TUGAS AKHIR – EE184801

PERANCANGAN KONTROLER PID DENGAN METODE *PARTICLE SWARM OPTIMIZATION* UNTUK SISTEM PENGATURAN CASCADE LEVEL DAN FLOW PADA PLANT PCT-100

KHOIRUS FAUZI RAHMADHANI

NRP 07111840000029

Dosen Pembimbing

Eka Iskandar, ST., MT

NIP 198005282008121001

Program Studi Teknik Elektro

Departemen Teknik Elektro

Fakultas Teknologi Elektro dan Informatika Cerdas

Institut Teknologi Sepuluh Nopember

Surabaya

2022

--Halaman ini sengaja dikosongkan--

TUGAS AKHIR – EE184801

PERANCANGAN KONTROLER PID DENGAN METODE PARTICLE SWARM OPTIMIZATION UNTUK SISTEM PENGATURAN CASCADE LEVEL DAN FLOW PADA PLANT PCT-100

KHOIRUS FAUZI RAHMADHANI

NRP 07111840000029

Dosen Pembimbing

Eka Iskandar, ST., MT

NIP 198005282008121001

Program Studi Teknik Elektro

Departemen Teknik Elektro

Fakultas Teknologi Elektro dan Informatika Cerdas

Institut Teknologi Sepuluh Nopember

Surabaya

2022

--Halaman ini sengaja dikosongkan--

FINAL PROJECT – EE184801

PID CONTROLLER DESIGN WITH PARTICLE SWARM OPTIMIZATION METHOD FOR CASCADE LEVEL AND FLOW CONTROL SYSTEMS AT PLANT PCT-100

KHOIRUS FAUZI RAHMADHANI

NRP 07111840000029

Advisor

Eka Iskandar, ST., MT

NIP 198005282008121001

Study Program Electrical Engineering

Department of Electrical Engineering

Faculty of Intelligent Electrical and Information Technology

Institut Teknologi Sepuluh Nopember

Surabaya

2022

--Halaman ini sengaja dikosongkan--

LEMBAR PENGESAHAN

PERANCANGAN KONTROLER PID DENGAN METODE PARTICLE SWARM OPTIMIZATION UNTUK SISTEM PENGATURAN CASCADE LEVEL DAN FLOW PADA PLANT PCT-100

TUGAS AKHIR

Diajukan untuk memenuhi salah satu syarat

memperoleh gelar Sarjana Teknik pada

Program Studi S-1 Teknik Sistem Pengaturan

Departemen Teknik Elektro

Fakultas Teknologi Elektro dan Informatika Cerdas

Institut Teknologi Sepuluh Nopember

Oleh: KHOIRUS FAUZI RAHMADHANI

NRP. 07111840000029

Disetujui oleh Tim Penguji Tugas Akhir:

1. Eka Iskandar, ST., MT — Pembimbing
2. Mohamad Abdul Hady,S.T.,M.T. — Ko-pembimbing
3. Prof. Dr. Ir. Achmad Jazidie, M.Eng. — Penguji
4. Dr. Ir. Ari Santoso, DEA. — Penguji
5. Dr. Trihastuti Agustinah, ST., MT. — Penguji
6. Ir. Ali Fatoni, M.T. — Penguji

SURABAYA

Desember, 2022

--Halaman ini sengaja dikosongkan--

VALIDITY SHEER

PID CONTROLLER DESIGN WITH PARTICLE SWARM OPTIMIZATION METHOD FOR CASCADE LEVEL AND FLOW CONTROL SYSTEMS AT PLANT PCT-100TUGAS AKHIR

Submitted to fulfill one of the requirements

obtained a Bachelor of Engineering degree in

Control System Engineering Undergraduate Program

Electrical Engineering Departmen

Faculty of Intelligent Electrical and Informatics Technology

Sepuluh Nopember Institute of Technology

Oleh: KHOIRUS FAUZI RAHMADHANI

NRP. 07111840000029

Disetujui oleh Tim Penguji Tugas Akhir:

1. Eka Iskandar, ST., MT — Advisor 
2. Mohamad Abdul Hady,S.T.,M.T. — Co-Advisor 
3. Prof. Dr. Ir. Achmad Jazidie, M.Eng. — Examiner 
4. Dr. Ir. Ari Santoso, DEA. — Examiner 
5. Dr. Trihastuti Agustinah, ST., MT. — Examiner 
6. Ir. Ali Fatoni, M.T. — Examiner 

SURABAYA

Desember, 2022

--Halaman ini sengaja dikosongkan--

PERNYATAAN ORISINALITAS

Yang bertanda tangan di bawah ini:

Nama mahasiswa / NRP	:	Khoirus Fauzi Rahmadhani / 07111840000029
Program studi	:	Teknik Sistem Pengaturan
Dosen Pembimbing / NIP	:	Eka Iskandar, ST., MT / 198005282008121001

dengan ini menyatakan bahwa Tugas Akhir dengan judul "PERANCANGAN KONTROLER PID DENGAN METODE PARTICLE SWARM OPTIMIZATION UNTUK SISTEM PENGATURAN CASCADE LEVEL DAN FLOW PADA PLANT PCT-100" adalah hasil karya sendiri, bersifat orisinal, dan ditulis dengan mengikuti kaidah penulisan ilmiah.

Bilamana di kemudian hari ditemukan ketidaksesuaian dengan pernyataan ini, maka saya bersedia menerima sanksi sesuai dengan ketentuan yang berlaku di Institut Teknologi Sepuluh Nopember.

Surabaya, 15 Desember 2022

Mengetahui
Dosen Pembimbing

Eka Iskandar, ST., MT
NIP. 198005282008121001

Mahasiswa

Khoirus Fauzi Rahmadhani
NRP. 07111840000029

--Halaman ini sengaja dikosongkan--

ABSTRAK

PERANCANGAN KONTROLER PID DENGAN METODE PARTICLE SWARM OPTIMIZATION UNTUK SISTEM PENGATURAN CASCADE LEVEL DAN FLOW PADA PLANT PCT-100

Nama Mahasiswa / NRP : **Khoirus Fauzi Rahmadhani / 07111840000029**
Departemen : **Teknik Elektro FTEIC - ITS**
Dosen Pembimbing : **Eka Iskandar, ST., MT**

Abstrak

Dalam sistem kontrol proses industri skala besar, variabel yang sering dikontrol adalah level cairan, laju aliran, suhu, dan tekanan. Lebih dari 90% aplikasi kontrol industri menggunakan kontroler PID untuk pengendalian otomatis. Namun terdapat controller PID masih belum optimal terutama adanya disturbance berupa ketidakstabilan sehingga output juga berubah. Kelemahan dari kontroler konvensional tidak dapat diabaikan karena sebagian besar industri berfokus pada akurasi, margin keuntungan, dan waktu operasi. Tujuan dari tugas akhir ini adalah merancang algoritma tuning PID dengan metode *particle swarm optimization* pada sistem pengaturan cascade level dan *flow*. Pada *Inner Loop* diperoleh besaran Kp, Ki, dan Kd yang dituning dengan PSO sebesar 20.8441, 10.0045, dan 5.8969 dengan indeks performansi ITAE sebesar 0.6534 dan ISE 0.2022. Memiliki Karakteristik sistem *max overshoot* 0.567%, *Rise time* 0.1544 detik dan *Settling time* 0.2473 detik. Pada Outer Loop diperoleh besaran Kp, Ki, dan Kd yang dituning dengan PSO sebesar 68.4639, 1.8237, dan 10.2855 memiliki indeks performansi error ITAE sebesar 1.51 dan ISE sebesar 0.5841. Sistem tersebut mengalami *max overshoot* sebesar 1.4001%, *Rise Time* 2.0383 detik, dan *Settling time* 3.2569 detik. Pada implementasi nyata kontroler PID dituning dengan PSO mampu menanggulangi gangguan yang ditimbulkan oleh bukaan *valve*

Kata kunci: ***Level, Flow, PID, Particle Swarm Optimization, Cascade***

--Halaman ini sengaja dikosongkan--

ABSTRACT

PID CONTROLLER DESIGN WITH PARTICLE SWARM OPTIMIZATION METHOD FOR CASCADE LEVEL AND FLOW CONTROL SYSTEMS AT PLANT PCT-100

Student Name / NRP	**:**	**Khoirus Fauzi Rahmadhani / 07111840000029**
Department	**:**	**Teknik Elektro FTEIC - ITS**
Advisor	**:**	**Eka Iskandar, ST., MT**

Abstract

In large-scale industrial process control systems, the variables that are often controlled are liquid level, flow rate, temperature, and pressure. More than 90% of industrial control applications use PID controllers for automated control. However, there is a PID controller that is still not optimal, especially if there is a disturbance in the form of instability so that the output also changes. The disadvantages of conventional controllers cannot be ignored as most of the industry focuses on accuracy, profit margins and uptime. The aim of this final project is to design a PID tuning algorithm using the particle swarm optimization method for a cascade level and flow control system. In the Inner Loop, the Kp, Ki, and Kd values were adjusted with a PSO of 20.8441, 10.0045, and 5.8969 with an ITAE performance index of 0.6534 and an ISE of 0.2022. Has the characteristics of a max overshoot system of 0.567%, Rise time of 0.1544 seconds and Settling time of 0.2473 seconds. In the Outer Loop, the Kp, Ki, and Kd values were adjusted with PSO of 68.4639, 1.8237, and 10.2855 with an ITAE error performance index of 1.51 and an ISE of 0.5841. The system experienced a max overshoot of 1.4001%, Rise Time of 2.0383 seconds, and Settling time of 3.2569 seconds. In the real implementation of the PID controller tuned with PSO is able to overcome the disturbance caused by the valve opening

Keywords:. ***Level, Flow, PID, Particle Swarm Optimization, Cascade***

--Halaman ini sengaja dikosongkan--

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

--Halaman ini sengaja dikosongkan--

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Dalam sistem kontrol proses industri skala besar, variabel yang sering dikontrol adalah level cairan, laju aliran, suhu, tekanan dan sebagainya, yang sebagian besar merupakan fungsi non linier dari input kontrol. Namun, untuk menggunakan teknik kontrol linear secara umum proses non linier perlu didekati untuk mendapatkan model linierisasi, salah satunya adalah fungsi alih. (Wen et al., 2012). Masalah utama dalam industri pengolahan adalah pengaturan level dan aliran cairan. Saat ini, sering terjadi masalah dengan alat kontrol pada tangki reaktor selama proses pengaturan level dan aliran. Contohnya, kerusakan pada instrumen pengukur dan gangguan tak terduga. Hal ini tentu saja mempengaruhi operasi sistem kontrol level dan aliran cairan. Untuk mencapai kinerja yang baik dalam sistem pengaturan level dan aliran, diperlukan pengontrol yang mampu menangani masalah yang muncul dalam sistem, yaitu kontrolir PID. (Pritandi, 2016). Kontrolir proporsional-integral-derivatif (PID), yang merupakan algoritma kontrol *feedback*, telah banyak diterapkan dalam kontrol industri karena kinerja yang robust dalam berbagai kondisi operasi dan kemudahan fungsional. Lebih dari 90% aplikasi kontrol industri didasarkan pada mekanisme kontrol PID, seperti sistem robotika dan pengendalian otomatis. (Xiang et al., 2019)

Namun dalam sistem pengaturan yang terdapat kontrolir PID masih belum optimal terutama adanya *disturbance* atau gangguan berupa ketidakstabilan laju aliran pada variabel yang dimanipulasi sehingga *output* juga berubah. Kontrol kaskade adalah salah satu metode paling sukses untuk meningkatkan kinerja kontrol *loop* tunggal dengan menstabilkan, mengurangi osilasi dan kesalahan integral dari gangguan pada variabel yang dimanipulasi (Tridianto et al., 2017). Sistem pengaturan kaskade pada umumnya masih menggunakan kontrolir PID konvensional pada industri proses terutama untuk mengatur level dan aliran. Kelemahan dari pengontrol konvensional tidak dapat diabaikan karena sebagian besar industri berfokus pada akurasi, margin keuntungan, dan waktu operasi. (Guong et al., 2018). Terinspirasi oleh perilaku sosial kawanan burung dan kelompok ikan, PSO diusulkan sebagai metaheuristik berbasis populasi teknik optimasi untuk solusi dari berbagai fungsi nonlinier dan masalah optimasi. Sudah beberapa tahun, PSO terbukti cukup populer di berbagai bidang terutama karena sederhana, *Robustness*, dan *high dimensionality* (Pano & Ouyang, 2014). Terdapat penelitian yang membahas perihal *Particle Swarm Optimization* diantaranya adalah PSO digunakan untuk desain yang optimal pengontrol PID untuk gerakan DC brushless linier(Ibrahim et al., 2014),dan Pengendalian PID multi-objektif Berdasarkan PSO Dengan aplikasi ke steam Pengendalian suhu di Boiler (Kumar et al., 2010).

1.2 Rumusan Masalah

Pada industri proses sering dijumpai sistem pengaturan kaskade di mana salah satunya adalah Sistem Pengaturan Kaskade Level dan *Flow* atau aliran. Sistem pengaturan kaskade terdiri dari *inner loop* dan *outer loop* dimana masing-masing terdapat kontroler PID, karena Sebagian besar industri sekarang berfokus pada akurasi, margin keuntungan dan waktu operasi maka Tuning PID konvensional belum optimal dalam menyelesaikan masalah tersebut. Oleh karena itu diperlukan cara tuning PID dengan Algoritma PSO untuk mendapatkan hasil yang optimal. Rumusan masalah pada Tugas Akhir ini adalah bagaimana cara menala kontrolir PID

dengan algoritma *Particle Swarm Optimization* dan Bagaimana respons sistem setelah *detuning* dengan Algoritma PSO tersebut.

1.3 Batasan Masalah

Dalam melaksanakan Tugas Akhir ini, terdapat berbagai batasan masalah yang perlu diperhatikan, diantaranya sebagai berikut:

1. *Plant* yang digunakan pada Tugas Akhir ini adalah PCT 100 dengan sistem yang digunakan adalah Sistem Pengaturan Level dan Sistem Pengaturan Flow yang selanjutnya dimodelkan persamaan model linear fungsi alih.
2. Implementasi dilakukan pada plant PCT 100 menggunakan ADAM serta bahasa pemrograman python, simulasi dan pengujian dengan menggunakan simulator MATLAB dan SIMULINK.
3. Parameter yang digunakan pada *Particle Swarm Optimization* dibuat tetap yakni c1=1.5, c2=1, iterasi maksimal =100, dan batas search space nilai 0 sampai dengan 100.

1.4 Tujuan

Tujuan dari tugas akhir ini adalah:

1. Merancang algoritma Tuning PID dengan metode *particle swarm optimization* sehingga di dapat nilai optimal dari masing-masing controller Proportional, Integral dan Derivatif.
2. Mengetahui Respons system setelah dituning dengan PID kontrolir menggunakan metode *particle swarm optimization* serta dianalisis karakteristik respons sistem

1.5 Manfaat

Manfaat penelitian ini diharapkan Tuning PID berbasi *Particle Swarm Optimization* pada sistem pengaturan kaskade level dan *flow* dapat menjadi solusi optimal pada permasalahan Tuning PID konvensional dan dapat diaplikasikan dalam industri kontrol proses, serta sebagai referensi pada penelitian selanjutnya.

BAB 2 TINJAUAN PUSTAKA

2.1 Plant PCT 100

Process Control Technology atau biasa disebut *PCT-100* adalah sebuah *plant* mewakili suatu sistem yang biasanya ditemukan di industri proses. Pada plant ini terdiri dari Modul Proses, Konsol Kontrol dengan Catu Daya, dan perangkat lunak kontrol interaktif.

Gambar 2-1 Plant PCT 100

Konsol Kontrol merupakan representasi dari modul proses yang terletak pada bagian depan dan terdapat *fault switch* dan titik uji. Konsol Kontrol terdapat koneksi untuk memungkinkan kontrol pada *Personal Computer* dengan menggunakan kabel (USB) dan PLC atau pengontrol PID. Pada plant PCT 100 terdapat cairan yang terkandung dalam *sump tank* dipompa di sekitar sistem ke dalam tangki proses di mana level, temperatur dan tekanan dapat dikontrol. Dalam tangki proses, level diukur menggunakan sensor magnetostriktif 0-10 Volt; tekanan diukur menggunakan sensor *gauge* 0-5 bar; dan PT1000 digunakan untuk mengukur suhu di tangki bah dan tangki proses. Sensor laju aliran turbin digunakan untuk mengukur laju aliran dalam sistem. Katup pengalih dapat digunakan untuk mengarahkan cairan melalui proses pendinginan udara untuk mendinginkan cairan dalam sistem. Dua katup proporsional digunakan untuk mengontrol aliran masuk dan keluar dari tangki proses, katup jarum yang dapat disesuaikan secara manual digunakan untuk menambah gangguan pada sistem dan katup pelepas tekanan yang dipasang untuk keselamatan.

Data yang dikumpulkan dapat ditampilkan pada LCD yang dipasang pada Modul Proses dan melalui perangkat lunak data dapat dipantau, disimpan maupun dicetak. Perangkat lunak pada modul ini digunakan sebagai modul untuk belajar kontroler PID dan menyediakan fitur *monitoring* serta akuisisi data dengan beberapa fitur yang sedang tren.

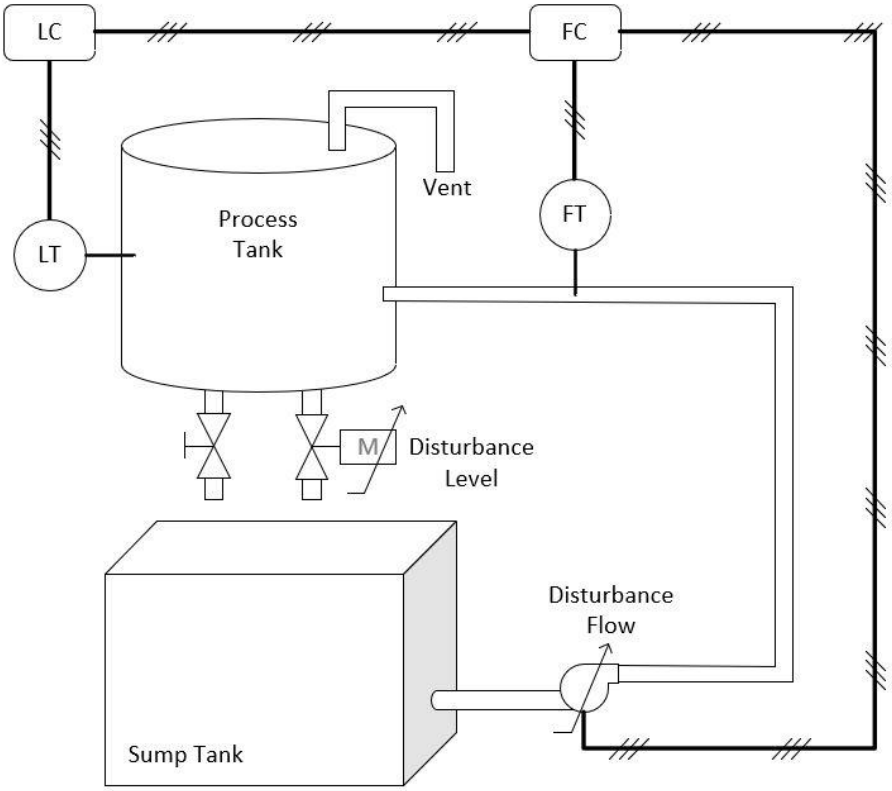
Pada PCT-100 ini terdapat modul untuk pengaturan dan eksekusi siklus proses, hingga optimalisasi pengontrol PID dengan memperkirakan parameter P, PI, dan PID menggunakan kurva reaksi proses *Ziegler Nichols* atau metode siklus akhir.(Bytronic Limited, n.d.)

Tabel 2-1 Spesifikasi PCT 100

Specification	
Process Module	
Sump tank volume	8 L
Process tank volume	4.5 L
Heating element power	48v dc 400W
Control elements	2 x Proportional valves
	1 x Manually adjustable needle valve
Pump	24v 0.5amp 6 Litres per minute with 1.5 bar safety cut-out
Cooling system	Radiator with forced air cooling
Maximum flow rate	3.5 Litres per minute
Level transducer	Magnetostrictive position sensor 0 – 10v
Pressure transducer	Gage 0 – 5 bar (0 – 2 bar is used as the range for laboratory exercises)
Number of LCD displays	5 x level, process tank and sump tank temperature, pressure and flow rate
Indicators	4 x heater on, drain valve open, diverter solenoid active and cooler active
Control Console	
Number of fault switches	6 x Illuminated
Test Points	5 x heater, level, temp; sump tank and process tank, flow rate
Indicators	6 x heater, drain , flow and diverter solenoid, pump and cooler active
	Graphical representation of Process Module
Connections	1 x 15 way ‘D ‘connector and 1 x 25 way ‘D’ connector
	1 x power connector for heater
	1 x USB connection
	1 x Mains power connector
Actuator voltage	24Vdc
Power Supply voltage	100-250V AC @ 50/60Hz.
Approximate weight	42Kg
Approximate dimensions	100x76x40cm
Software	Windows based software with SCADA type interface. PID and Manual Control of Flow, Level. Temperature, Pressure, Open Loop and Batch. Data logging with printing and saving features

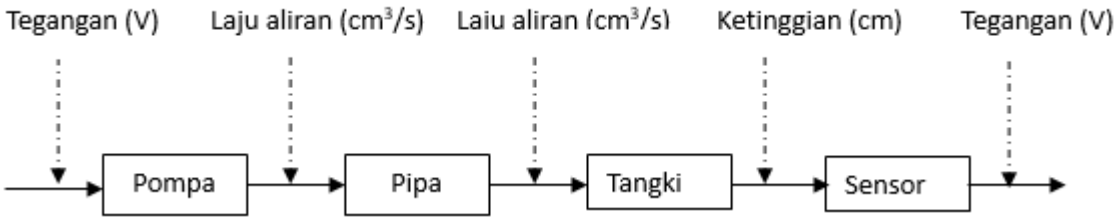
2.2 Pemodelan *Plant* PCT-100

Pada PCT-100 *plant* yang digunakan pada Gambar 2.1 menggambarkan proses pada sistem pengaturan level atau ketinggian suatu fluida. Adapun proses pengaturan level berupa pengisian tangki reaksi yang dilakukan oleh pompa sedangkan pembuangan fluida dilakukan oleh kontrol *valve* dan variabel *valve* pada PCT-100, kedua proses tersebut jika digabungkan maka akan mengontrol luaran pada sistem yakni sistem pengaturan level.



Gambar 2-2 Diagram P&ID Cascade Level dan Flow

Proses pengaturan level digambarkan dengan menggunakan diagram alir pada Gambar (Ket). Pemodelan ini dimulai dari pemodelan pompa, pemodelan pipa, pemodelan tangka, dan pemodelan sensor. Adapun diagram alirnya adalah:



Gambar 2-3 Diagram Proses Pemodelan Level

Parameter yang diperlukan dalam pemodelan *plant* terdapat pada pada Tabel 2.2 beserta keterangan dan satuan yang digunakan nilainya diambil dari *data sheet* dan pengukuran langsung pada *plant* PCT-100.

Tabel 2-2 Tabel Konversi Satuan

Simbol	Keterangan	Satuan
Q_{in}	Debit air yang masuk ke dalam tangki	cm^3 / s
Q_{out}	Debit ari yang keluar dari tangka	cm^3 / s
Q	Debit air dalam tangki	cm^3 / s
H	Ketinggian air dalam tangki	*Cm*
A	Luas alas tangki	cm^2
a	Diameter pipa	*cm*
h_0	Ketinggian air awal pada saat titik operasi	*cm*
Q_0	Laju aliran keluar saat kondisi di titik operasi	cm^3 / s
R	Resistansi *control valve*	*s / cm*
C	Kapaistansi tangki	cm^2
L	Panjang pipa	*cm*
θ	Waktu tunda	Detik

Untuk memperoleh model *plant* PCT 100 terutama sistem pengaturan level maka dalam prosesnya harus digunakan persamaan atau formula untuk memodelkan Pompa, Pipa, Tangki dan Sensor adapun formulanya adalah sebagai berikut.

2.2.1 Pemodelan Pompa

Pada pemodelan pompa ini bertujuan untuk mendapatkan nilai penguatan dari pompa yang dilambangkan dengan K1 serta dirumuskan melalui persamaan 2.1

$$K1 = \frac{\text{Rentang debit air yang keluar dari pompa } (\frac{cm^3}{s})}{\text{Rentang tegangan masukan (V)}} \quad (2.1)$$

2.2.2 Pemodelan Pipa

Tujuan dalam pemodelan pipa ini adalah untuk mengetahui besarnya waktu tunda yang diakibatkan oleh panjang pipa. Hubungan waktu tunda dengan panjang pipa dapat dirumuskan dalam persamaan berikut

$$\theta = \frac{Volume\ Pipa\ (cm^3)}{Debit\ air\ yang\ masuk\ pada\ pipa\ (^{cm^3}/_s} \quad (2.2)$$

Persamaan di atas dapat dibawa ke sistem pengaturan level dimana hubungan antara debit air yang dikeluarkan oleh pompa dan debit air yang masuk ke tangki adalah sebagai berikut

$$\frac{Q_{inT}}{Q_{outP}} = e^{-\theta s} \quad (2.3)$$

2.2.3 Pemodelan Tanki

Pada sebuah tangki reaksi sistem pengaturan level proses yang terjadi pada umumnya terdiri dari pengumpanan dan pembuangan air. Proses pengumpanan bertujuan untuk menambah volume air sedangkan proses pembuangan akan mengurangi volume air pada tangka reaksi. Parameter utama yang digunakan dalam sistem reservoir air berupa kapasitansi tanki dan resistansi jalur pembuangan. Kapasitansi reservoir didefinisikan sebagai besar perubahan

volume cairan yang diperlukan untuk membuat perubahan ketinggian sebesar satu satuan(Ogata, 2010). Kapasitansi suatu tangki atau reservoir sama dengan luas permukaan penampang tangki atau reservoir tersebut. Jika luas penampang konstan maka nilai kapasitansi juga konstan

$$C = \frac{perubahan\ volume\ cairan(cm^3)}{perubahan\ ketinggia\ cairan\ (cm^3)} \quad (2.4)$$

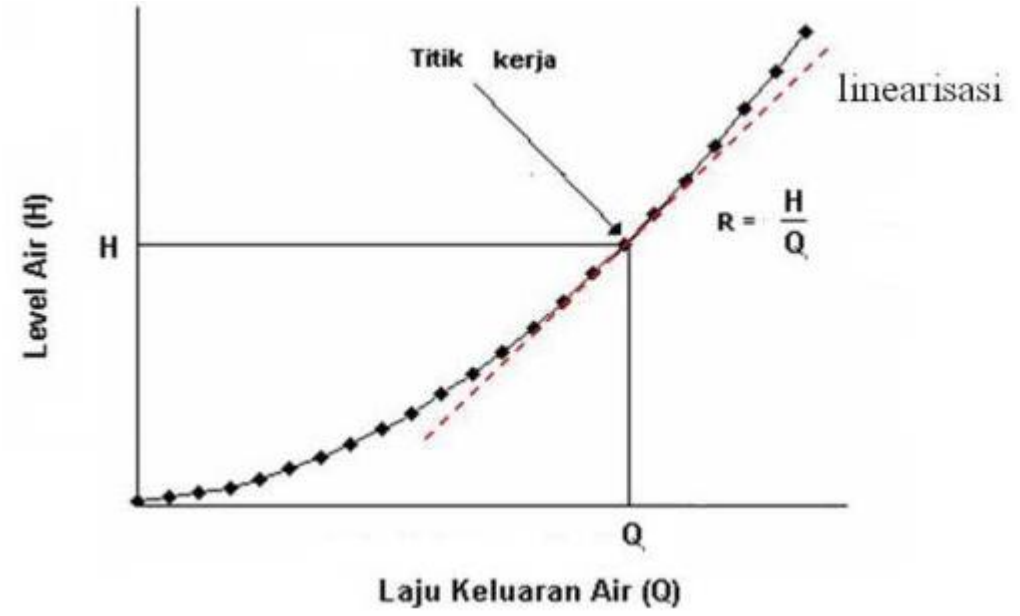
$$C = Luas\ penampang\ tanki = \frac{\pi d^2}{4} \quad (2.5)$$

Berdasarkan asumsi tersebut maka persamaan kesetimbangan massa adalah sebagai berikut:

$$Q_i - Q_0 = C\frac{dh}{dt}$$

$$Q_i - A_0\sqrt{2gh} = C\frac{dh}{dt}$$

Dengan demikian sistem tangki air sebenarnya merupakan sistem yang tidak linier. Pada umumnya linierisasi dapat dilakukan untuk sistem pengendalian level air dengan daerah kerja ketinggian level yang sudah ditetapkan, sehingga linierisasi nilai resistansi dapat dilakukan (Kawasan et al., 2011). Metode liniarisasi ditunjukkan pada gambar 2.4 di bawah ini



Gambar 2-4 Metode Linearisasi

Untuk melakukan linearisasi berdasarkan grafik di atas maka diperoleh persamaan yang dinyatakan sebagai berikut

$$R = \frac{perubahan\ tinggi\ muka\ (cm)}{perubahan\ laju\ aliran\ \left(cm^3/_s\right)} \quad (2.6)$$

$$R = \frac{H_{titik\ kerja}}{Q_{titik\ kerja}} \quad (2.7)$$

$$Q_0 = \frac{H}{R} \quad (2.8)$$

Dengan demikian rumus kesetimbangan massa pada persamaan 2.9 menjadi :

$$Qi - \frac{H}{R} = C\ \frac{dh}{dt} \quad (2.9)$$

$$QiR - H = RC\frac{dh}{dt} \quad (2.10)$$

Dari persamaan di atas dapat ditransformasikan Laplace sehingga diperoleh persamaan berikut:

$$QiR(s) - H(s) = RCsH(s)$$

$$QiR(s) = RCsH(s) + H(s)$$

$$H(s)[RCs + 1] = QiR(s) \quad (2.11)$$

Dan diperoleh fungsi alih sistem yakni:

$$\frac{H(s)}{Q_i} = \frac{R}{RCs + 1} = \frac{R}{\tau s + 1} \quad (2.12)$$

Dengan $\tau = RC$ adalah konstanta waktu sistem tanki, *Qi* adalah debit masukan air (cm^3/s), *Qo* adalah debit luaran air (cm^3/s), H adalah level air (cm), dan g adalah percepatan gravitasi sedangkan A_0 adalah luas penampang pipa keluaran.

