PERANCANGAN DAN IMPLEMENTASI PROTOTYPE PENYEIMBANG MOBIL PADA SAAT *DRIFTING*

Amirudin Fanani
Universitas Komputer Indonesia
Jl. Dipati Ukur No. 112, Bandung
Email : fanani.12@gmail.com

Abstrak

Prototype penyeimbang mobil pada saat *drifting* memodelkan sistem kendaraan yang dapat mengurangi terjadinya *drifting*. Sensor untuk mendeteksi proses terjadinya *drifting* digunakan *accelerometer* dan *gyroscope* tipe MPU 6050. *Drifting* terjadi akibat ban belakang tergelincir dengan alur yang lebih besar dari pada ban depan. Perubahan data pada sensor yang diakibatkan perubahan gerak secara mendadak akan direspon oleh mikrokontroller arduino leonardo. Selanjutnya mikrokontroller akan mengendalikan *steering* dan rem agar mobil tidak terjadi *drifting* yang berlebihan. Komponen *steering* dan rem dimodelkan dengan motor servo dc standar.

Kata Kunci : *Accelerometer, Gyroscope, Drifting,* PID, Penyeimbang mobil pada saat *drifting*

I. PENDAHULUAN

Banyaknya kasus kecelakaan roda empat di Indonesia, diakibatkan oleh dua faktor berbeda. Faktor pertama, diakibatkan oleh pengemudi yang tidak dapat mengendalikan kendaraannya dikarenakan jalan licin, pengereman mendadak, dan ban pecah. Faktor kedua, diakibatkan kendaraan yang belum menerapkan teknologi sistem penyeimbang pada saat terjadi *drifting* pada sistem pengereman dan *steering*. Seperti kasus kecelakaan yang dialami Dul putra musisi Ahmad Dhani, dari hasil olah tempat kejadian perkara yang dilakukan pihak kepolisian di lokasi kecelakaan, mobil Mitsubishi Lancer yang dikendarai Dul tergelincir karena terdapat jejak ban yang membekas di jalan. Penelitian tugas akhir ini, memodelkan sistem kendaraan yang dapat mengurangi terjadinya *drifting*. Sensor untuk mendeteksi proses terjadinya *drifting* digunakan *accelerometer* dan *gyroscope* tipe MPU 6050. *Drifting* terjadi akibat ban belakang tergelincir dengan alur yang lebih besar dari pada ban depan. Perubahan data pada sensor yang diakibatkan perubahan gerak secara mendadak akan direspon oleh mikrokontroller arduino leonardo. Selanjutnya mikrokontroller akan mengendalikan *steering* dan rem agar mobil tidak terjadi *drifting* yang berlebihan. Komponen *steering* dan rem dimodelkan dengan motor servo dc standar. Dari permasalahan tersebut maka dirancang suatu sistem yang dapat membantu pengemudi untuk menstabilkan kendaraan secara otomatis pada saat terjadi *drifting* sehingga kendaraan tidak lepas kendali, maka peneliti mengangkat judul "Perancangan dan Implementasi Prototipe Penyeimbang Mobil Pada Saat *Drifting"* sebagai tema dari Tugas Akhir ini.

II. LANDASAN TEORI

A. Sensor MPU 6050

Sensor MPU 6050 merupakan *combo* sensor antara *accelerometer* dan *gyroscope* yang terintegrasi dalam satu chip. Sensor ini merupakan 6 *axis motion processing unit* dengan penambahan regulator tegangan dan 3,3V sehingga bisa langsung dihubungkan ke

tegangan 5V. Sensor MPU 6050 memiliki dua buah *output* yaitu SCL dan SDA.

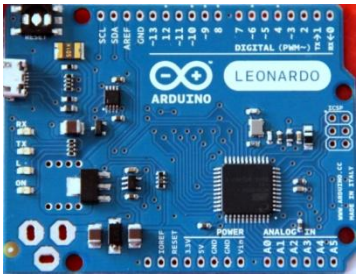


Gambar 1. Sensor MPU 6050 – 3 *axis accelerometer* + 3 *axis gyroscope*

B. Arduino Leonardo

Arduino Leonardo adalah papan mikrokontroler berbasis ATmega32u4. Arduino Leonardo memiliki 20 digital pin *input/output* (yang mana 7 pin dapat digunakan sebagai *output* PWM dan 12 pin sebagai *input* analog), 16 MHz kristal osilator, koneksi mikro USB, jack *power suplay* tegangan, tegangan operasi 5V, *flash memory 32* KB, *SRAM 2,5* KB, *header* ICSP, EEPROM 1 KB, dan tombol *reset*

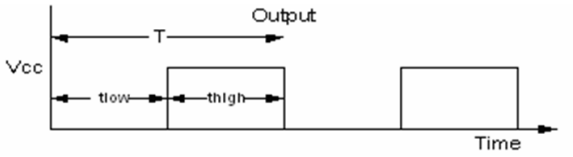
.



