DESAIN DAN IMPLEMENTASI KONTROLER PID *GAIN SCHEDULING* UNTUK SISTEM PENGATURAN PROSES *LEVEL* PADA *PROCESS CONTROL TECHNOLOGY* - 100

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Sarjana Teknik
Pada
Bidang Studi Teknik Sistem Pengaturan
Jurusan Teknik Elektro
Institut Teknologi Sepuluh Nopember

Menyetujui :

Dosen Pembimbing I	Dosen Pembimbing II
Ir. Joko Susila, MT.	**Imam Arifin, ST., MT**
NIP. 1966 06 06 1991 02 1001	NIP. 1973 02 22 2002 12 1001

SURABAYA
JULI, 2014

DESAIN DAN IMPLEMENTASI KONTROLER PID *GAIN SCHEDULING* UNTUK SISTEM PENGATURAN PROSES *LEVEL* PADA *PROCESS CONTROL TECHNOLOGY* - 100

Rachmad Dwi Raharjo-2212105091

Pembimbing I : Ir. Joko Susila MT.
Pembimbing II : Imam Arifin ST., MT.

ABSTRAK

Proses *level* saat mengalami perubahan nilai *set point* yang tidak sesuai dengan keluaran pompa air, sehingga perubahan tersebut dapat mengakibatkan keluaran menjadi tidak sesuai yang diinginkan. Oleh karena itu, dirancang suatu kontroler untuk mengatur kondisi menjadi linier pada *set point* yang sudah ditentukan, jika proses mengalami perubahan beban dan nilai *set point*. Kontroler yang akan digunakan adalah kontroler PID dan metode *gain scheduling*. Hasil dari penelitian ini, dapat terealisasi sebuah sistem kontrol proses dengan kontroler PID *gain scheduling*, di mana kontroler akan mengatur nilai *gain* saat varibel tertentu mengalami perubahan, sehingga hasil keluaran dari suatu sistem dapat sesuai dengan *set point* yang diinginkan. Kontroler PID *gain scheduling* dapat mengendalikan respon *plant* proses *level* pada PCT-100 yang mengalami perubahan kondisi dengan rata-rata kesalahan sebesar 0,06% pada simulasi dan kesalahan waktu tunak pada implementasi sebesar 0,12%.

Kata Kunci : *Proses Level,* PID, *Gain scheduling*

--halaman ini sengaja dikosongkan--

DESIGN AND IMPLEMENTATION OF A PID CONTROLLER GAIN SCHEDULING FOR LEVEL PROCESS CONTROL SYSTEM IN PROCESS CONTROL TECHNOLOGY - 100

Rachmad Dwi Raharjo-2212105091

1st Supervisor : Ir. Joko Susila, MT.
2nd Supervisor : Imam Arifin, ST., MT.

ABSTRACT

The level process of changing the set point value with the output of the water pump and, so that these changes can result in the output be desired. Therefore, a controller is designed to regulate the conditions become linear at a predetermined set point, if the load changes and the set point value. Controller that will be used is a PID controller and gain scheduling method. The results of this study, expected to be realized a process control system with PID controller gain scheduling, where the controller will adjust the gain value when certain variables change, so that the output of a system can be in accordance with the desired set point. Gain scheduling PID controller can control the level of the plant response to the PCT-100 that experienced a change in conditions with an average error of 0.06% at steady-state simulation and time error on the implementation of 0.12%.

Keywords*: Level Process,* PID, *Gain scheduling.*

--halaman ini sengaja dikosongkan--

BAB II
TEORI PENUNJANG

Pada bab ini dibahas mengenai teori-teori yang berkaitan dengan topik penelitian yang dilakukan. Beberapa hal yang dibahas meliputi tinjauan pustaka mengenai penelitian yang berkaitan, teori tentang sistem kontrol proses, identifikasi sistem, kontroler PID, konsep *gain scheduling*, *Process Control Technology-100*, gambaran umum mengenai OPC dan LabVIEW 2012.

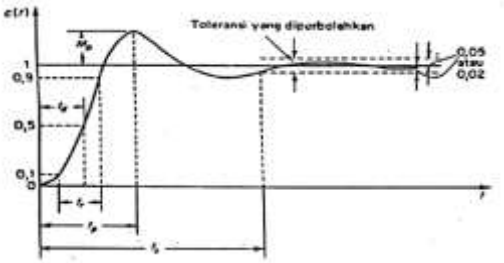
2.1. Sistem Kontrol Proses

Secara umum sistem kontrol adalah susunan komponen-komponen fisik yang dirakit sedemikian rupa sehingga mampu mengatur sistemnya sendiri atau sistem di luarnya. Sistem kontrol adalah proses pengaturan atau pengendalian terhadap satu atau beberapa besaran (variabel, parameter), sehingga berada pada suatu harga *range* tertentu. Istilah lain sistem kontrol atau teknik kendali adalah teknik pengaturan, sistem pengendalian, atau sistem pengontrolan.

Secara umum ada empat aspek yang berkaitan dengan sistem kontrol yaitu masukan, keluaran, sistem dan proses. Masukan (*input*) adalah rangsangan dari luar yang diterapkan ke sebuah sistem untuk memperoleh tanggapan tertentu dari sistem kontrol tersebut. Keluaran (*output*) adalah tanggapan sebenarnya yang didapatkan dari suatu sistem kontrol. Tanggapan ini bisa sama dengan masukan atau mungkin juga tidak sama dengan tanggapan pada masukannya.

Sistem kontrol proses adalah gabungan kerja dari alat-alat pengendalian otomatis. Semua peralatan yang membentuk sistem kontrol disebut instrumentasi kontrol proses. Contoh sederhana instrumentasi kontrol proses adalah saklar *level* yang bekerja secara otomatis mengendalikan suhu setrika. Instrumentasi pengendalinya disebut *level switch*, saklar akan memutuskan sumber tegangan pada pompa apabila *level* tangki air pada titik yang dikehendaki. Tujuan utama dari suatu sistem kontrol adalah untuk mendapatkan unjuk kerja yang optimal pada suatu sistem yang dirancang. Untuk mengukur performansi dalam pengaturan, biasanya dinyatakan dengan ukuran - ukuran waktu naik (τ_r), waktu puncak (τ_p), *settling time* (τ_s), *maximum overshoot* (Mp), waktu tunda/*delay time* (τ_d), nilai kesalahan, dan *damping ratio.*

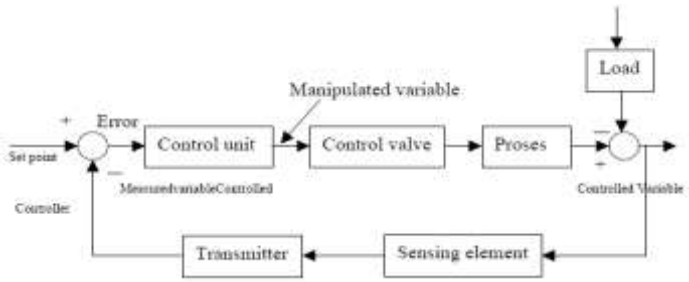
Nilai tersebut bisa diamati pada respon *transient* dari suatu sistem kontrol pada Gambar 2.1.



Gambar 2.1. Karakteristik respon *transient* sistem

Dalam optimisasi agar mencapai target optimal sesuai yang dikehendaki, maka sistem kontrol berfungsi untuk melakukan pengukuran (*measurement*), membandingkan (*comparison*), pencatatan dan penghitungan (*computation*) dan perbaikan (*correction*).

Semua analisa sistem kontrol selalu dimulai dengan menampilkan diagram blok sistem. Di dalam diagram blok sistem akan selalu ada komponen-komponen pokok seperti elemen proses, elemen pengukuran (*sensing element* dan *transmitter*), elemen kontroler (*control unit*), dan *final control element* (*control valve*). Dalam bentuk matematis semua blok elemen itu akan diisi persamaan-persamaan matematik yang merupakan fungsi alih elemen-elemen tersebut [1]. Secara umum, diagram blok sistem kontrol dapat dilihat seperti pada Gambar 2.2.



Gambar 2.2. Diagram blok sistem

Diagram blok semacam ini dapat ditemukan pada hampir semua literatur. Ada literatur lain yang menampilkan kotak-kotak elemen dalam susunan yang berbeda dengan yang ada dalam gambar diatas. Namun pada dasarnya semua cara penggambaran itu sama benarnya,

dan cara kerjanya pun tidak berbeda sama sekali. Sebenarnya diagram kotak ini disiapkan untuk analisa matematis.

Di dalam Gambar 2.2, bagian kontroler mempunyai *summing junction* dengan tanda positif-negatif (+/-). Di titik inilah langkah membandingkan dilakukan dengan menggunakan besaran *set-point* dengan sinyal hasil pengukuran. Hasilnya adalah sinyal kesalahan. Tanda negatif (-) di *summing junction* membawa arti yang sangat spesifik bagi seluruh sistem. Karena tanda inilah sistem kontrol otomatis juga biasanya disebut dengan umpan balik negatif.

2.2. Identifikasi Sistem

Model matematika sistem diperlukan untuk menggambarkan hubungan antara masukan dan keluaran sistem. Salah satu cara untuk mendapatkan model matematika sistem adalah dengan metode identifikasi. Metode identifikasi adalah suatu metode yang menggunakan hubungan data masukan dan keluaran yang selanjutnya dilakukan pengujian dan analisis dengan model pendekatan, sehingga dapat ditentukan nilai parameter secara analitis. Untuk mendapatkan respon sistem tersebut, dapat digunakan beberapa sinyal masukan, seperti sinyal *step, ramp, impulse*, dan sinusoidal. Terdapat 2 macam identifikasi sistem yaitu identifikasi dinamis dan statis. Pada identifikasi statis, sinyal yang diberikan berupa sinyal *step* yang konstan sampai sistem mencapai keadaan *steady state*. Pada identifikasi dinamis, sinyal yang digunakan sebagai masukan merupakan sinyal acak, sehingga menghasilkan nilai keluaran yang acak pula.

2.2.1 Karakteristik Respon Sistem Orde Satu [2]

Fungsi alih sistem orde satu dinyatakan oleh Persamaan (2.1)

$$\frac{C(s)}{R(s)} = \frac{K}{\tau s + 1} \tag{2.1}$$

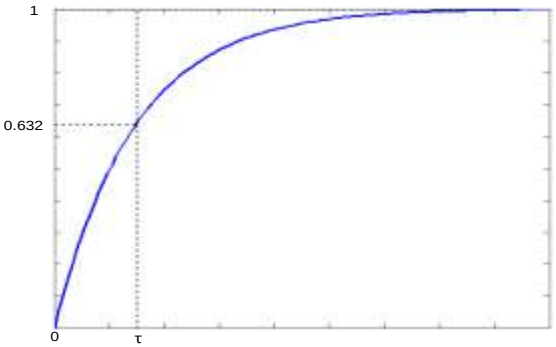
Dimana :
K = Gain Overall
τ = Konstanta Waktu

Untuk masukan sinyal *unit step*, $r(t) = u(t)$, transformasi Laplace dari sinyal masukan $r(t) = u(t) \rightarrow R(s) = 1/s$. Respon keluaran sistem orde satu dengan masukkan sinyal *step* dalam kawasan *s* dinyatakan dalam Persamaan (2.2).

$$C(s) = K(\frac{1}{s} - \frac{1}{(s + 1/\tau)}) \quad (2.2)$$

Dengan menggunakan inversi tranformasi Laplace diperoleh respon dalam kawasan waktu yang dinyatakan dalam Persamaan (2.3).

$$c(t) = K\left(1 - e^{-1/\tau t}\right), t \geq 0 \quad (2.3)$$



Gambar 2.3. Respon orde satu terhadap masukan *units step*

Kurva respon orde satu untuk masukan sinyal unit *step* ditunjukkan oleh Gambar 2.3. Ketika diberi masukan *unit step*, keluaran sistem *c(t)* mula-mula adalah nol dan terus naik hingga mencapai nilai K. salah satu karakteristik sistem orde satu adalah ketika nilai t = τ, yaitu ketika nilai keluaran mencapai 63,2% dari nilai akhirnya.

