

# **SELF-BALANCING SCOOTER MENGGUNAKAN METODE KENDALI PROPORSIONAL INTEGRAL DERIVATIF**

Bambang Nur Cahyono<sup>\*)</sup>, Sumardi, and Budi Setiyono

Jurusan Teknik Elektro, Universitas Diponegoro Semarang  
Jl. Prof. Sudharto, SH, Kampus UNDIP Tembalang, Semarang 50275, Indonesia

<sup>\*)</sup>E-mail: bambangnurcahyono08@gmail.com

## **Abstrak**

Self-Balancing Scooter merupakan suatu robot mobile yang memiliki dua buah roda disisi kanan dan kirinya yang tidak akan seimbang apabila tanpa adanya kontroler. Self-Balancing Scooter ini merupakan pengembangan dari self balancing robot, yang bisa ditumpangi manusia. Contoh real dari balancing scooter adalah SEGWAY Personal Transporter yang di produksi oleh SegWay TM. Tujuan utama dari penelitian ini adalah mengimplementasikan metode kontrol yang baik untuk menjaga badan robot seimbang dalam posisi tegak lurus terhadap permukaan bumi dan juga bisa dikendarai. Balancing scooter ini menggunakan sensor MPU - 6050 Module 3 Axis Gyroscope+Accelerometer untuk mendeteksi kemiringan serta mendeteksi kecepatan sudut badan robot ketika akan terjatuh. Sedangkan untuk penggeraknya digunakan dua buah motor DC. Dari hasil pengujian di dapatkan bahwa kontrol PID dengan nilai  $K_p = 40$   $T_i = 0,5$  dan  $T_d = 0,01$  merupakan hasil yang paling maksimal dengan nilai  $T_r$  sebesar 1 detik dan  $T_s$  sebesar 3 detik. Hasil pengujian dari complementary filter yang paling maksimal adalah dengan koefisien (a) sebesar 0,97.

*Kata Kunci: Complementary Filter, MPU-6050, PID, Self-Balancing Scooter*

## **Abstract**

Self-Balancing Scooter is a mobile robot that has two wheels on the right and the left that will not be balanced if without a controller. Self-Balancing Scooter is the development of self-balancing robot, which can be boarded humans. A real example of self-balancing scooter is SEGWAY Personal Transporter is produced by Segway TM. Self-Balancing Scooter requires a good control method and powerful tool to maintain the position of the robot in a position perpendicular to the earth's surface, without need for other control from the outside. The main purpose of this final project is to implement a good control method to keep your body in a balanced robot perpendicular to the surface of the earth and human beings can also be boarded. Balancing scooter uses sensors MPU - 6050 Module 3 Axis Gyroscope + accelerometer to detect tilt and angular velocity detecting robot body when it will fall. While for driving two DC motors are used. From the test results it appears that the PID control with the value of  $K_p = 40$   $T_i = 0.5$  and  $T_d = 0.01$  is the maximum result with the value of  $T_r$  1 second and  $T_s$  3 second. The test results of complementary filter are to maximal coefficient (a) of 0.97.

*Keywords: Complementary Filter, MPU-6050, PID, Self-Balancing Scooter*

## **1. Pendahuluan**

Perkembangan alat transportasi untuk aktifitas sehari-hari harus di dukung dengan teknologi terkini. Teknologi sekarang ini mempunyai standar yaitu ramah lingkungan, ringan, fleksibel dan memudahkan aktifitas sehari - hari. Karena kegiatan perkantoran yang semakin padat, mendorong peneliti melakukan terobosan alat transportasi yang ramah lingkungan fleksible dan efisien.

Pada tahun 2001 Dean Kamen mengumumkan bahwa Segway merupakan alat transportasi yang cocok untuk

keperluan hiburan, jalan-jalan, patroli dalam kota, aktifitas perkantoran. Belum lagi berkembangnya komponen elektronika yang sangat pesat dan murah, memicu banyak penelitian untuk mengembangkan dari tingkat pendidikan sampai komersil. Robot keseimbangan merupakan robot yang memiliki prinsip kerja mirip seperti sistem inverted pendulum<sup>[10]</sup>.

penelitian ini adalah untuk mendesain dan membangun *self-balancing scooter* yang mampu menyeimbangkan dirinya tegak lurus terhadap permukaan bumi di daerah bidang datar dan juga mampu ditumpangi manusia sebagai alat transportsi personal. Pada penelitian ini

digunakan mikrokontroler AVR ATmega8535, sensor *accelerometer* dan *gyroscope* serta kendali Proporsional Integral Derivatif (PID) sebagai metode pengendali.

