Teknika 18 (2): 367 – 376

JURNAL TEKNIKA
ISSN: 0854-3143
e-ISSN: 2622-3481
Journal homepage: http://jurnal.polsri.ac.id/index.php/teknika
Journal Email: teknika@polsri.ac.id

Pengujian FactoryIO sebagai Alat Peraga Kendali *Proportional Integral Derivative* (PID) pada PLC M221

Enas Dhuhri Kusuma [*1], Nur Wijaya Kusuma[2], Addin Suwastono[3]

[1,2,3]Dept. Teknik Elektro dan Teknologi Informasi, Fakultas Teknik, Universitas Gadjah Mada, Yogyakarta, Indonesia
*Email Penulis Korespondensi: enas@ugm.ac.id

Abstrak

Praktikum dan hands-on merupakan kegiatan yang dapat membantu meningkatkan pemahaman mahasiswa terkait objek yang dipelajari. Kegiatan ini membutuhkan alat peraga yang dapat diamati dan dipelajari secara langsung oleh mahasiswa. Tetapi pembuatan alat peraga fisik membutuhkan waktu dan biaya tambahan. Solusi yang diajukan adalah dengan menggunakan perangkat lunak simulasi pabrik, yaitu FactoryIO. Maka dari itu penelitian ini bertujuan untuk menguji penggunaan FactoryIO sebagai peraga kendali PID pada PLC M221. Pengujian dilakukan dengan menetapkan parameter PID yang didapatkan dari hasil perhitungan menggunakan respon undak sistem tangki air dari FactoryIO. Hasil yang didapatkan adalah FactoryIO memiliki kalkulasi gravitasi yang mengakibatkan adanya tunda waktu dan osilasi pada pembacaan ketinggian air. Hal ini mengakibatkan sistem dan kontroler PID yang sudah dimodelkan tidak bekerja sesuai target yang dituju. Tetapi FactoryIO tetap dapat digunakan sebagai alat peraga PLC baik untuk komunikasi Modbus atau kontrol PID.

Kata kunci— *PID, PLC, FactoryIO, Modicon, Peraga*

Abstract

Practicums and hands-on are activities that can help improve students' understanding of the objects being studied. This activity requires props that students can observe and learn directly. But making physical training kit requires additional time and costs. The proposed solution is to use factory simulation software, namely FactoryIO. Therefore, this research aims to test the use of FactoryIO as a PID control demonstration on the M221 PLC. Testing was carried out by determining the PID parameters obtained from calculation results using the step response of the water tank system from FactoryIO. The results obtained are that FactoryIO has gravity calculations which result in time delays and oscillations in the water level readings. This results in the system and PID controller that has been modeled not working according to the intended target. But FactoryIO can still be used as a PLC trainer for either Modbus communication or PID control.

Keywords— *PID, PLC, FactoryIO, Modicon, Training Kit*

1. PENDAHULUAN

Praktikum dan *hands-on* merupakan salah satu model penyampaian materi kepada mahasiswa. Hands-on atau kegiatan tangan mendorong mahasiswa untuk mengamati, melakukan, dan mengidentifikasi masalah secara langsung terhadap objek yang dipelajari. Metode ini cocok untuk diterapkan pada mata kuliah *Programmable Logic Control* (*PLC*) yang mana melatih kemampuan menganalisis program dan menerapkannya pada PLC untuk menguji fungsionalitasnya.

PLC memiliki berbagai bentuk implementasi pada pabrik. Salah satunya adalah untuk mengontrol volume tangki air. Sudah ada banyak literatur yang berhubungan dengan kontrol keluaran untuk mencapai target tertentu. Seperti pada literatur [1] dan [2] di mana navigasi robot dilakukan dengan membandingkan pembacaan sensor dengan target secara terus-menerus hingga robot mencapai target tujuan dan seketika berhenti atau melakukan tindakan lain. Metode tersebut kurang cocok diterapkan pada sistem pengisian tangki air karena kurang presisi sehingga terlalu sering menyalakan-mematikan motor mengakibatkan konsumsi daya yang kurang efisien dan berpotensi memperpendek umur motor meskipun lebih sederhana untuk diimplementasikan.

Solusi lain dalam sistem pengisian tangki air adalah menggunakan keluaran yang terkontrol. Kontroler *Proportional-Integral-Derivative* (*PID*) sudah biasa digunakan dalam otomasi untuk meminimalisir distrubsi dan menjaga stabilitas sistem. Bahkan 90% dari kontroler otomasi selalu menggunakan kontroler PID karena kesederhanaan, keandalan, dan efektifitasnya [3]. Terdapat banyak studi terkait prinsip kontroler PID, contohnya literatur [4] dan [5] yang memanfaatkan PID untuk kontrol kecepatan pada motor.

