

## BAB II

### TINJAUAN PUSTAKA

#### 2.1 Studi Pustaka

Penelitian sebelumnya berjudul “*Feedforward Feedback* Kontrol Sebagai Pengontrol Suhu Menggunakan Proportional – Integral berbasis Mikrokontroler ATMEGA 8535”[3]. Tahapan awal penelitian ini dengan melakukan penalaan suhu dengan *auto tuning* Ziegler – Nichols, penalaan parameter – parameter Proportional – Integral dilakukan agar dapat diperoleh parameter yang sesuai untuk *plant* model. Setelah parameter PI didapatkan kemudian peneliti melakukan perbandingan tanggapan suhu secara *single control* dengan tanggapan suhu secara *feedforward feedback control* pada *gain* yang berbeda-beda. Dari hasil perbandingan tersebut didapatkan tanggapan keluaran pengendalian suhu secara *single control* memiliki kelemahan respon cenderung naik dan memiliki deviasi yang lebar sedangkan tanggapan suhu secara *feedforward feedback control* cenderung naik akan tetapi memiliki kelebihan yakni deviasi yang lebih sempit untuk gangguan yang sama, begitu juga dengan kestabilan yang dimiliki *feedforward feedback control* lebih cepat dari pada *single control*.

Penelitian yang lebih spesifik tentang *feedback feedforward control* sebagai pengontrol suhu sudah pernah dilakukan sebelumnya. Penelitian ini berjudul “Permodelan dan Simulasi PID Kontrol Pada Alat Penukar Panas”[4]. Pada penelitian ini peneliti melakukan perbandingan pengendali PID (*Proportional, Integral, Derivatif*) pada *Feedback* kontrol, *Feedback Plus Feedforward* kontrol, dan IMC (*Internal Model Controller*). *Feedback* kontrol dicapai dengan

menggunakan *PID controller*. Pengendalian proses kontrol *feedback* dan *feedforward* dijumlahkan untuk memberikan sinyal kontrol gabungan. Sedangkan IMC dirancang dengan melakukan invers model beserta fungsi transfer filter. Ketiga pengendali tersebut memberikan hasil yang berbeda-beda. Kinerja pengendali *PID Feedback* kontroler masih mempunyai kelemahan yang menghasilkan *overshoot* yang sangat tinggi. Pada pengendali *PID Feedback Plus Feedforward* kontroler, gabungan dua sistem kendali ini memiliki kelebihan dengan menurunkan *overshoot* hingga 56,1 %. Berbeda dengan pengendali *PID IMC* yang memberikan kinerja proses pengendalian temperatur yang sangat baik dengan hasil *overshoot* 6,2% dan *settling time* 88,8 detik.

Penelitian lainnya berjudul “ Simulasi Pengendalian Temperatur Pada *Heat Exchanger* Menggunakan Teknik Neuro – Fuzzy Adaptif “[5]. Hampir sama dengan penelitian yang sebelumnya, peneliti juga melakukan perbandingan antara pengendali *Feedforward control* , *Feedback control*, dan *Feedback Feedforward control*, hanya saja pada penelitian ini peneliti menggunakan teknik Neuro-Fuzzy Adaptif sebagai pengendalinya. Pada pengendali jenis *feedforward* tanggapan sistem sudah memberikan hasil yang relatif lebih baik dibanding tanpa pengendali. Pada pengendali jenis *Feedback*, kelemahan pada pengendali ini memiliki respon yang dihasilkan relatif lambat dibanding respon pengendali *Feedforward*, karena kerja pengendali *feedback* yang selalu harus mengevaluasi setiap keluaran sebelumnya secara loop sehingga lebih lama dibandingkan dengan *feedforward*. Berbeda dengan pengendali jenis *Feedback Feedforward* memiliki kelebihan dengan menghasilkan respon paling baik diantara pengendali – pengendali sebelumnya. Kelebihan dari gabungan pengendali ini, respon yang

dihasilkan relatif cepat dibanding dengan pengendali *Feedback*, karena kerja pengendali ini tidak murni *feedback* yang selalu harus mengevaluasi setiap keluaran sebelumnya secara loop. Dengan demikian komputasi yang dibutuhkan relatif lebih singkat dibanding pengendali *feedback*.

