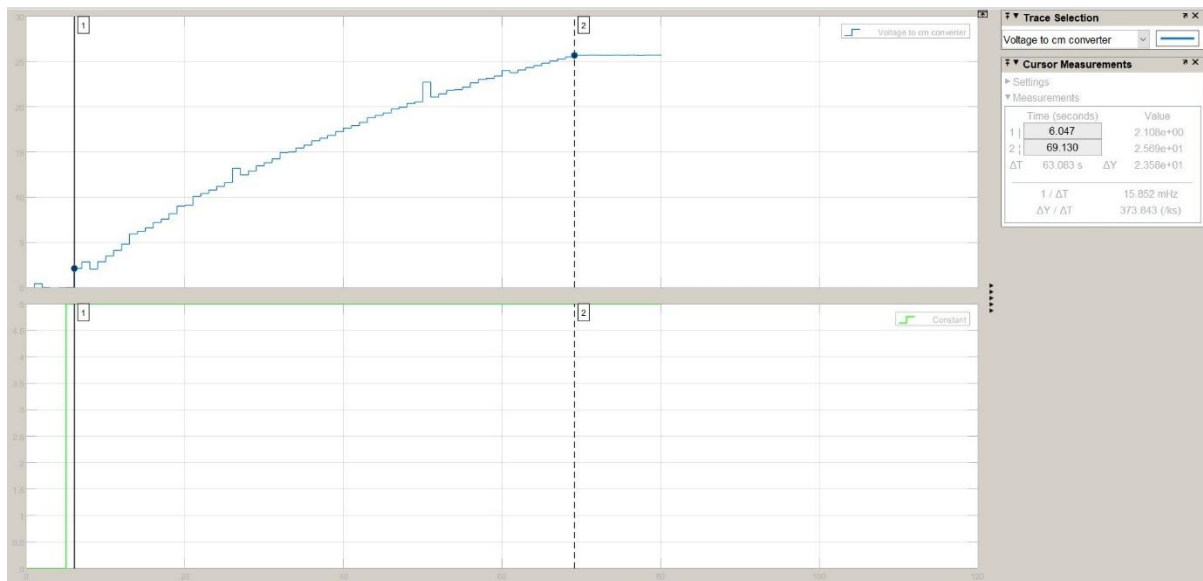


Figure 5-1: Open loop simulation of a step change for the primary system

the calculations were performed to find the $K_{p,p}$ and $K_{i,p}$ parameter of the controller using the open-loop simulation result,

The slope of the open-loop simulation was obtained using the equation 2-11.

$$\begin{aligned}
 S_p &= \frac{\Delta y}{\Delta x} \\
 &= \frac{25.6 - 0}{63} \\
 &= 0.40634901
 \end{aligned} \tag{5-1}$$

The open-loop process gain calculated using the equation 2-11.

$$\begin{aligned}
 K_{c,p} &= \frac{S}{U} \\
 &= \frac{0.40634901}{5} \\
 &= 0.0812698
 \end{aligned} \tag{5-2}$$

According to the open-loop process simulation, the time delay and the time constant was calculated, and the values are as follows.

$$\begin{aligned} T_{c,p} &= 15.75s \\ T_{au,p} &= 1s \end{aligned} \quad (5-3)$$

Where

- $T_{c,p}$ is the time-constant of the primary process.
- $T_{au,p}$ is the Time-delay of the primary process.

Using the equation (2-12) and (2-13), the controller parameters are calculated.

Proportional gain is calculated as:

$$K_{p,p} = \frac{1}{K_{c,p}(T_{c,p} + T_{au,p})}$$

$$K_{p,p} = \frac{1}{0.0812698 * (15.75 + 1)} \quad (5-4)$$

$$K_{p,p} = 0.73460821$$

Integral time is calculated as;

$$K_{i,p} = 2(T_{c,p} + T_{au,p})$$

$$K_{i,p} = 2(15.75 + 1) \quad (5-5)$$

$$K_{i,p} = 33.5$$

Integral gain is calculated as;

$$K_{i,p} = \frac{K_{p,p}}{T_{i,p}}$$

$$K_{i,p} = \frac{0.13460821}{33.5} \quad (5-6)$$

$$K_{i,p} = 0.0219286$$

The calculated parameters were fed into the PI block and were simulated. The result is shown in figure 5-2.

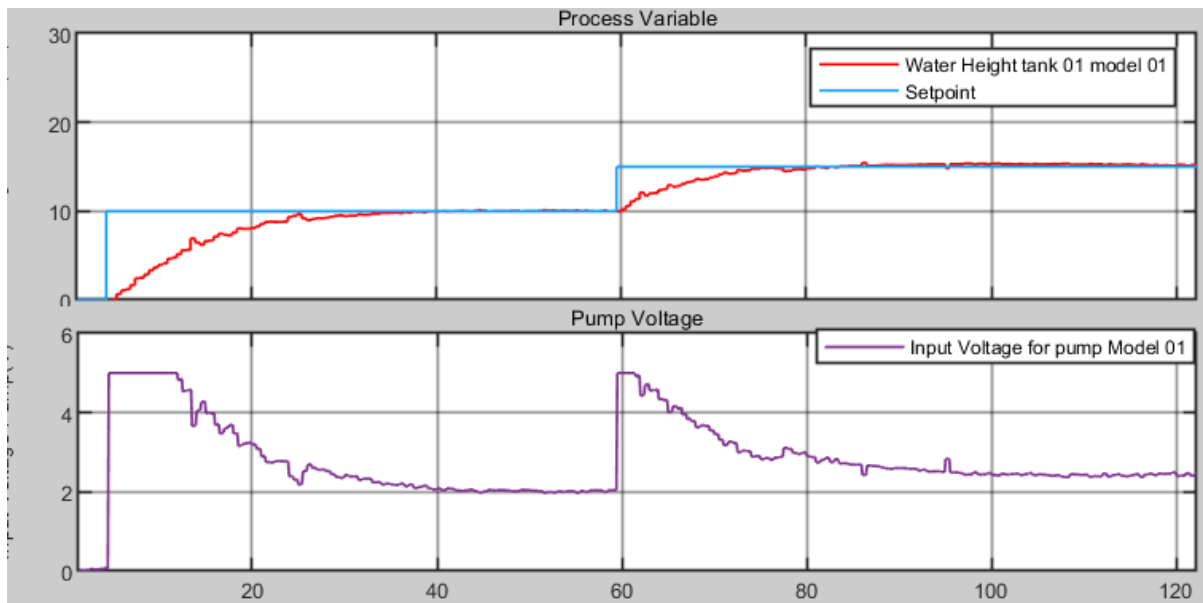


Figure 5-2: Close loop simulation of the primary system with tuned control parameters.

The result shows that proper tuning parameters were gained, and the process behaves very well for the set-point tracking. This result can be taken as a very good as for a process of the tank. There is no considerable overshoot in the output of the system.

5.1.1 Simulating the Process with the same parameters for the DeltaV system

For the DeltaV system, the same calculated tuning parameters were fed into the control module, and the results were gained are as follows.

Figure 5-3 shows the output of the DeltaV controller process for the same PI controller parameters. The DeltaV system output was not the same as from the MATLAB configured process. The figure shows that even though the process reaches the desired set-point, there is a considerable amount of overshoot and undershoot in the DeltaV controlled process before the process reaches the steady-state.

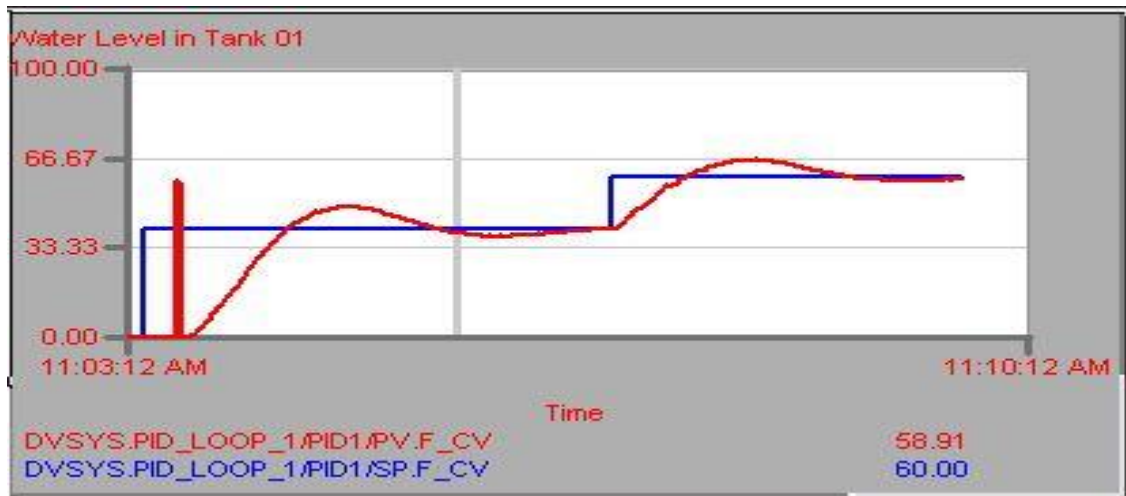


Figure 5-3: Simulation of Secondary System with the Emersons' DeltaV DCS system using the tuned parameter.

5.1.2 Simulation of both identical process in Simulink with Same control Parameters

As for the technical limitation of connecting the DeltaV node to an internet connection during the pandemic, it is decided to analyse the thesis objectives further, using MATLAB Simulink. Therefore, both the system were configured with the controller setup available in the MATLAB Simulink library. The systems were simulated initially using the same controller parameters gained during the tuning of the primary system.