Meskipun kontroler PID sederhana dan efektif dalam mengendalikan berbagai sistem, dibutuhkan nilai konstanta yang tepat agar keluaran sesuai dengan yang diharapkan. Sehingga terdapat beberapa literatur yang mengembangkan metode kontroler PID agar konstantanya berubah-ubah sesuai dengan kondisi sistem saat ini. Literatur [6] dan [7] yang memanfaatkan metode *fuzzy* untuk menentukan konstanta PID. Contoh lain adalah literatur [8] yang memanfaatkan *swarm optimization* dan literatur [9] yang memanfaatkan metode *cascade*.

Materi penerapan PID pada PLC membutuhkan alat peraga untuk menunjukkan hasil dari program yang telah dibuat. Tetapi pembuatan alat peraga fisik mengenai tangki pengisian air akan membutuhkan waktu dan biaya yang tidak sedikit. Selain itu, penggunaan air di dalam lab memiliki resiko tertumpah sehingga berpotensi merusak peralatan lainnya. Maka salah satu solusi permasalahan ini adalah menggunakan alat peraga berupa simulasi yaitu dengan *software FactoryIO*.

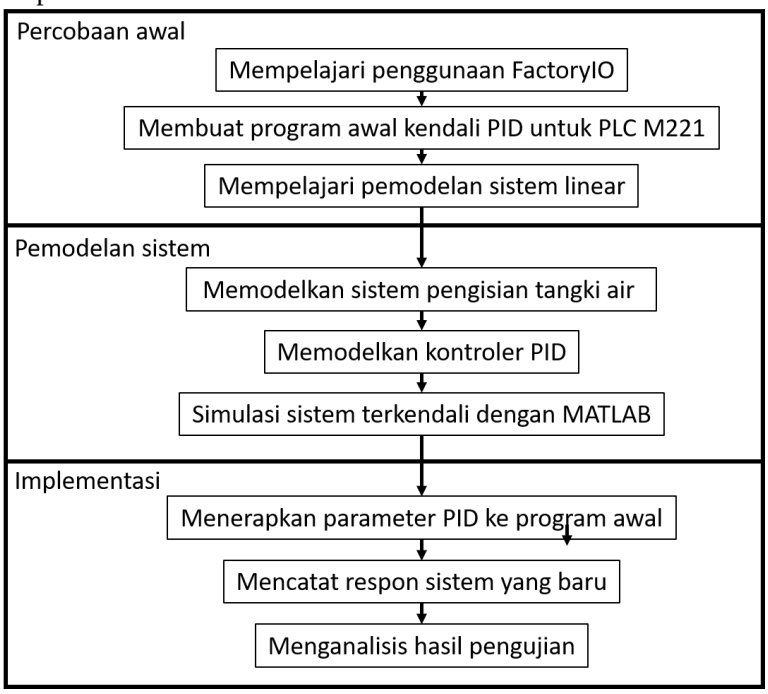
FactoryIO adalah *software* simulasi pabrik dengan berbagai fitur mulai dari sabuk konveyor, elevator, tangki air, dan lain-lain. Penggunaan *FactoryIO* sebagai alat peraga telah dilakukan pada beberapa literatur. Literatur [10] dan [11] membahas penerapan PID pada PLC menggunakan model peraga *FactoryIO*. Selanjutnya literatur [12] mengembangkan penerapan algoritma *neural network* untuk penerapan PID dengan hasil pengujian memanfaatkan *FactoryIO*.

Seperti yang dijelaskan sebelumnya, praktikum membutuhkan alat peraga agar mahasiswa dapat menyelesaikan masalah secara langsung. Tetapi pembuatan alat peraga kontrol volume tangki air memiliki kekurangannya sendiri. Sehingga, *FactoryIO* sebagai *software* simulasi pabrik dipilih sebagai alat peraga untuk pembelajaran PLC. Maka dari itu, penelitian ini bertujuan untuk mempelajari karakteristik FactoryIO, yang secara spesifik terkait penggunaan simulasi tangki air, untuk digunakan sebagai peraga implementasi PID pada praktikum PLC.

