PID TUNING OF PROCESS PLANT USING EVOLUTIONARY ALGORITHM

By

ROBERT LING LEH GUONG

FINAL PROJECT REPORT

Submitted to the Electrical & Electronics Engineering Programme
in Partial Fulfillment of the Requirements
for the Degree
Bachelor of Engineering (Hons)
(Electrical & Electronics Engineering)

Universiti Teknologi Petronas
Bandar Seri Iskandar
31750 Tronoh
Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

PID TUNING OF PROCESS PLANT USING EVOLUTIONARY ALGORITHM

by

Robert Ling Leh Guong

A project dissertation submitted to the
Electrical & Electronics Engineering Programme
Universiti Teknologi PETRONAS
in partial fulfilment of the requirement for the
Bachelor of Engineering (Hons)
(Electrical & Electronics Engineering)

Approved:

_____

Assoc. Prof. Dr. Irraivan Elamvazuthi

Project Supervisor

UNIVERSITI TEKNOLOGI PETRONAS

TRONOH, PERAK

September 2014

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

_____

ROBERT LING LEH GUONG

ABSTRACT

PID controller is one of the most robust and well-implemented controller in today's industry. The mature and stable performance of it had increased the usage of the PID controller in multiple fields such as process control, robotic and chemical plants. However, the advancement of technology has urged the industry to improve overall process in term of its overshoot, rise time, settling and other domains. In this project, Evolutionary algorithm (Particle Swarm Optimization) is implemented to optimize the controller parameters in order to improve the system performance of the real pressure plant. Simulation and experimental work are carried out side by side to prove the feasibility of the PSO method. The results show that PSO had successfully improved the overall system performance of the real pressure plant in term of percentage overshoot, rise time. There is always a trade-off for the system performance parameters (percentage overshoot, rise time and settling time) and it is depending on the type of applications.

.

ACKNOWLEDGEMENTS

Throughout the final year project, many people have been assisting and supporting author in completing the works. Therefore, the author would like to express the highly gratitude:

First, FYP supervisor Assoc. Prof. Dr. Irraivan Elamvazuthi for the knowledge he taught throughout the final year project. He guided all the way until the author achieved succeeds in his final year project. Moreover, the author was able to stay in the track and followed up the progress because of the helped from him in discussing the problems and solutions. Besides that, the author gained valuable knowledge from him in both academic and daily lives skills.

Next, the author would like to thank M Azhar Zainal Abidin for his help given, and he always assisted and setup the process plant for the author. He always gave the guidance and explained the details of the plant to the author. Indeed, his help has been very useful in understanding what the overall plant is about and let the author gaining knowledge of the project.

Furthermore, the most sincere appreciation greeted to my family and fellow friends who helped and supported all the way until the author completed the project. They always encouraged and gave advice whenever the author faced the hardship or being stuck in the progress. Their countenance and supportive deeds are indeed the main reason for the author achieved succeeds.

Last but not least, special thanks to Electrical & Electronics Engineering Programme Universiti Teknologi PETRONAS, giving chance to the author embarked in the final year project which prepared him for the job after graduation. The FYP course indeed is a huge challenge to the author and it gave the author a big view of the future works. The author managed applies the degree knowledge in the project and FYP did achieve its own objectives. Thereby, the author hopes that FYP will be continued and strengthen in order to achieve its objectives.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS AND NOMENCLATURES

PSO	Particle Swarm Optimization
CC	Cohen Coon
EA	Evolutionary Algorithm
PID	Proportional- Integral-Derivative
GA	Generic Algorithm

Greek letters

θ	Theta
τ	Tau

CHAPTER 1
INTRODUCTION

1.1 Background Study

Control mechanism is undoubtedly important in our daily lives to yield the better performance or to produce output of one system. The desired output of the system can be obtained by implementing various kinds of controllers. Until now, Proportional-Integral-Derivative (PID) controller remains its usability in the control engineering fields because of the effective yet simple implementation features. This kind of controller exhibits a good performance based on its mature implementation. Figure 1.1 shows the classic feedback loop diagram with PID controller.

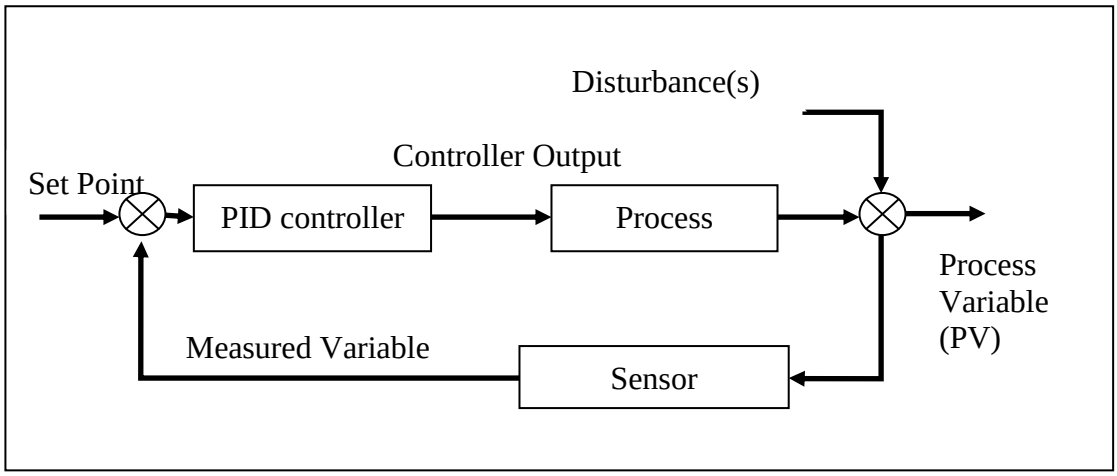

Figure 1.1: Classic feedback loop diagram with PID controller

Based on the figure 1.1, the PID controller plays the most important role to ensure the optimal output for the overall system. PID controller consists of three main parameters which are proportional, integral and derivative. By manipulating the parameters values, the controller can be designed for different specification applications. The outcome of the controller can be examined through the error compared to the previous set point, overshoot percentage, system stability, decay ratio and rise time. However, PID controller does not ensure the final output or system converge to the optimal results. Hence, the PID controllers have been enhanced by

some tuning techniques or algorithms in order to obtain the best output and adapt to the dynamic control problems.

The conventional PID controller such as Ziegler Nicholas (Z-N) method remains popular due to its simplicity yet this conventional tuning method yield a very high overshoot value. The conventional tuning method has indeed successfully improved the system performance. However, there are many optimization methods have been introduced to polish the current techniques. Optimization methods refer to the self tuning methods where program is developed to find the best solution. The iterations are performed throughout the process until the best/optimal solution is obtained. The latest search methods involved in the control engineering plant are genetic algorithm [1], evolutionary algorithm [2], particle swarm optimization [3], simulated annealing [4], and ant colony optimization [5].

1.2 Problem Statement

Based on the review given in Chapter 2, there are many shortcomings for the current techniques that are used for tuning the controllers. They are:

1. The existing control techniques are widely used in the process plant control to improve the output or to enhance the possible solutions in specified plant. However, conventional tuning method does not yield good results for system performance in term of percentage overshoot, rise time and settling.
2. Besides that, some of the methods are only done in simulation but not proved by the experimental work. Furthermore, the worst scenario happened when the running iterations converge to the wrong path and mislead to the final solutions.

1.3 Objectives

Based on the problem encountered in section 1.2, there are several objectives or goals for this project. There are:

1. To explore the existing algorithms which are particle swarm optimization algorithm. This method will be deeply studied and the experiment will be carried to find the optimal solution for the PID controller. The plants involved for the experiment is pressure plant.
2. To implement evolutionary algorithm into the tuning work. The method will be tested and the ultimate goal for the research is to obtain system performance with lowest overshoot, settling time and rise time.
3. Both simulation and experimental work will be carried out to check the check and validate the finalize results.

1.4 Scope of Study

The proposed solution of this research will focus on how to improve the system performance in both simulation and real-time application.

1. The main scope of the project is applying evolutionary algorithm to obtain the best tuning parameters. The evolutionary algorithms that are chosen for the project is particle swarm optimization (PSO). The PSO program is designed to yield the optimal controller parameters to improve the system performance of PID controller.
2. The simulation work will be done first then after that the optimal solution which obtained from the PSO program will be run in the real plant. After that, comparison will be made between the conventional tuning method and evolutionary algorithm, PSO.

CHAPTER 2

LITERATURE REVIEW

2.1 PID tuning

PID controller has been utilized in control engineering fields since 1940, and it is approximately 90% of the control mechanism adopted this controller [6]. It is believed that the PID controller could provide simplest yet effective results if compared to manual handling and other types of controller. The idea of PID controller has been implemented anywhere such as transportation, process plant, production, and manufacturing. PID control is normally combined with different function blocks, sequential function and logic. PID controller will react to the disturbance of the process or the set points given to the system and corrective action will take place in the valve in order to maintain the performance. PID controller can be described in Equation (1):

$$MV(t) = K_c\left[E(t) + \frac{1}{T_I}\int_0^{\infty} E(t')dt' - T_D\frac{dCV(t)}{dt}\right] \qquad (1)$$

In the Equation (1), it is clearly shown that proportional gain, integral time and derivative time will affect the opening or the movement of the control valve. Hence, tuning is undoubtedly an important task in order to make sure the performance of the system is always on top of it. Hence, different tuning method had been introduced into PID tuning world for the past few decades. The conventional tuning method still remains its popularity because of the simple implementation and some of them are Ziegler Nicholas (Z-N) method [7], on-line tuning [8], and auto tuning methods [9]. Yet, these kinds of low order process tuning methods do not applicable to dynamics control mechanisms because system will suffer from stability problem [10].

2.2 Related Work

Table 2.1 and table 2.2 show the convention and intelligent PID controller which have been used few decades ago.

Table 2.1: Literature of Conventional PID Controller

No	Author	Year	Title	Method Involved	Application	Merits	Demerits
1	A. A. Vodat And I. D. Landau	1995	A Method For The Auto-calibration Of PID Controllers	Auto-calibration method ZN	PI PID	Auto calibration perform better in term of overshoot and time response	ZN has large overshoot
2	Jyh-cheng Jeng, Wan-ling Tseng And Min-sen Chiu	2013	A One-step Tuning Method For PID Controllers with robustness Specification Using Plant Step-response data	One-step Tuning Method	PID Controller	Obtains the PID settings directly using the step response data of the process	Susceptible to uncertain process
3	Saeed Tavakoli and Mahdi Tavakoli	2013	Optimal Tuning Of PID Controllers For First Order Plus Time Delay Models Using Dimensional Analysis	Proposed Method ZN Cohen Coon	PID Controller	Proposed method have the best results among all the methods	ZN has largest overshoot, Cohen Coon methods have bigger ISE/IAE

Table 2.2: Literature of Intelligent PID Controller

No	Author	Year	Title	Method Involved	Application	Merits	Demerits
1	Enzeng Dong, Shuxiang Guo, Xichuan Lin, Xiaoqiong Li and Yunliang Wang	2012	A Neural Network-based Self-tuning PID Controller of anAutonomous Underwater Vehicle	Self tuning neural network	PID controller	fast convergence rate	Complex network involved
2	B Vasu Murthy, Y V Pavan Kumar,3U V Ratna Kumari	2012	Application of Neural Networks in Process Control: Automatic/Online Tuning of PID Controller Gains for + 10% Disturbance Rejection	Artificial Neural Network - Intelligent predictors	PID controller	Good disturbance rejection up to ± 10%	Simulation results only
3	Tao Ai, Jun-qi Yu,Yan-feng Liu,Jiang Zhou	2010	Study on Neural Network Self-tuning PID Control forTemperature of Active Solar House Heating System	Neural network self tuning	PID Controller	Improve control accuracy and good adaption to the system	-

Table 2.1 and table 2.2 show the convention and intelligent PID controller which have been used few decades ago. The stable and mature implementation of these controllers had proven that its usability still remains popular and many industries are still using these controllers in their operations. The Ziegler Nicholas tuning method remains its usability at certain plant process but this method will lead to large overshoot and undesirable damping ratio [7, 9]. While the on-line tuning method and auto tuning method did improve the overall system but further modification and refinement is needed to develop more sophiscated tuning strategy. In conjunction with the tuning methods above, some researchers found that intelligent PID controllers would be better because the existing PID controller is integrated with intelligent control technology such as genetic algorithm (GA), fuzzy control and neural network [11]. Fuzzy PID controller is proved that it can adapt well in nonlinear complicated process under high uncertainty of noise and different parameters [11, 12].

