Pengontrolan Kecepatan pada Permainan Jungkat-Jungkit Automatis Berbasis Metode PID

Automatic Seesaw Speed Design Controller Using PID Method

Septiani Maulizar[1], Ir. Porman Pangaribuan.,M.T[2], Agung Surya Wibowo.,S.T.,M.T[3]
[1,3]Prodi S1 Teknik Elektro, Fakultas Teknik Elektro, Universitas Telkom
[1]septianimaulizar@students.telkomuniversity.ac.id, [2]porman@telkomuniversity.ac.id, [3]agungsw@telkomuniversity.ac.id

Abstrak

Jungkat-jungkit adalah sebuah permainan yang terbuat dari papan lurus yang panjang dimana titik tumpuannya ada pada posisi tengah, jika salah satu sisinya naik sisi yang lain akan turun. Permainan ini biasanya terdapat di taman bermain. Cara bermain jungkat-jungkit adalah masing-masing anak duduk di setiap ujung, kemudian setiap anak bergiliran melonjakkan kaki dari tanah agar jungkat- jungkit dapat dimainkan. Lonjakan pada kedua sisi tentunya akan berbeda.

Tugas akhir yang dirancang ini bertujuan untuk mengontrol kecepatan pergerakan jungkat-jungkit tetap dalam keadaan yang stabil pada kedua sisi yang berat beban nya berbeda secara automatis, Di sisi lain, jungkat-jungkit ini juga dirancang agar tetap bisa dimainkan oleh satu orang anak yang hanya ada di satu sisinya saja.

Jungkat-jungkit ini menggunakan *prototype* sebagai simulasi awal dengan batasan berat di masing - masing sisi maksimum ±300gram dan panjang papan 20 cm di setiap sisi. Alat ini menggunakan motor DC sebagai pengerak, dikontrol oleh mikro*controller* ATMEGA32, menggunakan sensor *rotary encoder* sebagai *feedback* sistem, dan menggunakan *Liqiud Crystal Display* (LCD) sebagai monitoring dan menampilkan kecepatan. Kecepatan putar pada motor DC akan memberikan pergerakan naik turun pada jungkat-jungkit. Algoritma yang digunakan adalah metode PID. Metode ini dilakukan agar mencapai hasil *setpoint* yang di inginkan.

Sistem ini dirancang dengan cukup sederhana dan menggunakan metode PID, dengan menggunakan metode PID diharapkan dapat menghasilkan kinerja sistem jungkat jungkit yang baik, diantaranya *overshoot* dan *settling time* yang rendah pada proses yang berlangsung lambat, namun kelemahan pengendalian PID adalah parameter pengendalian sangat bergantung pada objek kendali.

Kata Kunci: Motor DC, PID, ATMEGA32, Sensor *Rotary Encoder*.

Abstracts

Seesaw is a game that made from a long straight board which the equilibirium point at its center, if one of its side rise then another side will be down. This game usually exist in the playground. The playing method of seesaw is each kid sit in each end, then each kid take turns rise their foot from the ground so that seesaw can be played. The increaement of both side will be different.

This thesis that I designed aim to control velocity of seesaw's movement in stable state at both side that have different load automatically. In other side, this seesaw also designed so that can be played by one child that only exist in one side.

This seesaw use a prototype as initial simulation with maximum load at±300 gram and the length of board is 20 cm in each side. This tool use DC motor as activator, controlled by microcontroller ATMEGA32 that use rotary encoder censor as feedback system, and use Liqiud Crystal Display (LCD) to monitor and display the velocity of seesaw. Angular velocity in DC motor will bring up-down movement on seesaw. The Algorithm that used is PID Method. This method is done to reach desired setpoint.

The system is simply designed and use PID method. Using this PID method is expected can generate good seesaw system performance such as low overshoot and setting time on slow process, but the weakness of PID is the parameter to control depend on the obect.

Keyword: *Speed Controll, PID, ATMEGA32, Rotary Encoder*

1. Pendahuluan

Pada zaman yang modern ini kemajuan teknologi dalam hal permainan anak– anak di Indonesia semakin maju sehingga mengenyampingkan permainan tradisional seperti jungkat-jungkit sudah jarang diminati. Anak - anak lebih tertarik dengan permainan yang terdapat dalam *gadget*. Hal ini menyebabkan anak - anak lebih bersifat individualisme dalam bermain, dan akan menyebabakan kurangnya sosisalisai oleh anak yang berdampak pada proses pembentukan mental, relasi maupun pengetahuan khusunya di bidang permainan tradisional.

