Jurnal Teknik Elektro, Vol. 17, No. 1, Maret 2024: 16-21
P-ISSN 1411-870X, E-ISSN 2656-3304

SISTEM KONTROL KESEIMBANGAN ROBOT BERODA SATU

Hans Junian, Thiang
Program Studi Teknik Elektro, Universitas Kristen Petra, Surabaya

Email: hansjnian179698@gmail.com, email korespondensi: thiang@petra.ac.id

Abstrak – Makalah ini menjelaskan tentang *self balancing robot* dimana robot tersebut hanya mempunyai satu roda. Robot beroda satu ini dapat menyeimbangkan dirinya sendiri. Dalam prototipe ini, robot yang dibuat dalam bentuk miniatur. *Hardware* robot ini terdiri atas baterai Lipo 3S 1500Mah, regulator lm 2596, sensor IMU MPU 6050, kontroler Arduino Uno, driver L298N untuk menggerakkan motor DC. Metode kontrol yang digunakan untuk menyeimbangkan robot adalah kontroler PID. Sensor accelerometer dan gyroscope yang terdapat dalam IMU digunakan untuk membaca posisi dan kemiringan robot yang kemudian dikirimkan ke kontroler arduino melalui komunikasi serial I2C. Pengujian sistem telah dilakukan dengan mencari nilai kombinasi PID yang terbaik, initial position maksimal yang dapat ditempuh oleh robot, serta sudut kemiringan maksimal yang dapat ditempuh oleh robot. Nilai kombinasi PID yang baik adalah Kp sebesar 30, Ki sebesar 200 dan Kd sebesar 1. Sedangkan untuk initial position maksimal yang dapat ditempuh oleh robot berada di range 175° hingga 185°. Dan untuk sudut kemiringan maksimal yang dapat ditempuh robot adalah sebesar 5°. Secara umum, sistem yang telah dibuat berhasil untuk menyeimbangkan robot berroda satu.

Kata Kunci – *Self balancing*, robot berroda satu, PID, arduino uno, IMU MPU 6050

I. PENDAHULUAN

Perkembangan teknologi pada dunia robotika telah membuat standar kualitas kehidupan manusia semakin tinggi. Hal ini disebabkan oleh perkembangan dunia robotika yang sangat pesat dari tahun ke tahun. Pada zaman modern ini, perkembangan teknologi pada dunia robotika juga dinilai mampu meningkatkan kualitas dan kuantitas berbagai industri. Teknologi robotika ini mampu menjangkau sisi hiburan maupun pendidikan bagi manusia. Perkembangan teknologi robotika ini membuat penelitian mengenai robot sangatlah banyak dan berkembang pesat. Cara – cara yang dapat dilakukan guna meningkatkan tingkat kecerdasan sebuah robot adalah dengan menambahkan sensor, metode kontrol, maupun memberikan sebuah kecerdasan buatan pada robot tersebut. Salah satu robot yang dimaksud adalah self balancing robot.

Self balancing robot pertama kali diperkenalkan oleh peneliti bernama Dean Kamen pada tahun 2001. Dean Kamen menamai sistem ini dengan nama *segway*. *Segway* ini kemudian dikenal oleh banyak orang sebagai *the first self balancing, electric powered transportation device. Self balancing robot* ini merupakan pengembangan dari model pendulum terbalik yang diletakkan di atas kereta beroda. Menyeimbangkan robot memerlukan sebuah perangkaian *hardware* yang sudah direncanakan, serta metode kontrol yang handal guna mempertahankan posisi robot tetap dalam keadaan tegak lurus terhadap permukaan bumi[1]. *Self balancing robot* ini biasanya diterapkan dalam *unicycle* dan *segway.*

