PENGEMBANGAN *TWO WHEELS SELF BALANCING ROBOT* DENGAN PI *CONTROLLER* BERBASIS LABVIEW 2014

Bagus Rio Rynaldo
S1 Teknik Elektro, Fakultas Teknik, Universitas Negeri Surabaya
e-mail : bagusrio44@gmail.com

Endryansyah
Dosen Teknik Elektro, Fakultas Teknik, Universitas Negeri Surabaya
e-mail : endryansyah@gmail.com

Abstrak

Two Wheels Self Balancing Robot merupakan robot beroda dua disisi kanan dan kirinya yang membutuhkan kontrol agar bisa berdiri dengan seimbang. Dari beberapa penelitian sebelumnya tentang *balancing robot* memiliki beberapa kekurangan yaitu tidak ada pemodelan sistem robot dan nilai kontroler PID pada *hardware* robot tanpa melalui simulasi. Tujuan penelitian ini adalah menghasilkan pemodelan simulai *balancing robot* untuk mencari nilai kontroler PI (*Proporsional-Integral*) dan menerapkannya pada *hardware* robot dengan cara memasukannya kedalam *sketch* Arduino IDE. Penelitian ini menggunakan *software* LabVIEW 2014 untuk simulasi dan GUI robot. *Hardware* robot menggunakan Arduino UNO R3 sebagai mikrokontroler, MPU6050 sebagai sensor, motor DC sebagai aktuator dan nRF24L01 sebagai pengirim data dari robot ke laptop. Hasil penelitian menunjukkan bahwa nilai konstanta kontroler PI dari hasil simulasi model matematika menggunakan *second metod Ziegler-Nichole* didapatkan nilai Kp = 67,5 dan Ki = 83,509. Nilai Kp dan Ki yang telah didapatkan akan dimasukkan *sketch* Arduino IDE dan hasilnya robot dapat mempertahankan posisinya tegak lurus dengan seimbang. Selain itu, robot juga mampu mengatasi kemiringan maksimal sampai 13 derajat.
Kata Kunci : *Two Wheels Self Balancing Robot*, PI *Controller*, LabVIEW

Abstract

Two Wheels Self Balancing Robot is a two-wheeled robot on the right and left that requires control in order to stand in balance. From some previous research on balancing robot has some drawbacks that there is no modeling robot system and PID controller value on robot hardware without going through simulation. The purpose of this research is to produce a robust balancing modeling model to find the value of PI (Proportional-Integral) controller and apply it to robot hardware by inserting it into Arduino IDE sketch. This research uses LabVIEW 2014 software for simulation and GUI robot. Robot hardware uses Arduino UNO R3 as a microcontroller, MPU6050 as sensor, DC motor as actuator and nRF24L01 as sender data from robot to laptop. The results showed that the value of PI control constant from simulation of mathematical model using second method Ziegler-Nichole got Kp = 67,5 and Ki = 83,509. The value of Kp and Ki that have been obtained will be inserted Arduino IDE sketch and the result of robot can maintain its position perpendicular to the balance. In addition, the robot is also able to overcome the maximum slope to 13 degrees.
Keywords : Two Wheels Self Balancing Robot, PI Controller, LabVIEW

PENDAHULUAN

Pada tahun 2011, Mochamad Mobed Bachtiar, Bima Sena Bayu D. dan A. R. Anom Besari melakukan penelitian dengan judul "Sistem Kontrol *Inverted Pendulum* Pada *Balancing Mobile Robot*". Tujuannya adalah merancang sistem untuk menggerakkan robot agar selalu seimbang saat diberikan gangguan. Dalam penelitian tersebut tidak ada pemodelan sistem dan kontroler PID langsung diterapkan pada *hardware robot.*

Kemudian pada tahun 2017, Raranda melakukan penelitian dengan judul "Implementasi Kontroler PID pada *Two Wheels Self Balancing Robot* Berbasis Arduino UNO". Tujuannya untuk menerapkan kontroler PID pada *balancing robot* agar robot berdiri seimbang. Dalam penelitian tersebut juga tidak didapatkan pemodelan matematika robot serta kontrol PID didapatkan tanpa simulasi terlebih dahulu.

Dari beberapa penelitian yang telah diuraikan tersebut dilakukan dengan cara langsung memberikan nilai kontroler PID ke *hardware* robot tanpa melalui simulasi terlebih dahulu, sehingga *tuning* PID dilakukan tanpa mempertimbangkan spesifikasi dari robot yang digunakan. Dari permasalahan tersebut maka penulis mengambil judul penelitian "Pengembangan *Two Wheels Self Balancing Robot* Dengan PI *Controller* Berbasis Labview 2014".

Tujuan penelitian ini untuk menghasilkan pemodelan matematika dari *balancing robot*. Model matematika akan digunakan untuk merancang kontroler PI. Nilai kontroler yang didapatkan akan diterapkan pada *hardware robot* melalui *sketch* Arduino IDE.

Penelitian menggunakan *software* LabVIEW 2014 untuk menghasilkan pemodelan sistem *balancing robot* berdasarkan spesifikasi dari *hardware* robot. Dari pemodelan tersebut akan diketahui respon dari robot yang kemudian digunakan untuk mencari nilai dari kontroler PI yaitu Kp (*Konstanta Proportional*) dan Ki (*Konstanta Integral*) untuk menstabilkan sudut kemiringan badan robot terhadap permukaan bidang datar.

Nilai Kp dan Ki yang telah diketahui akan diterapkan pada *hardware* robot dengan cara memasukannya kedalam *sketch* Arduino IDE. Selain itu, ketika *hardware* robot sedang melakukan *self balancing control* dapat ditampilkan dengan GUI LabVIEW dan digunakan modul *transceiver* yaitu NRF24L01 untuk mengirimkan data dari robot ke laptop yang sedang menjalankan GUI *balancing robot*. *Hardware* dari robot menggunakan Arduino UNO R3 sebagai mikrokontroler, MPU6050 sebagai sensor dan motor DC sebagai aktuator

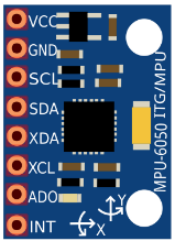
KAJIAN PUSTAKA

Balancing Robot

Balancing robot merupakan robot beroda dua disisi kanan dan kirinya yang membutuhkan kontrol agar bisa berdiri dengan seimbang. Pada *balancing robot* motor DC akan berputar searah jarum jam ataupun berlawanan dengan jarum jam ketika sasis robot condong ke arah depan atau ke arah belakang. Hal ini dilakukan untuk menjaga keseimbangan robot agar robot tidak jatuh (Ketaren, 2015).