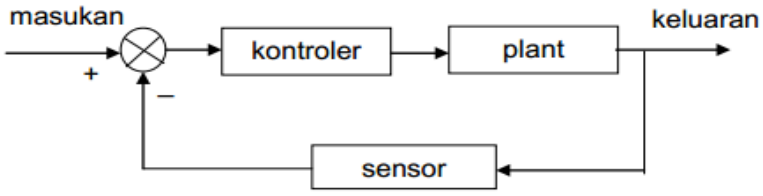
Permainan jungkat-jungkit adalah permainan yang di mana papan panjang dan sempit berporos di tengah, sehingga di saat salah satu ujungnya bergerak naik maka ujung yang lain bergerak turun. Papan jungkat-jungkit di dalam taman bermain lazimnya dirancang agar seimbang di tengah, dimana pada permaianan jungkat-jungkit ini dibutuhkan dua orang yang memiliki berat badan seimbang yang akan mengisi dari setiap ujung dari sisi kanan dan kiri pada permainan ini agar dapat bekerja dengan baik.

Penulis merancang tugas akhir ini mengunakan metode pengontrolan PID yang dapat mengatur posisi naik, turun dan juga posisi stabil pada jungkat-jungkit agar bekerja secara otomatis untuk membantu anak - anak agar dapat bermain jungkat-jungkit dengan masalah tersebut. Dengan berdasarkan deskripsi perancangan sistem dirancang cukup sederhana dan penggunaan metode PID adalah metode yang cukup sesuai karena penggunaan metode ini juga cukup sederhana dan mempunyai kinerja yang baik, diantaranya *overshoot* dan *settling time* yang rendah pada proses yang berlangsung lambat, namun kelemahan pengendalian PID adalah parameter pengendalian sangat bergantung pada objek kendali.

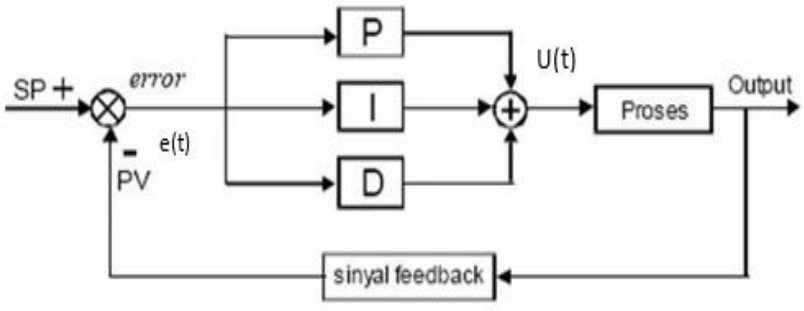
2. Dasar Teori dan Perancangan

2.1. Sistem Kendali

Sistem Kendali adalah proses pengendalian terhadap suatu mesin atau mekanisme yang *output*-nya bisa diubah-ubah sesuai dengan keinginan. Parameter yang mempengaruhi kerja dari sistem kendali adalah pengukuran, perbandingan, perhitungan, dan perbaikan. Dalam sistem kendali terdapat dua jenis kendali, yaitu sistem kendali *closed loop* dan sistem kendali *open loop*. Perbedaan dari kedua jenis sistem kendali tersebut terletak di bagian umpan balik (*feedback*). *Feedback* sendiri adalah fungsi *output* dari respon sistem dengan masukan *input reference*. Sistem kendali *open loop* tidak memiliki *feedback*, tetapi dalam sistem kendali *closed loop* terdapat *feedback* sehingga dalam kasus sistem *closed loop* akan diketahui *output* sistemnya. Pada sistem kendali *closed loop* sinyal *error* dapat diketahui dari perbedaan antara *input reference* dengan sinyal *feedback*, dimana pengendali akan mengurangi atau bahkan menghilangkan *error* dan akan memberikan *output* sistem sesuai dengan yang diinginkan. Dibawah ini adalah diagram blok untuk sistem kendali *close loop*.