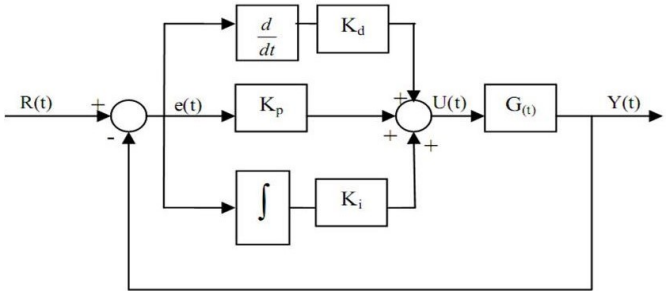
Dari segi penelitian, dapat diketahui bahwa jumlah roda pada mobile robot semakin berkurang seiring penelitian yang terus berkembang. Salah satu contohnya adalah one wheeled electric vehicle yang mengintegrasikan prinsip sistem self balancing robot yang didesain sebagai alat transportasi manusia. Prinsip mekanik dari self balancing ini berdasarkan prinsip dari pendulum terbalik [2]. Sebelumnya, muncul juga two wheeled self balancing robot yang dirakit oleh Yamamoto pada tahun 2009 [3], ballbot oleh Fong pada tahun 2002 [4], Micycle yang diteliti oleh Kadis dan Caldecott pada tahun 2010 [5], one wheeled vehicle yang diteliti oleh Shao Zhiyu, dan Liu Daliang pada tahun 2010 [6]. Banyak penelitian yang didasarkan pada kalman filter untuk menelusuri respon dari sinyal output sensor dari gabungan accelerometer dan gyroscope. Penelitian milik Yamamoto masih menggunakan dua roda dan dikontrol dengan metode LQR. Sedangkan milik Fong, roda yang digunakan adalah bola berbentuk bulat dan dikontrol dengan metode LQR. Untuk Micycle milik Kadis dan Caldecott, algoritma yang digunakan untuk mengontrol keseimbangan robot adalah kontrol PD. Sementara milik Shao Zhiyu menggunakan metode kontrol PI yang dikombinasikan dengan metode LQR. Pada paper ini akan dibuat prototype robot beroda satu yang dikontrol dengan menggunakan metode PID berbasis Arduino yang dilengkapi dengan mode DMP.

II. DESAIN SISTEM

A. Desain Software Sistem

Perancangan dan pembuatan alat dalam penelitian ini terbagi menjadi dua bagian, yaitu bagian *software* dan *hardware*. Bagian *software* terdiri dari PID dan program *balancing*.

1. Kontroler PID



Gambar 1. Diagram Blok Kontroler PID

Kontrol PID adalah kontroler yang digunakan untuk menentukan presisi suatu sistem instrumentasi elektronika dengan adanya umpan balik pada sebuah sistem yang disebut feedback. Sistem kontrol ini terdiri dari tiga buah kontrol, yaitu kontrol proportional, integral, dan derivative. Masing – masing kontrol tentunya memiliki kelebihan dan kekurangannya masing – masing. Dalam implementasinya, kontrol – kontrol ini dapat bekerja sendiri ataupun gabungan diantaranya. Dalam perancangan sistem PID, perlu diatur parameter P, I, dan D agar tanggapan atau output sinyal sesuai dengan yang diharapkan. Gambar 1 menunjukkan gambar diagram blok dari kontroler PID.
Karena algoritma kontroler PID diterapkan di mikrokontroler maka berikut adalah persamaan yang digunakan untuk perhitungan *output* kontroler PID:

$$Error = setpoint - input \quad (1)$$

$$cumulative\_err = error * elapsedtime + cumulative\_err \quad (2)$$

$$rateerror = \frac{(error - lasterror)}{elapsedtime} \quad (3)$$

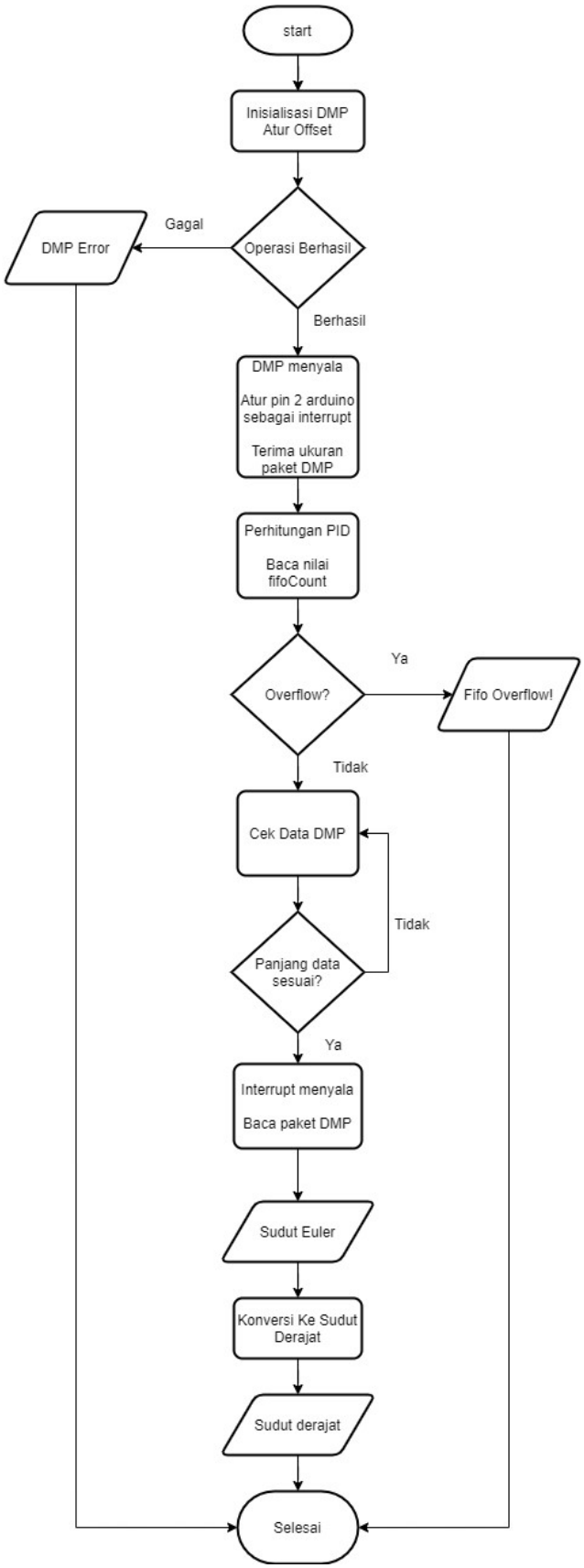
$$Out = (Kp * error) + (Ki * cumulative\_err) + (Kd * rateerror) \quad (4)$$