## 2. Metode

### 2.1 Kendali Proporsional Integral Derivatif (PID)

Pengendali PID adalah suatu sistem pengendali yang merupakan gabungan antara pengendali proporsional, integral, dan turunan (derivative). Dalam waktu kontinyu, sinyal keluaran pengendali PID dapat dirumuskan sebagai berikut.

$$u(t) = K_p e(t) + K_i \int_0^t e(t) dt + K_d \frac{de(t)}{dt}$$

atau

$$u(t) = K_p \left( e(t) + \frac{1}{T_i} \int_0^t e(t) dt + T_d \frac{de(t)}{dt} \right)$$

dengan:

$u(t)$  = sinyal keluaran pengendali PID

$K_p$  = konstanta proporsional

$T_i$  = waktu integral

$T_d$  = waktu turunan

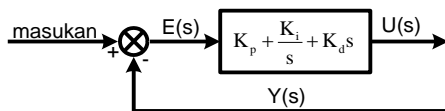
$K_i$  = konstanta integral

$K_d$  = konstanta turunan

$e(t)$  = sinyal kesalahan (referensi – output)

Fungsi alih pengendali PID (dalam domain  $s$ ) dapat dinyatakan sebagai berikut.

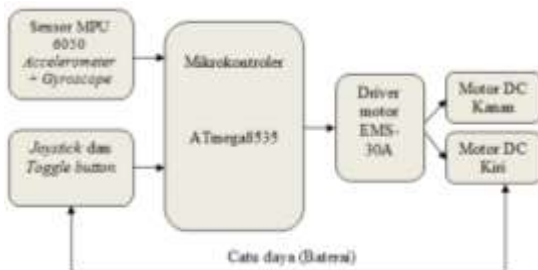
$$G_c(s) = K_p + \frac{K_i}{s} + K_d s$$



Gambar 1 Diagram blok pengendali PID

### 2.2 Perancangan Perangkat Keras

Secara umum perancangan perangkat keras *self-balancing scooter* dapat dilihat pada diagram blok Gambar 1.



Gambar 2 Diagram blok perancangan perangkat keras

Tiap-tiap bagian dari diagram blok sistem di atas dapat dijelaskan sebagai berikut :

1. Sensor MPU-6050 3-Axis Accelerometer + 3-Axis Gyroscope merupakan sensor yang digunakan untuk mendeteksi sudut kemiringan *self-balancing scooter* terhadap permukaan bumi.
2. Joystick berfungsi untuk mengendalikan *self-balancing scooter* (belok kiri atau belok kanan) saat *self-balancing scooter* dalam mode *ride on*.
3. Toggle button berfungsi untuk memilih mode *self balancing* atau *ride on* pada *balancing scooter*.
4. Driver motor EMS-30A sebagai driver motor DC, yang terdiri dari dua buah untuk menggerakkan motor DC kanan dan kiri.
5. Motor DC berfungsi sebagai sistem penggerak *self-balancing scooter* beroda dua.
6. Baterai berfungsi sebagai catu daya sistem keseluruhan pada *self-balancing scooter*.
7. Mikrokontroler ATmega8535 berfungsi sebagai pusat pengendali pada *self-balancing scooter*.

#### 2.2.1 Desain Mekanik

Sistem mekanik dalam pembuatan *self-balancing scooter* ini sangat berpengaruh pada proses keseimbangan, desain mekanik *self-balancing scooter* ini diusahakan dapat seimbang pada titik pusat massanya. Perbedaan berat antara sisi depan dan sisi belakang pada *self-balancing scooter* berpengaruh terhadap kestabilan sistem dan akan mempersulit pengendalian *self-balancing scooter*.