2.2. Metode Sistem Kontrol PID



Sistem Kontrol PID *(Proportional–Integral–Derivative*) merupakan *controller* untuk menentukan presisi suatu sistem instrumentasi dengan karakteristik adanya umpan balik pada sistem tersebut (*Feedback*).

Sistem *controll* PID terdiri dari tiga buah cara pengaturan yaitu *controll* P (*Proportional*), D (*Derivative*) dan I (*Integral*), dengan masing-masing memiliki kelebihan dan kekurangan. Dalam implementasinya masing-masing cara dapat bekerja sendiri maupun gabungan diantaranya. Dalam perancangan sistem *controll* PID yang perlu dilakukan adalah mengatur parameter P, I atau D agar tanggapan sinyal keluaran system terhadap masukan tertentu sebagaimana yang diinginkan. Algortima PID *independent* dapat direpresentasikan pada persamaan berikut [6]:

$$U(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_D \frac{de(t)}{dt} \quad \text{.....(2.1)}$$

Dimana:

U(t) = *Output* Kendali

e(t) = Selisih Antara *Setpoint* dengan *Output* dari sinyal *Feedback*

SP = *Setpoint*

PV = *Output* dari Sinyal *Feedback*

K_p = *Gain Proporsional*

K_i = *Gain Integral*

K_D = *Gain Derivative*

Sedangkan dalam bentuk PID *dependent* kita hanya perlu mengganti $K_i = K_p/T_i$ dan $KD = K_p.T_D$. Bisa dilihat bahwa untuk bentuk *dependent* dari sebuah PID nilai dari K_i dan K_D bergantung dengan nilai dari K_p.

2.2.1 Pengedalian *Proportional*

Pengendali ini dikenal sebagai *gain* / penguatan. Pertambahan harga Kp akan menaikkan penguatan sistem sehingga dapat digunakan untuk memperbesar kecepatan respon dan mengurangi *error steady state* (penyimpangan dalam keadaan mantap) penggunaan kendali tipe *proporsional* ini sering tidak memuaskan karena penambahan Kp selain akan membuat sistem lebih sensitif tetapi juga cenderung mengakibatkan ketidakstabilan.

$$u(t) = Kp\ e(t) \quad \text{.....(2.2)}$$

2.2.2 Pengedalian *Integral*

Pengendali ini memastikan bahwa *output* akan sesuai dengan input pada *steady state*. Dengan kendali integral, sedikit *error positif* akan meningkatkan sinyal kendali dan sedikit *error negative* akan menurunkan sinyal kendali.

$$u(t) = Ki \int_0^t e(t)d\tau \quad \text{.....(2.3)}$$

Selama eτ tidak bernilai nol, persamaan diatas menunjukkan bahwa sinyal kendali ut akan konstan. Pengendali integral akan selalu menghasilkan *error steady state* bernilai nol.

2.2.3 Pengendali *Derivative*

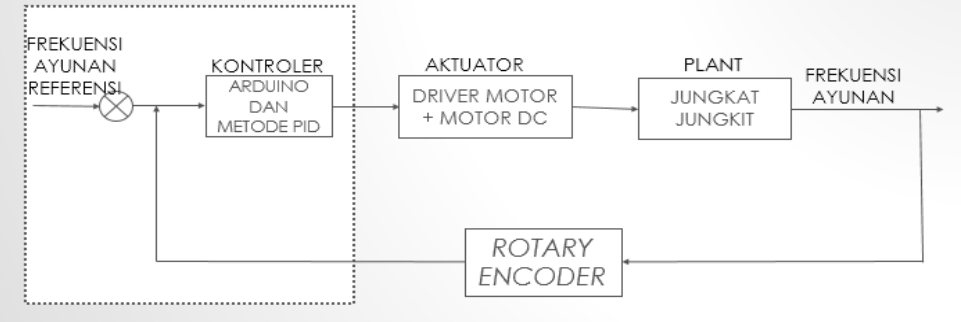
Pengendali ini meningkatkan kestabilan pada sistem kendali *close loop*. Karena terjadi proses yang dinamis, hal tersebut akan memerlukan sedikit waktu sebelum terjadi perubahan *variabel* kendali terlihat pada *output* yang akan menyebabkan sistem kendali akan terlambat untuk memperbaiki *error* yang ada. Kendali *Derivative* hanya berubah saat ada perubahan *error* sehingga saat error statis kontrol ini tidak akan bereaksi.

$$u(t) = Kd \frac{de(t)}{dt} \quad \text{.....(2.4)}$$

Kerja pengontrol *diferensial* hanyalah *efektif* pada lingkup yang sempit, yaitu pada periode peralihan. Oleh sebab itu pengontrol *diferensial* tidak pernah digunakan tanpa ada *controller* lainnya.