dimana:
error = selisih antara *set point* dengan nilai sekarang
cumulative_err = akumulasi dari seluruh *error* dikalikan dengan waktu interval
rateerror = besar *error* sekarang dikurangi dengan *error* sebelumnya lalu dibagi dengan waktu interval

Berdasarkan persamaan 1 hingga 3, bisa dilihat bahwa kontroler P berkaitan erat dengan *error* sekarang. Sedangkan kontroler I berkaitan erat dengan *error-error* sebelumnya yang telah diakumulasi. Dan yang terakhir, kontroler D berkaitan erat dengan selisih *error* sebelumnya terhadap rentang waktu tertentu. Kontroler I masih mempertimbangkan sejarah error yang terjadi. Integrasi sendiri adalah penjumlahan yang kontinyu. Integrasi *error* dari waktu ke waktu sama dengan meringkas riwayat *error* dalam rentang waktu tertentu.

2. Program *Balancing*

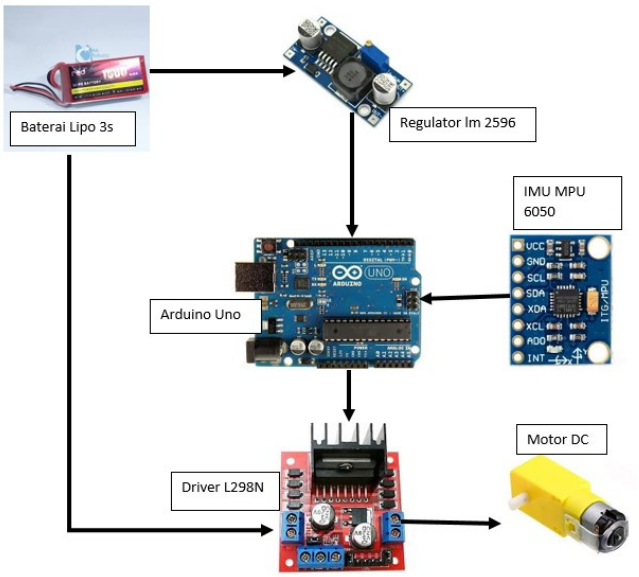
Selain PID, bagian *software* pada penelitian ini terletak pada program *balancing*. *Flowchart* program *balancing* dapat dilihat pada gambar 2.
Pada gambar 2, bisa dilihat bahwa *coding* sistem diawali dengan inisialisasi DMP dan pengaturan *offset*. *Offset* setiap MPU tentu berbeda-beda sehingga MPU wajib untuk dikalibrasikan terlebih dahulu. Setelah selesai menentukan *offset*, akan dilanjutkan ke pemioihan, apabila pengoperasian *device* berhasil, maka DMP akan dinyalakan. Namun, jika pengoperasian *device* gagal, maka serial monitor akan menampilkan pesan DMP *error*. Pada dasarnya ketika data DMP sudah siap dikirimkan, maka *interrupt* akan menyala. Setelah DMP diaktifkan, akan dilanjutkan dengan penerimaan ukuran dari paket DMP. Sambil menunggu *interrupt*, akan dilakukan perhitungan PID menggunakan persamaan 1 sampai 4.
Sesuai dengan *flowchart* pada gambar 2, setelah dilakukan perhitungan PID, akan dilanjutkan dengan pembacaan nilai *fifocount*. Saat pembacaan nilai *fifocount*, akan dilakukan pengecekan apakah terjadi *overflow* pada FIFO atau tidak. Apabila terjadi overflow, maka *serial monitor* akan menuliskan FIFO *overflow*. Jika tidak terjadi *overflow*, akan dilanjutkan ke pengecekan data DMP. Setelah itu, tunggu hingga panjang data sudah sesuai. Setelah panjang data sesuai, akan dilakukan *interrupt* pada data DMP. Setelah *interrupt* menyala, paket DMP akan dibaca. Paket DMP ini berisi sudut *euler*, yang nantinya sudut *euler* ini akan dikonversikan menjadi sudut dalam derajat. Setelah selesai dikonversi menjadi sudut dalam derajat, sudut inilah yang akan menjadi nilai untuk variabel *input*.