Besides that, neural network PID controller is more practical and have a robust result than the conventional PID controller [13]. Apart from that, author of [14] utilize self tuning of PID controller with neural fuzzy to solve the problem arise from solar housing heating system. The work proved that neural fuzzy self tuning method is able to tackle the shock problem and it exhibited shorter adjustment time with zero offset at steady state. [15] also implemented intelligent tuning method (neural network) to maneuver autonomous underwater vehicle (AUV). Two layer of neural network is used to control the parameters of PID controller while three layer neural networks are used to identify the linear velocity of the AUV. Results showed that the AUV is able to track different signal and can be controlled precisely with the implementation of self tuning neural network method.

Apart from that, the drawback of the controllers cannot be ignored since most of the industry is focusing on the accuracy, profit margin and the operation time. Hence, there are some algorithms have been implemented recently in order to produce robust and practical system especially when these algorithms could provide optimal results than other normal tuning methods. These algorithms may refer to artificial intelligence (AI) which exhibited by software that can provide maximum or optimal results to one's system. Table 2.3 shows the literature of artificial intelligence or evolutionary algorithm of the pass research.

Table 2.3: Literature of the Artificial Intelligence/Evolutionary algorithm

No	Author	Year	Title	Method involved	Application	Merits	Demerits
1	B.Nagaraj and Dr.N.Murugananth	2010	A comparative study of PID controller tuning using GA, EP, PSO and ACO	GA EP PSO ACO	PID controller	All algorithm perform better than conventional PID controller	Algorithms cannot guarantee the convergence of the final solution
2	*MohammadSadegh Rahimian and Kaamran Raahemifar*	2011	Optimal Pid Controller Design For Avr System Using Particle Swarm Optimization Algorithm	PSO GA	PID controller-automatic voltage regulator	PSO use less number of iterations to achieve final optimal results.	-
3	Rushil Raghavjee and Nelishia Pillay	2012	A Comparison of Genetic Algorithms and Genetic Programming in Solving the School Timetabling Problem	GP GA	School Timetabling Problem	GP yield better result and scale	Take time to evolve a program
4	Suraj Sharma Sanjeev S. Tambe	2014	Soft-sensor Development for Biochemical Systems using Genetic Programming	GP MLP SVR	Soft sensor	GP yield most accurate result and generalization capability	-

In [16], results proved that PSO and GA methods surpass other tuning methods such as Z-N method, EP and ACO in term of its settling time, rise time and overshoot. This has clearly stated that using algorithms in PID problem could yield better result than conventional PID controller as well as intelligent PID controller.

Furthermore, genetic algorithm (GA) is a metaheuristic /heuristics method used to optimize the search problems. This algorithm uses mutation, crossover, and reproduction to achieve the optimal results but it requires control theory to have proper initial control values [16, 17]. Another type of optimization method in PID control field is particle swarm optimization (PSO). The method could solve almost all the non linear optimization problem but the output may easily fall into local minima [18]. Besides that, this method requires less time than others and it can be applied to any PID parameters control by simply change some of the constraints [19]. Moreover, genetic programming is another type of evolutionary algorithms use to find a computer program that can perform the ordered tasks. This method requires zero knowledge on control theory and it can evolve any kind of structure to solve the problems [17].

2.3 Discussion

Table 2.4 shows the literature or the related work of the Particle Swarm Optimization.

Table 2.4: Literature of the Particle Swarm Optimization

No	Type of application	Author	Year	Title	Merits	Demerits
1	Process Plant	Zhangjun, Zhang Kanyu	2011	A Particle Swarm Optimization Approach for Optimal Design of PID Controller for Temperature Control in HVAC	Good convergence result and precise computation	-
2		Mohd Shariq Khan, Yuli Amalia Husnil, Yong Soo Kwon, and Moonyong Lee	2011	Automated Optimization of Process Plant Using Particle Swarm Optimization	Gradient free method	Time consuming
3	Other Application	Wei Tao and Zhang Shun Yi	2008	Active Queue Management Based on Particle Swarm Optimization PID Algorithm	Adapt to dynamic network, and minimize the queue error	-
4		Sebnem Demirkol Akyol and G. Mirac Bayhan	2011	A Particle Swarm Optimization Algorithm for Maximizing Production Rate and Workload Smoothness	Solve real line balancing (non-linear) problem	-
5		Luis Rodríguez-García, Sandra Pérez-Londoño and Juan Mora-Flórez	2013	Particle Swarm Optimization applied in Power System Measurement-Based Load Modeling	Able to obtain sufficient estimation of parameters	-
6		Makoto Tokuda and TOfU Yamamoto	2010	A Data-Driven Modeling Method Using Particle Swarm Optimization	Accurate modeling	-

Among all the tuning methods, particle swarm optimization is chosen as the tuning method in this research. [19] compared two optimization algorithm which are PSO and GA and found that PSO was able to achieve final optimal results with less iterations. Besides that, the author claimed that PSO is able to apply in other fields if some changes made to the basic parameter and constraints. In [3], PSO is used to optimize the process plant under the process simulator (Hysys), and it showed that PSO is able to predict the optimum value by ignoring the gradient problem yet it is time consuming in objective function. Apart from that, [20] has proposed PSO to maximize the production rate and workload smoothness in the industry applications. The results showed that PSO is able to solve the line balancing problem with minimum iteration times.

Table 2.4 provides the literature for particle swarm optimization for the previous work and it is divided into process plant and other application. [3] and [21] described the related work PSO algorithm in process plant. These two papers provides encouraging outcome where [21] had proven that PSO give a better convergence results and it is able perform computation precisely. Besides that, [20,

22-24] are the literature for other application apart from process plant. This showed that PSO not only utilize in solving PID controller problems, it does use to resolve other real life problem such as line balancing, data driven modeling, production rate and other domains. The related work for particle swarm optimization had proved that PSO is able to solve the real plant problem and it does yield the better performance in solving any kind of problem. Hence, this evolutionary algorithm will be used to investigate the improvement or performance toward the targeted plant in this project.

CHAPTER 3

Methodology

3.1 Project methodology

After reviewing some EA method, particle swarm optimization is chosen as the research focus point. The project can be divided into two parts which are real plant experiment and simulation. The experimental work is done in the laboratory of control system while the simulation can be run in the MATLAB program. For the experimental work, it can be divided into two parts which are conventional tuning method (Cohen Coon) and PSO, while simulation work is also divided into two parts which are conventional tuning method (CC) and PSO.

The software used in the project is MATLAB while the real process plant (pressure) will be used to run the experiment. The final result of the simulation and plant will be compared in order to have more convincing result. The process of the research is shown in Figure 3.1. The Gantt chart and the key milestone of the project are shown in Appendix A.

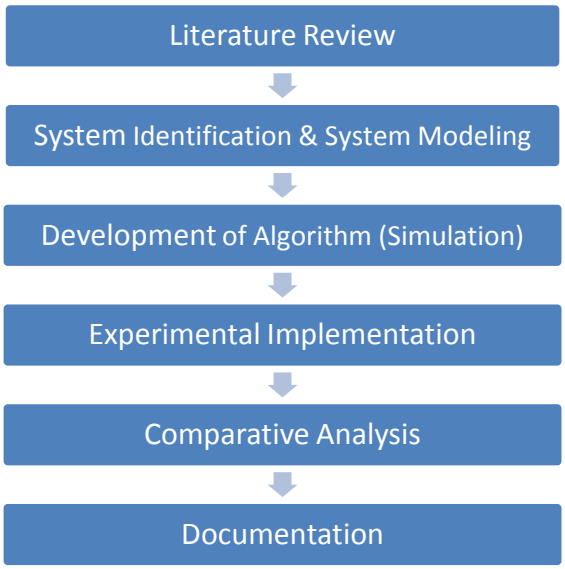

Figure 3.1: Process of the research for Final Year Project

3.2 System Identification and System Modeling

The chosen plant for the project is the pressure plant as shown in figure 3.2.

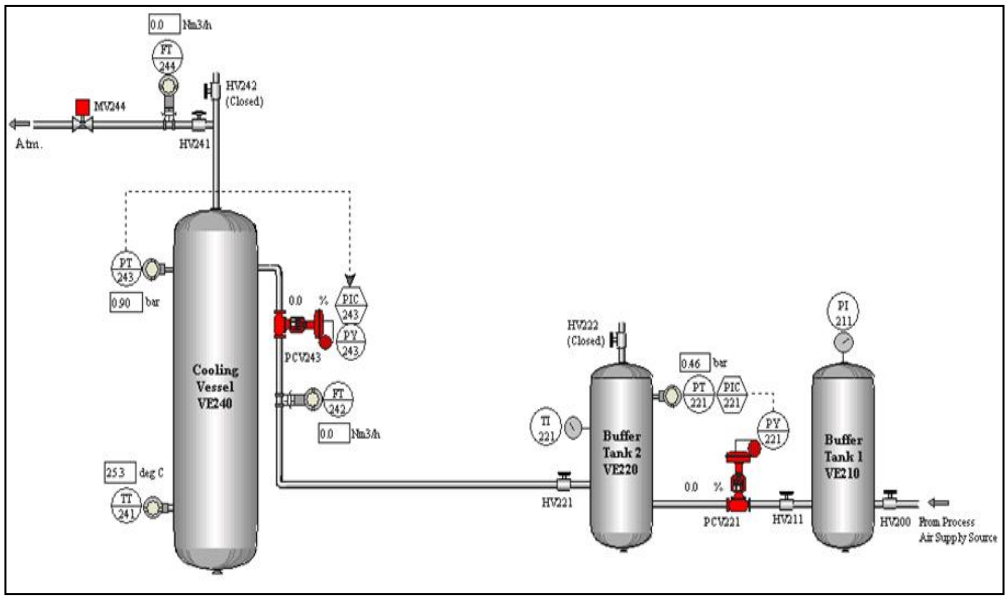

Figure 3.2: Schematic Diagram of Pressure Plant

System identification is the first process and it is the most important part of the project. System identification and system modeling are used to conceptualize and structure the system of its input-output data in a unique way. Open loop test is performed in order to obtain an input-output curve from the process. The open loop test is carried by adjusting the controller to manual mode. After obtaining the PRC, the modeling method will be used to extract the data and the method used is empirical modeling. The calculation of the empirical modeling is shown in figure 3.3.