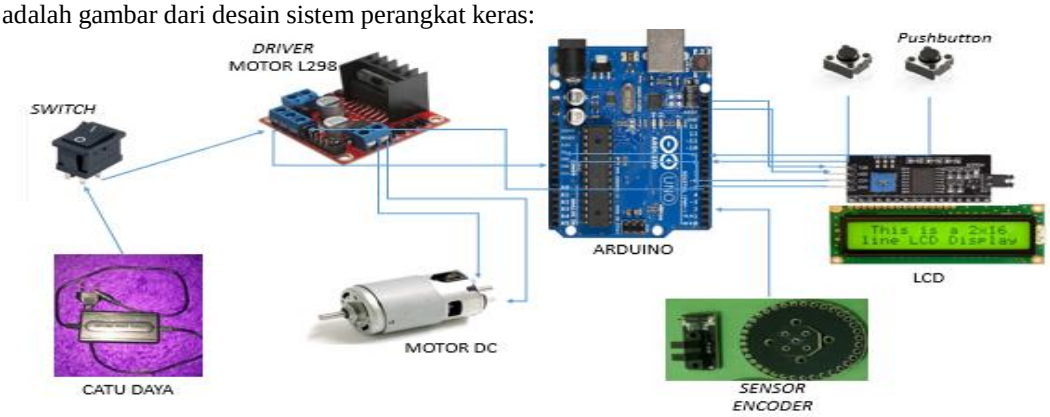
2.3. Blok Diagram Umum

Sistem jungkat jungkit ini dirancang bertujuan untuk mempertahankan kecepatan pada jungkat jungkit apabila salah satu berat badan anak-anak yang sangat jauh berbeda ataupun anak-anak yang ingin bermain tanpa ada pasangan untuk menemani pada sisi lain, sistem akan *automatis* berkerja untuk menyeimbangkan kecepatan sesuai *setpoint* yang di berikan. Perancangan sistem kontrol tugas akhir ini secara garis besar dapat digambarkan sebagai berikut:



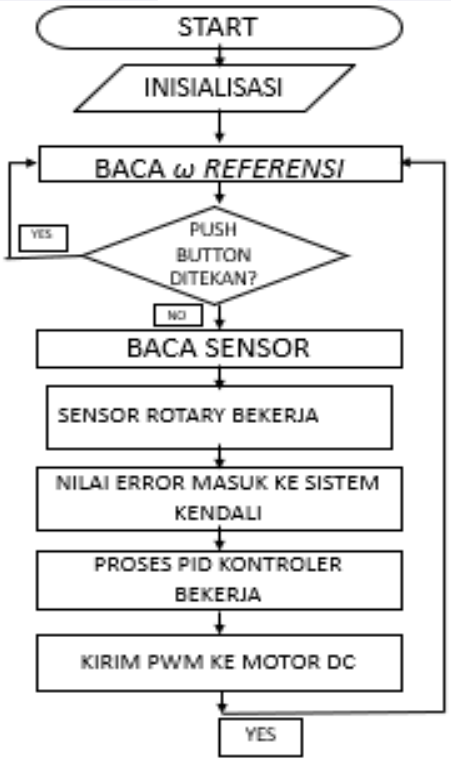
2.4. Blok Diagram keras

Untuk pengimplementasian dan merealisasikan sistem, dirancanglah sebuah perangkat keras. Berikut adalah gambar dari desain sistem perangkat keras:



2.5. Diagram Alir

Diagram alir sistem kendali tugas akhir ini dapat digambarkan sesuai gambar 3.



3. Hasil Percobaan dan Analisa

3.1. Respon Kecepetan Motor DC Terhadap Nilai PWM

Pengujian ini bertujuan untuk mengetahui respon kecepatan putar terhadap nilai PWM yang diberikan. Respon yang terlihat akan ditampilkan berupa jumlah pulsa (count) yang terhitung tiap satu detik. Pengujian ini menggunakan Arduino uno, motor DC, dan sensor rotary encoder serta piringan nya. Respon kecepatan motor terhadap nilai PWM dapat dilihat pada Tabel IV-1.