2.2.4 Pemodelan Sensor

Pemodelan sensor bertujuan untuk mendapatkan nilai penguatan yang dilambangkan dengan K2. Nilai K2 dirumuskan dengan rentang kerja dari pompa terhadap masukan berupa tegangan seperti Persamaan 2.13 dibawah ini

$$K2 = \frac{Rentang\ tegangan\ keluaran\ (V)}{Rentang\ ketinggian\ air\ dalam\ tangki\ (cm)} \quad (2.13)$$

Dari uraian persamaan pemodelan di atas didapat persamaan *plant* sistem pengaturan level seperti berikut:

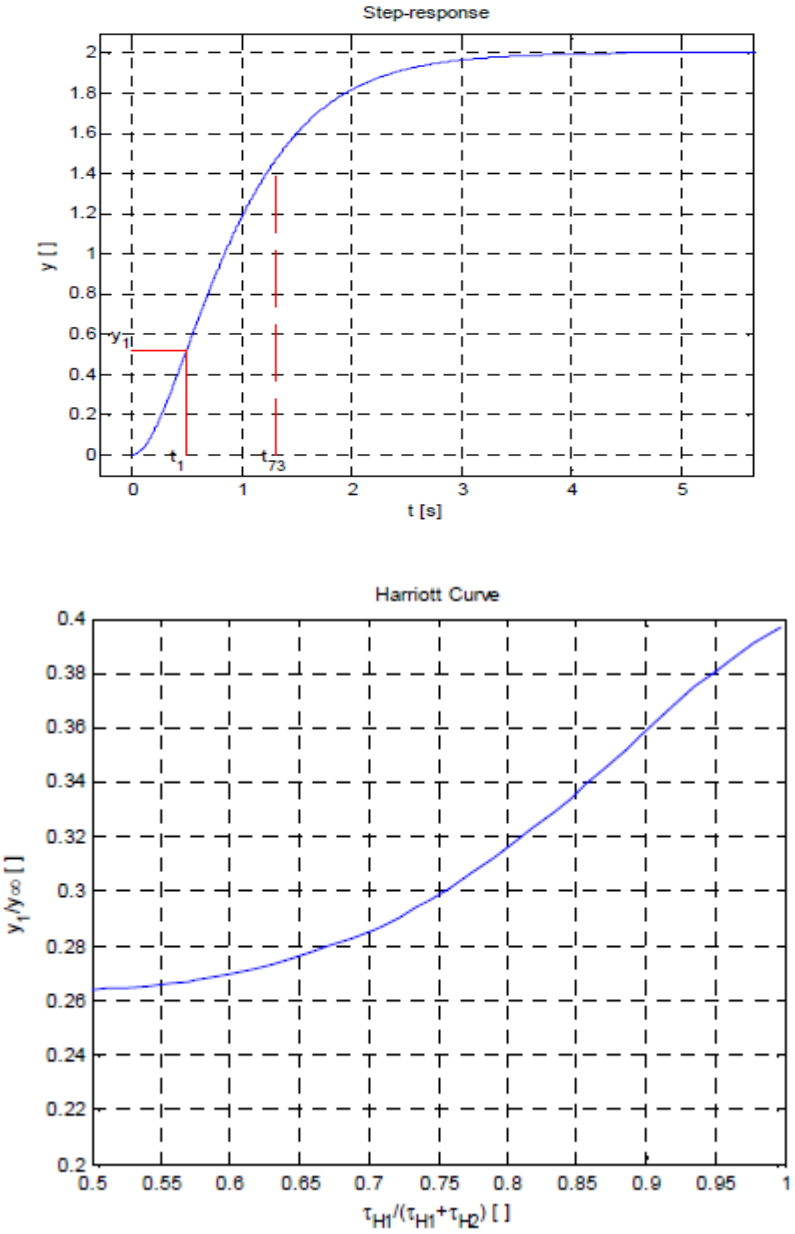
$$\frac{H(s)_{total}}{Qin(s)_{total}} = K1 * e^{-\theta s} * \frac{H(s)}{Qin(s)}\ K2 \quad (2.14)$$

2.3 Pemodelan Sistem Dengan Menggunakan Kurva Reaksi

Metode kurva reaksi adalah sebuah metode yang digunakan untuk mengidentifikasi model matematis sebuah sistem dinamik. Metode ini mengandalkan pengukuran kurva respons sistem terhadap suatu sinyal *input* yang diketahui. Dengan menggunakan metode ini, sistem dinamik dapat diidentifikasi sebagai suatu model matematis seperti persamaan diferensial atau sistem persamaan linear.

Metode Harriott (Jakoubek, 2009) mencoba untuk menyimulasikan sistem dengan menggunakan model orde 2 yang memungkinkan untuk menyertakan delay waktu. Fungsi transfer model yang dihasilkan dapat dituliskan sebagai berikut:

$$G_H(s) = \frac{K}{(\tau_{H1}+1)\,(\tau_{H2}+1)} e^{-T_{dH}s} \quad (2.15)$$



Gambar 2-5 Harriot's Curve

Berikut adalah tahapan untuk mendapatkan parameter τ_{H1}, τ_{H2} dan T_{dH} :

2.3.1 Menentukan jumlah dari parameter τ_{H1} dan τ_{H2}

Jumlah dari parameter τ_{H1} dan τ_{H2} dapat dirumuskan sebagai berikut:

$$\tau_{H1} + \tau_{H2} = \frac{t_{73}}{1.3} \quad (2.16)$$

Dengan t_{73} adalah waktu saat respons system berada pada kondisi 73% mendekati luaran *steady state* (Y_{ss}).

2.3.2 Menentukan parameter t_1 dan Y_1

Parameter t_1 parameter t_1 diperlukan untuk menghitung parameter Y_1, yang diperoleh dari respon sistem yang telah diidentifikasi. Rumus untuk menghitung parameter t_1 adalah sebagai berikut:

$$t_1 = \frac{\tau_{H1} + \tau_{H2}}{2} \quad (2.17)$$

2.3.3 Menentukan parameter pada kurva Harriot

Pada kurva Harriott, parameter yang diberikan terletak pada sumbu X dan Y. Parameter yang ditunjukkan pada sumbu Y adalah sebagai berikut:

$$\frac{Y1}{Y\infty}[\,] \; atau \; \frac{Y1}{Yss} \quad (2.18)$$

Setelah menentukan parameter $\frac{Y1}{Y\infty}[\,]$ pada sumbu Y, garis dapat ditarik untuk mendapatkan parameter $\tau_{H1}/(\tau_{H1} + \tau_{H2})$ pada sumbu X. Parameter ini merupakan koordinat sumbu X pada kurva Harriott. Dengan demikian, kita dapat menggambarkan kurva Harriott dengan menggunakan parameter yang diperoleh pada kedua sumbu tersebut.

2.3.4 Menentukan parameter τ_{H1}

Untuk mencari parameter τ_{H1} maka persamaan nilai $\tau_{H1}/(\tau_{H1} + \tau_{H2})$ dapat diasumsikam pada kurva Harriot sebagai x, kemudian substitusikan persamaan 2.19

$$\frac{\tau_{H1}}{(\tau_{H1} + \tau_{H2})} = x \quad (2.19)$$

$$\frac{\tau_{H1}}{{t_{73}}/{1.3}} = x \;\rightarrow\; \tau_{H1} = x * {t_{73}}/{1.3} \quad (2.20)$$

2.3.5 Menentukan Parameter τ_{H2}

Untuk mendapatkan parameter , subsitusikan persamaan 2.19 pada persamaan 2.22 sehingga diperoleh:

$$\tau_{H1} + \tau_{H2} = \frac{t_{73}}{1.3} \quad (2.21)$$

$$x * {t_{73}}/{1.3} + \tau_{H2} = \frac{t_{73}}{1.3} \quad (2.22)$$

$$\tau_{H2} = \frac{t_{73}}{1.3} - \left(x * \frac{t_{73}}{1.3}\right) \quad (2.23)$$

2.3.6 Menentukan parameter T_{dH}

T_{dH} adalah *time delay* yang dapat di metode Harriot dirumuskan sebagai berikut:

$$T_{dH} = 1.937\, t_{33} - 0.937 t_{70} \quad (2.24)$$

Untuk menggunakan hasil parameter τ_{H1} , τ_{H2} , dan T_{dH} Parameter tersebut dapat memasukkannya ke dalam persamaan 2.24. Jika nilai waktu t_{33} dan t_{70} kurang dari 0 atau bernilai negatif, maka sistem dianggap tidak memiliki time delay. Dengan menggunakan hasil parameter tersebut dalam persamaan 2.24, sehingga dapat ditentukan kinerja sistem tersebut.

2.4 Kontroler PID

Algoritma Kontroler PID merupakan kontrolir yang paling popular pada proses industri, meskipun ada kemajuan terus menerus pada teori kontrol. Hal ini dikarenakan algoritma PID memiliki struktur yang sederhana dan mudah dipahami secara konseptual serta dalam praktiknya kontroler ini memberikan respons kerja yang sangat memadai dalam aplikasinya pada *plant* di industri. Kontrolir PID bisa dianggap sebagai bentuk "ekstrem' dari kompensator lead-lag dengan fase satu kutub di titik asal dan yang lainnya pada titik tak terhingga. Adapun bentuk kontrolir PID ditulis dalam bentuk persamaan paralel di mana (Xiong & Fan, 2007)

$$G_{PID}(s) = Kp + \frac{Ki}{s} + Kd.s = Kp\ \left(1 + \frac{1}{Ti.s} + Td.s\right) \quad (2.25)$$

Di mana KP adalah *gain proportional*, Ki adalah *gain integral*, Kd adalah *gain derivative*, Ti adalah konstanta *integral time*, dan Td adalah konstanta *derivative time.*

$$u(t) = K_p e(t) + K_i \int e(\tau)d\tau + Kd\frac{de(t)}{dt} \quad (2.26)$$

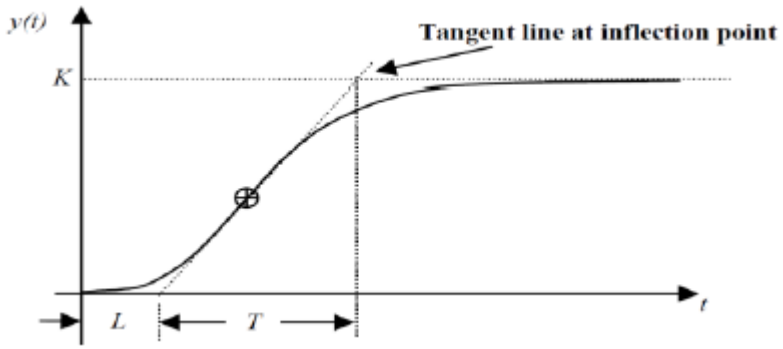
Untuk mendapatkan hasil tuning kontroler PID yang maksimal pada umumnya digunakana aturan tuning yakni *Ziegler-Nichols* atau *Coheen Coon*. Adapun karakteristik kontroler PID (Ang et al., 2005)adalah sebagai berikut

Tabel 2-3 Karakteristik Kontroller PID

Respon *Closed Loop*	***Rise Time***	***Overshoot***	***Settling time***	***Steady state error***	**Stabilitas**
Menaikkan Kp	Berkurang	Mengalami perubahan	Sedikit mengalami kenaikan	Mengalami penurunan	Menurunkan
Menaikkan Ki	Berkurang	Bertambah	Meningkat	Menurunkan signifikan	Menurunkan
Menaikkan Kd	Sedikit Perubahan	Berkurang	Berkurang	Berdampak minor	Menaikkan

2.5 Tuning PID dengan Ziegler Nichols

Metode pertama yang diperkenalkan oleh Ziegler-Nichols berdasarkan pada cara bagaimana sistem menanggapi suatu perubahan yang terjadi secara tiba-tiba. Dalam metode ini, kurva bagaimana sistem merespons perubahan harus terlihat seperti bentuk huruf S, yang ditandai oleh dua hal, yaitu berapa lama sistem tertunda untuk merespons dan berapa lama sistem memerlukan untuk kembali normal. Kedua hal tersebut ditentukan melalui cara sistem merespons perubahan yang terjadi. Seperti pada gambar berikut



Gambar 2-6 Kurva S Ziegler Nichols

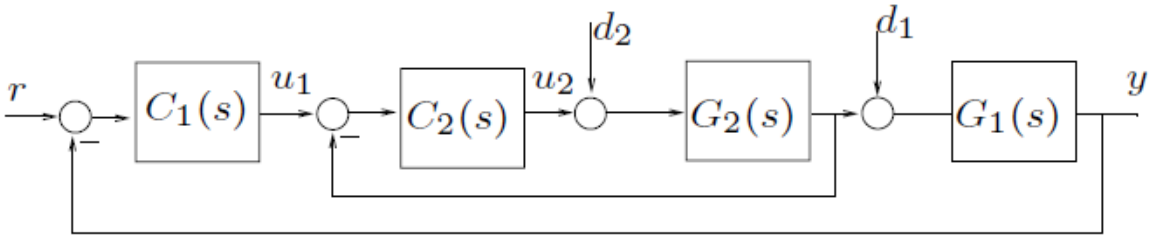
Titik di mana kemiringan step response memiliki nilai maksimum pertama ditentukan, dan sebuah gari ditarik pada titik tersebut. Persimpangan antara garis tersebut dan sumbu waktu memberikan parameter L, sementara persimpangan antara garis tersebut dan garis y(t)=K memberikan parameter T. Parameter PID dapat diberikan langsung sebagai fungsi dari a=KL/T dan L (Abdulameer et al., 2016).

Tabel 2-4 Tabel Tuning PID Controller dengan Zigler Nichols

Controller	Kp	Ti	Td
P	1/*a*	-	-
PI	0.9/*a*	3*L*	-
PID	1.2/*a*	2*L*	*L*/2

2.6 Sistem Pengaturan *Cascade*

Sistem pengaturan *cascade* adalah sistem yang terdiri dari *inner loop* dan *outer loop* seperti pada gambar (ket). di mana *inner loop* berasal dari pengenalan sensor tambahan untuk memisahkan sebanyak mungkin dinamika sistem proses cepat dan lambat. Akibatnya, *control* konfigurasi sistem memiliki pengontrol bagian dalam *C2(s)* dengan bagian dalam proses loop *G2(s)* dan kontrolir *loop* luar *C1(s)* dengan *outer* proses *loop G1(s*). Gangguan bisa masuk di dua kemungkinan titik yang berbeda: *d1* dan *d2*. Alasan di balik konfigurasi ini adalah untuk dapat dikompensasi yang terbaik, yang mungkin gangguan *d2*, sebelum direfleksikan ke *output loop* luar. Untuk mencapai tujuan atau *set point* yang diinginkan adalah penting bahwa *inner loop* menunjukkan dinamika yang lebih cepat yang memungkinkan untuk kompensasi.



Gambar 2-7 Diagram Cascade Control

Berdasarkan uraian di atas proses *G(s) = G1(s)G2(s)* dibagi menjadi dua bagian *G1(s)* dan *G2(s)* dan yang terkait pengontrol. Kedua pengontrol tersebut adalah pengontrol umpan balik standar yang diasumsikan mengambil bentuk PID biasa yakni:

$$C1(s) = K_{p1}\left(1 + \frac{1}{T_{i1}s} + \frac{T_{d1}s}{(1 + {}^{T_{d1}s}/_{N_1})s}\right) \quad (2.27)$$

$$C2(s) = K_{p2}\left(1 + \frac{1}{T_{i2}s} + \frac{T_{d2}s}{(1 + {}^{T_{d2}s}/_{N_2})s}\right) \quad (2.28)$$

Dimana *Kp1* dan *Kp2* adalah *gain* proporsional, T_{i1} dan T_{i2} adalah *gain* integral, T_{d1} dan T_{d2} *gain* turunan dan serta *N1* dan *N2* konstanta filter *noise* atau gangguan waktu *derivative*.(Vilanova & Arrieta, 2008)

2.7 Particle Swarm Optimization

Pendekatan PSO menggunakan algoritma optimasi stokastik berbasis populasi yang diusulkan oleh Eberhart dan Kennedy pada tahun 1995. Hal tersebut terinspirasi dari simulasi komputer yang mengambil contoh dari perilaku sosial kawanan burung, simulasi tersebut dicetuskan oleh Reynolds pada tahun 1987. Reynolds menggunakan grafik komputer untuk memodelkan perilaku berkelompok burung yang rumit. Dia tertarik pada simulasi pola terbang burung untuk tujuan simulasi komputer visual, mengamati bahwa kawanan tampaknya berada di bawah kendali terpusat. Reynolds mulai memodelkan penemuannya menggunakan tiga aturan sederhana, yaitu penghindaran tabrakan, pencocokan kecepatan, dan pemusatan kawanan. Dengan menggunakan aturan ini, Reynolds menunjukkan bagaimana perilaku masing-masing agen di dalam *flok* dimodelkan dengan vektor sederhana. Karakteristik ini merupakan salah satu konsep dasar PSO (Pillay, 2013).

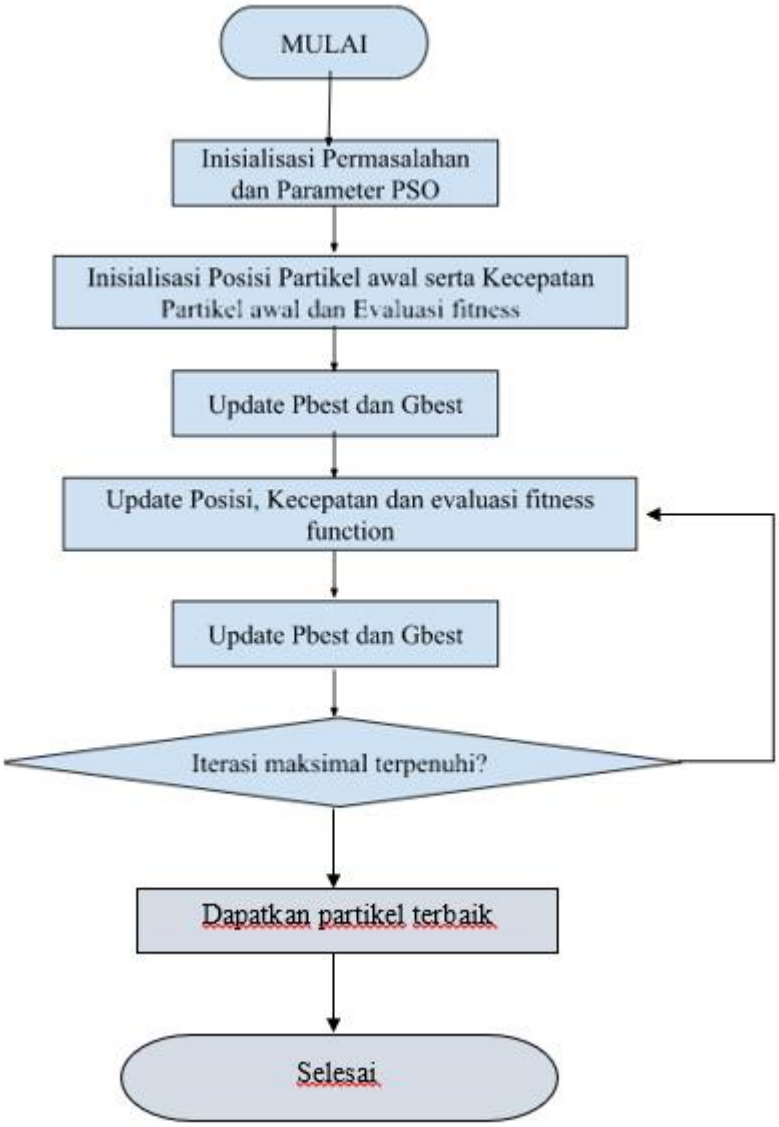
PSO adalah metode pencarian heuristik yang terinspirasi oleh interaksi sosial dari hewan yang hidup berkelompok. PSO dimulai dengan sekelompok partikel yang dihasilkan secara acak, yang masing-masing merupakan solusi yang mungkin untuk masalah optimasi. Setiap partikel diberi nilai posisi, dievaluasi oleh *fitness fuction* dirancang untuk mewakili masalah yang akan dioptimalkan. Selanjutnya, setiap partikel memiliki kecepatannya sendiri yang menentukan gerak partikel dalam ruang solusi (Pano & Ouyang, 2014).

Berdasarkan uraian tersebut maka dapat dinyatakan dalam bentuk persamaan yakni :

$$v_i^{k+1} = v_i^k + c_1 rand_1\, x\, (pbest_i - s_i^k) + c_2 rand_2\, x\, (gbest - s_i^k)$$

$$s_i^{k+1} = s_i^k + v_i^{k+1}$$

Di mana v_i^k adalah kecepatan sekarang agen i pada iterasi k, v_i^{k+1} adalah pembaruan kecepatan agen i pada iterasi k, $c1$ dan $c2$ merupakan faktor *learning*, *rand* adalah *factor random* yang terdistribusi pada [0,1], $pbest$ merupakan solusi optimal suatu partikel, dan $gbest$ merupakan solusi optimal seluruh populasi, sementara itu *s* merupakan lokasi partikel dan v merupakan kecepatan partikel.



Gambar 2-8 Flowchart Algoritma Particle Swarm Optimization

2.8 Indeks Performansi

Dalam mencari suatu fungsi optimal tidak dapat didefinisikan dengan tepat. Solusi yang menurut sebuah masalah adalah kondisi optimal, mungkin bukan nilai yang optimal bagi permasalahan lain. Indeks performansi banyak digunakan oleh kalangan praktisi dan juga akademisi untuk membantu dalam menentukan kualitas dari sebuah sistem. Indeks performansi sendiri sebenarnya adalah fungsi hubungan di mana beberapa karakteristik sistem seperti kondisi optimal dari sistem didefinisikan

2.8.1 Integral Time Absolute Error

Indeks performansi dari ITAE ditunjukkan pada Persamaan. Bobot pengali waktu pada ITAE membuatnya lebih cepat untuk meminimalkan kesalahan dibandingkan IAE.

$$\int_0^\infty t|e(t)|d(t)$$

ITAE memiliki nilai kesalahan awal yang besar dan kemudian kesalahan menghilang secara perlahan.

2.8.2 Integral Square Error

Integral Square Error (ISE) merupakan salah satu metode yang digunakan untuk mengukur kualitas dari suatu model kontrol atau identifikasi. ISE mengukur seberapa besar eror atau selisih antara sinyal yang dihasilkan oleh model dengan sinyal aktual yang ada pada sistem. ISE dihitung dengan mengalikan eror pada setiap waktu dengan durasi waktunya, lalu menjumlahkan semuanya. Nilai ISE yang lebih kecil menunjukkan bahwa model yang digunakan lebih baik dibandingkan dengan model yang lain.

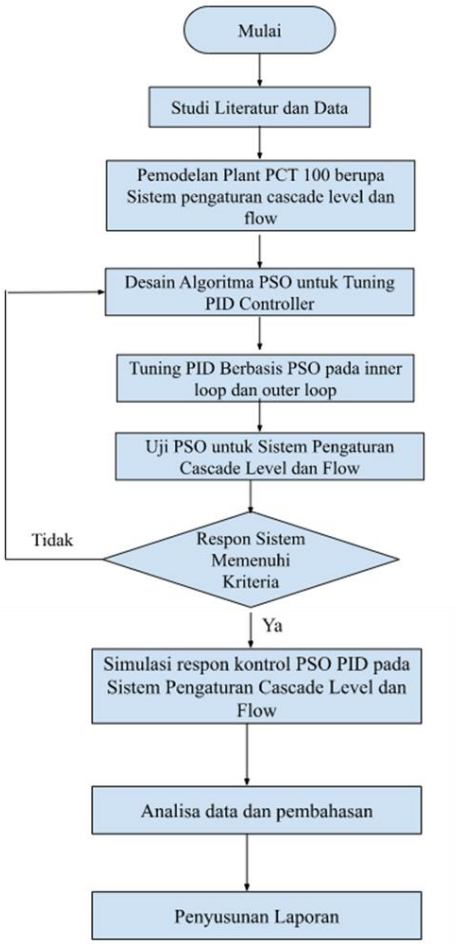
$$ISE = \int_{0}^{tsim} e2(t)\, dt$$

--Halaman ini sengaja dikosongkan--

BAB 3 PERANCANGAN SISTEM

3.1 Metode Penelitian

Dalam tugas akhir ini, metode utama yang digunakan yaitu simulasi. Pada simulasi, terdapat beberapa langkah khusus akan dilakukan, mulai dari identifikasi sistem, pemodelan sistem, perancangan kontrolir, hingga pengambilan data simulasi. Metode identifikasi sistem dalam simulasi akan dilakukan dengan memberikan sinyal uji pada *flow plant* PCT 100 sehingga diketahui karakteristik dari dari *flow plant* tersebut. Sedangkan untuk sistem pengaturan level akan dilakukan *modelling* dengan matematis berdasarkan parameter level tangki pada PCT 100. Perancangan kontrolir akan dilakukan dengan mengaplikasikan algoritma PSO mencari parameter kontrolir PID serta dibandingkan dengan Tuning PID *Ziegler Nichols*. Adapun metodologi pada penelitian ini adalah:



Gambar 3-1 Flowchart Diagram Alur Penelitian

3.2 Bahan dan Peralatan yang Digunakan

3.2.1 Process Control Technology 100 Byotronics

Gambar 3-2 Module Control PCT 100

PCT-100 adalah *miniplant* salah satu produk dari *Bytronic* yang digunakan untuk mendemonstrasikan prinsip kontrol dalam proses instrumentasi dan sistem kontrol proses. Pada peralatan ini terdapat proses yang terjadi pada tangki utama, yakni ada empat variabel proses yang diatur yaitu tekanan *(pressure*), level air, aliran air *(flow*), dan suhu (*temperature*). Aktuator yang terdapat pada PCT -100 adalah pompa dan pemanas. Pompa digunakan untuk menyalurkan air dari tangki penyimpanan menuju tangki utama. Sementara pemanas digunakan dalam proses sistem pengaturan suhu. Keempat variabel tersebut diatur dengan menggunakan kontrolir PID pada program bawaan alat ini

3.2.2 Adam 5000 L TCP/IP

Gambar 3-3 ADAM Advantech 5000L/TCP

ADAM-5000L / TCP dan ADAM-5000 / TPC adalah sistem I / O berbasis Ethernet yang dirancang untuk konfigurasi yang mudah dan pengelolaan yang efisien. Mereka dapat digunakan untuk mengonfigurasi dan memantau data secara jarak jauh dari berbagai perangkat, seperti sensor dan aktuator, melalui jaringan Ethernet. Kedua model memiliki jarak komunikasi hingga 100 m tanpa perlu *repeater*, dan hingga delapan PC dapat secara bersamaan mengakses data. Ini menjadikan mereka solusi yang efektif dan fleksibel untuk berbagai aplikasi otomasi.

Pada Tugas Akhir ini ADAM 5000L digunakan sebagai *Input Ouput Module* dikarenakan kecocokan pada sistem. Fungsi dari ADAM sendiri untuk memberi sinyal perintah berupa tegangan kepada Pompa PCT 100 dan pembacaan sensor *flow rate* dari *plant* PCT 100 dan juga

sebagai penerima sinyal dari sensor yang berupa tegangan. Spesifikasi dari ADAM 5000L dapat dilihat dari Gambar Tabel 3.1

Tabel 3-1 Spesifikasi ADAM

Konsumsi Daya	4W, 24 VDC (tdaik termasuk modul I/O)
Power Input	10-30 VDC
Konektor	1 x DB9-M/DB9-F/screw terminal untuk RS-485 (komunikasi), 1 x DB9-F untuk RS-232 (penggunaan internal), 1 x Screw terminal untuk power input, 1 x RJ-45 untuk LAN

3.2.3 Matlab

Gambar 3-4 Ikon Matlab

MATLAB adalah singkatan dari *Matrix Laboratory* dan merupakan sebuah bahasa pemrograman dan lingkungan komputasi numerikal yang digunakan untuk mengelola data, menghitung, dan mengembangkan algoritma. MATLAB dikembangkan oleh *MathWorks* dan menyediakan fungsi-fungsi matematis, visualisasi data, dan pemodelan integratif yang kuat untuk menganalisis dan memvisualisasikan data serta membangun aplikasi yang didasarkan pada hasil analisis tersebut. MATLAB juga dapat digunakan untuk membangun sistem kontrol, memproses sinyal, dan menganalisis data secara real-time.

3.2.4 Simulink

Gambar 3-5 Simulink

Simulink adalah sebuah perangkat lunak yang dikembangkan oleh *MathWorks* yang memungkinkan untuk menggambar dan mengubah model sistem dinamis dalam bentuk diagram blok. *Simulink* memiliki banyak library yang berisi blok-blok yang mewakili fungsi-fungsi yang biasa digunakan dalam pemodelan sistem dinamis, seperti filter, sumber, integrator,

dan lainnya. *Simulink* terdapat bagian blok diagram dan antar blok dapat dihubungkan untuk membangun model sistem yang sesuai dengan spesifikasi yang diinginkan. Kemudian, *Simulink* digunakan untuk menganalisis dan menjalankan simulasi model tersebut untuk menguji dan mengevaluasi bagaimana sistem akan berperilaku dalam situasi nyata.