Gambar 2 Arduino Leonardo

C. *Pulse Width Modulation (PWM)*

PWM merupakan sebuah metoda untuk membangkitkan sinyal *output* berupa pulsa yang periodanya berulang antara *high* atau *low* dimana kita dapat mengontrol duraasi sinyal *high* atau *low* sesuai dengan yang kita inginkan. PWM bias dibangkitkan secara *software* maupun *hardware*, sehingga dapat dibentuk gelombang dengan *duty cycle* yang dapat diatur sesuai dengan program. *Duty cycle* merupakan perbandingan periode lamanya suatu sistem bernilai logika *high* atau *low*. Variasai *duty cycle* ini memberikan harga tegangan rata-rata yang berbeda-beda.



Gambar 3. Sinyal PWM

D. Kendali PID

PID merupakan kontroller untuk menentukan presisi suatu sistem instrumentasi dengan karakteristik adanya umpan balik pada system tersebut. Komponen control PID ini terdiri dari tiga jenis yaitu *proportional*, *integrative*, dan *derivative*. Ketiganya dapat dipakai bersamaan maupun ssendiri-sendiri tergantung dari respon yang kita inginkan terhadap suatu *plant*, sehingga kelemahan-kelemahan pada salah satu komponen dapat ditutupi oleh komponen yang lain. Komponen I dan D tidak dapat berdiri sendiri dan selalu dikombinasikan dengan komponen P, menjadi pengontrol PI atau PID. Pengontrol PID akan mengeluarkan aksi kontrol dengan membandingkan kesalahan atau *error* yang merupakan selisih dari proses variable dan *set point*, yang akan digunakan sebagai *input* pengontrol untuk mengeluarkan sinyal kontrol.

E. Motor Servo

Motor servo adalah sebuah motor DC dengan sistem umpan balik tertutup dimana posisi rotornya akan diinformasikan kembali ke rangkaian kontrol yang ada di dalam motor servo. Motor ini terdiri dari sebuah motor DC, serangkaian *gear*, potensiometer, dan rangkaian kontrol. Potensiometer berfungsi untuk menentukan batas sudut dari putaran servo. Sedangkan sudut dari sumbu motor servo diatur berdasarkan lebar pulsa yang dikirim melalui kaki sinyal dari kabel motor. Pada motor servo biasanya terdapat tiga buah kabel, yang pertama untuk tegangan masukkan, yang kedua untuk

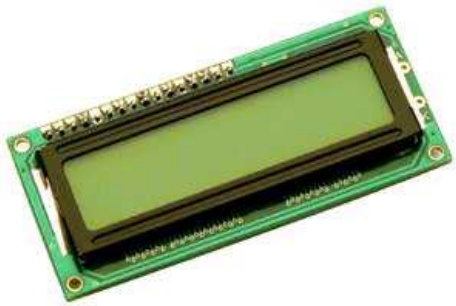
menerima sinyal PWM, dan yang ketiga untuk *ground*.



Gambar 4. Motor Servo

F. *Liquid Crystal Display (LCD)*

Liquid Crystal Display (LCD) adalah satu layar bagian dari modul peraga yang menampilkan karakter yang diinginkan. Layar LCD menggunakan 2 buah lembaran bahan yang dapat mempolarisasikan dan kristal cair diantara kedua lembaran tersebut. Arus listrik yang melewati cairan menyebabkan kristal merata sehingga cahaya tidak dapat melalui setiap kristal. Sehingga dapat mengubah bentuk kristal cairnya membentuk tampilan angka atau huruf pada layar.



Gambar 5. 16x2 *Character* LCD *Module*

G. Motor DC

Motor DC adalah mesin yang mengubah energi listrik arus searah menjadi energi mekanis**.** Cepat lambatnya motor DC dapat diatur berdasarkan *duty cycle* yang diberikan, jika *duty cycle* pada sinyal PWM besar, maka akan menghasilkan rata-rata tegangan yang besar pula sehingga motor DC akan berputar cepat, begitu juga sebaliknya jika *duty cycle* yang diberikan kecil akan mengahasilkan tegangan rata-rata yang kecil dan motor DC akan berputar lebih lambat.