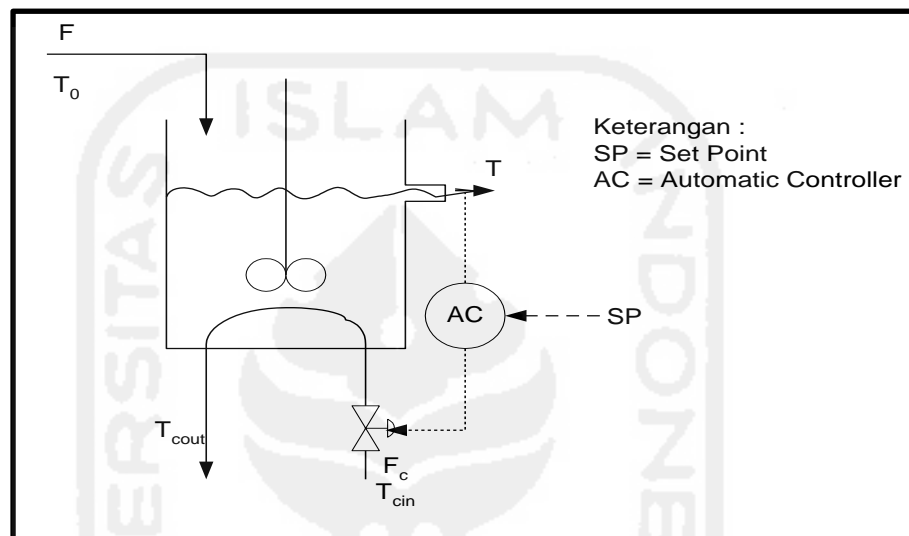
Dalam penelitian yang dilakukan oleh Jayanti [6], menggunakan IT2FLC sebagai pengendali pada sistem *heat exchanger*. Hasil yang diperoleh dari penelitian tersebut, model simulasi mampu menghasilkan temperatur minimal  $74,04^{\circ}\text{C}$  dan temperatur maksimal  $148,9^{\circ}\text{C}$  dalam waktu 100 detik. Respon sistem yang didapat dari pengujian model *plant stirred tank heat exchanger* berbasis IT2FLC dengan tipe PI-like fuzzy dapat diperbaiki dengan lebar FOU pada *membership function*nya. Optimalisasi respon sistem IT2FLC lebih luas dibandingkan T1FLC. Kelebihan IT2FLC memiliki performa yang lebih baik bila dibandingkan dengan T1FLC jika diimplementasikan dengan desain yang tepat. Begitu juga apabila sistem diberi keadaan *noise*, respon dari IT2FLC tetap lebih baik bila dibandingkan dengan T1FLC. Kelebihan dari pengendali ini, respon sistem IT2FLC dapat lebih baik apabila FOU mendapat lebar yang tepat pada MF-nya. Sistem kendali *stirred tank heat exchanger* dengan IT2FLC memiliki respon yang lebih baik atau lebih tepat dalam menangani adanya gangguan berupa perubahan data *plant* diantaranya perubahan kecepatan aliran panas, temperatur cairan pendingin, dan temperatur cairan panas dengan IT2FLC. [6]

## **2.2 Dasar Teori**

### **2.2.1 Model Stirred Tank Heat Exchanger**

Dibutuhkan sebuah model matematis pada simulasi kendali proses yang menggunakan komputer sebagai perantara dalam mensimulasikan proses, dimana

model ini merupakan variabel, dan data-data besaran fisik. Model matematis ini yang akan mewakili proses/*plant* yang berbentuk sebuah hardware menjadi sebuah formulasi dan diimplementasikan ke dalam *software* untuk disimulasikan. Langkah pertama dalam merancang pengendali adalah menentukan model matematis sistem yang akan dikendalikan.



Gambar 2.1 *Heat Exchanger* [1]

Pada penelitian ini digunakan model *plant Heat Exchanger* [1] untuk menguji metode yang diusulkan. *Heat Exchanger* merupakan sebuah proses yang mempunyai aliran masukan berupa fluida yang terdiri dari komponen tunggal dengan laju aliran  $F$  dan temperatur awal  $T_0$ . Fluida tersebut dimasukkan kedalam tangki selanjutnya terdapat aliran fluida pendingin yang masuk kedalam tangki dengan temperatur  $T_{cin}$ .

Pada sistem seperti ditunjukkan pada Gambar 2.1, cairan bersuhu panas  $T_0$  akan didinginkan hingga mencapai suhu cairan output  $T$  dengan cara mengalirkan cairan pendingin (*coolant*). Besarnya aliran cairan pendingin dapat dikendalikan sehingga suhu output  $T$  dapat mencapai nilai *setpoint*. Asumsi-asumsi yang

digunakan pada pemodelan sistem *stirred tank heat exchanger* pada penelitian ini adalah sebagai berikut (Marlin, 2000) :

- Tangki terisolasi dengan baik, sehingga tidak ada panas yang ditransfer keluar
- Akumulasi energi pada dinding tangki dan koil pendingin cukup kecil dibandingkan dengan akumulasi di dalam cairan sehingga dapat diabaikan.
- Cairan di tangki teraduk dengan rata
- Parameter – parameter fisik konstan
- Sistem *steady state* pada kondisi awal

Keseluruhan keseimbangan material dan energi diperlukan untuk menentukan aliran dan temperatur pada tangki tersebut. Keseimbangan material keseluruhan bersifat tetap pada tangki dengan pengaduk, dengan hasil bahwa level mendekati konstan dan  $F_0 = F_1 = F$ . Untuk sistem ini, akumulasi energi kinetik dan energi potensial adalah nol, dan bagian input-output-nya akan saling menghilangkan jika tidak nol. Rumus keseimbangan energi pada sistem dengan volume konstan ditunjukkan pada persamaan 2.1.

$$\frac{dU}{dt} = H_0 - H_1 + Q - W_s \quad (2.1)$$

$H_0$  = entalpi sebelum

$H_1$  = entalpi saat ini

$Q$  = energi yang ditransfer sistem ke lingkungan

$U$  = energi internal

$W_s$  = usaha *shaft*

Tujuan selanjutnya adalah menyatakan energi internal ( $U$ ) dan entalpi ( $H$ ) kedalam variabel terukur. Hal ini dapat dinyatakan dengan persamaan thermodinamik 2.2 dan 2.3.

$$\frac{dU}{dt} = V\rho C_v \frac{dT}{dt} \approx V\rho C_p \frac{dT}{dt} \quad (2.2)$$

$V$  = volume cairan ( $m^3$ )

$\rho$  = massa jenis cairan ( $g/m^3$ )

$C_v$  = kapasitas panas pada volume konstan ( $cal/(g \text{ } ^\circ C)$ )

$C_p$  = kapasitas panas pada tekanan konstan ( $cal/(g \text{ } ^\circ C)$ )

$$H_i = C_p \rho F_i (T_i - T_{ref}) \quad (2.3)$$

$F$  = kecepatan aliran cairan ( $m^3/min$ )

$T$  = suhu cairan ( $^\circ C$ )

Substitusi persamaan 2.2 dan 2.3 ke persamaan 2.1 menghasilkan persamaan berikut

$$V\rho C_p \frac{dT}{dt} = C_p \rho F (T_0 - T_{ref}) - T_1 - T_{ref} + Q \quad (2.4)$$