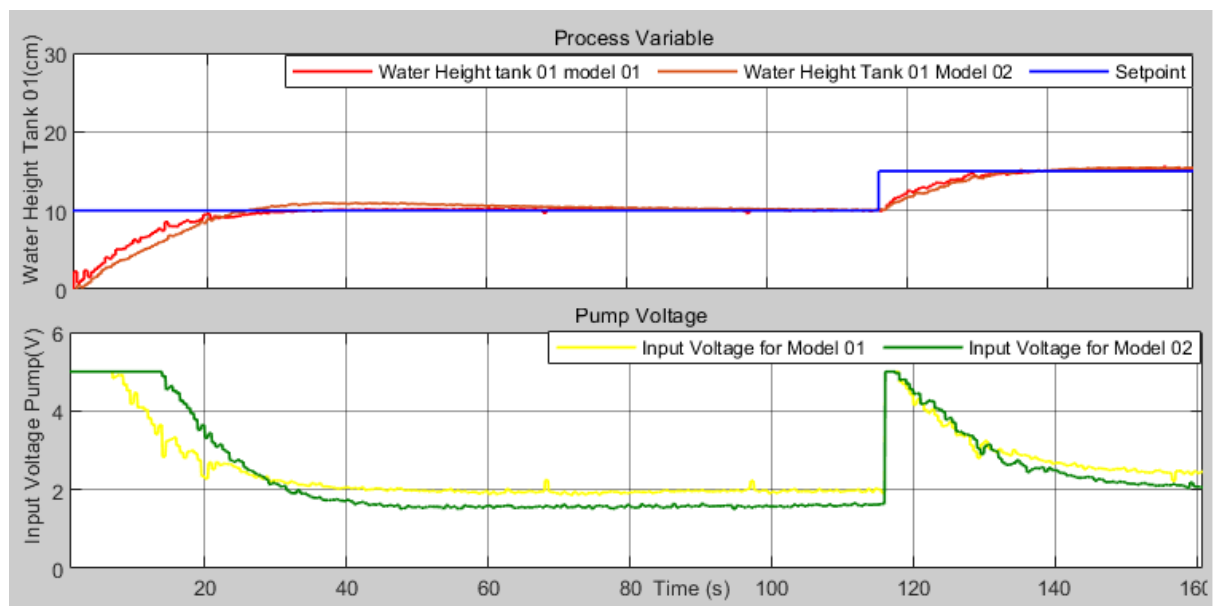


Figure 5-4: Simulation of both primary and secondary system with the tuned controller parameters in MATLAB Simulink.

5.2 Optimization of the Model Parameters

The mathematical model was simulated against the real process using MATLAB Simulink. The data were logged into a MATLAB file. The data gathered were used to build a non-linear optimization problem to fine-tune the model's unknown parameters. The saved data were used to design a non-linear optimization problem. The parameters, pump coefficient, and valve opening coefficients were optimized using the “fmincon” MATLAB function because these parameters are not measured physically.

The optimization problems are as follows.

$$\sum_i^{i+nt} \frac{(y_p - y_m)^2}{2} \quad (5-7)$$

with subject to

$$C_p, C_1, C_2 > 0$$

Where

- y_p - is the plant output.
- y_m - is the output from the model (The equation for y_m is given by the equation (3-10) and (3-11).
- i - the initial time of the simulation.
- $i+n$ - the final time of the simulation.
- n - no of time steps in the total simulation.
- t - Step time of the simulation.

Table 5-1: Values of tune parameters

Primary Model	Parameter value used for simulation	Tuned parameter value
$C_{p,p}$	1.9406e-05	1.9906e-05
$C_{1,p}$	0.7	0.8031
$C_{2,p}$	0.7	0.7740
Secondary Model		
$C_{p,s}$	1.9406e-05	1.4062e-05
$C_{1,s}$	0.7	0.6673
$C_{2,s}$	0.7	0.6591

Figures 5-5 and 5-6 show the optimisation results using MATLAB script and the non-linear optimization algorithm “fmincon” for both types of untuned and tuned parameters compared in the same figures. After obtaining the tuned parameters, the parameters were defined into the model, and the models were simulated in real-time along with the actual process. Figure 5-7 and 5-8 shows the real-time simulations of the tuned primary and secondary model along with their real process, respectively.

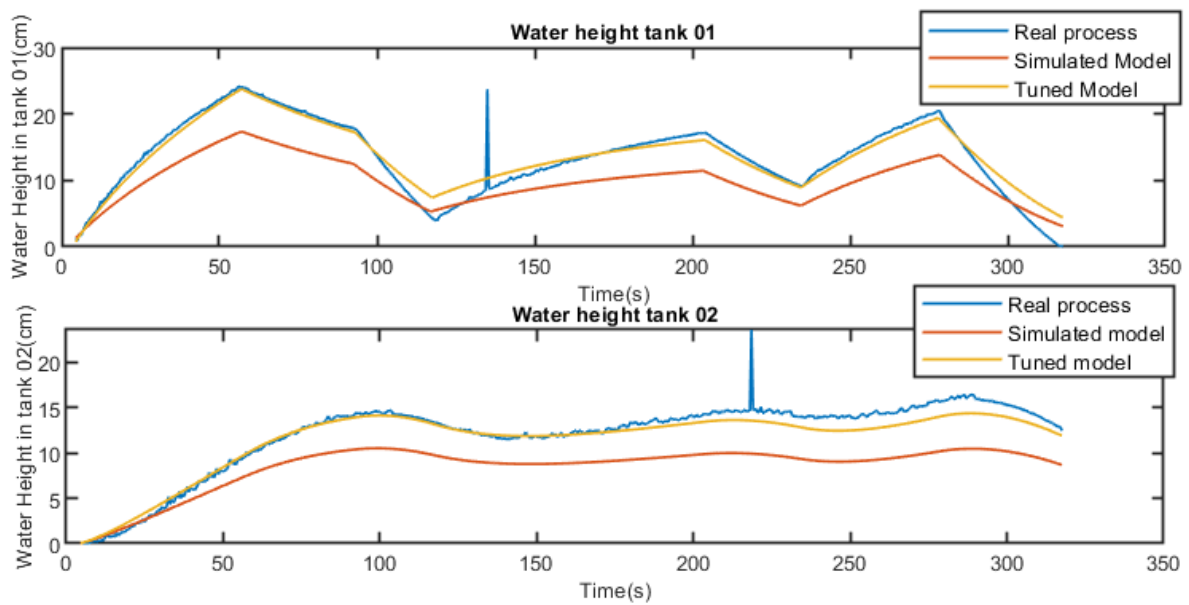


Figure 5-5: Simulation of the primary process before and after the model tuning against the real process using MATLAB script.

The simulation of the primary model shows that the tuned model behaves nearly equally to the real process, especially in the initial stage of the simulation.

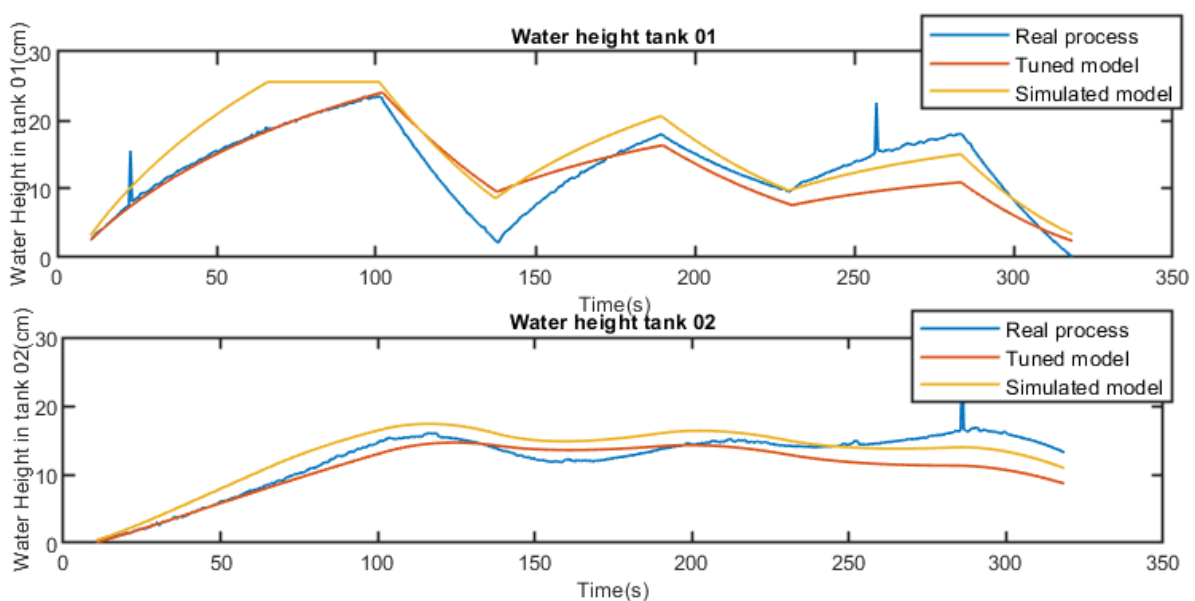


Figure 5-6: Simulation of the Secondary process before and after the model tuning against the real process using MATLAB script.

The simulation of the Secondary model also shows that the tuned model behaves better than the un-tuned model on par with the real process, especially in the initial stage of the simulation.