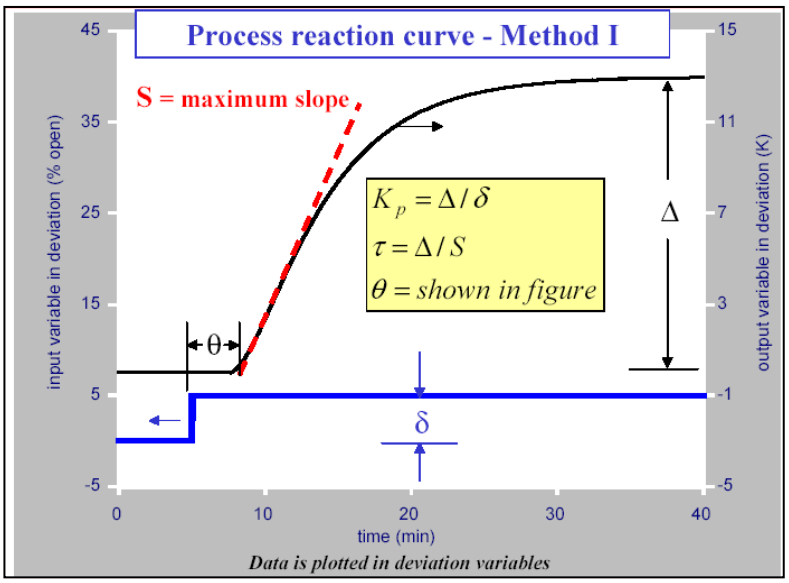

Figure 3.3: Empirical modeling (Method I)

3.3 Development of Algorithm (Simulation)

There are two main parts for the algorithm or the program in this project. The first program is used to checking the system performance of the PID controllers, while the second program is the PSO program which used to optimize the controller parameter. The first program is designed to measure the percentage overshoot, rise time and settling time of the controllers and it is shown in figure 3.4.

```
s = tf('s');

G = exp(-θ*s)*(Kp/(τ*s+1));
C = pid(kp,ki,kd);

Tcl = feedback(G*C,1);
Tnd1 = pade (Tcl,1);
t=0:0.01:100;

step (Tnd1,t);
S = stepinfo(Tcl,'RiseTimeLimits',[0,0.9])
```

Figure 3.4: System Performance Program

The ‘pid’ function is the existing function of the MATLAB software, the value of proportional, integral and derivative are directly inserted into it and the program can be used to calculate the system performance. ‘Step’ function is used to perform the step change and ‘stepinfo’ will provide the details of the system response such as overshoot, undershoot, settling time and rise time. The ‘feedback’ function will automatically provide the feedback formula to the equation and the sensor in the program above is set to be 1 which called unity sensor.

Lastly, the ‘pade’ function is the Padé approximation and the degree of the approximation is the one which indicates the approximation will provide first order formula. The first order with dead time formula obtained from the process reaction curve is simplified by using Padé approximation [25] and the approximation formula is shown is Equation (2).

$$e^{-\theta s} = \frac{1-\frac{\theta}{2}s}{1+\frac{\theta}{2}s} \qquad (2)$$

By changing the exponential term into linear term, the characteristics of the overall transfer function will alter but conclusively it does simplified the calculation and it does cater for PSO programming.

The second program that is used in the project is PSO program. There are two parts for the PSO program which are the main program and the SIMULINK. The main program is used to compute the calculation of positions and velocities of the particles and the details of the main program will be discussed in section 3.5. Learning factors (cognitive weight, c1 and social weight, c2) and number of particles, n are the important parameters that decide the final solution of the program. Hence, the program will be tested with different combination of these parameters.

Besides that, SIMULINK is designed to compute the iteration of closed-loop system as shown in figure 3.5. For the PSO program, process equation is the only information that is needed from the real pressure in order to optimize the final controller parameters and it is shown in section 3.4.

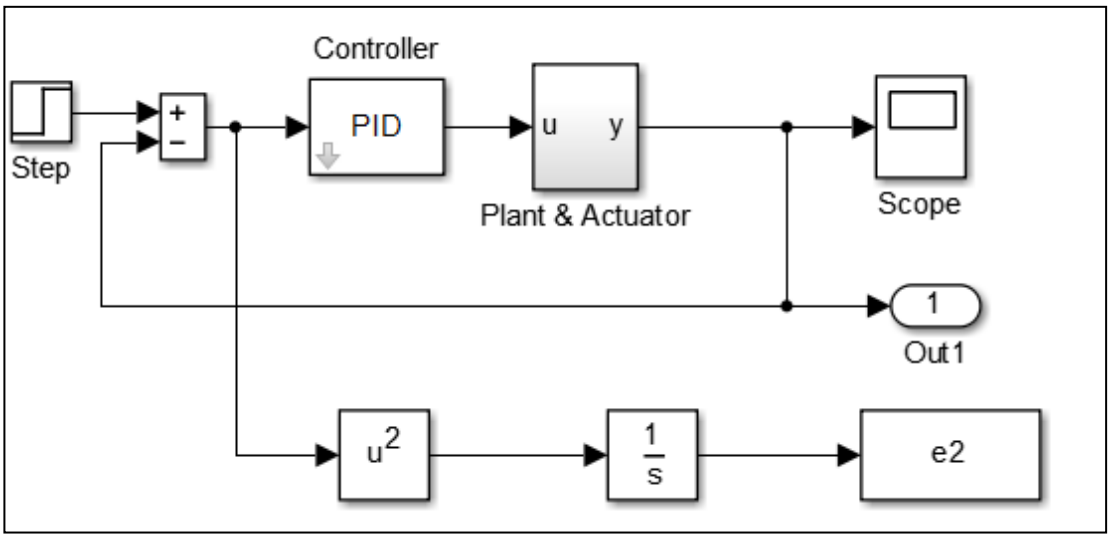

Figure 3.5: SIMULINK Diagram of PSO

3.4 Experimental Implementation

3.4.1 Conventional Tuning Method

Firstly, the experimental work is carried out on the interested plants which are pressure plant in order to obtain process reaction curve. The reaction curve can be obtained from the plant by manually operating the system which performing the open loop test. After obtaining the PRC, the useful parameters such as steady state gain (K_p), time constant (τ) and dead time (θ) can be extracted from it. The first order with dead time formula is used to illustrate the result from the PRC graph and it is illustrated in Equation (3):

$$G_p(s) = \frac{K_p e^{-\theta s}}{\tau s + 1} \qquad (3)$$

The controller transfer function can be found with FODT formula by using different tuning method such as Ziegler Nicholas, Ciancone and Cohen Coon and it is shown in Equation (4):

$$G_c(s) = K_p + \frac{K_i}{s} + K_d s \qquad (4)$$

In this project, Cohen Coon tuning method is chosen to calculate the parameters (K_p, K_i, K_d) of the controller of the plant. The formula of Cohen Coon tuning method can be found in Appendix B. After obtaining the different controller parameters (P, PI and PID), the system performance is checked by inserting the P, I and D values into the real system. The system response curve is recorded and further analysis is done on the response curve.

3.4.2 PSO

Experimental work of evolutionary algorithm will be performed after the PSO program yield the optimal solution. The PSO program will give optimal solution when the stopping criterion is achieved and the values of controller parameter are put back into the real system to carry out the performance test. The response curve will be recorded and comparison will be made between conventional tuning method and evolutionary algorithm.

3.5 Particle Swarm Optimization (PSO)

3.5.1 Principle of PSO

Particle swarm optimization (PSO) algorithm is developed by James Kennedy & Russell Eberhart in 1995 and it is inspired by the behavior of fishes and flock of birds. This algorithm has similar concept with genetic algorithm but it converge faster and PSO requires less parameter to tune the controller parameters. Besides that, PSO has the advantage of finding solution in large search space by investigating through position and velocity of the swarm (particles). The PSO equation is represent in Equation (5, 6)

$$S_i^{k+1} = S_i^k + V_i^{k+1} \quad (5)$$

$$V_i^{k+1} = w*V_i^k + c_1*rand_1( )* (pbest_i - s_i^k) + c_2*rand_2( )*(gbest - s_i^k) \quad (6)$$

Where v_i^k : velocity of agent i at iteration k,

w: inertia weight,

c_j : learning factor,

rand : uniformly distributed random number between 0 and 1,

s_i^k : current position of agent i at iteration k,

$pbest_i$: pbest of agent i,

gbest: gbest of the group.

There are few important parameters in particle swarm optimization tuning process which are learning factor (c1 and c2), inertia weight (w) and number of particles involved. The path taken for a particle is described in figure 3.6.

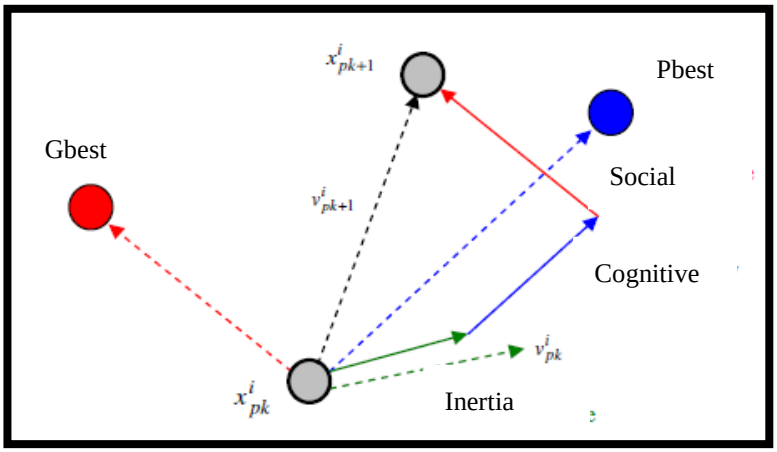

Figure 3.6: Pathway Description of Particles Velocity

The cognitive factor, c1 which is also known as learning factor is the parameter that influences the swarm velocity toward the local best position. While the social factor, c2, is the factor that affect the particles towards the global best position. The range of cognitive weight and social weight is bounded between 0 and 4. If the selected value is too large, the particles may fly over the convergence range and bound outside the range, and if the selected value is too small, it takes times to find the optimal local particles position.

Besides that, inertia weight is considered as an important factor that will influence the convergence rate of PSO's program. It is used to adjust the outcome of the previous velocities on the subsequent velocities. Inertia weight is first introduced in 1998 by Shi and Eberhart [26]. The inertia will affect the convergence rate or in other word, larger value of *w* will give thorough global search while small value of *w* will provide good local exploration. In this project, we fix the value of inertia weight to maximum of 0.9 as proved in [19] and [27]. Lastly, number of particles, n will influence the search ability by given the same size of space and large number of particles will definitely give a better and stable optimal solutions.

3.5.2 PSO algorithm

There are several steps in this algorithm as shown in Figure 3.7.