Sensor MPU6050

Inertial Measurement Unit (IMU) adalah sensor yang digunakan untuk mengukur sudut kemiringan atau keseimbangan. IMU terdiri dari dua buah alat ukur yaitu *accelerometer dan gyroscope*. *Accelerometer* digunakan sebagai sensor posisi dan perpindahan sedangkan *gyroscope* digunakan sebagai sensor sudut/gerak rotasi. Pada *self balancing robot* digunakan modul MPU6050 yang didalamnya telah terdapat sensor *accelerometer* dan *gyroscope* (Raranda, 2017). Berikut Gambar 1. Sensor MPU-6050



Gambar 1. MPU-6050
(Sumber : InvenSense,2018)

Arduino Uno R3

Arduino Uno R3 adalah sebuah papan elektronik berupa *hardware* yang telah dilengkapi dengan *software* yang didalamnya terdapat sebuah chip mikrokontroler jenis tertentu. Mikrokontroler yang digunakan pada Arduino Uno adalah jenis Atmel seri ATmega 328 (Wardana, 2015). Untuk *supply* tegangan Arduino didapatkan dari koneksi USB dan *power supply external*. Berikut Gambar 2. Arduino UNO R3

Gambar 2. Arduino UNO R3
(Sumber : Arduino,2018)

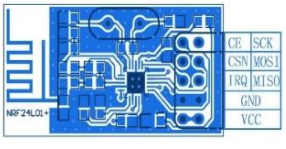
Motor Driver

Motor *driver* berfungsi sebagai pengatur arah putaran motor maupun kecepatan putaran motor. *Driver* motor diperlukan untuk *board* Arduino karena Arduino hanya mampu mengeluarkan arus yang kecil sehingga tidak mampu memenuhi kebutuhan motor DC, sehingga perlu *driver* motor untuk menyesuaikan tegangan dan arus yang dibutuhkan motor tersebut (Grace, 2015). Berikut Gambar 3. Motor Driver L298 *Module*

Gambar 3. Motor Driver L298N Module
(Sumber: STMicroelectronics,2003)

Modul Radio nRF24L01

Radio telemetri adalah suatu alat komunikasi dengan menggunakan gelombang radio untuk mengirimkan sebuah informasi dengan jarak yang jauh. nRF24L01 adalah modul radio *transceiver* (dapat digunakan sebagai *transmitter* atau *receiver*) dengan jangkauan yang luas yaitu 2.4 – 2.5 GHz yang bebas lisensi. nRF24L01 dirancang untuk menggabungkan komunikasi dengan kecepatan sangat tinggi (hingga 2 Mbit/s) dengan daya sangat rendah 60 Mw (Nordic, 2008). Berikut Gambar 4. Modul nRF24L01.



Gambar 4. Modul nRF24L01
(Sumber: Nordic, 2008)

Kontroler PI

Kontroler PI memiliki keunggulan yaitu untuk mempercepat reaksi sebuah sistem dan menghilangkan *offset*. Kontroler dengan kontrol proporsional ditambahkan dengan kontroler integral hubungannya adalah u(t) sebagai output dari kontroler dan e(t) adalah sinyal *error* (Ogata, 1985). Aksi kontrol Proporsional-Integral tersebut didefinisikan sebagai :

$$\boldsymbol{u(t) = K_p e(t) + \frac{K_p}{T_i}\int_0^t e(t)dt} \quad (1)$$

$$\boldsymbol{\frac{U(s)}{E(s)} = K_p\left(1 + \frac{1}{T_i s}\right)} \quad (2)$$

keterangan :
e(t) : sinyal error
u(t) : output dari kontroler
Kp : konstanta proporsional
Ti : waktu integral

Metode *Tuning* Ziegler-Nichols

Aturan Ziegler-Nichols digunakan untuk *tuning* nilai kontroler PID. Ada dua metode pada aturan *tuning* Ziegler-Nichols yaitu metode pertama dan metode kedua. Pada penelitian ini menggunakan metode kedua, berikut tabel aturan *tuning* Ziegler-Nichols orde 2 :

Tabel 1. Aturan *tuning* Ziegler-Nichols (Orde 2)

Tipe Kendali	Kp	Ti	Td
P	$0{,}5K_{cr}$	∞	0
PI	$0{,}45K_{cr}$	$\frac{1}{1{,}2}P_{cr}$	0
PID	$0{,}6K_{cr}$	$0{,}5P_{cr}$	$0{,}125P_{cr}$

(Sumber: Ogata, 1985)

keterangan :
Td : waktu dertivatif
Kcr : konstanta kritis
Pcr : waktu kritis

METODE PENELITIAN

Pendekatan Penelitian

Pada penelitian ini menggunakan *software* LabVIEW 2014 dan Arduino IDE 1.8.5. *Software* LabVIEW akan digunakan untuk mencari fungsi alih dari *Two Wheels Self Balancing Robot*, mendesain kontrolernya, yaitu kontroler PI (*Proporsional-Integral*) dengan *second metod* Ziegler-Nichols serta menampilkan respon sistem dari *hardware* robot yang sedang menerapkan *self balancing control*. Sedangan *software* Arduino IDE digunakan untuk memprogram *hardware* robot.

Tempat dan Waktu Penelitian

Penelitian dilaksanakan di Laboratorium Sistem Kendali dan Laboratorium Fisika Teknik Jurusan Teknik Elektro Universitas Negeri Surabaya yang dilaksanakan pada semester genap 2017/2018.

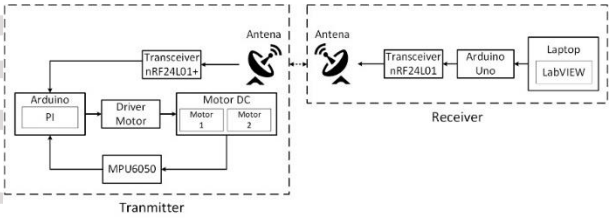
Rancangan Penelitian

Tahapan perancangan penelitian ini secara garis besar dijelaskan dalam Gambar 5.

Gambar 5. Diagram Alir Tahapan Penelitian
(Sumber : Dokumen Pribadi, 2018)

Desain Sistem

Desain sistem dari *two wheels self balancing robot* dengan kontroler PI berbasis Arduino UNO dan GUI LabVIEW ditunjukan pada Gambar 6.