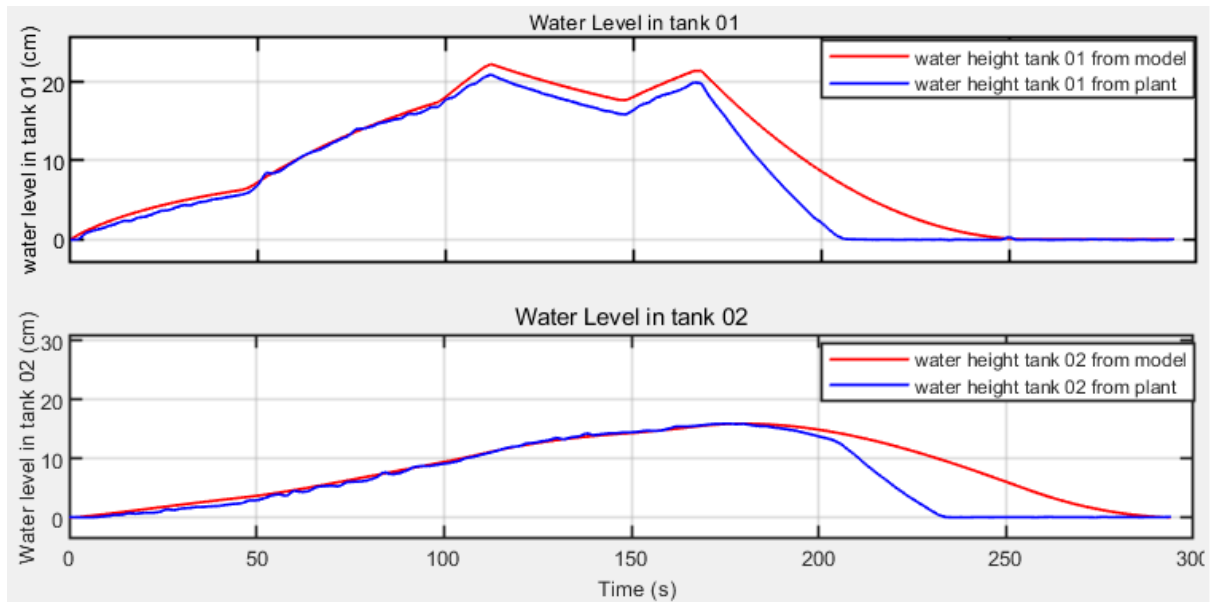


Figure 5-7: Real-time simulation of the tuned primary model along with the primary process in MATLAB Simulink.

Both models behave approximately equal to the actual process, but errors are visible between the model and the real process when the input voltage to the pump is turned off, suggesting that the valve coefficient parameter is inaccurate.

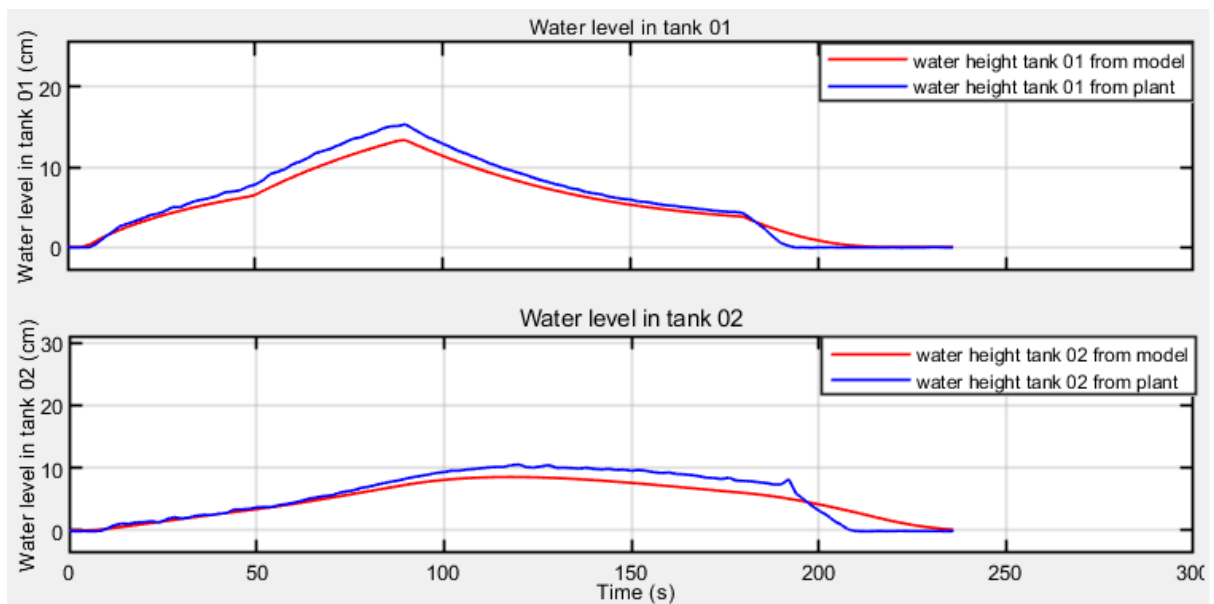


Figure 5-8: Real-time simulation of the tuned Secondary model along with the Secondary process in MATLAB Simulink.

5.3 Optimization Techniques

5.3.1 Optimization of the control Inputs

A setup was created to analyse the model's outcome when the input to the model is optimized to minimize the error between them. An assumption is made that there are no external disturbances in the system, and the systems are identical to each other. The result is as follows.

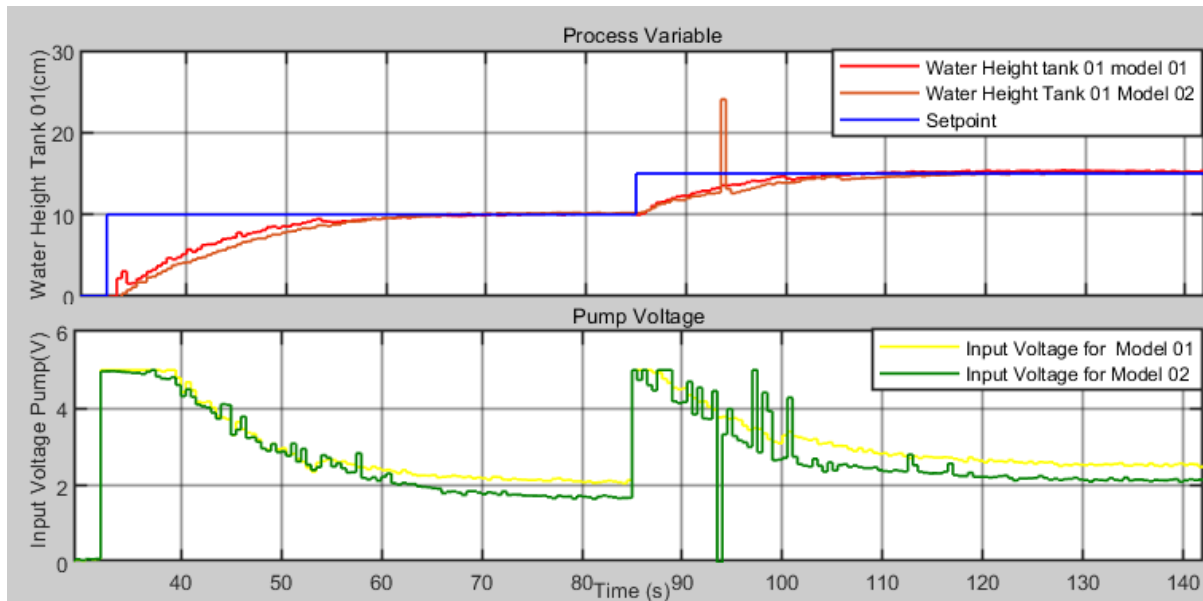


Figure 5-9: Result of Simulation of the processes in which the secondary process is simulated with the optimized parameters to match the input signal.

Figure:5-9 shows the output of the secondary process along with the primary process output. The parameters $K_{p,s}$ and $K_{i,s}$ of the secondary controller is optimized to match the primary controller output in every timestep iteration of the simulation. Compared with the figure:5-4 the result is better, which can be seen as the overshoot in the figure:5-4 is wholly removed. Moreover, the error between the two processes also has been minimized.

5.3.2 Optimization of the secondary control parameters using the Model

Using the theory stated in chapter 4.3.2, an optimization algorithm was developed in the MATLAB Simulink using the function block. The setup was shown in figure 4-29. For every time step iteration, the processes outputs were compared and using the algorithm, the optimized values for parameters of the secondary system needed to minimize the error were calculated. Then the calculated parameters were fed into the secondary controller to calculate the following control input.