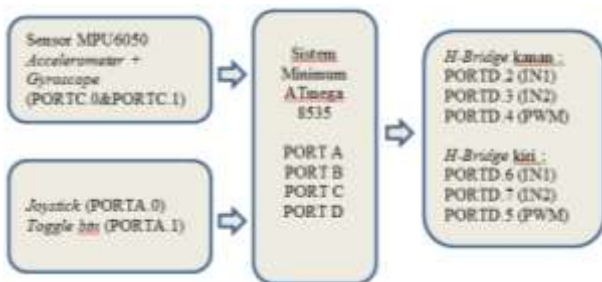
Gambar 3 Ilustrasi mekanik *self-balancing scooter* tampak depan



Gambar 4 Ilustrasi mekanik *self-balancing scooter* tampak samping

### 2.2.2 AVR ATmega8535

Penggunaan (alokasi) port mikrokontroler ATmega 8535 yang digunakan adalah PC.0 & PC.1 sebagai masukan yang berasal dari sensor MPU6050 Accelerometer + Gyroscope melalui komunikasi I2C. PA.0 dihubungkan ke Joystick yang digunakan untuk melakukan belokan (turning) saat self-balancing scooter dalam mode ride on. PA.1 dihubungkan ke toggle button berfungsi untuk memilih mode self balancing atau mode ride on pada balancing scooter. Sedangkan PD.2-PD.7 dihubungkan ke driver motor DC EMS-30A.



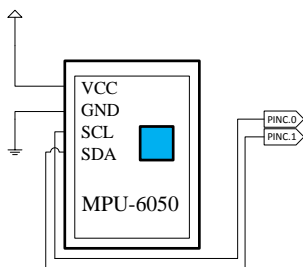
Gambar 5 Alokasi port ATmega 8535

### 2.2.3 Sensor MPU-6050

Sensor MPU-6050 membutuhkan tegangan kerja 3,3V. Tetapi modul ini telah dilengkapi dengan regulator tegangan 3,3V sehingga bisa langsung dihubungkan ke tegangan 5V. Sensor ini mempunyai dua buah keluaran yaitu SCL dan SDA masing-masing dihubungkan ke PC.0 dan PC.1.



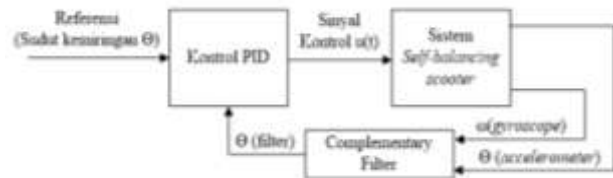
Gambar 7 Sensor MPU-6050 3-Axis Accelerometer+ 3-Axis Gyroscope



Gambar 8 Konfigurasi Sensor MPU-6050

### 2.3 Perancangan Perangkat Lunak

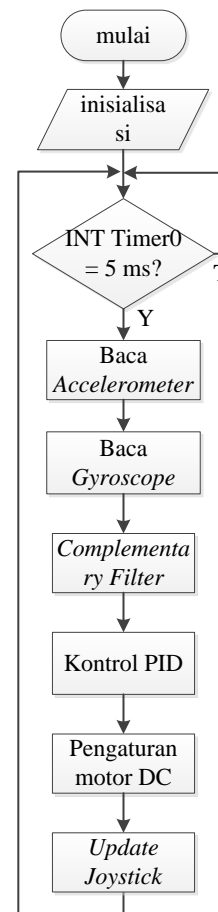
Metode kontrol yang digunakan untuk aplikasi ini adalah kontrol Proporsional Integral Derivatif, dengan referensi berasal dari kemiringan sudut.



Gambar 9 Diagram blok sistem kontrol self-balancing scooter

#### 2.3.1 Perancangan Perangkat Lunak

Gambar 10 adalah flowchart program utama self-balancing scooter menggunakan metode kendali proporsional integral derivatif.