Persamaan 2.4 merupakan persamaan keseimbangan energi pada tangki, satu buah persamaan dengan dua variabel,  $T$  dan  $Q$ . Untuk memenuhi model *plant*, panas yang ditransfer harus dihubungkan pada suhu tangki dan juga variabel eksternal (aliran dan suhu pendingin). Selanjutnya, submasalah berkaitan dengan keseimbangan energi pada koil cairan pendingin yang telah didefinisikan dan dipecahkan. Dengan asumsi sebagai berikut:

- Koil cairan pada keadaan *quasi-steady*
- Properti fisik *coolant* konstan
- Gaya dorong untuk transfer energi dapat diperkirakan sebagai rata-rata antara masukan dan keluaran

Berdasarkan asumsi-asumsi tersebut. Keseimbangan energi pada koil pendingin adalah :

$$T_{cout} = T_{cin} - \frac{Q}{\rho_c C_{pc} F_c} \quad (2.5)$$

$T_{cout}$  = suhu output cairan *coolant* ( $^{\circ}\text{C}$ )

$T_{cin}$  = suhu input cairan *coolant* ( $^{\circ}\text{C}$ )

$\rho_c$  = massa jenis cairan *coolant* ( $\text{g}/\text{m}^3$ )

$C_{pc}$  = kapasitas panas *coolant* pada tekanan konstan ( $\text{cal}/(\text{g } ^{\circ}\text{C})$ )

$F_c$  = kecepatan aliran cairan *coolant* ( $\text{m}^3/\text{min}$ )

*Subscript c* menunjukkan cairan *coolant*. Dan, dua hubungan pokok digunakan untuk memenuhi persamaan model. Sehingga, transfer panas dapat dituliskan:

$$Q = -UA \Delta T_{lm} \approx UA \frac{T - T_{cin} + T - T_{cout}}{2} \quad (2.6)$$

$UA$  = koefisien transfer panas

Koefisien transfer panas tergantung pada koefisien lapisan dan resistansi dinding. Dalam berbagai desain, resistansi lapisan *stirred tank* dan resistansi dinding bagian luar, lebih kecil dibanding resistansi lapisan dalam tangki, sehingga  $UA \approx h_m A$ . Koefisien lapisan dalam dapat dikaitkan dengan aliran menggunakan hubungan empiris dengan persamaan 2.7.

$$UA = a F_c^b \quad (2.7)$$

$a$  = konstanta transfer panas ( $\text{cal}/\text{min } ^{\circ}\text{C}$ )

$b$  = konstanta

Persamaan 2.6 dan 2.7 dapat digabungkan untuk menghilangkan  $T_{cout}$  dan  $UA$  sehingga didapatkan persamaan 2.8.

$$Q = -\frac{aF_c^{b+1}}{F_c + \frac{aF_c^b}{2\rho_c C_{pc}}} T - T_{cin} \quad (2.8)$$

Penyelesaian terhadap submasalah ini, menyatakan transfer panas dalam kaitannya dengan variabel eksternal dan suhu pada tangki, yang merupakan variabel yang akan ditentukan. Persamaan 2.8 dapat disubstitusikan ke persamaan 2.4 untuk mendapatkan model *stirred-tank heat exchanger* seperti ditunjukkan pada persamaan 2.9

$$V\rho C_p \frac{dT}{dt} = C_p \rho F T_0 - T - \frac{aF_c^{b+1}}{F_c + \frac{aF_c^b}{2\rho_c C_{pc}}} T - T_{cin} \quad (2.9)$$

Model tersebut merupakan model tak linier karena ada bagian  $F_c$  yang dipangkatkan dan adanya perkalian antara  $F_c$  dan  $T$ . Data konstanta dan parameter yang digunakan, disajikan pada (Martin, 2000)

Tabel 2. 1 Data model *heat exchanger* (Martin, 2000)[1]

Besaran	Notasi	Nilai	Satuan
Aliran masuk	$F$	0.085	m <sup>3</sup> /min
Volume cairan dalam tangki	$V$	2.1	m <sup>3</sup>
Suhu <i>steady-state</i>	$T_s$	85.4	°C
Massa jenis cairan	$\rho$	10 <sup>6</sup>	g/m <sup>3</sup>
Kapasistas panas cairan pada tekanan tetap	$C_p$	1	cal/(g °C)
Suhu awal cairan masuk	$T_0$	150	°C
Suhu awal cairan <i>coolant</i>	$T_{cin}$	25	°C
Aliran <i>coolant</i> pada kondisi <i>steady-state</i>	$F_{cs}$	0.5	m <sup>3</sup> /min
Kapasistas panas cairan <i>coolant</i> pada tekanan tetap	$C_{pc}$	1	cal/(g °C)
Massa jenis cairan <i>coolant</i>	$\rho_c$	10 <sup>6</sup>	g/m <sup>3</sup>
Konstanta transfer panas	$a$	1.41x10 <sup>5</sup>	cal/min °C
Konstanta	$b$	0.5	





istilah “*closed loop*” berarti menggunakan aksi *feedback* untuk mengurangi kesalahan dalam sistem serta untuk membedakan antara *open-loop* dan *closed-loop*.

Kendali *feedback* lebih sering digunakan dibanding *open loop* atau kendali *feedforward*. Sistem *closed-loop* memiliki banyak kelebihan dibanding dengan sistem *open-loop*. Kelebihan utama dari sistem kendali *feedback closed-loop* adalah kemampuannya yang membuat respon sistem relatif kurang peka terhadap gangguan eksternal. Sistem kendali *feedback* memiliki empat komponen dasar yaitu : [7]

1. Karakteristik atau kondisi yang dikendalikan diukur dari outputnya
2. Sensor yang mengukur karakteristik atau kondisi tersebut
3. Unit pengendalian yang membandingkan hasil output sensor dengan suatu nilai
4. Unit pengatur yang menghasilkan tindakan penyesuaian untuk masukan selanjutnya.

