JREC (*Journal of Electrical and Electronics*)
ISSN (p) : 2302-5883 ISSN (e) : 2550-0899
Vol.8 No.1 Mei 2020, 23-34

Kinerja Robot Self Balancing pada Lintasan Bergelombang Menggunakan Pengendali PID Digital

Bayu Cendana, Sugeng, Muhammad Ilyas Sikki
Program Studi Teknik Elektro Fakultas Teknik Universitas Islam "45" Bekasi
Email: b.cendana95@gmail.com

Abstrak

Robot *self balancing* beroda dua (*two wheels self-balancing robot*) merupakan robot *mobile* yang memiliki dua buah roda di sisi kanan dan kirinya. Robot ini tidak akan seimbang tanpa adanya kontroler. Pengendali PID dapat digunakan untuk mengendalikan posisi robot supaya tetap seimbang pada saat berjalan di atas permukaan yang bergelombang dengan mengatur nilai konstanta proporsional (Kp), konstanta integral (Ki), dan konstanta derivatif (Kd) yang sesuai dengan menggunakan metode *trial* and *error*. Analisa sistem kendali berdasarkan 3 aspek analisis, yaitu analisis respon *transient*, analisis respon *steady state*, dan analisis stabilitas. Hasil yang didapat menggunakan metode *trial and error* pada penelitian ini, yaitu robot berhasil mempertahankan keseimbangannya saat berjalan di atas permukaan yang bergelombang dengan nilai Kp = 50, Ki = 170, dan Kd = 2,2. Hasil analisis dari parameter PID tersebut, respon robot *self balancing* termasuk dalam kategori *overdamped* dengan nilai % *error steady state* 1,02% dan kestabilan robot termasuk dalam kategori marjinal stabil.

Kata Kunci: Robot *Self Balancing*, PID, *Trial and Error*.

Abstract

A two-wheeled self-balancing robot is a mobile robot with two wheels on each side. It is equipped with a controller which balances it. A PID controller can be used to control robot's position so that it can move steadily in bumpy surfaces. Its PID controller moves by calculating suitable proportional constant (Kp), constant integral (Ki) and derivative constant (Kd) using trial and error method. The analysis of controlling system is based on three aspects of analysis; transient response analysis, steady state response analysis, and stability analysis. The results of present research show that the robot maintained its balance when it moved on a bumpy surface with Kp of 50, Ki of 170 and Kd of 2,2. The analysis of PID parameter suggests that the response of self balancing robot is categorized as overdamped with the percentage of error steady state of1,02% while its stability is categorized as marginally stable.

***Key Words**: Self Balancing Robot, PID,Trial, and Error.*

PENDAHULUAN

Saat ini, perkembangan teknologi robotika mampu meningkatkan kualitas berbagai sektor kehidupan manusia baik industri, hiburan, dan pendidikan, terutama dengan adanya robot cerdas. Beberapa cara yang dapat dilakukan untuk menambah tingkat kecerdasan pada robot adalah dengan menambahkan sensor, metode kontrol, hingga kecerdasan buatan.

Ada dua tingkatan pengendali robot, yaitu pengendali tingkat rendah (*low level controller*) dan pengendali tingkat tinggi (*high level controller*)[1]. Pengendali tingkat rendah (*low level controller*) digunakan untuk memprogram *hardware* agar robot dapat melakukan perilaku yang diinginkan. Sebagai contoh, memprogram *hardware* untuk mengendalikan posisi robot[2]. Sementara itu, pengendali tingkat tinggi (*high level controller*) digunakan untuk memprogram perilaku robot agar dapat melaksanakan tugas yang diinginkan. Misalnya, memprogram perilaku robot, sehingga robot dapat berpindah dari satu tempat ke tempat lain yang diinginkan atau menghindari sebuah rintangan[3]-[6].

Robot *self balancing* beroda dua (*two wheels self-balancing robot)* merupakan robot *mobile* yang memiliki dua buah roda di sisi kanan dan kirinya. Robot jenis ini tidak akan seimbang tanpa adanya kontroler[7].

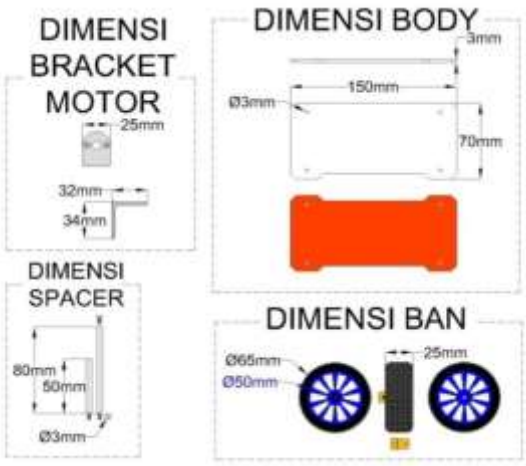
Pengendali PID digital merupakan bentuk lain dari pengendali PID yang diprogram dan dijalankan menggunakan komputer atau mikrokontroler. Untuk dapat mengimplementasikan PID digital di komputer atau mikrokontroler, pengendali PID analog harus diubah terlebih dahulu ke bentuk digital[8]. Pengendali PID dapat berperan sebagai pengendali tingkat rendah (*low level controller*) bagi robot *self balancing*.