Karakteristik respon sistem orde satu dilihat berdasarkan respon sistem ketika sistem diberi masukan sinyal *step*. Karakteristik sistem orde satu dibedakan menjadi karakteristik respon *transient* dan karakteristik respon keadaan tunak atau *steady state*. Berdasarkan Persamaan 2.1, terdapat 2 parameter yaitu *K* yang menyatakan *gain overall* dan *τ* yang menyatakan *time constant* (konstanta waktu). Nilai *K* merupakan hasil perhitungan dari Persamaan 2.4:

$$K = \frac{Yss}{Xss} \quad (2.4)$$

dengan *Yss* adalah keluaran saat *steady state* dan *Xss* adalah masukan saat *steady state*.

Terdapat 2 macam karakteristik respon pada orde satu yaitu karakteristik respon *transient* dan *steady state*. Karakteristik respon *transient* pada orde satu terdiri dari:

1. Spesifikasi Teoritis
   Time constant atau konstanta waktu (τ) adalah waktu yang dibutuhkan respon mulai t = 0 sampai respon mencapai 63,2% dari respon *steady state*. Konstanta waktu yang lebih kecil akan mempercepat respon sistem.

2. Spesifikasi Praktis
   a) *Settling time* atau waktu tunak (t_s) adalah ukuran waktu yang menyatakan bahwa respon sistem telah masuk pada daerah *steady state*. Jika dihubungkan dengan konstanta waktu τ, maka t_s dapat dirumuskan dengan:
      t_s ($\pm$ 5%) ≈ 3τ (2.5)
      t_s ($\pm$ 2%) ≈ 4τ (2.6)
      t_s ($\pm$ 0,5%) ≈ 5τ (2.7)
   b) *Rise time* atau waktu naik (t_r), adalah ukuran waktu yang menyatakan bahwa respon sistem telah naik dari 5% ke 95% atau 10% ke 90% dari nilai respon *steady state*.
      t_r (5% - 95%) = τ *ln* 19 (2.8)
      t_r (10% - 90%) = τ *ln* 9 (2.9)
   c) *Delay time* atau waktu tunda (t_d), adalah waktu yang dibutuhkan respon mulai t = 0 sampai respon mencapai 50% dari nilai *steady state*. Waktu tunda menyatakan besarnya faktor keterlambatan respon akibat proses *sampling*.
      t_d = τ *ln* 2 (2.10)

Karakteristik respon *steady state* sistem orde satu diukur berdasarkan kesalahan pada keadaan tunak atau *error steady state* (e_{ss}), yaitu:

$$e_{ss} = Y_{ss} - X_{ss} \quad (2.11)$$

2.2.2 Karakteristik Respon Sistem Orde Dua [2]

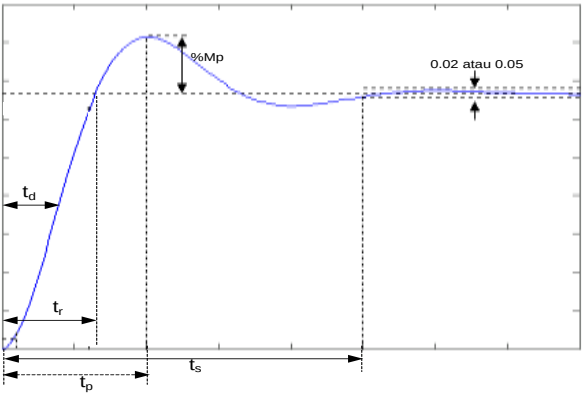
Persamaan umum sistem orde dua dinyatakan dalam Persamaan (2.12).

$$\frac{C(s)}{R(s)} = \frac{K\omega_n^2}{s^2 + 2\xi\omega_n s + \omega_n^2} \tag{2.12}$$

Dimana :

K = Gain Overall
ξ = Rasio Redaman
ω_n= Frekuensi Alami

Bentuk umum kurva respon orde dua untuk masukan sinyal *unit step* ditunjukkan oleh Gambar 2.4.



Gambar 2.4. Respon orde dua terhadap masukan *unit step*

Dari grafik di atas, diketahui karakteristik keluaran sistem orde dua terhadap masukan *unit step*, yaitu:

1. Waktu tunda (*delay time*), t_d
   Ukuran waktu yang menyatakan faktor keterlambatan respon keluaran terhadap masukkan, diukur mulai t = 0 s/d respon mencapai 50% dari respon *steady state*. Persamaan (2.13) menyatakan besarnya waktu tunda dari respon orde dua.

$$t_d = \frac{0.742}{\xi\omega_n} \tag{2.13}$$

2. Waktu naik (*rise time*), t_r

Waktu naik adalah ukuran waktu yang diukur mulai dari respon t = 0 sampai dengan respon memotong sumbu *steady state* yang pertama. Besarnya nilai waktu naik dinyatakan pada persamaan (2.14).

$$t_r = \frac{1}{\omega_n \sqrt{1-\xi^2}} \left(\pi - \tan^{-1} \frac{\sqrt{1-\xi^2}}{\xi} \right) \tag{2.14}$$

3. Waktu puncak (*peak time*), t_p

Waktu puncak adalah waktu yang diperlukan respon mulai dari *t=0* hingga mencapai puncak pertama *overshoot*. Waktu puncak dinyatakan pada persamaan (2.15).

$$t_p = \frac{1}{\omega_n \sqrt{1-\xi^2}} \tag{2.15}$$

4. *Overshoot* maksimum, *Mp*

Nilai relatif yang menyatakan perbandingan antara nilai maksimum respon (*overshoot*) yang melampaui nilai *steady state* dibanding dengan nilai *steady state*. Besarnya persen *overshoot* dinyatakan dalam persamaan (2.16)

$$\%M_p = \exp\left(-\frac{\pi\xi}{\sqrt{1-\xi^2}} \right) \times 100\% \tag{2.16}$$

5. Waktu tunak (*settling time*), t_s

Waktu tunak adalah ukuran waktu yang menyatakan respon telah masuk ±5%, atau ±2%, atau ±0.5% dari keadaan *steady state*, dinyatakan dalam persamaan (2.17).

$$t_s(\pm 5\%) = \frac{3}{\xi\omega_n} \quad ; \quad t_s(\pm 2\%) = \frac{4}{\xi\omega_n} \quad ; \quad t_s(\pm 0.5\%) = \frac{5}{\xi\omega_n} \tag{2.17}$$

2.2.3 Metode Identifikasi Sistem Tanpa Osilasi [3]

Identifikasi sistem tanpa osilasi dengan masukan respon *step* dapat dilakukan dengan beberapa metode. Metode tersebut ada 7 metode yaitu Metode Vitečková Orde 1, Metode Vitečková Orde 2, Metode Latzel, Metode Harriott, Metode Smith, Metode Strejc, dan Metode Sundaresan

& Krishnaswamy. Untuk semua metode, terdapat *gain overall* yaitu *K*. di mana *K* adalah hasil perhitungan keluaran *steady state* dibagi dengan masukan *steady state* yang dirumuskan sebagai berikut:

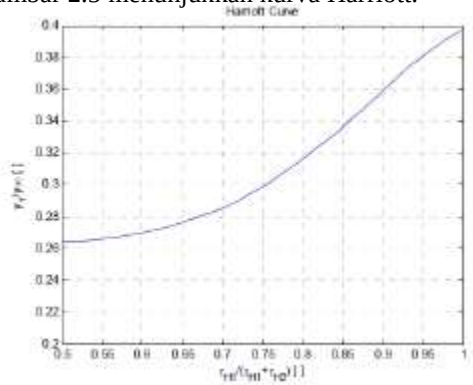
$$K = \frac{Yss}{Xss} \tag{2.18}$$

dengan *Yss* adalah keluaran saat *steady state* dan *Xss* adalah masukan saat *steady state*.

Salah satu metode yang digunakan adalah metode Harriott, metode ini didekati dengan pendekatan orde 2 dengan kemungkinan adanya *time delay*. Fungsi alih untuk metode Harriott diberikan pada persamaan 2.19.

$$G_H(s) = \frac{K}{(\tau_{H1}s + 1)(\tau_{H2}s + 1)} e^{-T_{dH}S} \tag{2.19}$$

Di mana τ_{H1} dan τ_{H2} adalah parameter yang dihitung dari pembacaan kurva Harriott. Gambar 2.5 menunjukkan kurva Harriott.



Gambar 2.5 Kurva Harriott untuk Parameter τ_{H1} dan τ_{H2}

Untuk mendapatkan parameter τ_{H1}, τ_{H2}, dan T_{dH} dapat dilakukan tahapan sebagai berikut:

1. Mencari Jumlah τ_{H1} dan τ_{H2}
   Jumlah dar parameter τ_{H1} dan τ_{H2} dirumuskan sebagai berikut :

$$\tau_{H1} + \tau_{H2} = \frac{t_{73}}{1{,}3} \quad (2.20)$$

Dengan t_{73} adalah waktu saat respon berada pada kondisi 73% dari keluaran *steady state* (Y_{ss}).

2. Mencari parameter t_1 dan Y_1
   Parameter t_1 dibutuhkan untuk mendapatkan parameter Y_1, dimana t_1 dan Y_1 diperoleh dari respon sistem yang diidentifikasi. Perhitungan untuk mendapatkan parameter t_1, dirumuskan sebagai berikut:

$$t_1 = \frac{\tau_{H1}+\tau_{H2}}{2} \quad (2.21)$$

3. Mencari parameter pada kurva Harriott
   Parameter yang diberikan pada kurva Harriott adalah parameter yang ada pada sumbu X dan Y. parameter sumbu Y pada kurva Harriott diberikan dengan:

$$\frac{Y_1}{Y_\infty}[\,] \text{ atau } \frac{Y_1}{Y_{ss}} \quad (2.22)$$

   Setelah diperoleh parameter $\frac{Y_1}{Y_\infty}[\,]$ sebagai koordinat sumbu Y, dapat ditarik garis untuk mendapatkan parameter $\tau_{H1}/(\tau_{H1} + \tau_{H2})$ [ ] di mana parameter tersebut adalah koordinat sumbu X.

4. Mencari parameter τ_{H1}
   Diasumsikan nilai parameter $\tau_{H1}/(\tau_{H1} + \tau_{H2})$ [ ] pada kurva Harriott adalah x, kemudian subsitusikan persamaan 2.20.

$$\frac{\tau_{H1}}{\tau_{H1} + \tau_{H2}} = x \quad (2.23)$$

$$\frac{\tau_{H1}}{t_{73}/1{,}3} = x \rightarrow \tau_{H1} = x.\frac{t_{73}}{1{,}3} \quad (2.24)$$

5. Mencari parameter T_{H2}
   Untuk mendapatkan parameter T_{H2}, subsitusikan persamaan 2.24 pada persamaan 2.20

$$\tau_{H1} + \tau_{H2} = \frac{t_{73}}{1{,}3} \tag{2.25}$$

$$\left(x.\frac{t_{73}}{1{,}3}\right) + \tau_{H2} = \frac{t_{73}}{1{,}3} \tag{2.26}$$

$$\tau_{H2} = \frac{t_{73}}{1{,}3} - \left(x.\frac{t_{73}}{1{,}3}\right) \tag{2.27}$$

6. Mencari parameter T_{dH}
   T_{dH} adalah *time delay* yang dirumuskan sebagai berikut:

$$T_{dH} = 1{,}937\ t_{33} - 0{,}937\ t_{70} \tag{2.28}$$

   dengan t_{33} dan t_{70} adalah waktu saat respon berada pada kondisi 33% dan 70% dari keluaran *steady state* (Y_{ss}). Jika T_{dH}bernilai kurang dari 0 atau negatif, maka sistem dianggap tidak memiliki *time delay*.