Sementara ini, sistem kendali *feedback* yang baik memiliki banyak kelebihan bila dibandingkan dengan sistem kendali *feedforward*. Kelemahan dari sistem kendali *feedback* ini yakni sistem kendali *feedback* lebih kompleks dengan memiliki satu atau lebih jalur *feedback*. Selain itu gain didalam pengendali yang terlalu sensitif terhadap perubahan input menjadi tidak stabil dan osilasi dikarenakan pengendali mencoba mengoreksi dirinya secara berlebihan. [7]







Dari Gambar 2.5 dan Persamaan 2.10 diatas dapat dilihat pengendali PID memiliki tiga parameter yaitu konstanta proporsional ( $K_p$ ), konstanta integral ( $K_i$ ), dan konstanta derivative ( $K_d$ ). Ketiga bagian pada pengendali PID tersebut memiliki pengaruh yang berbeda – beda terhadap respon sistem. Bagian proporsional memberikan aksi kendali sebanding dengan nilai *error*, bagian integral memberikan aksi kendali sebanding dengan nilai integral *error*, sedangkan derivatif memberikan aksi kendali sebanding dengan nilai perubahan *error*. Bagian proporsional cenderung untuk memperbaiki respon transisi, bagian integral untuk memperbaiki *steady state error* dan bagian derivatif berfungsi memperbaiki *overshoot*. Untuk menghasilkan respon sistem yang sesuai dengan yang dikehendaki, ketiga parameter pengendali PID harus ditentukan nilainya (*tuning* dengan tepat).

Secara umum sistem pengendali PID terbagi menjadi dua jenis yaitu *Open Loop Control System* dan *Closed Loop Control System*. Pada pengendali digital dikenal beberapa istilah, antara lain :

1. *Set Point* adalah harga atau nilai dari keadaan yang ingin dicapai pada proses
2. *Error* adalah selisih antara *Set Point* dan *Process Variable*
3. *Manipulated Variable* adalah harga atau nilai yang diatur agar proses menjadi stabil, *Manipulated Variable* biasanya dihubungkan dengan input aktuator.
4. *Process Variable* adalah sinyal hasil pemantauan terhadap proses atau *plant*. *Process Variable* pada umumnya adalah hasil pembacaan dari suatu sensor.

5. *Plant* adalah objek yang akan dikendalikan.

Persamaan pengendali PID juga dapat dituliskan sebagai berikut :

$$mv(t) = K_p \left( e(t) + \frac{1}{T_i} \int_0^t e(t) dt + T_d \frac{de(t)}{dt} \right) \quad (2.12)$$

Keterangan :

$mv(t)$  = output dari pengontrol PID atau *Manipulated Variable*

$K_p$  = *Konstanta Proporsional*

$T_i$  = *Integral Time*

$T_d$  = *Derivative Time*

$e(t)$  = *error* (selisih antara *set point* dan nilai aktual)

Sistem kendali yang kedua adalah *Closed Loop System* atau sistem pengendali *loop* tertutup, yaitu sistem pengendali dimana objek yang dikontrol di-*feedback* ke input pengendali. Input yang diberikan ke pengendali merupakan selisih antara besaran (PV) dan besaran (SP). Nilai selisih sering disebut *error*. Tujuan dari pengendali adalah membuat nilai *Process Variable* (PV) sama dengan nilai *Set Point* (SP) atau bisa juga disebut nilai  $error = 0$ . Sinyal *error* akan diolah oleh pengendali agar (PV) sama dengan nilai (SP). Pengendali jenis ini bersifat otomatis karena objek yang akan dikendalikan dibandingkan lagi dengan input keadaan yang diinginkan, sehingga intervensi manusia dapat dihilangkan.

Setiap parameter dalam PID memiliki pengaruh yang berbeda –beda, diantaranya :

a. Pengendali Proportional ( $K_p$ )

- Apabila nilai  $K_p$  semakin besar, maka respon dari sistem tersebut akan memiliki nilai *rise time* dan *steady state error* yang semakin kecil.

- Dampak negatif dari semakin besarnya nilai  $K_p$  yaitu respon dari sistem tersebut mempunyai nilai *overshoot* yang semakin tinggi pula.

b. Pengendali Integral ( $T_i$ )

- Semakin besar nilai  $T_i$ , maka sistem akan memiliki *rise time* yang semakin rendah, bahkan akan menghilangkan nilai *steady state error* pada *plant*
- Semakin besar  $T_i$ , akan semakin meningkatkan nilai *overshoot* dan *settling time* dari respon sistem tersebut.

c. Pengendali Derivatif ( $T_d$ )

- Semakin besar nilai  $T_d$ , maka respon sistem akan memiliki nilai *overshoot* dan *settling time* yang semakin kecil.
- Pengaruh nilai  $T_d$  terhadap *rise time* dan *steady state error* kurang berpengaruh.

Tabel 2. 2 Pengaruh Penggunaan Parameter PID terhadap Respon Sistem

<b>Parameter</b>	<b><i>Rise Time</i></b>	<b><i>Overshoot</i></b>	<b><i>Settling Time</i></b>	<b><i>Steady State Error</i></b>
<b>Proporsional</b>	Menurun	Meningkat	Pengaruh kecil	Menurun
<b>Integral</b>	Menurun	Meningkat	Meningkat	Menghilangkan
<b>Derivative</b>	Pengaruh kecil	Menurun	Menurun	Pengaruh kecil

Pada Tabel 2.2 dapat dilihat pengaruh parameter PID terhadap respon sistem. *Rise time* adalah waktu yang diukur ketika respon = 0 sampai respon memotong sumbu *steady state* yang pertama. *Overshoot* adalah nilai yang menyatakan perbandingan antara nilai maksimum respon yang melampaui nilai *steady state* dibanding nilai *steady state*. *Settling time* adalah waktu yang



menyatakan respon telah masuk  $\pm 5\%$  atau  $\pm 2\%$  atau bisa juga  $\pm 0,5\%$  dari keadaan *steady state*.