Gambar 6. Motor DC

H. Catu Daya (Baterai)

Baterai *Lithium Polimer* atau biasa disebut dngan LiPo merupakan salah satu jenis baterai yang sering digunakan dalam dunia RC. Baterai ini bersifat cair (*Liquid*), menggunakan elektrolit polimer yang padat, dan mampu menghantarkan daya lebih cepat dan jenis baterai ini adalah hasil pengembangan dari *Lithium Ion*. Baterai Li-Po ini disebut sebagai baterai ramah lingkungan.



Gambar 7. Baterai Li-Po

I. Ubec

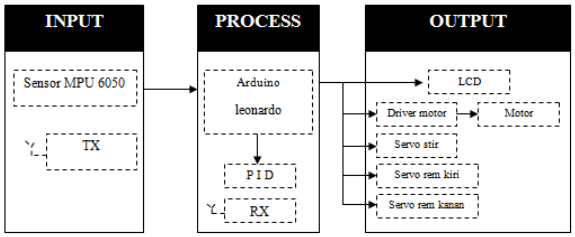
UBEC 3A adalah mode switch DC/DC regulator, mengeluarkan voltase yang konstan dan aman untuk *receiver*, *gyro* dan servo. Tegangan *Output* Ubec yaitu 5 Volt atau 6 Volt. Tegangan *Input* 5,5 V- 26 V.



Gambar 8. Ubec

III. PERANCANGAN ALAT

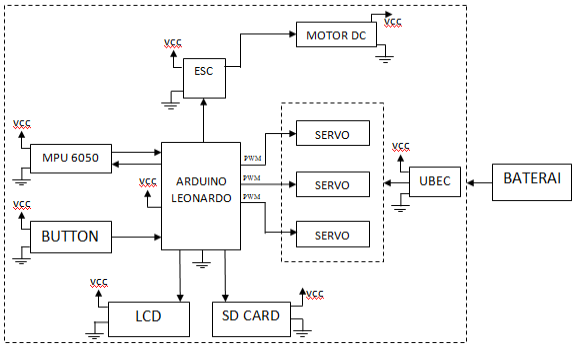
Pada perancangan sistem penyeimbang mobil pada saat *drifting* ini, secara umum terdiri dari 3 bagian utama, yaitu masukan *(input)*, proses, dan keluaran *(output)*. Setiap bagian dari sistem mempunyai fungsi masing-masing yang akan saling berkaitan dalam sistem penyeimbang mobil pada saat *drifting*, seperti terlihat pada gambar 9 dibawah.



Gambar 9. Blok Diagram Sistem

A. Perancangan Perangkat Keras

Perancangan perangkat keras sistem penyeimbang mobil pada saat *drifting* pada tugas akhir ini dimodelkan pada mobil-mobilan *remote control*. Model mobil-mobilan yang dirancang memiliki 1 buah sensor MPU 6050 sebagai pendeteksi perubahan gerak mendadak, 1 buah motor DC beserta *driver* ESC sebagai penggerak mobil, 3 buah motor servo (1 buah untuk *steering* dan 2 buah untuk penggerak rem), dan 1 buah arduino leonardo sebagai pusat pengendali mobil. Adapun detail hubungan antara perangkat *input*, proses, dan *output* seperti terlihat pada *wiring diagram* dibawah.



Gambar 10. *Wiring Diagram* Sistem

Dari diagram *wiring* diatas masing-masing komponen terintegrasi satu dengan yang lain. Tabel 1 dibawah menggambarkan hubungan antara pin arduino dengan perangkat *input* MPU 6050. Sensor MPU 6050 berfungsi memberikan posisi *accelerometer* dan *gyroscope* pada arduino.

Tabel 1. Konfigurasi PIN Sensor MPU 6050 dengan Arduino

No	**Sensor MPU 6050**	**Arduino Leonardo**
1	VCC	5V
2	GND	GND
3	SDA	SDA
4	SCL	SCL

Tabel 2 dibawah menggambarkan antara pin arduino dengan perangkat input *Remote Control*. *Remote control* menerima data dari *remote control* operator berupa frekuensi yang kemudian memberi perintah maju, mundur, kiri, kanan, dan *brake.*

Tabel 2 Konfigurasi PIN *remote control* dengan Arduino

No	*Remote control*	Arduino Leonardo
1	Ch1	Pin 4
2	Ch2	Pin 5
3	Ch3	Pin 6
4	Ch4	Pin7

Tabel 3 dibawah menggambarkan hubungan antara pin arduino dengan perangkat input LCD. LCD ini menunjukkan data sensor, data PID, data *RC*, dan perintah peyimpanan data ke *SD card.*

Tabel 3 Konfigurasi PIN LCD dengan Arduino

No	*LCD*	Arduino Leonardo
1	K	GND
2	A	VCC
3	D7	8
4	D6	9
5	D5	10

6	D4	11
7	E	12
8	RS	13

Tabel 4 dibawah menggambarkan hubungan antara pin arduino dengan *SD Card* yang digunakan untuk menyimpan semua data dari sistem penyeimbang mobil pada saat *drifting* ini.