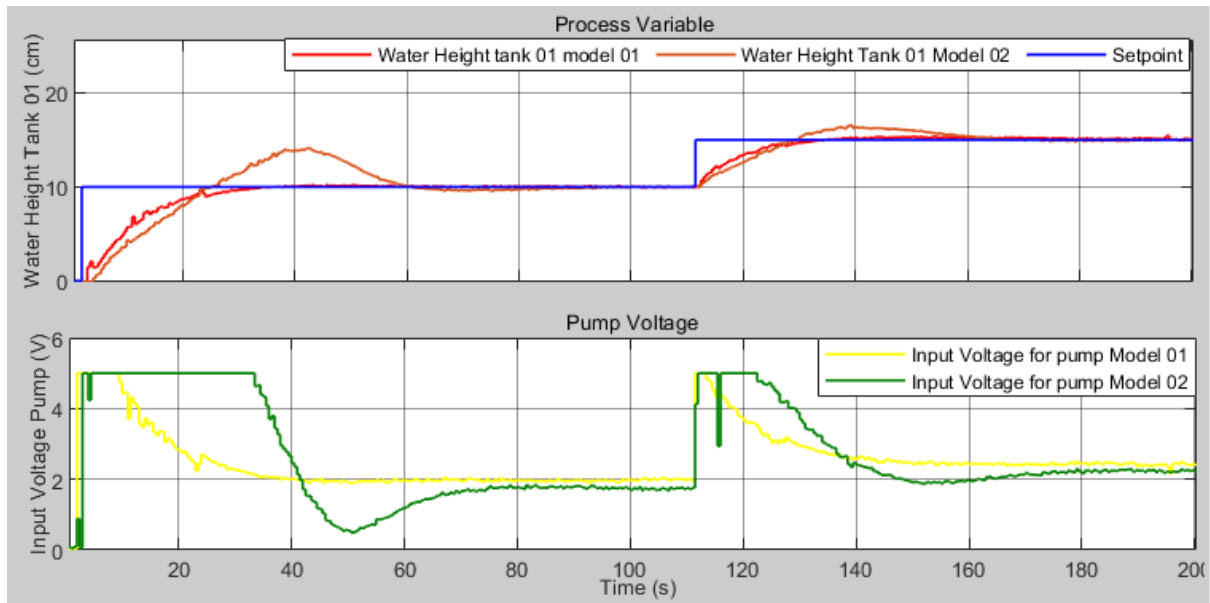


Figure 5-10: Comparison of process outputs for tuning the secondary controller parameters using the process model.

The result shows that even though the secondary process reaches the set-point, it has a high overshoot compared to using the same controller parameters gained for the primary controller. The reason can be that the model used does not precisely behave the same as the actual process. So, the parameters of the models need to be fine-tuned furthermore in order to re-evaluate this technique.

5.3.3 Optimization of the controller parameters using the gradient descent rule

This technique uses the method approximating a gradient descendant rule to minimize the squared sum of error of performance criteria. This adjustment technique was first proposed by MIT scholars in 1961, also known as the MIT gradient descendant rule for model-based adaptive control. This rule does not give a global convergent value that minimises the performance criteria, but it has a satisfying performance over achieving the performance criteria [20]. This technique is only used to tune the proportional gain of the controller and uses the mathematical model of the process to determine the output. In this experiment, rather than using the mathematical model, the actual process is used as the reference model. Moreover, this gradient descent technique is used to calculate the controller parameter $K_{p,s}$ and $K_{i,s}$ tuning outcome using the same rule.

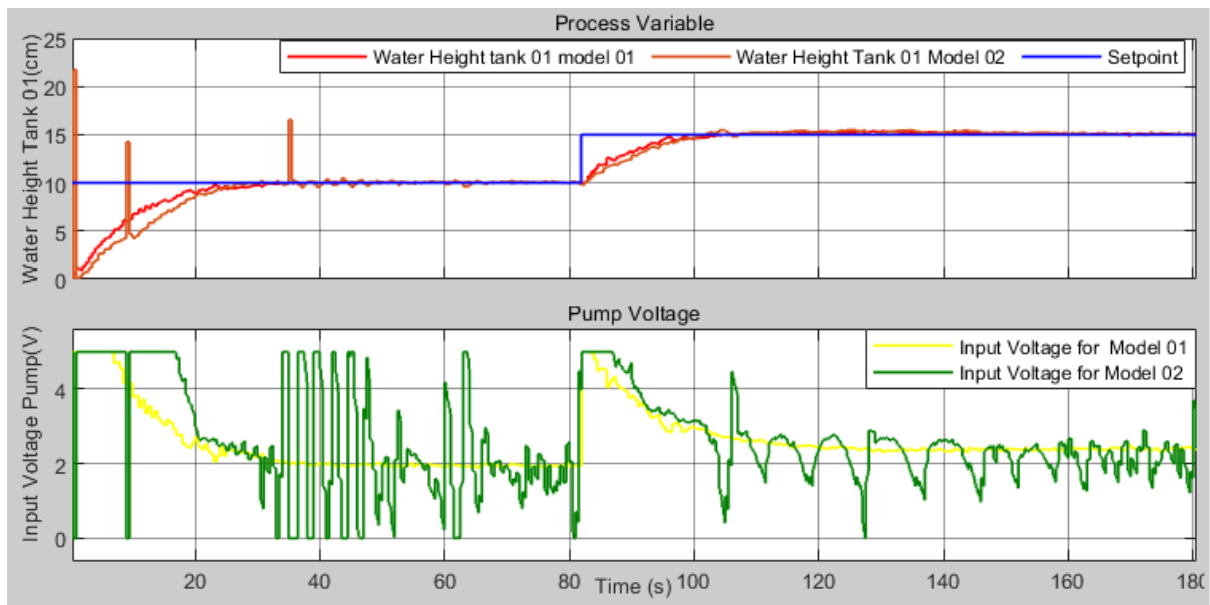


Figure 5-11: Optimization of secondary control parameter $K_{p,s}$ and using the same $K_{i,p}$ value from the primary controller.

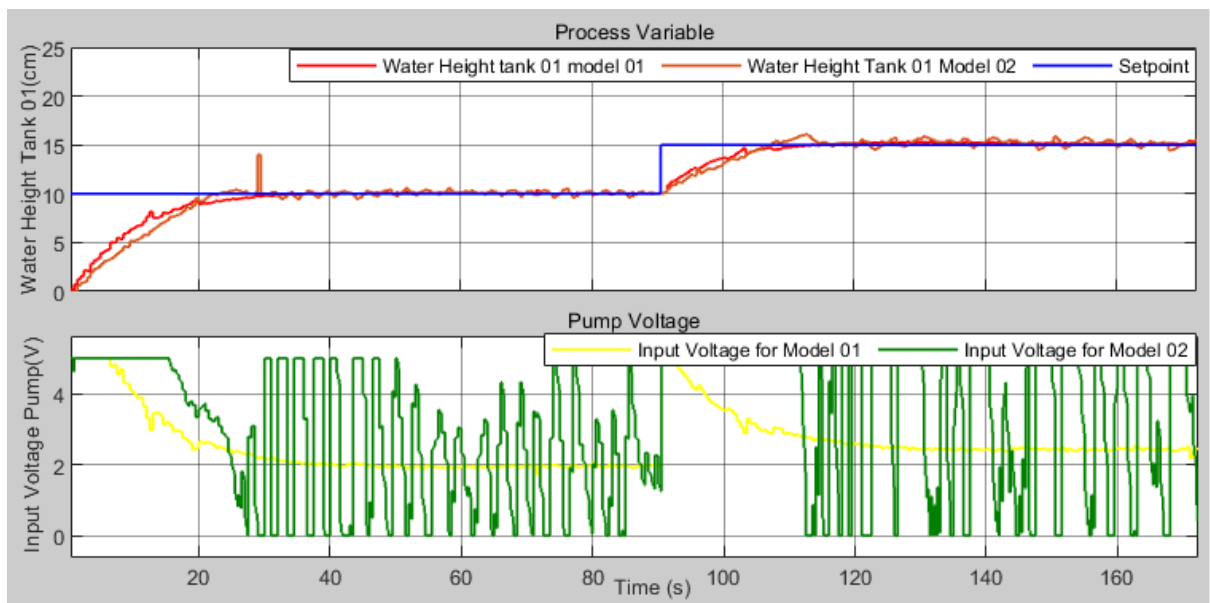


Figure 5-12: Optimization of secondary control parameter $K_{p,s}$ and corresponding $K_{i,s}$ value calculated using equation(2-13) from the primary controller.

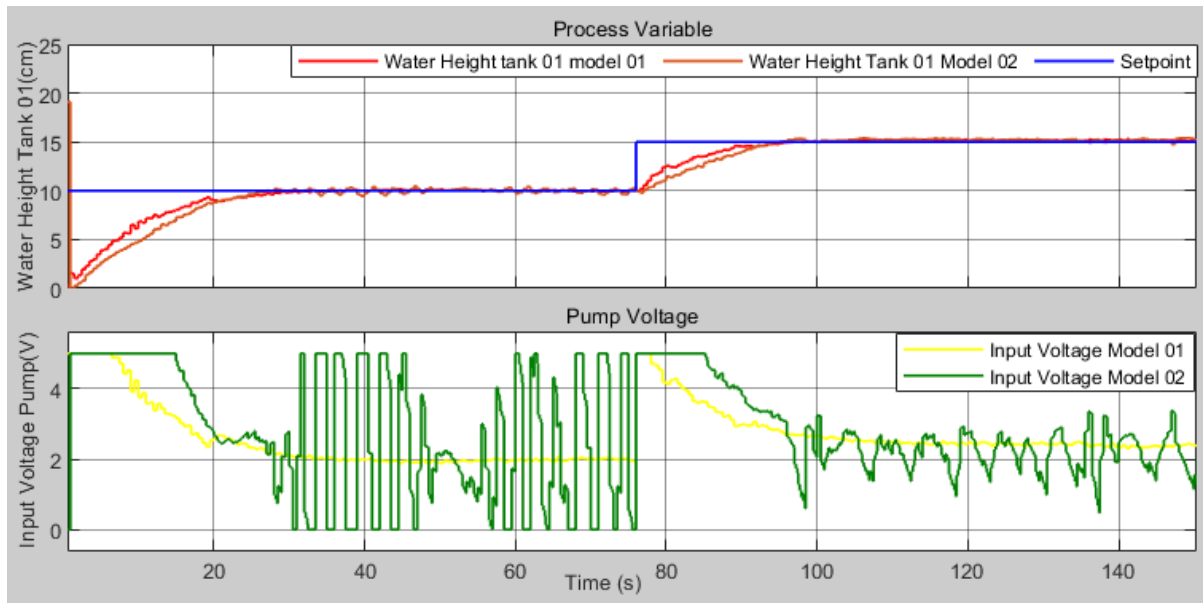


Figure 5-13: Optimization of secondary control parameter $K_{p,s}$ and $K_{i,s}$ value using gradient descendant rule.