Dalam pengendali PID tidak perlu semua parameter dalam PID diterapkan dalam sistem, hanya jika dibutuhkan. Apabila kendali PI sudah menghasilkan respon yang sesuai maka tidak perlu menambahkan pengendali D pada sistem yang dikendalikan.

### 2.2.5 LabVIEW

Software LabVIEW (*Laboratory Virtual Instrument Engineering Workbench*) merupakan bahasa pemrograman berbasis grafis atau blok, sedangkan pemrograman lainnya menggunakan *text*. Program LabVIEW dikenal dengan sebutan VI atau *virtual instrument* karena penampilannya dan operasinya dapat meniru sebuah instrumen sebenarnya dalam bentuk simbol-simbol. *Software* LabVIEW sendiri memiliki tiga komponen utama yaitu *front panel*, *block diagram*, *function palette*, dan *control palette*.

LabVIEW merupakan sebuah bahasa pemrograman grafis, LabVIEW tidak menggunakan teks untuk membuat suatu aplikasi melainkan dengan ikon-ikon yang telah disediakan. Perbedaan pemrograman LabVIEW dengan pemrograman teks yaitu pada pemrograman teks instruksi yang menentukan eksekusi program, sedangkan LabVIEW menggunakan program aliran data, dimana aliran dari data yang menentukan eksekusi.

Dengan LabVIEW dapat membuat *user interface* dengan menggunakan tools dan object tertentu. *User Interface* disebut juga sebagai *front panel*. Kita dapat memberikan kode dengan menggunakan grafis yang mewakili fungsi untuk









keluaran cairan pendingin ini adalah dengan mengatur bukaan *valve* pada aliran pendingin.

Pada Gambar 3.1 diatas menunjukkan bahwa *plant/process* akan dikendalikan dengan dua jenis sistem kendali yaitu *feedback control* (kendali umpan balik) serta *feedforward control* (kendali umpan maju). Setiap sistem kendali tersebut memiliki sebuah pengendali (*controller*). Sistem kendali *feedback* menggunakan *PI Controller*, sedangkan pada sistem kendali *feedforward* menggunakan *Gain Controller*.

Sistem kendali diatas dimulai dengan memodelkan *plant heat exchanger* ke dalam LabVIEW menggunakan persamaan model dari *plant* tersebut serta menguji respon pada *open loop* guna mendapatkan hasil yang maksimal [1]. Langkah awal yang dilakukan setelah menguji respon *open loop* yakni melalui pemberian *set point* ke dalam sistem *closed loop (feedback control)*. Nilai *set point* ini dapat diubah seiring berjalannya proses di dalam *plant*. Peran *feedback control* inilah yang berfungsi untuk mengatur *plant* supaya hasil proses sesuai dengan *set point* serta menjaganya agar tetap berada pada nilai yang diinputkan (*set point*). Hal ini dilakukan dengan cara *men-tuning* parameter pengendali PI. Output kendali PI merupakan input *heat exchanger* untuk mengendalikan kecepatan aliran cairan pendingin ( $F_c$ ).

Apabila hasil respon dalam sistem *feedback* sudah sesuai *setpoint* kemudian sistem kendali *feedback* tersebut dipadukan bersama sistem kendali *feedforward* dengan *disturbance* sebagai input pada kendali *feedforward*. *Disturbance* ini merupakan representasi dari adanya pengganggu yang akan mengganggu jalannya pengendalian terhadap proses yang sedang dilakukan. Sistem kendali *feedforward*

iniilah yang berperan dalam menghilangkan atau mengurangi efek yang ditimbulkan dari *disturbance*. Diharapkan dengan adanya dua pengendali tersebut menghasilkan respon yang maksimal serta mencapai tujuan sistem pengendalian.

### 3.2 Pemodelan sistem *Heat Exchanger*

Pemodelan sistem *plant* pada penelitian ini menggunakan fasilitas *control* dan *function palette* pada LabVIEW. Proses transfer panas atau pendinginan cairan ditunjukkan pada model sistem *stirred tank heat exchanger* pada bab II Gambar 2.1.

Pada sistem seperti ditunjukkan pada Gambar 2.1, sebuah cairan bersuhu panas mempunyai laju aliran sebesar  $F$  dengan suhu awal  $T_0$  akan didinginkan hingga mencapai suhu cairan output  $T$  dengan cara mengalirkan cairan pendingin (*coolant*). Besarnya aliran cairan pendingin dapat dikendalikan untuk menghasilkan produk cairan dengan suhu output  $T$  mencapai nilai *setpoint*.

Model *stirred tank heat exchanger* yang direpresentasikan dalam bentuk persamaan matematis ditunjukkan pada persamaan 2.9.