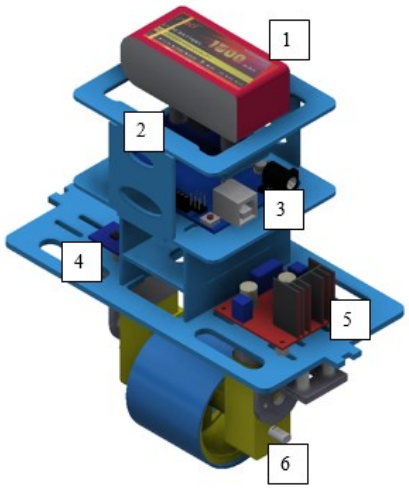
Gambar 2. *Flochart* Program *Balancing*

B. Desain Hardware Sistem

Desain *hardware* dimulai dengan perancangan diagram blok sistem secara keseluruhan. Diagram blok sistem dapat dilihat pada gambar 3.



Gambar 3. Diagram Blok *Hardware* Sistem



Gambar 4. Mekanik Robot dan Peletakkan Komponen Robot

Keterangan gambar 4:
1) Baterai Lipo 3s 11.1V
2) Regulator lm2596
3) Arduino Uno
4) IMU MPU 6050
5) Driver L298N
6) Motor DC dengan *gearbox*

Pada gambar 3, dapat dilihat beberapa komponen penitng dari sistem ini meliputi baterai lipo 3s, regulator lm 2596, Arduino uno, IMU MPU 6050, *driver* L298N, dan motor DC dengan *gearbox*. Komponen di atas dipilih bukan tanpa alasan. Baterai Lipo 3s dipilih karena baterai Lipo 3s memiliki tegangan ± 12V. Regulator lm 2596 dipilih untuk menurunkan tegangan keluar baterai Lipo dari 12v ke ± 7V. Angka 7V dipilih karena sesuai dengan tegangan masuk untuk Arduino *uno*. Apabila melebihi dari tegangan ini, Arduino tidak dapat berfungsi. Arduino *uno* digunakan untuk melakukan *coding* mengenai sistem kontrol dimana kontrol sistem merupakan unsur penting dari laporan ini. Komponen berikutnya adalah *driver* L298N. *Driver* ini mendapatkan tegangan masuk langsung dari baterai Lipo yaitu sebesar ± 12V. *Driver* ini nantinya akan terhubung ke motor DC yang merupakan aktuator dari prototipe robot ini.

Perancangan *hardware* terdiri dari dua bagian, yaitu perancangan dari segi mekanik dan perancangan dari segi elektronik. Perancangan mekanik difokuskan untuk tata letak pemasangan sensor dan bentuk fisik robot secara umum.