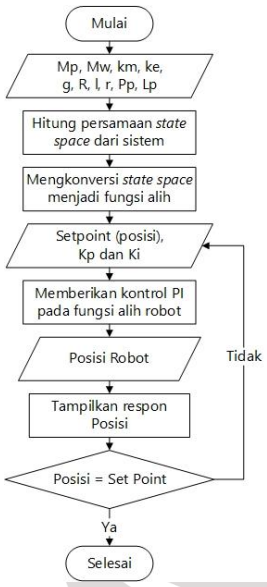
Gambar 6. Desain Sistem *Balancing Robot*
(Sumber : Dokumen Pribadi, 2018)

Gambar 6. merupakan desain sistem *two wheels self balancing robot* yang terdiri dari blok *transmitter* dan blok *receiver*. Pada blok *transmitter* terdiri dari sensor MPU6050, *transceiver*, arduino, driver motor dan Motor DC. *Transmitter* berfungsi untuk mengirimkan data dari MPU6050 yang selanjutnya akan dikirim ke *receiver*. Pada blok receiver data akan diterima kemudian akan ditampilkan dengan GUI (*Graphical User Interface*) pada *software* LabVIEW 2014. Data tersebut akan

ditampilkan secara *real time* oleh LabVIEW 2014, serta data tersebut dapat disimpan dalam bentuk file.

Rancang Bangun Software LabVIEW

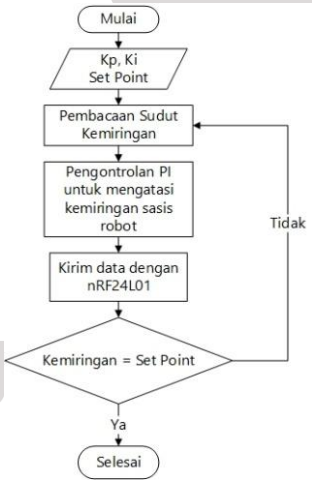
Berikut adalah desain algoritma untuk simulasi sistem robot dengan *software* LabVIEW 2014 :



Gambar 7. Diagram Alir Simulasi *Balancing Robot*
(Sumber : Dokumen Pribadi, 2018)

Rancang Bangun *Software* Arduino IDE

Berikut adalah rancangan diagram alir dari perancangan *software* Arduino IDE :



Gambar 8. Diagram alir *software* Arduino IDE
(Sumber : Dokumen Pribadi, 2018)

HASIL DAN PEMBAHASAN

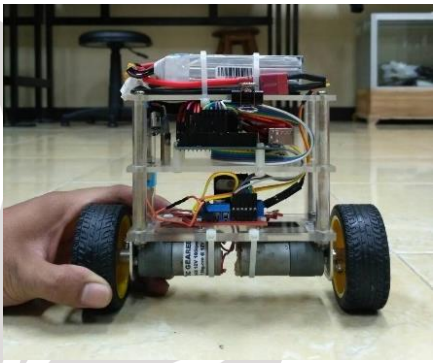
Pengujian diagram Vi pada *software* LabVIEW 2014 dilakukan dengan dua tahap yaitu adalah pengujian simulasi dari fungsi alih dari *two wheels self balancing robot* dan pengujian terhadap GUI (*Grapic User Interface*) hasil dari pembacaan sensor MPU6050 yang dikirim oleh *transmitter* pada *hardware two wheels self balancing robot* ke *receiver* yaitu Laptop.

1) Pengujian Simulasi Sistem

Pada pengujian simulasi sistem akan dicari pemodelan dari *balancing robot* kemudian dirancang kontroler PI dan terakhir melihat respon sistem dengan kontroler PI .

Pemodelan *Balancing Robot*

Untuk pemodelan robot harus diketahui spesifikasi dari robot. Berikut adalah Gambar 9. spesifikasi *balancing robot* dan Tabel 2. spesifikasi dari robot.



Gambar 9. Tampilan Robot dari Sisi Depan
(Sumber : Dokumen Pribadi, 2018)

Tabel 2. Spesifikasi robot setelah dilakukan perubahan.

No	Nama Alat	Simbol	Spesifikasi	Satuan
1.	Massa Roda	M_W	0,05	Kg
2.	Jari-jari roda	r	0,0327	m
3.	Massa sasis robot	M_p	0,6	Kg
4.	Panjang sasis	P_p	0,1244	m
5.	Lebar sasis	L_p	0,0588	m
6.	Tinggi sasis	T_p	0,0868	m
7.	Jarak pusat massa ke platfrom	l	0,0872	m
8.	Resistansi Terminal	R	97,168	Ohm

(Sumber : Dokumen Pribadi, 2018)

Setelah diketahui spesifikasi *hardware* robot maka selanjutnya mendesain pemodelan sistem berdasarkan rumus persamaan *state space* (Ooi, 2013) yaitu :

$$\begin{bmatrix}\dot{x}\\ \ddot{x}\\ \dot{\emptyset}\\ \ddot{\emptyset}\end{bmatrix}=\begin{bmatrix}0 & 1 & 0 & 0\\ 0 & \frac{2k_mk_e(M_plr-I_p-M_pl^2)}{Rr^2a} & \frac{M_p^2gl^2}{a} & 0\\ 0 & 0 & 0 & 1\\ 0 & \frac{2k_mk_e(r\beta-M_pl)}{Rr^2a} & \frac{M_pgl\beta}{a} & 0\end{bmatrix}\begin{bmatrix}x\\ \dot{x}\\ \emptyset\\ \dot{\emptyset}\end{bmatrix}+$$

$$\begin{bmatrix}0\\ \frac{2k_m(I_p+M_pl^2-M_plr)}{Rra}\\ 0\\ \frac{2k_m(M_pl-r\beta)}{Rra}\end{bmatrix}V_a \qquad (3)$$

$$y=[0\quad 0\quad 1\quad 0]\begin{bmatrix}x\\ \dot{x}\\ \emptyset\\ \dot{\emptyset}\end{bmatrix}+[0][V_a] \qquad (4)$$

keterangan :

x = posisi roda (m)

$\dot{x}$ = kecepatan roda (m/s)

$\ddot{x}$ = posisi roda (m/s^2)

$\emptyset$ = posisi sudut sasis (rad)

$\dot{\emptyset}$ = kecepatan sudut sasis (rad/s)

$\ddot{\emptyset}$ = percepatan sasis (rad/s^2)