Selain itu, pengendali PID juga berguna untuk mengendalikan posisi robot agar tetap seimbang dengan mengatur nilai konstanta proposional (Kp), konstanta integral (Ki), dan konstanta derivatif (Kd) yang sesuai[9][10]. Kinerja PID pada robot *self balancing* perlu dianalisa, apakah pengendali PID mampu menjaga keseimbangan robot *self balancing*, sehingga robot dapat beradaptasi terhadap gangguan. Salah satunya ketika robot *self balancing* berjalan di lintasan yang bergelombang[11][12]. Penelitian ini bertujuan untuk menganalisis sistem pengendali tingkat rendah (*low level controller*) pada robot *self balancing*, dimana pengendali yang digunakan adalah pengendali PID dan dengan keadaan robot *self balancing* yang berjalan pada lintasan bergelombang.

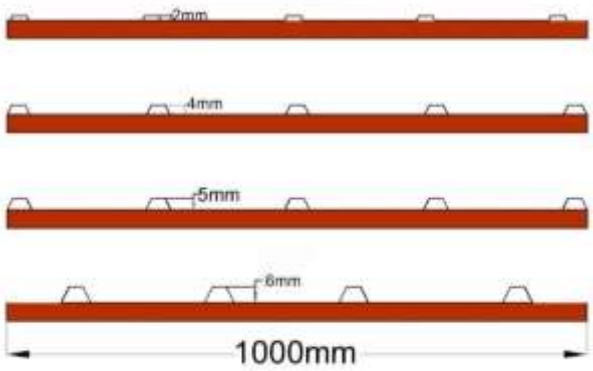
METODE PENELITIAN

Perancangan Sistem Mekanik

Meliputi perancangan dimensi, material, dan desain struktur rangka robot *self balancing*, serta desain struktur lintasan bergelombang. Material yang digunakan untuk rangka robot *self balancing* adalah bahan *acrylic* dengan struktur dan dimensi sebagaimana terlihat pada gambar 2. Sedangkan material yang digunakan untuk lintasan bergelombang yaitu menggunakan bahan kayu triplek sebagai alas ditambah dengan *ducting* kabel berbahan PVC sebagai pembentuk gelombang untuk lintasan. Adapun struktur dan dimensi lintasan bergelombang sebagaimana terlihat pada gambar 3.



Gambar 1. Perancangan Sistem Mekanik

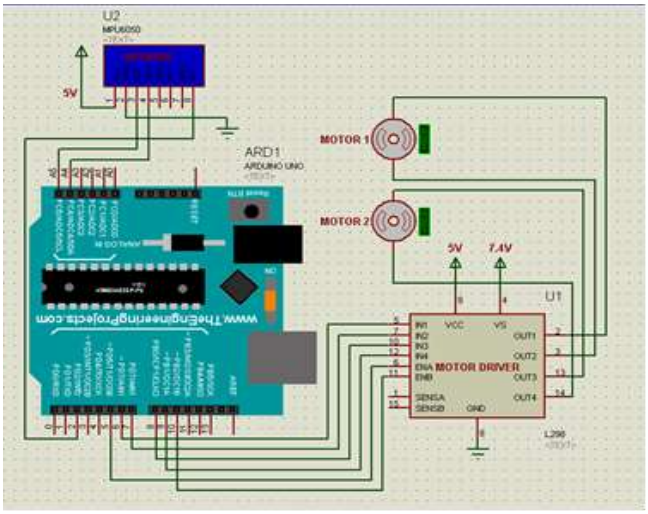


Gambar 2. Perancangan Sistem Mekanik Lintasan Bergelombang

Perancangan Sistem Elektrik

Perancangan sistem elektrik pada penelitian ini meliputi 1 unit sensor MPU 6050, 1 unit *motor driver* L298N, 2 unit motor DC, 1 unit mikrokontroler Arduino Uno, dan 3 unit baterai *Li-ion* sebagai *power*

supply. Perakitan sistem elektrik dapat dilihat pada gambar 4, sementara untuk penjelasan konfigurasi pin yang saling terhubung dapat dilihat pada tabel 1.