Gambar 10 Flowchart program utama

Program utama dari perancangan perangkat lunak self-balancing scooter ini meliputi beberapa subrutin, yaitu :

1. Interupsi timer 0  
Merupakan subrutin program yang digunakan sebagai waktu *sampling* nilai sensor, *complementary filter* dan kontrol PD.
2. Baca Accelerometer  
Merupakan subrutin program yang digunakan untuk membaca data dari sensor *accelerometer*.
3. Baca gyroscope  
Merupakan subrutin program yang digunakan untuk membaca data dari sensor *gyroscope*.
4. Complementary filter  
Merupakan subrutin program yang digunakan untuk menggabungkan data sensor *accelerometer* dan *gyroscope*.
5. Kontrol PID  
Merupakan subrutin program yang digunakan sebagai algoritma pengontrolan self-balancing scooter.
6. Pengaturan motor DC  
Merupakan subrutin program yang digunakan untuk mengatur kecepatan dan arah putar motor DC.
7. Update Joystick  
Merupakan subrutin program yang digunakan untuk pembacaan *joystick* dan *toggle button*.

### 3. Hasil dan Analisis

#### 3.1 Pengujian Perangkat Keras

##### 3.1.1 Pengujian Sensor Accelerometer

*Self-balancing scooter* hanya membutuhkan pembacaan sudut kemiringan sekitar  $40^{\circ}$  sampai  $-40^{\circ}$ , maka pengujian hanya dilakukan pada jangkauan  $60^{\circ}$  sampai  $-60^{\circ}$  dengan kelipatan 10.

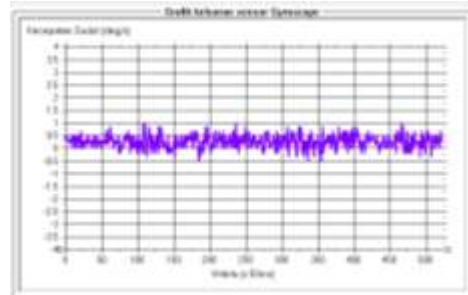
Tabel 1 Hasil pengukuran sudut kemiringan sensor accelerometer

Sudut aktual ( $^{\circ}$ )	Sudut sensor accelerometer ( $^{\circ}$ )	Error ( $^{\circ}$ )
-60	-59,7	-0,3
-50	-48,9	-1,1
-40	-40,1	0,1
-30	-29,6	-0,4
-20	-19,0	-1
-10	-8	-2
0	0,2	-2
10	11,5	-1,5
20	22,5	-2,5
30	32,2	-2,2
40	43,4	-3,4
50	53,2	-3,2
60	63,3	-3,3
RMSE		2,08

Dari Tabel 1, terlihat bahwa sensor *accelerometer* masih terdapat kesalahan (*error*), hal ini dapat disebabkan karena pengukuran yang kurang tepat, pembulatan pecahan dalam pemrograman dan *noise* yang dihasilkan sensor itu sendiri.

##### 3.1.2 Pengujian Sensor Gyroscope

Pengujian dilakukan dengan melihat keluaran dari sensor *gyroscope* melalui grafik.



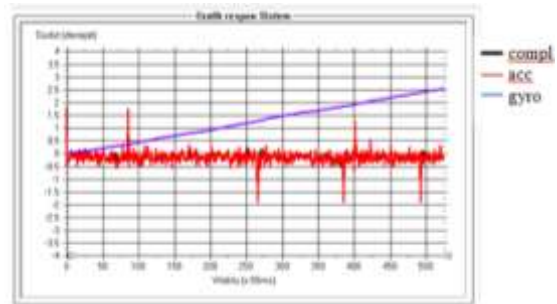
Gambar 11 Grafik sensor *gyroscope* ketika diam

Sinyal yang dihasilkan mempunyai nilai dan berubah-ubah. Seharusnya jika sensor *gyroscope* dalam keadaan diam, keluarannya sama dengan 0. Hal ini dikarenakan sensor *gyroscope* mempunyai nilai bias.