Parameter-parameter dari *balancing robot* telah dicantumkan pada Tabel 2 dimasukkan ke persamaan 3 dan persamaan 4 melalui *Mathscript Node*, sehingga akan dihasilkan keluaran *state space model balancing robot* yaitu :

$$\begin{bmatrix}\dot{x}\\ \ddot{x}\\ \dot{\emptyset}\\ \ddot{\emptyset}\end{bmatrix}=\begin{bmatrix}0 & 1 & 0 & 0\\ 0 & -0{,}000103549 & 0{,}0487132 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0{,}000221985 & 6{,}34631 & 0\end{bmatrix}\begin{bmatrix}x\\ \dot{x}\\ \emptyset\\ \dot{\emptyset}\end{bmatrix}+$$

$$\begin{bmatrix}0\\ 0{,}000556278\\ 0\\ -0{,}00119253\end{bmatrix}V_a \qquad (5)$$

$$y=[0\quad 0\quad 1\quad 0]\begin{bmatrix}x\\ \dot{x}\\ \emptyset\\ \dot{\emptyset}\end{bmatrix}+[0][V_a] \qquad (6)$$

Persamaan 5 dan persamaan 6 tersebut kemudian dirubah menjadi *transfer function*, menggunakan rumus :

$$\frac{Y(s)}{U(s)}=C(sI-A)^{-1}B+D \qquad (7)$$

keterangan :

Y(s) = output sistem

U(s) = input sistem

A = matrik state A

B = matrik state B

C = matrik state C

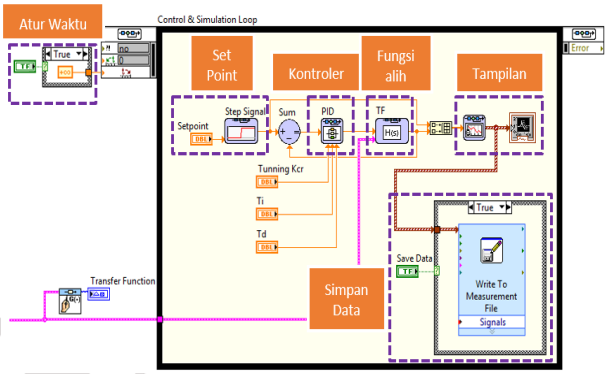
sI = matrik identitas dalam variabel s

Setelah dihitung menggunakan persamaan 7 akan didapatkan *transfer function* (TF) :

$$TF=\frac{0{,}00119253s^2}{s^4+0{,}000103549s^3-6{,}34631s^2-0{,}000667971s} \qquad (8)$$

variable s menunjukan bahwa persamaan diatas adalah transformasi laplace.

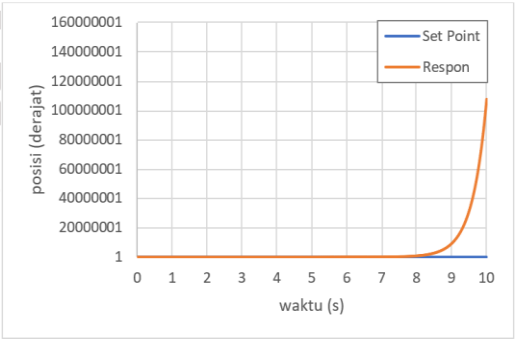
Diagram blok sistem *balancing robot* digunakan untuk melihat respon sistem dari robot. Berikut adalah blok Vi diagram untuk simulasinya :



Gambar 10. Vi Diagram Loop Tertutup dengan Kontrol PI
(Sumber : Dokumen Pribadi, 2018)

Gambar 10 adalah Vi diagram untuk simulasi dari *balancing robot*. Blok atur waktu berfungsi untuk memberikan pilihan loop program. Pada sistem ini *set point* adalah *step signal* yang dapat diubah-ubah nilainya. Untuk *tuning* digunakan blok kontroler PID dengan nilai masukannya adalah Kp, Ki dan Kd. Karena hanya menggunakan kontroler PI (proporsional-integral) maka hanya menggunakan Kp dan Ki. Kemudian fungsi alih berisi pemodelan pada persamaan 8. Kemudian untuk keluaran respon sistem akan ditampilkan menggunakan *waveform chart*. Dan data hasil simulasi sistem akan disimpan dalam bentuk file dengan blok *write to measurement file*.

Pertama dilakukan pengujian dilakukan untuk respon sistem robot sebelum diberikan kontroler.

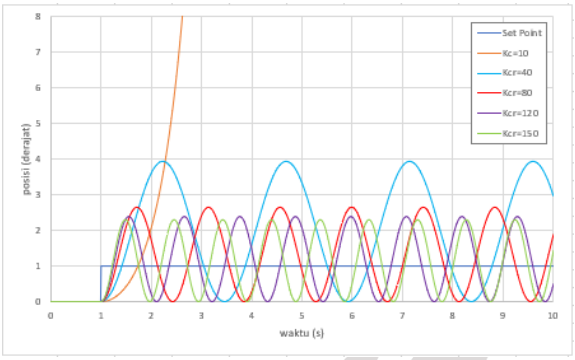


Gambar 11. Respon Sistem Robot tanpa Kontroler
(Sumber : Dokumen Pribadi, 2018)

Gambar 11. terlihat bahwa respon sistem dari *balancing robot* tidak bisa mengikuti *set point* yang diberikan. Respon sistem robot langsung menuju nilai tak hingga dan ini membuktikan bahwa *balancing robot* tidak bisa membertahankan posisinya tegak lurus terhadap permukaan bumi sehingga perlu dirancang sebuah kontroler.

Merancang Kontroler PI

Kontroler PI dirancang menggunakan *tuning second metod Ziegler-Nichols.* Pertama, mencari nilai Kcr yaitu dengan variasi nilai Kcr=40, Kcr=80, Kcr=120 dan Kcr=150. Berikut hasil *tuning* :



Gambar 12. Hasil respon sistem dengan variasi nilai Kcr
(Sumber : Dokumen Pribadi, 2018)

Data hasil *tuning* dengan nilai Kcr yang bervariasi ditampilkan pada Tabel 3.