3.2.5 Personal Computer

Komputer yang digunakan pada Tugas Akhir ini memiliki spesifikasi sebagai berikut

Tabel 3-2 Spesifikasi Komputer yang Digunakan

Prosesor	Intel(R) Core(TM) i5-8250U
RAM	12GB
STORAGE	1TB SATA (5400 rpm)

3.3 Urutan pelaksanaan penelitian

3.3.1 Pemodelan Sistem Pengaturan Level pada PCT 100

3.3.1.1 Pemodelan Pompa

Mengacu pada persamaan 2.1 dengan memasukkan rentang debit air yang keluar dari pompa antara 0 - 45 cm/s . Rentang tegangan masukan antara 0 – 10 V didapatkan hasil pada Persamaan 3.1

$$K1 = \frac{45 - 0}{10 - 0} = 4.5\ cm^3/sV \tag{3.1}$$

3.3.1.2 Pemodelan Pipa

Mengacu pada persamaan 2.2 dengan memasukkan data diameter pipa sebesar 1 cm, panjang pipa sebesar 90 cm dan dengan debit air yang masuk 16.67 cm3/detik didapatkan persamaan 3.2

$$\theta = \frac{70{,}69}{16.67} = 4.23\ s \tag{3.2}$$

Diperoleh *time delay*:

$$e^{-\theta s} = e^{-4.23s} \tag{3.3}$$

3.3.1.3 Pemodelan Tangki

Nilai Q0 didapatkan dengan melakukan uji pengosongan tangki. Diambil titik kerja ketika nilai h0 sebesar 7.5 cm didapatkan nilai Q0 seperti pada persamaan berikut

$$Q0 = \frac{V}{t} = \frac{1507.2}{110} = 13.701\frac{cm3}{s} \tag{3.4}$$

V adalah volume air yang keluar dan t adalah waktu yang dibutuhkan untuk mengosongkan tangki. Mengacu pada persamaan 2.4 dan persamaan 2.5 didapatkan nilai R dan C pada persamaan berikut

$$R = \frac{7.5}{13.701} = 0.547s/cm2 \tag{3.5}$$

$$C = 200.96\ cm2 \tag{3.6}$$

Sehingga di dapat fungsi alih:

$$\frac{H(s)}{Qin\ (s)} = \frac{0.547}{109.92s + 1} \tag{3.7}$$

3.3.1.4 Pemodelan Sensor

Mengacu pada Persamaan (Ket) dengan memasukkan rentang tegangan keluaran antara 0 – 10 V dan rentang ketinggian air dalam tangka antara 0 – 18 cm data sesuai dengan user manual PCT-100 , maka penguatan sensor dirumuskan pada Persamaan berikut

$$K2 = \frac{10 - 0}{18 - 0} = 0.55\frac{V}{cm} \tag{3.8}$$

3.3.1.5 Persamaan Model Sistem Pengaturan Level Total

Mengacu pada persamaan 2.12 dapat dihitung fungsi alih keseluruhan dari plant dengan cara mengalikan semua komponen pemodelan seperti pada persamaan berikut

$$\frac{H(s)total}{Qin\ (s)total} = 5.83 * e^{-4.23s} * \frac{0.547}{109.92s + 1} * 0.55$$

$$\frac{H(s)}{Qin\ (s)} = \frac{1.754e^{-4.23}}{109.92s + 1} \tag{3.9}$$

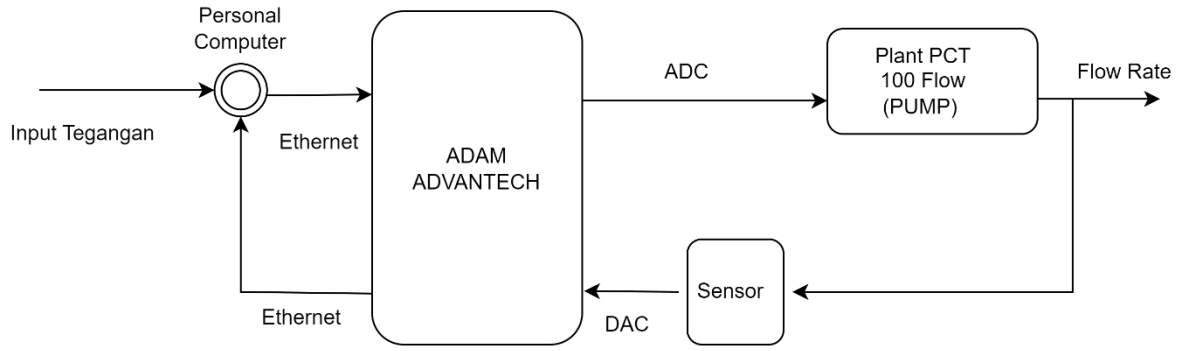
3.3.2 Pemodelan Sistem Pengaturan Flow

Dalam proses mengembangkan dan mendesain kontrolir untuk suatu plant, pengetahuan tentang sifat dan karakteristik plant tersebut sangat penting. Salah satu cara untuk mengetahui sifat dan karakteristik *plan*t adalah dengan menganalisis fungsi alih yang dimiliki oleh plant tersebut. Dengan mengetahui fungsi alih ini, kita dapat memahami karakteristik dari plant tersebut dan mengembangkan kontrolir yang sesuai untuk mengendalikannya. Salah satu permasalahan yang sering muncul ketika menghadapi suatu *plant* adalah tidak adanya dokumen yang menyediakan informasi tentang fungsi alih dari plant tersebut. Tanpa informasi ini, sulit bagi para peneliti untuk memahami karakteristik plant dan mengembangkan kontrolir yang sesuai untuk mengendalikannya. Oleh karena itu, penting bagi para peneliti untuk mencari tahu informasi ini sebelum mulai mengembangkan kontrolir untuk suatu *plant*.

Proses identifikasi adalah suatu cara untuk menentukan nilai parameter-parameter dari suatu plant dengan menganalisis respon plant terhadap sinyal uji yang diberikan. Proses ini biasanya dilakukan secara *open loop,* di mana sinyal uji diberikan dari komputer kontrolir ke *plant*, dan respon dari plant terhadap sinyal uji tersebut diterima kembali oleh komputer kontrolir. Kemudian, keluaran dari plant diolah menjadi sebuah grafik, dan dari grafik tersebut kita bisa mendapatkan nilai parameter-parameter plant. Identifikasi biasanya dilakukan pada kondisi *set point* yang berbeda-beda.

Menurut jurnal dengan penulis Ing. Pavel Jakoubek, terdapat 7 metode identifikasi yang dapat digunakan untuk sistem tanpa osilasi dengan masukan respons step. Setelah diperoleh fungsi alih sistem, ketujuh metode ini dibandingkan dengan respons hasil pengukuran untuk menentukan metode dengan pendekatan model yang terbaik. Metode validasi yang biasa digunakan dalam kasus ini adalah *ISE (Integral Square Error*). Dengan menggunakan metode ini, kita dapat membandingkan hasil dari model yang diperoleh dengan data yang diukur dari *plant* untuk menentukan metode yang paling baik untuk digunakan.

Untuk memodelkan flow kita menggunakan Adam Advantech sebagai jembatan untuk memperoleh data, Adam Advantech juga berfungsi sebagai ADC maupun DAC yang berguna untuk mengubah sinyal analog maupun digital atau converter tentunya berguna dalam control kontinu maupun diskrit. Adapun diagaram pemodelan *plant flow* adalah sebagai berikut:



Gambar 3-6 Diagram Block Proses Identifikasi

3.3.2.1 Pengambilan data Plant PCT 100 Sistem Pengaturan Flow

Untuk mendapatkan fungsi alih dari sistem dirancang sistem *closed loop* pada gambar 3.6 yang terdiri personal computer, ADAM 5000 L TCP/IP, dan Plant PCT 100. Setelah itu digunakan input berupa tegangan 5 Volt pada ADAM 5000L TCP/IP untuk memperoleh respons sistem yang akan dibaca oleh sensor flow. Digunakan algoritma 1 berikut ini untuk membaca hasil output dari sensor.

ALGORITME 1. Algoritme Pemodelan Plant Flow pada PCT 100

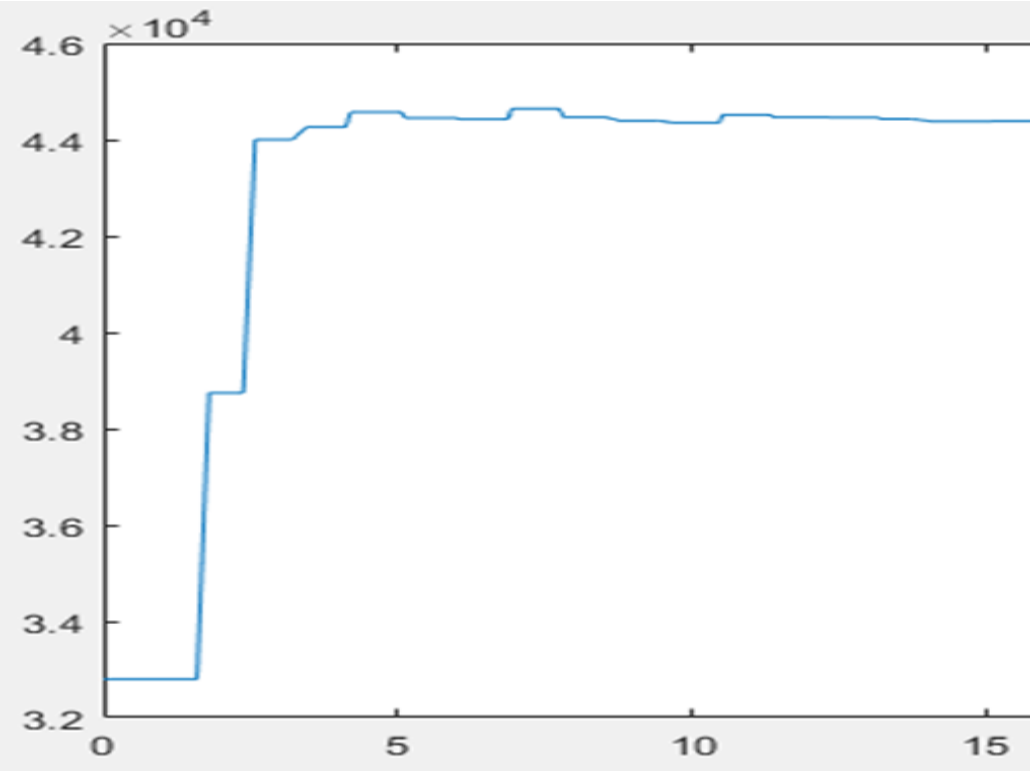
Input: Tegangan DC 5 Volt
Output : Bacaan sensor basis uint16

1. ***Data* : Plot Grafik uint16 (1:0.01:30)**
2. ***Set* : Alamat IP 169.254.126.231 pada port 502 menggunakan protokol Modbus**
3. ***Set* : Timeout untuk koneksi Modbus 30 detik**
4. ***Set* : ID server = 1**
5. ***Write : nilai 0 menggunakan fungsi “holdingregs” dengan alamat PORT = 17***
6. ***Read : Baca data dari” holdingregs” PORT = 9 dan simpan sebagai "data"***
7. ***Set : variable waktu t = 0***
8. ***For***

Baca data dari “holdingregs” PORT = 9 dan tambahkan ke "data"
Tambahkan "t" dengan waktu yang telah berlalu sejak iterasi sebelumnya
Plot "t" dan "data"
Gambarkan plot pada layar
Jeda selama 0,01 detik

9. ***end***
10. ***Putuskan koneksi dari perangkat***

Berdasarkan algoritma tersebut kemudian diimplementasikan pada Matlab untuk memperoleh data dari sensor *Flow Rate* diperoleh respons sistem berikut pada gambar 3.7



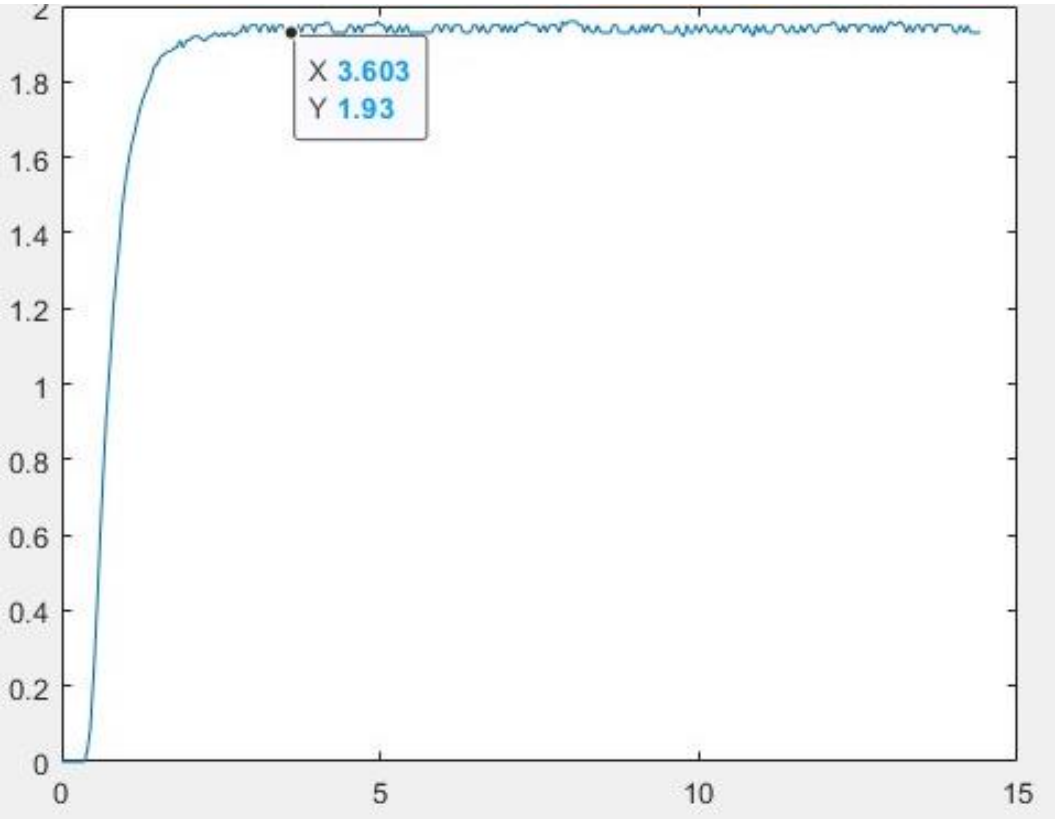
Gambar 3-7 Grafik Response dalam bentuk bilangan Bit

Dari respons sistem tersebut masih dalam berbentuk bilangan bit yakni uint16 untuk itu perlu dilakukan konversi ke bilangan asli pembacaan sensor sesuai spesifikasi sensor *flow rate* pada PCT 100. Bilangan uint 16 sendiri adalah bilangan dengan panjang 16 bit dimana dimulai dari 0-6555. Untuk respons sistem yang dihasilkan angka 0 liter/menit dimulai dari bilangan bit sebesar 32768 dan 2.7 liter/menit diangka 65535. Oleh karena itu dilakukan konversi dengan algoritma sebagai berikut:

ALGORITME 2. Algoritme Konversi bilangan uin16 ke bilangan basis sensor

1. ***Set :*** uint16_value = 32768;
2. // Konversi bilangan uint16 ke bilangan satuan sensor 2.7 liter/menit
   Hitung sensor_value = (uint16_value / 65535) * 2.7;
3. // Cetak hasil konversi
   Print(sensor_value);
4. // Baca file CSV
   data = ReadCSV('nama_file.csv');
5. // Konversi nilai dalam matriks menjadi bilangan satuan sensor 2.7 liter/menit
   converted_data = **Convert**(data, @(x) (x / 65535) * 2.7);
6. // Buat plot grafik dari hasil konversi
   Plot(converted_data);

Dari algoritma tersebut diperoleh grafik respons system sebagai berikut pada gambar 3.8. Dimana dari respons sistem tersebut akan dianalisis untuk dicari model fungsi alih dari sistem.



Gambar 3-8 Grafik Respons Sistem Open Loop Flow

3.3.2.2 Fungsi Alih Sistem

Untuk mendapatkan fungsi alih sistem, digunakan respons saat set poin adalah 5 volt atau 50% dari *flow rate* Gambar 3.8 menunjukkan respon level pada set point 50%. Dari respons tersebut, diperoleh nilai keluaran *steady state* pada ketinggian 1,939, yang diperoleh dari rata-rata seluruh data keluaran steady state. Berdasarkan respons pada Gambar 3.8, diperoleh *gain overall* respons pada persamaan berikut.

$$Y_{ss} = 1.939$$

$$X_{ss} = 2$$

$$K = \frac{Y_{ss}}{X_{ss}} = \frac{1.95}{1.89} = 1.03 \qquad (3.10)$$

Jakoubek mengembangkan 7 metode untuk mengidentifikasi *plant* atau sistem kontrol. Salah satu metode yang dapat digunakan dalam pendekatan model matematika plant adalah metode Harriott. Pemilihan metode terbaik dapat dilakukan dengan menggunakan validasi. Salah satu metode validasi yang dapat digunakan adalah ISE (*Integral Square Error*). Semakin kecil nilai ISE, maka semakin baik fungsi alih yang dibuat. Dengan menggunakan pendekatan sistem model matematika *plant* proses level dan berdasarkan respons pada Gambar 3.8, dapat diperoleh nilai ISE yang sesuai

$$t33 = 0.811$$

$$t70 = 1.716$$

$$t73 = 1.809$$

Sehingga diperoleh parameter:

$$T_{dH} = 1.937\, t_{33} - 0.937 t_{70} \tag{3.11}$$

$$T_{dH} = 1.937 * 0.811 - 0.937 * 1.716 = 1.57 - 1.607 = -0.037 \tag{3.12}$$

$$\tau_{H1} + \tau_{H2} = \frac{t_{73}}{1.3}$$

$$\tau_{H1} + \tau_{H2} = \frac{1.809}{1.3} = 1.39 \tag{3.13}$$

$$t_1 = \frac{\tau_{H1} + \tau_{H2}}{2} = \frac{1.39}{2} = 0.695 \rightarrow y_1 = 0.77 \tag{3.14}$$

Berdasarkan kurva Harriott pada Gambar 2.5, diperoleh nilai:

$$\frac{y_1}{Y_{ss}} = \frac{0.77}{1.939} = 0.397 \rightarrow \frac{\tau_{H1}}{(\tau_{H1} + \tau_{H2)}} = 0.561 \tag{3.15}$$

$$\frac{\tau_{H1}}{(\tau_{H1} + \tau_{H2)}} = x \rightarrow = \tau_{H1} = x.\frac{t_{73}}{1.3} \tag{3.16}$$

$$\tau_{H1} = 0.561\, x\, \frac{1.89}{1.3} = 0.815 \tag{3.17}$$

$$\tau_{H2} = \frac{t_{73}}{1.3} - \left(x * \frac{t_{73}}{1.3}\right) = 1.39 - 0.815 = 0.575 \tag{3.18}$$

Karena parameter *delay* bernilai negatif, sehingga nilai *delay* tidak dianggap. Fungsi alih untuk metode Harriot adalah:

$$GH(s) = \frac{K}{(\tau H1s + 1)(\tau H2s + 1)}\, e^{-TdHs}$$

$$GH(s) = \frac{1.03}{(0.815s + 1)(0.575s + 1)}$$

$$GH(s) = \frac{1.03}{0.468s^2 + 1.39s + 1} \tag{3.19}$$

Penelitian ini mengukur respon dari model terhadap masukan sinyal uji untuk menentukan kesesuaiannya dengan *real plant* dalam proses identifikasi. Hasil percobaan menunjukkan bahwa ISE sebesar dan model matematikanya merupakan pendekatan orde 2, yang umumnya sesuai dengan sistem orde 2 pada dunia industri.

3.3.3 Pemodelan State Space Sistem Pengaturan Cascade Level dan Flow

Pada tahap ini simulasi plant Model matematika pada pemodelan fisik plant, didapatkan persamaan seperti (3.9) dan (3.22) dapat ditampilkan seperti berikut:

Sistem Pengaturan Level: $\frac{H(s)}{Qin\,(s)} = \frac{1.754e^{-4.23}}{109.92s+1}$

Sistem Pengaturan Flow: $GH(s) = \frac{1.03}{0.468s^2+1.39s+1}$

Dari persamaan di atas, terdapat 2 loop yang berbeda, yaitu loop dalam dan loop luar. Oleh karena itu, langkah pertama yang harus dilakukan adalah menghitung loop dalam atau loop untuk mengontrol kecepatan, lalu menggabungkannya dengan loop luar atau loop untuk menyesuaikan level. kemudian diperoleh persamaan fungsi alih dari persamaan di atas, yaitu:

$$\frac{Y}{u} = \frac{1.848}{51.44\,s^3 + 153.3\,s^2 + 224.5\,s + 3.878} \tag{3.20}$$

$$51{,}44s^3Y + 153.2s^2Y + 224.5sY + 3.878Y = 1.848u$$

$$51.44\,\frac{d^3Y}{dt^3} + 153.3\frac{d^2Y}{dt^2} + 224.5\frac{dY}{dt} + 3.878Y = 1.848u$$

Dimisalkan:

$$x1 = Y$$

$$x2 = \dot{x1} = \dot{Y}$$

$$x3 = \dot{x2} = \ddot{x1} = \ddot{Y}$$

$$\dot{x3} = \ddot{x2} = \dddot{x1} = \dddot{Y}$$

Berdasarkan permisalan di atas diperoleh:

$$\dddot{Y} = -\frac{153.3}{51.44}\ddot{Y} - \frac{224.5}{51.44}\dot{Y} - \frac{3.878}{51.44}\,Y + \frac{1.848}{51.44}u$$

$$\dot{x3} = -2.9802x3 - 4.3643x2 - 0.0754x1 + 0.0359$$

Sehingga diperoleh matriks dari persamaan tersebut sebagai berikut:

$$\begin{bmatrix}\dot{x1}\\ \dot{x2}\\ \dot{x3}\end{bmatrix} = \begin{bmatrix}0 & 1 & 0\\ 0 & 0 & 1\\ -0.0754 & -4.3643 & -2.98017\end{bmatrix}\begin{bmatrix}x1\\ x2\\ x3\end{bmatrix} + \begin{bmatrix}0\\ 0\\ 0.03592\end{bmatrix}u \tag{3.21}$$

$$y = [1 \quad 0 \quad 0]\begin{bmatrix} x1 \\ x2 \\ x3 \end{bmatrix} + [D]u \tag{3.22}$$

Untuk persamaan state space di atas masih dalam bentuk cascade *Single Input Single Output*. Untuk mengubah dalam bentuk Single Input Multi Output maka dilakukan modifikasi matriks state space dari persamaan sistem pengaturan level dan sistem pengaturan flow yang terdapat pada persamaan 3.9 dan 3.19. Dari persamaan tersebut diperoleh persamaan state space dari sistem pengaturan level yang terdapat pada persamaan 3.23 dan 3.24. Persamaan tersebut diperoleh dengan menggunakan fungsi tf2ss dari matlab.

$$[\dot{x1}L] = [-0.0091][x1L] + [1]u \tag{3.23}$$
$$y = [0.0160][x1L] + [0]u \tag{3.24}$$

Selanjutnya dimodelkan state space untuk sistem pengaturan flow pada persamaan 3.19 dengan menggunakan matlab diperoleh matriks state spacenya pada persamaan 3.25 dan 3.26.

$$\begin{bmatrix} \dot{x1}F \\ \dot{x2}F \end{bmatrix} = \begin{bmatrix} -2.971 & -2.1368 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} x1F \\ x2F \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix}u \tag{3.25}$$

$$y = [0 \quad 2.2009]\begin{bmatrix} x1F \\ x2F \end{bmatrix} + [0]u \tag{3.26}$$

Kemudian dari persamaan 3.23, 3.24, 3.25, dan 3.26 diperoleh matriks state space gabungan untuk Sistem Pengaturan Single Input Multi Output pada persamaan 3.27 dan 3.28.

$$\begin{bmatrix} \dot{x1}L \\ \dot{x1}F \\ \dot{x2}F \end{bmatrix} = \begin{bmatrix} -0.0091 & 0 & 0 \\ 0 & -2.971 & -2.137 \\ 0 & 1 & 0 \end{bmatrix}\begin{bmatrix} x1L \\ x1F \\ x2F \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}u \tag{3.27}$$

$$y = [0.0160 \quad 0 \quad 2.2009]\begin{bmatrix} x1L \\ x1F \\ x2F \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}u \tag{3.28}$$

Dengan keterangan sebagai berikut:

$x1L = level$

$x1F = Laju\ aliran$

$x2F = Perubahan\ laju\ aliran$

$\dot{x1}L = Laju\ level\ naik\ turun$

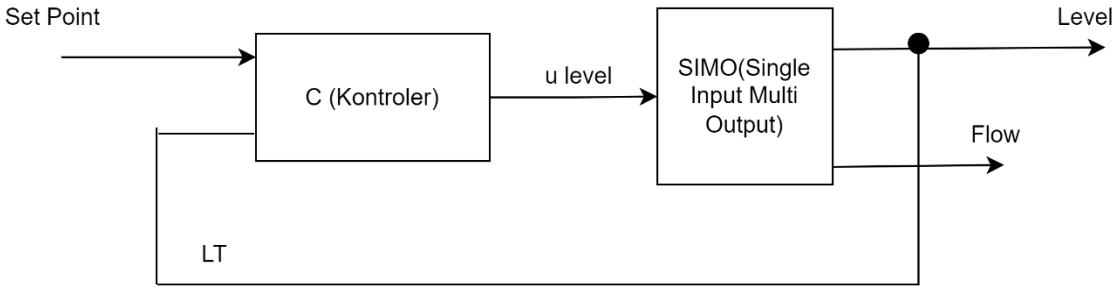
$\dot{x1}F = Perubahan\ laju\ aliran$

$\dot{x2}F = Laju\ perubahan\ flow$

3.3.4 Tuning PID Menggunakan pada Sistem Pengaturan Cascade

Cascade control adalah metode kontrol yang menggunakan dua atau lebih PID (*proportional-integral-derivative*) kontrolir untuk mengendalikan suatu sistem. Dalam kontrol kaskade, salah satu *PID controller* disebut "kontrolir primer" dan yang lainnya disebut

"kontrolir sekunder". Dalam tugas akhir ini dirancang sebuah sistem kaskade pada matlab yang terdiri dari 2 PID kontrolir masing masing primer dan sekunder serta persamaan 2 buah *plant* yakni Sistem pengaturan *flow* dan sistem pengaturan level seperti pada gambar 3.9.



Gambar 3-9 Sistem Pengaturan Cascade Single Input Multi Output

Untuk mengatur *PID controller* dengan menggunakan kontrol kaskade, pertama-tama kita perlu mengidentifikasi sistem yang akan dikendalikan dan menentukan apa yang akan menjadi variabel *output* dan *input*. Kemudian, kita perlu menentukan kontrolir primer dan kontrolir sekunder, serta mengatur parameter PID untuk setiap kontrolir.

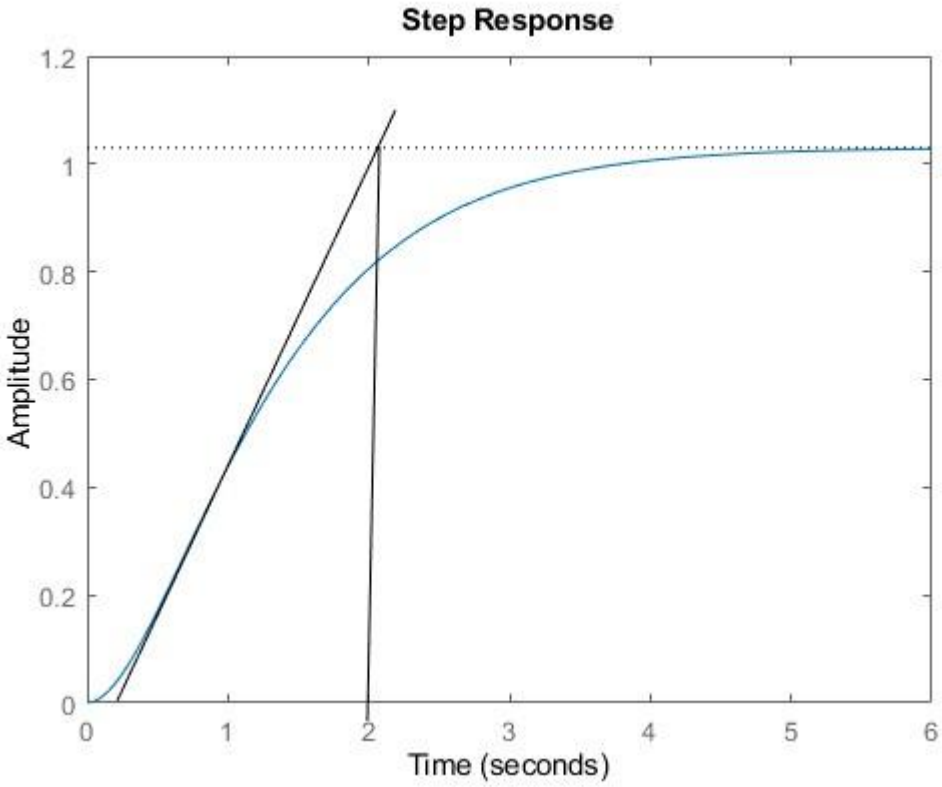
Setelah kontrolir telah dikonfigurasi, langkah selanjutnya adalah mengukur respons sistem terhadap kontroler dan mengoptimalkan parameter PID agar sistem dapat beroperasi dengan stabil dan memberikan hasil yang diinginkan. Ini bisa dilakukan dengan menggunakan teknik seperti trial and error, atau dengan menggunakan metode matematik yang lebih rumit seperti *Ziegler-Nichols* atau *Cohen-Coon*.

Setelah parameter PID telah dioptimalkan, kita dapat memantau sistem secara terus-menerus untuk memastikan bahwa sistem tetap stabil dan memberikan hasil yang diinginkan. Jika diperlukan, kita dapat mengubah parameter PID secara berkala untuk menyesuaikan diri dengan perubahan kondisi sistem atau tujuan kontrol.

Dengan menggunakan kontrol kaskade, kita dapat meningkatkan keandalan dan stabilitas sistem dengan mengendalikan beberapa variabel *output* secara bersamaan. Ini memungkinkan kita untuk mencapai hasil yang lebih baik daripada menggunakan satu PID kontrolir saja.