Gambar 3. Perancangan Sistem Elektrik

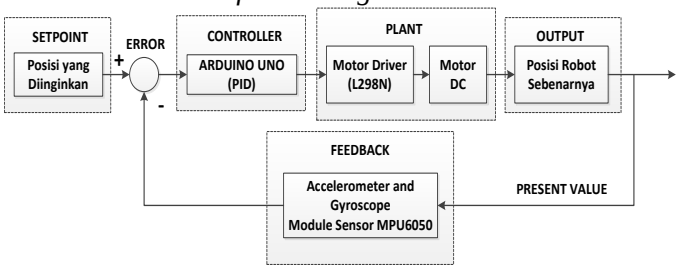
Tabel 1. Konfigurasi Pin

No.	Pin					
	Arduino Uno	**MPU 6050**	**L 298N**	**Motor DC 1**	**Motor DC 2**	**Baterai**
1	Vin	-	12V	-	-	**(+)**
2	Gnd	Gnd	Gnd	-	-	**(-)**
3	5V	Vcc	5V	-	-	-
4	A4	SDA	-	-	-	-
5	A5	SCL	-	-	-	-
6	D2	INT	-	-	-	-
7	D5	-	EN A	-	-	-
8	D6	-	IN 1	-	-	-
9	D7	-	IN 2	-	-	-
10	D8	-	IN 3	-	-	-
11	D9	-	IN 4	-	-	-
12	D10	-	ENB	-	-	-
13	-	-	Out1	**A**	-	-
14	-	-	Out2	**B**	-	-
15	-	-	Out3	-	**A**	-
16	-	-	Out4	-	**B**	-

Perancangan Sistem Kendali

Yang termasuk dalam perancangan sistem kendali pada penelitian ini adalah sebagai berikut:

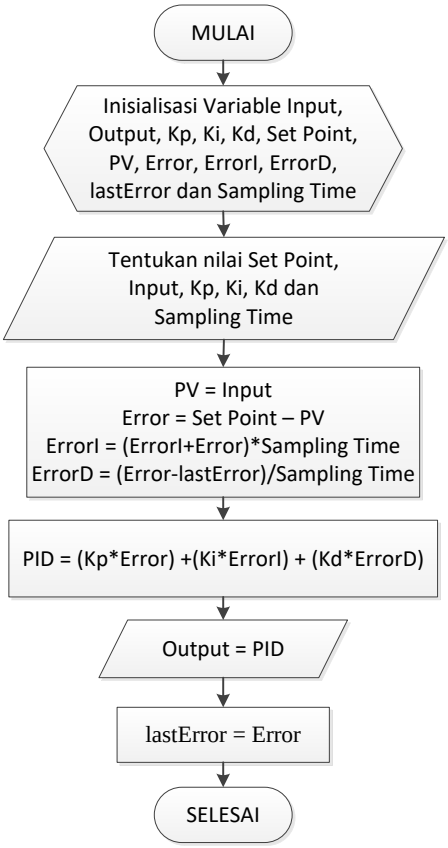
a. Blok diagram sistem kendali Robot *Self Balancing*



Gambar 4. Blok Diagram Robot *Self Balancing*

Sistem kerja dari robot *self balancing* menggunakan sistem kendali *close* loop. Sebab, diperlukan adanya umpan balik (*feedback*) untuk mengendalikan *output* sistem. Secara umum blok diagram sistem kendali robot *self balancing* dapat dilihat pada gambar 4. Nilai masukan (*set point*) dari sistem berupa nilai derajat yang dapat diatur pada kode program. Namun, pada penelitian ini penulis mengatur nilai *set point* sebesar 174. Keluaran sistem kendali adalah nilai tegangan berbentuk *Pulse Width Modulation* (PWM). Adapun Sensor MPU 6050 berfungsi mengukur nilai posisi robot (*Present Value*) kemudian membuat nilai pembacaan menjadi nilai umpan balik (*feedback*) yang bernilai negatif.

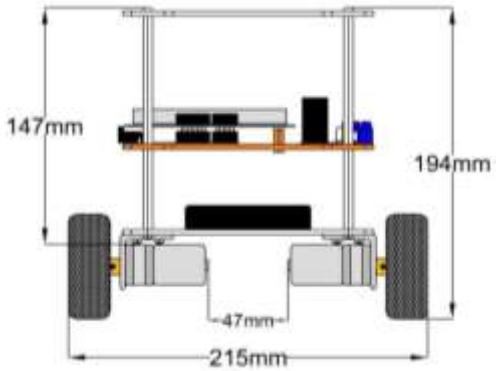
b. Diagram alir PID
Diagram alir PID dapat dilihat pada gambar 5.



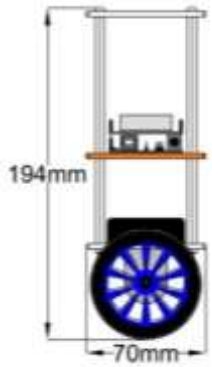
Gambar 5. Diagram Alir PID

Perakitan Sistem Mekanik

Perakitan sistem mekanik pada penelitian ini dapat dilihat pada Gambar 6-8.