Tabel 4 Konfigurasi PIN *SD Card* dengan Arduino

No	*SD Card*	Arduino Leonardo
1	MOSI	MOSI
2	MISU	MISU
3	SCK	SCK
4	CS	A5
5	GND	GND
6	VCC	5V

Tabel 5 dibawah menggambarkan hubungan antara pin arduino dengan perangkat motor dan tombol. Motor yang digunakan yaitu motor DC standar yang terhubung pada ESC dan 3 buah motor servo. 1 motor servo untuk *steering* dan 2 buah motor servo untuk rem.

Tabel 5 Konfigurasi PIN Arduino dengan motor dan tombol

No	Arduino Leonardo	Keterangan
1	A0	ESC (motor)
2	A1	Servo *Steering*
3	A2	Servo rem kiri
4	A3	Servo rem kanan
5	A4	Tombol/*button*

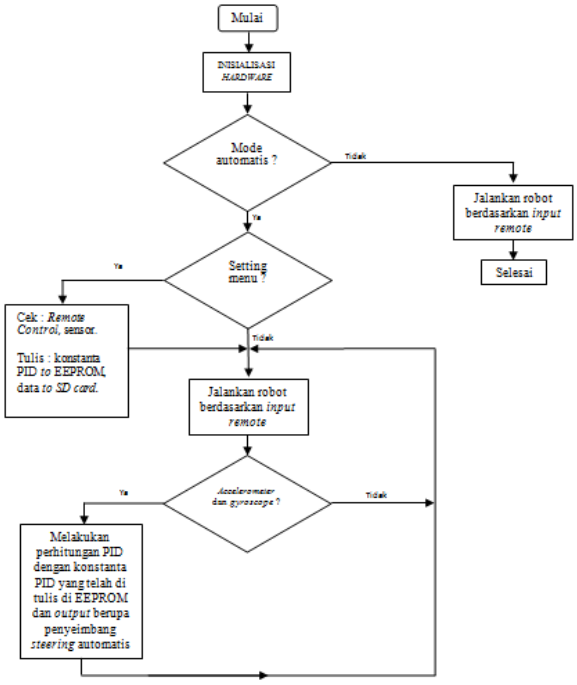
B. Implementasi *hardware* dari sistem ini terbagi menjadi beberapa bagian yang meliputi, implementasi *remote control*, implementasi *power supply* (baterai), implementasi Arduino Leonardo, implementasi sensor MPU 6050 (*accelerometer* dan *gyroscope)*, implementasi servo *steering*, implementasi servo rem, implementasi dan motor DC, implementasi media penyimpanan data, dan implementasi LCD.



Gambar 11. Prototype penyeimbang mobil pada saat *drifting*

C. Perancangan Perangkat Lunak

Perancangan perangkat lunak (*software*) bertujuan untuk menentukan setiap alur eksekusi dari perangkat sistem penyeimbang mobil pada saat *drifting*. Setiap *input* akan diterima dan diproses oleh *software* yang nantinya akan menentukan *output* dari sistem. Berikut alur kerja (*flowchart*) dari sitem yang dirancang.



Gambar 12. *Flowchart* Sistem Penyeimbang Mobil Pada Saat *Drifting*

D. Algoritma

Perancangan algoritma perangkat lunak merupakan tahap pengkodean antara perangkat *input*, *output* terhadap sistem kendali proses (arduino) yang akan digunakan untuk program penyeimbang mobil pada saat *drifting.*

Accelero Gyro {melakukan pemanggilan data posisi dari sensor MPU 6050 berupa posisi *accelerometer* dan *gyroscope*}

Algoritma

```
//inisialisasi mpu6050
void Gyro_acc_init() {

Wire.beginTransmission(MPU6050_A
DDRESS);
  writeTo(0x6B, 0x80); //reset
device
  delay(10);
  writeTo(0x6B, 0x03);
//referensi penggunaan gyro z
PLL with z axis gyro
configuration
  writeTo(0x1A,
MPU6050_DLPF_CFG); //konfigurasi
  writeTo(0x1B, 0x18);//gyro
config range +- 2000
derajat/detik
  writeTo(0x1C, 0x10);//acc
config range +- 8g
  Wire.endTransmission();
}

//pengambilan data posisi
accelerometer dan gyro
void getData_mpu6050(){
  uint8_t BytesToRead = 6;
//kaki no 6
  readFrom( DATAXG1,
BytesToRead, _buff); //gyro
  x_g= (( (((int)_buff[0]) << 8)
| _buff[1])/8)+5; //lakukan
operasi or antara buff[0] OR
buff[1]/8
  y_g= (( (((int)_buff[2]) << 8)
| _buff[3])/8);
  z_g= (( (((int)_buff[4]) << 8)
| _buff[5])/8)-2;
  readFrom( DATAXA1,
BytesToRead, _buff); //accel
  x_a= ( (((int)_buff[0]) << 8)
| _buff[1])/8-10;
  y_a= ( (((int)_buff[2]) << 8)
| _buff[3])/8;
  z_a= ( (((int)_buff[4]) << 8)
| _buff[5])/8;

  z_g = z_g * -1; //karena z
gyroscope digunakan sebagai
referensi, maka pemasangan
hardware terbalik

}
```