Tabel 3. Analisis respon sistem hasil simulasi

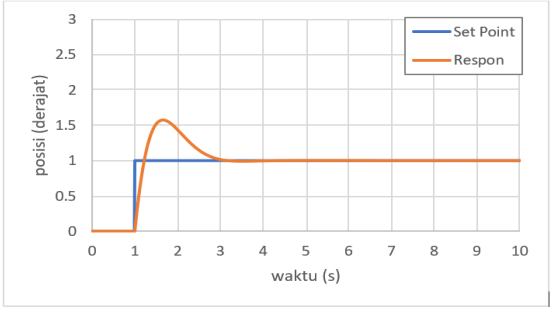
No	Kcr	Pcr	Mp	tr	tp
1.	10	-	-	-	-
2.	40	2.47	2.94284	0.42	1.23
3.	80	1.43	1.653782	0.3	0.71
4.	120	1.11	1.393002	0.25	0.55
5.	150	0.97	1.302191	0.23	0.48

(Sumber : Dokumen Pribadi, 2018)

Berdasarkan Tabel 3 nilai Kcr yang paling sesuai adalah ketika respon sistem memiliki nilai *Maximum peak (Mp)* yang paling kecil dan *rise time* (tr) dan *peak time* (tp) yang paling cepat. Dari beberapa variasi *tuning* nilai Kcr maka dapat disimpulkan bahwa nilai Kcr yang paling sesuai adalah 150. Kemudian dihitung nilai Kontroler PI berdasarkan tabel 1 *aturan* tuning *Ziegler-Nichols (Orde 2)* yaitu :

$Kp = 67{,}5$
$Ti = 0{,}8083$
$Ki = 83{,}509$

Setelah disimulasikan akan didapatkan respon keluaran sistem seperti Gambar 13.

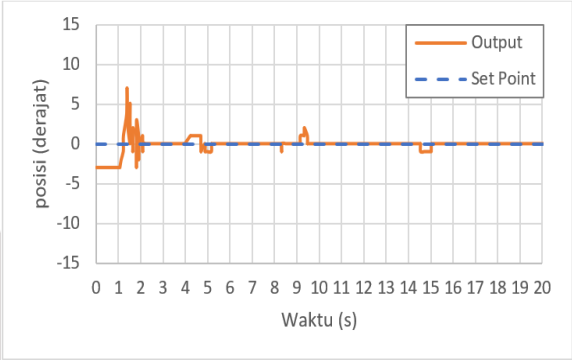


Gambar 13. Respon Sistem dengan Kontroler PI
(Sumber : Dokumen Pribadi, 2018)

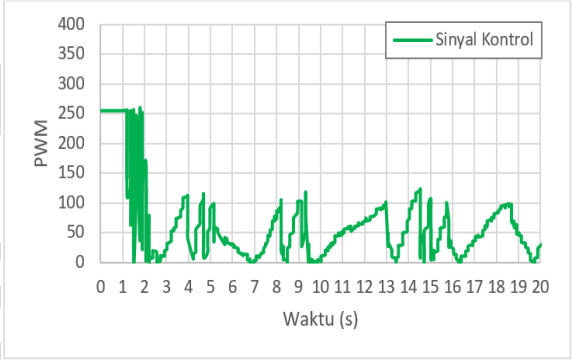
Pada Gambar 13. terlihat sistem lebih stabil dengan osilasi yang kecil. Respon sistem sesuai dengan karakteristik dari kontroler PI (*proportional-integral*) yaitu sistem memiliki respon atau tanggapan yang cepat dan menghilangkan respon *steady state error*.

2) **Pengujian *Hardware Robot***

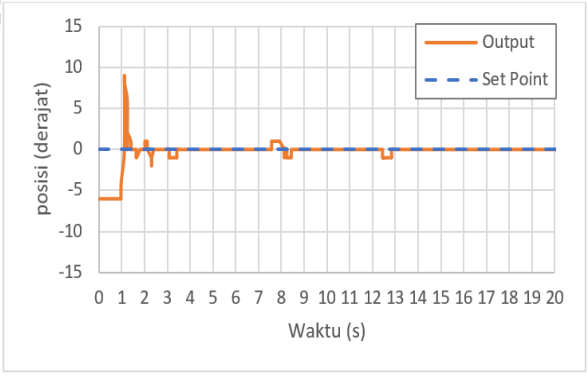
Pada pengujian *hardware robot* dilakukan dengan cara memberikan robot posisi awal kemudian dilihat apakah robot bisa mengatasi kemiringan tersebut dan tetap berdiri dengan seimbang. Kondisi awal (KA) meliputi -3 derajat, -6 derajat, -9 derajat, -13 derajat, -16 derajat, 3 derajat, 6 derajat, 9 derajat, 13 derajat dan 16 derajat. Berikut adalah grafik respon dari *hardware robot* :



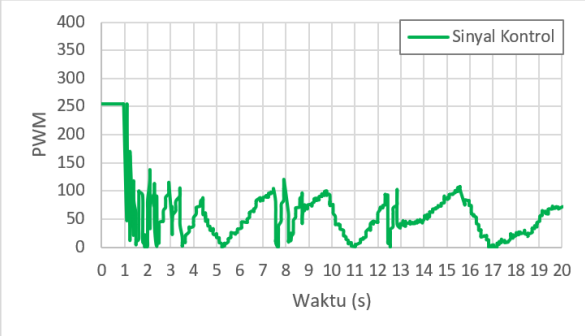
Gambar 14. Respon Posisi pada -3 derajat
(Sumber : Dokumen Pribadi, 2018)



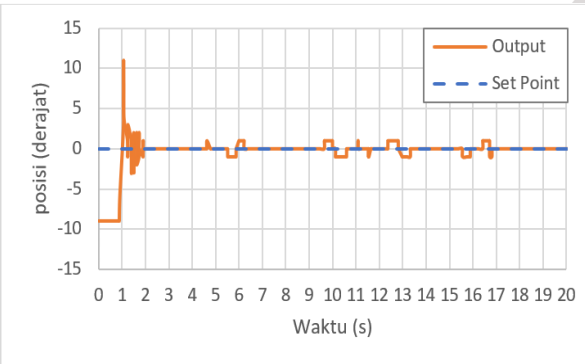
Gambar 15. Sinyal Kontrol pada -3 derajat
(Sumber : Dokumen Pribadi, 2018)



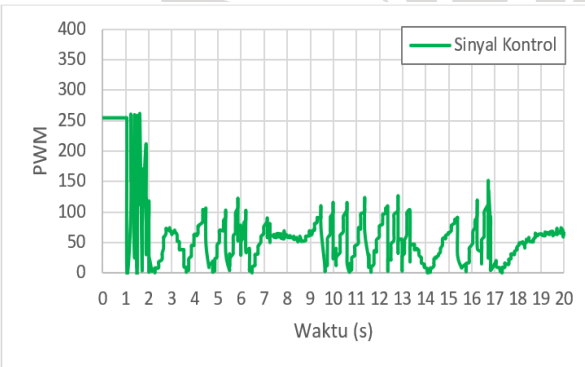
Gambar 16. Respon Posisi pada -6 derajat
(Sumber : Dokumen Pribadi, 2018)