2. METODE PENELITIAN

Penelitian ini dilakukan menggunakan PLC keluaran *Schneider* dengan model M221 yang diprogram menggunakan *software Machine Expert Basic*. Selanjutnya penelitian dibagi

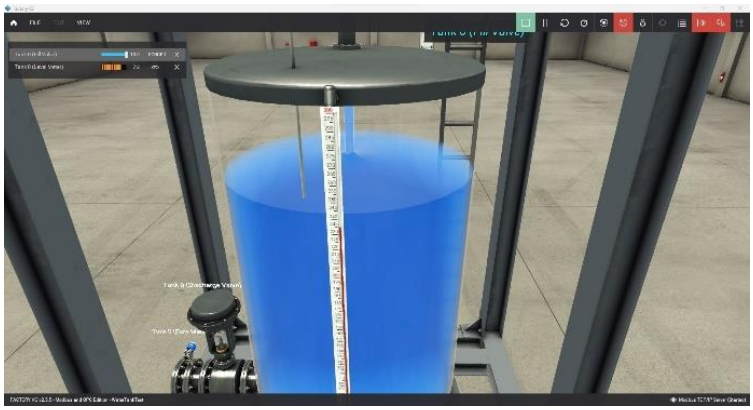
menjadi tiga bagian, yaitu: percobaan awal, pemodelan sistem, dan implementasi. Detil langkah penelitian ditunjukkan pada Gambar 1.



Gambar 1 Bagan alir metode penelitian

2. 1 Percobaan Awal

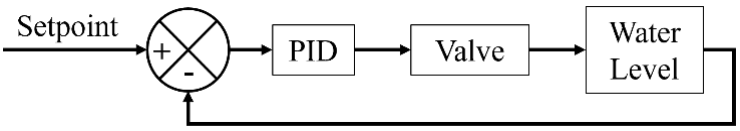
Pada tahapan ini dilakukan studi literatur dan program yang digunakan. *FactoryIO* sebagai *software* peraga pabrik dapat berkomunikasi dengan PLC menggunakan protokol Modbus. Tangki air yang digunakan sebagai peraga pada *FactoryIO* membutuhkan alokasi alamat *Modbus* sepanjang empat *word*. Alokasi alamat ini digunakan sebagai berikut: *fill valve*, *level meter*, *discharge valve*, dan *flow meter*. Percobaan PID ini hanya akan memanfaatkan dua data yaitu aktuator *fill valve* untuk mengatur debit air untuk mengisi tangki dan sensor *level meter* untuk mengukur ketinggian air. Tampilan tangki air pada *FactoryIO* ditunjukkan pada **Error! Reference source not found.**.



Gambar 2 Tampilan Tangki Air di *FactoryIO*

Setelah diketahui cara peggunaan FactoryIO, langkah berikutnya adalah menghubungkannya dengan program PLC. PLC yang diprogram menggunakan *Machine Expert* Basic memiliki fitur kendali PID sebagai *function* [13]. *Function* ini memungkinkan memberikan PID pada keluaran yang telah ditentukan dengan parameter K_p, T_i, dan T_d yang dapat diubah-

ubah. Parameter PID tersebut memiliki satuan yang telah ditentukan dari *Machine Expert*, yaitu K_p adalah 0,01 satuan, T_i adalah 0,1 satuan, dan T_d adalah 0,1 satuan. Maka ketika program PID pada Machine Expert dihubungkan dengan tangki air pada FactoryIO, siklus datanya dapat dituliskan dengan diagram blok yang ditunjukan pada Gambar 3. Pada langkah ini, dilakukan penerapan PID dengan metode *tuning trial and error* untuk memastikan bahwa program PID dapat bekerja pada PLC meskipun menggunakan parameter yang belum benar.



Gambar 3 Diagram blok untuk tangki air

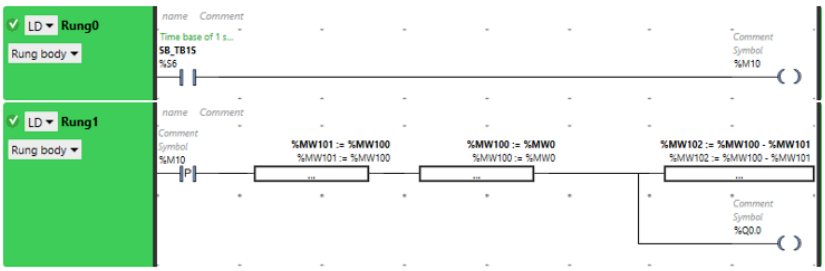
2. 2 Pemodelan Sistem

2. 3.1 Pemodelan Sistem

Model sistem tangki air FactoryIO didapatkan dengan melakukan pengujian terhadap *step response*. Karena sensor yang digunakan adalah ketinggian air pada tangki, maka debit air yang dikeluarkan katup (*valve*) diukur dengan membandingkan perubahan ketinggian air dalam satu satuan waktu.

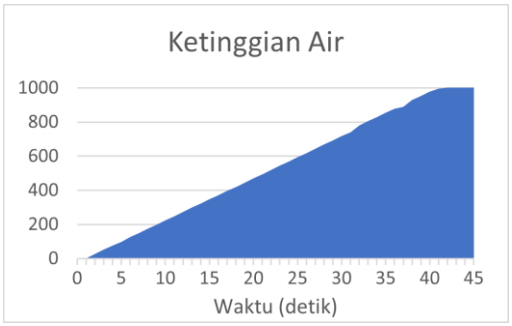
Pengukuran debit dilakukan dengan dua metode secara bersamaan. Metode pertama adalah dengan mengukur nilai hasil ketinggian air yang diterima PLC kemudian membandingkan perubahan nilainya setiap detik menggunakan *spreadsheet*. Metode kedua adalah dengan membandingkan perubahan ketinggian setiap 1 detik menggunakan program pada PLC.