Setelah diperoleh parameter τ_{H1}, τ_{H2}, dan T_{dH}, subsitusikan hasilnya pada persamaan 2.19.

2.3. Integral *Square Error* (ISE) [3]

Integral *Square Error* atau ISE digunakan untuk mengetahui kualitas sistem. Validasi ini diperlukan untuk mengetahui kualitas hasil pemodelan terhadap respon plant sebenarnya. Perhitungan untuk ISE dirumuskan pada persamaan 2.29.

$$ISE = \int_0^t e(t)^2 dt \tag{2.29}$$

Di mana *e* adalah hasil pengurangan dari data hasil pemodelan dan data hasil pengukuran terhadap waktu. Nilai *e* tersebut kemudian dikuadratkan dan dijumlahkan sesuai dengan banyaknya data terhadap waktu.

Hasil perhitungan ISE menunjukkan seberapa baik fungsi alih merepresentasikan respon hasil pengukuran. Semakin kecil nilai ISE, semakin baik fungsi alih yang dibuat.

2.4. Kontroler PID

Kontroler mempunyai perkembangan yang baru dari segala aspek dan mempunyai teknik kontrol yang dapat berkembang lebih lagi selama

tiga atau empat dekade terakhir ini. Kontroler PID sering digunakan dalam dunia aplikasi kontrol, seperti di industri. Sebab kontrol ini termasuk kontrol yang modern dalam segi teknik yang sering digunakan, memiliki kemudahan dan kontrol sederhana [4]. Kontroler PID (Proposional Integral *Derivative*) merupakan satu-satunya strategi yang paling banyak diadopsi pada pengontrolan variabel proses di industri. Berdasarkan survey, dijumpai kenyataan bahwa 97% industri yang bergerak dalam bidang proses (seperti industri kimia, makanan, minyak dan gas) menggunakan PID sebagai komponen utama dalam pengontrolannya [5]. Kepopuleran PID sebagai komponen kontrol proses dilatarbelakangi terutama oleh kesederhanaan struktur, serta kemudahan dalam melakukan *tuning* parameter kontrolnya. Selain sederhana, kepopuleran PID disebabkan juga oleh alasan historis. Dalam hal ini, PID telah diterapkan di industri secara luas jauh sebelum era digital berkembang, yaitu dimulai sekitar tahun 1980-an, di mana saat itu strategi kontrol PID diimplementasikan dengan menggunakan rangkaian elektronika analog, bahkan banyak diantaranya direalisasikan dengan menggunakan komponen mekanis dan pneumatik murni [6]. PID *controller* merupakan algoritma metode kendali yang terdiri dari tiga konsep matematika yaitu proporsional, integral, dan *derivative*. PID merupakan algoritma kontrol yang paling sering digunakan. PID adalah kontroler yang berbasis *negative feedback control*. Untuk kontroler PID, sinyal kesalahan *e(t)* merupakan masukan kontroler sedangkan keluaran kontroler adalah sinyal kontrol *u(t)* [7]. Hubungan antara masukan kontroler *e(t)* dan keluaran kontroler *u(t)* dapat ditulis dalam persamaan fungsi alih dari kontroler PID ditunjukkan seperti pada Persamaan (2.30).

$$u(t) = K_p\left(e(t) + \frac{1}{T_i}\int_0^t e(t)dt + T_d\frac{de(t)}{dt}\right) \tag{2.30}$$

Persamaan tersebut didapat dari tiga konsep kontroler yaitu kontroler proporsional, integral dan *derivative*. Ketiga kontroler tersebut mempunyai persamaan fungsi alih sendiri, dimana persamaan tersebut yang menjadi persamaan fungsi alih kontrol PID. Kontroler proporsional merupakan kontroler yang aksi kontrolnya proporsional terhadap sinyal kesalahan [7]. Karakteristik kontroler proporsional dideskripsikan dengan mengetahui batas sinyal kontrol maksimum dan minimum, serta nilai *band* proporsional yaitu rentang *error*, maka secara matematis dapat dituliskan pada persamaan (2.31).

$$u_{\max} - u_{\min} = KP_b \tag{2.31}$$

Kontroler proporsional yang semakin besar akan memperkecil kesalahan keadaan tunak, namun akan membawa sistem ke daerah tidak stabil, dan memperkecil rentang operasi nilai *set point*, sehingga kondisi ini tidak memungkinkan untuk membuat sebuah sistem menjadi *zero error steady state*, sehingga dikembangkan kontroler integral. Aksi kontroler integral digunakan untuk meyakinkan bahwa respon sistem memiliki nilai kesalahan keadaan tunak sama dengan atau berada direlatif nol. Selama respon sistem memiliki nilai kesalahan, walaupun kecil, secara matematis akan mengakumulasi nilai kesalahan yang mengakibatkan peningkatan nilai sinyal kontrol. Oleh karena itu, aksi kontroler integral akan selalu berusaha untuk menghasilkan respon nilai kesalahan keadaan tunak nol, sehinggga mendapatkan nilai aksi kontrol yang konstan [7]. Secara matematis dapat sinyal kontrol keadaan tunak dapat dituliskan pada persamaan (2.32).

$$u(t) = \frac{1}{T_i}\int_0^t e(t)dt \tag{2.32}$$

Konsekuensi dengan memperbesar *gain* integral atau memperkecil konstanta waktu integral dapat membuat sistem lebih cepat "mengejar" nilai keadaan tunak, namun sistem cenderung akan berosilasi. Dalam keadaan seperti ini, diperlukan sebuah metode yang dapat memperkecil amplitudo osilasi dan nilai *overshoot* dari respon.

Secara intuitif, respon sistem yang berosilasi disebabkan pada beberapa hal. Proses dinamik dari sebuah *plant* menyebabkan respon sebuah *plant* tidak langsung berubah dengan adanya perubahan sinyal kontrol, namun membutuhkan waktu proses. Waktu ini akan membuat sistem kontrol mengalami keterlambatan untuk mengkoreksi kesalahan. Untuk itu dibutuhkan kontroler yang dapat memprediksi kesalahan dari sebuah sistem. Nilai kesalahan prediksi dapat diturunkan dari persamaan taylor, secara matematis dapat dituliskan pada persamaan (2.33).

$$e(t+Td) \approx e(t) + T_d \frac{de(t)}{dt} \tag{2.33}$$

Dengan mengatur gradien dari garis prediksi, maka akan diperoleh kesalahan prediksi yang lebih akurat, dapat dilihat pada persamaan (2.34).

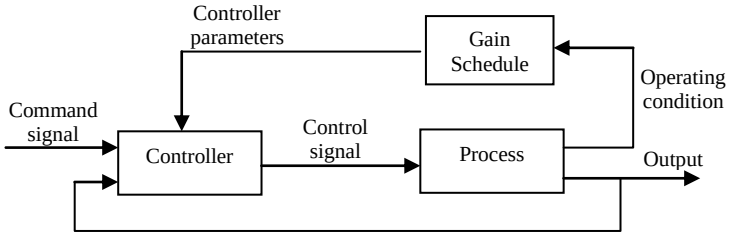
$$u(t) = K_p(e(t) + T_d \frac{de(t)}{dt}) \quad (2.34)$$

Sebagai validasi, seperti yang telah dipaparkan bahwa osilasi dari respon sistem terjadi karena sinyal kontrol berlebihan yang diberikan ke *plant*, maka diperlukan mekanisme untuk mengurangi nilai sinyal kontrol yang berlebihan tersebut. Kontroler derivatif yang ditambahkan pada kontroler proporsional dapat mengurangi aksi kontrol yang berlebihan.

Dari Persamaan (2.34), dapat dilihat bahwa keadaan awal saat sistem memiliki kesalahan positif maka kontroler proporsional akan berusaha mengejar nilai *set point*. Dengan adanya respon tersebut, maka perubahan kesalahan proporsional dengan negatif kesalahan. Hal ini akan membuat kontroler derivatif akan berusaha meredam aksi kontroler proporsional yang berlebihan. Berkurangnya nilai sinyal kontrol akan berdampak pada kecepatan respon sistem yang semakin berkurang [7].

2.5. Metode *Gain Scheduling*

Persepsi *gain scheduling* dapat beragam tergantung dari desain yang dibuat. Secara umum nilai *gain* yang bergantung dari variabel dapat dikatakan sebagai *gain scheduling*. Jika diimplementasikan dalam kontroler PID, seperti terlihat pada Persamaan (2.30), bahwa parameter proporsional, integral dan derivatif kontroler berubah tergantung variabel tertentu. Variabel tertentu yang berubah kontinyu terhadap waktu akan mempengaruhi nilai dari *gain* kontroler PID. Untuk mendapatkan nilai parameter dari kontroler PID terdapat beberapa cara. Salah satu caranya adalah membagi sampel yang sangat banyak dari variabel yang mempengaruhi terhadap nilai parameter kontroler. Oleh, karena itu metode *gain scheduling* harus membagi variabel menjadi beberapa nilai operasi [8]. Prinsip dari *gain scheduling* dapat disederhanakan dalam bentuk blok diagram pada Gambar 2.6.



Gambar 2.6. Diagram blok sistem *gain scheduling*

Kontroler PID *gain scheduling* berdasarkan pengukuran dari kondisi pada *plant*. Suatu plant mempunyai kondisi yang berbeda atau mengalami perubahan, jika sistem tersebut non linier. Kontroler tersebut akan mengontrol kondisi dari *plant* tersebut agar tetap stabil pada kondisi yang diinginkan dalam sistem yang linier, sehingga *plant* yang mempunyai kondisi yang berbeda akan diberikan nilai *gain* pada tiap komponen kontroler PID yaitu proposional, integral dan *derivative* berbeda-beda sesuai dengan kondisi pada plant. *Gain scheduling* mempunyai nilai kontrol yang berbeda-beda pada kondisi *plant* yang berubah, sehingga nilai kontroler PID *gain scheduling* terjadwal sesuai dengan kondisi-kondisi yang sudah ditentukan.

2.6 OLE *for Process Control* (OPC) [9]

Perkembangan teknologi konektivitas OPC untuk tukar menukar informasi antara *hardware* di *plant* dengan *software* untuk *monitoring point* diawali oleh perkembangan standar komunikasi yang bernama OLE. OLE (*Object Linking and Embedding*) merupakan suatu teknologi yang dikembangkan oleh Microsoft untuk menghubungkan suatu objek atau dokumen dengan objek yang lainnya. Dengan adanya OLE seorang editor dapat mengambil sebagian informasi dari suatu dokumen dan mengimpor informasi tersebut ke aplikasi editor yang lain. Sebagai contoh, seorang editor dapat melakukan transfer data antar aplikasi yang berbeda menggunakan operasi "*drag and drop*" atau dengan operasi "*clipboard*".

Sedangkan OPC (*OLE for Process Control*) adalah OLE yang didesain untuk menjembatani berbagai aplikasi berbasis Windows dengan *hardware* maupun *software* untuk kontrol proses. OPC dikembangkan oleh industri otomasi sebagai suatu standar terbuka yang menjelaskan tentang komunikasi data *real-time* dari *plant* antar perangkat kontrol yang diproduksi oleh manufaktur yang berbeda. Sebelum OPC ditemukan, setiap perangkat kontrol memerlukan driver yang spesifik agar dapat berkomunikasi dengan aplikasi *client* yang membutuhkan data dari *hardware* kontrol tersebut. Hal ini akan menjadi tidak efisien ketika perangkat yang digunakan di *plant* sangat banyak dan berasal dari manufaktur yang berbeda merek. Karena biasanya suatu alat dari manufaktur tertentu memiliki cara komunikasi yang berbeda dengan alat yang diproduksi oleh manufaktur yang lain walaupun memiliki fungsi yang sama.