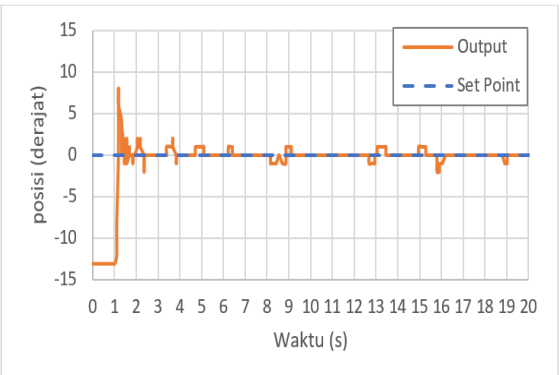
Gambar 17. Sinyal Kontrol pada -6 derajat
(Sumber : Dokumen Pribadi, 2018)



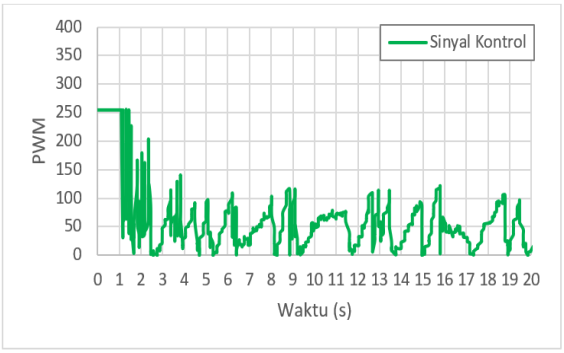
Gambar 18. Respon Posisi -9 derajat
(Sumber : Dokumen Pribadi, 2018)



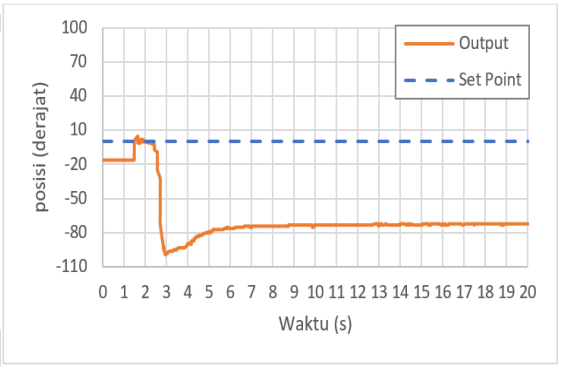
Gambar 19. Sinyal Kontrol pada -9 derajat
(Sumber : Dokumen Pribadi, 2018)



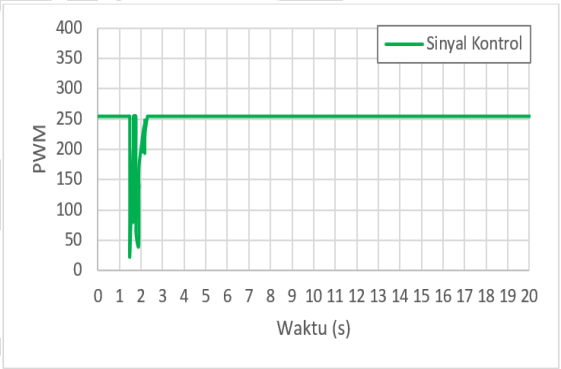
Gambar 20. Respon Posisi pada -13 derajat
(Sumber : Dokumen Pribadi, 2018)



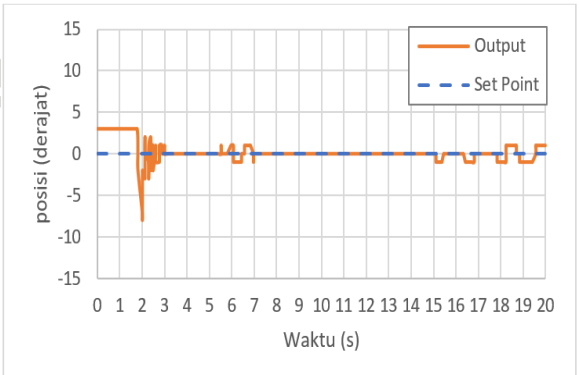
Gambar 21. Sinyal Kontrol pada -13 derajat
(Sumber : Dokumen Pribadi, 2018)



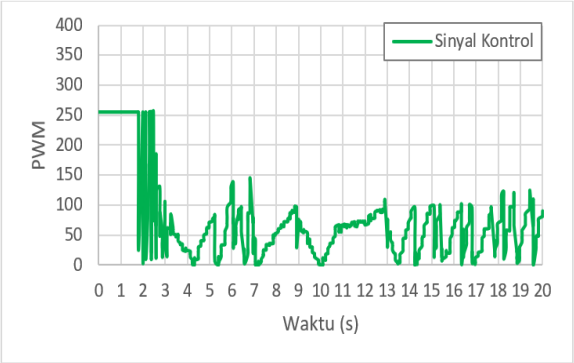
Gambar 22. Respon Posisi pada -16 derajat
(Sumber : Dokumen Pribadi, 2018)



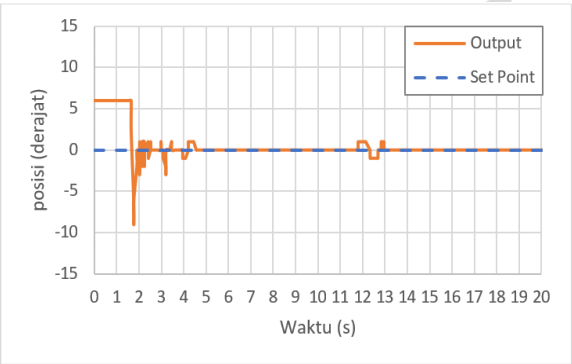
Gambar 23. Sinyal Kontrol pada -16 derajat
(Sumber : Dokumen Pribadi, 2018)



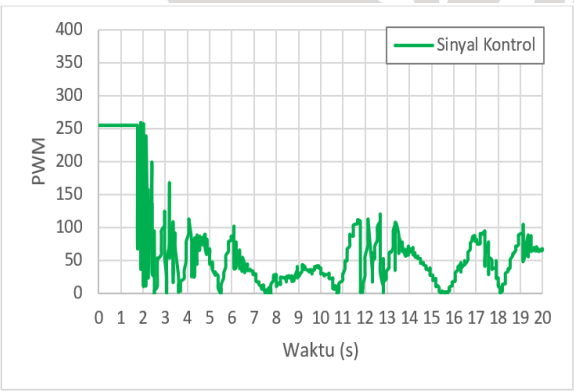
Gambar 24. Respon Posisi pada 3 derajat
(Sumber : Dokumen Pribadi, 2018)