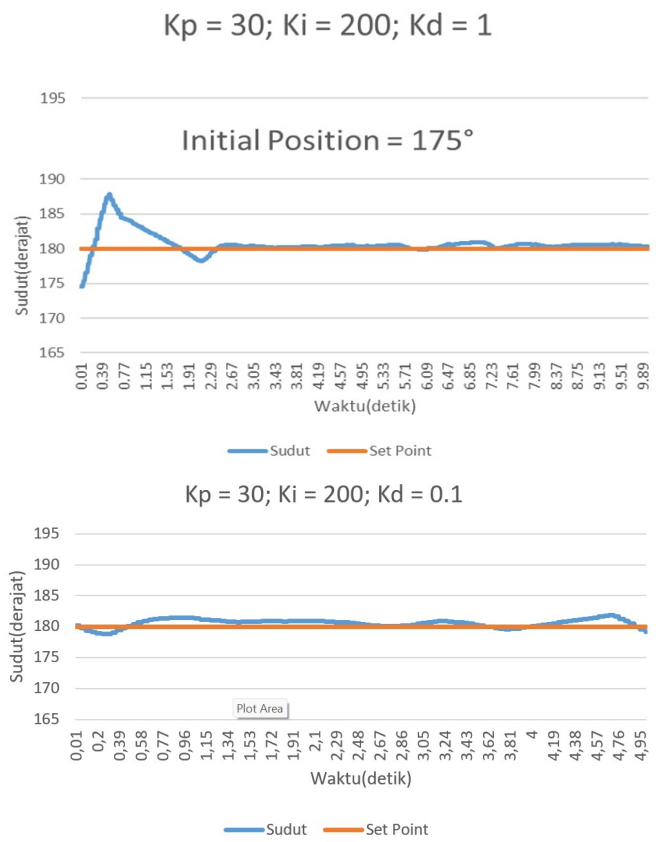
Perancangan elektronik difokuskan pada perancangan rangkaian modul-modul yang digunakan. Seluruh rangkaian dirancang dalam bentuk rangkaian yang terletak atau disusun di atas PCB. Gambar 4 menunjukkan gambar mekanik robot dan peletakkan komponen-komponen robot.

III. HASIL PENGUJIAN SISTEM

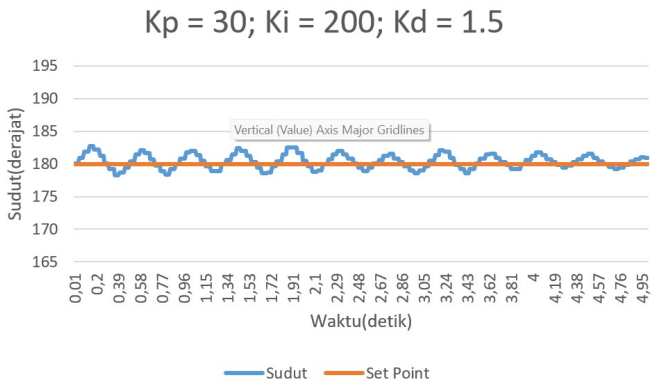
Beberapa pengujian sistem telah dilakukan untuk melihat kenadalan sistem yang telah didesain. Pengujian tersebut adalah pengujian untuk mencari nilai parameter kontroler PID yang terbaik, pengujian dengan beberapa *initial position* dan pengujian pada bidang miring.

A. Pengujian dengan Variasi Nilai PID Terbaik

Pada pengujian ini, akan dicari besar nilai konstanta proportional, integral dan derivatif terbaik. Pengujian dilakukan dengan memodifikasi besar nilai konstanta proporsional, integral, dan derivatif. Modifikasi dilakukan secara bertahap dimulai dengan menemukan konstanta proporsional terbaik, diikuti oleh konstanta integral dan derivatif terbaik. Pengujian ini dilakukan dengan metode *trial and error*. Karena pengujian baru memasuki tahap pertama, maka robot akan diuji dengan posisi awal sudah dalam keadaan tegak. *Set point* dalam pengujian ini adalah 180°. Sumbu X sebagai waktu dengan satuan detik (s), dan sumbu Y mewakili besaran nilai sudut. Parameter yang digunakan adalah apakah robot dapat tetap stabil di *range* angka 179°-181°, dengan *error* yang sekecil mungkin. Parameter ini digunakan karena posisi awal robot sudah berada pada *set point.* Parameter yang lainnya akan digunakan saat pengujian *initial position*. Berikut adalah beberapa hasil pengujian yang telah dilakukan, yang dapat dilihat pada gambar 5, 6 dan 7. Dari seluruh hasil pengujian yang telah dilakukan dapat disimpulkan bahwa untuk sistem ini, konstanta proportional, integral dan derivatif terbaik yang didapatkan adalah Kp = 30, Ki = 200, Kd = 1 dimana grafik respon sistemnya dapat dilihat pada gambar 5. Pada gambar 5, bisa dilihat pada grafik bahwa robot berjalan sangat dekat dengan *set point* dengan durasi yang dibilang cukup lama.