OPC memberikan satu standardisasi komunikasi yang berlangsung antara perangkat dengan aplikasi, sehingga *client* tidak perlu lagi

mendefinisikan format komunikasinya sendiri-sendiri untuk masing-masing perangkat yang berbeda *vendor*. Standardisasi yang dilakukan OPC adalah pada teknologinya, bukan pada alatnya sehingga tidak perlu dilakukan perubahan fisik pada *hardware*.

2.7 Perangkat Lunak LabVIEW 2012

Perangkat lunak LabVIEW 2012 merupakan sebuah sistem yang dikembangkan untuk pengukuran di bidang perindustrian, eksperimen, ataupun pendidikan dan aplikasi otomasi yang berdasarkan pemrograman secara gambar, yang berbeda dengan pemrograman secara teks. Akan tetapi, pemrograman secara teks juga bisa dilakukan di LabVIEW 2012. LabVIEW 2012 mempunyai banyak fungsi-fungsi untuk analisa numerik, desain dan menggambarkan hasil data.

LabVIEW 2012 mempunyai beberapa *toolkit* dan modul yang membuat fungsi LabVIEW 2012 setingkat dengan *Matlab* dan *Simulink* di dalam analisa dan desain kontrol, pengolahan sinyal, identifikasi sistem, matematika, simulasi, dan lain-lain.

Pada saat pertama kali menggunakan perangkat lunak ini maka akan muncul *Window Getting Started* seperti pada Gambar 2.7.

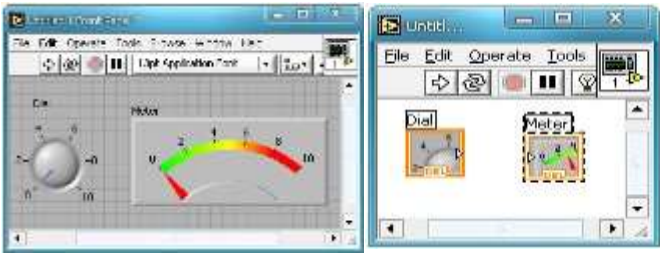
Gambar 2.7 Tampilan awal *software*

Di tampilan awal LabVIEW 2012, kita dapat melakukan:

- Membuka program yang sudah ada dengan cara klik salah satu *file* di kolom *Open*, atau klik *File/Open* di bagian menu kiri atas, atau dengan tombol *shortcut* Ctrl+O.

- Melihat LabVIEW 2012 *Help* dengan cara mengklik pada *toolbar Help*, atau klik *Help/Search the LabView Bookshelf* di bagian menu kiri atas, atau dengan tombol *shortcut* Ctrl+?.
- Mencari contoh-contoh program dengan cara mengklik pada *Find Example* di kolom *Help*.
- Membuat program baru dengan cara mengklik *Blank VI* (VI singkatan dari *Virtual Instrument*) pada kolom *New*, atau klik *File/New VI* di bagian menu kiri atas, atau dengan tombol *shortcut* Ctrl+N.

Untuk memulai program baru, maka akan muncul tampilan *Front Panel*, yaitu tempat untuk membuat tampilan dari program yang akan dibuat, biasanya digunakan untuk *display, indicator, tuning, switch,* dan yang lainnya. Tampilan f*ront panel* ditunjukkan pada Gambar 2.8.



Gambar 2.8 Hubungan *front panel* dan *block diagram*

Selain itu, juga menampilkan *Block Diagram*, bagian ini digunakan untuk membuat fungsi atau jalannya program dari tombol dan *display* yang ada pada *Front Panel*.

2.8 Pengenalan *Plant* dan Perangkat Keras

Untuk mendapatkan data dari suatu *plant* dibutuhkan pengenalan terhadap *plant* yang akan digunakan. Selain itu, perangkat keras pendukung sistem tersebut perlu dikenali terlebih dahulu sebelum menggunakannya.

2.8.1 *Process Control Technology-100* [10]

PCT-100 merupakan modul *training* kontrol proses yang dilengkapi dengan beberapa sensor dan aktuator untuk aplikasi dari suatu proses. Sensor yang terdapat pada PCT-100 yaitu sensor *level, temperature, flow,* dan *pressure*, sedangkan aktuatornya berupa *pump,*

valve, heating dan *cooling*. *Plant* kontrol proses dapat dilihat pada Gambar 2.9.

Gambar 2.9. *Process Control Technology*-100 (PCT-100)

Pada PCT-100 dapat diaplikasikan berbagai operasi kerja kontrol proses, tetapi secara umum proses dari *plant* tersebut adalah air yang berada pada tangki penampung air akan dipompa ke tangki ukur dengan volume 4,5 liters, didalam tangki penampung suhu air dapat diukur dengan sensor PRT (*Platinum Resistance Thermometer*). Pompa yang digunakan adalah pompa DC dengan tegangan 24 volt. Air yang dialirkan ke dalam tangki akan melewati *diverter valve* terlebih dahulu, fungsi dari *diverter valve* sebagai *valve* pengalir untuk di alirkan ke salah satu jalur yaitu antara langsung menuju tangki air atau melalui mesin pendingin terlebih dahulu. Setelah melewati *diverter valve*, air yang dialirkan melewati *propotional valve*. Untuk *propotional valve* pengaturan yang dilakukan berbeda dengan *diverter valve*, *propotional valve* yaitu dengan sistem analog, sedangkan untuk *diverter valve* secara sistem digital. Dalam sistem PCT-100 juga terdapat *flow sensor* yang berfungsi untuk mengetahui kecepatan dari debit air yang masuk ke dalam tangki. Air didalam tangki dapat diukur ketinggian air dengan *level transducer* yaitu *magnestostrictive position sensor*. Tangki air tersebut juga terdapat *heating elemen power* yang berfungsi untuk memanaskan air dan sensor PRT untuk mengetahui suhu air dalam tangki. Ketika air didalam tangki air dipanaskan tentu akan ada tekanan dalam tangki tersebut yang dapat diketahui dengan menggunakan *pressure sensor* untuk mengetahui kondisi tekanan dalam tangki air.

Selain itu juga terdapat *drain valve* yang berfungsi untuk *valve* pengeluaran air dari tangki air.

2.8.2 Perangkat keras *Advantech* ADAM 5000 TCP

Perangkat keras *ADAM 5000 TCP* pada Tugas Akhir ini, digunakan sebagai pengontrol / eksekutor untuk menjalankan proses. Hal ini dikarenakan, perangkat keras *ADAM 5000 TCP* bekerja setelah diberikan masukan dari PC *Human Interface Station*. Perangkat keras *ADAM 5000 TCP* lebih banyak digunakan sebagai I/O dari proses kontrol yang dirancang. Untuk menjalankan alat ini, perintah dikirim melalui jaringan *ethernet* dengan komunikasi *modbus.* Gambar 2.10 menunjukan bentuk dari perangkat keras *ADAM 5000 TCP*.

Gambar 2.10. Perangkat keras *Advantech ADAM 5000 TCP*

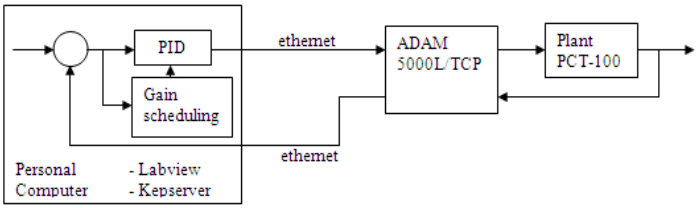
Advantech ADAM 5000 TCP terdiri empat modul, yaitu modul 5050, modul 5024, modul 5024 dan modul 5017. Modul 5050 merupakan masukan atau keluaran digital, sedangkan modul 5024 dan modul 5017 merupakan keluaran dan masukan analog. Pada perancangan sistem kontrol proses ini, hanya menggunakan dua modul. Dua modul yang digunakan yaitu, modul *ADAM* 5024 yang merupakan modul keluaran analog. Modul *ADAM* 5024 terdiri dari empat *channel* untuk memberikan masukan berupa arus dan empat *channel* masukan berupa tegangan. Modul lain yang digunakan pada perancangan sistem kontrol proses, yaitu modul *ADAM* 5017 yang merupakan modul masukan analog. Pada modul ini, terdiri dari delapan *channel* yang digunakan untuk melihat besarnya nilai keluaran dari *plant*.

BAB III
PERANCANGAN SISTEM

Pada bab ini akan dibahas mengenai perancangan sistem meliputi arsitektur sistem, identifikasi *plant* proses *level* dari data yang sudah diambil secara *open loop* pada beberapa titik kerja. Dari model *plant* tersebut akan dirancang kontroler PID dengan metode *gain scheduling* untuk mengatur keluaran sistem sesuai dengan karakteristik yang ditentukan. Pada akhir Bab ini, ditunjukkan blok Simulink MATLAB untuk simulasi dan menggunakan Labview untuk implementasi sistem proses *level* pada PCT-100.

3.1 Arsitektur Sistem Kontrol Proses

Untuk merealisasikan sistem kontrol proses secara menyeluruh, diperlukan suatu gambaran dari arsitektur sistem yang akan dirancang. Rancangan tersebut, harus mencakup semua peralatan yang dibutuhkan dan menunjukkan jalannya suatu proses serta diagram jaringan secara keseluruhan. Bentuk arsitektur dari sistem kontrol proses yang dibangun ditunjukkan dalam Gambar 3.1 dalam bentuk desain diagram blok.



Gambar 3.1. Diagram blok kontrol sistem proses *level* pada PCT-100

Perancangan ini meliputi konfigurasi antara perangkat keras yaitu PCT-100 proses *level* dengan perangkat lunak yang digunakan yaitu *software* Labview dan Kepserver sebagai OPC. Komputer yang digunakan pada Tugas Akhir ini, memiliki spesifikasi Prosessor Intel(R) Pentium(R) 4 CPU 1.80GHz 1.80GHz, memori 1 GB, dan operating system Windows 7 Ultimate 32-bit. Komputer Labview 2012 memiliki fungsi sebagai kontroler yang dihubungkan dengan PCT-100 menggunakan perangkat keras Advantech ADAM 5000 TCP. Dari komputer ini, digunakan untuk pengiriman dan penerimaan data. Pengiriman data ini berupa referensi atau *set point level* pada *plant*, sedangkan untuk penerimaan data, berupa data yang diambil dari *plant* melalui sensor *level* PCT-100.

3.2 Kebutuhan Sistem

Definisi kebutuhan digunakan agar untuk mengetahui sistem seperti apakah yang dibutuhkan dalam realita. Definisi kebutuhan meliputi *survey* kebutuhan yang menjelaskan karakteristik sistem dan alasan mengapa karakteristik sistem tersebut dibutuhkan.

3.2.1 *Advantech* ADAM 5000 TCP

Perangkat keras ADAM 5000 TCP pada Tugas Akhir ini, digunakan sebagai I/O untuk menjalankan proses. Hal ini dikarenakan, perangkat keras ADAM 5000 TCP bekerja setelah diberikan masukkan dari komputer Labview 2012. Nilai yang dibaca pada perangkat keras ADAM 5000 TCP adalah nilai desimal. Untuk menjalankan alat ini, perintah dikirim melalui jaringan *ethernet* dengan komunikasi *modbus*.