Remote Control {menerima data dari *remote control* operator berupa frekuensi}

Algoritma

```
void read_rc() {

  ch2 = pulseIn(5, HIGH,
TimeOut);
  ch3 = pulseIn(7, HIGH,
TimeOut);
  ch2 = constrain(ch2, 900,
1750);

  if (ch3 > 1500)
  {mode_numb++;}

  if (mode_numb > 1)
  {mode_numb = 0;}

  if(mode_numb == 0)
  {mode = 'Manual';}

  if(mode_numb == 1)
  {mode = 'Auto';}

  if(ch2 >= 1250 && ch2 <=
1300){
    Arah = 'Stop';
  }
  if (ch2  < 1150 && ch2 >
```

```
1050) {
    Arah = 'Mundur';
    ch2 = 1100;
    //ch2 =
map(ch2,1500,1850,1500,1900);
  }
  if (ch2 <= 1000) {
    Arah = 'Brake';
    ch2 = 1275;
  }
  else if (ch2 > 1300)
  {Arah = 'Maju';}

}
```

PID {menerima data dari sensor MPU 6050, selanjutnya diproses dan hasil proses dikeluarkan untuk *steering* dan *brake*}

```
void pid_gyro(){
//gyro  roll

  set_point = 0;

  pv =z_g;
  error = set_point-pv;
  P = gyro_gain * error;

  d1 = (gyro_gain/20) * 10;
  d2 = d1/Ts;
  d3 = error - last_error_G;
  D = d2*d3;


  last_error_G=error;
  Pid_G= (P+D)/100;
  //Serial.print(Pid_G);
}

void pid_acc_X(){
  set_point = 0;
  pv =x_a;

  error = set_point-pv;
  P = Kp * error;
  d1 = Kd * 10;
  d2 = d1/Ts;
  d3 = error - last_error_Y;
  D = d2*d3;
  i1 = Ki/10;
  i2 = error+last_error_Y;
  i3 = i1*i2;
  I = i3*Ts;

  last_error_Y=error;
  Pid_A_X  = (P+D+I)/10;

}


void Calc(){
  if((Pid_A_X + Pid_G) <50 &&
(Pid_A_X + Pid_G) > -50){
    servo_out = ch1;
  }


  else {
    servo_out = ch1+(Pid_A_X
+ Pid_G) ;
  }

  servo_out =
constrain(servo_out,1450,1750
);
}

void data_out(){
  if(Arah == 'Brake'){
    rem_L_out = 1300;
    rem_R_out = 1700;
  }

  if(Arah != 'Brake'){
    rem_L_out = 1500;
    rem_R_out = 1500;
  }
```

```
  if(mode == 'Auto'  && Arah
!= 'Brake' && (Pid_A_X +
Pid_G) > 40){
    rem_R_out = 1700;
  }
  if(mode == 'Auto'  && Arah
!= 'Brake' && (Pid_A_X +
Pid_G) < -40){
    rem_L_out = 1300;
  }

servo.writeMicroseconds(servo
_out);

esc.writeMicroseconds(esc_out
);

rem_L.writeMicroseconds(rem_L
_out);

rem_R.writeMicroseconds(rem_R
_out);
       }
```

LCD {menampilkan data yang diterima dari proses pengolahan arduino berupa data sensor MPU 6050, nilai PID dan mode dari *remote control*}