Gambar 6. Perakitan Mekanik Tampak Depan

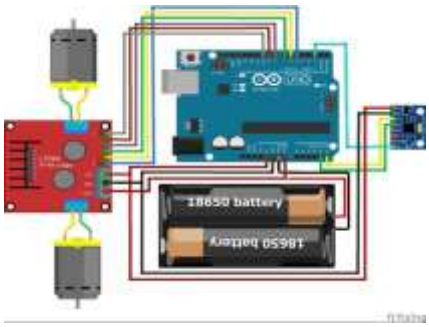


Gambar 7. Perakitan Mekanik Tampak Samping

Gambar 8. Perakitan Mekanik Tampak Atas

Perakitan Sistem Elektrik

Perakitan sistem mekanik pada penelitian ini dapat dilihat pada gambar 9.



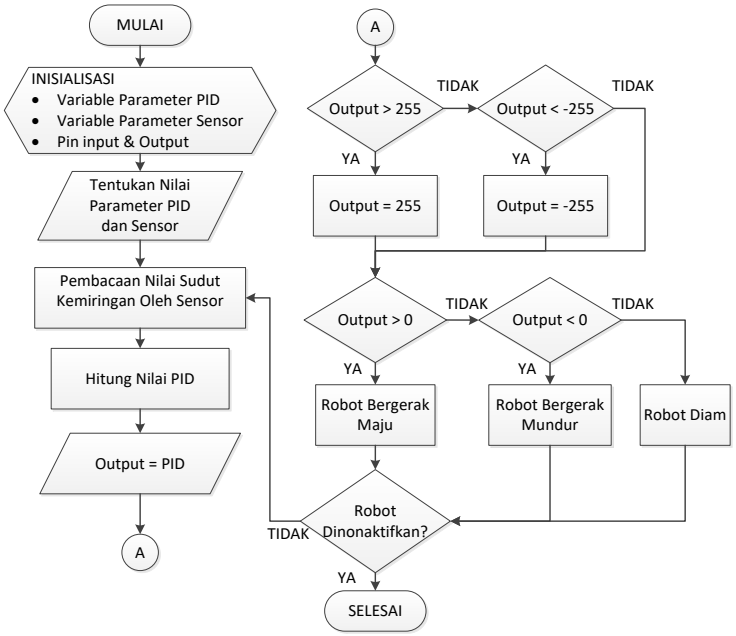
Gambar 9. Perakitan Sistem Elektrik

Perakitan Sistem Kendali

Pembuatan kode program pada *software* Arduino IDE dapat dijelaskan dengan diagram alir pada gambar 11, yang menunjukkan urutan kerja sistem robot *self balancing* yang dibuat. Langkah pertama, yaitu mengatur parameter PID dan sensor MPU6050. Langkah selanjutnya yaitu menentukan keseimbangan robot. Terakhir, jika nilai-nilai yang sudah di set pada langkah sebelumnya sudah dimasukkan maka program akan menghitung kecepatan serta arah putar motor DC.

Tuning PID

Penalaan konstanta PID, yaitu meliputi konstanta proporsional (Kp), konstanta integral (Ki), dan konstanta derivatif (Kd), merupakan salah satu hal yang terpenting pada penelitian ini. Sebab, robot *self balancing* memerlukan nilai konstanta PID yang tepat untuk menjaga posisi robot dari gangguan, sehingga robot bisa tetap menjaga keseimbangannya. Proses *tuning* PID dapat dilakukan dengan berbagai metode, dan metode yang digunakan pada penelitian ini adalah *trial and error*.



Gambar 10. Diagram Alir Algoritma Kode Program Robot *Self Balancing*

Pengujian Sistem Elektrik

Pengujian sistem elektrik dilakukan dengan cara menguji komponen elektrik pada robot *self balancing* yang meliputi pengujian *power supply*, pengujian mikrokontroler, pengujian sensor, pengujian *motor driver*, dan pengujiian motor DC.

Pengujian Sistem Kendali

Pengujian sistem kendali dilakukan dengan membandingkan respon robot ketika berjalan di atas permukaan yang bergelombang dengan ketinggian gelombang yang berbeda seperti pada gambar (tinggi 2 mm, 4 mm, 5 mm, dan 6 mm). Pengujian ini bermaksud mencari nilai ketinggian gelombang maksimum yang dapat dilalui oleh robot *self-balancing*.