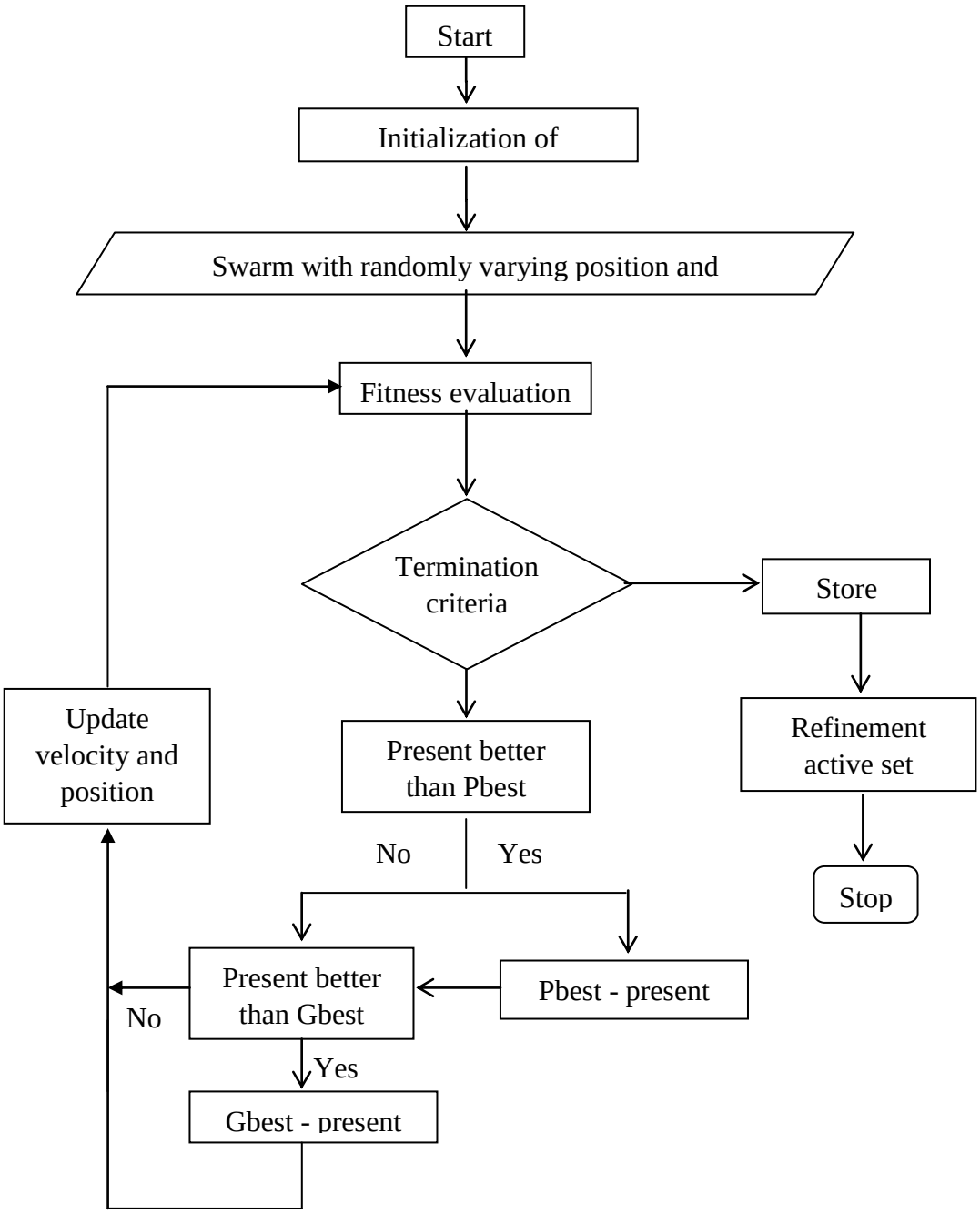

Figure 3.7: Flow chart of Particle Swarm Optimization algorithm

The flow chart of the PSO algorithm is shown in figure 3.7 and each step is clearly described as below.

1. Population (particles) initialization with stochastic position and velocities within the search space.
2. Each particle in the swarm is evaluated with the desired fitness function.
3. Local best particle (Pbest) is compared with the current particles, if the current particle is better than the Pbest, then current particles will be the Pbest and its location refers to the current location in the d-dimensional space.
4. Compare the fitness with the overall previous best particles. If the current is better than the previous particles, then latest array index and value will be the current particle.
5. Manipulate the position and velocities of the particles according to the Equation (5,6):
6. Perform the step 2 until step 5 until the terminal criterion is achieved and the output will be the optimal solution for the controller parameter.

3.6 System Performance

After completing the experimental work and simulation, the results are tabulated into table form and further analysis will be made to compare the system performance of the different tuning method. The parameters that are interested for the system performance are percentage overshoot, rise time and settling.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Experimental Work

4.1.1 System Identification

Figure 4.1 show the process reaction curve of pressure plant and it is obtained through open loop test (without controller).

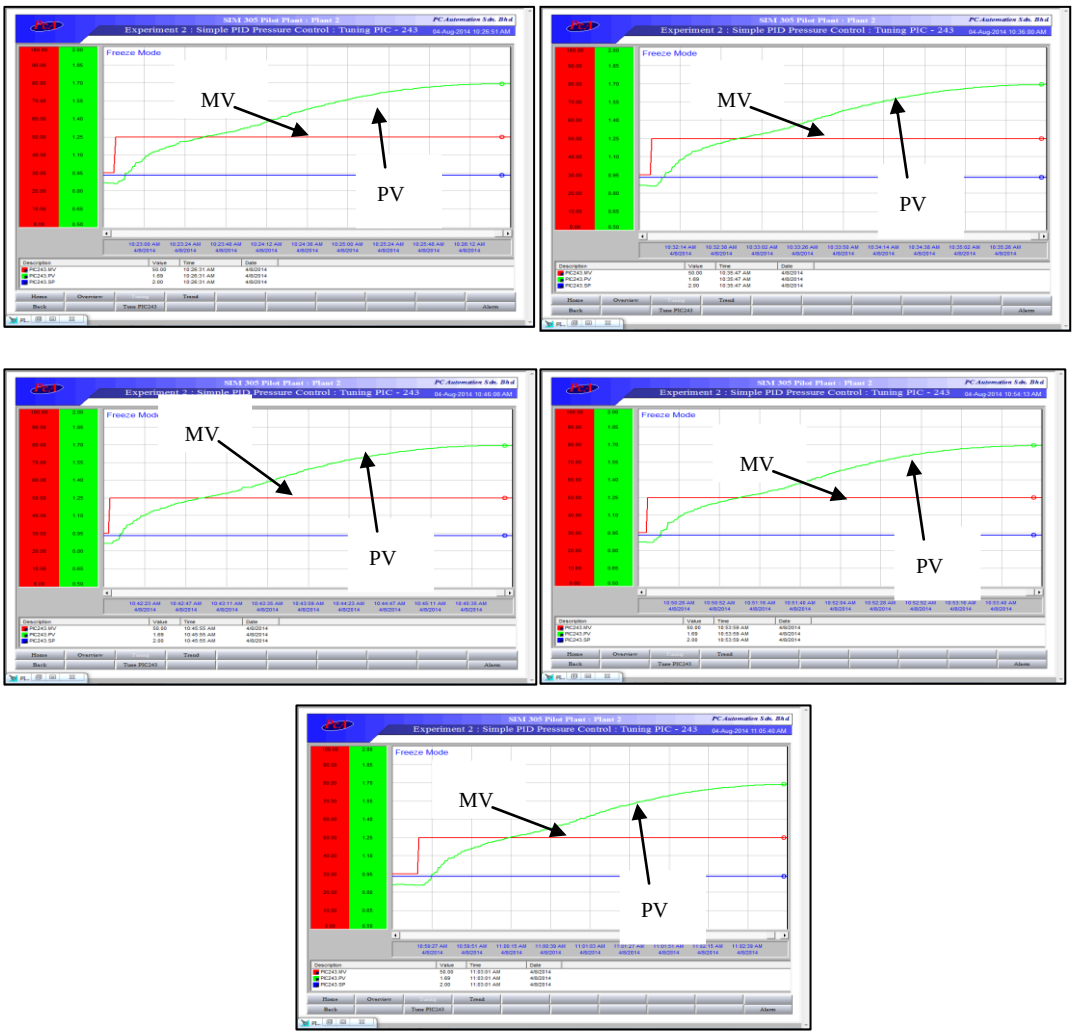

Figure 4.1: Process Reaction Curve (PRC) of Pressure Plant

5 open loop experiments are carried out in order to obtain the average result and increase the accuracy of the outcome. The process reaction curve is basically the reaction of the process plant when the step change is applied to the process. The curve of the process can be used to calculate steady state process gain (K_p), apparent time constant (τ), and dead time (θ). Method 1 of empirical modeling is used to extract the data from the process reaction curve and the detail of the method is shown in figure 3.3. The result is tabulated into table 4.1.

Table 4.1: Results of Process Reaction Curve (Method 1)

Measurement	**Value**
Change in perturbation / MV, σ	0.2
Change in output / PV, Δ	0.82
Maximum slope, S	0.015185
Calculations	Value
Steady State Process Gain, $K_P = \Delta / \sigma$	4.1
Apparent time constant, $\tau = \Delta / S$	54 seconds
$\theta = 0.63\Delta - \tau$	3 seconds
$R = \frac{\theta}{\tau}$	0.055555

After obtaining the overall results of the PRC, completed first order with dead time formula can be obtained by applying the formula, the FODT formula of the pressure plant is shown in Equation (7).

$$G_p(s) = \frac{K_p e^{-\theta s}}{\tau s+1} = \frac{4.1e^{-3s}}{54s+1} \quad (7)$$

This is the estimated process formula and it will be further used during the simulation process in MATLAB.

4.1.2 Performance of different Controllers

The value of steady state process gain, apparent time constant and dead time is used to calculate the controller parameter which are proportional, integral and derivative values. Cohen Coon tuning method (conventional) is used to test the system performance and CC open loop Correlations table is shown in Appendix B. Three controllers are used to compare the performance and the value of tuning parameters is recorded in table 4.2.

Table 4.2: PID controller Parameter for Pressure Plant

Tuning Parameters:	P-only	PI	PID
Proportional Gain, K_c	4.47	3.97	5.91
Integral Time, T_I (minutes/repeat)	-	8.95	7.21
Derivative Time, T_D (minutes/repeat)	-	-	1.08

Step change is applied to the system to observe the system performance and the system response graphs are shown in the figure 4.2-4.4.