Gambar 6. Respon Sistem dengan Kp=30, Ki=200, Kd=0,1



Gambar 7. Respon Sistem dengan Kp=30, Ki=200, Kd=1,5

B. Pengujian Variasi Initial Position

Setelah memperoleh nilai Kp, Ki, Kd terbaik dalam percobaan sebelumnya, akan dilanjutkan dengan pengujian berikutnya, yaitu pengujian *initial position*. Pada pengujian ini, akan diuji beberapa sudut awal robot ketika sistem kontrol dinyalakan. Pada pengujian ini, akan dicari *initial position* maksimal yang dapat dijalankan oleh robot. Parameter pengujian ini akan dilihat pada *rise time* dan *settling time*. *Rise time* adalah waktu yang diperlukan untuk mencapai *set point* atau *steady state* pertama kalinya. Sedangkan *settling time* adalah waktu yang diperlukan untuk mencapai keseimbangan. Dalam pengujian ini akan dilakukan percobaan *initial position* dimulai dari 175° hingga 185° (±5° dari *set point*). Karena *initial position* dalam percobaan ini berbeda-beda, maka dapat diamati *rise time* dan *settling time* masing-masing *initial position*. Semakin cepat *rise time* dan *settling time*, maka semakin bagus, karena tujuan dari robot ini adalah mempertahankan keseimbangan di 180°.

1. Pengujian dengan *Initial Position* 175°

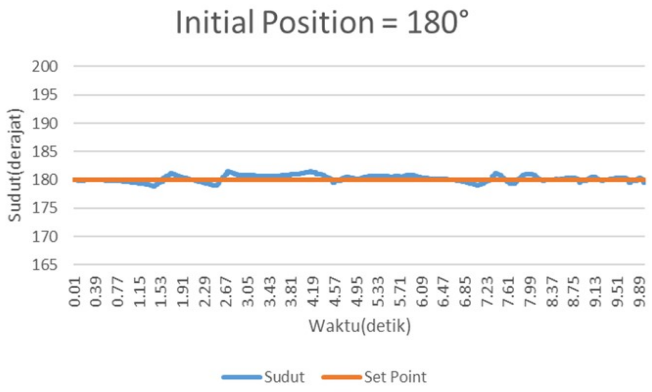
Pengujian pertama dengan *initial position* sebesar 175°. Hasilnya dapat dilihat pada gambar 8.

Gambar 8. Hasil Pengujian dengan *Initial Position* 175°

Berdasarkan grafik pada gambar 8, dapat dilihat bahwa robot dapat bertahan seimbang, bahkan bisa lebih dari 10 detik. Bahkan dari gambar 8 terlihat bentuk *output* PID yang hampir mendekati sempurna. Hasil pengujian menunjukkan *rise time* sebesar 0,22 s dan *settling time* sebesar 2,37 s.

2. Pengujian dengan *Initial Position* 180°

Pengujian berikutnya adalah *initial position* sebesar 180°. Metode yang digunakan masih sama. Hasilnya dapat dilihat pada gambar 9.

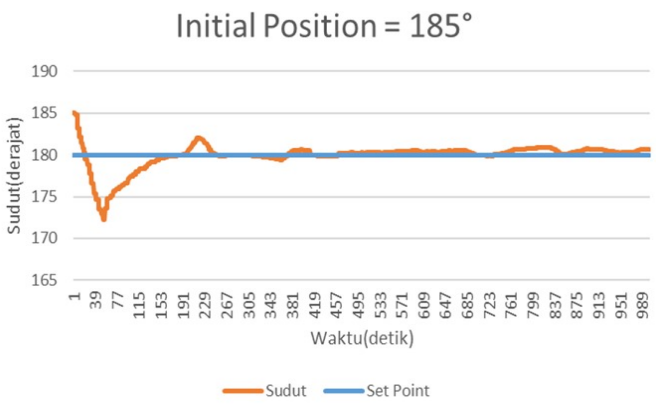


Gambar 9. Hasil Pengujian dengan *Initial Position* 180°

Pada gambar 9, dapat diketahui bahwa robot dapat mempertahankan keseimbangannya, bahkan hingga 10 detik lebih. Pengujian ini sebenarnya sama dengan ketika pengujian mencari Kp, Ki, Kd terbaik, karena saat mencari Kp, Ki, Kd terbaik, *initial position* yang digunakan adalah 180°. Oleh karena itu, tidak ada *rise time* dan *settling time*.

3. Pengujian dengan *Initial Position* 185°

Pengujian berikutnya adalah dengan *initial position* sebesar 185°. Hasilnya dapat dilihat pada gambar 10.