Algoritma

```
void print_lcd_MPU(){
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print(x_g);
  lcd.setCursor(6,0);
  lcd.print(y_g);
  lcd.setCursor(12,0);
  lcd.print(z_g);

  lcd.setCursor(0,1);
  lcd.print(x_a);
  lcd.setCursor(6,1);
  lcd.print(y_a);
  lcd.setCursor(12,1);
  lcd.print(z_a);
}


void  print_rc(){
  Serial.print("ch1= ");
  Serial.print(ch1);
  Serial.print(" ch2= ");
  Serial.print(ch2);
  Serial.print(" ch3= ");
  Serial.print(ch3);


}

void  print_sensor(){
  Serial.print("  Gyro X= ");
  Serial.print(x_g);
  Serial.print("  ,Y= ");
  Serial.print(y_g);
  Serial.print("  ,Z= ");
  Serial.print(z_g);
  Serial.print("   ");

  Serial.print(" ,ACC X= ");
  Serial.print(x_a);
  Serial.print("  ,Y= ");
  Serial.print(y_a);
  Serial.print("  ,Z= ");
  Serial.print(z_a);
  Serial.print("   ");

}


void print_pid(){
  Serial.print("  Pid_X= ");
  Serial.print(Pid_A_X);
  Serial.print("  Pid_Z= ");
  Serial.println(Pid_G);
       }
```

Tombol {menerima data analog yang berasal dari empat buah *button* yang disusun seri 4 buah resistor}

Algoritma

```
void tombol() {
```

```
  delay(50);
  baca_tombol =
analogRead(A4);

//Serial.println(baca_tombol)
;

  if (baca_tombol >= 160 &&
baca_tombol <= 180) {
    baca_tombol = 'tambah';
  }
  if (baca_tombol >= 225 &&
baca_tombol <= 245) {
    baca_tombol = 'kurang';
  }
  if (baca_tombol >= 80 &&
baca_tombol <= 100) {
    baca_tombol = 'OK';
  }
  if (baca_tombol <= 20) {
    baca_tombol = 'EXIT';
  }


        }
```

IV. PENGUJIAN DAN ANALISIS

A. Pengujian *Hardware*

Pada bab ini dibahas mengenai pengujian dan analisa dari sistem yang telah dibuat. Pengujian meliputi implementasi algoritma PID pada *hardware* yang bertujuan untuk mengetahui respon sistem pada saat terjadi *drifting.* Pengujian *hardware* pada lintasan datar meliputi pengujian manual, pengujian menggunakan kontrol PD, PI, dan PID.

1. Pengujian Sistem Manual

Pengujian menggunakan sistem manual diujicobakan karena untuk melihat seberapa besar tingkat *drifting* yang dapat dilihat dari data sensor *accelero x* dan *gyroscope z* yang di kendalikan oleh pengemudi. Selanjutnya akan diketahui waktu untuk kembali ke posisi lurus.

Tabel 6. Pengujian sistem manual

Accelero **x**	***Gyroscope*** **z**	**Servo**	**Rem Kiri**	**Rem Kanan**
`7	0	1695	1500	1500
6	0	1700	1500	1500
37	-5	1611	1500	1500
-13	-6	1607	1500	1500
-89	-67	1450	1500	1500
-39	-165	1450	1500	1500
-81	-126	1450	1500	1500
36	-113	1450	1500	1500
-86	-111	1450	1500	1500
-85	-107	1450	1500	1500
-8	-144	1450	1500	1500
43	-178	1450	1500	1500
-234	-161	1450	1500	1500
-242	-129	1450	1500	1500
-353	-85	1450	1500	1500
-217	-62	1450	1500	1500
-311	-60	1450	1500	1500
-98	-152	1450	1500	1500
-143	-224	1450	1500	1500
-151	-238	1450	1500	1500
-63	-217	1450	1500	1500
-93	-218	1450	1500	1500
-127	-168	1450	1500	1500
-56	-90	1450	1500	1500
-221	-74	1450	1500	1500
-37	-112	1450	1500	1500
-27	-101	1450	1500	1500
10	-105	1450	1500	1500
-109	-50	1450	1500	1500
18	-73	1450	1500	1500
123	-47	1603	1500	1500
-48	-6	1610	1500	1500
25	-7	1609	1500	1500
28	0	1603	1500	1500
41	62	1716	1500	1500
31	1	1635	1500	1500
16	-3	1605	1500	1500
19	1	1603	1500	1500
11	0	1716	1500	1500
14	0	1700	1500	1500

Tabel 6 diatas merupakan data dari prototype mobil penyeimbang mobil pada saat *drifting*. Data yang diperolah antara lain data nilai *accelero x, gyroscope z,*servo *steering*, servo rem kiri, dan servo rem kanan. Nilai *accelero* akan berubah-ubah sesuai pergerakan mobil kekiri atau kekanan. Untuk nilai *accelero* nilai pada

saat diam tidak pasti karena tergantung pada kemiringan dan pergeseran mobil. Untuk nilai *gyroscope* yaitu 0 karena nilai *gyroscope* akan berubah ketika ada hentakan atau perubahan gerak secara mendadak. Ketika prototype mobil berjalan lurus, nilai dari servo *steering* yaitu 1700. Sedangkan untuk nilai servo rem kiri dan servo rem kanan dalam keadaan diam yaitu 1500.