Figure:5-11 shows the output of the secondary process along with primary; In this experiment, only the $K_{p,s}$ value is optimized to meet the performance criteria mentioned in the equation. All three techniques have reduced the error between the output of the processes but using the corresponding $K_{i,s}$ value calculated from the $K_{p,s}$ value of the gradient descent rule, and there is a significant amount of oscillation in the secondary process output.

5.3.4 Optimization of the controller parameters using gradient descent rule and limit switch

To eliminate the output oscillations, the gradient descent technique was tested with a switch. The switch limits the output from the optimizer to the PI controller. A condition was configured for the switch to pass the previous optimizer output when the condition becomes valid. Memory cells available in the MATLAB Simulink library was used to save the previous calculated $K_{p,s}$ and $K_{i,s}$ values. Moreover, when the configured switch condition becomes active, the memory cell's values are fed into the PI controller as the parameter values. The setup for this experiment is shown in figure 4-31.

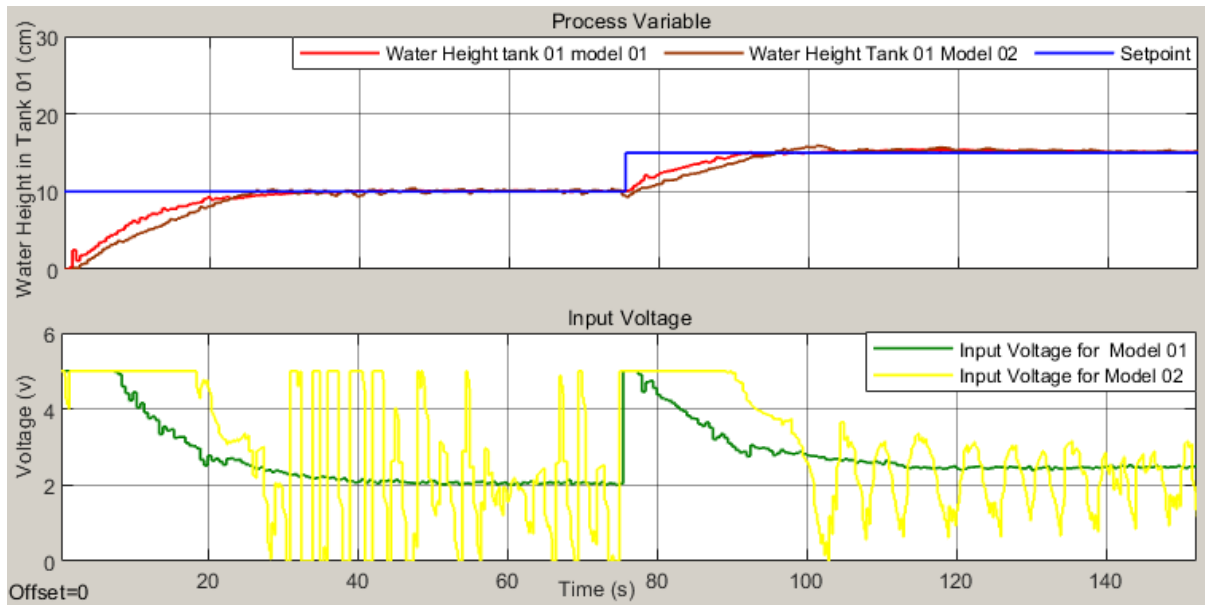


Figure 5-15: Optimization of secondary control parameter $K_{p,s}$ and using the same $K_{i,p}$ value from the primary controller with switch.

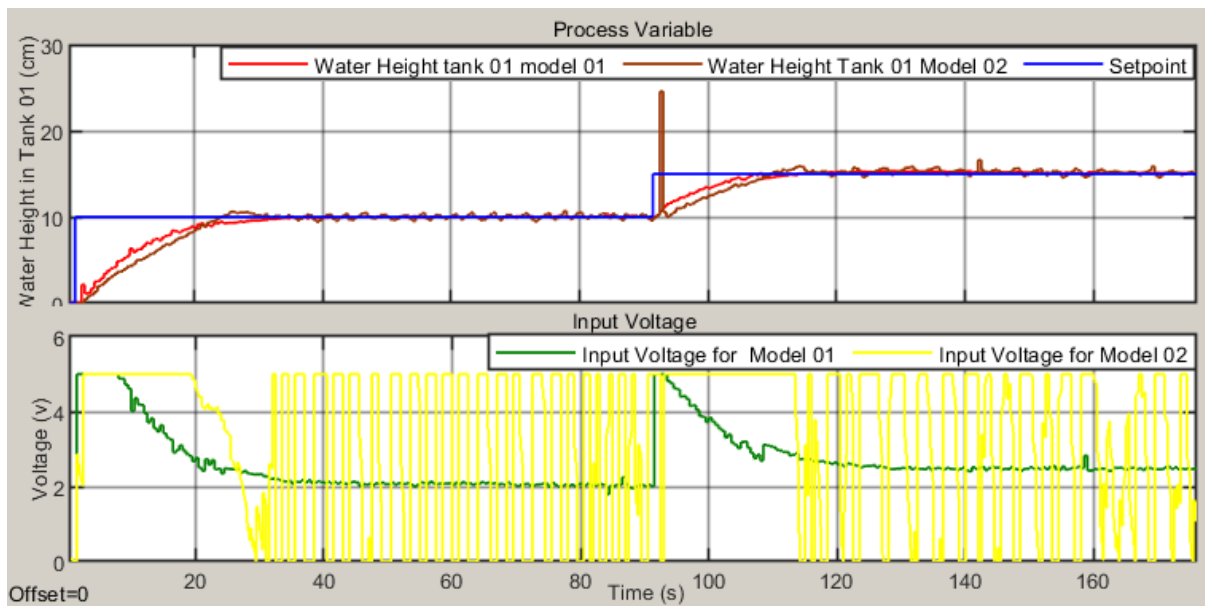


Figure 5-14: Optimization of secondary control parameter $K_{p,s}$ and corresponding $K_{i,s}$ value calculated using equation (2-13) from the primary controller.

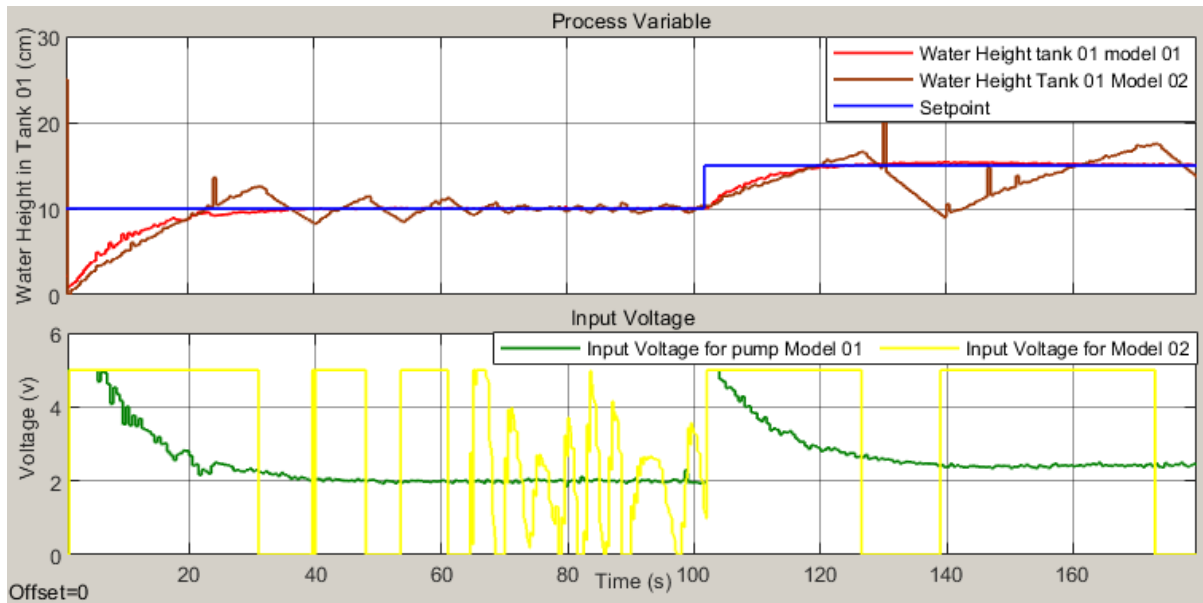


Figure 5-16: Optimization of secondary control parameter $K_{p,s}$ and $K_{i,s}$ value using gradient descent rule with switch.

Figures 5-11 and 5-14 show the secondary process's output without a conditional switch and with a conditional switch, respectively, for the same $K_{i,p}$ value. It shows that the output has not much changed for the same $K_{i,p}$ value technique when a switch is used. In the same way, not much difference can be observed for the technique of using the corresponding calculated $K_{i,s}$ value, which can be seen from Figures 5-12 and 5-14. When both secondary process controller parameters $K_{p,s}$ and $K_{i,s}$ are optimized using gradient descent rule and fed into the controller without a conditional switch and with a switch, the output result is given in Figures 5-13 and 5-16. Both figures show the process output has an oscillation, but the oscillation is more in the process with a switch.