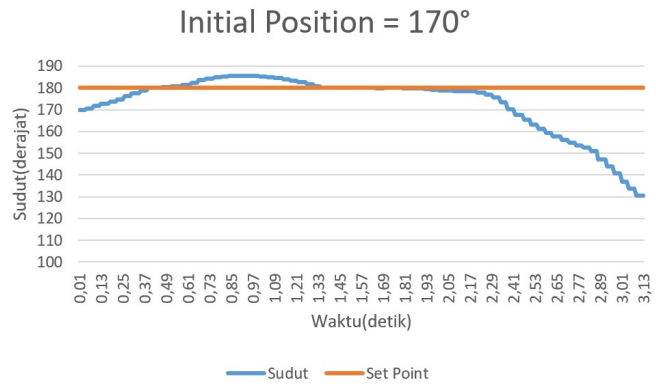


Gambar 10. Hasil Pengujian dengan *Initial Position* 185°

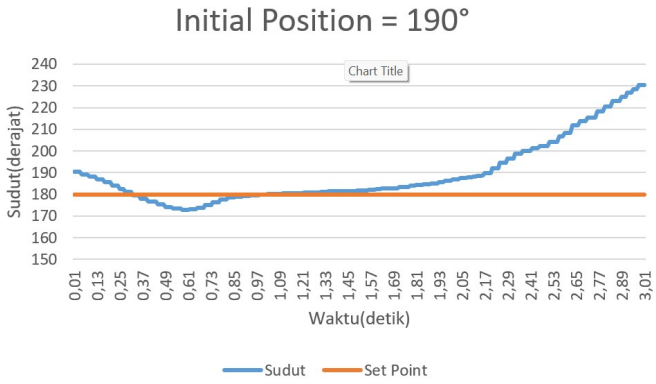
Pada gambar 10, dapat dilihat bahwa dengan *initial position* sebesar 185°, robot masih dapat mempertahankan keseimbangan. Gambar grafik menyerupai grafik saat *initial position* sebesar 175°, karena sama-sama selisih 5° dari *set point*. *Rise time* cukup cepat yaitu 0.2 s, sedangkan *settling time* sebesar 2.48 s.

4. Pengujian dengan *Initial Position* 170° dan 190°

Pada pengujian ini, sistem dijalankan dengan *initial position* robot 170° dan 190°. Hasil pengujian dengan *initial position* 170° dapat dilihat pada gambar 11 dan hasil pengujian dengan *initial position* 190° dapat dilihat pada gambar 12.



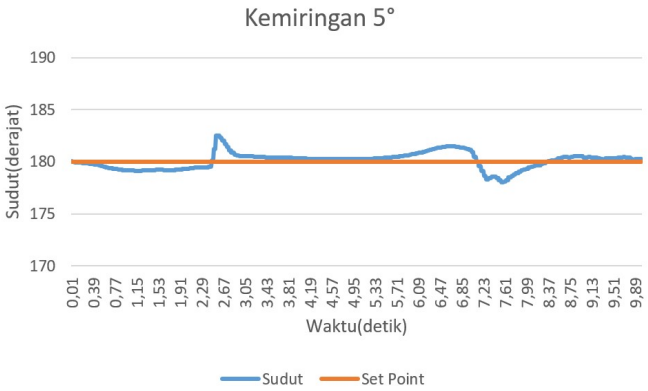
Gambar 11. Hasil Pengujian dengan *Initial Position* 170°



Gambar 12. Hasil Pengujian dengan *Initial Position* 190°

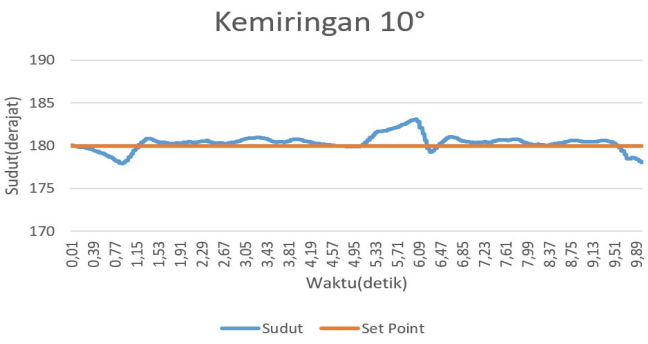
Dari gambar 11 dan 12, terlihat bahwa sistem berhasil menyeimbangkan robot hanya sekitar 1,5 sampai 2 detik setelah itu sistem gagal menyeimbangkan robot.
Berdasarkan percobaan *initial position* mulai dari 170° hingga 190°, dapat disimpulkan bahwa robot masih mampu menjalankan tugasnya untuk mempertahankan keseimbangan ketika *initial position* berada di *range* 175°-185° (hanya ±5° dari *set point* 180°).