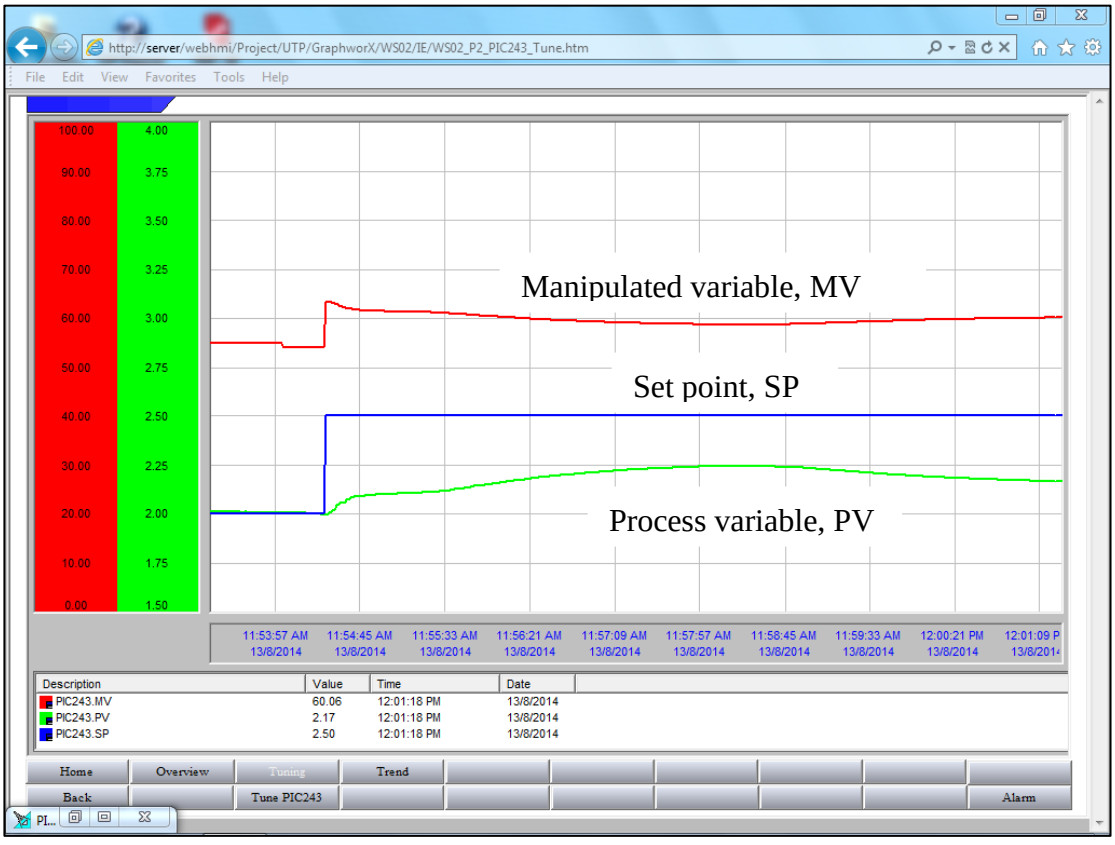

Figure 4.2: System Response Curve for P Controller

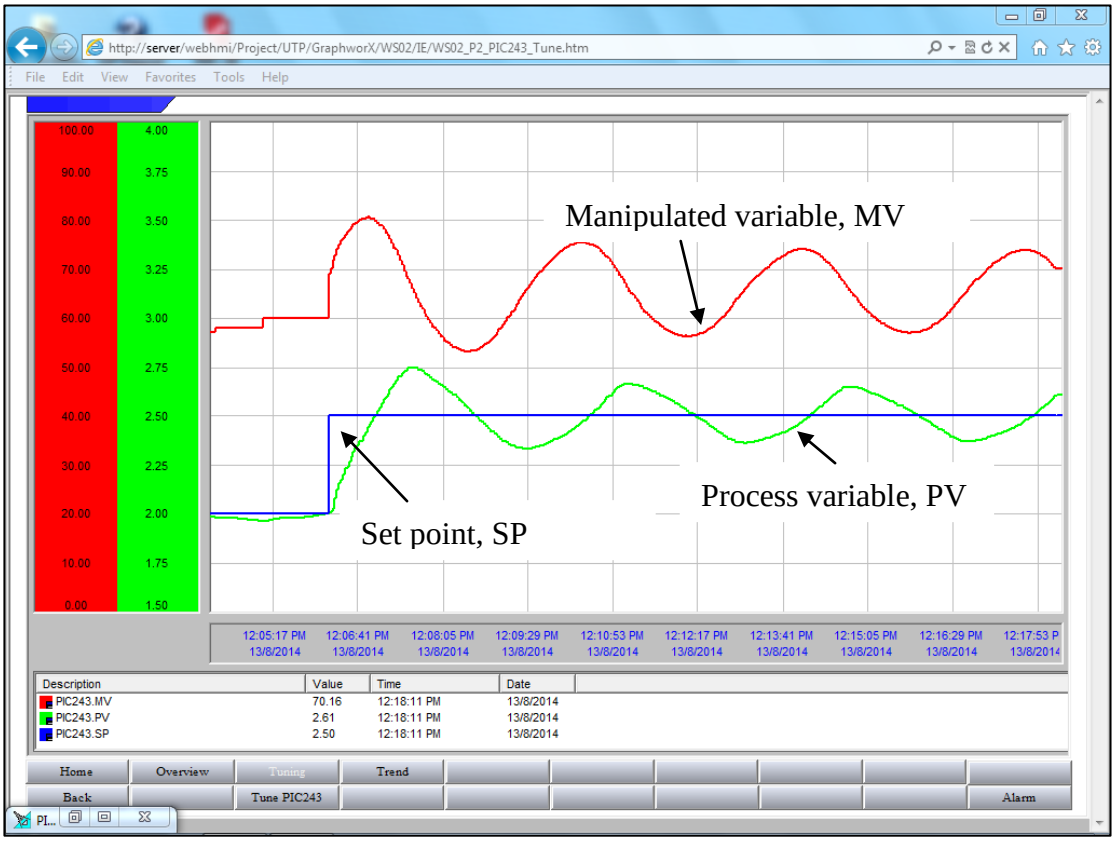

Figure 4.3: System Response Curve for PI Controller

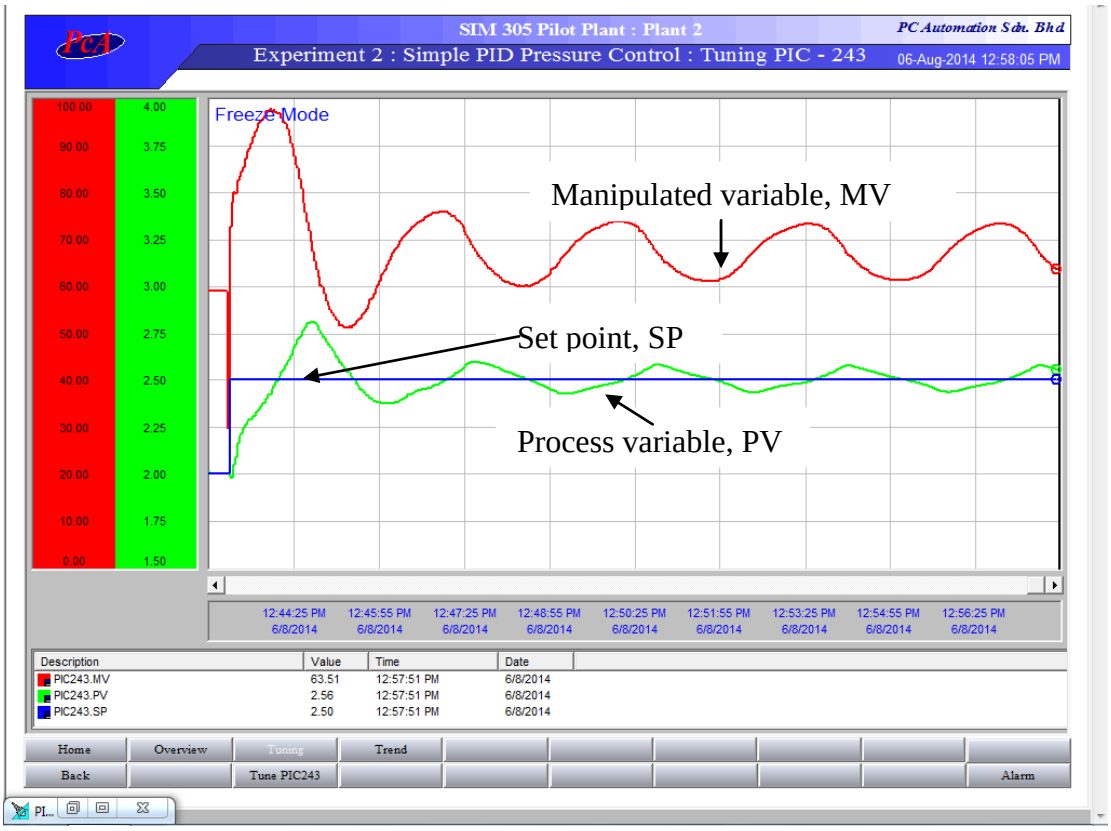

Figure 4.4: System Response for PID Controller

Based on the system response curve for P, PI and PID controller, the results are tabulated into table 4.3 to compare the system performance.

Table 4.3: System Performance of Three Different Controllers

Tuning Parameters:	P-only	PI	PID
Overshoot, %	-	50	62
Rise time, min	-	0.92	0.88
Settling time, min	6	9	7

Based on the results of system performance in table 4.3, we can conclude PID controller is the best controller because it exhibits the lowest overshoot, and fastest. Both of the PI and PID controller do not converge but PID controller have better performance in settling time because the valve opening is smaller and it has smaller range oscillation at the set point. P controller is neglected because it does not reach steady state as integral mode has not been applied in this controller, which means steady state error existed at the end of reaction. Hence, PID controller is proved to be the best controller and the focus point of the project will be PID controller instead of using P or PI controller. Besides that, the PID value (5.91, 7.21 and 1.08) from the conventional Cohen Coon method will be used to compare the system performance of PSO method in section 4.3.

4.2 Simulation Work

PSO program is designed to optimize the controller parameters of the pressure plant. The program itself will generate different values of final results based on the parameters inserted in the program. The main parameters that will influence the outcome are cognitive weight (c1), social weight (c2) and number of particles ( ) involved. In the project, some combination has been tested in order to find the best value of controllers' parameters.

4.2.1 The Effect of Varying the 'Number of Particles, n'

In order to investigate the effect of each parameters towards the values of controllers, first of all learning factor are kept constant at 2 while number of particles, n is varied with four different values as shown in table 4.4-4.7.

Table 4.4: Controllers Parameters with n values of 10

	n=10, c1=c2=2									
Kp	5.39	5.80	14.01	3.90	11.61	17.23	7.48	7.51	2.92	8.61
Ki	0.73	0.90	0.38	0.34	0.47	0.40	0.38	0.39	0.43	0.44
Kd	5.60	29.15	15.10	10.58	32.86	64.43	21.29	10.17	2.40	10.98

Table 4.5: Controllers Parameters with n values of 20

	n=20, c1=c2=2									
Kp	4.54	6.57	6.82	6.64	5.6	4.6	5.94	7.06	5.62	6.27
Ki	0.79	0.53	0.81	0.54	0.47	0.39	0.35	0.46	0.64	0.36
Kd	10.31	7.81	15.71	6.78	7.65	7.34	7.22	8.29	5.75	11.86

Table 4.6: Controllers Parameters with n values of 30

	n=30, c1=c2=2									
Kp	6.54	4.88	4.97	6.65	4.35	4.83	4.97	5.35	5.33	5.77
Ki	0.63	0.60	0.59	0.66	0.80	0.48	0.63	0.31	0.54	0.55
Kd	6.66	7.29	8.46	8.48	4.07	7.43	4.50	6.42	9.10	5.05

Table 4.7: Controllers Parameters with n values of 50

	n=50, c1=c2=2									
Kp	5.53	5.77	5.23	5.57	7.45	6.33	6.40	5.20	5.76	6.76
Ki	0.50	0.55	0.87	0.66	0.61	0.49	0.64	0.63	0.60	0.59
Kd	6.96	5.70	8.20	9.94	8.23	8.14	7.71	6.48	8.97	11.62

The range of each controller parameters is plotted into graph in figure 4.5-4.7 to investigate how the number of particles influences the output of the controller parameters.