Gambar 4 menunjukkan diagram *ladder* program untuk menghitung debit menggunakan PLC. Alamat MW0 adalah alamat yang menyimpan hasil pembacaan ketinggian air dari *FactoryIO*. Pembacaan tersebut dipindah ke MW100 sebagai ketinggian saat ini. Alamat MW101 adalah ketinggian 1 detik yang lalu, yang didapat dari memindahkan nilai dari MW100. Selanjutnya nilai perbedaan nilai MW100 dengan MW101 disimpan ke MW102 sebagai pembacaan debit air.



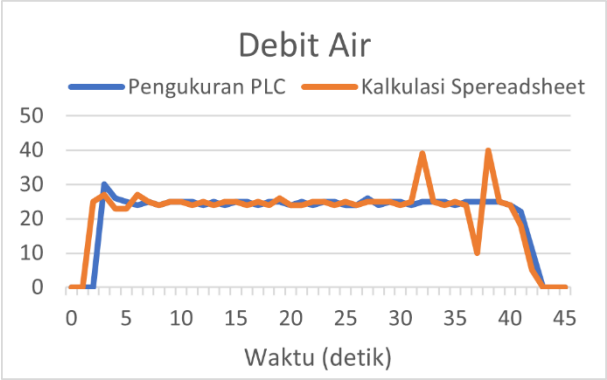
Gambar 4 Diagram Ladder Perhitungan Debit

Setelah dilakukan pengujian, didapatkan grafik ketinggian air seperti yang ditunjukkan pada Gambar 5. Dibutuhkan waktu 40 detik untuk mengisi tangki air dari kosong hingga penuh.



Gambar 5 Respon Ketinggian Air

Debit air dari respon tangki air ditunjukkan pada Gambar 6. Terlihat bahwa terdapat sedikit perbedaan hasil pengukuran debit air antara menggunakan program pada PLC dengan menghitung perubahan ketinggian air yang dicatat PLC dari *FactoryIO*. Hal ini dapat terjadi karena terdapat perbedaan siklus *clock* dengan siklus sampling pencatatan data ketika PLC menjalankan program. Selain itu, terlihat adaya osilasi pada hasil pengukuran debit. Hal ini disebabkan oleh permukaan air yang bergelombang ketika tangki air diisi. Tetapi kondisi ini tidak mempengaruhi pemodelan sistem tangki air.

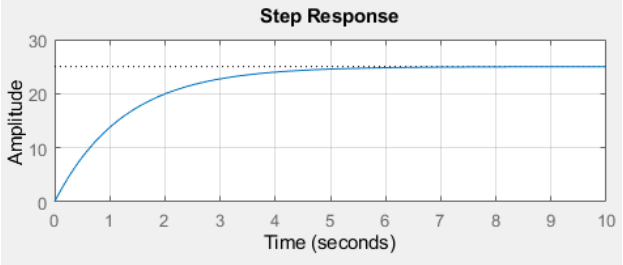


Gambar 6 Hasil Pengukuran Debit Air

Berdasarkan hasil pengukuran debit pada Gambar 7, didapati bahwa waktu untuk mencapai *steady-state* dari nilai 0 adalah 5 detik. Maka dengan persamaan (3) didapatkan $T = 1{,}25$. Nilai K didapatkan dari nilai steady-state maka $K = 25$ sehingga sistem memiliki *transfer function* yang dituliskan seperti persamaan (1)

$$G(s) = \frac{25}{1{,}25s + 1} \quad (1)$$

Ketika disimulasi menggunakan MATLAB, sistem tersebut memiliki *step response* seperti pada Gambar 7.



Gambar 7 *Step Response* Sistem

2. 2.2 Pemodelan PID

Pada penelitian ini, *tuning* PID dilakukan dengan menerapkan PID sebagai *compensator* sistem agar *transfer function* yang dihasilkan sesuai dengan target yang diharapkan. Kontroler PID dapat dituliskan *transfer function* $g_c(t)$ seperti persamaan (2) [5]:

$$g_c(t) = K_p e(t) + K_i \int_0^t e(t)\, dt + K_d \frac{de(t)}{dt} \tag{2}$$

Dengan Transformasi Laplace, didapatkan persamaan (3)

$$G_c(s) = K_p + \frac{K_i}{s} + K_d s \tag{3}$$

Dengan parameter K_i dan K_d dapat dituliskan seperti persamaan (4)