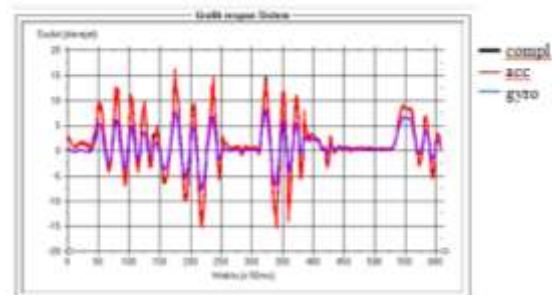
### 3.2 Pengujian Perangkat Lunak

#### 3.2.1 Pengujian Complementary Filter

Pengujian dilakukan dengan memberikan nilai koefisien filter sebesar 0,97 dan waktu *sampling* 5 ms. Pengujian dilakukan dengan melihat sinyal keluaran dari *complementary filter* saat sistem sedang diam dan saat berotasi.



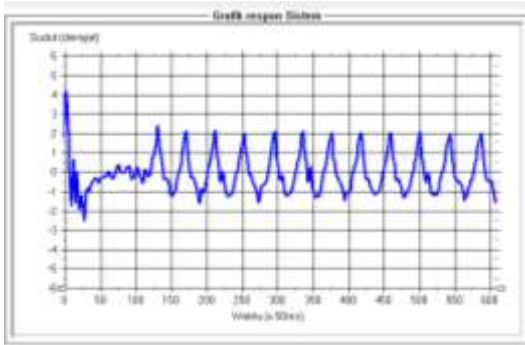
Gambar 12 Grafik *complementary filter* saat diam



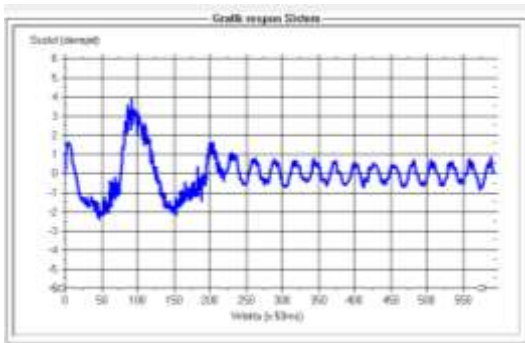
Gambar 13 Grafik *complementary filter* saat berotasi

### 3.2.2 Penentuan nilai Parameter Kontrol Proporsional Integral Derivatif

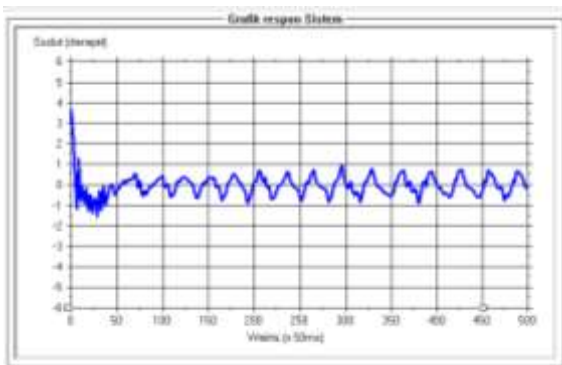
Penentuan nilai parameter kontrol Proporsional ( $K_p$ ) Integral ( $T_i$ ) dan derivative ( $T_d$ ) dilakukan dengan cara *trial and error* dengan nilai *set point*  $0^0$ . Pengujian algoritma kontrol Proporsional Integral Derivatif ini adalah mencari nilai optimal parameter kontrol Proporsional Integral dan Derivatif ( $K_p$   $T_i$  dan  $T_d$ ) terhadap respon sistem *self-balancing scooter*.



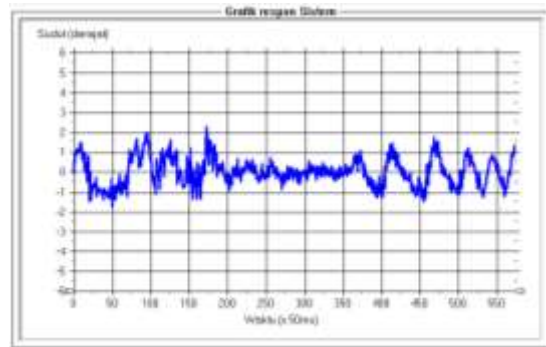
Gambar 14 Respon *self-balancing scooter* dengan nilai  $K_p=25$   $T_i=0,5$   $T_d=0,01$