Dalam persamaan 2.9 tersebut terdapat tujuh konstanta yaitu  $V$  (Volume),  $\rho$  (massa jenis),  $C_p$  (Kapasitas panas),  $C_{pc}$  (Kapasitas panas cairan *coolant*),  $\rho_c$  (massa jenis cairan *coolant*),  $a$  (konstanta transfer panas) dan  $b$  (konstanta) serta lima buah variabel yaitu  $T$  (temperatur output cairan),  $T_0$  (Temperatur input),  $F$  (aliran masuk),  $F_c$  (Aliran cairan *coolant*), serta  $T_{cin}$  (suhu awal cairan *coolant*). Proses variabel pada persamaan 2.9 dapat diklasifikasi sebagai berikut terdapat satu buah output yaitu  $T$  dan empat buah input yaitu  $F$ ,  $F_c$ ,  $T_0$  dan  $T_{cin}$ . Sedangkan data-data parameter yang digunakan pada pemodelan sistem *plant* ditunjukkan pada tabel 2.1

Dari Gambar 2.1 dan Tabel 2.1 terlihat sebuah cairan masuk ke dalam tangki berpengaduk dengan volume  $2,1 \text{ m}^3$  mengalir sebesar  $0,085 \text{ m}^3/\text{min}$ . Cairan ini memiliki temperatur awal sebesar  $150^\circ\text{C}$ , selanjutnya cairan tersebut didinginkan dengan cara mengalirkan cairan pendingin (*coolant*) yang mempunyai aliran coolant  $F_c$  sebesar  $0,5 \text{ m}^3/\text{min}$  pada kondisi *steady state*. Apabila tidak ada perubahan dari data – data tersebut, suhu cairan akan turun hingga mencapai temperatur *steady state* sebesar  $85,4^\circ\text{C}$ . Temperatur awal cairan  $T_0$  dan aliran masuk cairan  $F$  dapat berubah-ubah dikarenakan faktor eksternal. Oleh karena itu, perubahan temperatur  $T_0$  dan aliran masuk  $F$  dapat dimasukkan ke dalam variabel *disturbance* dalam sistem pengendalian proses ini.

Melalui simulasi pada LabVIEW, data parameter yang ditunjukkan pada tabel 2.1 disubstitusikan pada persamaan model matematis dari *plant*, sehingga dapat dituliskan dalam persamaan 3.1 sebagai berikut :

$$2.1 \times 10^6 \frac{dT}{dt} = (0.85 \times 10^6 \cdot 150 - T - \frac{1.41 \times 10^5 F_c^{0.5}}{F_c + (0.141) F_c^{0.5}} T - 25) \quad (3.1)$$

Pada persamaan tersebut dapat dilihat bahwa variabel yang dikendalikan adalah aliran *coolant* ( $F_c$ ) dan variable yang diinginkan adalah temperatur cairan ( $T$ ). Pembuatan model matematis *plant* menggunakan *control and function palette* yang terdapat pada *block diagram* LabVIEW.

### 3.3 Implementasi Sistem

Data-data untuk melakukan simulasi kendali proses telah dipaparkan pada sub bab 3.1 dan 3.2. Pada sub bab ini akan dijelaskan tentang implementasi penelitian yang dilakukan oleh peneliti dengan menggunakan *software* LabVIEW.





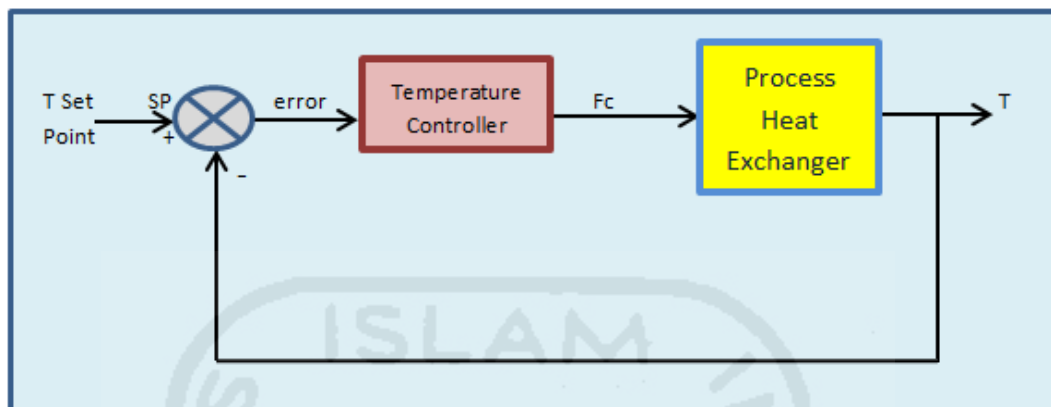
plant ini meliputi  $V$  (Volume),  $\rho$  (massa jenis),  $C_p$  (Kapasitas panas),  $C_{pc}$  (Kapasitas panas cairan *coolant*),  $\rho_c$  (massa jenis cairan *coolant*),  $a$  (konstanta transfer panas) dan  $b$  (konstanta) serta lima buah variabel yaitu  $T$  (temperatur output cairan),  $T_0$  (Temperatur input),  $F$  (aliran masuk),  $F_c$  (Aliran cairan *coolant*), serta  $T_{cin}$  (suhu awal cairan *coolant*). Variabel – variabel serta konstanta-konstanta yang telah disebutkan kemudian dihubungkan dengan operator matematika seperti pada Gambar 3.2. Kemudian dilanjutkan dengan memasukkan hasil operasi matematika tersebut ke dalam fungsi integrator (ditunjukkan pada angka 2) untuk menghasilkan temperatur output dari turunan  $T$ .

*Icon wait ms* yang ditunjukkan pada nomor 1 pada Gambar 3.2 merupakan waktu tunda (*delay*) dan berfungsi untuk mengatur kecepatan simulasi pada LabVIEW. Selanjutnya *toolbox* yang ditunjukkan pada angka 3 dan 4 merupakan alat yang digunakan untuk menampilkan grafik yang dihasilkan oleh keluaran *heat exchanger*. Sedangkan untuk kotak yang mencakup semua variabel, konstanta, dan *toolbox* adalah *control and simulation loop* yang mana merupakan *loop* simulasi pada sistem pengendalian ini. *Loop* ini akan membuat simulasi berulang secara terus menerus hingga berhenti pada waktu yang telah ditentukan sebelumnya.