3.3.5 Tuning Kontrolir PID Inner Loop dan Outer Loop dengan Metode Ziegler Nichols

Untuk menala Kontrolir PID dilakukan di inner loop terlebih dahulu yakni sistem pengaturan flow. Grafik respons sistem pengaturan flow pada inner loop terdapat pada gambar 3.10.



Gambar 3-10 Tuning PID Zieglers Nichols Inner Loop

Dari gambar 3.10 diperoleh:

$$K = 1.03$$

$$L = 0.25$$

$$T = 2 - L = 2 - 0.25 = 1.75$$

Berdasarkan persamaan tersebut diperoleh:

$$a = K * \frac{L}{T} = 0.147$$

$$Ti = 2 * L = 0.5$$

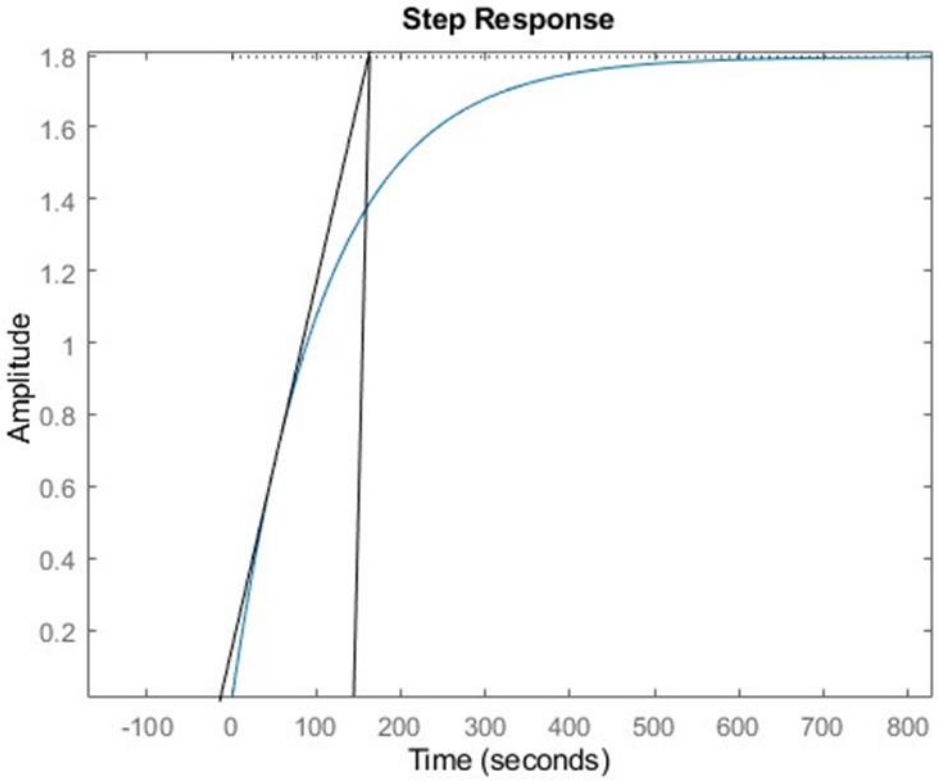
$$Td = \frac{L}{2} = 0.125$$

Berdasarkan Tabel 2.4 diperoleh besaran Kontrolir PID:

Tabel 3-3 Tabel PID Ziegler Nichols Inner Loop

Kp	Ki	Kd
8.1632	16.324	1.0204

Setelah menala kontroler PID Inner Loop pada kontroler kaskade dilakukan Tuning PID pada outer loop yakni sistem pengaturan level. Grafik respons open loop pada sistem pengaturan level ditunjukkan pada gambar



Gambar 3-11 Tuning PID Ziegler Nichols Outer Loop

Dari gambar 3.11 dapat diperoleh besaran K, T dan L untuk digunakan dalam menentukan besaran Kp, Ki, Kd sebagai berikut:

$$K = 1.8$$

$$L = 56$$

$$T = 145 - L = 89$$

Menurut tabel 2.4 diperoleh besaran a, Ti, dan Td sebagai berikut:

$$a = K * \frac{L}{T} = 1.132$$

$$Ti = 2 * L = 112$$

$$Td = \frac{L}{2} = 28$$

Dari data di atas diperoleh:

Tabel 3-4 Tuning PID Ziegler Nichols Outer Loop

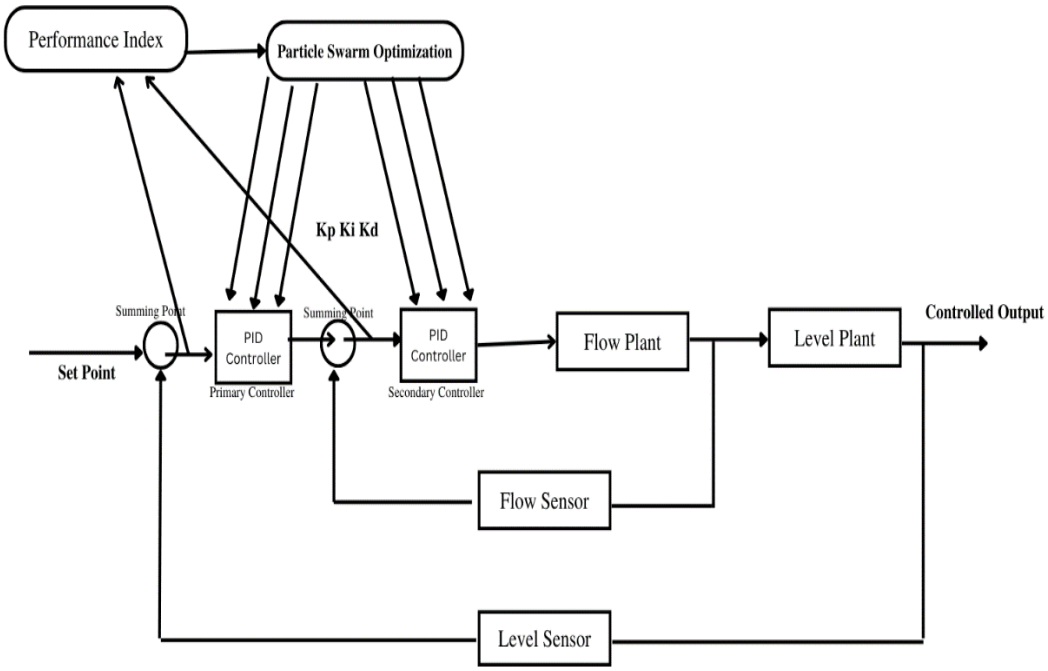
Kp	Ki	Kd
1.0595	0.0095	29.6667

3.3.6 Tuning PID dengan Metode Particle Swarm Optimization

Tuning PID menggunakan PSO merupakan salah satu metode yang dapat digunakan untuk mencari nilai konstanta PID (*Proportional, Integral, dan Derivative*) yang optimal untuk

mengendalikan sistem dinamis. Metode ini memanfaatkan kinerja dari kontroler PID yang sudah ada dengan mengoptimalkan nilai konstanta PID untuk mencapai performa yang lebih baik.

Dalam penerapannya, PSO akan mengoptimalkan nilai konstanta PID dengan memperhitungkan nilai error sistem, waktu respons, dan *overshoot*. PSO akan menggunakan partikel-partikel virtual yang mewakili nilai konstanta PID, dan mencari solusi terbaik dengan mengikuti aturan-aturan yang sama seperti kelompok serangga dalam lingkungannya. Nilai konstanta PID yang dihasilkan dari proses optimasi menggunakan PSO diharapkan dapat meningkatkan performa kontrolir PID dalam mengendalikan sistem dinamis.



Gambar 3-12 Diagram Block Tuning PID-PSO Pada Sistem Pengaturan Cascade Level dan Flow

Berikut ini adalah algoritma tuning PID menggunakan PSO dalam bentuk pseudo code yang terdapat input, output, variabel, dan parameter:

ALGORITME 3. Algoritme Tuning PID menggunakan PSO

INPUT:

- Transfer function (TF)
- Jumlah partikel (N)
- Nilai w (inertia weight)
- Nilai c1 (cognitive weight)
- Nilai c2 (social weight)
- Nilai error konvergensi (error_threshold)
- Batas maksimum iterasi (max_iter)

OUTPUT:

- Nilai PID yang teroptimal

VARIABLE:

- velocity[N]
- position[N]
- error[N]
- fitness[N]
- best_point[N]
- best_fitness[N]

PARAMETER:

- Nilai kecepatan awal (initial_velocity)
- Nilai titik awal (initial_point)

ALGORITMA:

a. Set initial_velocity dan initial_point sebagai nilai awal untuk setiap partikel
b. Untuk setiap partikel:
   - Hitung error menggunakan TF dan nilai PID pada posisi partikel tersebut
   - Hitung fitness menggunakan nilai error
   - Set best_point sebagai posisi partikel yang memiliki fitness terbaik
   - Set best_fitness sebagai fitness terbaik yang diperoleh
c. Ulangi langkah 2 hingga kriteria konvergensi tercapai atau iterasi mencapai batas maksimum
d. Set nilai PID yang teroptimal sebagai posisi partikel yang memiliki fitness terbaik (best_fitness)
e. Selesai.*Top of Form*

3.3.7 Pseudocode Tuning PID menggunakan Particle Swam Optimization

a. Input
   - *Objective Function* (fitness function), *upper bound* (ub) dan *lower bound* (lb), jumlah populasi (N), faktor inersia (w), individual dan social cognitive (c1 dan c2), Jumlah iterasi T.

b. Inisialisasi
   - Inisial *random* posisi (x) dan kecepatan (v) dalam batas ruang
   - Menetapkan pbest dan gbest (berdasarkan fungsi tujuan)

c. Loop
   - For t = 1 : T
   - For i= 1 : N
   - *Update* kecepatan: $vi(t) = wvi(t-1) + c1r1\big(pbesti - xi(t-1)\big) + c2r2\big(gbest - xi(t-1)\big)$
   - *Update* posisi: $xi(t) = xi(t-1) + vi(t)$
   - Periksa apakah xi(t) masih dalam batas: if $xi(t) > xub \rightarrow xi(t) = xub$ & $xi(t) < xlb \rightarrow xi(t) = xlb$
   - Evaluasi fungsi tujuan fxi

- *Update* Pbest dan Gbest
- Jika tidak ada konvergensi pada solusi sekarang dan if t>T maka Kembali ke loop

d. *Print* Gbest dan Fgbest

3.3.8 Parameter Pengujian

Pada percobaan ini dilakukan dengan fungsi tujuan adalah meminimalkan indeks performansi dari *Integral Time Absolute Error* sehingga diperoleh nilai minimal dari ITAE sehingga diperoleh besaran Kp, Ki, dan Kd. Sebelumnya Kp, Ki, Kd adalah dimisalkan 3 buah partikel yang mencari nilai optimal pada sebuah ruang 3 dimensi sumbu X, Y dan Z. Dari setiap masing-masing partikel akan mencari posisi yang paling optimal berdasarkan *objective function*-nya. Fitness value pada metode tuning PID adalah nilai hasil paling minimal dari ITAE sehingga jika sudah mencapai paling minimal maka iterasi akan berhenti. Dari *pseudocode* dan deskripsi yang telah disampaikan akan dilakukan pengujian dengan parameter c1=1.5, c2=1, jumlah maksimal iterasi adalah 100 dan batas atas serta bawah dimulai dari 0 sampai dengan 100. Kemudian untuk pengujian akan dilakukan variasi w=0.4, w=0.8 dan w=0.9. serta jumlah partikel N=10, N=30 dan N=50. Dari ketiga variasi pengujian tersebut akan dapat ditentukan dampak dari masing-masing jumlah partikel dan bobot inersia. Sehingga dapat ditentukan mana nanti yang terbaik akan digunakan untuk pengujian Tuning PID menggunakan PSO untuk sistem pengaturan *cascade* level dan *flow*.

--Halaman ini sengaja dikosongkan--

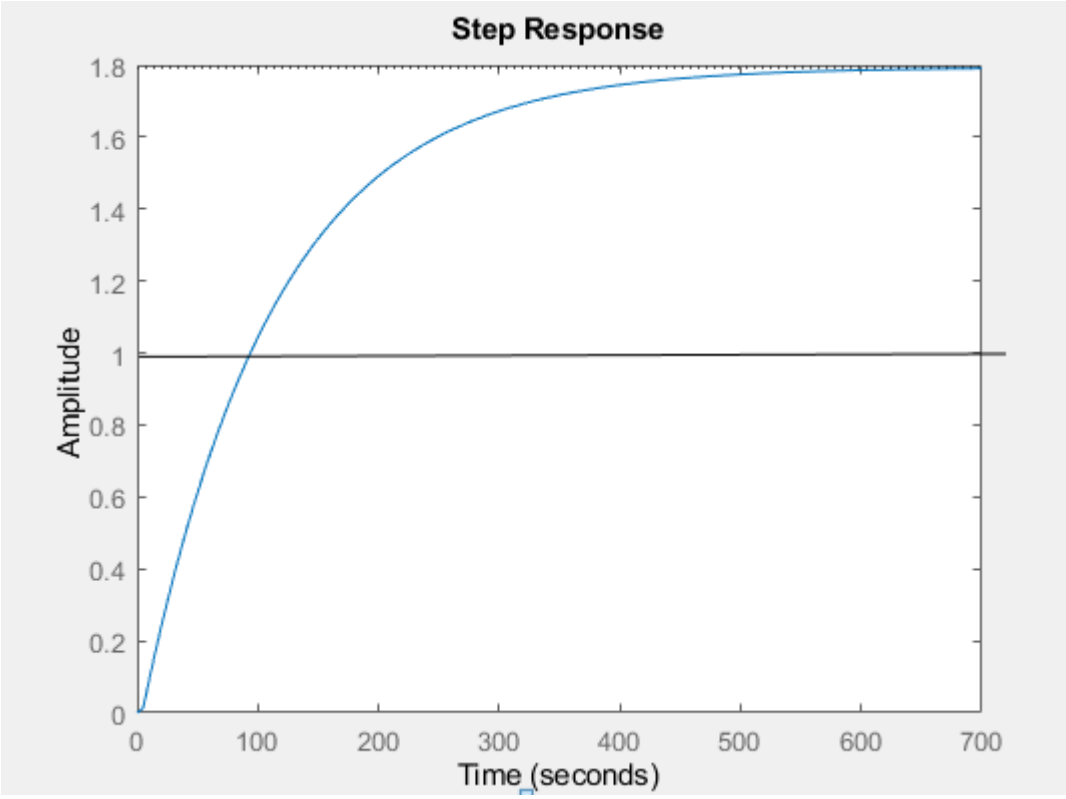
BAB 4 HASIL PENELITIAN DAN PEMBAHASAN

4.1 Hasil Pemodelan dari Sistem Pengaturan Level

Berdasarkan hasil pemodelan sistem pada Bab 3 didapatkan model akhir *plant* seperti pada Persamaan 4.1

$$\frac{H(s)}{Qin\ (s)} = \frac{1.754e^{-4.23}}{109.92s + 1} \qquad (4.1)$$

Dilakukan pengujian sistem umpan balik tertutup menggunakan sinyal unit step pada Persamaan 4.1 didapatkan hasil respons pada Gambar 4.1.



Gambar 4-1 Open Loop Response Level

Berdasarkan respon loop terbuka yang diperoleh dari pemodelan matematik sistem, masih terdapat *error steady state* yang besar, yaitu 0.75. Dengan adanya permasalah tersebut, perlu digunakan kontrolir untuk meminimalkan kesalahan waktu tunak pada sistem.

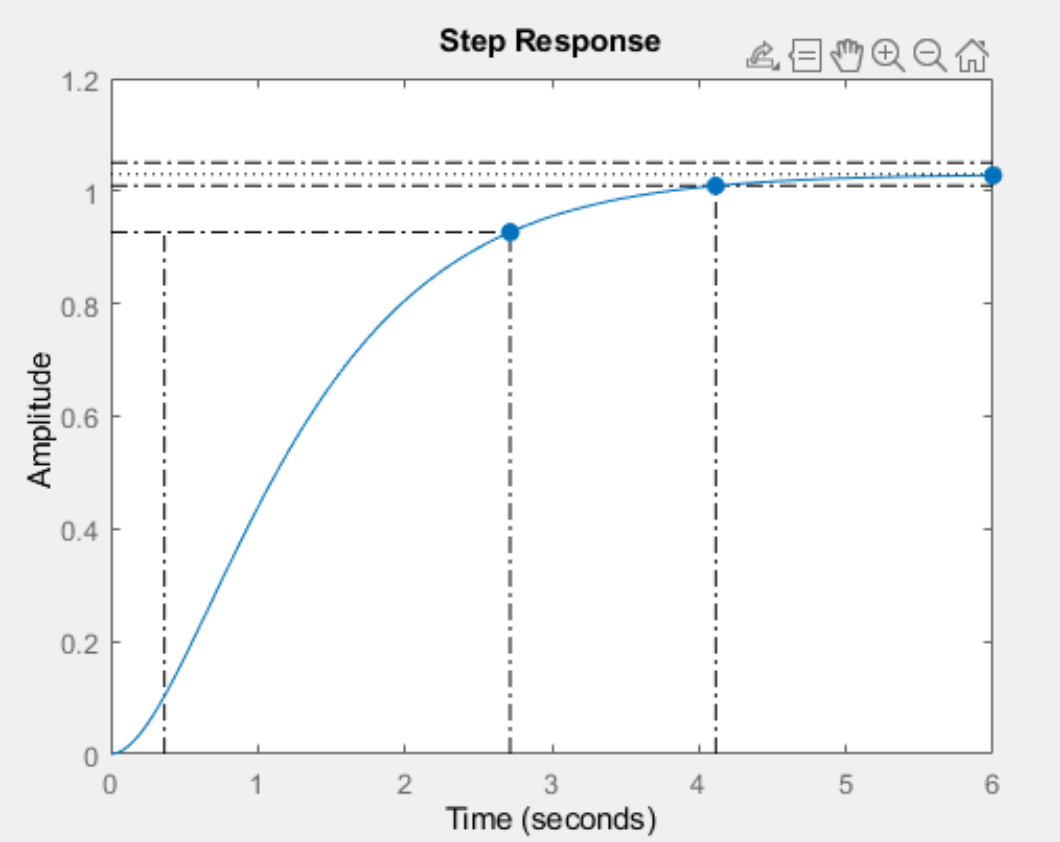
Pada sistem tersebut juga diperlukan kontrolir yang dapat mengurangi efek pembebanan yang berasal dari perubahan nilai beban R. Perubahan nilai tersebut disebabkan oleh perubahan bukaan *control valve* pada sistem.

4.2 Hasil Pemodelan dari Sistem dari Sistem Pengaturan Flow

Berdasarkan hasil pemodelan sistem pada Bab 3 didapatkan model akhir *plant* seperti pada Persamaan 4.2

$$GH(s) = \frac{1.03}{0.468s^2 + 1.39s + 1} \qquad (4.2)$$

Dilakukan pengujian *open loop* menggunakan sinyal unit step pada Persamaan 4.2 didapatkan hasil respon pada Gambar 4.1. Gambar 4.1 Respons *open loop* sistem pengaturan *flow* tanpa kontrolir.



Gambar 4-2 Respons Open Loop Flow

Diperoleh hasil karakteristik sistem tersebut adalah :

Tabel 4-1 Karakteristik sistem open loop flow

RiseTime	2.3516
TransientTime	4.1141
SettlingTime	4.1141
SettlingMin	0.9307
SettlingMax	1.0291
Overshoot	0
Undershoot	0
Peak:	1.0291
PeakTime:	6.7395

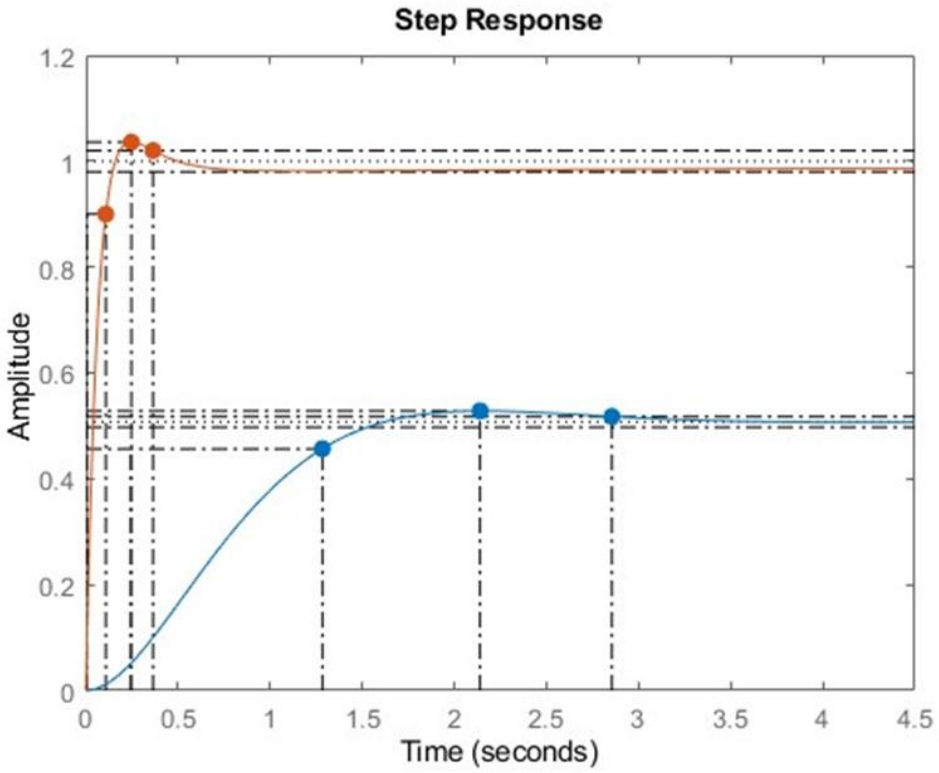
Dari data tersebut *steady state* pada 1.03 dari masukan set point atau sinyal step 1 sehingga didapat error steady state sebesar 1 – 1.03 = 0.03.Pada kondisi belum di-*tuning* dan respons sistem open loop, sistem ini memiliki karakteristik rise time 2.35 detik, *settling time* yakni waktu mendekati *steady state* sebesar 4.11 detik dan tidak memiliki *overshoot*. Kesalahan keadaan tunak tersebut yang menjadikan dasar perlunya kontroler pada sistem pengaturan level PCT-100. Spesifikasi yang diinginkan adalah mampu menghilangkan kesalahan keadaan tunak serta mampu mempercepat respons sistem.

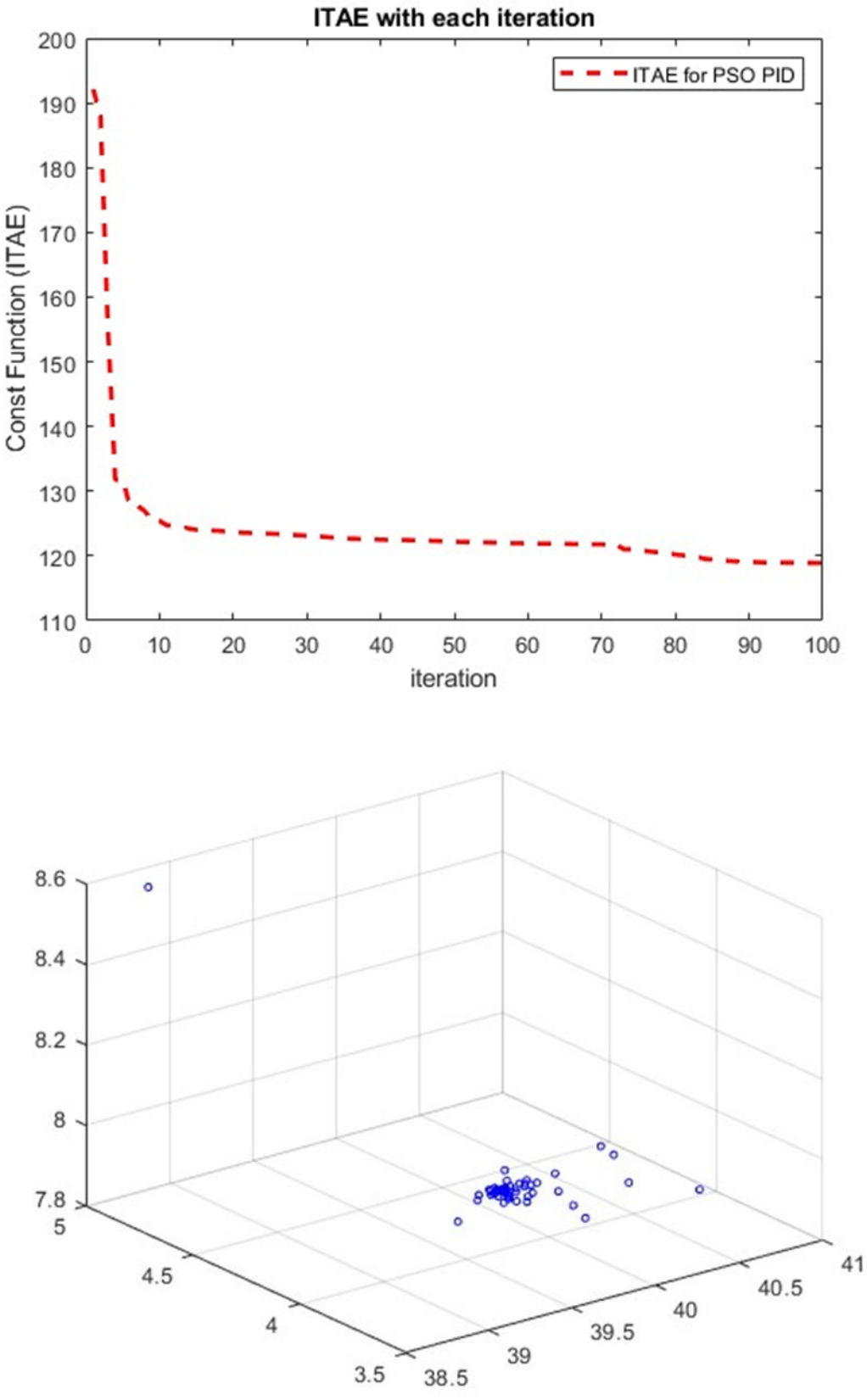
4.3 Hasil Pengujian Tuning PID dengan PSO Inner Loop

Pada pengujian inner loop dilakukan variasi pengujian bobot inersia sebesar 0.4, 0.8 dan 0.9, serta dilakukan pengujian variasi jumlah partikel sebesar 10, 30, dan 50.

4.3.1 Pengujian Variasi w=0.4.

Dalam pengujian ini, variabel w ditetapkan dengan nilai 0.4 dan parameter N sebesar 50, c1 sebesar 1.5, c2 sebesar 1, serta jumlah iterasi maksimal ditentukan sebesar 100 kali iterasi. Grafik step respons, fungsi global per iterasi, serta posisi *swarm* dari variable Kp, Ki, dan Kd yang optimal diperoleh melalui algoritma PSO. Data karakteristik sistem sebelum dan sesudah di-*tuning* tersedia pada tabel 4.2. Hasil terbaik dari pengujian ini adalah nilai minimum ITAE sebesar 118.9098 dari *cost function* 314.1645 dan nilai gain optimal Kp, Ki, dan Kd masing-masing adalah 39.9480, 4.2363, dan 7.8599.