Gambar 15 Respon *self-balancing scooter* dengan nilai  $K_p=25$   $T_i=1,5$   $T_d=0,1$



Gambar 16 Respon *self-balancing scooter* dengan nilai  $K_p=40$   $T_i=0,5$   $T_d=0,01$

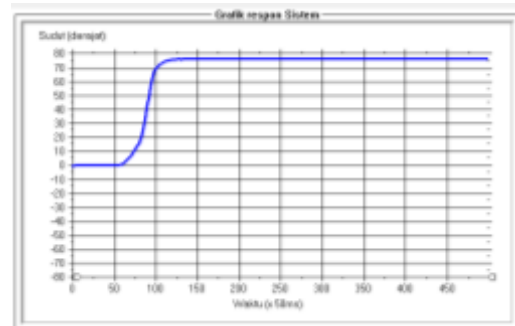


Gambar 17 Respon *self-balancing scooter* dengan nilai  $K_p=40$   $T_i=1,5$   $T_d=0,1$

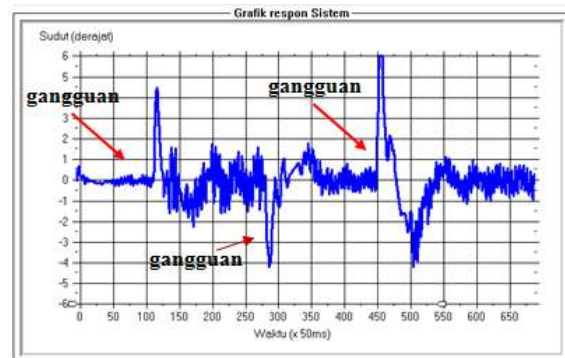
Respon yang terbaik atau optimal ditunjukkan pada Gambar 16 dengan nilai  $K_p=40$   $T_i=0,5$  dan  $T_d=0,01$ .

### 3.2.3 Pengujian Kontrol *Open Loop* dan *Close Loop* terhadap Gangguan saat mode *Self-balancing* dan mode *Ride on*.

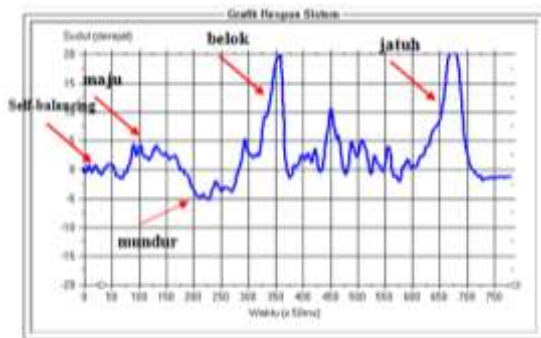
Gangguan yang diberikan adalah dengan memberikan dorongan kearah depan atau belakang saat mode *self-balancing* dan mode *ride on*. Jika menggunakan kontrol *open loop*, *self-balancing scooter* tidak dapat menyeimbangkan diri walaupun tanpa adanya gangguan.



Gambar 18 Grafik respon kontrol *open loop self-balancing scooter*



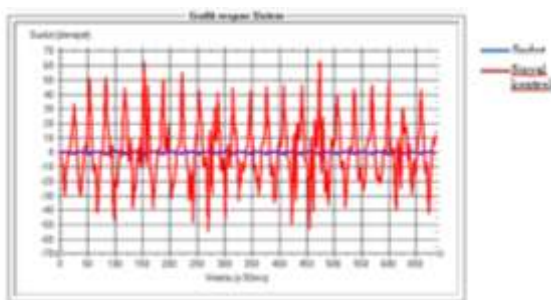
Gambar 19 Grafik respon *self-balancing scooter* saat mode *self-balancing* ketika terjadi gangguan



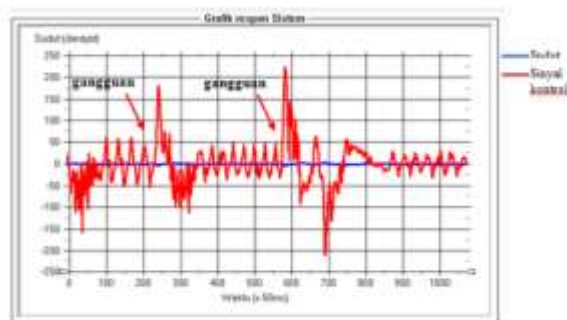
Gambar 20 Grafik respon self-balancing scooter saat mode ride on

### 3.2.4 Pengujian Sinyal Kontrol

Pengujian dilakukan dengan melihat keluaran sinyal Kontrol dengan gangguan dan tanpa adanya gangguan.