$$K_i = \frac{K_p}{T_i} \text{ dan } K_d = K_p T_d \tag{4}$$

Maka *transfer function* secara utuh didapatkan seperti pada persamaan (5)

$$\frac{C(s)}{R(s)} = \frac{G_c(s)\, G(s)}{1 + G_c(s)\, G(s)} \tag{5}$$

$$= \frac{\left(K_p + \frac{K_i}{s} + K_d s\right)\left(\frac{25}{1{,}25s + 1}\right)}{1 + \left(K_p + \frac{K_i}{s} + K_d s\right)\left(\frac{25}{1{,}25s + 1}\right)}$$

$$= \frac{25\left(K_p s + K_i + K_d s^2\right)}{(1{,}25s^2 + s) + 25\left(K_p s + K_i + K_d s^2\right)}$$

$$= \frac{25\left(K_p s + K_i + K_d s^2\right)}{(25K_d + 1{,}25)s^2 + \left(25\, K_p + 1\right)s + 25\, K_i}$$

$$= \frac{25\left(K_p s + K_i + K_d s^2\right)}{(25K_d + 1{,}25)s^2 + \left(25\, K_p + 1\right)s + 25\, K_i}$$

Persamaan (5) merupakan salah satu sitem orde 2 sehingga hasilnya dapat dituliskan seperti persamaan (6)

$$s^2 + 2\,\zeta\omega_n s + \omega_n^2 = s^2 + \frac{25K_p + 1}{25K_d + 1{,}25}\, s + \frac{25K_i}{25K_d + 1{,}25} \tag{6}$$

Jika diharapkan sistem akhir memiliki *overshoot* maksimal 10% dengan waktu tunak 2 detik, maka didapatkan *damping ratio* seperti persamaan (7)

$$M_P = e^{-\left(\frac{\zeta}{\sqrt{1-\zeta^2}}\right)\pi} \tag{7}$$

$$0{,}1 = e^{-\left(\frac{\zeta}{\sqrt{1-\zeta^2}}\right)\pi}$$

$$-2{,}3 = -\left(\frac{\zeta}{\sqrt{1-\zeta^2}}\right)3{,}14$$

$$0{,}73 = \frac{\zeta}{\sqrt{1-\zeta^2}}$$

$$0{,}53 = \frac{\zeta^2}{1-\zeta^2}$$

$$0{,}53 - 0{,}53\zeta^2 = \zeta^2$$

$$\zeta^2 = \frac{0{,}53}{1{,}53}$$

$$\boldsymbol{\zeta = 0{,}59}$$

Sedangkan *natural frequency* dapat dituliskan seperti persamaan (8)

$$T_s = \frac{4}{\zeta \omega_n} \tag{8}$$

$$2 = \frac{4}{0{,}59\ \omega_n}$$

$$\omega_n = \frac{4}{2 * 0{,}59}$$

$$\boldsymbol{\omega_n = 3{,}39}$$

Menggunakan koefisien *s* pada persamaan (6) didapatkan persamaan parameter K_p dan K_d seperti persamaan (9)

$$2\zeta\omega_n = \frac{25K_p + 1}{25K_d + 1{,}25} \tag{9}$$

$$2 * 0{,}59 * 3.39 = \frac{25K_p + 1}{25K_d + 1{,}25}$$

$$4(25\ K_d + 1{,}25) = 25\ K_p + 1$$

$$100K_d + 5 = 25K_p + 1$$

$$K_p = 4\ K_d + 0{,}16$$

Selanjutnya menggunakan bagian konstanta pada persamaan (6) didapatkan persamaan parameter K_i dan K_d seperti pada persamaan (10)

$$\omega_n^2 = \frac{25K_i}{25K_d + 1{,}25} \tag{10}$$

$$3{,}39^2 = \frac{25K_i}{25K_d + 1{,}25}$$

$$11{,}49(25\ K_d + 1{,}25) = 25\ K_i$$

$$K_i = 11{,}49\ K_d + 0{,}58$$

Jika kontroler yang digunakan adalah PI, sehingga $K_d = 0$, maka persamaan (9) dan (10) memberikan nilai K_p dapat dituliskan seperti persamaan (11) dan K_i seperti (12)