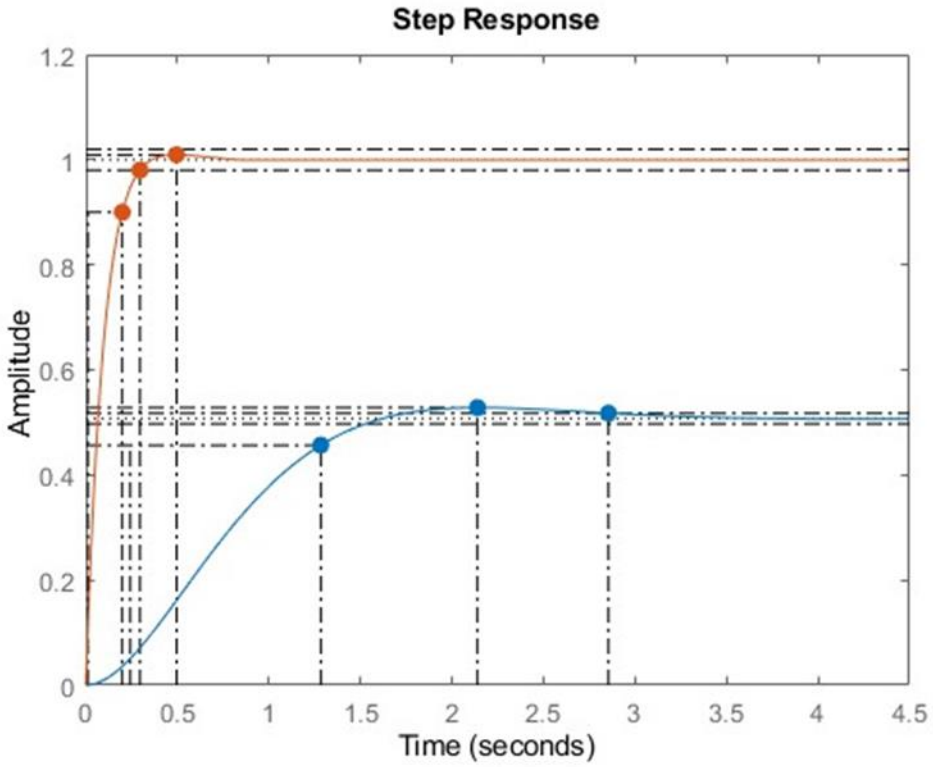


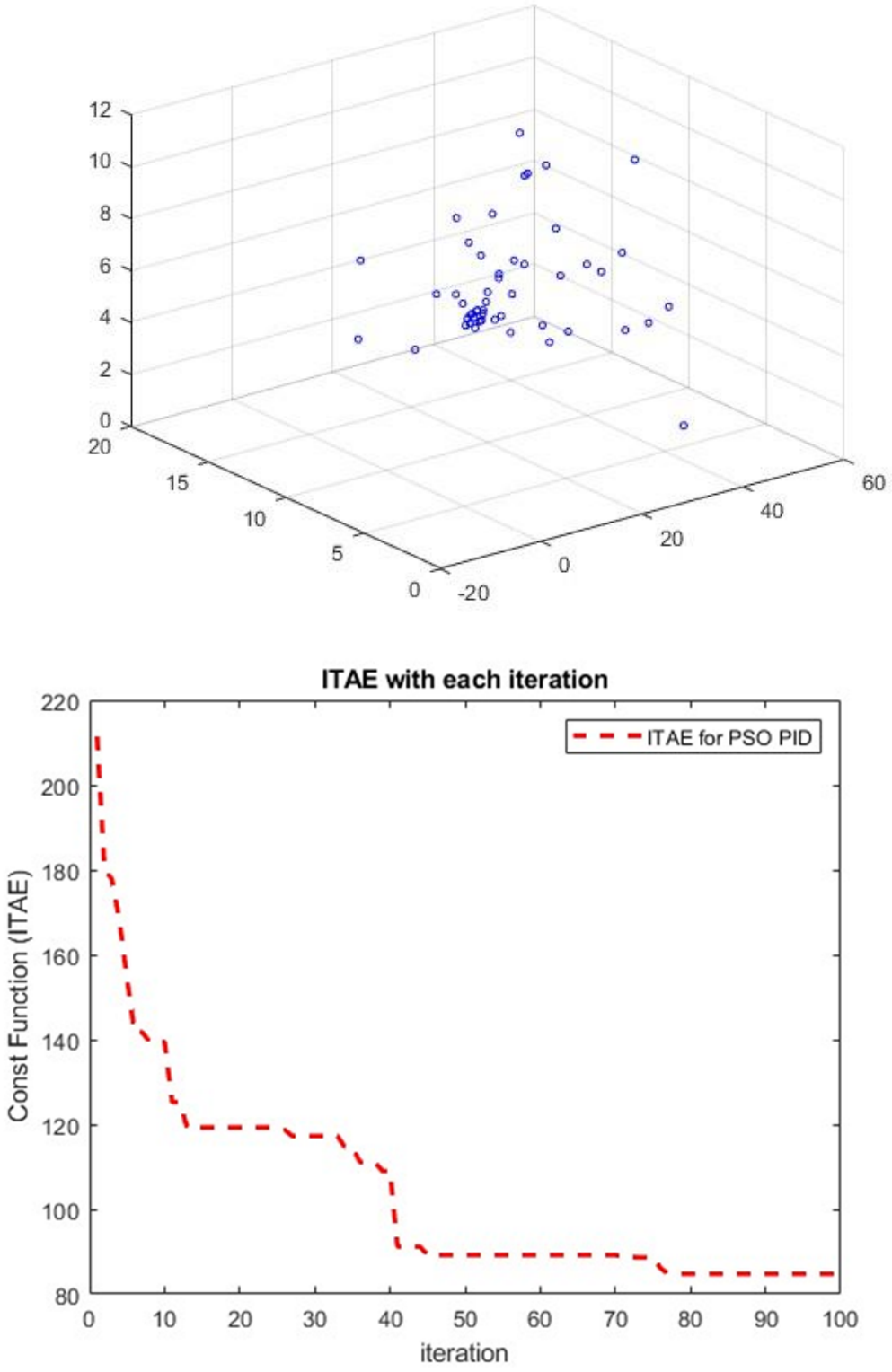


Gambar 4-3 Respon sistem pengaturan flow dituning dengan parameter w=0.4

4.3.2 Pengujian Variasi w=0.8

Gambar 4.4 merupakan pengujian dengan variabel w = 0.8 dan dengan parameter N=50, c1=1.5, c2=1 dan iterasi maksimal sebesar 100 kali iterasi dan luasan pencarian antara 0 sampai dengan 100. Didapatkan hasil plot grafik step respons, fungsi global per iteration sampai didapatkan nilai minimum *Integral Time Absolute Error*, serta plot swarm position dimana berisi variable Kp, Ki dan Kd yang paling optimal menurut algoritma PSO. Adapun data karakteristik system sebelum dan sesudah di tuning dapat dilihat pada tabel 4.2. Pada percobaan ini diperoleh nilai cost function terbaik yakni minimum ITAE sebesar 84.7756 dari *cost function* 300.4918. Pada pengujian ini diperoleh nilai gain optimal Kp 16.7649, Ki 10.0828 dan, Kd 4.7702.



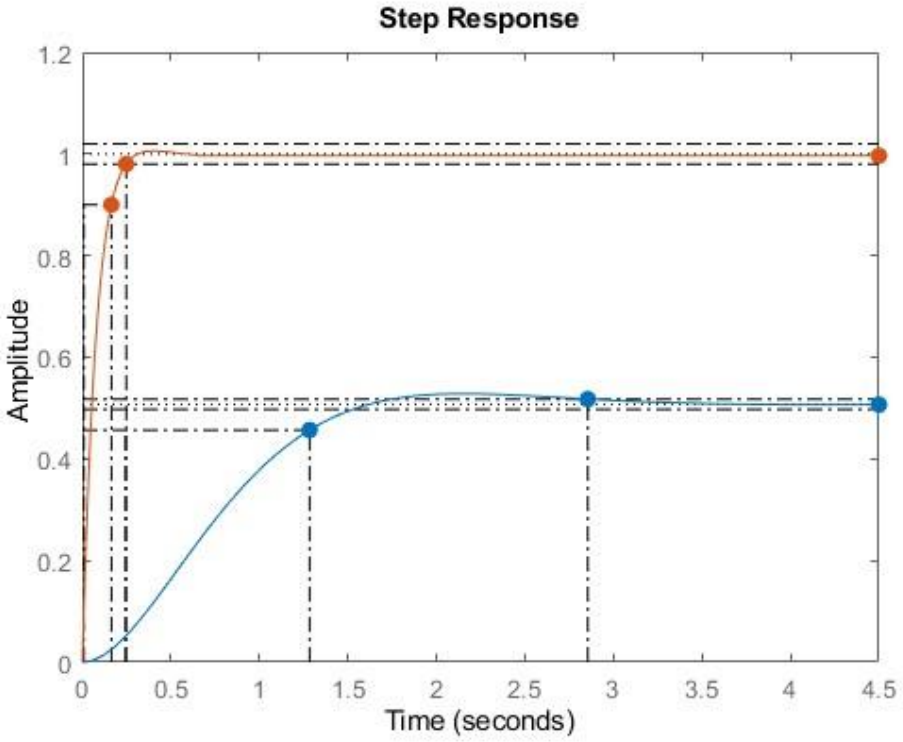


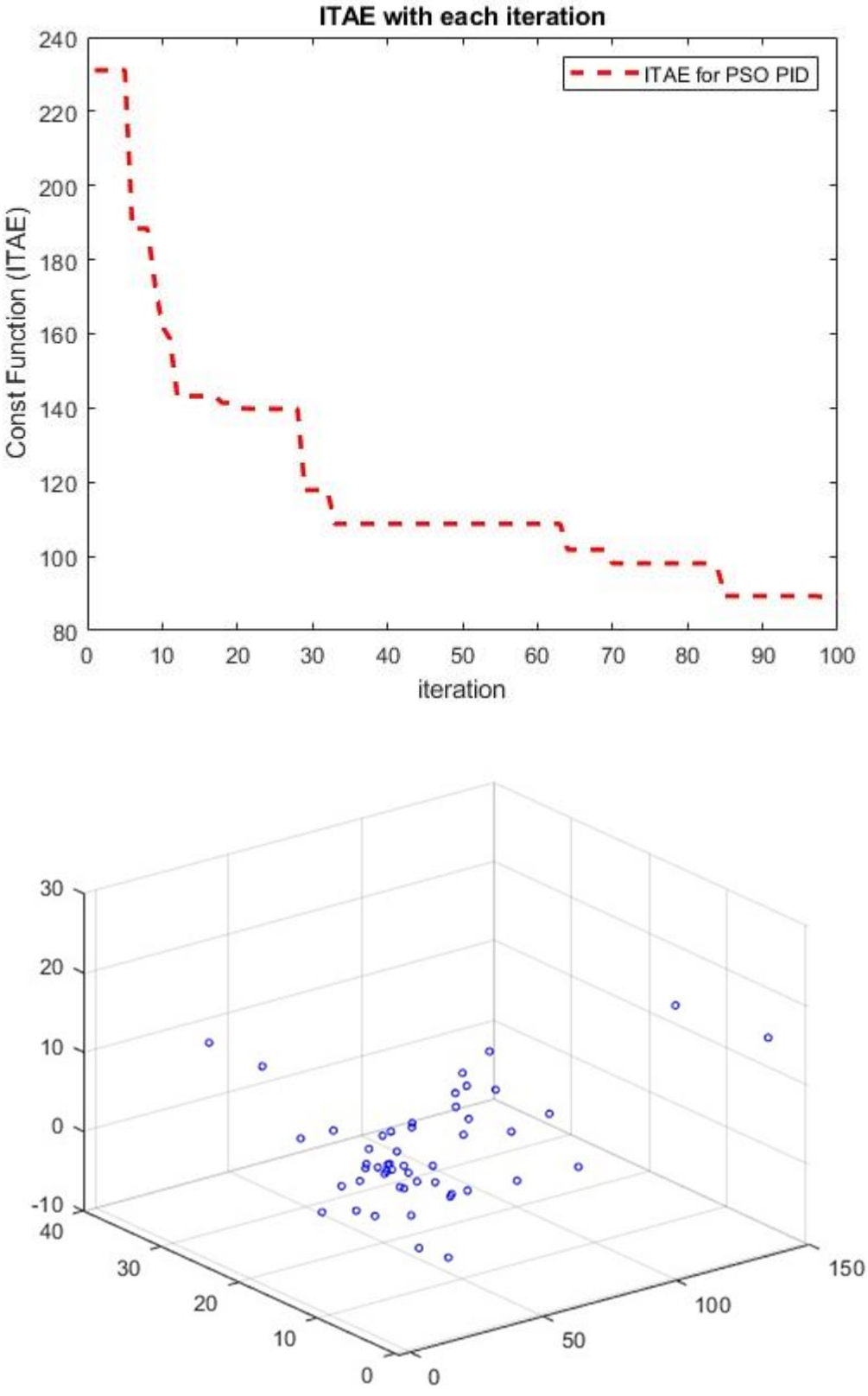
Gambar 4-4 Respon sistem pengaturan flow setelah dituning dengan w = 0.8

4.3.3 Pengujian Variasi w = 0.9

Pada pengujian ini dilakukan variasi variabel dengan w = 0.9, N = 50, c1 = 1.5, c2 = 1 dan jumlah maksimal iterasi yang dilakukan adalah 100 kali. Area pencarian yang digunakan adalah antara 0 hingga 100. Hasil dari pengujian ini menunjukkan grafik respons, fungsi keseluruhan dengan iterasi yang mencapai nilai kesalahan absolut minimum selama waktu integral, dan plot

posisi dari "swarm" yang berisi variabel Kp, Ki, dan Kd yang paling optimal menurut Algoritma PSO Pada Gambar 4.5. Data yang biasa dari sistem sebelum dan setelah penyetelan ditampilkan pada Tabel 4.2. Nilai yang terbaik dari cost function, ITAE minimum, adalah 89.0250 dari nilai cost function yaitu 306.9654. Nilai gain optimal yang diperoleh dari pengujian ini adalah Kp 20.8441, Ki 10.0045, dan Kd 5.8969.

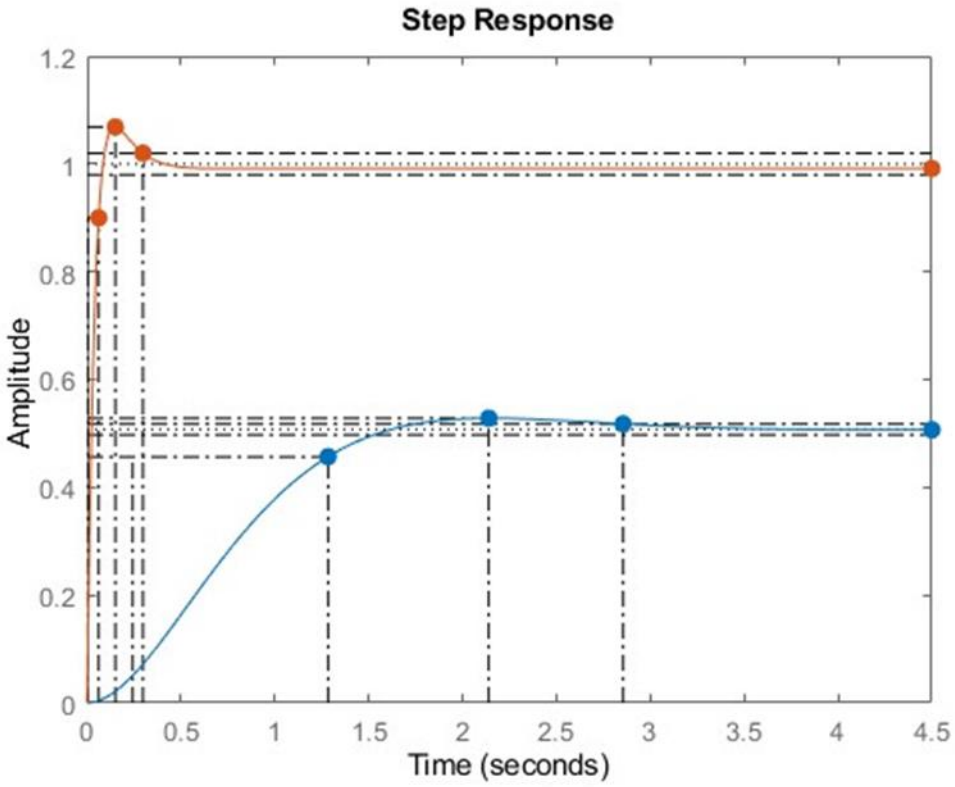


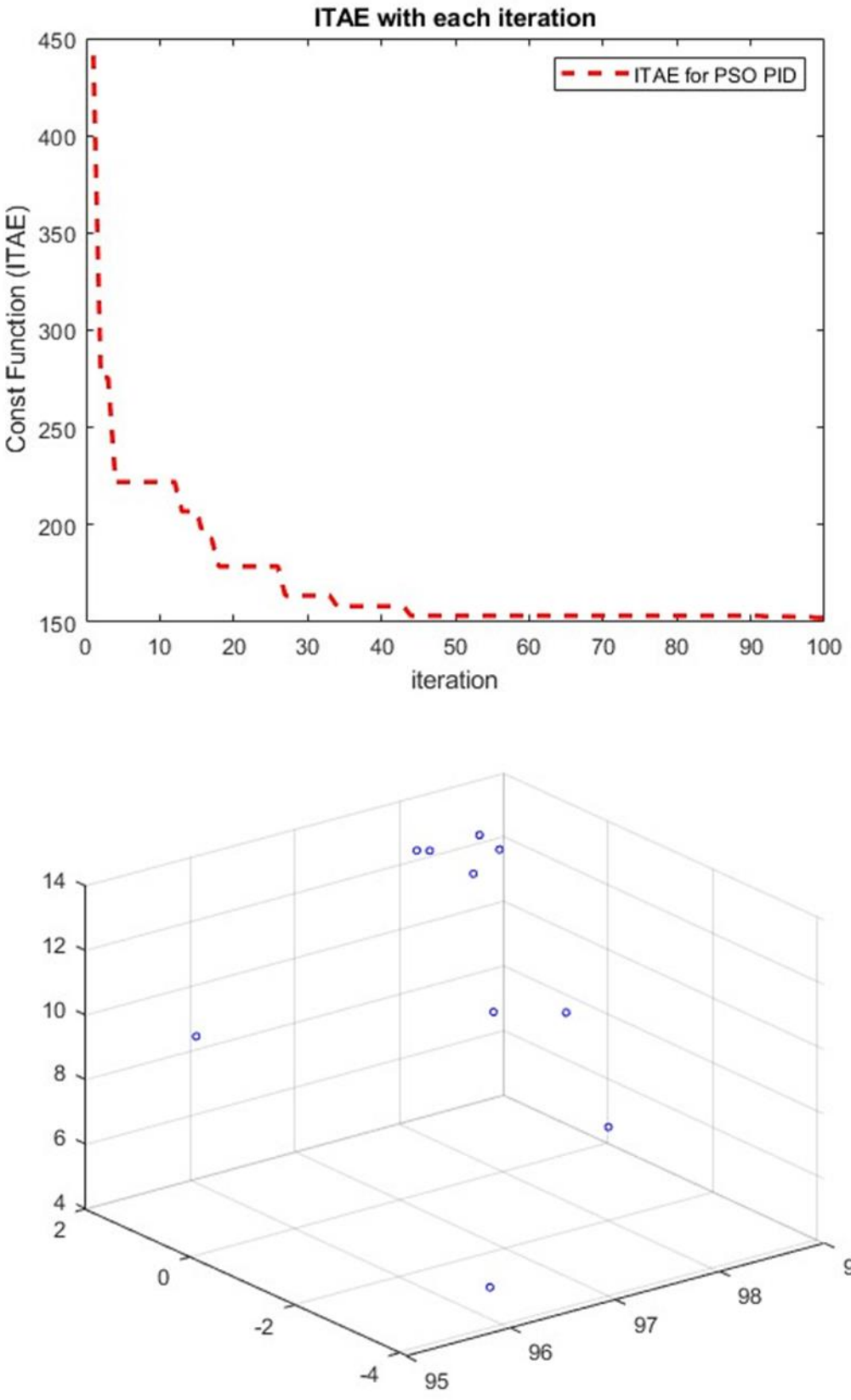


Gambar 4-5 Respon sistem pengaturan flow setelah dituning dengan w=0.9

4.3.4 Pengujian Jumlah Partikel (N) = 10

Pada gambar 4.6 merupakan hasil plot dari percobaan pengujian variasi pada variabel jumlah partikel (N)=10, parameter yang digunakan pada pengujian ini adalah w=0.9, c1=1.5, c2=1 serta iterasi maksimal 100 kali dan luas pencarian adalah 0 sampai dengan 100. Pengujian ini memperoleh hasil nilai cost function terbaik yakni minimum ITAE sebesar 152.1769 dari *cost function* 441.1213. Pada pengujian ini diperoleh nilai gain optimal Kp 97.8894, Ki 0.4766 dan, Kd 12.8622.

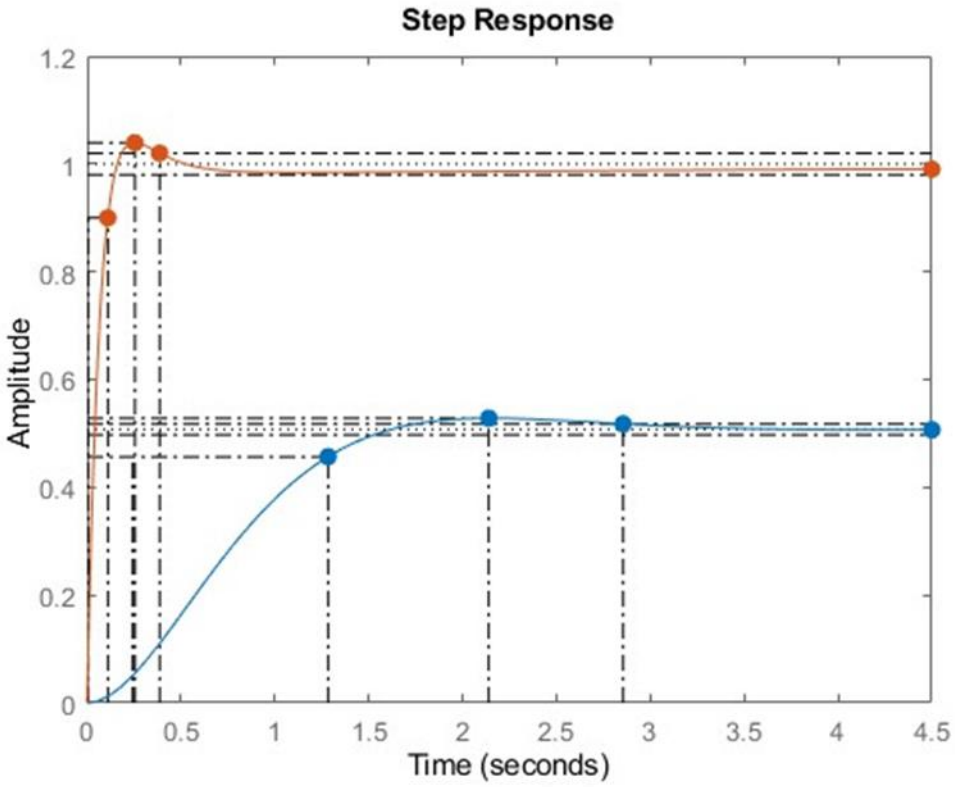


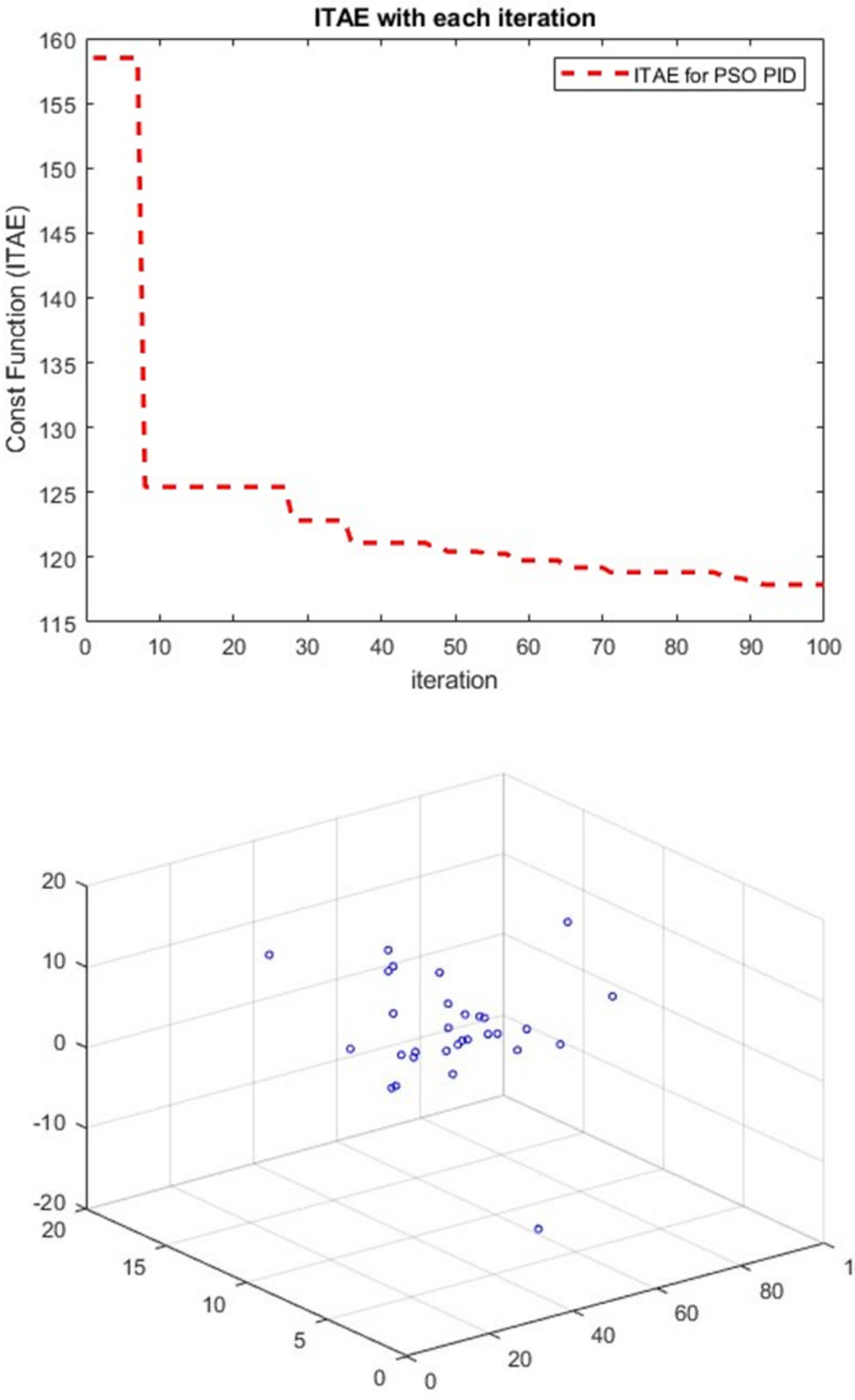


Gambar 4-6 Respon sistem pengaturan flow setelah dituning dengan N=10

4.3.5 Pengujian Jumlah Partikel (N)=30

Pengujian ini menggunakan algoritma Particle Swarm Optimization (PSO) untuk menyetel parameter dari suatu sistem kontrol. Dalam penelitian ini, penulis menggunakan varian PSO yang disebut "Global Best PSO", dengan jumlah partikel yang tetap (N = 30), dan nilai yang spesifik untuk parameter w (0.9), c1 (1.5), dan c2 (1). Algoritma dijalankan selama maksimum 100 iterasi, dan ruang pencarian untuk parameter adalah antara 0 dan 100. Algoritma berhasil menemukan nilai optimal Kp=39.1889, Ki=6.7229, dan Kd=7.6187.

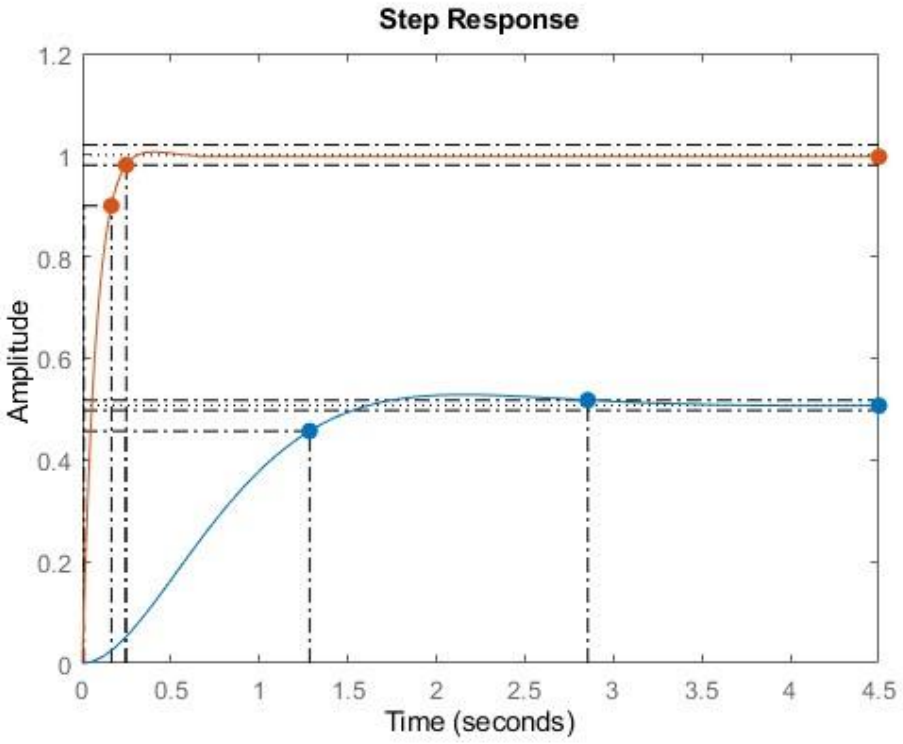


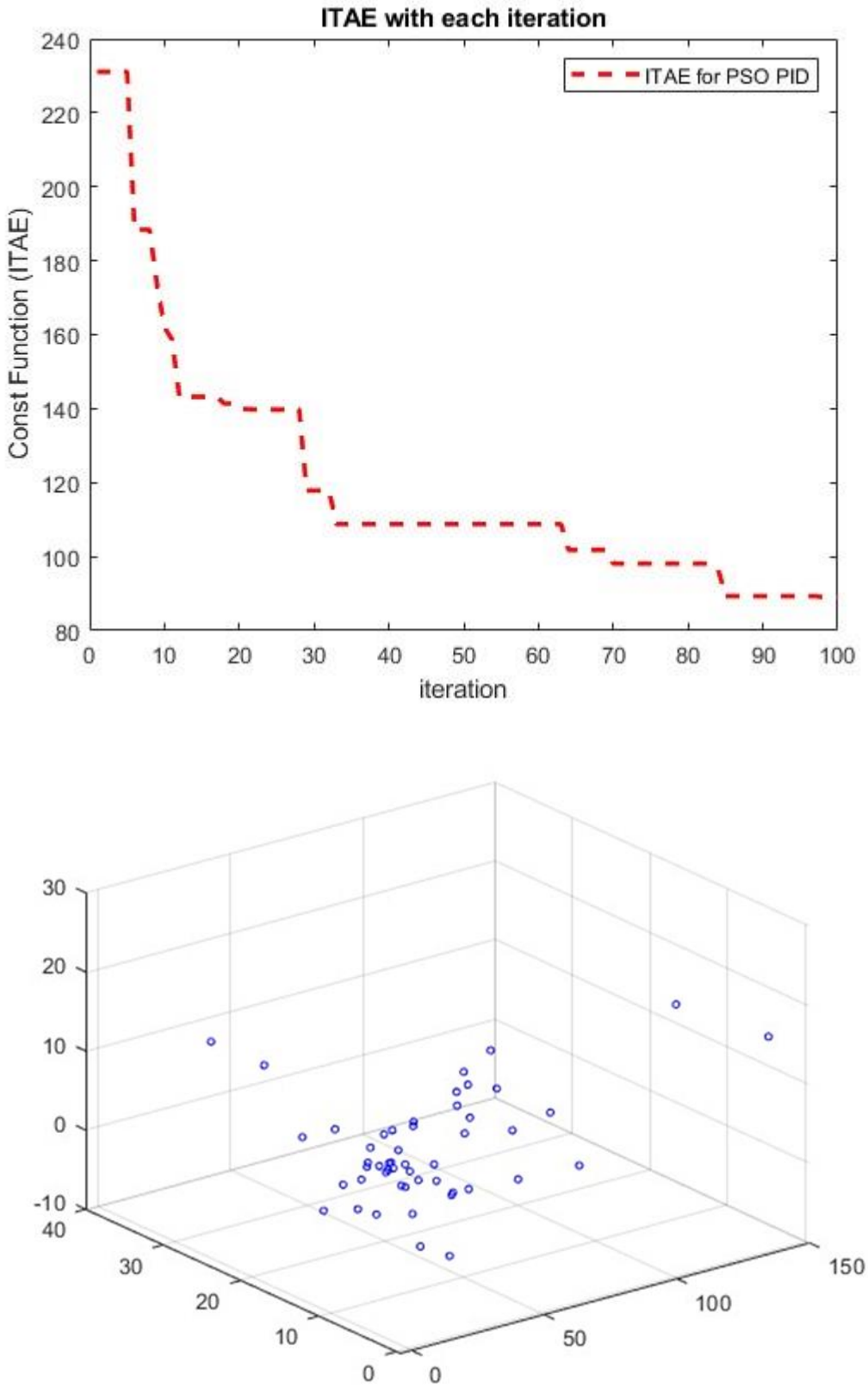


Gambar 4-7 Respon sistem pengaturan flow setelah dituning dengan N=30

4.3.6 Pengujian Jumlah Partikel (N)=50

Pada pengujian ini dilakukan variasi variable Jumlah Partiel (N)=50 dengan parameter w=0.9, c1=1.5, c2=1 dan iterasi maksimal sebesar 100 kali iterasi dan luasan pencarian antara 0 sampai dengan 100. Didapatkan hasil plot grafik step response, function global per iteration sampai didapatkan nilai minimum Integral Time Absolute Error, serta plot swarm position dimana berisi variable Kp, Ki dan Kd yang paling optimal menurut algoritma PSO. Adapun data karakteristik system sebelum dan sesudah di tuning dapat dilihat pada tabel 4.3. Pada percobaan ini diperoleh nilai cost function terbaik yakni minimum ITAE sebesar 89.0250 dari cost function 306.9654. Pada pengujian ini diperoleh nilai gain optimal Kp 20.8441, Ki 10.0045 dan, Kd 5.8969.