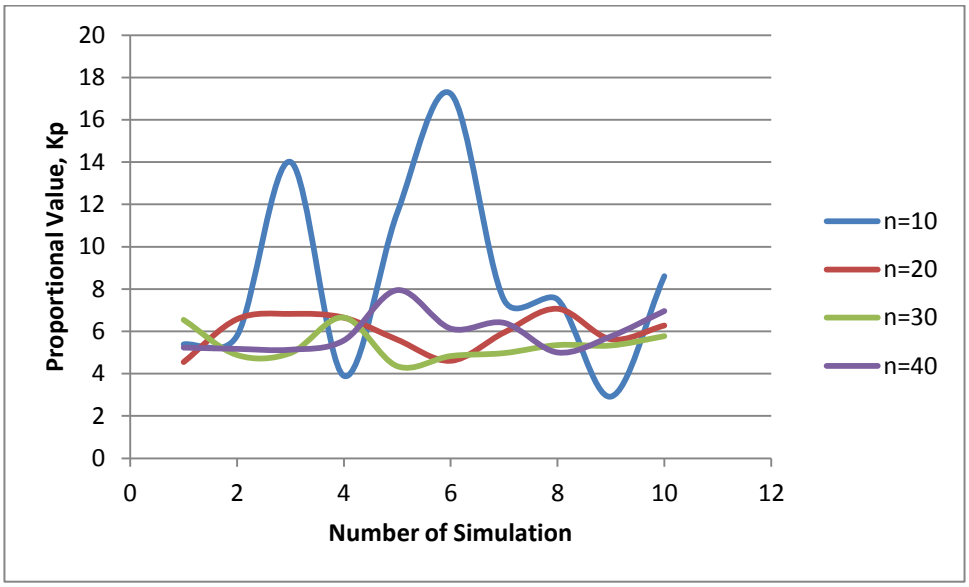

Figure 4.5: Kp value against number of simulation

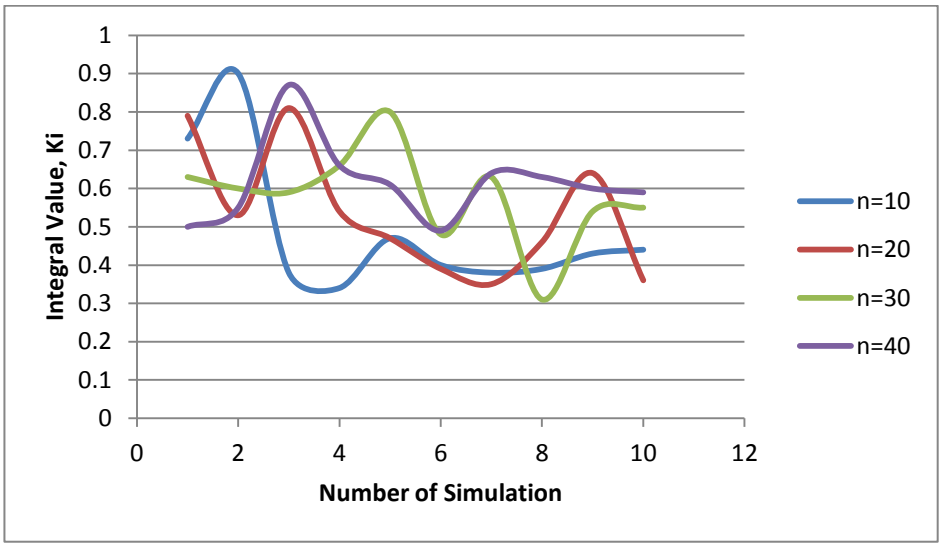

Figure 4.6: Ki value against number of simulation

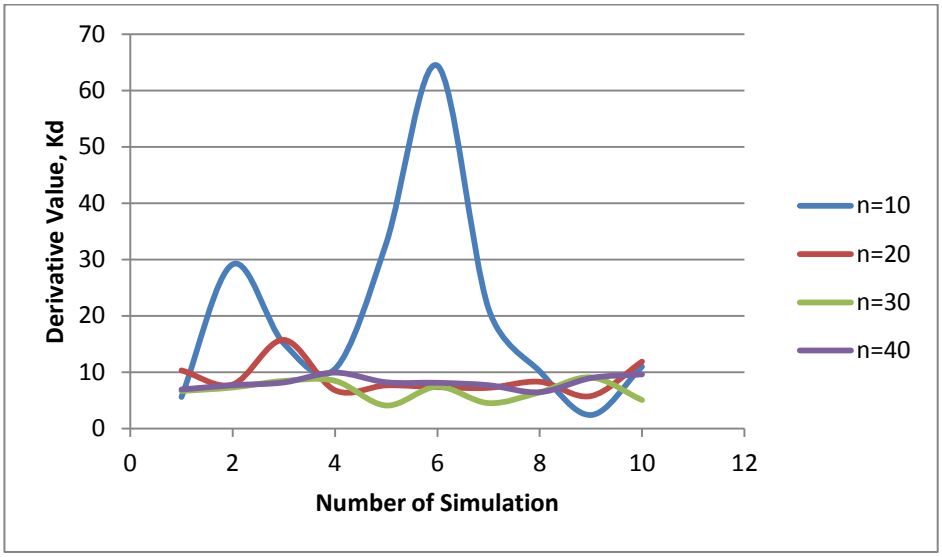

Figure 4.7: Kd value against number of simulation

Based on the graph from the figure 4.5-4.7, we can conclude that when the number of particles, n involved in the swarm increased, the outcome will be more stable and the range of the final optimal value will converge to specified local point. Besides that, larger number of particles will yield better results and it has higher chance to get the best optimal results. From the figure 4.5-4.7, we can deduce that P, I and D value become stable and constant within certain range when the number of particles more than 30. However, increasing number of particles will increase the computational time and the average computational time for the simulation above is tabulated in the table 4.8.

Table 4.8: Simulation Time for Different Number of Particles

n	10	20	30	50
Computational time, min	0.83	1.75	2.67	7.58

From the table 4.8, the simulation time for number of particles from 10-30 is considered shorter and faster. Yet, computational time for n=50 took 7.58 min to compute the final optimal solution. Hence, the ideal and optimum value for the number of particles is set to be 30 because it can yield stable results and it does not take long time to compute. So, 30 particles will be used in other combinations for the simulation program.

4.2.2 The Effect of Varying the 'Learning Factors (c1 and c2)'

By knowing the suitable number of particles (n=30), we can tune the PSO program by changing or varying the cognitive weight, c1 and social weight, c2 in order to find the final optimal value. Some of the combinations have tried to compute the final results and the results are shown in table 4.9-4.14.

Table 4.9: Learning factor (c1=c2=1) against PID value

	n=30, c1=c2=1									
Kp	6.54	4.25	5.19	4.09	5.06	5.31	5.50	5.73	4.87	4.23
Ki	0.49	0.31	0.44	0.44	0.33	0.48	0.39	0.40	0.40	0.63
Kd	8.44	4.99	4.83	7.62	4.75	5.77	5.91	5.94	6.70	6.80

Table 4.10: Learning factor (c1=c2=1.5) against PID value

	n=30, c1=c2=1.5									
Kp	4.54	6.58	4.67	4.77	4.62	4.43	5.55	5.06	4.97	4.88
Ki	0.45	0.34	0.31	0.34	0.51	0.58	0.82	0.32	0.59	0.72
Kd	4.33	6.77	5.62	3.70	7.01	5.55	5.60	5.82	5.53	4.64

Table 4.11: Learning factor (c1=1, c2=2) against PID value

	n=30, c1=1, c2=2										
Kp	5.42	5.51	5.35	6.12	4.80	5.21	5.10	5.57	4.88	4.80	52.74
Ki	0.69	0.41	0.40	0.46	0.33	0.65	0.62	0.34	0.35	0.43	4.67
Kd	6.88	6.68	5.28	10.28	5.81	10.03	8.24	5.61	5.29	4.22	68.30

Table 4.12: Learning factor (c1=1.8, c2=2) against PID value

	n=30, c1=1.8, c2=2									
Kp	4.16	4.86	5.91	4.91	5.54	5.97	5.52	4.65	5.39	4.52
Ki	0.37	0.61	0.50	0.28	0.44	0.86	0.52	0.52	0.29	0.43
Kd	4.48	7.11	3.64	5.61	4.67	3.96	6.86	6.51	6.14	9.37

Table 4.13: Learning factor (c1=2.2, c2=2) against PID value

	n=30, c1=2.2, c2=2									
Kp	5.16	5.28	5.84	5.59	5.43	5.98	5.88	4.44	4.48	4.44
Ki	0.38	0.37	0.41	0.53	0.48	0.43	0.59	0.30	0.26	0.30
Kd	6.15	7.42	3.02	10.15	7.78	9.61	12.60	5.46	5.74	5.46

Table 4.14: Learning factor (c1=2, c2=2) against PID value

	n=30, c1=c2=2									
Kp	6.54	6.24	6.03	6.07	5.06	5.31	5.5	5.73	6.23	6.37
Ki	0.49	0.39	0.39	0.38	0.33	0.48	0.39	0.4	0.47	0.4
Kd	8.44	8.46	8.68	7.63	4.75	5.77	5.91	5.94	8.41	9.05

Based on the table 4.9-4.14, the average value of each combination is plotted into table 4.15.

Table 4.15: Average Value of Controllers Parameter for Different Learning Factor

	Learning Factor											
	c1	c2	c1	c2	c1	c2	c1	c2	c1	c2	c1	c2
	1	1	1.5	1.5	1	2	1.8	2	2.2	2	2	2
Kp	5.077		5.01		5.27		5.14		5.25		5.91	
Ki	11.78		10.05		11.29		10.67		12.97		14.34	
Kd	1.22		1.09		1.30		1.13		1.40		1.24	

After obtaining the average value of the each combination, the graphs are plotted for every simulated value as shown in table 4.15. The system response graph is shown in the figure 4.8-4.13.

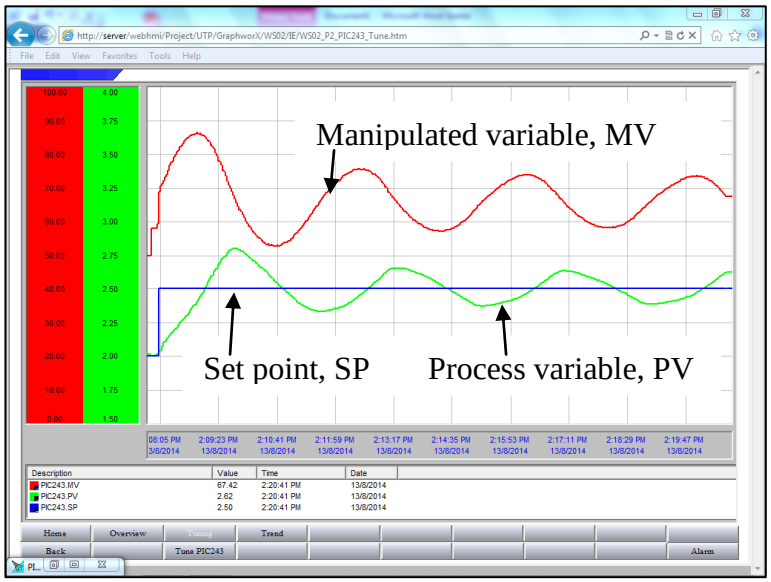

Figure 4.8: System Response for PID Controller (c1=1, c2=1)

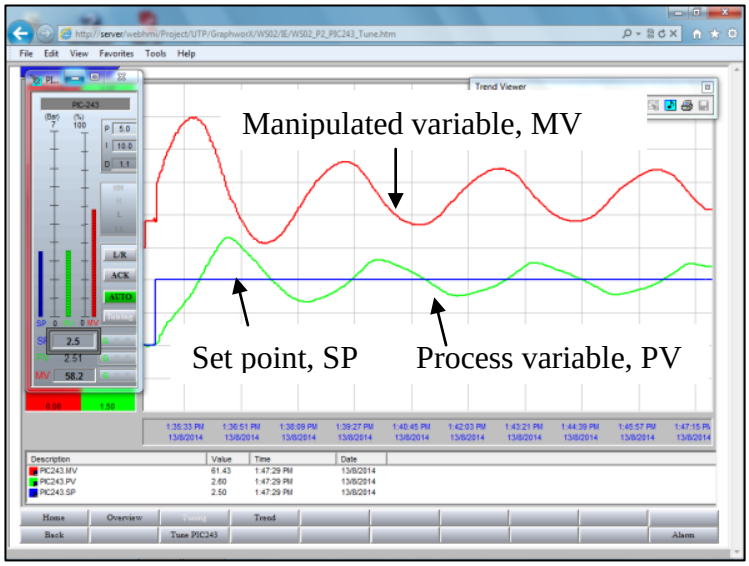

Figure 4.9: System Response for PID Controller (c1=c2=1.5)

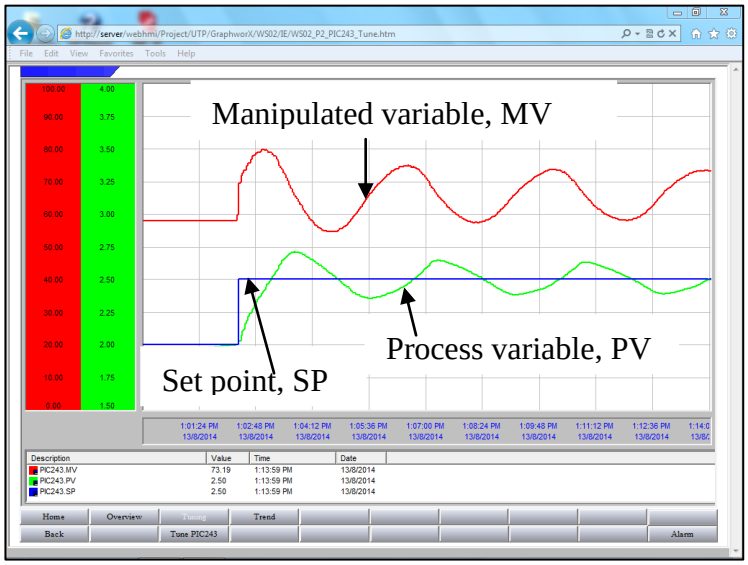

Figure 4.10: System Response for PID Controller (c1=1, c2=2)