$$K_p = 0{,}16 \tag{11}$$

$$K_i = 0{,}58 \tag{12}$$

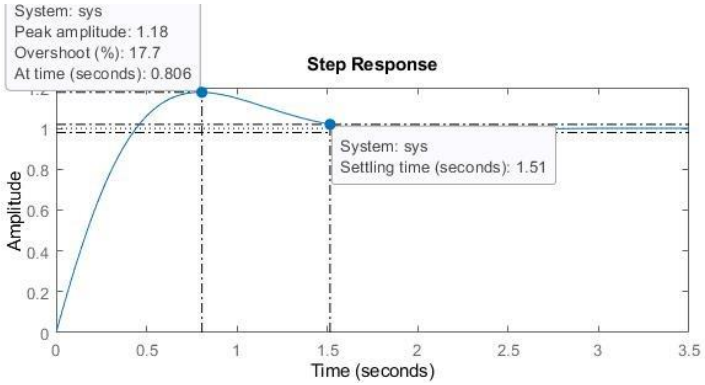
Sehingga T_i yang digunakan dapat dituliskan seperti persamaan (13)

$$T_i = \frac{K_p}{K_i} = \frac{0.16}{0.58} = 0{,}28 \tag{13}$$

Karena Machine Expert hanya memiliki satuan 0.1 pada T_i, maka nilai yang digunakan adalah $\boldsymbol{T_i = 0{,}3}$.

2. 2.3 Simulasi MATLAB

. Nilai parameter T_i berubah menjadi 0.3 sehingga nilai $K_i = 0{,}53$. Maka dengan $K_p = 0{,}16$, $K_i = 0{,}53$, dan $K_d = 0$ dengan sistem dari persamaan (10), didapatkan respon seperti Gambar 8.



Gambar 8 Respon Sistem menggunakan Kontroler PID

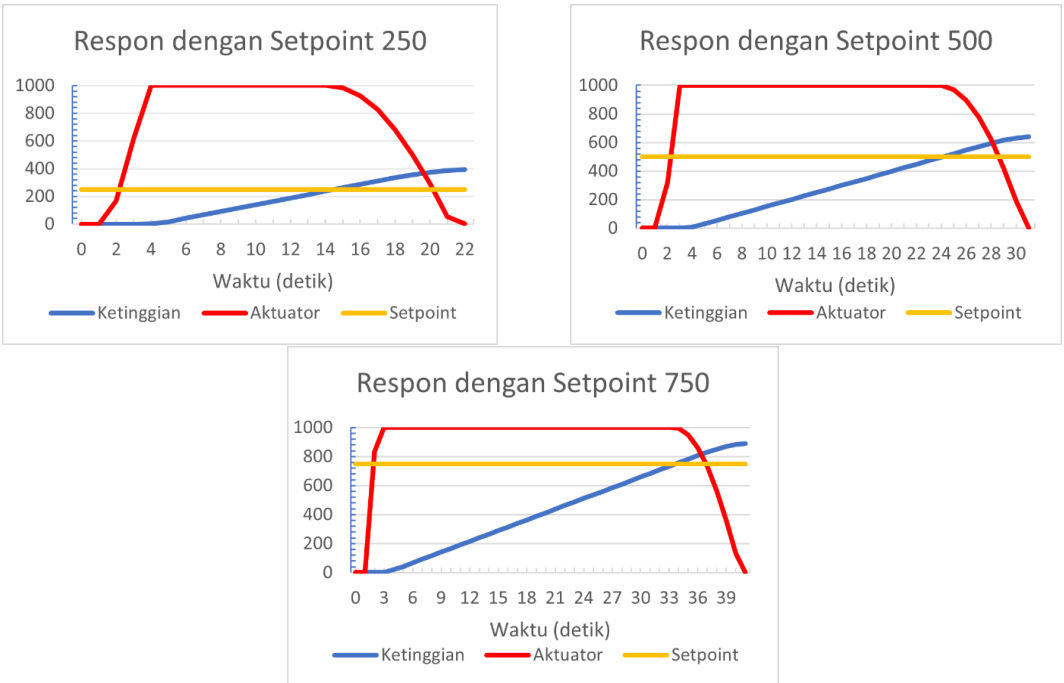
Respon yang didapatkan sedikit berbeda dengan target yang diinginkan, yaitu *overshoot* sebesar 17,7% dan waktu tunak 1,51 detik. Kedua nilai ini selanjutnya digunakan sebagai acuan terhadap respon yang didapatkan menggunakan PLC.

2. 3 Implementasi

Bagian terakhir adalah untuk mengimplementasikan PID ke program PLC yang sudah dicoba sebelumnya. Hasil yang didapat akan dijelaskan lebih detil pada bagian selanjutnya.

3. HASIL DAN PEMBAHASAN

Parameter untuk kontroler PID sudah ditetukan pada bab sebelumnya, yaitu $K_p = 0{,}16$, $T_i = 0{,}3$, dan $T_d = 0$. Pengujian dilakukan sebanyak tiga kali dengan *setpoint* 250, 500, dan 750. Gambar 9 menunjukkan hasil dengan *setpoint* 250, *setpoint* 500, dan *setpoint* 750.