Analisa Data

Data yang utama pada penelitian ini berupa grafik respon dari sistem kendali pada robot *self balancing* dimana respon dari sistem kendali tersebut dianalisa berdasarkan hal-hal sebagai berikut:

1. Analisis Respon *Transient*
   Respon *transient* adalah respon sistem yang berlangsung dari keadaan awal hingga masuk dalam keadaan *steady state.* Parameter yang digunakan untuk mengukur kualitas respon *transient* antara lain:
   a) *Time Constant*
      Time Constant adalah waktu yang diperlukan oleh sistem untuk mencapai 63% dari nilai target akhir.
   b) *Rise Time*
      Rise time adalah waktu untuk respon bergerak dari 10% sampai 90% dari nilai *setpoint*.
   c) *Settling Time*
      Settling time adalah ukuran waktu yang menyatakan respon telah masuk 2% dari nilai *steady state*.
   d) Frekuensi Natural
      Frekuensi natural menjelaskan kecepatan respon. Rasio redaman menjelaskan tentang *respon transient* alami dan berapa banyak *overshoot* yang terjadi.
      1) Jika rasio redaman = 0, maka perilaku *undamped*
      2) Jika rasio redaman < 1, maka perilaku *underdamped*
      3) Jika rasio redaman = 1, maka perilaku *critical damped*

4) Jika rasio redaman > 1, maka perilaku *overdamped*

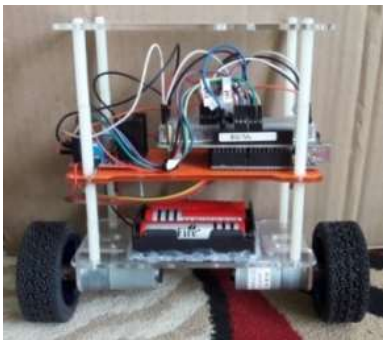
e) *Peak time*
Waktu yang diperlukan respon dari awal hingga mencapai puncak pertama *overshoot* (nilai respon maksimum).

2. Analisa *Response Steady State*
Respon *steady state* adalah respon sistem yang berlangsung saat respon mulai masuk dalam keadaan *steady state* sampai waktu yang tak terbatas. Parameter yang digunakan untuk menganalisa respon *steady state* yaitu *error steady state*. Untuk memperoleh gambaran aspek *error steady state*, maka sistem diujikandengan menggunakan beberapa sinyal uji yaitu sinyal *step*, sinyal *ramp* dan sinyal *parabola*. Namun pada penelitian kali ini sistem hanya diuji menggunakan sinyal *step* yang berfungsi untuk menguji seberapa besar kemampuan akselerasi sistem kendali dapat menjaga posisi robot tetap pada nilai *setpoint* yang diinginkan.
3. Analisa Stabilitas
Definisi dari stabilitas, ketidakstabilan dan stabilitas marjinal sebagai berikut:
   - Sistem linear dan waktu-invarian stabil jika responnatural mendekati nol ketika waktu mendekati tak terhingga
   - Sistem linear dan waktu-invarian tidak stabil jika responnatural berkembang tanpa terikat ketika waktu mendekati tak terhingga
   - Sistem linear dan waktu-invarian marjinal stabil jika responnatural tidak berkembang tetapi tetap konstan atau berisolasi ketika waktu mendekati tak terhingga.

HASIL DAN PEMBAHASAN

Hasil Perakitan Sistem Mekanik

Berdasarkan perancangan sistem mekanik pada gambar 2 dan perakitan sistem mekanik pada gambar 7, maka bentuk dari sistem mekanik robot *self-balancing* tampak pada Gambar 11-13. Pada gambar tersebut dapat dilihat bahwa robot *self-balancing* sudah terpasang dengan sangat baik.

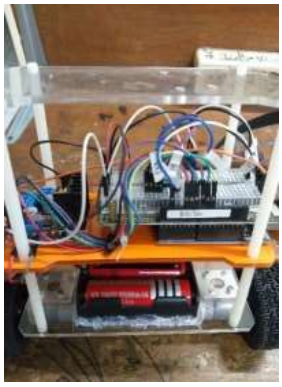


Gambar 11. Hasil Perakitan Sistem Mekanik Tampak Depan

Gambar 12. Hasil Perakitan Sistem Mekanik Tampak Samping

Gambar 13. Hasil Perakitan Sistem Mekanik Tampak Atas

Hasil Perakitan Sistem Elektrik



Gambar 14. Hasil Perakitan Sistem Elektrik Robot *Self Balancing*

Sistem elektrik robot *self-balancing* secara keseluruhan yaitu terdiri dari mikrokontroler ArduinoUno sebagai pengendali bagiseluruh komponen elektrik yang terdapat pada robot *self-balancing*, sensor MPU6050 yang berfungsi sebagai pembacaan posisi kemiringan badan robot, *motordriver* yang dapat mengendalikan dua motor DC dan motor DC sebagai komponen penggerak robot *self balancing*. Sistem elektrik yang sudah dirangkai dan dimasukan kedalam kerangka robot *self-balancing* dapat dilihat pada gambar 15.