3.2.2 *Plant Level Process* PCT-100

Pengerjaan Tugas Akhir ini menggunakan *plant level process* PCT-100. *Plant* ini digunakan untuk mendapatkan data dari sistem kontrol proses yang telah dibangun. *Plant level process* PCT-100 dipilih karena dianggap sebagai salah satu bentuk proses yang terdapat di industri nyata. Pada *plant* ini, nantinya akan dikontrol pengisian air yang masuk dalam tangki dengan pompa. Untuk mendapatkan data dari *plant* proses tersebut, data dari komputer Labview dikirimkan melalui jaringan, menuju perangkat keras dari ADAM 5000 TCP. Data yang terkirim pada perangkat keras ADAM 5000 TCP, kemudian dikirimkan menuju *plant level process* PCT-100. Untuk mendapatkan sistem kontrol proses, data keluaran dari *plant level process* PCT-100 dikirimkan kembali ke perangkat keras ADAM 5000 TCP, kemudian dikirimkan untuk di proses kembali pada komputer Labview.

3.2.3 Perangkat Lunak OLE *for Process Control*

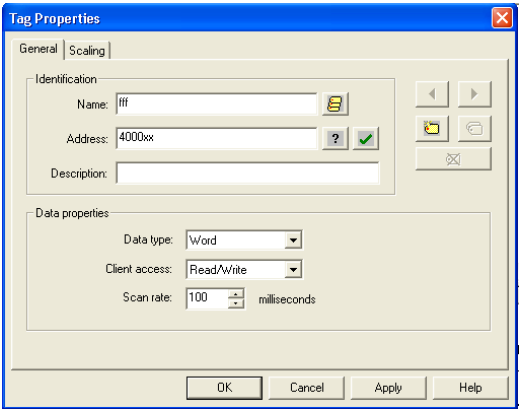
OLE for process control (OPC), merupakan perangkat lunak yang digunakan sebagai standar industri untuk interkonektivitas sistem. OPC siap untuk berbagai antarmuka, sehingga berbagai data yang datang dari beragam sumber seperti PLC, DCS, *analyzer*, *gauge*, semuanya akan diterima oleh berbagai aplikasi dalam format yang sama. Penggunaan OPC pada pengerjaan Tugas Akhir ini sebagai penghubung untuk pengiriman data dari PC Labview menuju perangkat keras *Advantech ADAM 5000 TCP*. Setelah data tersebut terkirim, data tersebut digunakan untuk mengontrol *plant.* OPC yang digunakan adalah OPC *KEPServer*, berfungsi untuk menghubungkan *tag name* yang dibuat pada PC Labview dengan perangkat keras ADAM 5000 TCP.

3.2.4 Koneksi Labview 2012 dengan ADAM 5000 TCP

Koneksi dari Labview 2012 dengan *ADAM 5000 TCP*, menggunakan perangkat lunak OPC. Hal ini dilakukan, karena untuk menghubungkan kedua perangkat yang berbeda *vendor*, membutuhkan perangkat lunak OPC tersebut. Langkah pertama dalam koneksi ini, membuat program konversi untuk nilai masukkan dan keluaran *ADAM 5000 TCP*. Hal ini di perlukan, karena pada *ADAM 5000 TCP* nilai yang terbaca merupakan nilai desimal (DEC). Agar nilai desimal yang terbaca pada modul *ADAM 5000 TCP*, diperlukan pengujian untuk mengetahui berapa nilai desimal yang di perlukan untuk mengeluarkan tegangan dari modul analog pada *ADAM 5000 TCP*. Setelah menentukan besarnya nilai desimal untuk modul masukkan dan keluaran analog dari *ADAM 5000 TCP*, dibuat program konversi dari Labview.

Setelah program konversi telah di buat, berikutnya mengoneksikan Labview dengan *ADAM 5000 TCP*. Proses penggabungan Labview dan ADAM 5000 TCP menggunakan protokol *modbus ethernet*. Protokol ini digunakan untuk mengirimkan data ke *Advantech ADAM 5000 TCP*. Hal ini dikarenakan, pada *Advantech ADAM 5000 TCP* menggunakan komunikasi *modbus.*

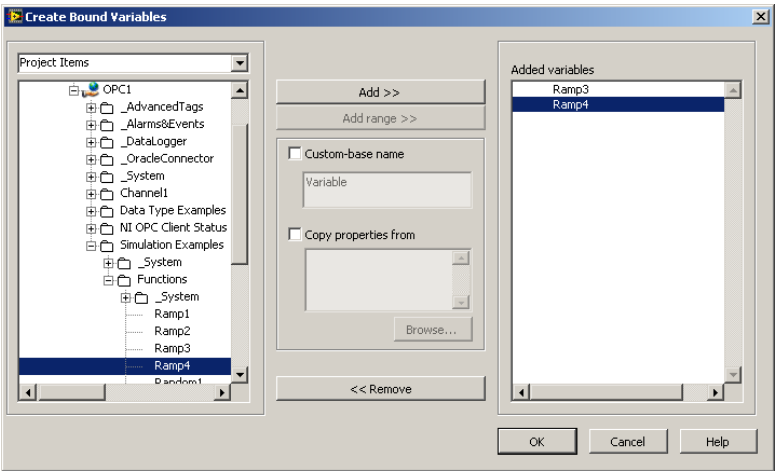
Pada pilihan *modbus ethernet*, diperlukan pembuatan *tag name* untuk *channel* yang akan digunakan pada modul ADAM 5024 dan modul ADAM 5017, karena masing-masing *channel* dari modul tersebut memiliki alamat. Gambar 3.2 menunjukkan *setting tag name* untuk setiap *channel* pada *Advantech ADAM 5000 TCP*.



Gambar 3.2. Pembuatan *tag name* pada OPC *KEPServer*

Setelah pembuatan *tagname* selesai, langkah yang dilakukan adalah menghubungkan Labview dengan OPC tersebut. Hal ini dilakukan, karena nilai yang dibaca pada *ADAM 5000 TCP* juga terbaca pada OPC *KEPServer*.

Pada Labview terdapat I/O server yang sudah dibuat terlebih dahulu pada file project, fungsi ini bertujuan untuk menghubungkan nilai yang terbaca pada Labview dengan *tagname* pada OPC. Nilai yang terbaca pada Labview, nantinya akan terbaca pula pada *ADAM 5000 TCP*. Gambar 3.3 menunjukan pilihan untuk menggabungkan Labview dengan OPC.



Gambar 3.3. Mengambil *tag name* OPC pada Labview

Tag name yang sudah dibuat dapat langsung didrag atau diseret ke layar *front panel* atau *blok diagram panel*. Blok yang ada pada *blok diagram* dari hasil *tag name* akan berfungsi secara *real time* dengan nilai yang dikeluarkan oleh ADAM 5000 TCP sesuai dengan nilai pada Labview.

3.3 Proses Identifikasi *Plant*

Dalam mengembangkan maupun mendesain kontroler untuk suatu *plant*, diperlukan pengetahuan mengenai sifat dan karakteristik dari *plant* tersebut. Hal itu dapat diketahui dari fungsi alih yang dimiliki oleh suatu *plant.* Permasalahan yang sering kali muncul ketika menghadapi

suatu *plant* adalah tidak adanya dokumen yang memberikan informasi fungsi alih dari *plant*.

Proses identifikasi dilakukan secara *open loop*, sinyal uji diberikan dari komputer kontroler dan dikirim ke *plant*. Respon dari *plant* terhadap sinyal uji yang diberikan diterima oleh ke komputer kontroler. Pada komputer kontroler, keluaran dari *plant* diplot menjadi sebuah grafik. Dari grafik tersebut, bisa didapatkan nilai parameter-parameter *plant*. Identifikasi dilakukan pada kondisi *set point* yang berbeda-beda. Berdasarkan paper dengan penulis Ing. Pavel Jakoubek, terdapat 7 metode identifikasi yang dapat digunakan untuk sistem tanpa osilasi dengan masukan respon *step*. Setelah diperoleh fungsi alih sistem, ketujuh metode ini dibandingkan dengan respon hasil pengukuran untuk mendapatkan metode dengan pendekatan model yang terbaik. Metode validasi yang digunakan adalah ISE (*Integral Square Error*).

3.3.1 Fungsi Alih sistem

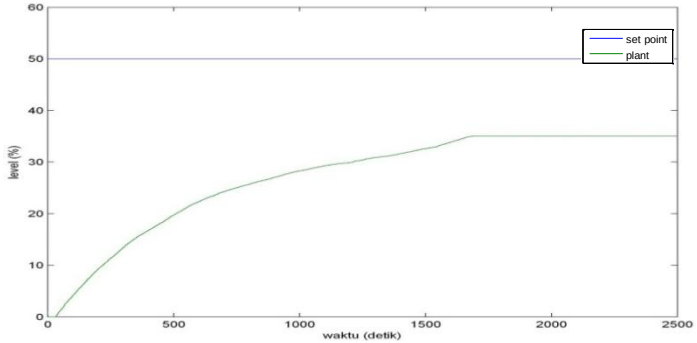
Untuk mendapatkan fungsi alih sistem, respon yang digunakan adalah respon saat *set point* 50%. Respon *level* pada *set point* 50% ditunjukkan pada Gambar 3.4 Pada respon tersebut, diperoleh nilai keluaran *steady state* pada ketinggian 34,97. Nilai tersebut diperoleh dari rata-rata seluruh data keluaran *steady state*.

Berdasarkan respon pada Gambar 3.4, diperoleh *gain overall* respon pada persamaan 3.1.

Y_{ss} = 34,97

X_{ss} = 50

$$K = \frac{Y_{ss}}{X_{ss}} = \frac{34{,}97}{50} = 0{,}6994 \qquad (3.1)$$



Gambar 3.4. Respon *plant* proses *level* pada PCT-100

dari 7 metode Jakoubek untuk identifikasi *plant*, metode Harriott yang digunakan dalam pendekatan model matematika plant yang sesuai, karena pemilihan metode terbaik terhadap data pengukuran dilakukan dengan validasi. Metode validasi yang digunakan adalah ISE *(Integral Square Error)*. Semakin kecil nilai ISE, maka semakin baik fungsi alih yang dibuat. Pendekatan sistem model matematika *plant* proses *level* yaitu berdasarkan respon pada Gambar 3.4, diperoleh :

$t_{33} = 254{,}8$ detik (3.2)

$t_{70} = 712{,}9$ detik (3.3)

$t_{73} = 788{,}6$ detik (3.4)

sehingga diperoleh parameter:

$$T_{dH} = 1{,}937\ x\ 254{,}8 - 0{,}937\ x\ 712{,}9 = -174{,}44 \quad (3.5)$$

$$\tau_{H1} + \tau_{H2} = \frac{788{,}6}{1{,}3} = 606{,}62 \quad (3.6)$$

$$t_1 = \frac{\tau_{H1} + \tau_{H2}}{2} = 303{,}31 \rightarrow y_1 = 13{,}44 \quad (3.7)$$

Berdasarkan kurva Harriott pada Gambar 2.5, diperoleh nilai :

$$\frac{y_1}{Y_{ss}} = \frac{13{,}44}{34{,}97} = 0{,}384 \rightarrow \frac{\tau_{H1}}{(\tau_{H1} + \tau_{H2})} = 0{,}96 \quad (3.8)$$

Sehingga nilai τ_{H1} dan τ_{H2} dapat diketahui, untuk nilai τ_{H1} sesuai dengan persamaan 2.24, sedangkan untuk nilai τ_{H2} adalah persamaan 2.27, diperoleh :

$\tau_{H1} = 18{,}85$ (3.9)

$\tau_{H2} = 587{,}77$ (3.10)

Parameter delay T_{dH} bernilai negatif, sehingga nilai delay T_{dH} tidak dianggap. Fungsi alih untuk metode Harriot adalah:

$$G_H\ s = \frac{0{,}6994}{18{,}85\ s + 1\ (587{,}77\ s + 1)} \quad (3.11)$$

$$G_H\ s = \frac{0{,}6994}{11079{,}46\ s^2 + 606{,}62\ s + 1} \quad (3.12)$$

Dari fungsi alih yang didapatkan, dapat dilihat respon dari model terhadap masukkan sinyal uji yang sama pada proses identifikasi untuk melihat kesesuaian antara model dengan *real plant*. Dari hasil percobaan didapatkan ISE sebesar 0,6656. Model matematika yang didapat merupakan pendekatan orde 2, hal ini pada dunia industri umumnya *plant* proses merupakan sistem orde 2.