Gambar 4-8 Respon sistem pengaturan flow setelah dituning dengan N=50

4.3.7 Pembahasan

Berikut adalah tabel hasil tuning PID dengan metode Particle Swarm Optimization dengan pengujian variasi bobot inersia (w). Data disajikan dalam bentuk tabel 4.2, sedangkan pada tabel 4.3 merupakan hasil tuning PID menggunakan variasi jumlah partikel (N) pada algoritma PSO.

Tabel 4-2 Hasil Tuning variasi w

ITEM Rata rata		w = 0.4	w = 0.8	w = 0.9
Parameter PID	Kp	39.9480	16.7649	20.8441
	Ki	4.2363	10.0828	10.0045
	Kd	7.8599	4.7702	5.8969
Performansi PSO	Cost Function	314.1645	300.4918	306.9654
	Fitness Value (min ITAE)	118.9098	84.7756	89.0250
Spesifikasi Performansi Dinamik	Overshoot	3.5957	0.9355	0.5769
	Rise Time	0.1019	0.1890	0.1554
	Settling time	0.3639	0.2954	0.2473
	Error Steady State	0.0360	0.0094	0.0058

Bobot Inersia digunakan untuk meredam kecepatan partikel ketika iterasi sehingga memungkinkan partikel menuju konvergensi ke titik target secara akurat dan efisien. Nilai bobot inersia yang tinggi digunakan untuk mencari nilai global explorasi, sedangkan nilai yang rendah digunakan untuk menekan pencarian nilai lokal. Dari tabel tersebut, terlihat bahwa nilai Kp, Ki, dan Kd berbeda pada setiap nilai w. Nilai w sendiri merupakan bobot yang digunakan dalam algoritma Particle Swarm Optimization (PSO). Pada setiap nilai w yang berbeda, akan terjadi perbedaan pada nilai Kp, Ki, dan Kd yang dihasilkan oleh PSO.

Performansi PSO dapat dilihat dari nilai *Cost Function* dan *Fitness Value*. Nilai *Cost Function* menunjukkan seberapa baik algoritma PSO dalam menemukan solusi yang optimal, sedangkan nilai *Fitness Value* menunjukkan seberapa baik hasil yang diperoleh dari algoritma PSO dibandingkan dengan metode lain. Dari tabel tersebut, terlihat bahwa nilai Cost Function bervariasi terhadap tiap tiap nilai w namun dari hasil run program selalu mencari nilai terbaik yakni minimize *Integral Time Absolute Error*, terlihat bahwa setiap pertambahan nilai w nilai Fitness value menurun, kecuali pada data w=0.9 fitness value tidak menjadi optimal namun hal tersebut tidak mengapa karena pada bobot w=0.9 menghasilkan sistem tuning dengan spesifikasi performansi dinamik yang paling baik dibandingkan dengan nilai w yang lain yakni w=0.4 dan w=0.8. Dari uraian tersebut nilai w yang semakin bertambah menghasilkan *fitness value* yang paling optimum.

Spesifikasi Performansi Dinamik menunjukkan bagaimana sistem berperilaku setelah diberikan suatu sinyal masukan. Nilai *Overshoot* menunjukkan seberapa jauh sistem melebihi nilai set poin, *Rise Time* menunjukkan berapa lama sistem mencapai nilai set poin setelah diberikan sinyal masukan, *Settling Time* menunjukkan berapa lama sistem mencapai kestabilan setelah diberikan sinyal masukan, dan *Error Steady State* menunjukkan seberapa dekat sistem dengan nilai set poin setelah mencapai kestabilan. Dari tabel tersebut, terlihat bahwa semua nilai spesifikasi performansi dinamik menurun pada setiap peningkatan nilai w. Hal ini menunjukkan bahwa sistem menjadi lebih stabil dan mendekati nilai set poin pada setiap peningkatan nilai w. Dari semua nilai w dipilih nilai w=0.9 dikarenakan menghasilkan

performansi dinamik yang paling optimal dimana sistem yang dituning dengan nilai w=0.9 memiliki nilai *overshoot, settling time, rise time* dan *error steady state* yang paling kecil di antara nilai w yang lain yakni w=0.4 dan w=0.8.

Tabel 4-3 Hasil Tuning variasi N

ITEM Rata rata		N = 10	N = 30	N = 50
Parameter PID	Kp	97.8894	39.1889	20.8441
	Ki	0.4766	6.7229	10.0045
	Kd	12.8622	7.6187	5.8969
Performansi PSO	*Cost Function*	441.1213	209.6887	306.9654
	Fitness Value (min ITAE)	152.1769	117.8835	89.0250
Spesifikasi Performansi Dinamik	*Overshoot*	6.8550	3.9259	0.5769
	Rise Time	0.0580	0.1040	0.1554
	Settling time	0.2969	0.3862	0.2473
	Error Steady State	0.0686	0.0393	0.0058

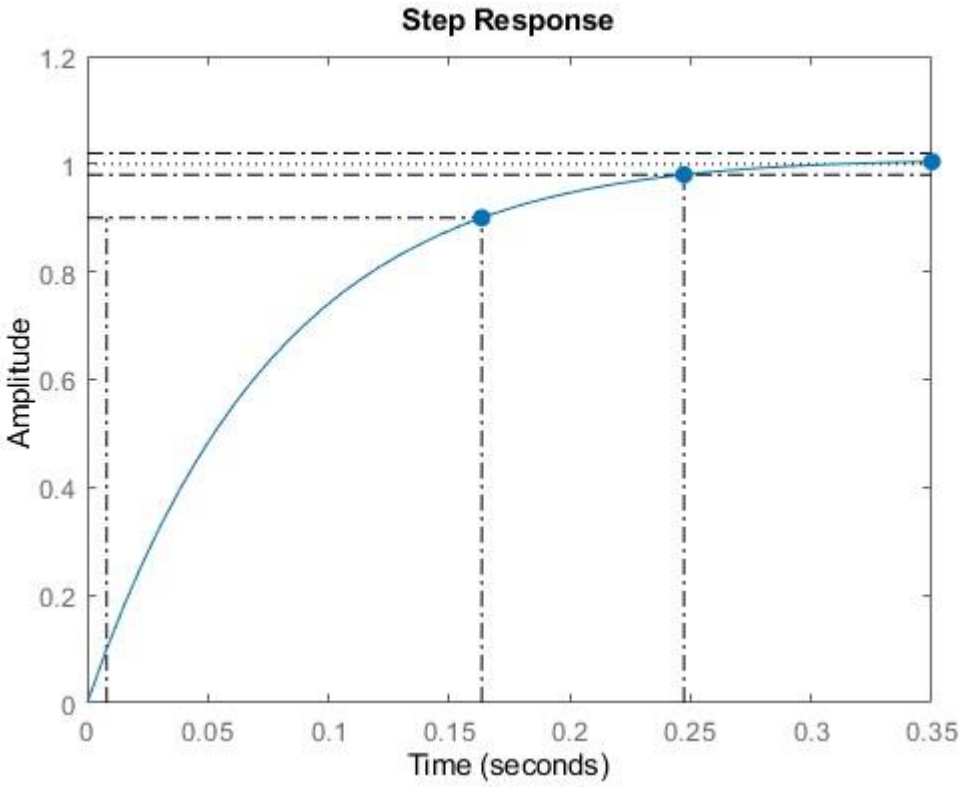
Pada tabel di atas menunjukkan setiap peningkatan nilai N menghasilkan parameter Kp, Ki, Kd yang berbeda-beda bergantung pada parameter PSO yang digunakan untuk tuning dan nilai optimum dari *Cost Function* serta *fitness value*. Percobaan di atas menggunakan parameter w=0.9, c1=1.5, c2=1 dan maksimum iterasi 100 kali serta ruang pencarian antara 0 sampai dengan 100.

Pada performansi PSO bertambahnya nilai N menunjukkan bahwa *fitness value* semakin mengecil. Hal ini menunjukkan bahwa nilai N atau jumlah partikel yang semakin besar menunjukkan performansi PSO yang optimal. Namun hal tersebut juga bergantung dengan parameter yang dimasukkan seperti nilai bobot inersia atau w.

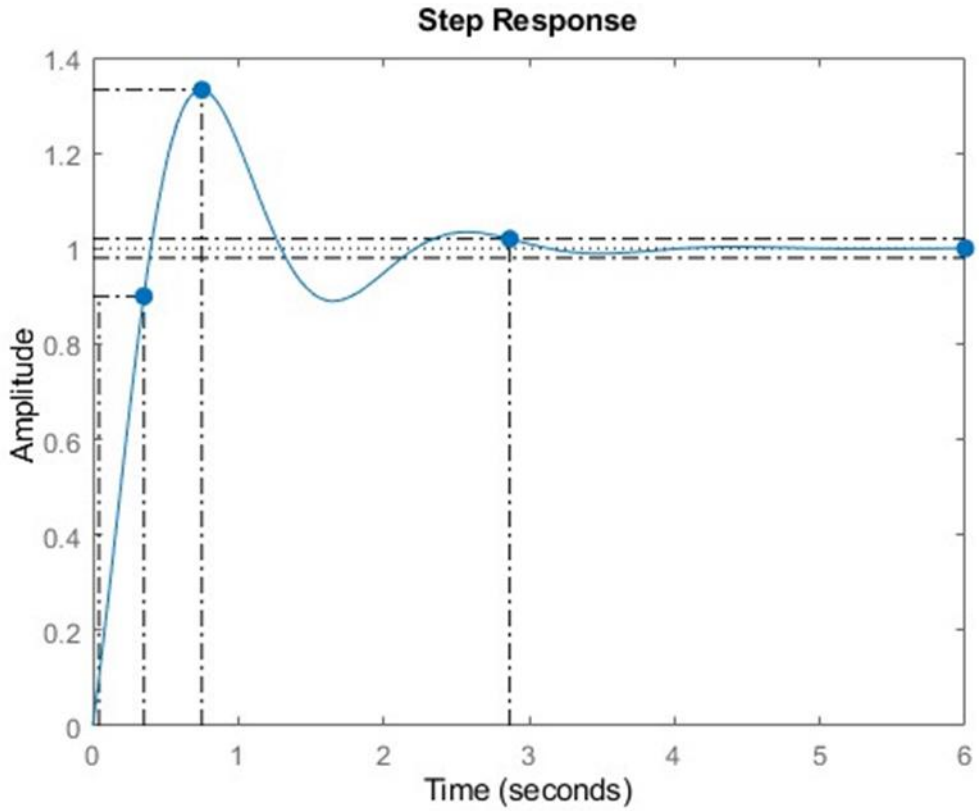
Semakin banyak jumlah partikel dapat meningkatkan performansi dinamik dimana bertambahnya nilai partikel menyebabkan nilai overshoot mengecil serta nilai *steady state error* yang kecil juga meskipun *rise time* dan *settling time* bervariasi. Hal ini dikarenakan peningkatan partikel semakin lama dalam pencarian nilai objektif akibat setiap partikel melakukan iterasi secara terus menerus sampai nilai minimum fungsi objektif diinginkan. Peningkatan jumlah partikel menyebabkan semakin banyak titik titik area yang dijajaki sehingga semakin meningkatkan peluang dalam menentukan titik optimal dari seluruh area yang dilewati. Sedangkan jika jumlah partikel semakin kecil akan menyebabkan konvergensi menjadi premature karena banyak titik titik kosong yang tidak dijajaki, sehingga tidak jarang hanya menemukan nilai lokal minimum. Dari percobaan tersebut jumlah partikel terbaik yang akan digunakan dalam pengujian ini adalah 50 karena menghasilkan performasin system yang terbaik.

4.3.8 Perbandingan Tuning PSO-PID dengan Ziegler Nichols pada Inner Loop Flow

Berikut adalah perbandingan hasil tuning PID menggunakan Ziegler Nichols dan Menggunakan Tuning PSO pada inner loop yakni kontrol aliran. Pada gambar di bawah merupakan hasil step respons parameter Kp, Ki, Kd yang dimasukkan ke dalam persamaan fungsi transfer *flow plant*.



Gambar 4-9 Hasil Respon Sistem Pengaturan Flow Tuning Dengan PSO-PID



Gambar 4-10 Hasil Respon Sistem Pengaturan Flow Tuning Dengan Ziegler Nichols

Tabel 4-4 Perbandingan PSO PID dengan Ziegler Nichols pada Sistem Pengaturan Flow

Tuning PID	Parameter PID			Indeks Perfomansi Pengendalian				
	Kp	Ki	Kd	ITAE	ISE	Max Overshoot (%)	Rise Time	Settling Time
PSO-PID	20.8441	10.0045	5.8969	0.6534	0.2022	0.5769	0.1554	0.2473
Ziegler Nichols	8.1553	16.3107	1.0194	0.4211	0.2984	33.2133	0.3095	2.8674

Dari tabel tersebut, terlihat bahwa nilai Kp, Ki, dan Kd berbeda pada metode PSO-PID dan metode Ziegler Nichols. Metode PSO-PID menggunakan algoritma *Particle Swarm Optimization* (PSO) untuk menentukan nilai Kp, Ki, dan Kd, sedangkan metode Ziegler Nichols menggunakan rumus matematis tertentu untuk menentukan nilai Kp, Ki, dan Kd. Nilai PID yang digunakan pada Particle Swarm Optimization adalah nilai dari hasil tuning parameter PSO pada N=50 dan w=0.9 pada persamaan *transfer function flow plant* yang dituning dengan PSO. Sedangkan nilai PID Ziegler Nichols diperoleh dari perhitungan matematis sistem open-loop pada persamaan transfer *function flow plant*.

Berdasarkan tabel, Indeks performansi pengendalian mengukur sejauh mana metode dapat mengendalikan sistem. Dari tabel, metode PSO-PID memiliki nilai ITAE dan ISE yang lebih kecil dibandingkan metode Ziegler Nichols, menunjukkan bahwa metode PSO-PID lebih baik dalam mengendalikan sistem.

Nilai *Max Overshoot* menunjukkan sejauh mana sistem melampaui nilai set poin setelah diterima sinyal masukan. Nilai *Max Overshoot* lebih kecil pada metode PSO-PID dibandingkan metode Ziegler Nichols, menunjukkan bahwa sistem lebih stabil pada metode PSO-PID.

Rise Time menunjukkan berapa lama sistem mencapai nilai set poin setelah menerima sinyal masukan, dan *Settling Time* adalah waktu sistem mencapai kestabilan setelah menerima sinyal masukan. Kedua nilai ini lebih kecil pada metode PSO-PID dibandingkan metode Ziegler Nichols, menunjukkan bahwa sistem lebih cepat mencapai nilai set poin dan kestabilan pada metode PSO-PID.

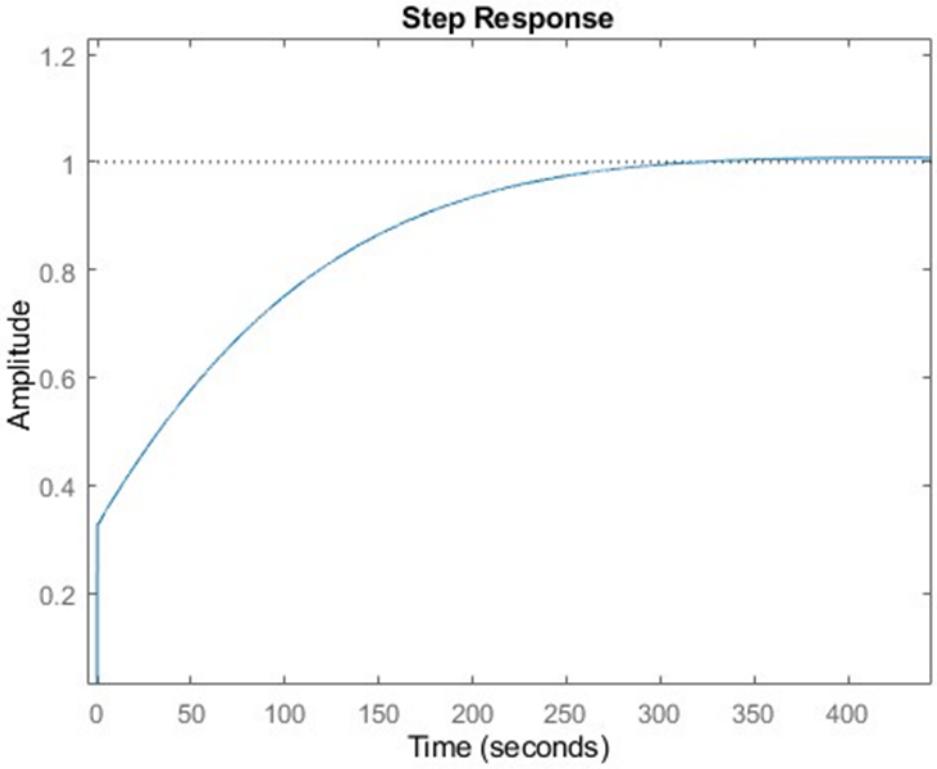
Kesimpulannya, metode PSO-PID lebih baik dalam mengendalikan sistem dibandingkan dengan metode Ziegler Nichols, karena memiliki nilai ITAE dan ISE yang lebih kecil, *Max Overshoot* yang lebih kecil, *Rise Time* yang lebih kecil, dan *Settling Time* yang lebih kecil.

4.4 Hasil Pengujian Tuning PID pada Outer Loop Level Plant

Setelah di tuning pada *inner loop* pada sub bab 4.3 langkah selanjutnya adalah hasil tuning PID menggunakan Ziegler Nichols dibandingkan dengan menggunakan Algoritma PSO diterapkan pada *loop* luar pada Sistem Pengaturan Cascade Level dan Flow

4.4.1 Hasil Tuning Outer Loop Menggunakan Ziegler Nichols

Berikut ini adalah grafik respons sistem yang dituning menggunakan metode Ziegler Nichols ditunjukkan pada gambar 4.11.



Gambar 4-11 Hasil Tuning Outer Loop menggunakan Ziegler Nichols

Gambar 4.11 merupakan hasil tuning *inner loop* dikalikan dengan persamaan *plant* level sehingga terbentuk sistem *open-loop* yang kemudian persamaan tersebut diberikan sinyal step open loop untuk menentukan parameter nilai Kp, Ki, Kd dengan menggunakan Ziegler Nichols Tuning sehingga diperoleh besaran Kp 1.0595, Ki 0.0095 dan Kd 29.6667

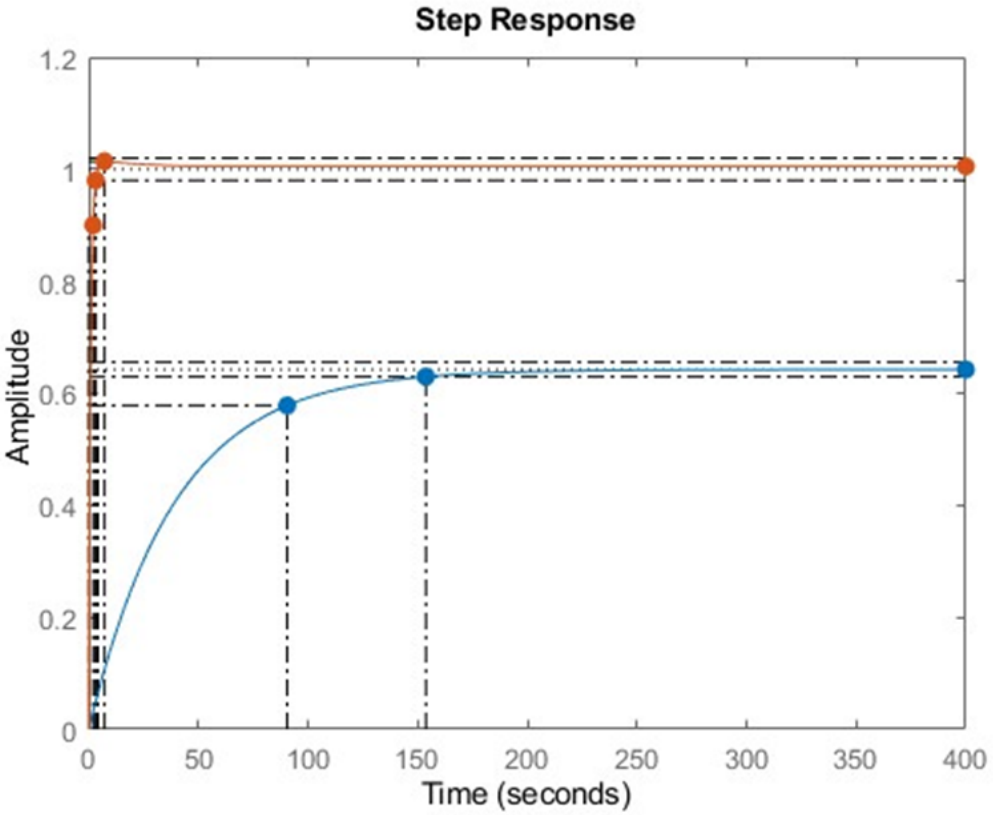
Tabel 4-5 Karakteristik Dinamis Sistem Outer Loop

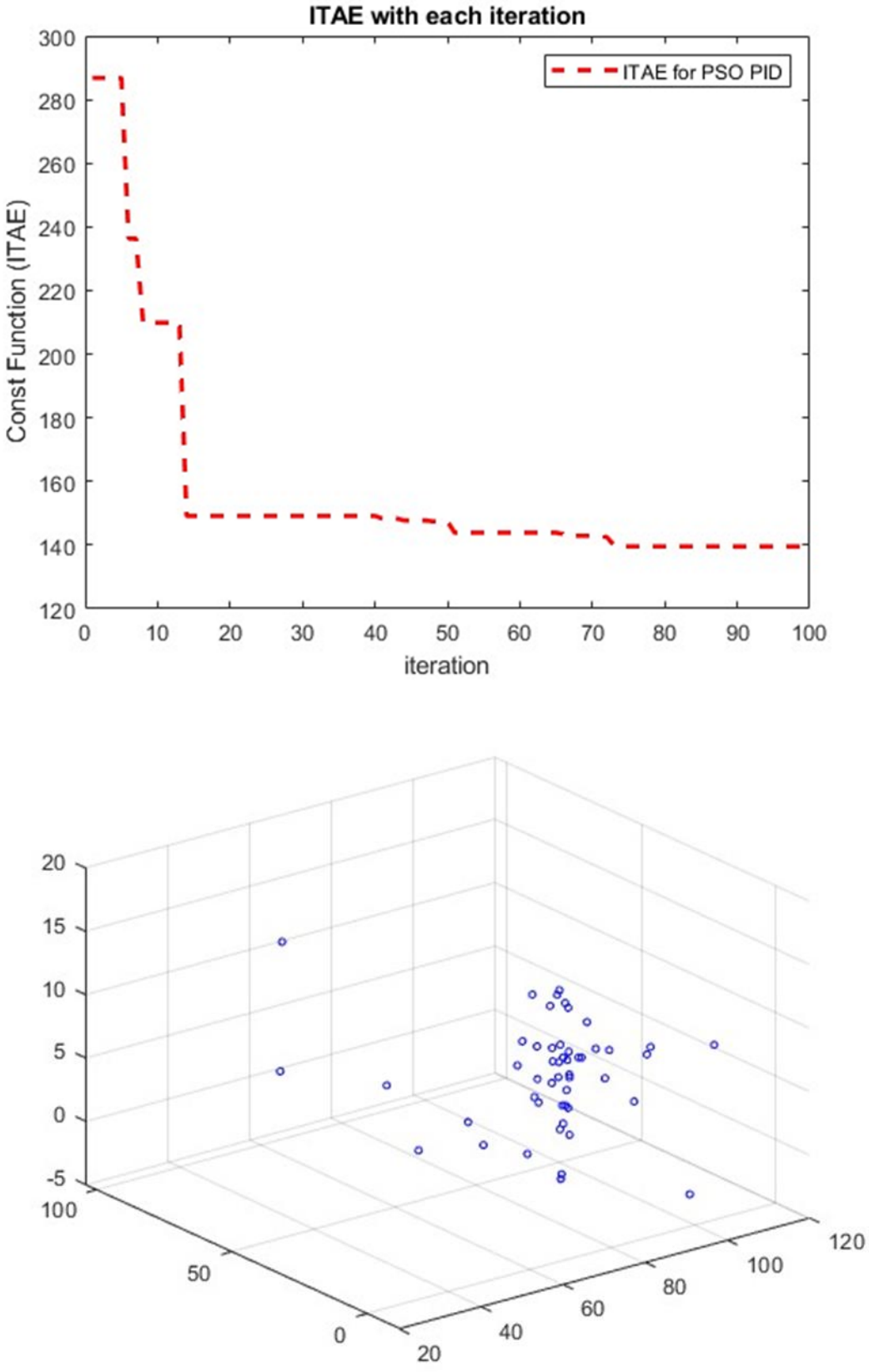
RiseTime	171.7323
Transient Time	260.7723
SettlingTime	260.7723
Overshoot	0.7598
Undershoot	0
Peak	1.0076
Peak Time	424.3901

Dari data di atas diperoleh karakteristik sistem sebagai berikut *Rise Time* sistem adalah 171.7323, *Transient Time* adalah 260.7723, dan *Settling Time* adalah 260.7723. *Overshoot* yang terjadi adalah 0.7598, tidak ada *undershoot* yang terjadi, dan nilai puncak yang tercapai adalah 1.0076 pada waktu puncak 424.3901.

4.4.2 Hasil Tuning Outer Loop Menggunakan Particle Swarm Optimization

Berikut adalah hasil tuning PID *outer loop* menggunakan metode PSO dengan parameter w=0.9, N=50, c1=1.5, c2=1 dan maksimal iterasi adalah 100 serta *search space* 0 sampai dengan 100. Data percobaan dapat dilihat pada gambar berikut.





Gambar 4-12 Hasil tuning Outer loop menggunakan PSO PID

Dari data gambar 4.12 diperoleh Nilai *cost function* adalah 301.6425, nilai Minimum ITAE atau *fitness value* adalah 139.5481, nilai kp adalah 68.4639, nilai ki adalah 1.8237, nilai kd adalah 10.2855, dan nilai *error steady state* adalah 0.0140. Kemudian diperoleh karakteristik sistem setelah di tuning menggunakan PID pada tabel berikut.

Tabel 4-6 Karakteristik respon outer loop setelah dituning dengan PSO-PID

RiseTime	2.0383
Transient Time	3.2569
SettlingTime	3.2569
Overshoot	1.4001
Undershoot	0
Peak	1.0140
Peak Time	7.2045

Dari tabel di atas didapatkan nilai dari *Rise Time* sistem sebesar 2.0383, *Transient Time* adalah 3.2569, dan *Settling Time* sebesar 3.2569. Sistem mengalami *Overshoot* sebesar 1.4001 %, serta tidak terdapat *undershoo*, dan nilai puncak yang tercapai adalah 1.0140 pada waktu puncak 7.2045 detik.

4.4.3 Pembahasan

Dari tabel 4.6, terlihat bahwa nilai Kp, Ki, dan Kd berbeda pada metode PSO-PID dan metode Ziegler Nichols. Metode PSO-PID menggunakan algoritma *Particle Swarm Optimization* (PSO) untuk menentukan nilai Kp, Ki, dan Kd, sedangkan metode Ziegler Nichols menggunakan rumus matematis tertentu untuk menentukan nilai Kp, Ki, dan Kd.