C. Pengujian pada Bidang Miring

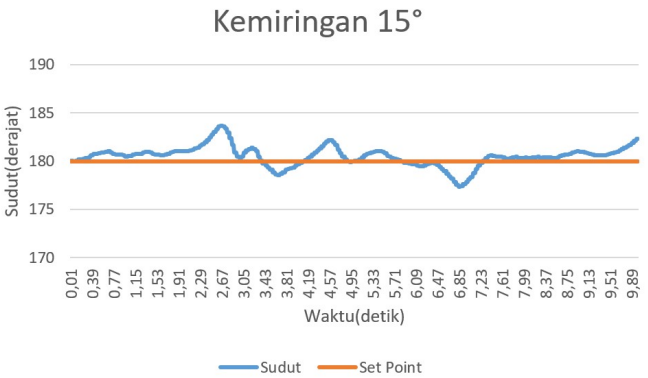


Gambar 13. Hasil Pengujian dengan Sudut Kemiringan 5°

Pada pengujian terakhir, akan diuji tingkat kemiringan maksimal yang dapat dilalui oleh robot. Robot akan diuji berjalan di lintasan dengan sudut kemiringan 5°, 10° dan 15°. Parameter yang digunakan adalah melihat apakah robot dapat berjalan maju dan mempertahankan keseimbangannya di atas lintasan bidang miring. Hasil pengujian dengan sudut kemiringan 5°, 10° dan 15° dapat dilihat pada gambar 13, 14, dan 15.



Gambar 14. Hasil Pengujian dengan Sudut Kemiringan 10°



Gambar 15. Hasil Pengujian dengan Sudut Kemiringan 15°

Pada gambar 13, terlihat bahwa robot dapat mempertahan keseimbangannya dengan baik saat melalui lintasan dengan kemiringan 5°. Mayoritas sudut poisi robot yang terbaca berada di *range* 179°-181° dimana artinya robot ini dapat maju dan mundur serta menyeimbangkan diri dengan baik. Sedangkan untuk lintasan dengan kemiringan 10° (Gambar 14), terlihat bahwa robot masih dapat menyeimbangkan diri, namun tidak sebaik saat kemiringan 5°. Sedangkan untuk

lintasan dengan kemiringan 15° (Gambar 15), terlihat bahwa robot mengalami kesusahan dalam menyeimbangkan diri.
Dari beberapa kemiringan yang dicoba, dapat dilihat bahwa robot ini hanya mampu menanjak dengan baik di kemiringan 5°. Robot sudah tidak mampu menanjak dengan baik di kemiringan lebih dari 5°.

IV. KESIMPULAN

Berdasarkan hasil pengujian yang telah dilakukan maka dapat disimpulkan bahwa sistem yang telah didesain dengan kontroler PID dapat berjalan dengan baik. Robot dapat dikontrol untuk berjalan sambil menyeimbangkan dirinya. Pada sistem ini, nilai Kp, Ki, Kd terbaik adalah Kp sebesar 30, Ki sebesar 200, dan Kd sebesar 1. Ada beberapa batasan dalam sistem yang telah didesain yaitu *initial position* untuk robot dapat menyeimbangkan dirinya dengan baik adalah sebesar 175°-185°. Bila di luar range itu, maka robot cendereung untuk jatuh. Untuk lintasan bidang miring, robot hanya mampu berjalan dan menyeimbangkan dirinya dengan baik pada lintasan dengan kemiringan maksimal 5°.

DAFTAR PUSTAKA

[1] R. Bimarta, A. E. Putra, and A. Dharmawan, "Balancing Robot Menggunakan Metode Kendali Proporsional Integral Derivatif," *IJEIS (Indonesian J. Electron. Instrum. Syst.*, vol. 5, no. 1, p. 89, 2015.
[2] Z. Wanli, L. Guoxin, and W. Lirong, "Research on the control method of inverted pendulum based on kalman filter," *Proc. - 2014 World Ubiquitous Sci. Congr. 2014 IEEE 12th Int. Conf. Dependable, Auton. Secur. Comput. DASC 2014*, vol. 2, no. 2, pp. 520–523, 2014.
[3] Y. Yamamoto, *NXTway-GS (Self-Balancing Two-Wheeled Robot) Controller Design - File Exchange - MATLAB Central,* 2009. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/19147-nxtway-gs-self-balancing-two-wheeled-robot-controller-design. [Accessed: 21-Oct-2019].
[4] S. U. Justin Fong, *Preliminary Report: Ballbot,* 2002.
[5] A. Kadis *et al.*, "Modelling, simulation and control of an electric unicycle," *Proc. 2010 Australas. Conf. Robot. Autom. ACRA 2010*, 2010.
[6] Z. Shao and D. Liu, "Balancing control of a unicycle riding," *Proc. 29th Chinese Control Conf. CCC'10*, pp. 3250–3254, 2010.