3.4 Perancangan Kontroler

Setelah diperoleh model matematika dan konstanta maka dapat dilakukan perancangan kontroler dengan mencari parameter-parameter kontroler yang diperlukan.

3.4.1 Kontroler PID

Perancangan kontroler PID dilakukan secara analitik dari parameter fungsi alih yang diperoleh dari hasil identifikasi. Parameter fungsi alih tersebut menentukan parameter K_p, τ_i dan τ_d. Untuk parameter fungsi alih *plant*, diberikan persamaan sebagai berikut:

$$G\ \ s\ \ = \frac{K}{\frac{1}{\omega n^2}s^2 + \frac{2\zeta}{\omega n}s + 1} \tag{3.13}$$

Berdasarkan persamaan 3.13 dengan model matematika plant yang didapat, maka dengan perhitungan berikut, parameter fungsi alih plant pada persamaan 3.12 diperoleh:

$$K = 0{,}6994 \tag{3.14}$$

$$\frac{1}{\omega n^2} = 11079{,}46 \rightarrow \omega n = \sqrt{\frac{1}{11079{,}46}} = 0{,}0095 \tag{3.15}$$

$$\frac{2\zeta}{\omega n} = 606{,}62 \rightarrow \zeta = \frac{606{,}62\ x\ 0{,}0095}{2} = 2{,}88 \tag{3.16}$$

Untuk perancangan kontroler PID pada *plant* proses *level* dengan *set point* 50, nilai τ yang diinginkan (τ*) adalah 400 detik. Sehingga, diperoleh parameter kontroler sebagai berikut:

$$\tau^* = 400 \tag{3.17}$$

$$\tau_d = \frac{1}{2\ \omega n\ \zeta} = \frac{1}{2\ x\ 0{,}0095\ x\ 2{,}88} = 18{,}275 \tag{3.18}$$

$$\tau_i = \frac{2\zeta}{\omega n} = 606{,}62 \tag{3.19}$$

$$K_p = \frac{2\zeta}{\tau^* \, \omega n \, K} = \frac{2 \; x \; 2{,}88}{400 \; x \; 0{,}0095 \; x \; 0{,}6994} = 2{,}167 \tag{3.20}$$

$$K_i = \frac{K_p}{\tau_i} = \frac{2{,}167}{606{,}62} = 0{,}0036 \tag{3.21}$$

$$K_d = K_p \; x \; \tau_d = 39{,}60 \tag{3.22}$$

Hasil perancangan kontroler PID pada kondisi *set point* yang ditentukan diberikan pada Tabel 3.1.

Tabel 3.1 Parameter Kontroler PID

Parameter	25%	50%	75%	100%
K_p	1,083	2,167	3,254	4,337
τ_i	0,0018	0,0036	0,0054	0,0071
τ_d	19,74	39,60	59,46	79,25

3.4.2 Kontroler PID *Gain Scheduling*

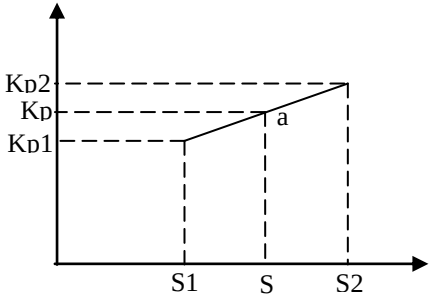
Secara umum nilai *gain* yang bergantung dari variabel dapat dikatakan sebagai *gain scheduling*. Jika diimplementasikan dalam kontroler PID, bahwa parameter proporsional, integral dan derivatif kontroler berubah tergantung variabel tertentu. Variabel tertentu yang berubah kontinyu terhadap waktu akan mempengaruhi nilai dari *gain* kontroler PID. Untuk mendapatkan nilai parameter dari kontroler PID terdapat beberapa cara. Salah satu caranya adalah membagi sampel yang sangat banyak dari variabel yang mempengaruhi terhadap nilai parameter kontroler. Oleh, karena itu metode *gain scheduling* harus membagi variabel menjadi beberapa nilai operasi.

Nilai operasi ini dibagi menjadi 4 kondisi nilai masukan yang berbeda – beda. Nilai operasi ini dibuat dalam bentuk tabel, di mana tabel tersebut terdapat nilai parameter dari kontroler PID yang berbeda pada tiap kondisinya. Tabel tersebut dapat dilihat pada Tabel 3.2.

Tabel 3.2 Parameter kontroler PID pada kondisi yang berbeda

No.	Kondisi *set point (SP) level*	*Kp*	*Ki*	*Kd*
1.	0% ≤ SP ≤ 25%	1,083	0,0018	19,79
2.	25% > SP ≤ 50%	2,167	0,0036	39,60
3.	50% > SP ≤ 75%	3,254	0,0054	59,46
4.	75% > SP ≤ 100%	4,337	0,0071	79,25

Berdasarkan Tabel 3.2, akan disimulasikan untuk mengetahui hasil respon *plant* yang mengalami perubahan kondisi. Program simulasi untuk metode *gain scheduling* menggunakan pendekatan nilai interpolasi. Mencari nilai interpolasi dengan mengambil dua kondisi *set point*, kemudian diambil nilai tengah tersebut dari grafik yang dibentuk dua nilai kondisi.



Gambar 3.5 Kurva interpolasi linier

Berdasarkan kurva pada Gambar 3.5, menentukan nilai *a* adalah sebagai berikut :

$$\frac{a}{Kp2 - Kp1} = \frac{s - s1}{s2 - s1} \tag{3.23}$$

maka,

$$a = \frac{Kp2 - Kp1}{s2 - s1} \; x \;\; s - s1 \tag{3.24}$$

Dari persamaan 3.24 maka untuk mencari nilai Kp dengan interpolasi adalah :

$$Kp = Kp1 + a \tag{3.25}$$

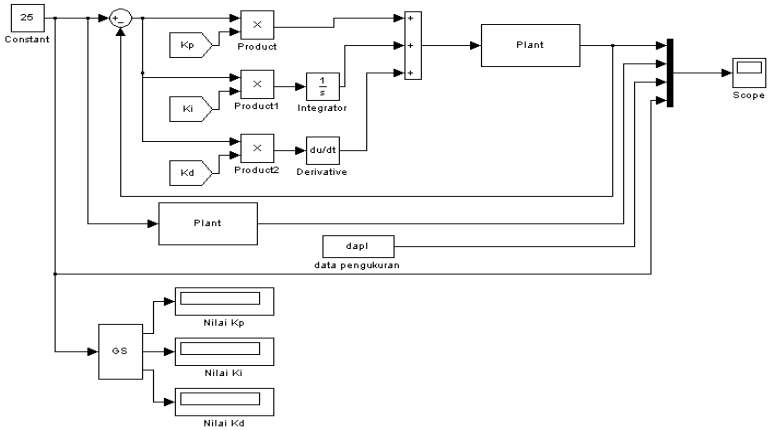
Atau

$$Kp = Kp1 + \frac{Kp2 - Kp1}{s2 - s1}\ x\ \ s - s1 \tag{3.26}$$

Dengan persamaan 3.26, maka program untuk simulasi dapat dibuat. Di mana metode interpolasi dapat menentukan nilai parameter yang sesuai pada kondisi *range* atau batas penjadwalan yang sudah ditentukan.

3.5 Perancangan Simulasi

Perancangan simulasi untuk proses *level* menggunakan aplikasi Simulink MATLAB dapat dilakukan. Di mana setelah menghitung atau menentukan model matematika *plant* dan kontroler PID dengan metode *gain scheduling* terlebih dahulu, maka blok diagram simulink simulasi proses *level* menggunakan kontroler PID dengan metode *gain scheduling* dapat dilihat pada Gambar 3.6.

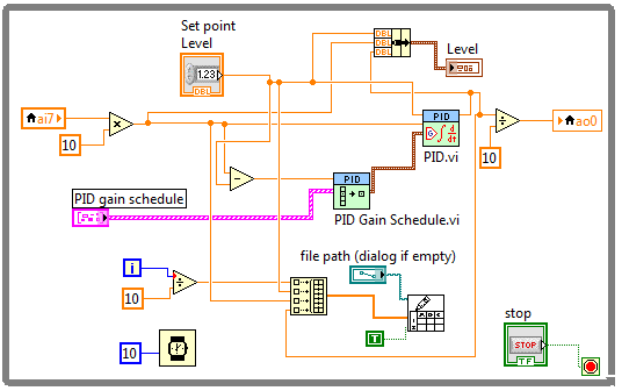


Gambar 3.6. Diagram blok simulink untuk simulasi proses *level*

Simulasi proses *level* akan menunjukkan respon yang dihasilkan dari sistem tersebut. Proses *level* diberi *set point level* yang diinginkan, maka hasil dari nilai parameter kontroler PID dapat ditentukan dengan metode *gain scheduling* yang sudah dibuat, kemudian akan masuk sinyal kontroler tersebut ke model matematika *plant*.

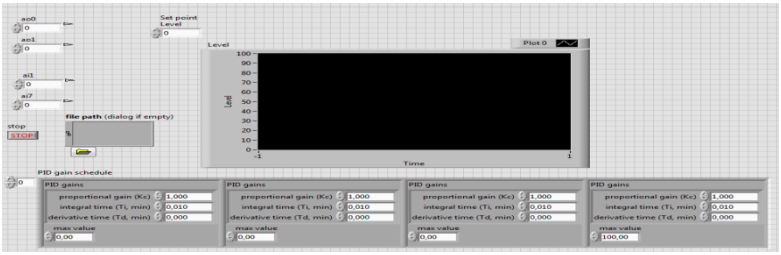
3.6 Perancangan Implementasi

Dalam merealisasikan sistem, perancangan implementasi juga sangat diperlukan. Perancangan implementasi pada Tugas Akhir ini yaitu pembuatan program *close loop* dengan kontroler PID *gain scheduling* menggunakan Labview 2012. Gambar 3.7 menunjukkan program kontrol proses *level*. Program Labview 2012 merupakan jenis program dengan menggunakan blok diagram, tetapi ada beberapa bagian dari program dapat berupa *list programme* atau *structure list programme*.



Gambar 3.7 Program Labview kontrol proses *level*

Sedangkan tampilan untuk kontrol proses level tersebut, dapat dilihat di front panel pada Labview. Gambar 3.8 menunjukkan tampilan kontrol proses *level*. Program yang dibuat mengggunakan fungsi *toolkit PID and Fuzzy* pada Labview. Fungsi dari *toolkit PID and Fuzzy* merupakan fungsi tambahan untuk membuat program PID *gain scheduling*. program yang digunakan merupakan jenis program dengan blok diagram.



Gambar 3.8 Tampilan kontrol proses *level*

--halaman ini sengaja dikosongkan--

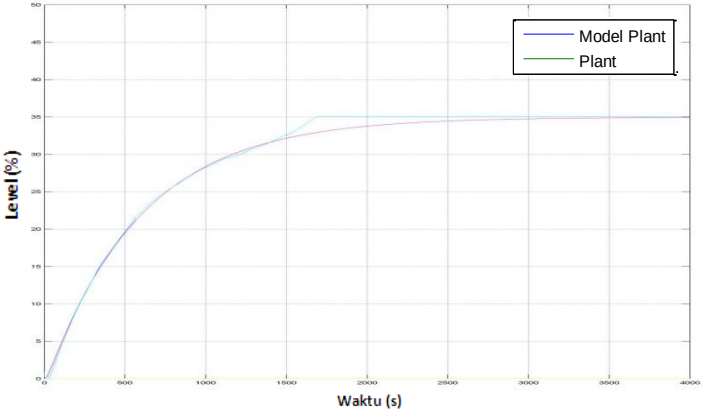
BAB IV
PENGUJIAN DAN ANALISIS

Bab IV membahas tentang hasil pengujian dan analisis dari perancangan yang telah dijelaskan pada bab sebelumnya. Bab ini meliputi pengujian fungsi alih hasil identifikasi, pengujian kontroler secara simulasi dan pengujian kontroler secara implementasi. Pengujian dilakukan untuk mengetahui respon keluaran *plant* dan mengetahui performansi dari sistem secara keseluruhan.