Pada tabel 6 nilai *accelero x* yaitu 7, 6, 37, -13 dan seterusnya menunjukkan pergerakan mobil dan pergeseran mobil terhadap sumbu x. Nilai *gyroscope z* yaitu 0, 0, -5, -6, dan seterusnya menunjukkan perubahan gerak mobil secara mendadak. Nilai *gyroscope* bernilai negatif jika pergerakan mobil berputar berlawanan dengan arah jarum jam dan akan bernilai positif jika searah jarum jam. Servo *steering* pada tabel 6 diperoleh nilai 1695, 1700,1611,1607, dan seterusnya. Saat nilai servo 1700, servo *steering* bergerak lurus. Jika nilai servo *steering* dibawah 1700, mobil berbelok ke kiri dan jika servo bernilai diatas 1700, maka mobil berbelok ke kanan.

2. Pengujian Sistem Menggunakan Kontrol PD

Pengujian menggunakan algoritma PD dilakukan untuk membandingkan hasil *output* yang di dapat dari pengambilan data sensor pada prototype mobil RC. Adapun pengaturan nilai Kp = 42 dan Kd = 12. Berikut tabel 7 menjelaskan respon sensor dan *steering* hasil pengujian menggunakan kontrol PD.

Tabel 7. Pengujian sistem menggunakan kontrol PD

***Accelero* x**	***Gyroscope* z**	**Servo**	**Rem Kiri**	**Rem Kanan**
12	0	1697	1500	1500
16	1	1700	1500	1500
7	1	1700	1500	1500
0	-1	1689	1500	1500
-4	4	1685	1500	1500
66	76	1775	1500	1500
14	107	1763	1500	1500
51	126	1784	1500	1500
-61	151	1729	1500	1500
210	143	1341	1300	1500
82	98	1315	1300	1500
154	135	1315	1300	1500
150	128	1329	1300	1500
2	78	1745	1500	1500
-36	168	1708	1500	1500
59	294	1737	1300	1500
108	381	1728	1300	1500
78	400	1786	1300	1500
84	621	1733	1300	1500
15	551	1742	1300	1500
-135	462	1730	1500	1500
-122	406	1737	1500	1500
-84	304	1710	1500	1500
8	221	1695	1500	1500
-1	42	1703	1500	1500
4	2	1700	1500	1500
2	0	1696	1500	1500
1	0	1700	1500	1500
1	0	1722	1500	1500
1	0	1701	1500	1500

Tabel 7 diatas menunjukkan hasil pengambilan data sistem automatis menggunakan kontrol PD. Data *gyroscope z* menunjukkan nilai 4, 76, 107, dan seterusnya, menunjukkan bahwa prototype mobil terjadi *drifting* ke arah kiri atau searah dengan jarum jam. Pada saat terjadi *drifting* berlebihan, servo rem akan membantu mengurangi putaran roda yang berlebihan agar posisi mobil cepat kembali lurus. Pada saat rem kiri aktif maka servo akan bernilai 1300.

3. Pengujian Sistem Menggunakan Kontrol PI

Pengujian menggunakan algoritma PI dilakukan untuk membandingkan hasil *output* yang di dapat dari pengambilan data sensor pada prototype mobil RC. Adapun pengaturan nilai Kp = 42 dan Ki = 22. Berikut tabel 8 menjelaskan respon sensor dan *steering* hasil algoritma PI.

Tabel 8. Pengujian sistem menggunakan kontrol PI

Accelero x	*Gyroscope* z	Servo	Rem Kiri	Rem Kanan
10	0	1696	1500	1500
6	2	1671	1500	1500
-19	-1	1710	1500	1500
29	-4	1755	1500	1500
23	29	1800	1500	1500
99	58	1800	1300	1500
35	84	1800	1500	1500
61	78	1800	1500	1500
6	81	1800	1500	1500
83	104	1800	1300	1500
144	109	1800	1300	1500
121	48	1713	1	1500
-86	-71	1800	1500	1700
-7	-26	1699	1500	1500
-12	2	1700	1500	1500
25	-2	1693	1500	1500
4	-1	1689	1500	1500
28	-1	1686	1500	1500
21	-1	1626	1500	1500
10	-10	1737	1500	1500
25	-2	1672	1500	1500
-9	-1	1716	1500	1500
10	0	1698	1500	1500
39	2	1701	1500	1500
17	0	1693	1500	1500
1	0	1699	1500	1500

Tabel 8 menunjukkan hasil data pengujian sistem menggunakan kontrol PI. Dapat terlihat dari data *gyroscope z*, bahwa ketika terjadi *drifting* sistem dengan cepat memperbaiki *error*. Akan tetapi terjadi *overhoot* yang ditunjukkan dari nilai *gyroscope z* yang awalnya positif berubah ke nilai negatif.