6 Discussion

The discussion chapter is divided into three subcategories: tuning of the primary process, optimization of the mathematical model of the processes and optimization of the secondary controller parameters. The results of the experiments from chapter 5 are discussed in this topic.

6.1 Tuning of the Primary process

The primary process was tuned using the Skogestad method of PI controller[3]. The good gain method introduced by Finn [2] was also tested, but the parameters gained from the Skogestad gave a good result over the good gain method.

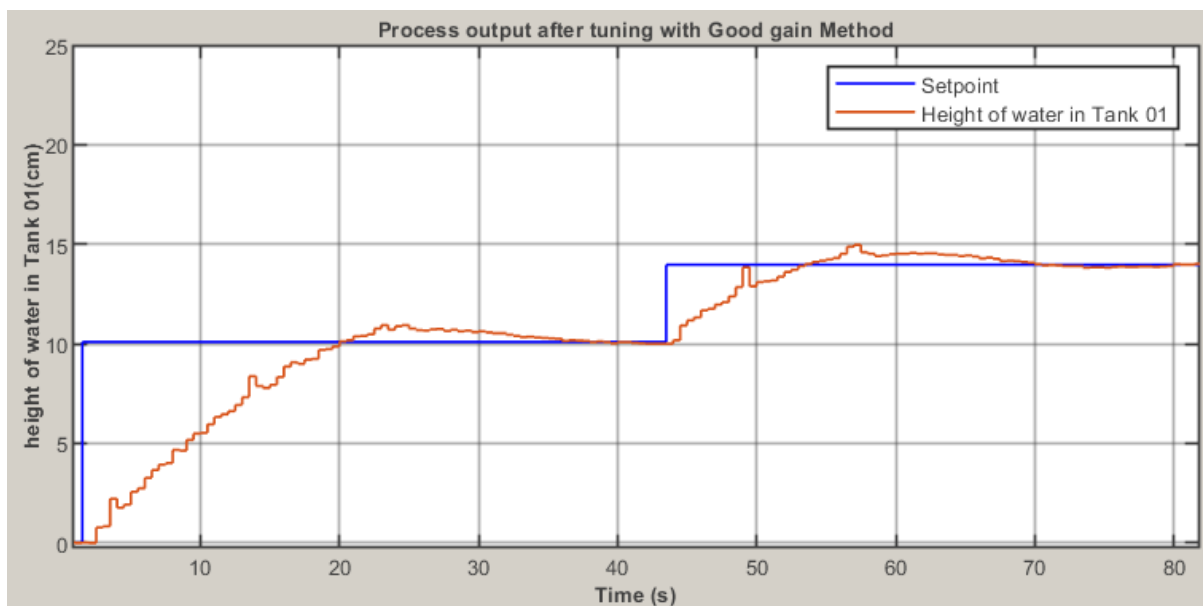


Figure 6-1: Simulation of the primary process with good gain method tuning.

Comparing figure 6-1 with figure 5-2 shows that the Skogestad method's parameters have given an accurate output for the primary process. No overshoot in the process variables can be seen for primary process output. So, it can be said that the Skogestad method gives a better value for the parameters. For the simulation of the tuned process using the DeltaV DCS system of the secondary process, which is shown in figure 5-3, even though the process output reaches the set-point, there is an overshoot and undershoot before the process reaches the steady-state. It does not behave exactly the same as the primary process controlled by the MATLAB Simulink. It may be because using different controllers to control the process or the DCS system does not compensate for the K_p and K_i parameters' decimal values as only a certain number of decimal places can be given as the input in the DCS system.

When both the primary and secondary process is connected with the MATLAB Simulink, simulated for a step change using the same controller parameters calculated and given by the equation (5-4) and (5-6), both the systems behave nearly the same except there is a noticeable amount of overshoot in the secondary system output which is shown in figure 5-4. Furthermore,

at the initial time of the process simulation, a gap between both systems can be seen until both systems reach the set-point range; this may be because the primary system's pump coefficient is larger than that of the secondary system. So the rate of water fill in the primary is higher than in the secondary system. Nevertheless, the process error of the secondary process is comparatively less than the output observed when using the DeltaV DCS system as the controller.

6.2 Optimization of the mathematical model

A mathematical model of the process was built using the 1st principles theory. The methodology, related calculations and formulated equations are shown in chapter 3. Both models have the same model structure, but when the model was simulated against the real processes, the model output vastly deviated from the real process. Some of the model parameters were physically calculated, and others needed extensive experiments to calculate them numerically. However, to do the experiments, the process rig must be modified; this was not done because it may interfere with the integrity of the process physical identification. Therefore, it is decided to optimize the unknown parameters using the non-linear optimization with respect to the performance criteria, output error sum of the process and the mathematical model. This procedure is clearly documented in chapter 5.2. The codes used to do the optimization is given in appendix A.

The outputs from optimization are shown in figure 5-5 and 5-6. Moreover, figures 5-7 and 5-8 show the real-time simulation of the primary and secondary model against their real processes, respectively. Table 5-1 shows the parameter values before and after tuning for both models. From figure 5-7 and 5-8, it can be clearly seen that the model developed behaves nearly to the real processes, but there are errors between the outputs. This may be due to the fact that the non-linear optimizer does not guarantee a global optimal solution[21]. Therefore, the solution given by the optimizer may be a local optimal solution.

6.3 Optimization of the secondary controller parameters

The optimization techniques used have been thoroughly documented in chapter 4.3, including the MATLAB Simulink setup configuration. Chapter 5.3 describes the outcome from the techniques described in chapter 4.3.

6.3.1 Optimization of the input signal

According to the assumptions declared in chapter 4.3.1 for an ideal twin process. The input signals were optimized using the minimization of error between the calculated controller signal as the performance criteria, which was given by the equation (4-5). The result is shown in figure 5-9, which shows the input signal's optimisation also minimises the error between the output of the processes. It is unclear whether this can be taken as a prominent technique because, in the real world, the assumptions made to realize this technique cannot exist. If even systems can be built identically, their dynamics may not be the same nor their disturbances. However, this technique has given a positive result and has minimized the error between the processes outputs.

6.3.2 Optimization of the secondary control parameters using the Model

The theoretical explanation experimental setup behind this technique has been explained in chapter 4.3.2. The result of the theory execution is given in chapter 5.3.1. Figure 5-10 shows the comparison between the primary and secondary processes tank 01 output and their controller input, respectively. The secondary process output has given a significant error compared with the output gained using the same controller parameter. The error may be due to the fact that the model does not exactly represent the real process. Fortunately, the controller output is very smooth compared with the other techniques. The model tuned controller takes a longer time to react to the error than the controller with the same parameter, which can be seen by the pump control signal of the secondary process from figure 5-4 and 5-10. This may state a delay in the system, or the K_p value's difference between the two techniques is comparatively high.

This technique has to be re-evaluated using a re-tuned model where the parameters C_p , C_1 , C_2 are experimentally obtained as the initial input for the non-linear optimizer.

6.3.3 Optimization of the controller parameters using the gradient descent rule

The last technique used to evaluate the thesis theory is the gradient descent rule, also known as the MIT rule. The theory behind this technique and the experimental setup is clearly stated in chapter 4.3.3. The performance criteria for the calculation of the controller parameters is given by the equation 4-6. The rule for calculating the K_p and K_i is given by the equation (4-7) and (4-8), respectively. The results obtained from the simulation are reported in chapter 5.3.3. This technique has been simulated using three different scenarios; using calculated $K_{p,s}$ values and same $K_{i,p}$ primary controller value, using calculated $K_{p,s}$ value and calculated corresponding $K_{i,s}$ values and using calculated $K_{p,s}$ and $K_{i,s}$ value from MIT rule.

Figures 5-11, 5-12 and 5-13 show the optimisation results for above stated different scenarios, respectively. All three scenarios give a positive output for optimizing the output between the models but with some oscillations near the steady-state. However, when using primary controller parameter $K_{i,p}$ value (shown in figure 5-11) and using $K_{i,s}$ value calculated from the gradient descent rule (shown in figure 5-13), the results obtained was comparatively the same. The oscillations are high when the corresponding calculated value of $K_{i,s}$ is used compared with other scenarios (shown in figure 5-12).

The oscillations may have occurred for constant optimization of the controller parameters at every time step; therefore, a switch was introduced to minimize the constant update of the controller parameter. The switch limits the new calculated parameter to be fed into the controller when a condition becomes active, and the condition was set as the error between the processes to be less than 5 per cent. The results are shown in figure 5-14, 5-15, and 5-16 for the scenarios state above, respectively.

The output shows that not much of a difference is observed when using switch was used for the scenario of using the same $K_{i,p}$ value (shown in figure 5-11 and 5-14) and using corresponding calculated $K_{i,s}$ value (shown in figure 5-12 and 5-15), but the oscillation in the steady-state becomes more significant for the scenario when $K_{p,s}$ and $K_{i,s}$ values were calculated with gradient descent rule (shown in figure 5-13 and 5-16)

The limit switch on the optimized parameter does not have an effect in reducing the oscillation around the set-point. For a system to have oscillations, the system's gain must be larger than

the desired gain, or the integral time of the system must be lesser than the desired controller value. Therefore an upper boundary must also be introduced in the optimization algorithm to find an optimized value that does not give an oscillation. However, this may arise the question of how to set the limit for the optimization variables. So this technique has to be further analyzed and needed a literature review. The authors' opinion can be suggested to test the optimizer with the upper bound of the variables with a twenty per cent increase from the primary controller variable's value.