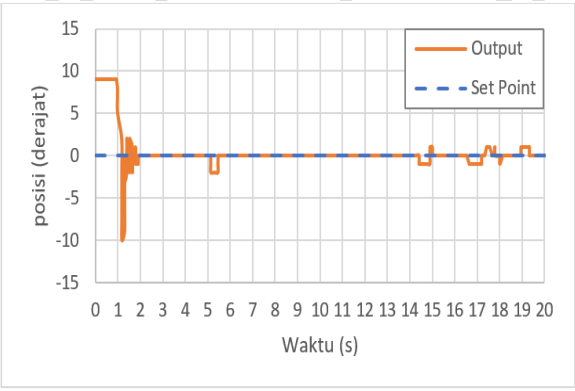
Gambar 25. Sinyal Kontrol pada 3 derajat
(Sumber : Dokumen Pribadi, 2018)



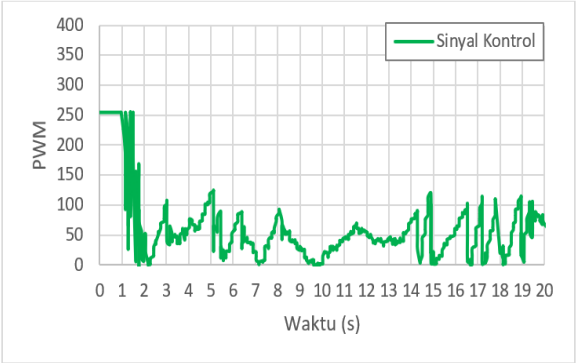
Gambar 26. Respon Posisi pada 6 derajat
(Sumber : Dokumen Pribadi, 2018)



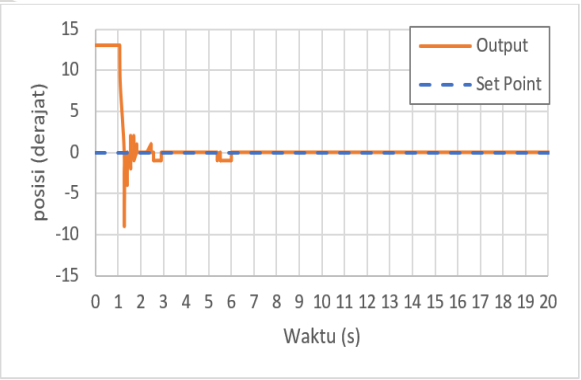
Gambar 27. Sinyal Kontrol pada 6 derajat
(Sumber : Dokumen Pribadi, 2018)



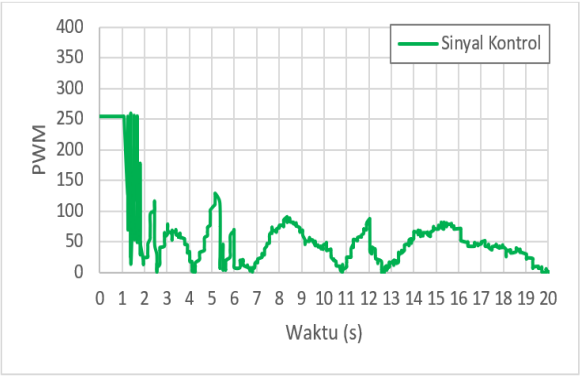
Gambar 28. Respon Posisi pada 9 derajat
(Sumber : Dokumen Pribadi, 2018)



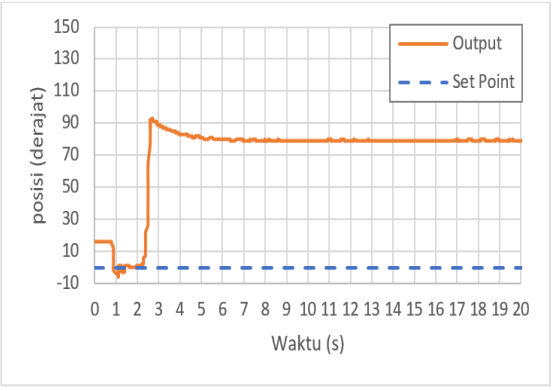
Gambar 29. Sinyal Kontrol pada 9 derajat
(Sumber : Dokumen Pribadi, 2018)



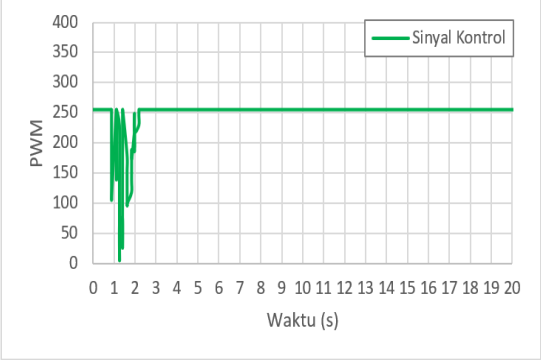
Gambar 30. Respon Posisi pada 13 derajat
(Sumber : Dokumen Pribadi, 2018)



Gambar 31. Sinyal Kontrol pada 13 derajat
(Sumber : Dokumen Pribadi, 2018)



Gambar 32. Respon Posisi pada 16 derajat
(Sumber : Dokumen Pribadi, 2018)



Gambar 33. Sinyal Kontrol pada 16 derajat
(Sumber : Dokumen Pribadi, 2018)

Pada Gambar 14, Gambar 16, Gambar 18, Gambar 20, Gambar 22, Gambar 24, Gambar 26, Gambar 28, Gambar 30 dan Gambar 32 adalah respon posisi robot ketika diberikan gangguan. Pada gambar tersebut terlihat robot dapat mempertahankan posisinya namun robot terjatuh ketika diberikan kondisi awal 16 derajat dan -16 derajat.

Sedangkan pada Gambar 15, Gambar 17, Gambar 19, Gambar 21, Gambar 23, Gambar 25, Gambar 27, Gambar 29, Gambar 31 dan Gambar 33 adalah *output* pengendalian PID untuk mengendalikan sudut kemiringan sasis robot. Terlihat *output* pengendalian PID menghasilkan keluaran yang tidak beraturan dengan rentang waktu naik dan turun yang cepat karena pengendalian sudut kemiringan sasis robot harus agresif agar robot dapat berdiri tegak dengan seimbang.