4.1. Pengujian Fungsi Alih Hasil Identifikasi

Pengujian fungsi alih yang sudah diperoleh dari hasil identifikasi, digunakan untuk mengetahui perbandingan antara respon hasil simulasi fungsi alih dan respon hasil pengukuran *plant*. Gambar 4.1 menunjukkan perbandingan antara respon hasil pengukuran *plant* dengan respon fungsi alih saat kondisi *set point* 50%.

Pada respon yang didapat menghasilkan τ sebesar 606,8 detik, t_s (± 5%) sebesar 1820,4 detik, t_r (5% - 95%) sebesar 1786,7 detik, t_d sebesar 420,6 detik. Respon ini memberikan keluaran *steady state* 34,97, sehingga nilai *error steady state* adalah 30,06%.



Gambar 4.1 Perbandingan respon hasil pengukuran dan fungsi alih

Berdasarkan hasil respon fungsi alih, jika kondisi *set point* diubah maka tiap kondisi memiliki kesalahan yang berbeda. Tabel 4.1 menunjukkan hasil kesalahan dari pengujian tiap kondisi. Spesifikasi kesalahan dari tiap kondisi yang akan dikontrol. Fungsi alih tersebut

memiliki respon yang lambat. Dari model *plant* yang didapat, maka dapat dibuat sebuah kontroler yang agar sesuai dengan *set point.*

Tabel 4.1 Hasil kesalahan pada tiap kondisi

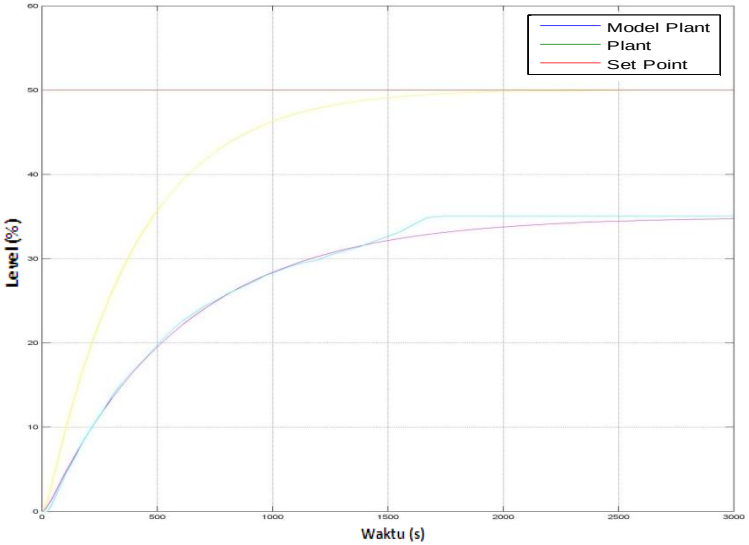
Set Point	*Gain Overall* (*K*)	*Error Steady State* (e_{ss})
25 %	1,3998	0,3988
50 %	0,6994	0,3006
75 %	0,4662	0,5338
100 %	0,3497	0,6503

Karakteristik model *plant* yang didapat pada tiap kondisi *set point* untuk nilai *gain overall* berbeda-beda, tetapi untuk nilai respon *transient* mencapai *steady state* pada tiap kondisi yaitu sama. Hal ini dikarenakan spesifikasi dari *plant* proses *level* PCT-100 dengan kondisi yang sama.

4.2 Simulasi Pengujian Kontroler

Sebelum dilakukan implementasi, terlebih dahulu dilakukan simulasi hasil perancangan kontroler untuk mengetahui performansi sistem. Simulasi pengujian kontroler dilakukan dengan perangkat lunak Simulink MATLAB 2009. Pengujian dilakukan pada kondisi *set point* 50%, dengan tegangan awal pompa 2 volt. Nilai tegangan awal tersebut merupakan spesifikasi dari sistem proses *level*. Parameter kontroler PID yang diperoleh pada Tabel 3.2, diuji dengan respon fungsi alih hasil identifikasi.

Berdasarkan Gambar 4.2, simulasi proses *level* akan menunjukkan respon yang dihasilkan dari sistem tersebut. Proses *level* diberi *set point level* 50% maka hasil dari nilai parameter kontroler PID dapat ditentukan dengan metode *gain scheduling* yang sudah dibuat, kemudian akan masuk sinyal kontroler tersebut ke model matematika plant. Hasil simulasi didapat nilai Kp, Ki, dan Kd untuk *set point level* 50% adalah 2,167; 0,0036; 39,6. Respon grafik yang dihasilkan dapat dilihat pada Gambar 4.2. respon yang didapat menghasilkan τ sebesar 404,5 detik, t_s ($\pm$ 5%) sebesar 1213,5 detik, t_r (5% - 95%) sebesar 1191,03 detik, t_d sebesar 280,4 detik. Respon dengan kontroler memberikan keluaran *steady state* 49,89, sehingga nilai *error steady state* adalah 0,08%.



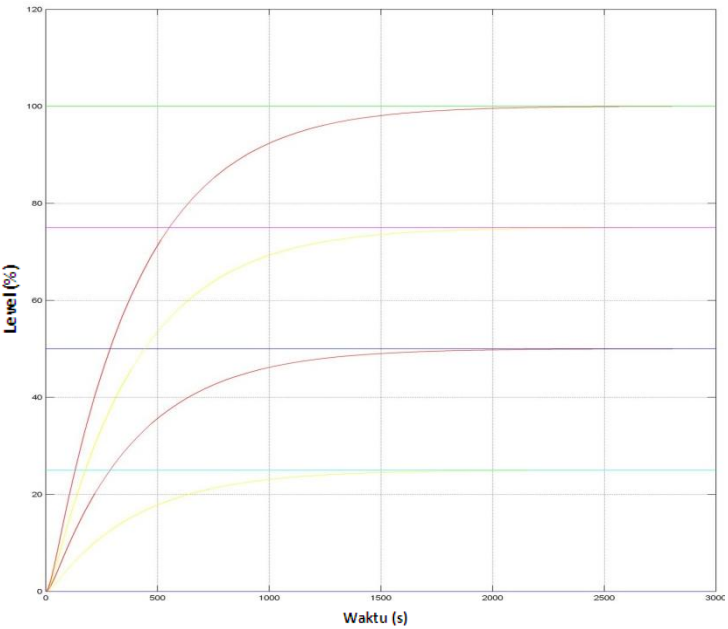
Gambar 4.2 Hasil respon saat kondisi *set point level* 50%

Nilai parameter kontroler PID tersebut yang akan mengendalikan *plant* proses *level* agar sesuai dengan *set point* dan kecepatan respon yang sudah ditentukan. Sinyal model matematika *plant* yang berwarna merah mendekati dengan grafik data pengukuran. Model matematika *plant* tersebut mempunyai kesalahan (e_{ss}) sebesar 30,06% terhadap *set point*. Respon yang dihasilkan tidak ada osilasi atau *overshoot*.

Pada kondisi dan Parameter kontroler PID yang diperoleh, dapat ditunjukan pada Gambar 4.3, hasil respon yang didapat dari sinyal dengan kontroler PID memiliki karakteristik sesuai dengan sistem yang dikehendaki. Kesalahan yang diperoleh pada sistem dengan kontroler PID yang dibuat memiliki kesalahan pada keadaan waktu tunak sangat kecil. Kesalahan sistem pada waktu tunak dapat dilihat pada Tabel 4.2.

Tabel 4.2 Kesalahan waktu tunak pada sistem fungsi alih

Set Point	*Error Steady State* (e_{ss})
25%	0,08%
50%	0,08%
75%	0,04%
100%	0,05%



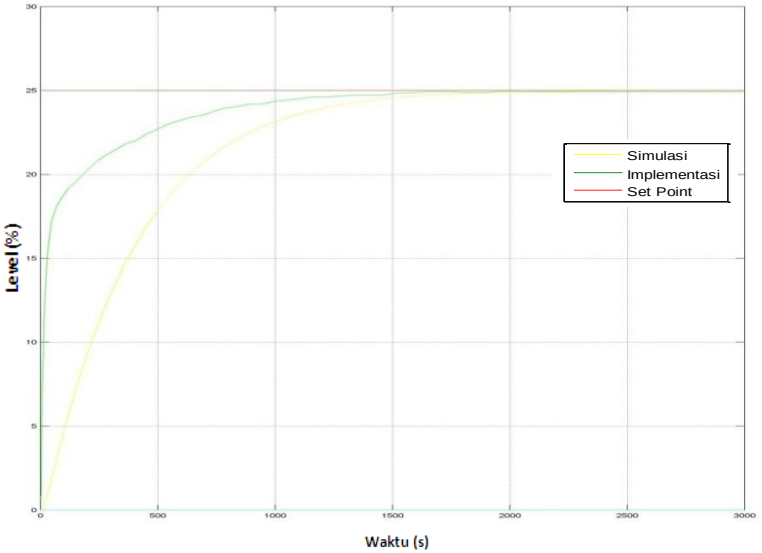
Gambar 4.3 Respon sistem dengan kontroler pada kondisi berbeda

Pada Gambar 4.3, terlihat respon dari sistem *plant* dengan kontroler pada kondisi *set point* yang berbeda-beda memiliki karakteristik respon *transient* yang sama, sehingga perubahan beban pada tangki air untuk pengisian tidak akan membutuhkan waktu yang lama dalam kondisi *steady state* meskipun beban *set point* diubah.

4.3 Implementasi Pengujian Kontroler

Implementasi dilakukan menggunakan perangkat lunak Labview 2012. *Interface* perangkat keras dan perangkat lunak menggunakan Advantech ADAM 5000 TCP serta konfigurasi ADAM dan OPC Kepserver. Implementasi bertujuan untuk mengetahui performansi sistem pada *plant* yang sebenarnya dengan penerapan kontroler PID *gain scheduling*. Program Labview yang ditunjukkan pada Gambar 3.7 merupakan program implementasi dengan *toolkit PID and Fuzzy* pada Labview. Program PID *gain scheduling* pada Labview yang digunakan yaitu program blok dengan memasukkan nilai parameter kontroler dan batas atau syarat kontroler PID pada kondisi tertentu. Pengambilan data sistem proses *level* dengan kontroler PID *gain scheduling* dilakukan

dengan memberi *set point* 25%. Data yang diambil merupakan data aktual dari hasil respon *plant* proses *level* dengan kontroler PID *gain scheduling*. Nilai kontroler PID yang digunakan pada set point 25% yaitu Kp sebesar 1,083, Ki sebesar 0,0018 dan Kd sebesar 19,79. Gambar 4.4 menunjukkan perbandingan hasil respon simulasi dengan implementasi.