7 Conclusion

The error between the primary and secondary process outputs was reduced when optimization techniques were used to calculate the secondary controller parameters compared to using the $K_{p,p}$ and $K_{i,p}$ controller parameters to both models. So, it is concluded that these methods can reduce the error between twin processes and behave like mirror processes. The Model tuning process has some uncertainties and requires more experimental studies and modelling re-tuning.

“It was hypothesised that in a parallel configuration, by optimization of the controller parameters using the MIT rule of gradient descent and error sum of input control, the twin process could behave like mirror processes.”

8 Recommendations

This chapter lists some of the recommendations needed to further study this hypothesis using the DeltaV or MATLAB controllers.

- The process rigs need modification on the opening of both tanks' bottom valves to stop the flow so that the tank can be filled, and the valve can be let open without any input to the tank to find out the valve discharge coefficients of both tanks.
- The pump coefficient has to be determined experimentally by closing the tank 01 valve and calculating the fill time of a predetermined level for different input voltages.
- The pump dynamics must be tested, and a new pump controller module has to be installed for both processes, which gives the same amount of output for the same amount of input voltage of 0 - 5v.
- The mathematical models of the processes have to be re-tuned using the obtained values of pump and valve coefficients as the initial value. If there is any difference between the model output and the real process outputs
- The optimization technique has been implemented in real-time and for every iteration of the simulation. In the future, variables can be created with fixed-size vectors to save the processes' outputs rather than real-time optimisation. The optimization can be implemented for the vector to obtain a single controller's parameter value using the variables. Then the output of the optimizer can be appended into the controller for simulation. This may reduce the small oscillations around the steady-state.

References

- [1] V. T. Wivestad, "Learning by DIY-Exploring the Pedagogical Potential of the Serial Two-Tank System in a Control Theory Context," NTNU, 2017.
- [2] F. Haugen, "The Good Gain method for simple experimental tuning of PI controllers," 2012.
- [3] S. Skogestad, "Probably the best simple PID tuning rules in the world," in *AICHE Annual Meeting, Reno, Nevada*, 2001, vol. 77.
- [4] N. Instruments, "USER GUIDE NI USB-6001/6002/6003 Low-Cost DAQ USB Device." [Online]. Available: <https://www.ni.com/pdf/manuals/374259a.pdf>
- [5] S.-L. Jämsä-Jounela, "Future trends in process automation," *IFAC Proceedings Volumes*, vol. 40, no. 1, pp. 1-10, 2007.
- [6] S. Haag and R. Anderl, "Digital twin–Proof of concept," *Manufacturing Letters*, vol. 15, pp. 64-66, 2018.
- [7] E. E. Co. "Delta-distributed-control-system." Emerson Electric Co. <https://www.emerson.com/en-us/automation/control-and-safety-systems/distributed-control-systems-dcs/deltav-distributed-control-system#:~:text=The%20DeltaV%E2%84%A2%20Distributed%20Control,complexity%20and%20lowers%20project%20risk.&text=The%20inherent%20i> (accessed 02.03, 2021).
- [8] R. S. Smith and J. Doyle, "The two tank experiment: A benchmark control problem," in *1988 American Control Conference*, 1988: IEEE, pp. 2026-2031.
- [9] J. Mikleš and M. Fikar, "Process modelling, identification, and control," 2007.
- [10] F. Haugen, *PID Control*. Tapir Academic Press, 2004.
- [11] M. Jarang and S. Sondkar, "PID Tuning and Implementation for Level Control Loop Using Delta V DCS," in *2017 International Conference on Computing, Communication, Control and Automation (ICCUBEA)*, 2017: IEEE, pp. 1-4.
- [12] J. G. Ziegler and N. B. Nichols, "Optimum settings for automatic controllers," *trans. ASME*, vol. 64, no. 11, 1942.
- [13] C. C. Hang, K. J. Åström, and W. K. Ho, "Refinements of the Ziegler–Nichols tuning formula," in *IEE Proceedings D (Control Theory and Applications)*, 1991, vol. 138, no. 2: IET, pp. 111-118.
- [14] H. Zipper and C. Diedrich, "Synchronization of industrial plant and digital twin," in *2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2019: IEEE, pp. 1678-1681.
- [15] S. Pankaj, J. S. Kumar, and R. Nema, "Comparative analysis of MIT rule and Lyapunov rule in model reference adaptive control scheme," *Innovative Systems Design and Engineering*, vol. 2, no. 4, pp. 154-162, 2011.
- [16] M. Vukobratović, D. Stokić, and N. Kirćanski, "Adaptive Control Algorithms," in *Non-Adaptive and Adaptive Control of Manipulation Robots*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1985, pp. 212-265.

References

- [17] P. Swarnkar, S. Jain, and R. Nema, "Effect of adaptation gain on system performance for model reference adaptive control scheme using MIT rule," *International Journal of Electrical and Computer Engineering*, vol. 4, no. 10, pp. 1547-1552, 2010.
- [18] A. Neumaier, "Mathematical Modeling," *Retrieved January*, vol. 18, p. 2011, 2003.
- [19] V. M. Hung, I. Stamatescu, C. Dragana, and N. Paraschiv, "Comparison of model reference adaptive control and cascade PID control for ASTank2," in *2017 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, 2017, vol. 2: IEEE, pp. 1137-1143.
- [20] I. M. Mareels, B. D. Anderson, R. R. Bitmead, M. Bodson, and S. S. Sastry, "Revisiting the MIT rule for adaptive control," in *Adaptive Systems in Control and Signal Processing 1986*: Elsevier, 1987, pp. 161-166.
- [21] T. M. Inc. "Can You Certify That a Solution Is Global?"
https://se.mathworks.com/help/gads/can-you-certify-a-solution-is-global.html#responsive_offcanvas (accessed 18.05.2021, 2021).
- [22] E. P. Management, "DeltaV Users Benefit From Matlab Computing ". [Online]. Available:
https://classes.engineering.wustl.edu/che433/tutorials/DeltaV_MatlabIntegration_rev.ppt

Appendices

Appendix A - Master thesis description

Appendix B - MATLAB script for model tuning

Appendix C - MATLAB script for optimization of controller parameters using process input performance criteria.

Appendix D - MATLAB script for optimization of controller parameters using process model and output error as the performance criteria.

Appendix E - MATLAB script for optimization of controller parameters using MIT gradient descent rule with calculated $K_{p,s}$ value and same $K_{i,p}$ value and output error as the performance criteria.

Appendix F - MATLAB script for optimization of controller parameters using MIT gradient descent rule with calculated $K_{p,s}$ value and calculated corresponding $K_{i,s}$ value and output error as the performance criteria.

Appendix G - MATLAB script for optimization of controller parameters using MIT gradient descent rule with calculated $K_{p,s}$ and calculated $K_{i,s}$ value and output error as the performance criteria.

Appendix H - MATLAB script for connecting the OPC server with MATLAB.

Appendix A



Faculty of Technology, Natural Sciences and Maritime Sciences, Campus Porsgrunn.

FMH606 Master's Thesis

Title: Parallel configuration, control and mirror operation of twin processes using Delta V

USN supervisor: Carlos F. Pfeiffer

External partner: Emerson Porsgrunn (Rune Anderson)

Task background:

The concept of Digital Twin, where a computer model mimics the behaviour of a real process, has recently generated much interest in the industry because of its potential for applications in advanced process control, preventive maintenance, and fault detection and diagnosis, among other uses. A variation of this topic is the “mirror” operation of two similar or “twin” process in different locations. The data from both processes are collected and compared, and the control configurations and parameters are adjusted such that the process “mirror” each other in the behaviour of key variables. This concept can be very valuable when operating plants when the same process is replicated in different production lines, and it is desired that all the production lines have similar behaviour.

Task description:

The main goal of this master thesis is to propose and test strategies to compare in real-time the data of two similar “twin” existing processes and propose and test control strategies to adjust the process parameters in such a way that the process behaves similarly with respect to key selected variables. For this thesis, the experimental work will use two prototype systems for the level control of two tanks operating in cascade. One of the systems will be operating in the Emerson installations, and the other in the process hall at USN. The task will include the following activities:

- Complete a literature review of different approaches for comparing and controlling processes to “mirror” each other (operate in a similar form).
- Develop first-principles models for the two systems and compare them with an open-loop simulation (MATLAB and Simulink).
- Tune the systems to get the desired set point (tune the PI Controller of the systems) and evaluate their performance using simulations (MATLAB and Simulink).
- Feed the obtained parameter to both the system and observe the behaviour and output of the systems using simulations (MATLAB and Simulink).

- Test the proposed strategies simulating the two tanks' systems experimental rigs (using MATLAB and Simulink).
- Investigate and document the necessary configurations and connections to test the proposed strategies using the Delta V system.

- Investigate and document the necessary configuration skills to monitor a process remotely using the Delta V system.
- Test the proposed control strategies using the Delta V in the process hall. This task can be omitted if the process hall is not accessible because of covid restrictions, or if necessary, security connectivity updates of the Delta V system cannot be implemented on time to carry the experiments.
- Complete and deliver the master thesis report.
- Prepare and deliver a thesis presentation.

Student category: IIA

Is the task suitable for online students (not present at the campus)? No

Practical arrangements:

This master thesis requires the Delta V system. Emerson will provide training and support for the DeltaV system. The experimental work will be carried partially at the process hall at USN and partially at Emerson's installations, both in Porsgrunn. Part of the configuration work can be carried remotely.

Important: since this thesis involves a great amount of experimental work, the task description will require major adjustments if the process hall is not available because of COVID regulations.