Tabel III-1 Respon Kecepatan Motor Terhadap Nilai PWM

No	PWM	Tegangan (*Volt*)	Kecepatan Putar Motor DC (*Count*/detik)	Frekuensi Ayunan
1	0	0	0	0
2	10	0,1 v	0	0
3	20	0,259 v	0	0
4	30	0,418 v	0	0
5	40	0,585 v	0	0
6	50	2,29 v	0	0
7	60	3,64 v	62	1,72222
8	70	4,64 v	71	1,97222
9	80	5,44 v	86	2,38889
10	90	6,12 v	96	2,66667

Hasil yang didapat menunjukkan bahwa motor DC baru mulai bergerak pada saat PWM 60 dengan tegangan 3,64 Volt.Hasil yang didapat menunjukkan bahwa setiap makin bertambahnya PWM yang diberikan, maka kecepatan putar dan frekuensi ayunan yang terhitung pada sensor *encoder* akan semakin besar. Jadi, PWM, kecepatan putar motor, dan frekuensi ayunan berbanding lurus. Apabila nilai PWM yang diberikan semakin tinggi, maka semakin tinggi pula kecepatan putar motor, dan frekuensi ayunan atau sebaliknya. Grafik respon kecepatan motor DC terhadap nilai PWM dapat tergambar pada Gambar IV.3. Data yang didapat menunjukan bahwa sistem bersifat *non-linear*.

3.2 Analisis Pengujian dengan *Controller Proportional* yang Berbeda-beda

Dari berbagai pengujian yang telah dilakukan didapatkan hasil berupa tabel lengkap sebagai berikut:

Hasil pengujian perubahan nilai respon motor DC yang hanya menggunakan *controller Proportional*:

Tabel IV. 1 Pengujian dengan *Controller Proportional* yang Berbeda-beda

Nilai Kontroler	*Setpoint*	Pencapaian Stabil	Keterangan
Kp = 0.55	60 CPS	8 detik	Sistem masih mengalami osilasi yang besar, *settling time* tercapai pada saat 8 detik menuju *setpoint*
Kp = 0.6	60 CPS	6 detik	Sistem masih mengalami osilasi, pencapaian *settling time* menuju *setpoint* adalah 6 detik
Kp = 0.69	60 CPS	4 detik	Sistem masih mengalami osilasi, *settling time* dicapai pada saat 4 detik menuju *setpoint*

Dari Table IV.1 nilai parameter Kp = 0.55 pencapaian nilai menuju *setpoint* adalah 5 detik dan tetap mengalami osilasi. Parameter Kp = 0.6 masih mengalami osilasi dan *overshoot* pencapaian nilai menuju stabil adalah 6 detik. Dan untuk parameter Kp = 0.69 tetap masih mengalami *overshoot* dan osilasi dan nilai yang dicapai menuju stabil adalah 6 detik.

Dari ketiga pengujian maka nilai Kp yang baik untuk di implementasikan adalah sebesar 0.55 karena, osilasi dan pencapaian nilai menuju nilai stabil lebih baik dari pada kedua parameter Kp lainnya yaitu 0.6 dan 0.69. Pada percobaan ini hanya menggunakan *controller* P. Nilai dari P itu sendiri akan terus bernilai sebanding (*proportional*) dengan *variabel error* sistem (P =Kp.*error*). Hal tersebut sesuai dengan sifat *controller proportional* yaitu memberikan efek pengurangan *rise time* tetapi tidak menghilangkan *error* nya**.**

3.2 Analisis Pengujian dengan *Controller Proportional-Integral* yang Berbeda-beda

Hasil pengujian perubahan nilai respon motor DC yang hanya menggunakan *controller Proportional-Integral* yang:

Tabel IV. 2 Pengujian dengan *Controller Propotional-Integral* yang Berbeda-beda

Nilai *Controller*	SetPoint	Settling time	Keterangan
Kp=0.55 Ki= 0.3	80 CPS	7 detik	Sistem masih mengalami *overshoot dan* osilasi pencapaian *settling time* menuju *setpoint* selama 7 detik
Kp=0.55 Ki= 0.4	80 CPS	5 detik	*Overshoot dan* osilasi sistem membaik dan pencapaian *settling time* menuju *setpoint* selama 5 detik
Kp=0.55 Ki= 0.5	80 CPS	11 detik	*Overshoot dan* osilasi sistem baik dan pencapaian *settling time* selama 11 detik untuk menuju *setpoint*