Hasil Perakitan Sistem Kendali

Hasil perakitan sistem kendali dari robot *self balancing* adalah sebagai berikut:

1. Kode Program Robot *Self Balancing*
   Berdasarkan diagram alir algoritma kode program robot *self balancing* pada gambar 3.12, maka didapatkan kode program PID pada *software* Arduino IDE.
2. *Tuning* Pengendali PID
   Tuning (penalaan) pengendali PID pada penelitian ini menggunakan metode *trial and error*. Setelah dilakukan banyak percobaan, maka hasil yang didapat sebagai berikut:
   Adapun hasil percobaan pada lintasan dengan tinggi gelombang 5mm dapat dilihat pada tabel 2.

Tabel 2. *Tuning* PID

No.	Kp	Ki	Kd	Respon Robot
1	0	0	0	Robot tidak stabil dan tidak dapat berdiri seimbang
2	20	0	0	Robot tidak stabil dan tidak dapat berdiri seimbang
3	30	0	0	Robot tidak stabil dan tidak dapat berdiri seimbang
4	50	0	0	Kestabilan meningkat tetapi osilasi tinggi dan robot masih terjatuh
5	50	0	1	Osilasi menurun tetapi robot masih terjatuh
6	50	0	1,5	Osilasi menurun tetapi robot masih terjatuh
7	50	0	2	Osilasi menurun tetapi robot masih terjatuh

8	50	160	2	Respon robot cepat, osilasi menurun tetapi robot masih terjatuh
9	50	160	2,2	Respon robot cepat, osilasi menurun tetapi robot masih terjatuh
10	50	170	2,2	Respon robot cepat, osilasi menurun tetapi robot masih terjatuh

Berdasarkan percobaan yang telah dilakukan, dapat disimpulkan bahwa nilai konstanta PID denganKp = 50, Ki = 170 dan Kd = 2,2 yang paling optimal, karena mampu melewati lintasan dengan tinggi gelombang 5mm.

Hasil Pengujian Sistem Elektrik

a) Pengujian *Power Supply*
Hasil pengukuran menggunakan AVO Meter, besarnya nilai tegangan *powersupply* adalah 12,53V.

b) Pengujian Sensor
Setelah sensor MPU6050 berhasil dipasang pada kerangka robot, langkah selanjutnya adalah melakukan pengujian sensor dengan 2 langkah. Langkah pertama yaitu melakukan pengukuran tegangan pada tiap-tiap pin sensor MPU6050, langkah selanjutnya yaitu melakukan pengujian pembacaan nilai sudut sensor MPU6050 mengggunakan *software* Arduino IDE dengan 3 posisi robot yang berbeda, yaitu poisi jatuh ke kanan, posisi berdiri tegak dan posisi jatuh ke kiri. Adapun hasil pengujian sensor MPU6050 dapat dilihat pada tabel 3 dan 4.

Table 3. Hasil Pengukuran Sensor MPU6050

No.	Pin sensor MPU 6050	Nilai Pengukuran (V)
1	Vcc	5,17
2	SCL	2,93
3	SDA	2,93
4	INT	0,8

Tabel 4. Hasil Pengujian Nilai Sudut Sensor MPU6050

No.	Posisi Robot	Nilai Pembacaan (*)
1	Jatuh ke kanan	253
2	Berdiri tegak	174
3	Jatuh ke kiri	99

c) Pengujian *Motor Driver*
Pengujian dilakukan dengan mengukur nilai tegangan pada masing-masing pin *motor driver* L298N dengan kondisi kecepatan putaran maksimum. Adapun hasilnya dapat dilihat pada tabel 5.

Tabel 5. Hasil Pengukuran *Motor Driver*

Pin *motor driver*	Nilai Pengukuran (V)
12V	9.92
5V	4.99
EN A	4.43
IN 1	-0,16
IN 2	4.73
IN 3	-0.31
IN 4	4.79
EN B	3.3

d) Pengujian Motor DC
Pengujian dilakukan dengan mengukur tegangan serta kecepatan putaran motor DC pada kondisi kecepatan putaran maksimum.Adapun hasil pengujian motor DC dapat dilihat pada tabel 6.

Tabel 6. Hasil Pengukuran Motor DC

Motor DC	Nilai Pengukuran
Tegangan Motor 1	8.53 V
Tegangan Motor 2	8.53 V
Kecepatan Putaran Motor 1	400 Rpm
Kecepatan Putaran Motor 2	400 Rpm

Hasil Pengujian Sistem Kendali

Hasil pengujian sistem kendali pada robot *self balancing* adalah sebagai berikut:

1. Pengujian dilakukan dengan menguji robot *self balancing* berjalan di atas permukaan bergelombang dengan tinggi sebesar 2mm. Hasil yang di dapat yaitu robot mampu berjalan di atas permukaan bergelombang dengan jarak tempuh 1 meter dan dalam waktu tempuh 1 detik, maka kecepatan robot adalah 1 m/detik.
2. Pengujian dilakukan dengan menguji robot *self balancing* berjalan di atas permukaan bergelombang dengan tinggi sebesar 4mm. Hasil yang di dapat yaitu robot mampu berjalan di atas permukaan bergelombang dengan jarak tempuh 1 meter dan dalam waktu tempuh 2 detik, maka kecepatan robot adalah 0,5 m/detik.
3. Pengujian dilakukan dengan menguji robot *self balancing* berjalan di atas permukaan bergelombang dengan tinggi sebesar 5mm. Hasil yang didapat yaitu robot mampu berjalan di atas permukaan bergelombang dengan jarak tempuh 1 meter dan dalam waktu tempuh 3 detik, maka kecepatan robot adalah 0,33 m/detik.
4. Pengujian dilakukan dengan menguji robot *self balancing* berjalan di atas permukaan bergelombang dengan tinggi sebesar 6mm. Hasil yang didapat adalah robot *self balancing* tidak mampu berjalan di atas permukaan bergelombang dengan tinggi sebesar 6mm.

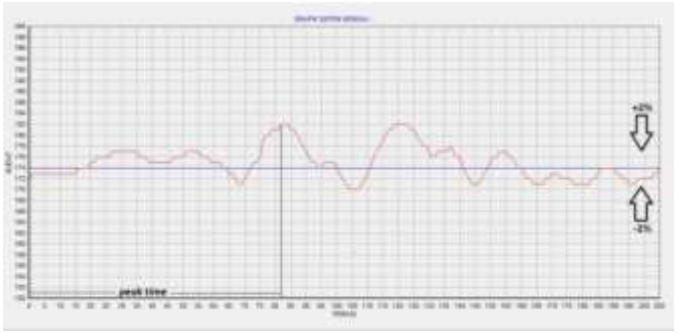
Berdasarkan pengujian di atas, respon robot *self balancing* yang paling bagus yaitu pada pengujian di atas lintasan dengan tinggi gelombang 2mm, terlihat bahwa respon robot lebih stabil dan *overshoot* lebih rendahserta kecepatan robot lebih tinggi dari percobaan yang lainnya. Selanjutnya untuk analisa karakteristik respon robot *self balancing*, penulis hanya menganalisa respon pada tinggi gelombang maksimum yang dapat dilalui robot yaitu 5mm.

Pembahasan

Setelah dilakukannya proses *tuning* PID, maka di dapatkan hasil berupa nilai konstanta proporsional (Kp), konstanta integral (Ki) dan konstanta deivatif (Kd) yang paling optimal bagi robot *self balancing* pada saat berjalan di atas permukaan yang bergelombang yaitu, Kp = 50, Ki = 2,2 dan Kd = 170 dengan grafik respon seperti terlihat pada gambar 16.

Langkah selanjutnya yaitu melakukan analisa sistem kendali robot *self balancing* berdasarkan 3 aspek analisis, antara lain:

1. Analisis Respon *Transient* (Respon Peralihan)
   a) *Time Constant*
   Time constant adalah waktu yang dicapai respon untuk mencapai 63% dari *setpoint*. Berdasarkan grafik respon pada gambar 4.19 terlihat bahwa respon dari iterasi = 1 sudah memasuki 63% dari nilai *setpoint*, hal ini dikarenakan robot diuji dari posisi tegak.



Gambar 15. Grafik Analisis Respon *Transient*

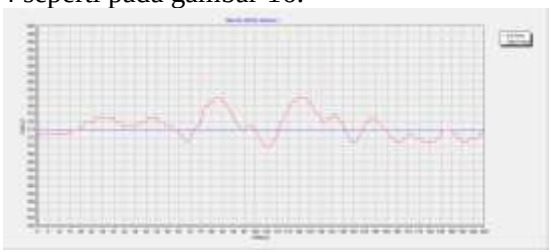
b) *Rise Time*
Rise time adalah waktu untuk respon bergerak dari 10% sampai 90% dari nilai *setpoint*. Berdasarkan grafik respon pada gambar 16 terlihat bahwa respon dari iterasi = 1 sudah memasuki 90% lebih dari nilai *setpoint*, hal ini dikarenakan robot diuji dari posisi tegak.

c) *Settling Time*
Settling time adalah waktu untuk respon mencapai dan bertahan pada jangkauan 2% dari nilai *steady state*, yaitu saat respon hanya bertahan pada nilai 170,52 sampai 177,48. Berdasarkan grafik respon pada gambar 16 terlihat bahwa respon berosilasi dan tidak bertahan pada jangkauan 2% dari nilai *steady state*, hal ini dikarenakan robot diuji pada permukaan yang bergelombang.

d) Frekuensi Natural
Berdasarkan grafik respon pada gambar 16 dapat dilihat bahwa respon robot *self balancing* tidak memiliki rasio redaman dan terus berosilasi hingga nilai iterasi = 205, sebab robot diuji pada permukaan yang bergelombang. Oleh karena itu respon dapat dikategorikan sebagai *Undamped Response*.

e) *Peak Time*
*Peak time adalah w*aktu yang diperlukan respon dari awal hingga mencapai puncak pertama *overshoot* (nilai respon maksimum). Berdasarkan grafik respon pada gambar 16 terlihat bahwa pada iterasi ke 82 respon mencapai puncak pertama *overshoot* yaitu sebesar 182. Maka nilai *peak time* nya adalah 82 x 5ms = 410ms.