Tabel 4-7 Perbandingan Hasil tuning PSO PID dengan Ziegler Nichols pada outer loop

Tuning PID	**Parameter PID**			**Indeks Perfomansi Pengendalian**				
	Kp	Ki	Kd	ITAE	ISE	Max Overshoot (%)	Rise Time	Settling Time
PSO-PID	68.4639	1.8237	10.2855	1.51	0.5481	1.4001	2.0383	3.2569
Ziegler Nichols	1.0595	0.0095	29.6667	46.26	8.918	0.7598	171.7323	260.7723

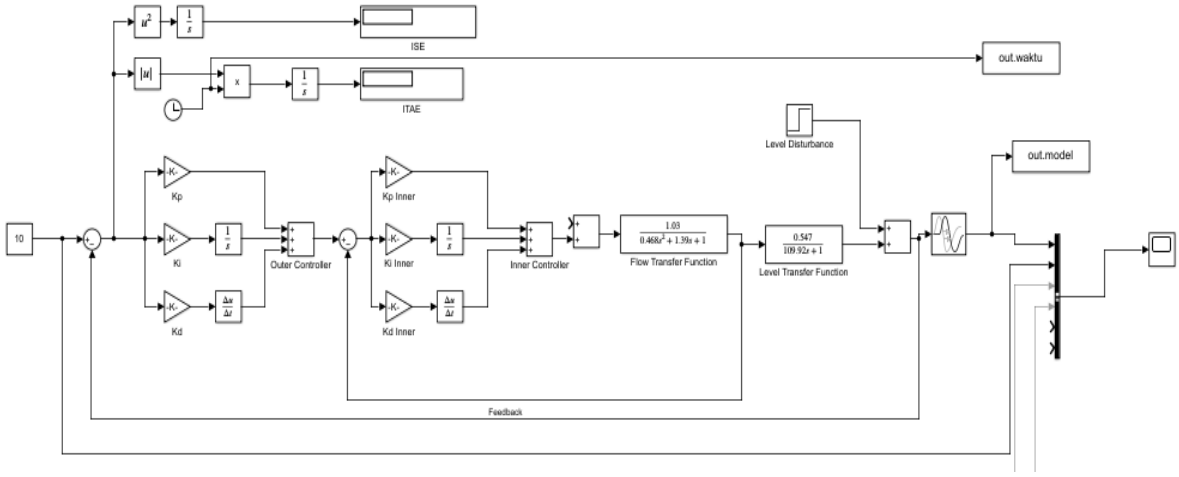
Berdasarkan tabel 4.7 Indeks kinerja kontrol menunjukkan seberapa baik metode mengontrol sistem. Tabel tersebut menunjukkan bahwa nilai ITAE *(Integral Time Absolute Error*) dan ISE (*Integral Squared Error*) lebih kecil untuk metode PSO-PID dibandingkan dengan metode Ziegler-Nichols. Hal ini menunjukkan bahwa metode PSO-PID dapat mengontrol sistem lebih baik daripada metode Ziegler-Nichols.

Nilai *Max Overshoot* merupakan seberapa banyak sistem melampaui setpoint setelah menerima sinyal input. Metode PSO-PID memiliki nilai *Max Overshoot* yang lebih tinggi dibandingkan dengan metode Ziegler-Nichols. Meskipun *overshoot*, skema PSO-PID memiliki kinerja sistem yang lebih baik daripada skema Ziegler-Nichols. Nilai rise time menunjukkan

waktu yang dibutuhkan sistem untuk mencapai set point setelah menerima sinyal input, dan nilai settling time menunjukkan berapa lama waktu yang dibutuhkan sistem untuk stabil setelah menerima sinyal input menunjukkan waktu. Nilai rise time dan settling time lebih kecil untuk metode PSO-PID dibandingkan dengan metode Ziegler-Nichols, menandakan bahwa sistem mencapai setpoint dan stabilitas lebih cepat untuk metode PSO-PID. Singkatnya, meskipun metode PSO-PID memiliki nilai *overshoot* maksimum yang lebih besar, metode ini memiliki nilai ITAE dan ISE yang lebih kecil, waktu naik yang lebih singkat, dan waktu penyelesaian yang lebih singkat, membuat sistem lebih efisien daripada metode Ziegler-Nichols.

4.5 Pengujian Hasil Tuning PSO-PID pada Sistem Cascade Level dan Flow Dengan Variasi Disturbance

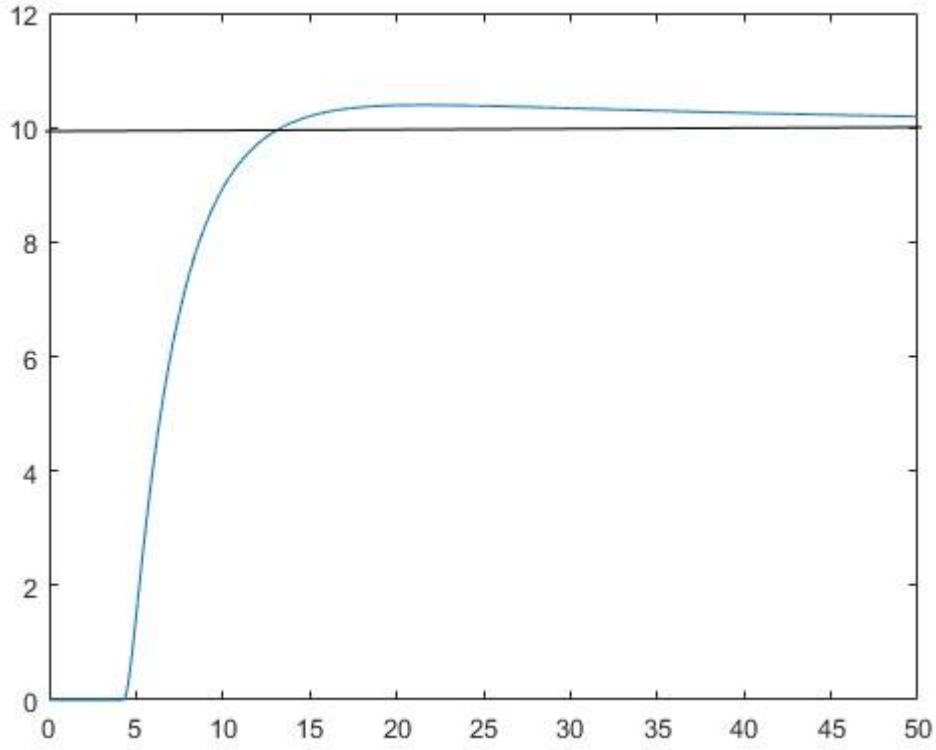
Pada pengujian ini dilakukan variasi level disturbance pada sistem pengaturan cascade level dan *flow* yakni tanpa *disturbance*, *disturbance*=0.5, *disturbanc*e=1, *disturbance*=3. Adapun diagram blok sistem yang digunakan dalam pengujian adalah sebagai berikut.



Gambar 4-13 Block Diagram Simulink Sistem Pengaturan Cascade Level dan Flow

4.5.1 Pengujian dengan Set Point 10 tanpa Disturbance

Berikut ini adalah hasil dari uji coba yang dilakukan dengan nilai set poin 10 cm dan tidak diberikan gangguan Data yang ditampilkan dalam bentuk gambar 4.14.

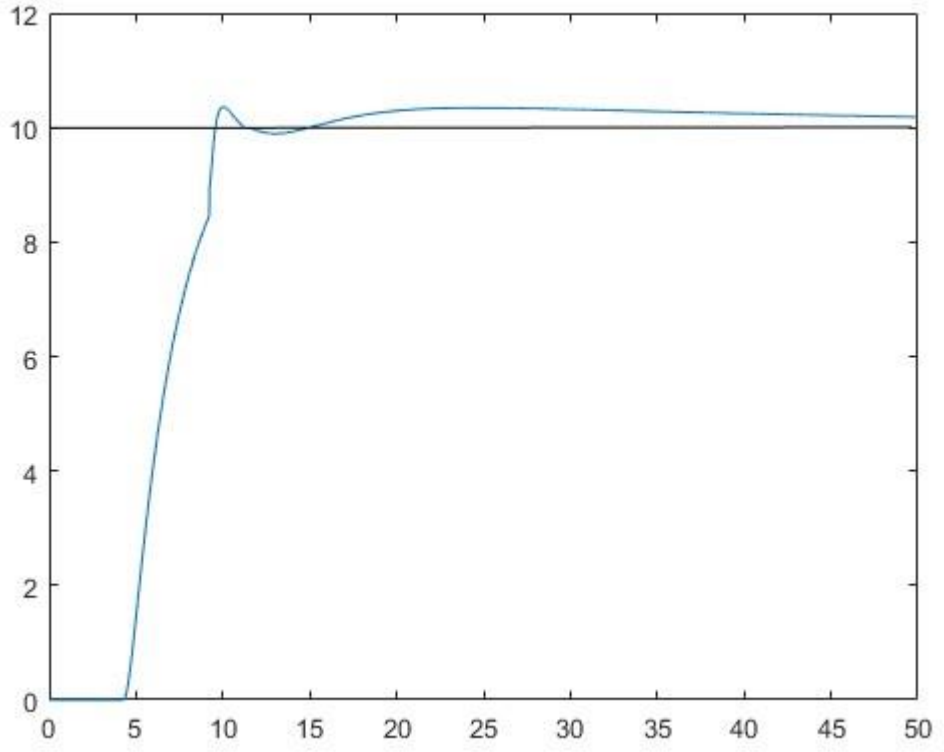


Gambar 4-14 Hasil Pengujian Simulasi tanpa Disturbance

Berdasarkan plot gambar di atas, karakteristik sistem diperoleh sebagai berikut: settling time sebesar 13.3135, rise time sebesar 5.616, dan overshoot sebesar 1.9663. Sistem ini memiliki dua jenis kesalahan, yaitu Integral Square Error sebesar 167.091 dan Integral Time Absolute Error sebesar 376.977.

4.5.2 Pengujian dengan Set Point 10 Dengan Disturbance=0.5

Pengujian pada Sistem Pengaturan Cascade level dan Flow dilakukan dengan besaran set point 10 dan diberikan gangguan berupa set poin 0.5. Berikut disajikan data percobaan berupa grafik respons sistem sebagai berikut pada gambar 4.15

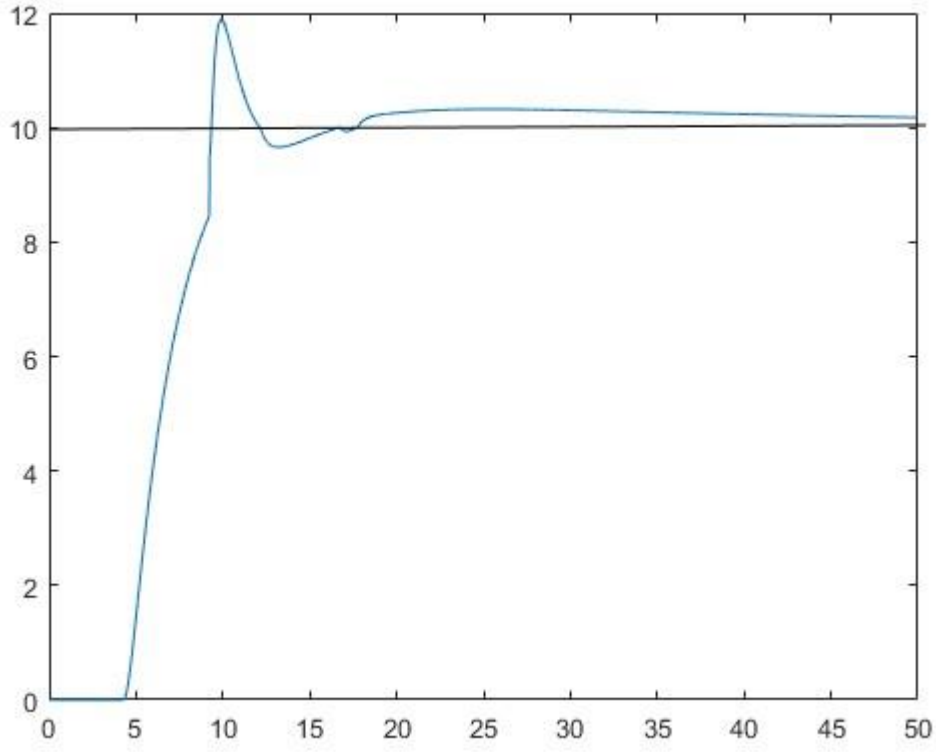


Gambar 4-15 Hasil Pengujian Simulasi dengan Disturbance 0.5

Grafik pada gambar 4.15 menunjukkan karakteristik sistem yakni *settling time* sebesar 14.7483, *Rise time* 4.4593, *Overshoot* 1.7053. *Integral Square Erro*r dan *Integral Time Absolute Error* untuk masing masing besarnya adalah 164.18 dan 337.973.

4.5.3 Pengujian dengan Set Point 10 Dengan Disturbance = 1

Berikut adalah hasil pengujian dengan set point 10 cm dan diberikan gangguan dengan *input* set poin sebesar 1. Adapun data yang disajikan berupa gambar 4.16

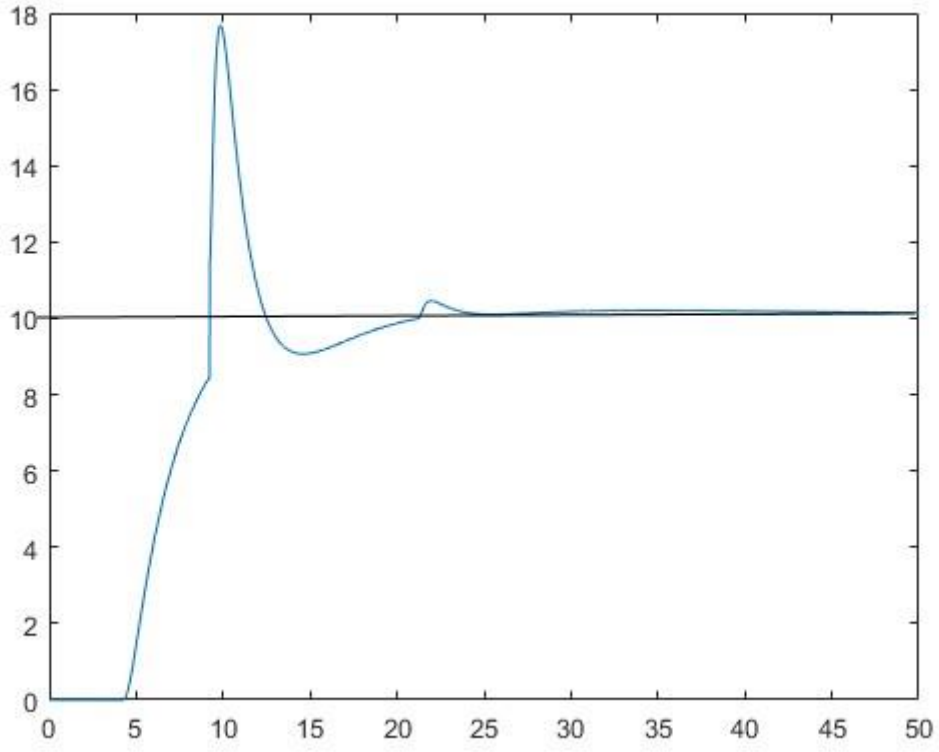


Gambar 4-16 Hasil Pengujian Simulasi dengan Disturbance 1

Dari hasil gambar plot di atas diperoleh karakteristik sistem yakni *settling time* sebesar 17.5337, *Rise time* 4.3777, *Overshoot* 16.7946. Sistem tersebut memiliki error yakni *Integral Square Error* dan *Integral Time Absolute Error* untuk masing-masing besarnya adalah 167.567 untuk ISE dan 338.055 untuk ITAE.

4.5.4 Pengujian dengan Set Point 10 Dengan Disturbance = 3

Hasil pengujian pada sistem pengaturan Cascade level dan Flow yang dilakukan dengan set poin 10 dan gangguan set point 3. Data yang ditampilkan dalam bentuk grafik respons sistem pada gambar 4.17



Gambar 4-17 Hasil Pengujian Simulasi dengan Disturbance 3

Gambar 4.17 menjelaskan bahwa sistem tersebut memiliki settling time sebesar 22.6438, *Rise time* 4.3741, Overshoot 74.1754. Error yang digunakan untuk mengukur performansi system adalah Integral Square Error dan Integral Time Absolute Error untuk masing masing besarnya adalah 231.957 untuk ISE dan 358.487 untuk ITAE.

4.5.5 Pembahasan Pengujian Pengujian Hasil Tuning PSO-PID pada Sistem Cascade Level dan Flow Dengan Variasi Disturbance

Berikut adalah data perbandingan pengujian hasil tuning PSO-PID Sistem Cascade Level dan Flow dengan disturbance atau gangguan. Data tersebut dapat dilihat pada tabel 4.8.

Tabel 4-8 Perbandingan Performansi Sistem Tiap Disturbance

Performansi Dinamik Sistem	No Disturbance	Disturbance=0.5	Disturbance=1	Disturbance=3
Settling Time	13.3135	14.7483	17.5337	22.6438
Rise Time	5.6161	4.4593	4.3777	4.3741
Overshoot	1.9663	1.7053	16.7946	74.1754
ISE	167.091	164.18	167.567	231.957
ITAE	376.977	337.973	338.055	358.487

Berdasarkan data yang disajikan di atas, dapat dilihat bahwa sistem kontrol yang digunakan memiliki performa dinamik yang bervariasi tergantung pada tingkat gangguan yang terjadi. Pada saat tidak ada gangguan, sistem kontrol memiliki settling time yang lebih rendah dibandingkan dengan kondisi gangguan yang lain. Rise time juga lebih rendah pada saat tidak ada gangguan, namun meningkat pada saat gangguan tingkat 1 dan 3.

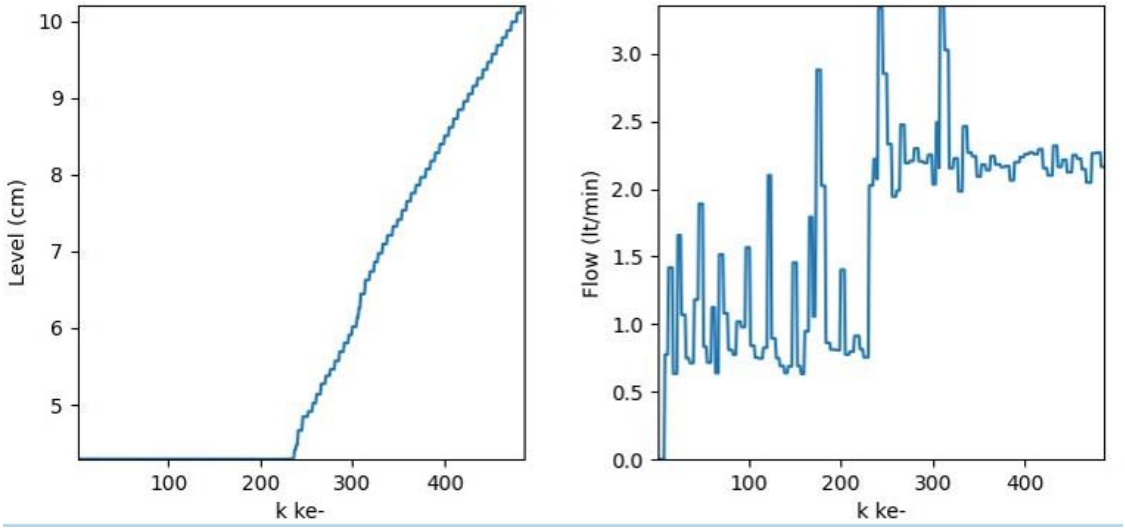
Overshoot juga meningkat secara signifikan pada saat gangguan tingkat 3, yang menunjukkan bahwa sistem kontrol kurang efektif dalam mengendalikan output pada saat gangguan yang tinggi.

Metrik ISE dan ITAE juga menunjukkan bahwa sistem kontrol lebih baik pada saat tidak ada gangguan, namun mengalami peningkatan pada saat gangguan tingkat 1 dan 3.

Berdasarkan analisis ini, dapat disimpulkan bahwa sistem kontrol lebih efektif dalam mengendalikan output pada saat tidak ada gangguan, namun sistem dapat menanggulangi gangguan yang ada sehingga dapat mencapai steady state Kembali dan menjaga sistem sesuai dengan set pointnya.

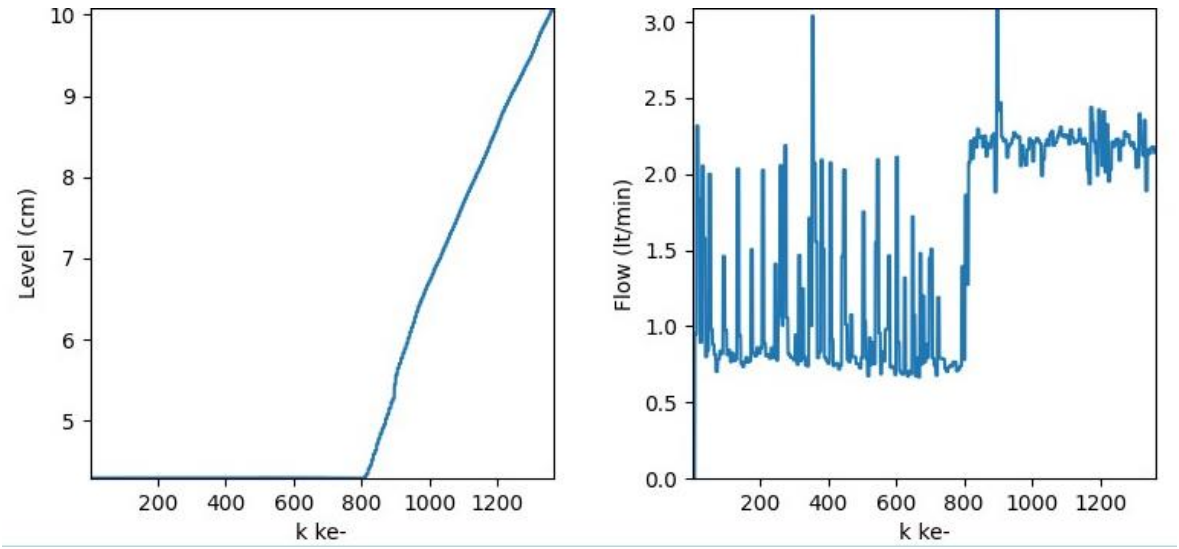
4.6 Hasil Implementasi Tuning PID Ziegler Nichols

Berikut merupakan gambar grafik respon sistem pengaturan cascade level dan flow yang dituning menggunakan metode Ziegler Nichols. Data disajikan pada gambar 4.18. Serta pada penelitian ini juga dilakukan percobaan dengan disturbance berupa bukaan valve dengan tuning berupa Ziegler Nichols. Hasil percobaan respon sistem sistem pengaturan *cascade level* dan *flow* dengan gangguan yang dituning dengan metode tersebut dapat dilihat pada gambar 4.19.



Gambar 4-18 Respon Sistem Tuning dengan Ziegler Nichols tanpa Gangguan

Pada gambar di atas merupakan implementasi tuning PID pada sistem pengaturan cascade level dan flow. Pada input berupa set point level sehingga output juga merupakan level yang diinputkan. Dari grafik tersebut diperoleh karakteristik *Rise Time* Level sebesar 62.4 detik, *Settling time* 92.64 detik dan sistem steady state pada 10.12 cm. Adapun *steady state error* pada sistem ini adalah 0.0012. Sedangkan pada inner loop yakni sistem pengaturan flow diperoleh hasil yakni pada tahap awal yakni pengisian tangka sistem stabil diantara 0.5 liter per menit sampai dengan 2 liter per menit apabila sensor level sudah terbaca flow akan naik dengan cepat dan mengalami overshoot hingga 3.5 liter per menit setelah itu akan stabil diantara 2 liter per menit sampai dengan 2.4 liter per menit. Meskipun pada sistem pengaturan flow mengalami overshoot pompa namun pada sistem pengaturan level respon sistem menjadi lebih halus.

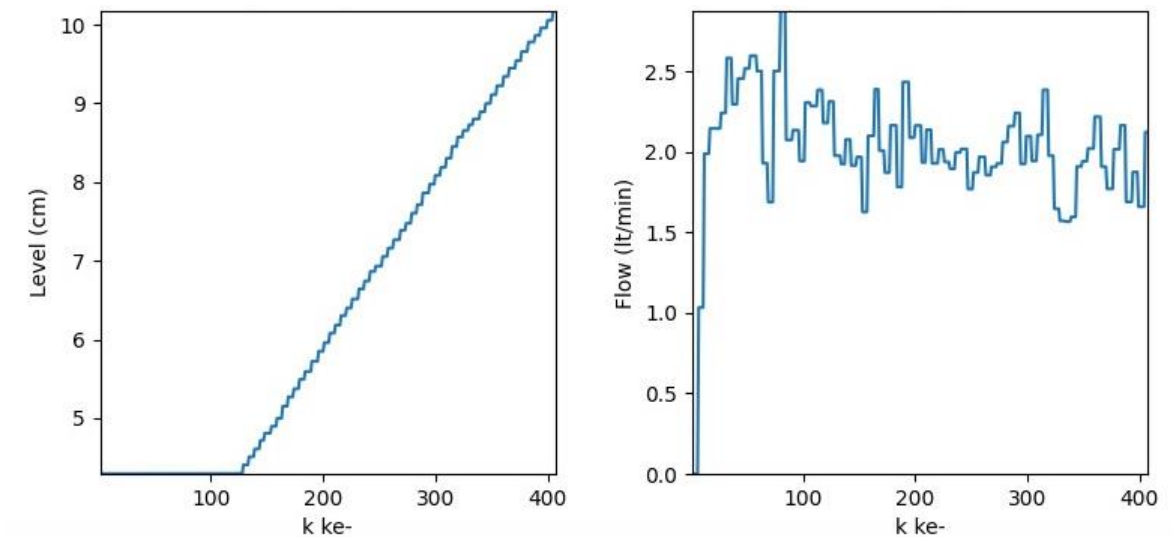


Gambar 4-19 Respon Sistem Tuning dengan Ziegler Nichols dengan Gangguan bukaan valve

Pada Gambar 4.19 di atas adalah grafik respon sistem cascade level dan flow yang di tuning dengan controller PID dengan metode Ziegler Nichols tetapi dalam realisasinya diberikan disturbance berupa bukaan valve. Dari respons sistem yang ada sistem tersebut pada sistem pengaturan level menuju set poin 10 mengalami kendala mencapai set poin lebih lama dikarenakan pada implementasi inner loop sistem berusaha mencari kestabilan dengan melakukan pengisian tangki terlebih dahulu untuk menanggulangi disturbance yang ada, setelah sensor sudah mampu terangkat dan membaca hasil sistem pengaturan flow kemudian akan mencari kestabilan yang baru untuk menuju set poin level yang ditentukan yakni 10 cm. Pada awalnya sistem pengaturan flow stabil antara 0.8 liter per menit hinga 2 liter per menit namun terdapat overshoot hingga liter per menit, setelah sensor level mulai terangkat sistem mengalami overshoot namun dapat mencapai steady state antara 2 liter per menit hingga 2.4 liter per menit. Dari sistem pengaturan level outer loop sistem mencapai *rise time* yakni 165,14 detik, *settling time* 252,12 detik.

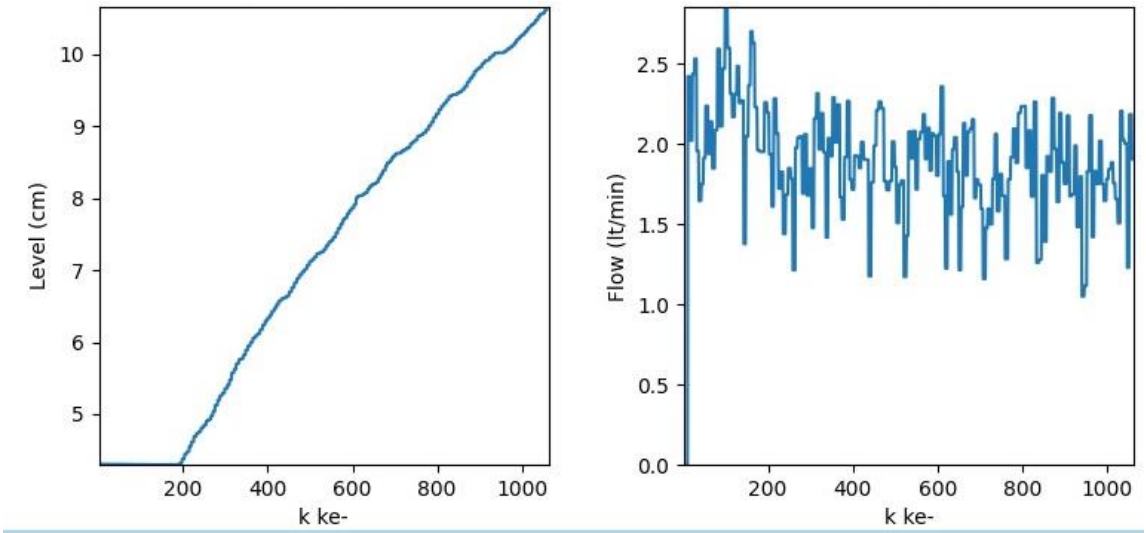
4.7 Hasil Implementasi Tuning PID dengan Metode PSO

Pada penelitian ini dilakukan juga implementasi pada Plant PCT 100 dengan tuning PID menggunakan metode *Particle Swarm Optimizartion*. Pada pengujian pertama yakni dilakukan tanpa gangguan atau disebut dengan kondisi ideal. Sedangkan pengujian kedua dilakukan dengan adanya gangguan berupa bukaan valve pada tangka. Hasil pengujian disajikan dalam bentuk gambar respons sistem pada Gambar 4.20 dan 4.21.



Gambar 4-20 Respon Sistem tuning dengan PSO PID tanpa gangguan

Grafik di atas merupakan grafik respons sistem implementasi controller PID dengan menggunakan PSO pada sistem pengaturan cascade level dan flow. Pada uji ini sistem tidak diberikan gangguan berupa bukaan valve sehingga hanya dikontrol dengan menggunakan pompa dan valve drain sebagai actuator didapatkan karakteristik sistem didapatkan *rise time* 48.29 detik, *Settling time* 70.81 detik dan *steady state* pada nilai 10.0433 sehingga dapat dikatakan sistem mengalami *overshoot* yang sangat kecil yakni 0.043%. Pada sistem pengaturan inner loop yakni flow di dapat *rise time* 1.9 detik, *settling time* 4.29 detik, dan *steady state* stabil diantara 1.5 liter per menit hingga 2.2 liter per menit.