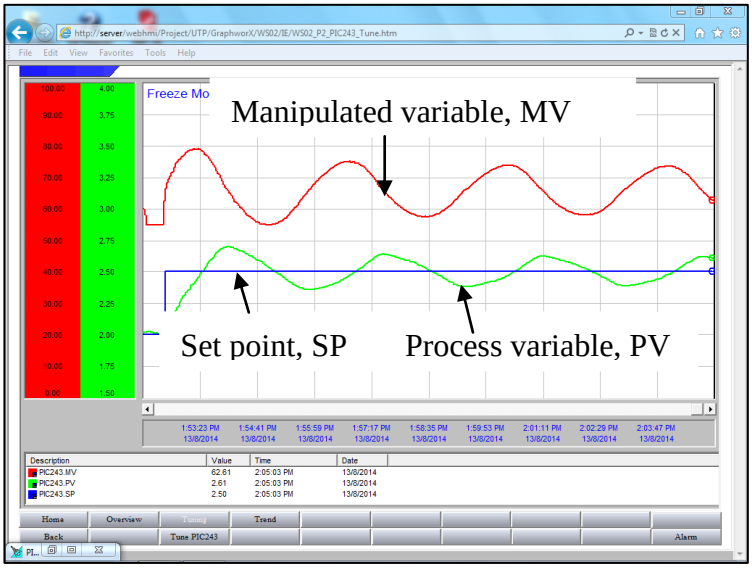

Figure 4.11: System Response for PID Controller (c1=1.8, c2=2)

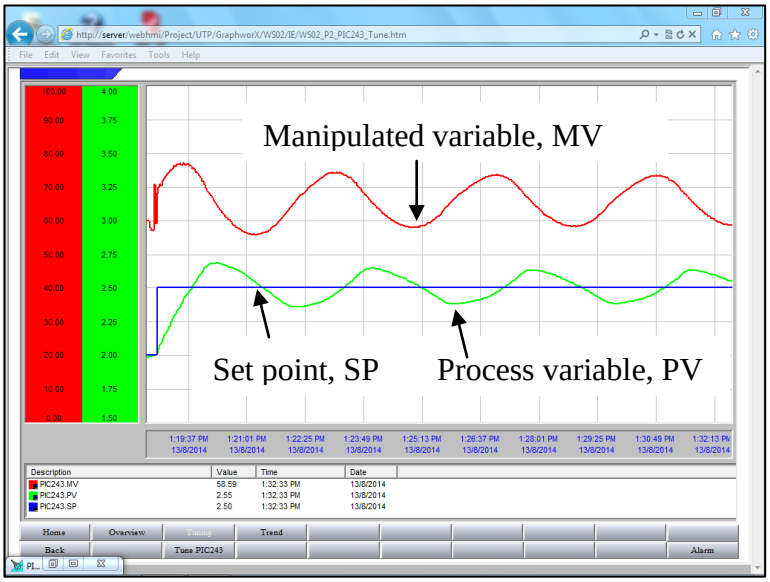

Figure 4.12: System Response for PID Controller (c1=2.2, c2=2)

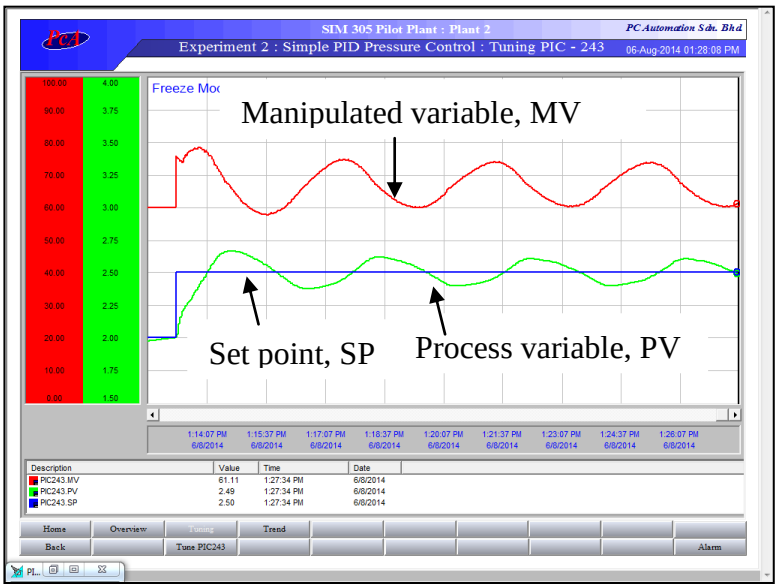

Figure 4.13: System Response for PID Controller (c1=2, c2=2)

By performing the step perturbation to the system, system response is obtained as shown in figure 4.8-4.13. The system performance of every set of experiment is tabulated in table 4.16.

Table 4.16: System Performance of Each Set of Experiment

	Learning Factor											
	c1	c2	c1	c2	c1	c2	c1	c2	c1	c2	c1	c2
	1	1	1.5	1.5	1	2	1.8	2	2.2	2	2	2
OS,%	60		64		42		40		36		34	
Tr,min	0.95		0.98		0.83		0.87		0.82		0.83	
Ts,min	9		9		8.67		8.5		8.9		8.75	

Based on the table 4.16, some of the combinations such as (c1=1.8 and c2=2, c1=2.2 and c2=2, and c1=2 and c2=2) did improve the system performance if compared to the conventional tuning method. However, since the optimization problem is giving random number for each simulation, average value may not be the best optimal solution however it is still good if compared to conventional tuning method. This indicates that individual set of P, I, and D values may perform better than average value yet the value still within the tolerance range ($\pm$5%) of average value. Hence, some of the values within the tolerance range are used to perform the test and finally one set value (optimal solution) is proved to be the best solution for the pressure plant and it is shown in section 4.3. To conclude, we can deduce that learning factors (c1 and c2) of 2 could provide the optimal solution for the controller parameters. [19] and [27] used value of 2 for their learning factor and it did provide the optimal solution for their application. Hence, learning factor of 2 is chose to be the best value in this project.

4.3 Comparison of Simulation and Experimental results

The finalize value of the PSO and CC method is shown in table 4.17.

Table 4.17: Controller Parameters for Cohen Coon and PSO method

	Cohen Coon	PSO (c1=c2=2, n=30)
Kp	5.91	6.23
Ki	7.21	13.26
Kd	1.08	1.35

By getting the final value of the both methods, system performance is tested in both simulation and real pressure plant. The system response of the optimal solution from PSO program is shown in figure 4.14.

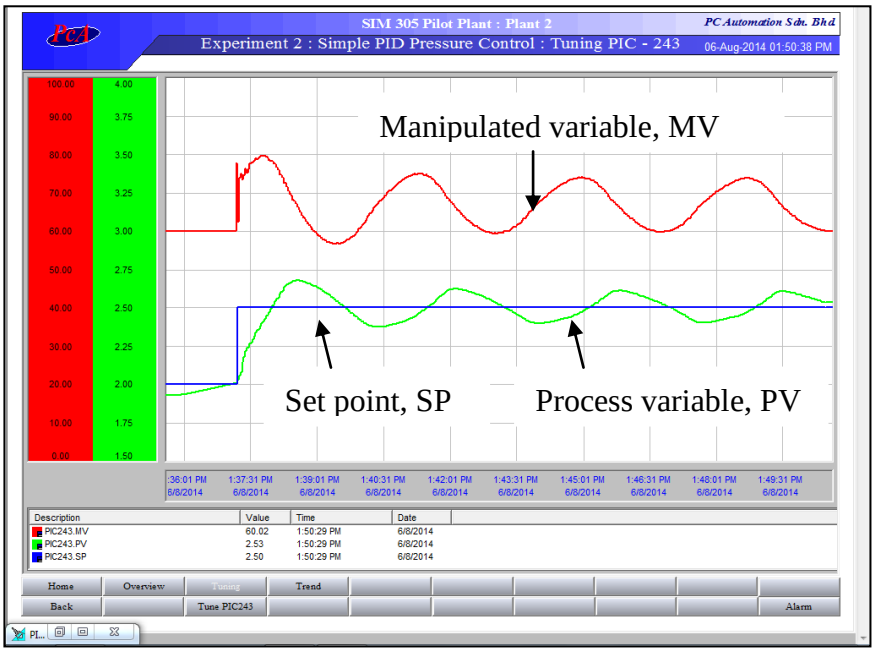

Figure 4.14: System Performance of best PSO-PID value

The best optimal value from individual set of experiment is **6.23, 13.26 and 1.35** for P, I and D value respectively. The response curve is shown in figure 4.14. Besides that, system response for CC method is shown in figure 4.15.

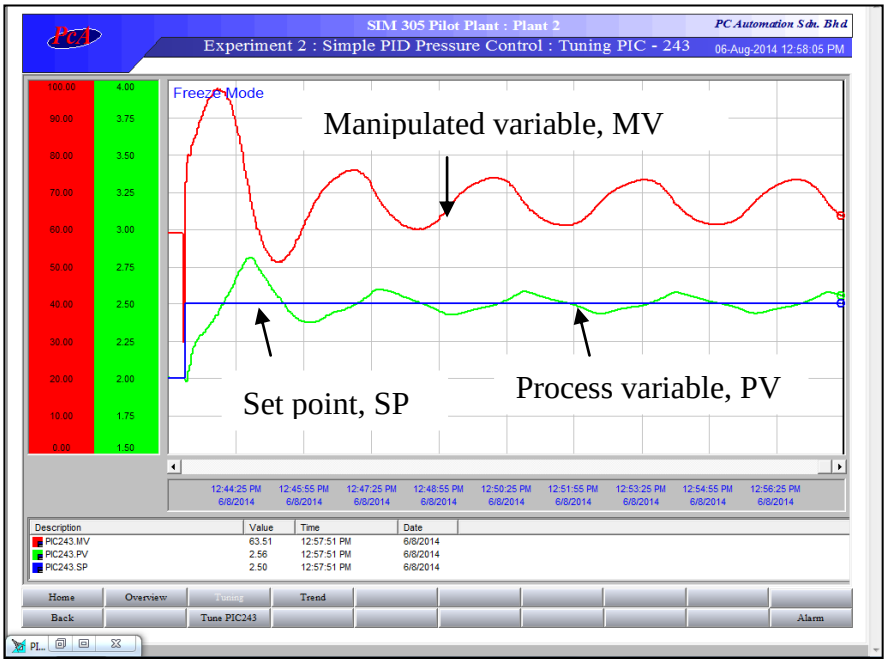

Figure 4.15: System response for Cohen Coon method

Apart from the experimental results, simulation is done to observe the system performance of both methods and it is used to compare the final results with the experimental work. The simulation graph for CC and PSO is shown in figure 4.16.

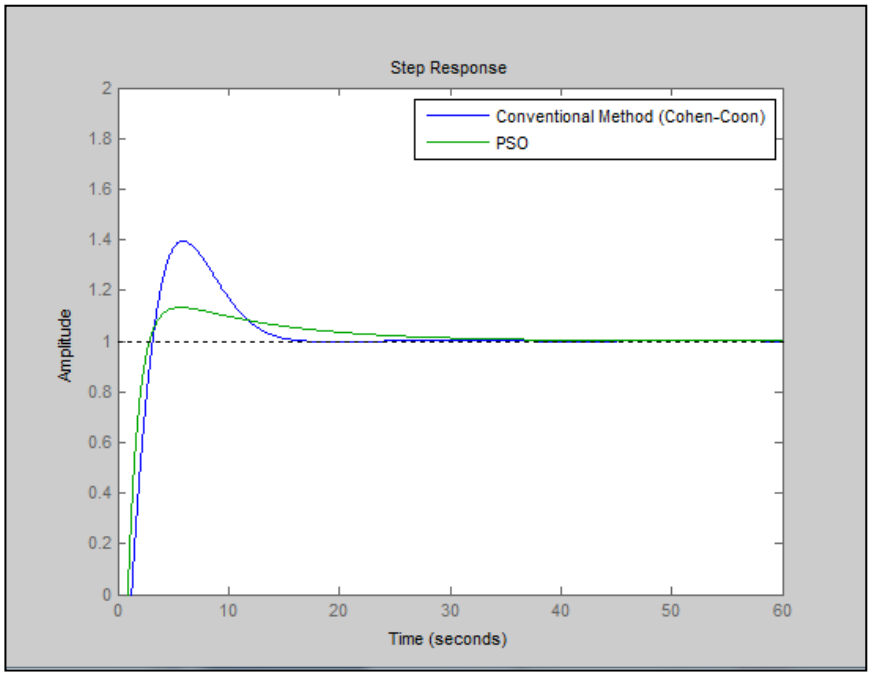

Figure 4.16: System Response of CC and PSO method

After getting the result of simulation (figure 4.16) and experimental work (figure 4.14 and 4.15) for CC and PSO method respectively, the system performance of these two methods will be further analyzed and comparison is done in table 4.18.