Gambar 9 Respon sistem dengan kontroler PID

Sumbu vertikal mewakili sinyal komunikasi yang dikirim dan diterima PLC menggunakan *Modbus*. Nilainya berkisar dari 0 hingga 1000 untuk mewakili 0 V hingga 10,0 V pada sensor dan aktuator. Kurva ketinggian mewakili sensor ketinggian air pada tangki FactoryIO. Setiap satuan sinyal dari sensor mewakili ketinggian 0.3 cm. Kurva aktuator mewakili sinyal tengangan keluaran aktuator untuk membuka aliran air dengan 0 V adalah menghentikan aliran air dan 10.0 V adalah terbuka seutuhnya. Kurva *setpoint* mewakili target ketinggian air yang akan dicapai. Dengan pengujian setpoin 250, 500, dan 750 maka target ketinggian adalah 75 cm, 150 cm, dan 225 cm.

Pengujian memberikan hasil bahwa dengan *setpoint* 250 terjadi overshoot sejauh 143 selama 8 detik. Pada *setpoint* 500 terjadi overshoot sejauh 140 selama 8 detik. Pada *setpoint* 750 terjadi *overshoot* sejauh 141 selama 8 detik. Maka jika direrata dari ketiga percobaan, *overshoot* menjadi

$$Overshoot = \frac{\frac{143}{250}+\frac{140}{500}+\frac{141}{750}}{3} * 100\% = 34{,}67\% \quad (14)$$

Nilai *overshoot* dan waktu tunak jauh dari target yang diinginkan yaitu 17,1% selama 1,51 detik. Meskipun begitu nilai *overshoot* dan waktu ini konsisten pada ketiga *setpoint*, yaitu pada 140 satuan (atau 42 cm) dengan waktu 8 detik. Hal ini dapat dimungkinkan karena tedapat waktu tunda antara waktu aktuator mengalirkan air hingga air jatuh ke tangki dan terdeteksi sensor ketinggian. Waktu tunda tersebut mempengaruhi hasil perhitungan kotroler PID sehingga terjadi *overshoot* yang berlebihan.

Hasil *overshoot* dan waktu tunak juga jauh dari hasil yang digunakan literatur [14]. Literatur menuliskan bahwa dengan $K_p = 25$, $K_i = 0{,}1$, dan $K_d = 0$ didapatkan waktu bangkit 396 detik, *overshoot* sejauh 0 cm, waktu tunak selama 430 detik, dan *steady-state error* 0,566 cm. Maka parameter PID yang digunakan dalam penelitian memberikan waktu tunak yang jauh lebih cepat, yaitu 8 detik tetapi nilai *overshoot* yang lebih besar yaitu 42 cm.

Solusi untuk memperbaiki *overshoot* dan waktu tunak yang berlebihan adalah dengan memanfaatkan parameter derivatif pada kontroler PID dengan melakukan pemodelan sistem yang lebih detil. Solusi lain adalah dengan memberikan *offset* untuk setpoin pada program PLC untuk menghilangkan *overshoot* konstan sebanyak 140.

4. KESIMPULAN

Setelah dilakukan percobaan simulasi tangki air menggunakan *software FactoryIO*, didapati bahwa *software* ini memiliki kalkulasi gravitasi yang meniru gravitasi di dunia nyata. Akibatnya, terlihat osilasi pada hasil pembacaan ketinggian air karena gelombang air serta tunggu waktu ketika air mengalir dari katup yang dibuka. Hal ini mengakibatkan kontroler PID yang dirancang sebagai *compensator* yang memiliki *overshoot* 17,7% dan waktu tunak 1,51 detik menjadi tidak bekerja sesuai harapan. Terjadi rerata *overshoot* sebesar 34,67% dengan waktu 8 detik. Meskipun hasil yang didapatkan tidak sesuai dengan harapan, pengujian tetap membuktikan bahwa *FactoryIO* dapat dimanfaatkan sebagai alat peraga untuk *hands-on PLC*, baik untuk simulasi komunikasi *Modbu*s atau untuk penerapan PID. Metode *tuning* lain dapat dimanfaatkan agar sistem dengan kendali PID memberikan respon yang diharapkan.

5. SARAN

Machine Expert sebagai program untuk memrogram PLC M221 memiliki keterbatasan waktu *sampling* untuk mencatat nilai yang tersimpan, yaitu per 1 detik. Sedangkan komunikasi *Modbus* memiliki siklus pengiriman data secara *default* 200 ms dan dapat dipercepat sampai dengan 10 ms. Maka agar pengukuran respon sistem menjadi lebih akurat membutuhkan alat atau metode tambahan. Dengan respon sistem yang lebih akurat diharapkan memungkinkan untuk menerapkan metode *tuning* PID lainnya pada praktikum.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Tim Redaksi Jurnal Teknika Politeknik Negeri Sriwijaya yang telah memberi memberi kesempatan, sehingga artikel ilmiah ini dapat diterbitkan.