2. Analisis Respon *Steady State* (Respon Keadaan Tunak)
Nilai*error steady state*adalah nilai persentasi perbedaan antara *setpoint* dengan *output*. Untuk mendapatkan nilai *error steady state* maka respon harus di uji dengan sinyal uji. Pada penelitian robot *self balancing* kali ini sinyal uji yang digunakan adalah sinyal *step* dengan cara memberi nilai *setpoint* sebesar 174 seperti pada gambar 16.



Gambar 16. Pengujian Sistem Dengan Sinyal *Step*

Berdasarkan pengujian sistem dengan sinyal *step*, maka didapatkan rata-rata *error steady state* sebesar 1,02%.

3. Analisis Stabilitas
Berdasarkan grafik respon pada gambar 16 dapat dilihat bahwa respon tidak berkembang tetapi tetap konstan atau berisolasi ketika waktu mendekati tak terhingga, maka sistem robot *self balancing* dengan nilai Kp = 50, Ki = 170 dan Kd =2,2 dapat dikategorikan stabil marginal (*marginally stable*).

PENUTUP

Kesimpulan

Kesimpulan dari penelitian ini yaitu, robot *self balancing* berhasil berjalan di atas permukaan tinggi gelombang kurang dari 6mm, menggunakan nilai parameter PID yaitu Kp = 50, Ki = 170, Kd = 2,2 dan nilai *setpoint* yang diatur sebesar 174. Karakteristik respon robot *undamped*, tidak mampu bertahan pada jangkauan ±2% dari nilai *steady state*, % *error steady state* sebesar 1,02%, dengan perilaku marjinal stabil.

Saran

Berikut ini adalah saran-saran yang dapat diberikan untuk pengembangan lebih lanjut terhadap penelitian ini :

1. Sebaiknya dalam pemilihan motor penggerak robot disarankan menggunakan motor dengan spesifikasi yang tinggi, supaya kinerja robot *self balancing* lebih maksimal.
2. Dilakukan pengembangan terhadap penelitian robot self *balancing*seperti memberikan sebuah tugas pada robot denganmenambahkan pengendali tingkat tinggi (*high level controller*).

REFERENSI

[1] Arifin, Mahardidan Harsono, Budi, 2016."*Two Wheels Self-balancing Robot* Berbasis *Arduino Nano* Menggunakan Metode PID".*Jurnal ELEKTRO*,9(2):69-80.

[2] Boby,G. dkk, 2015."Implementasi Robot Keseimbangan Roda Dua Berbasis Mikrokontroler". *Jurnal ELKOMIKA*,3(2):142-160.

[3] Dharmawan, A.dan Pramudita, S., 2015."Penerapan Sistem Kendali PID Untuk Kestabilan *Twin-Tiltrotor* dengan Metode DCM".*IJEIS*, 5(2):145-154.

[4] Firman,B., 2016. "Implementasi Sensor Imu Mpu6050 Berbasis Serial I2C Pada *Self-balancing Robot*", Universitas Syiah Kuala: Banda Aceh.

[5] Fahmizal dkk, 2011."Implementasi Sistem Navigasi *Behavior Based* dan Kontroler PID pada Manuver Robot *Maze*". Surabaya: Institut Teknologi Sepuluh Nopember.

[6] Ketaren, Lio P. dkk, 2015. "*Balancing* Robot Beroda Dua Menggunakan Metoda Kontrol Proporsional, Integral dan Derivatif". *Jurnal ELEMENTER*,1(2):39-48.

[7] Maung, M.M. dkk, 2018. "*DC Motor Angular Position Control using PID Controller with Friction Compensation*".*International Journal of Scientific and Research Publications*, 8(11):149-155.

[8] Nise, Norman S., 2011. *Control System Engineering Sixth Edition*.Pomona:California State Polytechnic University.

[9] Novandri,A. dkk,2017. "Rancang Bangun Robot *Self Balancing* Berbasis Mikrokontroler ATMega328P Dengan Kendali PID". *Jurnal Online Teknik Elektro*,2(2):15-23.

[10] Shanahan, M. dan Witkowski, M., 2000. "*High-Level Robot Control through Logic"*.London :Imperial College.

[11] Raranda dan Rusmianto, P.W.,2017. "Implementasi Kontroler PID Pada *Two Wheels Self Balancing Robot* Berbasis Arduino UNO". *Jurnal Teknik Elektro*, 6 (2):89-96.

[12] Widianto, Eko Didik, 2015. Sistem Kendali Robot. Semarang: UniversitasDiponegoro.