### 3.3.2 Penerapan *Feedback Control* menggunakan Sub VI *PID controller*

Implementasi atau penerapan selanjutnya yakni pembuatan sistem kendali *feedback* menggunakan pengendali PID. Pada sub bab sebelumnya telah dijelaskan bahwa sistem kendali *feedback* ini akan mengendalikan proses agar keluaran dari sistem tetap berada pada *set point* yang telah diberikan. Diagram

blok sistem kendali *feedback* dapat dilihat pada Gambar 3.3 untuk memudahkan peneliti dalam menerapkannya ke dalam LabVIEW.



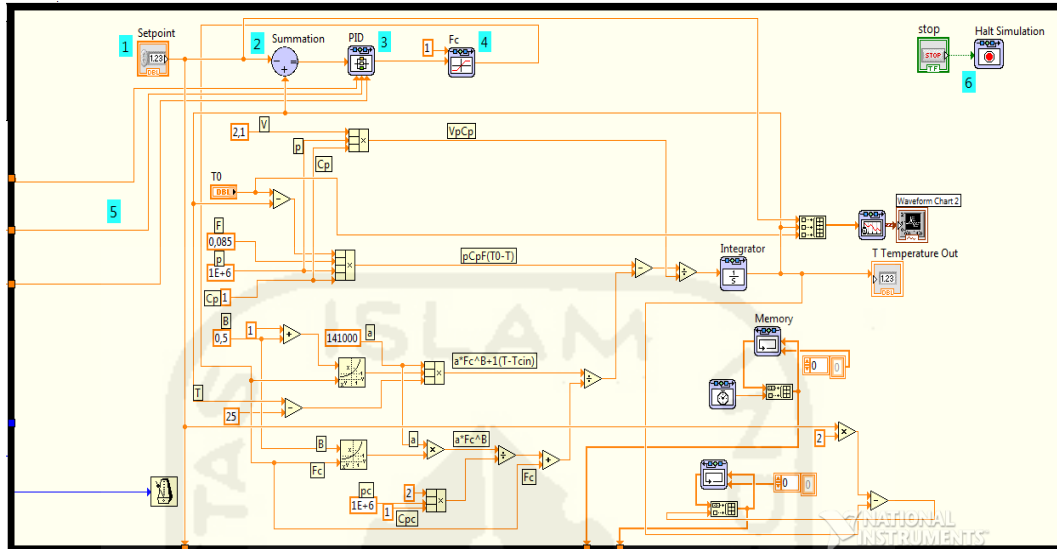
Gambar 3. 3 Diagram blok sistem kendali *feedback*

Dalam diagram blok pada Gambar 3.3 terdapat *set point* yang diinputkan kedalam sistem sebagai referensi pengendalian proses. Setelah di proses, hasil keluaran atau output proses dikembalikan menuju *summation*. Proses dikembalikan inilah yang dimaksud dengan *feedback*. Hasil penjumlahan dari nilai *set point* dengan output proses yang bernotasi negatif (*negative feedback*) pada *summation* ini digunakan untuk mendapatkan nilai *error* yakni selisih antara output proses dengan *set point*nya.

Nilai *error* tersebut digunakan sebagai masukan pada pengendali PID di dalam sistem kendali *feedback*. Pengendali PID ini akan menghasilkan sinyal kendali yang akan menjadi masukan pada proses/*plant* (yang dimaksud dalam hal ini adalah *manipulated variable* ). Dengan begitu pengendali PID pada sistem kendali *feedback* ini dapat mengendalikan proses *heat exchanger* yang menghasilkan output tetap berada pada *set point*nya.

Langkah selanjutnya yang akan dilakukan setelah membuat diagram blok yakni mengimplementasikan sistem kendali *feedback* dengan menggunakan

software LabVIEW. Penerapan sistem kendali *feedback* pada LabVIEW ditunjukkan pada Gambar 3.4.



Gambar 3. 4 Penerapan Sistem Kendali *Feedback* pada LabVIEW

Dengan mengacu pada diagram blok pada Gambar 3.3 didapatkan hasil penerapan sistem kendali *feedback* menggunakan program LabVIEW ditunjukkan pada Gambar 3.4. Pada Gambar 3.4 terlihat *icon* fungsi, variable dan konstanta yang telah diberi nomor guna mempermudah penjelasan masing masing *icon* dalam penerapan sistem kendali *feedback* ini. Berikut penjelasan dari masing masing *icon* yang telah diberi nomor :

1. *Icon* yang pertama yakni variabel nilai *set point* yang berfungsi untuk memberi referensi ke dalam pengendalian proses dengan sistem kendali *feedback* .
2. *Icon* yang kedua yakni *summation* dimana *icon* ini berfungsi untuk mencari selisih (*error*) antara nilai *set point* dengan nilai output proses (*T*). Nilai output proses (*T*) disini merupakan sinyal *feedback* yang diberi notasi negatif sehingga menjadi *negative feedback*.

3. *Icon* yang ketiga yakni fungsi pengendali PID. Pengendali PID ini mendapat nilai selisih (*error*) antara *set point* dengan keluaran dari *T*. Sedangkan nilai *T* tersebut didapat dari *feedback* pada output *heat exchanger* yang menuju *summation*. Dengan begitu persamaan pengendali PID dapat dituliskan pada persamaan 3.2 :

$$u(s) = Kp e(s) + \frac{Ki}{s} e(s) \quad (3.2)$$

Persamaan diatas menjelaskan bahwa pengaturan (*tuning*) parameter pengendali PID pada sistem ini dilakukan tanpa menggunakan parameter D karena respon yang dihasilkan sudah mencapai *set point* yang diinginkan.