Gambar 4-21 Respon Sistem tuning dengan PSO PID dengan gangguan bukaan valve

Grafik di atas merupakan grafik respons sistem implementasi controller PID dengan menggunakan PSO pada sistem pengaturan cascade level dan flow. Pada uji ini sistem diberikan gangguan berupa bukaan *valve*. Dilihat dari grafik di atas controller PID dengan Tuning PSO dapat mengatasi gangguan yang ada. Sehingga dapat disimpulkan bahwa optimasi perancangan kontroler PID dengan metode PSO dapat dikatakan berhasil memperoleh nilai Kp, Ki, dan Kd yang optimum. Dari grafik di atas diperoleh karakteristik sistem *rise time* sebesar 89.92 detik,

Settling time 146.15 detik. Sistem tersebut *steady state* pada nilai 10.125 cm, sehingga memiliki *error steady state* sebesar 0.0125%. Pada sistem pengaturan inner loop sistem mengalami transien stabil antara 1.3 liter per menit hingga 2.2 liter per menit namun pada awalnya mengalami overshoot hingga 2.8 liter per menit setelah itu sistem kembali ke keadaan tunaknya. Pada sistem implementasi inner loop memiliki karakteristik *rise time 2.02* detik, *Settling time 25.64* detik.

--Halaman ini sengaja dikosongkan--

BAB 5 KESIMPULAN DAN SARAN

5.1 Kesimpulan

Kesimpulan yang dapat diambil dalam tugas akhir ini antara lain, adalah :

a. Peningkatan nilai w atau bobot inersia pada algoritma PSO menyebabkan nilai Kp, Ki, dan Kd menjadi lebih bervariasi, performansi PSO meningkat, dan performansi sistem menjadi lebih optimal ditandai dengan minimnya rise time, settling time, minimal overshoot dan steady state error kecil.
b. Semakin banyak jumlah partikel dapat meningkatkan performansi dinamik. Hal ini dikarenakan. Peningkatan jumlah partikel menyebabkan semakin banyak titik titik area yang dijajaki sehingga semakin meningkatkan peluang dalam menentukan titik optimal dari seluruh area yang dilewati.
c. Pada Inner Loop diperoleh besaran Kp, Ki, dan Kd yang dituning dengan PSO sebesar 20.8441, 10.0045, dan 5.8969 dengan indeks performansi ITAE sebesar 0.6534 dan ISE 0.2022. Memiliki Karakteristik sistem max overshoot 0.567%, Rise time 0.1544 detik dan Settling time 0.2473 detik.
d. Pada Outer Loop diperoleh besaran Kp, Ki, dan Kd yang dituning dengan PSO sebesar 68.4639, 1.8237, dan 10.2855 memiliki indeks performansi error ITAE sebesar 1.51 dan ISE sebesar 0.5841. Sistem tersebut mengalami max overshoot sebesar 1.4001%, Rise Time 2.0383 detik, dan Settling time 3.2569 detik
e. Pada simulasi dengan disturbance controller PID yang dituning dengan PSO mampu mengatasi gangguan disturbance yang ada yakni sistem mengalami steady state Kembali dan nilai steady state error kecil.
f. Implementasi kontroller PID dengan PSO tanpa gangguan pada sistem pengaturan cascade level dan flow memiliki rise time 48.29 detik, settling time 70.81 detik, dan steady state 10.0433. Sistem hanya mengalami overshoot kecil 0.043%. Inner loop flow memiliki rise time 1.9 detik, settling time 4.29 detik, dan steady state stabil 1.5-2.2 liter/menit.
g. Implementasi kontroller PID yang dituning dengan PSO dan diberikan gangguan berupa bukaan valve pada sistem pengaturan cascade level dan flow memiliki karakteristik respons sistem *rise time* 89.92 detik, *settling time* 146.15 detik. Sistem stabil pada nilai 10.125 cm dan error steady state 0.0125%. Sistem inner loop stabil antara 1.3-2.2 liter/menit setelah awalnya overshoot 2.8 liter/menit. Inner loop memiliki rise time 2.02 detik, settling time 25.64 detik. Kontroller berhasil mengatasi gangguan valve dan memiliki error steady state kecil.

5.2 Saran

Adapun saran yang diberikan pada tugas akhir ini adalah sebagai berikut:

1. Pada plant yang digunakan yakni Sistem pengaturan Flow Plant PCT 100 seharusnya menggunakan valve flow control sehingga bisa menghasilkan respons yang lebih baik dan fungsi cascade control pada penelitian ini dapat menghasilkan respon sistem yang lebih baik.
2. Pada penelitian ini bisa dikembangkan algoritma tuning PID dengan menggunakan metode Adapative PSO, ACO, BFO, GA atau metode metaheuristic yang lain sehingga bisa lebih bervariasi.

DAFTAR PUSTAKA

Abdulameer, A., Sulaiman, M., Aras, M. S. M., & Saleem, D. (2016). GUI based control system analysis using PID controller for education. *Indonesian Journal of Electrical Engineering and Computer Science*, *3*(1), 91–101. https://doi.org/10.11591/ijeecs.v3.i1.pp91-101

Ang, K. H., Chong, G., & Li, Y. (2005). PID control system analysis, design, and technology. *IEEE Transactions on Control Systems Technology*, *13*(4), 559–576. https://doi.org/10.1109/TCST.2005.847331

Bytronic Limited. (n.d.). *"P r o c e s s C o n t r o l T e c h n o l o g y P C T - 1 0 0 " Manual Book*. Bytronic Limited.

Guong, R. L. L., Hisham, S. B., Elamvazuthi, I., Aole, S., Parasuraman, S., & Khan, M. K. A. A. (2018). PID Tuning of Process Plant using Particle Swarm Optimization. *2018 IEEE 4th International Symposium in Robotics and Manufacturing Automation, ROMA 2018*, *1*(1), 0–5. https://doi.org/10.1109/ROMA46407.2018.8986712

Ibrahim, H. E. A., Hassan, F. N., & Shomer, A. O. (2014). Optimal PID control of a brushless DC motor using PSO and BF techniques. *Ain Shams Engineering Journal*, *5*(2), 391–398. https://doi.org/10.1016/j.asej.2013.09.013

Jakoubek, P. (2009). Experimental Identification of Stabile Nonoscillatory Systems from Step-Responses by Selected Methods. *Student Conderence STC 2009*. http://stc.fs.cvut.cz/history/2009/sbornik/Papers/pdf/JakoubekPavel-304494.pdf

Kawasan, G., Serpong, P., & Tlp, T. (2011). *Penerapan pemodelan dan metode kurva reaksi proses untuk mengidentifikasi sistem duress*. 230–236.

Kumar, C. A., Nair, N. K., & Engineering, E. (2010). Multi-Objective Pid Controller Based on Adaptive Weighted Pso With Application To Steam Temperature Control in Boilers. *International Journal of Engineering Science and Technology*, *2*(7), 3179–3184.

Ogata, K. (2010). Modern Control Engineering 5th Edition. In *Prentice Hall PTR* (5th ed.).

Pano, V., & Ouyang, P. R. (2014). PSO gain tuning for position domain PID controller. *4th Annual IEEE International Conference on Cyber Technology in Automation, Control and Intelligent Systems, IEEE-CYBER 2014*, 377–382. https://doi.org/10.1109/CYBER.2014.6917493

Pillay, N. (2013). a Particle Swarm Optimization Approach for Tuning of Siso Pid Control Loops. *Engineering Research on Soft Computing Methods*, 235–258.

Pritandi, D. A. P. A. (2016). Perancangan Kontroler PID-Fuzzy untuk Sistem Pengaturan Cascade Level dan Flow pada Basic Process Rig 38-100. *Jurnal Teknik ITS*, *5*(2). https://doi.org/10.12962/j23373539.v5i2.16179

Tridianto, E., Ariwibowo, T. H., Almasa, S. K., & Prasetya, H. E. G. (2017). Cascaded PID temperature controller for FOPDT model of shell-and-tube heat exchanger based on Matlab/Simulink. *Proceedings IES-ETA 2017 - International Electronics Symposium on Engineering Technology and Applications*, *2017-Decem*, 185–191. https://doi.org/10.1109/ELECSYM.2017.8240400

Vilanova, R., & Arrieta, O. (2008). PID TUNING FOR CASCADE CONTROL SYSTEM DESIGN ETSE , Universitat Aut `. *Canadian Conference on Electrical and Computer*

Engineering, 2008. CCECE 2008., *1*, 1775–1778.

Wen, S., Deng, M., bi, S., & Wang, D. (2012). Operator-based robust nonlinear control and its realization for a multi-tank process by using a distributed control system. *Transactions of the Institute of Measurement and Control*, *34*(7), 891–902. https://doi.org/10.1177/0142331211424427

Xiang, Z., Ji, D., Zhang, H., Wu, H., & Li, Y. (2019). A simple PID-based strategy for particle swarm optimization algorithm. *Information Sciences*, *502*, 558–574. https://doi.org/10.1016/j.ins.2019.06.042

Xiong, A., & Fan, Y. (2007). Application of a PID controller using MRAC techniques for control of the DC electromotor drive. *Proceedings of the 2007 IEEE International Conference on Mechatronics and Automation, ICMA 2007*, 2616–2621. https://doi.org/10.1109/ICMA.2007.4303969

LAMPIRAN

Lampiran Tuning PID Inner Loop Menggunakan Algoritma PSO

```
clc; clear all; close all;

num = [1.03];
den = [0.468 1.39 1];
G = tf(num,den);
Gf = feedback(G,1);
step(Gf)
hold on

%PSO Constant Parameters
c1=1.5; c2=1; w=0.9; particles=100; iteration=100; Var=3;

% Search Space
a=0; %lower bound
b=100; %upper bound

% Optimization Steps
c_cf=0;

%initialization
for m=1:particles

    for n=1:Var
        v(m,n)=0;               %Velocity Particles
        x(m,n)=a+rand*(b-a);    %Position Particles
        xp(m,n)= x(m,n);
    end

    %Model Parameters
    kp = x(m,1);
    ki = x(m,2);
    kd = x(m,3);

    %Simulation Model
    Gc=pid(kp,ki,kd);
    Gcf=feedback(Gc*G,1);
    y=step(Gcf);

    % ITAE (Objective Function)
    ff1=0;
    sim1=size(y);
    for m1= 1:sim1
        ff1=ff1+((abs(y(m1)-1))*m1);
    end
    ITAE(m)=ff1;
end

%Find The Best Value
[Best_Performance, location]=min(ITAE);
fg=Best_Performance
xg(1)=x(location,1);
xg(2)=x(location,2);
xg(3)=x(location,3);
```

```
for i=1:iteration           % No Of Repetition

    for m=1:particles       % No Of Particles

        for n=1:Var         % No Of Var
            v(m,n)=(w*v(m,n))+(c1*rand*(xp(m,n)-x(m,n)))+(c2*rand*(xg(n)-x(m,n)));
% Update Velocity
            x(m,n)=x(m,n)+v(m,n);
% Update Position
        end

        positions(m,1) = x(m,1); % position of kp for particle m
        positions(m,2) = x(m,2); % position of ki for particle m
        positions(m,3) = x(m,3); % position of kd for particle m

        %Check Bound

        for n=1:Var
            if x(m,n)<a
                x(m,n)=a;
            end

            if x(m,n)>b
                x(m,n)=b;
            end
        end

        % Model Parameters
        kp=x(m,1);
        ki=x(m,2);
        kd=x(m,3);

        % Simulation Model
        Gc=pid(kp,ki,kd);
        Gcf=feedback(Gc*G,1);
        y=step(Gcf);

        % ITAE (Objective Function)
        ff1=0;
        sim1=size(y);
        for m1=1:sim1
            ff1=ff1+((abs(y(m1)-1))*m1);
        end
        ITAEp(m)=ff1;

        % Compare Local
        if ITAEp(m)<ITAE(m)
            ITAE(m)=ITAEp(m);
            xp(m,1)=x(m,1);
            xp(m,2)=x(m,2);
            xp(m,3)=x(m,3);
        end

    end

    [B_fg,location]=min(ITAE);

    % Compare Global
```

```
    if B_fg<fg
        fg=B_fg;                    % New Global Value
        xg(1)=xp(location,1);       % Position of Var 1
        xg(2)=xp(location,2);       % Position of Var 2
        xg(3)=xp(location,3);       % Position of Var 3
    end

        c_cf = c_cf+1;
        best_cf_ac(c_cf)=fg;
end

Min_ITAE=fg
kp=xg(1)
ki=xg(2)
kd=xg(3)

Gc=pid(kp,ki,kd);
Gcf=feedback(Gc*G,1);
step(Gcf);

t_cf=1:c_cf;
figure
plot(t_cf,best_cf_ac,'r--','LineWidth',2), xlabel('iteration'), ylabel('Const
Function (ITAE)')
legend('ITAE for PSO PID')
title('ITAE with each iteration')

figure
scatter3(positions(:,1), positions(:,2), positions(:,3));
xlabel('kp');
ylabel('ki');
zlabel('kd');
scatter3(positions(:,1), positions(:,2), positions(:,3), 10, 'blue');

% Plot step response and calculate steady-state error
step_info = stepinfo(Gcf);  % Get step response information
y_inf = step_info.Peak;  % Get final value of step response
t_inf = step_info.RiseTime;  % Get time corresponding to final value of step
response
y_inf = step_info.SettlingMax;  % Get final value of step response
t_inf = step_info.SettlingTime;  % Get time corresponding to final value of step
response
ess = abs(1 - y_inf);  % Calculate steady-state error as absolute difference
between desired value (1) and final value of step response

% Calculate RMSE for a sequence of steady-state errors
rmse = sqrt(ess^2);  % Calculate RMSE as the square root of the mean of the
squares of the steady-state errors
```

Lampiran Tuning PID Outer Loop Menggunakan Algoritma PSO

```
clc; clear all; close all;

num = [10.9 38.52 18.49];
den = [51.44 820.9 2477 1155 10.3];
G=tf(num,den);
Gf = feedback(G,1);
step(Gf)
hold on

%PSO Constant Parameters
c1=1.5; c2=1; w=0.9; particles=50; iteration=100; Var=3;

% Search Space
a=0; %lower bound
b=100; %upper bound

% Optimization Steps
c_cf=0;

%initialization
for m=1:particles

    for n=1:Var
        v(m,n)=0;               %Velocity Particles
        x(m,n)=a+rand*(b-a);    %Position Particles
        xp(m,n)= x(m,n);
    end

    %Model Parameters
    kp = x(m,1);
    ki = x(m,2);
    kd = x(m,3);

    %Simulation Model
    Gc=pid(kp,ki,kd);
    Gcf=feedback(Gc*G,1);
    y=step(Gcf);

    % ITAE (Objective Function)
    ff1=0;
    sim1=size(y);
    for m1= 1:sim1
        ff1=ff1+((abs(y(m1)-1))*m1);
    end
    ITAE(m)=ff1;
end

%Find The Best Value
[Best_Performance, location]=min(ITAE);
fg=Best_Performance
xg(1)=x(location,1);
xg(2)=x(location,2);
xg(3)=x(location,3);


for i=1:iteration           % No Of Repetition
```

```
    for m=1:particles       % No Of Particles

        for n=1:Var         % No Of Var
            v(m,n)=(w*v(m,n))+(c1*rand*(xp(m,n)-x(m,n)))+(c2*rand*(xg(n)-x(m,n)));
% Update Velocity
            x(m,n)=x(m,n)+v(m,n);
% Update Position
        end

        positions(m,1) = x(m,1); % position of kp for particle m
        positions(m,2) = x(m,2); % position of ki for particle m
        positions(m,3) = x(m,3); % position of kd for particle m

        %Check Bound

        for n=1:Var
            if x(m,n)<a
                x(m,n)=a;
            end

            if x(m,n)>b
                x(m,n)=b;
            end
        end

        % Model Parameters
        kp=x(m,1);
        ki=x(m,2);
        kd=x(m,3);

        % Simulation Model
        Gc=pid(kp,ki,kd);
        Gcf=feedback(Gc*G,1);
        y=step(Gcf);

        % ITAE (Objective Function)
        ff1=0;
        sim1=size(y);
        for m1=1:sim1
            ff1=ff1+((abs(y(m1)-1))*m1);
        end
        ITAEp(m)=ff1;

        % Compare Local
        if ITAEp(m)<ITAE(m)
            ITAE(m)=ITAEp(m);
            xp(m,1)=x(m,1);
            xp(m,2)=x(m,2);
            xp(m,3)=x(m,3);
        end

    end

    [B_fg,location]=min(ITAE);

    % Compare Global
    if B_fg<fg
        fg=B_fg;                    % New Global Value
        xg(1)=xp(location,1);       % Position of Var 1
```

```
        xg(2)=xp(location,2);       % Position of Var 2
        xg(3)=xp(location,3);       % Position of Var 3

    end
        c_cf = c_cf+1;
        best_cf_ac(c_cf)=fg;
end

Min_ITAE=fg
kp=xg(1)
ki=xg(2)
kd=xg(3)

Gc=pid(kp,ki,kd);
Gcf=feedback(Gc*G,1);
step(Gcf);

t_cf=1:c_cf;
figure
plot(t_cf,best_cf_ac,'r--','LineWidth',2), xlabel('iteration'), ylabel('Const
Function (ITAE)')
legend('ITAE for PSO PID')
title('ITAE with each iteration')

figure
scatter3(positions(:,1), positions(:,2), positions(:,3));
xlabel('kp');
ylabel('ki');
zlabel('kd');
scatter3(positions(:,1), positions(:,2), positions(:,3), 10, 'blue');

% Plot step response and calculate steady-state error
step_info = stepinfo(Gcf);  % Get step response information
y_inf = step_info.Peak;  % Get final value of step response
t_inf = step_info.RiseTime;  % Get time corresponding to final value of step
response
y_inf = step_info.SettlingMax;  % Get final value of step response
t_inf = step_info.SettlingTime;  % Get time corresponding to final value of step
response
ess = abs(1 - y_inf);  % Calculate steady-state error as absolute difference
between desired value (1) and final value of step response

% Calculate RMSE for a sequence of steady-state errors
rmse = sqrt(ess^2);  % Calculate RMSE as the square root of the mean of the
squares of the steady-state errors
```

Lampiran Code Python

```
import tkinter as tk
import matplotlib.pyplot as plt
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
import time
import csv
from pyModbusTCP.client import ModbusClient
import random
c = ModbusClient(host="10.0.0.1", port=502, auto_open=True, auto_close=True)

header = ['n', 'volt_flow', 'volt_level', 'volt_pot', 'timestamp']
f = open('closeloop.csv', 'w', encoding='UTF8', newline='') # open the file in
the write mode
writer = csv.writer(f) # create the csv writer
writer.writerow(header) # write the header
timeSampling = 0.05 # dalam detik, min 0.05s
n = 0 # tidak boleh nol agar index tidak e[-1]
# --------INISIASI Variabel PID----------
SP = 10
e=[0.0, 0.0]
# e[0]=0 # initial condition error, iterasi
sum_e=[0.0, 0.0]
de=[0.0, 0.0]
kp = 68
ki = 2
kd = 10

#-----plot data-----
def plot_data():
    # print("masuk plot data")
    global n,cond, arr_n, arr_volt_level, timeLast, sent
    if (cond==True):
        # print(cond)
        timeNow = time.time()
        # condition untuk sampling, jika sudah melebihi time sampling
        if True:
            n = n + 1
            # print("timeLast sudah bisa masuk")
            arr_n.append(n)
            volt_flow,volt_level,volt_pot = kontroller()

            arr_volt_level.append(volt_level)

            timeLast = timeNow
            ax.set_xlim(min(arr_n), max(arr_n))
            ax.set_ylim(min(arr_volt_level), max(arr_volt_level))
            # lines = ax.plot(arr_elapse, arr_volt_level, color='green')
            lines.set_xdata(arr_n)
```

```
            lines.set_ydata(arr_volt_level)
            # write multiple rows
            # print("v LT: " + str(volt_level) + " V")
            print("level: " + str(volt_level) + " cm ; volt pump: " +
str(sinyal_level) + " VDC")
            writer.writerow([n, volt_flow, volt_level, volt_pot, timeNow-
start_time])
            # print(arr_n)
            # print(arr_volt_level)
            time.sleep(timeSampling)
            canvas.draw()
        window.after(1, plot_data)
def plot_start():
    global cond, start_time
    cond = True
    start_time = time.time()
    # s.reset_input_buffer()
    print("start mulai yaa")
    # return cond
    window.after(1, plot_data)

def plot_stop():
    global cond, sent
    cond = False
    print("sudah stop yaa")
    sent = c.write_multiple_registers(16, [0, 4096])
    f.close()

def kontroller():
    global volt_flow,volt_level,volt_pot, sent,n, sinyal_level
    regs = c.read_holding_registers(8, 8)  # format: (address,quantity).
quantity tidak boleh lebih, tetapi boleh kurang
    bit_flow = regs[0]
    volt_flow = 20*bit_flow/65535 - 10

    bit_level = regs[1]
    # volt_level = (20 * bit_level / 65535 - 10)
    volt_level = 1.7055517310856645*(20*bit_level/65535 - 10) +
4.80943785889385

    bit_pot = regs[2]
    volt_pot = (20 * bit_pot / 65535 - 10)

    e[n] = SP - volt_level
    sum_e[n] = sum_e[n] + e[n]*timeSampling
    de[n] = (e[n] - e[n-1])/timeSampling

    P = kp*e[n]
    I = ki*sum_e[n]
```

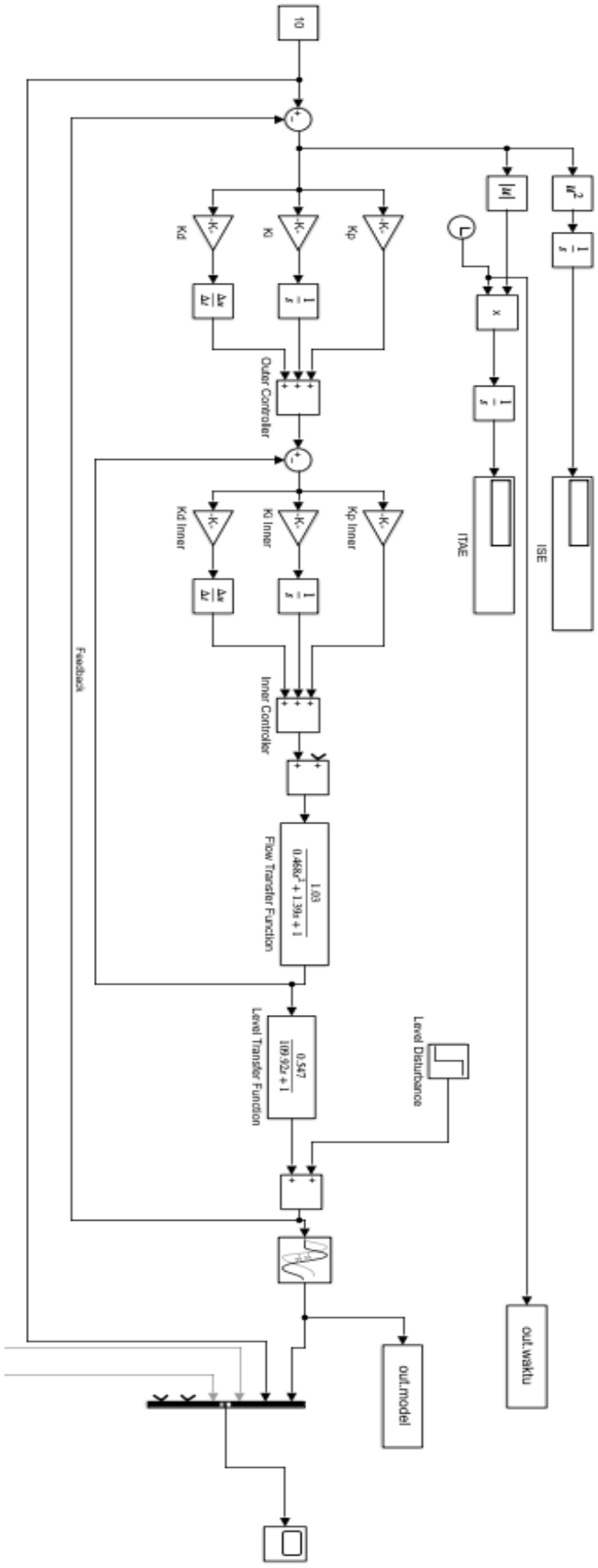
```
    D = kd*de[n]
    uPID = P+I+D
    sinyal_level= 0.586320532982868*uPID - 2.819872168774626  # volt sinyal
kirim level
    bit_uPID = 4096*sinyal_level/10
    if bit_uPID > 4096:
        bit_uPID = 4096
    if bit_uPID < 0:
        bit_uPID = 0
    e.append(e[n])
    sum_e.append(e[n])
    de.append(e[n])

    sent = c.write_multiple_registers(16, [int(bit_uPID), 0])  # list bit
pompa dan valve max.4096

    return volt_flow,volt_level,volt_pot

cond = False
timeLast = 0
arr_n = []
arr_volt_level = []


window = tk.Tk()
window.title('GUI Closed Loop PCT-100')
window.configure(background = 'light blue')
window.geometry("700x500")

fig = plt.Figure();
ax = fig.add_subplot(111)
ax.set_title('Closed Loop Level')
ax.set_xlabel('n ke-')
ax.set_ylabel('Level (cm)')
lines = ax.plot([],[])[0]

canvas = FigureCanvasTkAgg(fig, master=window)  # A tk.DrawingArea.
canvas.get_tk_widget().place(x = 10,y=10, width = 600,height = 400)
canvas.draw()
# ----------create button---------
window.update();
start = tk.Button(window, text = "Start", font = ('calbiri',12),command =
lambda: plot_start())
start.place(x = 100, y = 450 )

window.update();
stop = tk.Button(window, text = "Stop", font = ('calbiri',12), command =
lambda:plot_stop())
stop.place(x = start.winfo_x()+start.winfo_reqwidth() + 20, y = 450)
```

```
# window.after(1,plot_data)
window.mainloop()

# greeting = tk.Label(text="Hello, Tkinter")
# greeting.pack()
```

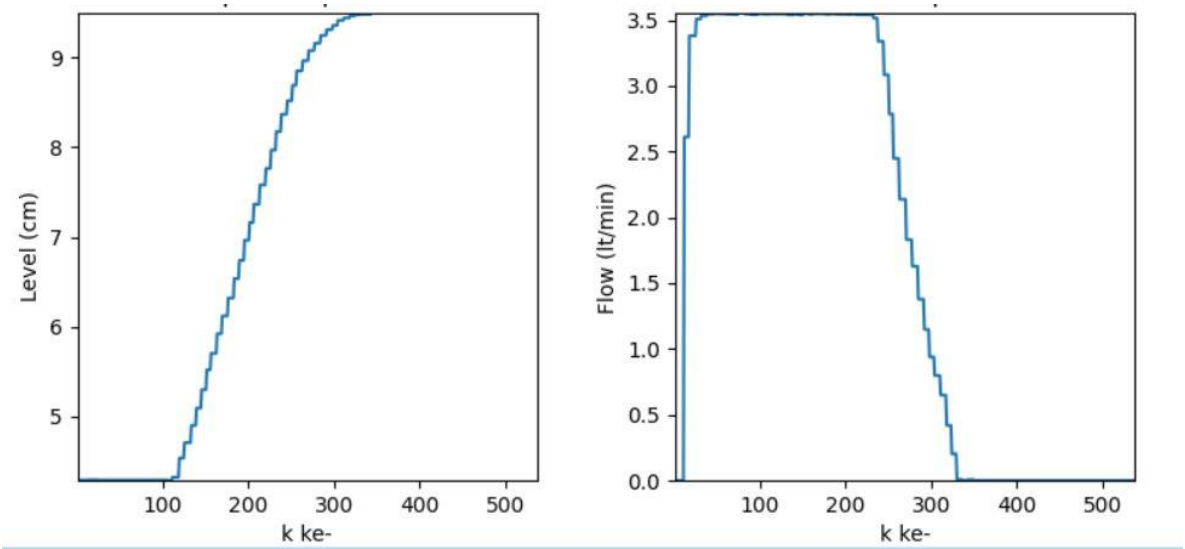
Lampiran Regresi Linier

```
# input : level sebenarnya
# output : level yg dibaca transmitter
level_sebenarnya = [0,0] #inisiasi level
voltOutput_LT = [0,0] #inisiasi volt output
# data n=1, awal
level_sebenarnya[0] = 4.5 #cm
voltOutput_LT[0] = -0.1814297703517198 #vdc
# data n terakhir
level_sebenarnya[1] = 20 #cm
voltOutput_LT[1] = 8.906538490882735 #vdc
# sinyal feedback LT, (volt LT ---> level)
# perhitungan pers. garis lurus
# m = (y2-y1) / (x2-x1)
m = (level_sebenarnya[1] - level_sebenarnya[0])/(voltOutput_LT[1] - voltOutput_LT[0])
# c = y1 - m*x1
c = level_sebenarnya[0] - m*voltOutput_LT[0]

# khusus inverse, sinyal kontrol kontroller (level --> voltage pump)
mi= 1/m
ci= -c/m
print(mi)
print(ci)
```

Lampiran Diagram Blok Sistem Cascade Level dan Flow



Lampiran Implementasi Sistem Pengaturan Level Close Loop



SP = 10cm,

Gain Kp = 10,

k terakhir = 537,

Level steady state: 9.49733033345646 cm

Flow steady state: 0.0 Volt,

BIODATA PENULIS

Penulis dilahirkan di Wonogiri, 1 Januari 2000, merupakan anak pertama dari 2 bersaudara. Penulis telah menempuh pendidikan formal yaitu di TK Mambaul Hikmah Selogiri, SDIT Al Huda Wonogiri, SMPN 1 Sukoharjo dan SMAN 1 Sukoharjo. Setelah lulus dari SMAN 1 Sukoharjo tahun 2018, Penulis mengikuti SNMPTN dan diterima di Departemen Teknik Elektro FTEIC - ITS pada tahun 2018 dan terdaftar dengan NRP 07111840000029.

Di Departemen Teknik Elektro Penulis sempat aktif di beberapa kegiatan Organisasi yang diselenggarakan oleh ITS yakni Staff JMMI pada 2018, Menjabat Sekretaris Lembaga Dakwah Islam KALAM ELITS dan aktif sebagai Asisten Laboratorium Kontrol dan Otomasi. Selain itu penulis juga aktif mengikuti PKM.