Pada ketiga pengujian nilai parameter Kp = 0.55 dan Ki = 0.3 membutuhkan waktu untuk menuju *setpoint* sebesar 7 detik. Pengujian pada parameter Kp = 0.55 dan Ki = 0.4 membutuhkan waktu untuk mencapai *setpoint* selama 5 detik. Dan untuk pengujian pada parameter Kp = 0.55 dan Ki = 0.5 membutuhkan waktu untuk pencapaian *setpoint* selama 11 detik, pada pengujian ini sistem tetap mengalami *overshoot* dan osilasi tetapi sangat kecil sehingga dapat dikatakan tidak ada. Bisa dilihat *overshoot* setelah sistem diberikan kendali Integral maka, *error* tidak sebesar *overshoot* dari sistem yang hanya menggunakan kendali proporsional. Hal tersebut sesuai

dengan sifat *controller Proportional-Integral* yaitu memberikan efek penambahan *rise time* tetapi mampu menghilangkan *error* nya.

Dari ketiga pengujian nilai parameter Kp dan yang baik untuk di implementasikan adalah sebesar Kp = 0.55 dan Ki = 0.4 karena, pencapaian nilai *setpoint* dan waktu pencapaian *setpoint* yang diinginkan lebih baik dari pada kedua parameter Kp dan Ki lainnya yaitu Kp = 0.55 dan Ki = 0.5 dan Kp = 0.55 dan Ki = 0.3.

3.3 Analisis Pengujian *Controller Proportional-Integral* dengan Perubahan *Setpoint*

Tabel IV. 3 Pengujian dengan *Controller Proportional-Integral* dengan Perubahan *Setpoint*

Nilai *Controller*	SetPoint	Settling Time	Keterangan
Kp=0.55 Ki= 0.4	60 RPS	11 detik	Osilasi sistem kurang baik, pencapaian *settling time* menuju *setpoint* adalah 11 detik
Kp=0.55 Ki= 0.4	70 RPS	12 detik	Osilasi membaik dan *settling time* menuju *setpoint* dicapai pada saat 12 detik
Kp=0.55 Ki= 0.4	80 RPS	10 detik	Osilasi membaik dan *settling time* dicapai pada detik ke-10

Berdasarkan hasil pengujian perubahan *setpoint*, ketika nilai *setpoint* 60 *settling time* yang diperoleh adalah 11 detik, ketika *setpoint* bernilai 70 *settling time* yang diperoleh adalah 12 detik, dan ketika nilai *setpoint* 80 *settling time* yang diperoleh adalah 10 detik, dapat disimpulkan bahwa sistem telah berjalan dengan baik karena sudah mengikuti *setpoint* yang di berikan walaupun *setpoint* dirubah secara langsung pada saat sistem sedang bekerja.

3.4 Analisis Pengujian dengan Beban

Berdasarkan hasil pengujian yang dilakukan maka, hasil pengujian respon frekuensi ayunan pada sistem jungkat-jungkit automatis dapat dilihat pada tabel berikut :

Tabel IV. 5 Hasil Respon Sistem Saat Ada Beban

Pengujian ke-	*Settling Time*
Pengujian 1(0gr-100gr)	7 detik
Pengujian 2(200gr-0gr)	6 detik
Pengujian 3(100gr-200gr)	8 detik
Pengujian 4(100gr-300gr)	6 detik
Pengujian 5(200gr-100gr)	6 detik
Pengujian 6(200gr-300gr)	8 detik
Pengujian 7(300gr-100gr)	6 detik
Pengujian 8(300gr-200gr)	7 detik

Berdasarkan hasil pengujian yang dilakukan, berat beban sisi satu 0gram dan sisi dua 100gram memperoleh *settling time* selama 7 detik. Saat berat beban disatu sisi diperbesar menjadi 200gram dan memperoleh *settling time* selama 6 detik. Pada saat berat disisi satu 100gram dan disisi dua 200gram dengan *settling time* selama 8 detik. Pada berat beban disisi satu sebesar 100gram dan disisi kedua 300 gram memperoleh *settling time* selama 6 detik. Pada berat beban disisi satu 200 gram dan disisi dua 100 gram memperoleh *settling time* selama 6 detik.