DAFTAR PUSTAKA

[1] R. Riansyah *et al.*, "Navigasi Garbage Robot (G-Bot) Menggunakan Environment Mapping," *J. Tek.*, vol. 14, no. 1, pp. 73–79, 2020.

[2] E. Prihatini *et al.*, "Pemanfaatan Sensor Jarak dan Sensor Warna pada Proses Penanaman Benih Menggunakan Smart Mini Robot Agriculture," *J. Tek.*, vol. 15, no. 1, pp. 143–151, 2021, [Online]. Available: https://jurnal.polsri.ac.id/index.php/teknika/article/view/3168

[3] S. J. Suji Prasad, R. Sureshkumar, A. Yogaabitha, B. Vijayakumar, M. Sakthivel, and S. Purshothaman, "Development of PLC Based Level Control System for Conical Tank System for Water Treatment Plant," *Proc. - 2023 3rd Int. Conf. Pervasive Comput. Soc. Networking, ICPCSN 2023*, pp. 1563–1568, 2023, doi: 10.1109/ICPCSN58827.2023.00260.

[4] D. Rahmatullah, P. Iradiratu Diah, B. Y. Dewantara, and F. Achmad, "Design and Build of 3 Phase Induction Motor Speed Regulation on Programmable Logic Controller (PLC) Using PID Control Method," *ICRACOS 2021 - 2021 3rd Int. Conf. Res. Acad. Community Serv. Sustain. Innov. Res. Community Serv. Better Qual. Life Towar. Soc. 5*, pp. 275–280, 2021, doi: 10.1109/ICRACOS53680.2021.9701992.

[5] P. Syukrilah, M. Z. Romdlony, and A. S. Wibowo, "Perancangan Alat Peraga Kendali Pid Analog Pada Sistem Kecepatan Putar Motor Dc," *eProceedings Eng.*, vol. 6, no. 2, pp. 2810–2817, 2019.

[6] Z. Jamal *et al.*, "Kinerja Fuzzy Tiga Partisi pada Pengendali Solid Level," *J. Tek.*, vol. 17, no. 2, pp. 373–378, 2023.

[7] F. Aldiansyah, S. K. Wijaya, and P. Prajitno, "Neuro-Fuzzy-based Water Flow Controller in Prototype Plant using Programmable Logic Controller (PLC)," *ICEEIE 2019 - Int. Conf. Electr. Electron. Inf. Eng. Emerg. Innov. Technol. Sustain. Futur.*, no. 2, pp. 64–69, 2019, doi: 10.1109/ICEEIE47180.2019.8981409.

[8] S. Howimanporn, S. Chookaew, and W. Sootkaneung, "Design of PLC for Water Leel Control Employing Swarm Optimization-Based PID Gain Scheduling," *2018 Int. Conf. Control Robot. ICCR 2018*, pp. 63–67, 2018, doi: 10.1109/ICCR.2018.8534490.

[9] S. Hu and P. Zheng, "Research on Cascade PID Advanced Control of Natural Gas Recondenser Based on PLC," *IEEE Jt. Int. Inf. Technol. Artif. Intell. Conf.*, vol. 2022-June, pp. 1440–1444, 2022, doi: 10.1109/ITAIC54216.2022.9836809.

[10] T. P. Hong, "Determine the parameters of PID algorithm in liquid level control controlled by PLC s7 1500 using 3d virtual reality model," *J. Thu Dau Mot Univ.*, no. May, pp. 87–93, 2021, doi: 10.37550/tdmu.ejs/2021.04.260.

[11] A. K. Nuhel, M. M. Sazid, K. Ahmed, M. N. Mahmud Bhuiyan, and M. Y. Bin Hassan, "A PI Controller-based Water Supplying and Priority Based SCADA System for Industrial Automation using PLC-HMI Scheme," *4th IEEE Int. Conf. Artif. Intell. Eng. Technol. IICAIET 2022*, pp. 1–6, 2022, doi: 10.1109/IICAIET55139.2022.9936768.

[12] J. Li and A. Gomez-Espinosa, "Improving PID Control Based on Neural Network," *Proc. - 2018 Int. Conf. Mechatronics, Electron. Automot. Eng. ICMEAE 2018*, pp. 186–191, 2018, doi: 10.1109/ICMEAE.2018.00042.

[13] Schneider Electric, "Modicon M221 Logic Controller Programming Guide," *Online*. Schneider Electric, 2022. [Online]. Available: https://www.se.com/id/id/download/document/EIO0000000976/

[14] S. Buwarda, O. Sistem Permesinan, and P. ATI Makassar, "Development of DCS SCADA Teaching Module," pp. 89–96, 2022.