Untuk lebih lengkapnya hasil pengujian *hardware robot* ketika mengatasi kemiringan sasis akan dijelaskan pada Tabel 4.

Tabel 4. Hasil pengujian *hardware balancing robot*

Initial Condition	*Mp*	*Peak Time (second)*	*Time Respon (second)*	*Error Steady State*
-3	7	0.321	0.851	0.095
-6	9	0.151	0.432	0.099
-9	11	0.184	0.852	0.190
-13	8	0.168	0.544	0.205
-16	5	0.163	-	-
3	-8	0.231	0.82	0.218
6	-9	0.117	0.596	0.214
9	-10	0.247	0.718	0.130
13	-9	0.2	0.618	0.064
16	-6	0.329	-	-

(Sumber : Dokumen Pribadi, 2018)

Pada penelitian sebelumnya (Raranda,2017), didapatkan nilai *error* sistem sebesar 0,11043 dan nilai *time response* sebesar 1,47 detik untuk gangguan pertama dan gangguan yang *time response* sebesar 1,48 detik. Penelitian ini menggunakan kontroler PID dengan nilai Kp=70, Ki=466,67 dan Kd=2,625.

Berdasarkan Tabel 4 diketahui bahwa *error steady state* paling kecil ketika kondisi awal sama dengan 13 derajat yaitu sebesar 0,064 dan didapatkan nilai *time response* yang paling kecil ketika kondisi awal sama dengan -6 derajat yaitu sebesar 0,432 detik. Penelitian ini dilakukan dengan kontroler PI dengan nilai Kp=67,5 dan Ki=83,509 yang didapatkan dari *metode Ziegler-Nichole* orde 2 berdasarkan Tabel 3. Sehingga secara garis besar kontroler PI lebih baik dalam *time response* sistem yang lebih cepat dari kontroler PID yaitu 0,432 detik dan kontroler PI memiliki *error steady state* yang lebih kecil dari kontroler PID yaitu 0,064 yang didapatkan pada Tabel 4 hasil pengujian *hardware*.

PENUTUP

Simpulan

Berdasarkan analisis dari hasil simulasi *self balancing robot* dan pengujian pada *hardware self balancing robot* yang telah dilakukan maka dapat diambil kesimpulan bahwa dalam penelitian ini didapatkan pemodelan matematika dari *balancing robot* sesuai spesifikasi robot yang digunakan yaitu pada persamaan 8. Model matematika tersebut digunakan untuk mendesain kontroler PI dengan *second metod Ziegler-Nichols* sehingga didapatkan nilai Kp=67,5 dan Ki=83,509.

Pada pengujian *hardware two wheels self balancing robot*, nilai Kp dan Ki yang didapatkan dari simulasi akan diterapkan pada *hardware balancing robot* dengan memasukannya ke *script* Arduino IDE. Dari hasil uji *hardware robot* bisa mengatasi kemiringan 3 derajat, -3 derajat , 6 derajat, -6 derajat, 9 derajat, -9 derajat, 13 derajat dan -13 derajat. Namun robot jatuh ketika kemiringan 16 derajat dan -16 derajat.

Saran

Berdasarkan penelitian yang telah dilakukan, ada beberapa saran yang dapat dilakukan untuk pengembangan sistem *two wheels balancing robot* selanjutnya yaitu sebagai Pada penelitian ini simulasi sistem *balancing robot* dilakukan dengan *tuning second metod* Ziegler-Nichols, untuk selanjutnya bisa menggunakan metode *tuning* PID lainnya misalnya metode Cohen-Coon atau Direct Synthesis. Kemudian dapat menggunakan kontroler PID karena pada penelitian ini *overshoot* dari robot cukup tinggi dan sebaiknya diberikan konstanta derivatif untuk mengatasi hal tersebut. GUI Labview hanya bisa digunakan untuk menampilkan data secara *real time*, maka untuk penelitian selanjutnya bisa menggunakan GUI untuk mengontrol gerakan robot seperti gerakan maju atau gerakan mundur.

DAFTAR PUSTAKA

Arduino.2018.”Tech Specs Arduino UNO R3”.(Online), (https://store.arduino.cc/arduino-uno-rev3, diunduh 29 April 2018).

Bachtiar. M. M. dkk. 2011. “Sistem Kontrol Inverter Pendulum pada Balancing Mobile Robot”. Makalah disajikan dalam *The 3th Industrial Electronics Seminar 2011*, Surabaya, 26 Oktober.

Bobby, Grace, Susanto, E., & Suratman, F.Y. (2015). “Implementasi Robot Keseimbangan Beroda Dua Berbasis Mikrokontroler”. *Jurnal Elkomika*. Vol. *3*(2): hal 142-160.

InvenSense. 2018. “Datasheet MPU-6050”. (Online),(https://www.invensense.com/products/motion-tracking/6-axis/mpu-6050/#product-documentation, diunduh 29 April 2018).

Ketaren, Lio Prisko. 2015. Balancing Robot Beroda Dua Menggunakan Metode Kontrol Proporsional, Integral dan Derivatif. *Jurnal Politeknik Caltex Riau*. Vol 1(2): hal 39-48

Nordic Semiconductor. 2008. “Datasheet. nRF24L01+ Single Chip 2.4 GHz Transceiver Product Specification v1.0”. (Online), (http://www.nordicsemi.com/eng/nordic/content_download/2726/34069/file/nRF24L01P_Product_Specification_1_0.pdf, diunduh 13 Februari 2018).

Ogata, Katsuhiko. 1985. *Teknik Kontrol Automatik jilid 1*. Terjemahan Edi Laksono. Jakarta: Erlangga.

Ooi, Rhi Chi. 2013. *Balancing a two-wheeled Autonomous Robot*. Australia: The University of Western Australia School of Mechanical Engineering.

Raranda. 2017. “Implementasi Kontroler PID Pada Two Wheels Self Balancing Robot Berbasis Arduino UNO”. *Jurnal Teknik Elektro*. Vol. 06: hal. 89-96.

STMicroelectronics.2003.”Dual Full-Bridge Driver”.(Online),(https://www.alldatasheet.com/datasheet_pdf/pdf/22440/STMICROELECTRONICS/L298N.html, diunduh 28 Mei 2018).

Wardana, I Nyoman Kusuma. 2015. *Teknik Antarmuka MATLAB dan Arduino*. Denpasar: Vaikutha International Publication.