4. *Icon* yang keempat yakni variable *Fc* yang mana dalam persamaan model *heat exchanger* ini sebagai *manipulated variable*. Dalam *block diagram* pada LabVIEW, *Flow Coolant (Fc)* ini sebagai fungsi saturasi untuk memotong nilai output. Nilai output fungsi saturasi (*Fc*) berguna untuk membatasi agar tidak melebihi batas ambang yang dinyatakan disamping kiri *icon* yakni 1 m<sup>3</sup>/min sebagai batas atasnya. Pada kondisi *steady state*, *valve* akan mengalirkan cairan pendingin dengan kecepatan 0,5 m<sup>3</sup>/min, sedangkan pada sistem *heat exchanger* maksimal kecepatan aliran pendingin adalah 1 m<sup>3</sup>/min (dalam keadaan *valve* terbuka penuh). Apabila nilai yang diberikan melebihi 1 m<sup>3</sup>/min maka output nilai dari variable *Fc* akan tetap menghasilkan nilai output maksimal 1 m<sup>3</sup>/min. Hal ini terjadi karena variabel *Fc* pada data *plant heat exchanger* adalah 0.5 m<sup>3</sup>/min pada saat kondisi *steady state*, sedangkan sinyal kendali yang dihasilkan oleh pengendali PID dapat melebihi nilai variabel *Fc* yang telah dibatasi oleh

karena itu, nilai aliran *coolant* maksimal sebesar  $1 \text{ m}^3/\text{min}$  (dalam keadaan *valve* terbuka penuh).

5. *Icon* yang kelima ini merupakan variabel input untuk pengendali PID, dengan *gain* sebagai input untuk kendali Proporsional, Integral dan Derivatif.
6. *Icon* yang keenam adalah *Halt Simulation* yang mana *icon* ini digunakan untuk menghentikan proses yang sedang berjalan, tetapi tetap mempertahankan hasil proses yang telah didapat sebelumnya.

### 3.3.3 Penerapan Kendali *Feedforward* menggunakan *Gain controller*

Kendali *Feedforward* merupakan kendali yang menggunakan *disturbance* sebagai input untuk melakukan pengendalian terhadap *disturbance* itu sendiri. Aksi kendali pada kendali *feedforward* menghasilkan sinyal kendali yang bergantung pada *disturbance* yang terukur. Biasanya sistem kendali *feedforward* bekerja hanya untuk mendampingi atau membantu sistem kendali *feedback* dalam mengendalikan sebuah proses. Inti dari penerapan kendali *feedforward* ini yaitu melakukan penambahan sistem kendali *feedforward* setelah membuat sistem kendali *feedback* pada program LabVIEW. Dengan begitu, didapatkan hasil gabungan dua sistem pengendali yakni *feedback* dan *feedforward* yang ditunjukkan pada Gambar 3.5



dalam penerapan sistem kendali *feedforward* ini. Berikut penjelasan dari masing-masing *icon* yang telah diberi nomor :

1. *Icon* yang pertama adalah sebuah konstanta yang digunakan untuk memberi referensi kepada pengendali *feedforward*, yaitu *gain controller*. Konstanta referensi ini digunakan untuk memberikan informasi nilai temperatur cairan yang masuk dan dianggap normal (tidak ada perubahan temperatur awal cairan).
2. *Icon* yang kedua yakni *summation*. Tugas *icon* hanya menjumlahkan nilai konstanta (notasi positif) dengan nilai temperatur awal cairan masuk (notasi negatif). Apabila tidak ada perubahan temperatur awal cairan masuk maka *summation* menghasilkan nilai 0, namun apabila terdapat perubahan temperatur awal input ( $T_0$ ) maka peran *summation* inilah yang menghasilkan selisih perubahan temperatur masukan terhadap referensi yang diberikan.
3. *Icon* yang ketiga adalah sebuah pengendali *gain* dari sistem kendali *feedforward*, yang mana akan menguatkan sinyal deviasi yang dihasilkan oleh *summation* pada *icon* nomor 2. Kemudian sinyal deviasi ini akan menjadi sinyal kendali *feedforward*, yang mana akan dijumlahkan dengan sinyal kendali *feedback* pada tahap selanjutnya.
4. *Icon* yang keempat merupakan sebuah *icon* yang merupakan masukan nilai variabel  $T$  pada pengendali *feedforward*, maka dari itu banyaknya penguatan yang diatur pengendali *gain* (*feedforward controller*) tergantung pada nilai yang diberikan pada variabel ini.



5. *Icon* yang kelima ini sama dengan *icon* yang keempat yakni sebuah *icon* yang merupakan masukan nilai variabel  $F$  pada pengendali *feedforward*,
6. *Icon* yang keenam ini juga sama dengan *icon* pada nomor tiga yaitu sebuah pengendali *gain* dari sistem kendali *feedforward*.
7. *Icon* yang ketujuh ini sama dengan *icon* yang kedua yakni *summation*. Hanya saja pada *icon* nomor tujuh ini berfungsi untuk menjumlahkan sinyal kendali *feedback* dengan sinyal kendali *feedforward*.
8. *Icon* yang kedelapan ini merupakan sebuah fungsi untuk menampilkan hasil dari sinyal kendali *feedback*, *feedforward* dan gabungannya ke dalam sebuah grafik (*waveform chart*), dengan begitu di dalam *waveform* terdapat tiga buah sinyal kendali.

### 3.4 Metode Pengujian

Pada sub bab ini akan dijelaskan mengenai bagaimana sistem atau penelitian yang dilakukan dapat memenuhi spesifikasi performa yang diinginkan. Berikut skema pengujian yang dilakukan :

1. Pengujian respon *plant* menggunakan sistem kendali *open loop*
2. Pengujian respon *plant* menggunakan sistem kendali *feedback* dengan PID sebagai pengendalinya.
3. Pengujian respon *plant* menggunakan kombinasi dua sistem kendali yakni *feedback* dan *feedforward* dengan pemberian *disturbance*.