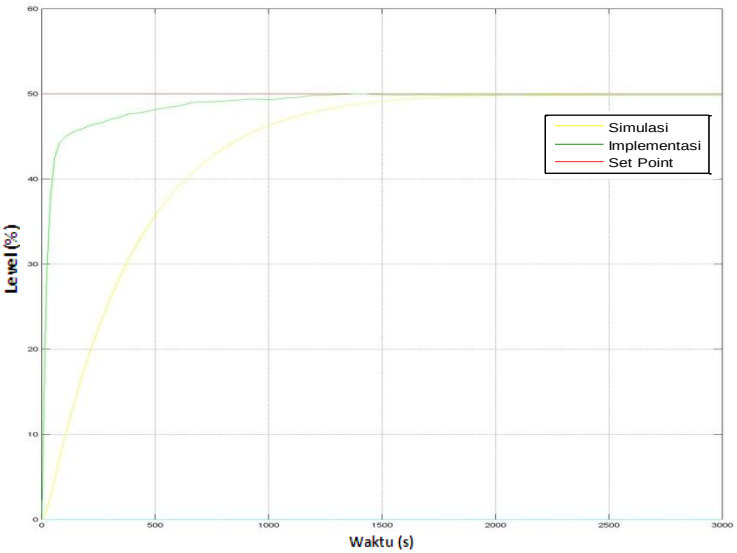


Gambar 4.4 Hasil implementasi *set point* 25%

Berdasarkan hasil respon pada Gambar 4.4, hasil respon implementasi untuk waktu *transient* dan τ lebih cepat dibandingkan dengan simulasi. Pada respon yang didapat menghasilkan τ sebesar 34,7 detik, t_s ($\pm$ 5%) sebesar 104,1 detik, dan t_r (5% - 95%) sebesar 102,17 detik. Respon ini memberikan keluaran *steady state* 24,97. sehingga nilai *error steady state* adalah 0,12%. Hasil respon yang dihasilkan pada *set point* 25% tidak memiliki *overshoot* atau tidak adanya osilasi pada respon tersebut.

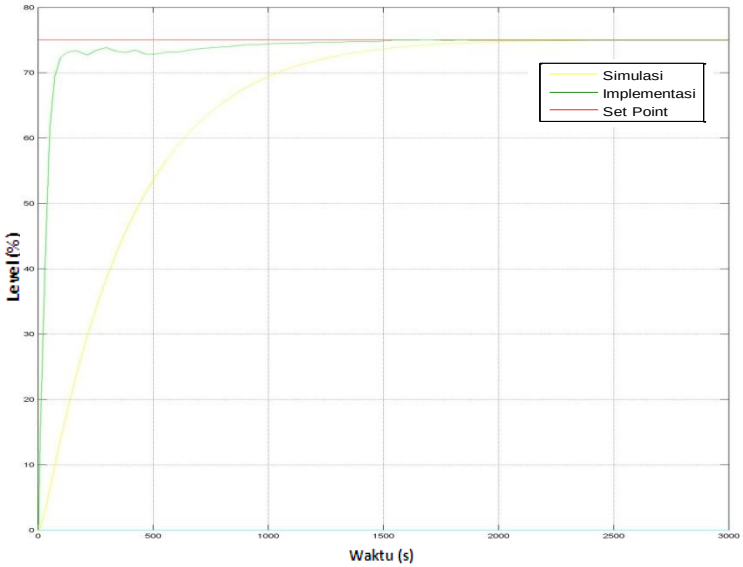
Berdasarkan kondisi *set point* 50% pada proses level Gambar 4.5, menunjukkan implementasi proses *level* respon yang dihasilkan dari sistem tersebut. Proses *level* diberi *set point level* 50% maka hasil dari nilai parameter kontroler PID yang didapat nilai Kp sebesar 2,167, Ki sebesar 0,0036 dan Kd 39,6. Respon yang didapat menghasilkan τ sebesar 27,38 detik, t_s ($\pm$ 5%) sebesar 82,14 detik dan t_r (5% - 95%)

sebesar 80,62 detik. Respon dengan kontroler memberikan keluaran *steady state* 49,85, sehingga nilai *error steady state* adalah 0,3%. Hasil respon yang dihasilkan pada *set point* 50% tidak memiliki *overshoot* atau tidak adanya osilasi pada respon tersebut. Nilai τ pada *set point* 50% lebih cepat dibandingkan dengan nilai τ pada *set point* 25%.

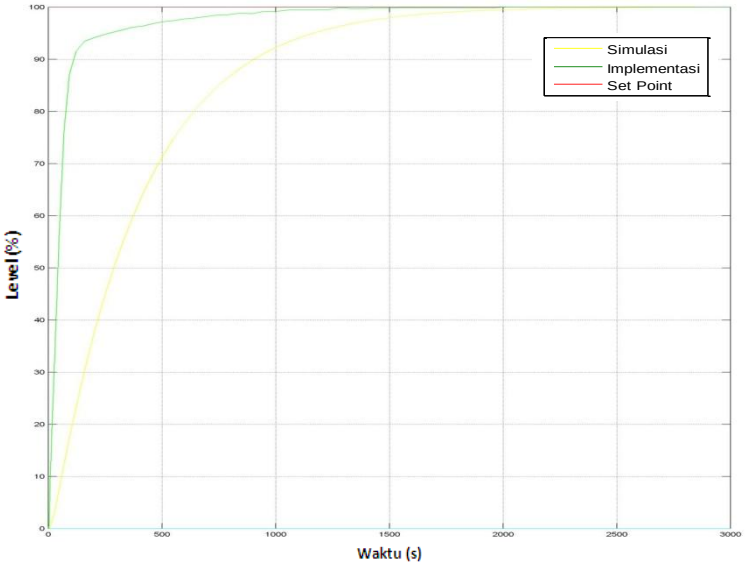


Gambar 4.5 Hasil Implementasi *set point* 50%

Dari Gambar 4.6 menunjukan respon implementasi proses *level* yang dihasilkan dari sistem tersebut. Proses *level* diberi *set point level* 75% maka hasil dari nilai parameter kontroler PID yang didapat nilai Kp sebesar 3,254, Ki sebesar 0,0054 dan Kd 59,46. Respon yang didapat menghasilkan τ sebesar 37,35 detik, t_s ($\pm$ 5%) sebesar 112,05 detik dan t_r (5% - 95%) sebesar 109,97 detik. Respon dengan kontroler memberikan keluaran *steady state* 74,95, sehingga nilai *error steady state* adalah 0,07%. Hasil respon yang dihasilkan pada *set point* 75% tidak memiliki *overshoot* atau tidak adanya osilasi pada respon tersebut.



Gambar 4.6 Hasil implementasi *set point* 75%



Gambar 4.7 Hasil implementasi *set point* 100%

Berdasarkan kondisi *set point* 100%, dari Gambar 4.7 dapat ditunjukan respon implementasi proses *level* yang dihasilkan dari sistem tersebut. Proses *level* diberi *set point level* 100% maka hasil dari nilai parameter kontroler PID yang didapat nilai Kp sebesar 4,337, Ki sebesar 0,0071 dan Kd 79,25. Respon yang didapat menghasilkan τ sebesar 55,3 detik, t_s ($\pm$ 5%) sebesar 165,9 detik dan t_r (5% - 95%) sebesar 162,83 detik. Respon dengan kontroler memberikan keluaran *steady state* 99,98, sehingga nilai *error steady state* adalah 0,02%. Hasil respon yang dihasilkan pada *set point* 100% tidak memiliki *overshoot* atau tidak adanya osilasi pada respon tersebut.

Hasil respon yang didapat dari sinyal dengan kontroler PID pada implementasi memiliki karakteristik sesuai dengan sistem yang dikehendaki. Kesalahan yang diperoleh pada sistem dengan kontroler PID yang dibuat memiliki kesalahan pada keadaan waktu tunak sangat kecil. Kesalahan sistem pada waktu tunak dapat dilihat pada Tabel 4.3.

Tabel 4.3 Kesalahan waktu tunak pada implementasi

Set Point	*Error Steady State* (e_{ss})
25%	0,12%
50%	0,3%
75%	0,07%
100%	0,02%

Respon yang dihasilkan pada implementasi memiliki respon *transient* yang lebih cepat, tetapi keluaran *steady state* memiliki nilai yang sama dengan respon simulasi.

BAB V
PENUTUP

5.1 Kesimpulan

Dari pengujian dan analisa yang telah dilakukan pada pengerjaan Tugas Akhir ini diperoleh beberapa kesimpulan. Dengan pendekatan metode Harriott, fungsi alih yang didapat memiliki hasil model respon pendekatan yang sama dengan τ sebesar 587,2 detik, t_s ($\pm$ 5%) sebesar 1685 detik, t_r (5% - 95%) sebesar 1475 detik, t_d sebesar 33,97 detik. Respon ini memberikan keluaran *steady state* 34,97. Kontroler PID *gain scheduling* dapat mengendalikan respon *plant* yang mengalami perubahan kondisi dengan rata-rata kesalahan waktu tunak sebesar 0,06% pada simulasi dan kesalahan waktu tunak pada implementasi sebesar 0,12% .

Secara umum nilai *gain* yang bergantung dari variabel. Implementasi kontroler PID, bahwa parameter proporsional, integral dan derivatif kontroler berubah tergantung variabel tertentu. Variabel tertentu yang berubah kontinyu terhadap waktu akan mempengaruhi nilai dari *gain* kontroler PID.

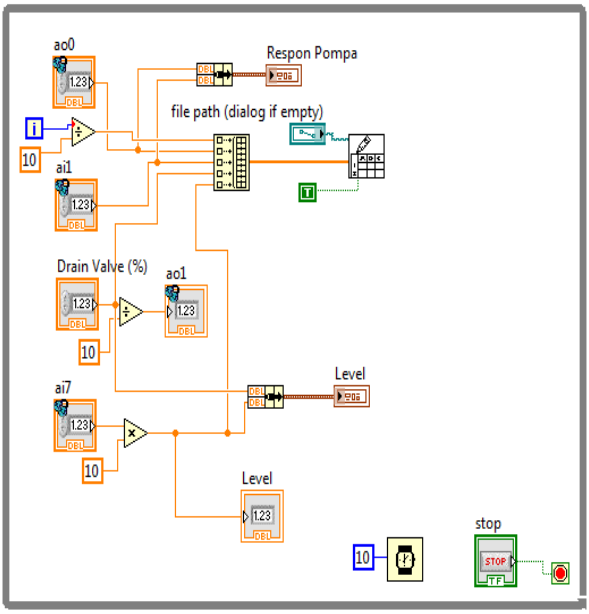
5.2 Saran

Dalam melanjutkan penelitian ini maka disarankan beberapa hal yaitu pemodelan dan pemahaman tentang *plant* yang digunakan dapat membantu perancangan sistem yang lebih baik. Membuat sistem yang lebih kompleks dan lengkap dengan alat PCT-100 yaitu mengaktifkan semua komponen baik sensor maupun aktuator.

--halaman ini sengaja dikosongkan--

LAMPIRAN

Program identifikasi *open loop* dengan *software* Labview



--halaman ini sengaja dikosongkan--

RIWAYAT PENULIS

Rachmad Dwi Raharjo dilahirkan di Lamongan, 8 Januari 1991. Merupakan putra ke-dua dari tiga bersaudara. Lulus dari SDN Sidoharjo I Lamongan pada tahun 2003, kemudian melanjutkan studinya ke SLTP N 3 Lamongan lulus pada tahun 2006. Kemudian melanjutkan ke SMA N 2 Lamongan dan lulus pada tahun 2009. Setelah menamatkan SMA, penulis menamatkan studinya pada jurusan Diploma 3 Teknik Elektro FTI-ITS pada tahun 2012. Kemudian penulis meneruskan studi sarjana pada program S1 Lintas Jalur Teknik Elektro, Fakultas Teknologi Industri, Institut Teknologi Sepuluh Nopember pada bulan Agustus tahun 2012 dan terdaftar dengan NRP 2212105091. Spesialisasi bidang studi yang ditekuni oleh penulis adalah Teknik Sistem Pengaturan. Pada bulan Juni 2014, penulis mengikuti seminar dan ujian tugas akhir di Bidang Studi Teknik Sistem Pengaturan sebagai salah satu persyaratan untuk memperoleh gelar Sarjana Teknik Elektro.

--halaman ini sengaja dikosongkan--