Pada berat beban disisi satu jungkat-jungkit sebesar 200gram dan disisi kedua sebesar 300gram memperoleh *settling time* selama 8 detik. Pada berat beban disisi satu 300 gram dan disisi dua 100 gram memperoleh *settling time* selama 6 detik. Pada pengujian 8, disisi satu jungkat-jungkit memiliki berat 300 gram dan disisi kedua 200gram memperoleh *settling time* selama 7 detik.

Frekuensi ayunan didapat dari hasil *count* yang dibaca sensor *encoder* dibagi dengan jumlah lubang piringan pembaca *count* yang berjumlah 36 lubang (*count*/36). Dari hasil pengujian dapat disimpulkan semakin besar berat beban yang dipakai dikedua sisi jungkat-jungkit, maka semakin besar pula osilasi yang dihasilkan tetapi sistem tetap akan selalu stabil pada frekuensi yang dihasilkan.

4. Kesimpulan

Berdasarkan hasil pengujian dan analisis sistem yang dibuat maka didapat beberapa kesimpulan antara lain sebagai berikut:

1. *Prototype* jungkat-jungkit *automatis* sudah berjalan dengan baik menggunakan metode kontrol PID.
2. Nilai CPS dan frekuensi ayunan yang didapat akan selalu menuju *setpoint* yang diinginkan.
3. *Prototype* jungkat-jungkit dapat dimainkan dengan perbedaan berat beban pada kedua sisi. Respon sistem akan tetap mengikuti *setpoint* yang diinginkan walaupun terdapat berat beban di salah satu sisi ataupun di kedua sisinya dengan menerapkan metode kontrol PID.
4. Dalam hasil pengujian didapatkan nilai parameter kendali Kp = 0.55 dan Ki = 0.4 yang bekerja dengan ts = 1 detik agar sistem bisa bekerja dengan optimal.
5. Respon kecepatan motor terhadap PWM berbanding lurus dengan kecepatan putar motor DC dan frekuensi ayunan. Apabila PWM semakin tinggi maka kecepatan putar motor dan ayunan yang terbaca semakin tinggi juga. Data yang didapat menunjukan bahwa sistem bersifat non-linear.

5. Daftar Pustaka

[1] Dionisius, Ramaditya Putra Fatruan. ”Rotary Encoder,” . Available : https://www.scribd.com/doc/95731167/Rotary-Encoder. [Diakses 04 November 2016, 12:26 WIB]
[2] Huang, Han-Way, *PIC Microcontroller: An Introduction to Software and Hardware Interfacing*. United States of America. ISBN 1-4018-3967-3
[3] Smith, Jack R, *Programming The PIC Microcontroller with MBasic*. United States of America. ISBN 0-7506-7946-8
[4] Barret, Steven F, *Arduino Microcontroller Processing for Everyone!*. *Second Edition.* Southbourn Melbourne University. ISBN 9781608458608
[5] Gani, Fadlan Nuran. “*Pulse Widht Modulation(PWM),*’. Available: http://robotic-electric.blogspot.co.id/2012/11/pulse-width-modulation-pwm.html . [Diakses 30 November 2016, 15:11 WIB]
[6] Sugiono, Djoko. 07 February 2015. Model Matematika Motor DC . Diakses Oktober 01 2017. http: // www.vedcmalang.com/ pppptkboemlg/index.php/ menuutama/ listrik-electro/1322-motor-dc
[7] E-educativa es, “Motor Dc,” 2012.
[8] C. T. Manual, “Pulse Width Modulated (PWM),” *ReVision*, pp. 1–13.
[9] D. M. A. Y. Occur, “Intel ® USB 3 . 0 eXtensible Host Controller Driver,” vol. 1, no. January, pp. 1–32, 2012.
[10] F. Breu, S. Guggenbichler, and J. Wollmann, “Arduino Uno,” *Vasa*, vol.328, p. 13, 2008.
[11] C. Knospe, “PID control,” *Control Syst. IEEE*, vol. 26, no. 1, pp. 216–251,2006.