Table 4.18: Comparison between CC and PSO

		% OS	Tr, min	Ts, min
Simulation	**Cohen Coon**	39.565	0.0441	0.228
	PSO	13.266	0.0313	0.261
Experiment	**Cohen Coon**	62	0.88	7
	PSO	36	0.78	8.25
% Improved	**Simulation**	66.47	29.02	- 14.47
	Experiment	41.94	11.36	-17.86

Based on the table 4.18, we can conclude that PSO did improve the overall system performance in term of percentage overshoot and rise time. However, PSO did not improve the settling time if compared to CC method, because there is a trade-off between the overshoot and rise time with the settling time. The total percentage improved for overshoot and rise time is 41.94% and 11.36% respectively while the percentage improved for settling time is -17.86%. Hence, we can deduce that the system performance of pressure plant had been improved by using PSO algorithm even if the settling time has not been improved. Moreover, the settling time of the system may achieve positive figure if we manipulate the percentage overshoot and rise time. There is a trade-off for the system for which dominant parameters have to be followed. If the settling time is the minor criteria or the influential factor, the output from the experiment is suitable for the application. However, we can manipulate the rise time and percentage overshoot in order to utilize the PSO program in another type of plant.

Apart from that, the simulation and experimental results did show the same final performance in term of the improvement; however the final results of both methods did not show the same figures. This is because some approximation is used when extracting the data from the real plant, and therefore the simulation may yield different results from the real plant. To conclude, PSO can generate better controller parameters and exhibit better system performance if compared with conventional tuning methods.

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

The PSO program has been extensively studied and investigated in this project and results showed that PSO did improve the overall system performance of the pressure plant. Besides that, PSO method is tested with different combination of tuning parameters (c1, c2 and n) and it is successfully improved the system performance in term of rise time and percentage overshoot. However, PSO program did not improve the settling time for the pressure plant, but it can be improved if we manipulate the value of rise time and percentage overshoot. To conclude, there is a trade-off for percentage overshoot, rise time and settling time, and it is depending on the type of the real applications. Moreover, the computational time of the PSO program is optimal and it took 2.67 min to generate final values. Apart from that, only process equation is needed from the user in order to perform the simulation work and it is convenient to be used since not much knowledge of the plant is needed. Lastly, the program only has few parameters to be tuned and the range of the tuned parameters is small.

5.2 Recommendations

The simulation and experimental work that was carried out in this project produced good results. However, it could be improved further by carrying out additional work. Some of the suggestions are as the following:

- More combination of PSO parameters can be tested out in order to find the suitable tuning value for each type of application.
- Use another type of modeling method in order to get the most accurate process equation from the real plant.
- PSO program can be embedded inside distributed control system (DCS) in order to perform on-line PID tuning.
- PSO program can be implemented to test on other plants such as temperature plant and level plant.

5.3 Concluding Remarks

The project is carried out by simulation and experimental work to prove the reliability of the PSO method. This way of running experiment can prove the feasibility of specified methods and the evolutionary method can be further used in any other application if the process plants share the same characteristics.

REFERENCES

[1] B. Yang, G.-a. Liang, J.-h. Peng, S.-h. Guo, W. Li, S.-m. Zhang, *et al.*, "Self-adaptive PID controller of microwave drying rotary device tuning on-line by genetic algorithms," *Journal of Central South University,* vol. 20, pp. 2685-2692, 2013/10/01 2013.

[2] L. Fei, M. Zhenzhen, and L. Shaojun, "A new evolutionary algorithm based on Alopex and Harmony Search algorithm," in *Natural Computation (ICNC), 2010 Sixth International Conference on*, 2010, pp. 3719-3723.

[3] M. S. Khan, Y. A. Husnil, K. Yong Soo, and L. Moonyong, "Automated optimization of process plant using particle swarm optimization," in *Advanced Control of Industrial Processes (ADCONIP), 2011 International Symposium on*, 2011, pp. 615-620.

[4] K. Haridass, J. Valenzuela, A. D. Yucekaya, and T. McDonald, "Scheduling a log transport system using simulated annealing," *Information Sciences,* vol. 264, pp. 302-316, 4/20/ 2014.

[5] T. Liao, T. Stützle, M. A. Montes de Oca, and M. Dorigo, "A unified ant colony optimization algorithm for continuous optimization," *European Journal of Operational Research*, vol. 234, pp. 597-609, 5/1/ 2014.

[6] C. Knospe, "PID control," IEEE Control Syst. Mag., vol. 26, no. 1, 2006, pp. 30–31.

[7] G. Almeida, V. Rocha e Silva, E. Nepomuceno, and R. Yokoyama, "Application of Genetic Programming for Fine Tuning PID Controller Parameters Designed Through Ziegler-Nichols Technique," in *Advances in Natural Computation*. vol. 3612, L. Wang, K. Chen, and Y. Ong, Eds., ed: Springer Berlin Heidelberg, 2005, pp. 313-322.

[8] S. Majhi and L. Litz, "On-line tuning of PID controllers," in *American Control Conference, 2003. Proceedings of the 2003*, 2003, pp. 5003-5004 vol.6.

[9] D. Pavković, S. Polak, and D. Zorc, "PID controller auto-tuning based on process step response and damping optimum criterion," *ISA Transactions,* vol. 53, pp. 85-96, 1// 2014.

[10] R. Dittmar, S. Gill, H. Singh, and M. Darby, "Robust optimization-based multi-loop PID controller tuning: A new tool and its industrial application," *Control Engineering Practice*, vol. 20, pp. 355-370, 4// 2012.

[11] J. Sun and X. Liu, "The application prospects of intelligent PID controller in power plant process control," in *Modelling, Identification & Control (ICMIC), 2012 Proceedings of International Conference on*, 2012, pp. 399-403.

[12] A. Savran and G. Kahraman, "A fuzzy model based adaptive PID controller design for nonlinear and uncertain processes," *ISA Transactions*.

[13] J. Yang, B. Li, and H. Tao, "Research of Some Autopilot Controller Based on Neural Network PID," in *Advances in Wireless Networks and Information Systems*. vol. 72, Q. Luo, Ed., ed: Springer Berlin Heidelberg, 2010, pp. 161-170.

[14] A. Tao, Y. Jun-qi, L. Yan-Feng, and Z. Jiang, "Study on Neural Network Self-Tuning PID Control for Temperature of Active Solar House Heating System," in *Intelligent Systems and Applications (ISA), 2010 2nd International Workshop on*, 2010, pp. 1-4.

[15] D. Enzeng, G. Shuxiang, L. Xichuan, L. Xiaoqiong, and W. Yunliang, "A neural network-based self-tuning PID controller of an autonomous underwater

vehicle," in *Mechatronics and Automation (ICMA), 2012 International Conference on*, 2012, pp. 898-903.

[16] B. Nagaraj and N. Murugananth, "A comparative study of PID controller tuning using GA, EP, PSO and ACO," in *Communication Control and Computing Technologies (ICCCCT), 2010 IEEE International Conference on*, 2010, pp. 305-313.

[17] A. Soltoggio, "A Comparison of Genetic Programming and Genetic Algorithms in the Design of a Robust, Saturated Control System," in *Genetic and Evolutionary Computation – GECCO 2004*. vol. 3103, K. Deb, Ed., ed: Springer Berlin Heidelberg, 2004, pp. 174-185.

[18] K. Zhou and J. Qin, "PID controller parameters tuning of main steam temperature based on chaotic particle swarm optimization," in *Computer Science and Automation Engineering (CSAE), 2011 IEEE International Conference on*, 2011, pp. 647-650.

[19] M. S. Rahimian and K. Raahemifar, "Optimal PID controller design for AVR system using particle swarm optimization algorithm," in *Electrical and Computer Engineering (CCECE), 2011 24th Canadian Conference on*, 2011, pp. 000337-000340.

[20] S. D. Akyol and G. M. Bayhan, "A particle swarm optimization algorithm for maximizing production rate and workload smoothness," in *Nature and Biologically Inspired Computing (NaBIC), 2011 Third World Congress on*, 2011, pp. 44-49.

[21] J. Zhang and K. Zhang, "A Particle Swarm Optimization Approach for Optimal Design of PID Controller for Temperature Control in HVAC," in *Measuring Technology and Mechatronics Automation (ICMTMA), 2011 Third International Conference on*, 2011, pp. 230-233.

[22] W. Tao and Z. Shun-yi, "Active queue management based on particle swarm optimization PID algorithm," in *Communication Systems, 2008. ICCS 2008. 11th IEEE Singapore International Conference on*, 2008, pp. 523-526.

[23] L. Rodriguez-Garcia, S. Perez-Londono, and J. Mora-Florez, "Particle swarm optimization applied in power system measurement-based load modeling," in *Evolutionary Computation (CEC), 2013 IEEE Congress on*, 2013, pp. 2368-2375.

[24] M. Tokuda and T. Yamamoto, "A data-driven modeling method using particle swarm optimization," in *Modelling, Identification and Control (ICMIC), The 2010 International Conference on*, 2010, pp. 209-214.

[25] B. Bandyopadhyay and S. S. Lamba, "Time-domain Pade approximation and modal-Pade method for multivariable systems," *Circuits and Systems, IEEE Transactions on*, vol. 34, pp. 91-94, 1987.

[26] S. Yuhui and R. Eberhart, "A modified particle swarm optimizer," in *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, 1998, pp. 69-73.

[27] D. Ze, H. Pu, W. Dongfeng, and J. Songming, "Thermal Process System Identification Using Particle Swarm Optimization," in *Industrial Electronics, 2006 IEEE International Symposium on*, 2006, pp. 194-198.

APPENDICES

Appendix A

Table A-1: Timeline for FYP I

No.	Details/ Week	FYP 1													
		1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	Literature Review														
2	Plan the flow of the experiment														
3	Get the process reaction curve for each plant														
4	Obtain overall transfer function														
4	Run the performance test of each plant														
5	Proposal Defense														
7	Run the simulation for PID controller														
9	Compare the result for both plant and experiment														

Process

Key Milestone

Table A-2: Timeline for FYP II

No.	Details/ Week	FYP II													
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	Development of Algorithm															
2	Simulation / Experimental Implementation															
3	Comparative analysis															
4	Progress Report															
5	Pre-sedex															
6	Dissertartion															
7	Project Viva															

Process

Kev Milestone

Appendix B

Table B-1: Cohen-Coon Closed Loop Correlations table

Control Modes	Parameters
P only	$K_C = \left(\frac{1}{RK_p}\right)\left(1+\frac{R}{3}\right)$
P + I	$K_C = \left(\frac{1}{RK_p}\right)\left(\frac{9}{10}+\frac{R}{12}\right)$
	$T_I = \theta\frac{(30+3R)}{(9+20R)}$
P + I + D	$K_C = \left(\frac{1}{RK_p}\right)\left(\frac{4}{3}+\frac{R}{4}\right)$
	$T_I = \theta\frac{(32+6R)}{(13+8R)}$
	$T_D = \theta\frac{4}{(11+2R)}$
P + D	$K_C = \left(\frac{1}{RK_p}\right)\left(\frac{5}{4}+\frac{R}{6}\right)$
	$T_I = \theta\frac{(6-2R)}{(22+3R)}$