Signatures:

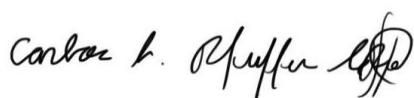
Student (date and signature):

01.04.2021



Supervisor (date and signature):

01.04.202



Appendix B

The MATLAB scripts that are used to optimize the model parameters are presented under this heading.

Model function

Model.m

```
function dh = model(h, u, Cp, C1, C2)

%% Extracting info from the inputs
h1 = h(1);
h2 = h(2);

%% Parameters
A1 = 0.01131;
C1 = C1;
alpha1 = 3.8485e-5;
g = 9.81;

A2 = 0.01131;
C2 = C2;
alpha2 = 3.8485e-5;

%% ODE
dh1 = Cp*u/A1 - C1*alpha1/A1*sqrt(2*g*h1);
dh2 = C1*alpha1/A2*sqrt(2*g*h1) - C2*alpha2/A2*sqrt(2*g*h2);

%% Model output
dh = [dh1 dh2];
```

A function was created using the Runge kuttas' equations from the model to find the next state.

```
function h_next = find_next_state(h0, u, Cp, c1, c2, dt)

% RK4 algorithm

dh1 = model(h0, u, Cp, c1, c2);
dh2 = model(h0+0.5*dt*dh1, u, Cp, c1, c2);
dh3 = model(h0+0.5*dt*dh2, u, Cp, c1, c2);
dh4 = model(h0+dt*dh3, u, Cp, c1, c2);

h_next = h0 + dt/6*(dh1 + 2*dh2 + 2*dh3 + dh4);

if h_next(1) > 0.25
    h_next(1) = 0.25;
```

```

end

if h_next(2) > 0.25
    h_next(2) = 0.25;
end

```

The objective function or the cost function for the optimizer is defined as a separate MATLAB function.

```

function f = objective_func(x)
% -----
% -----
%      % Loading experimental data (plant output)
%      -----
% load('modell1.mat')
%      t          = Modell1(:,1);
%      h_exp(:,1) = Modell1(:,4);
%      h_exp(:,2) = Modell1(:,6);
%      u          = Modell1(:,2);

load('model2.mat')
    t          = Modell1(:,1);
    h_exp(:,1) = Modell1(:,3);
    h_exp(:,2) = Modell1(:,4);
    u          = Modell1(:,2);

    Cp = x(1);
    c1 = x(2);
    c2 = x(3);

% -----
% -----
% Simulated data (model output)
% -----
% -----
    dt = 0.5;

    h = zeros(length(t),2);    h(1,:) = h_exp(1,:)/100;

    for k = 1:length(t)-1
        h(k+1,:) = find_next_state(h(k,:), u(k), Cp, c1,
c2, dt);
    end

```

```

h = h*100;
%-----
% Objective
%-----
f = sum(sum((h_exp - h).^2));
end

```

The optimization script was defined as a separate function.

```

clear
clc

fun = @(x)objective_func(x);
x0 = 1.904e-5;
lb = [0,0,0];
ub = [];
A = [];
b = [];
Aeq = [];
beq = [];
options = optimoptions('fmincon','Display','iter','Algorithm','sqp');
nonlcon = [];

[x,fval]= fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon,options);

Cp = x(1)
C1 = x(2)
C2 = x(3)

```

After the optimized value Cp, C1, C2 has obtained, these values were defined into a new script to plot the model's output.

```

clear

% load('modell1.mat')
% t = Modell(:,1);
% h_exp(:,1) = Modell(:,4);
% h_exp(:,2) = Modell(:,6);
% u = Modell(:,2);
% h_sim(:,1) = Modell(:,3);

```

```

%      h_sim(:,2) = Model1(:,5);

load('model2.mat')
    t          = Model1(:,1);
    h_exp(:,1) = Model1(:,3);
    h_exp(:,2) = Model1(:,4);
    u          = Model1(:,2);
    h_sim(:,1) = Model1(:,5);
    h_sim(:,2) = Model1(:,6);

%model 01 tuned paramater
% cp = 1.9906e-05 ;
% c1 = 0.8031;
% c2 = 0.7740;
% dt = 0.5;

%model 02 tuned paramater
cp = 1.8550e-05;
c1= 1;
c2 = 1;
dt = 0.5;

h = zeros(length(t),2);      h(1,:) = h_exp(1,:)/100; %
Preallocation

for k = 1:length(t)-1
    h(k+1,:) = find_next_state(h(k,:), u(k),cp,c1,c2,
dt);
end

h = h*100;

figure('name','Primary Model')
subplot(2,1,1)
plot(t, h_exp(:,1))
hold on
plot(t, h(:,1))
hold on
plot(t, h_sim(:,1))
title('Water height tank 01')
xlabel('Time(s)')
ylabel('Water Height in tank 01(cm)')
legend('Real process','Tuned model','Simulated model')

subplot(2,1,2)
plot(t, h_exp(:,2))
hold on

```

```
plot(t, h(:,2))  
hold on  
plot(t, h_sim(:,2))  
title('Water height tank 02')  
xlabel('Time(s)')  
ylabel('Water Height in tank 02(cm)')  
legend('Real process', 'Tuned model', 'Simulated model')
```

Appendix C

The MATLAB script used in the optimizer function block for optimization of the process input.

```
function [Kp,Ki] = Optimizer(e2,eint,u1)

fun = @(x)((u1 - (x(1)*e2 + x(2)*eint))^2)/2);

x0 = [0.73460821,0.0219286];
A = [];
b = [];
Aeq = [];
beq = [];
lb = [0,0];
ub = [inf,inf];
options = optimoptions('fmincon','Display','iter','Algo-
rithm','sqp');
x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub,[],options);

Kp = x(1);
Ki = x(2);
```


Appendix D

MATLAB script for optimization of controller parameters using process model and output error as the performance criteria.

```
function [Kp,Ki] = Optimizer(y1,h1,e1,eint)

% Model parameters
Cp = 1.4602e-05;
A1 = 0.01131;
C1 = 0.6673;
alpha1 = 3.8485e-5;
g = 9.81;
y = sqrt(2*g*h1);
fun = @(x)((y1 - ((Cp*(e1*x(1)+eint*x(2)))/A1 - C1*alpha1/A1*y))^2);

%optimization function parameters
x01 = [0.73460821,0.0219286];
A = [];
b = [];
Aeq = [];
beq = [];
lb = [0,0];
ub = [inf,inf];
options = optimoptions('fmincon','Display','iter','Algorithm','sqp');

x = fmincon(fun,x01,A,b,Aeq,beq,lb,ub,[],options);

Kp = x(1);
Ki = x(2);
```

Appendix E

MATLAB script for optimization of controller parameters using MIT gradient descent rule with calculated $K_{p,s}$ value and same $K_{i,p}$ value and output error as the performance criteria.

```
function [Kp,Ki] = optimizerforaplha(y1,y2,Kpint,Kpn)
```

```
Kiint = 0.0219286;
```

```
alpha = 0.1;
```

```
% Kpn = 0;
```

```
% Kin = 0;
```

```
e = ((y1-y2)^2)/2;
```

```
if y1>y2
```

```
    Kp = Kpint + alpha*(e/Kpint-Kpn);
```

```
    Ki = Kiint;
```

```
elseif y1<y2
```

```
    Kp = Kpint - alpha*(e/Kpint-Kpn);
```

```
    Ki = Kiint;
```

```
else
```

```
    Kp = Kpn;
```

```
    Ki = Kiint;
```

```
end
```

Appendix F

MATLAB script for optimization of controller parameters using MIT gradient descent rule with calculated $K_{p,s}$ value and calculated corresponding $K_{i,s}$ value and output error as the performance criteria.

```
function [Kp,Ki] = optimizerforalpha(y1,y2,Kpint,Kpn)

alpha = 0.1;

e = ((y1-y2)^2)/2;

if y1>y2
    Kp = Kpint + alpha*(e/Kpint-Kpn);
    Ki = Kp/33.5;

elseif y1<y2
    Kp = Kpint - alpha*(e/Kpint-Kpn);
    Ki = Kp/33.5;

else
    Kp = Kpn;
    Ki = Kiint;

end
```

Appendix G

MATLAB script for optimization of controller parameters using MIT gradient descent rule with calculated $K_{p,s}$ and calculated $K_{i,s}$ value and output error as the performance criteria.

```
function [Kp,Ki] = optimizerforaplh(y1,y2,Kpint,Ki-  
int,Kpn,Kiin)  
  
alpha = 0.1;  
  
e = ((y1-y2)^2)/2;  
  
if y1>y2  
    Kp = Kpint + alpha*(e/Kpint-Kpn);  
    Ki = Kiint + alpha*(e/Kiint-Kin);  
  
elseif y1<y2  
    Kp = Kpint - alpha*(e/Kpint-Kpn);  
    Ki = Kiint - alpha*(e/Kiint-Kin);  
  
else  
    Kp = Kpn;  
    Ki = Kin;  
  
end
```

Appendix H

MATLAB script for connecting the OPC server with MATLAB [22].

- Initialize Client (One time)
 - `hr=mxOPC('open','Opc.DeltaV.1','localhost');`
- Set Device Mode (One time)
 - `r=mxOPC('ReadMode','Device');`
- Set Scan - Sleep (One time)
 - `mxOPC('Sleep',500);`
- Activate Sleep / Test missing scans (Once per scan)
 - `Nmissed = mxOPC('Sleep');`
- Read Value from the loop (Once per read/read)
 - `[value,hr]=mxOPC('ReadDouble','PID/PID1/PV.CV');`

The “read” value from the loop script can be written for each of the individual variables available in the OPC server. Also, when the “read” option is changed to the “write” option. The values can be sent to the variables in the OPC server.