Gambar 21 Grafik respon sinyal kontrol proporsional integral derivatif tanpa gangguan



Gambar 22 Grafik sinyal kontrol PID self-balancing scooter ketika terjadi gangguan

## 4. Kesimpulan

Self-Balancing Scooter dapat digunakan sebagai alat transportasi personal pada permukaan yang datar, dengan cara mengendarai dimiringkan ke depan (maksimal  $15^\circ$ ) untuk arah maju dan untuk arah mundur dengan cara dimiringkan ke belakang (maksimal  $-15^\circ$ ). Semakin besar nilai koefisien filter ( $a$ ) menyebabkan lamanya waktu

update yang menyebabkan melambatnya respon keluaran sudut. Nilai koefisien filter dan waktu sampling yang paling optimal adalah  $a=0,97$  dan  $dt=5$  ms. Nilai parameter kontrol Proporsional Integral dan Derivatif yang paling optimal adalah saat  $K_p=40$   $T_i=0,5$  dan  $T_d=0,01$ . Dengan pegujian pemberian gangguan berupa dorongan self-balancing scooter mampu mengembalikan pada kondisi steady state dengan  $T_r$  1 detik dan  $T_s$  3 detik.

## Referensi

- [1]. Bejo, Agus, *C&AVR Rahasia Kemudahan Bahasa C dalam Mikrokontroler ATmega8535*. Graha Ilmu, Yogyakarta, 2008.
- [2]. Camacho, Oscar, *AC317: Solutions Based in Accelerometers*, Freescale Semiconductor Inc., USA, 2007.
- [3]. Clifford, Michelle and Leticia Gomez, *AN3107: Measuring Tilt with Low-g Accelerometers Rev. 0*, Freescale Semiconductor Inc., USA, 2005.
- [4]. Colton, Shane, *The Balance Filter: A Simple Solution for Integrating Accelerometer and Gyroscope Measurements for a Balancing Platform*, <http://web.mit.edu/scolton/www/filter.pdf>, September 2011.
- [5]. Elvin, Toh Boon Heng, *Development of a Self Balancing scooter*, Nanyang Technological University, Mei 2007.
- [6]. Kurniawan, Dayat, *Aplikasi Elektronika dengan Visual C# 2008 Express Edition*, Elex Media Komputindo, Jakarta, 2010.
- [7]. Ogata, Katsuhiko, *Teknik Kontrol Automatik Jilid 1*, Diterjemahkan Oleh Ir. Edi Leksono, Erlangga, Jakarta, 1994.
- [8]. Setiawan, Iwan, *Kontrol PID untuk Proses Industri*, Elex Media Komputindo, Jakarta, 2008.
- [9]. Tuck, Kimberly, *AN3461: Tilt Sensing Using Linear Accelerometers Rev. 2*, Freescale Semiconductor Inc., USA, 2007.
- [10]. -----, SegwayPT, [http://en.wikipedia.org/wiki/Segway\\_PT](http://en.wikipedia.org/wiki/Segway_PT), Februari 2013.
- [11]. -----, MPU-6050 3-axis accelerometer + 3-axis gyroscope Module Data Sheet, <http://www.invensense.com/mems/gyro/mpu6050.html>, Februari 2013.
- [12]. -----, Low-pass filter, [http://en.wikipedia.org/wiki/Low-pass\\_filter](http://en.wikipedia.org/wiki/Low-pass_filter), Januari 2013
- [13]. -----, Prinsip kerja motor DC, <http://www.merwardanaku.com/2011/11/prinsip-kerja-motor-arus-searah-dc.html>, Januari 2013
- [14]. -----, ATmega8535 Data Sheet, <http://www.atmel.com>.