4. Pengujian Sistem Menggunakan Kontrol PID

Pengujian menggunakan algoritma PID dilakukan untuk membandingkan hasil *output* yang di dapat dari pengambilan data sensor pada prototype mobil RC. Adapun pengaturan nilai Kp = 42, Ki = 22, dan Kd = 12. Berikut tabel 9 menjelaskan respon sensor dan *steering* hasil pengujian sistem menggunakan kontrol PID.

Tabel 9. Pengujian sistem menggunakan kontrol PID

Accelero x	*Gyroscope* z	Servo	Rem Kiri	Rem Kanan
5	0	1696	1500	1500
3	0	1696	1500	1500
4	0	1700	1500	1500
24	-3	1691	1500	1500
31	-52	1663	1500	1500
-25	-26	1639	1500	1500
27	-44	1688	1500	1500
15	-33	1675	1500	1500
11	-15	1695	1500	1500
-28	-23	1622	1500	1500
-84	-83	1450	1500	1700
-53	-121	1450	1500	1500
-66	-121	1450	1500	1500
-179	-199	1562	1500	1700
-276	-165	1611	1500	1700
114	-137	1580	1500	1700
-81	-130	1460	1500	1700
-9	-118	1487	1500	1500
-158	-120	1575	1500	1700
-23	-115	1513	1500	1500
-67	-91	1508	1500	1500
43	-111	1496	1500	1500
3	-111	1480	1500	1500
34	-86	1676	1500	1500
42	-13	1697	1500	1500
-46	-3	1701	1500	1500
9	-2	1697	1500	1500
16	5	1700	1500	1500
-2	-1	1688	1500	1500
7	0	1696	1500	1500
7	0	1694	1500	1500
7	0	1693	1500	1500
5	0	1697	1500	1500
7	0	1697	1500	1500

Dari tabel 9 diatas, dapat dilihat dari data nilai *gyroscope z* bernilai negatif menandakan gerak mobil berputar berlawanan arah jarum jam.

V. PENUTUP

A. Kesimpulan

Berdasarkan hasil perancangan dan pengujian serta analisis data dari sistem penyeimbang mobil pada saat *drifting* yang dibahas pada penulisan laporan tugas akhir ini, dapat diambil beberapa kesimpulan yang berkaitan dengan hasil analisis data yang mengacu kepada tujuan perancangan dan implementasi prototype penyeimbang mobil pada saat *drifting* ini.

1. Respon pengujian sistem secara manual membutuhkan waktu yang lama untuk membuat mobil stabil kembali karena tidak dibantu dengan *steering* dan rem automatis. Pengujian sistem secara manual membutuhkan waktu 2028 ms.
2. Respon pengujian sistem menggunakan kontrol PD membutuhkan waktu 1144 ms untuk kembali ke posisi stabil. Sistem menggunakan kontrol PD, ketika terjadi *drifting* langsung menurunkan *error* namun responnya lambat dan masih terjadi osilasi.
3. Respon pengujian sistem menggunakan kontrol PI membutuhkan waktu 1040 ms. Sistem mengunakan kontrol PI memiliki respon yang cepat untuk menurukan *error*, namun bisa terjadi *overshoot* dan osilasi.
4. Respon pengujian sistem menggunakan kontrol PID membutuhkan waktu 1300 ms untuk kembali ke posisi stabil. Pengujian menggunakan control PID relatif tidak terjadi osilasi namaun membutuhkan waktu agak lama dibandingkian menggunakan kontrol PI dan PD.

B. Saran

Untuk pengembangan dan peningkatan lebih lanjut dari perancangan dan implementasi prototype penyeimbang mobil pada saat *drifting* ini ada beberapa point yang perlu diperhatikan dalam perancangannya.

1. Untuk pengembangan lebih lanjut, sebaiknya menggunakan mekanik yang lebih baik khususnya pada bagian *steering*.
2. Sensor *gyroscope* memiliki nilai bias, sebaiknya ditambahkan filter.

DAFTAR PUSTAKA

[1] Hidayat, Rahmat, Rancang Bangun Sistem Penstabil Kamera Untuk Foto Udara Berbasis Wahana Udara Quadcopter

[2] Kompasiana, teknologi mengenal sistem keselamatan pada mobil. [Online] Avalaible:teknologi.kompasiana.com/otomotif/2014/08/14/mengenal-sistem-keselamatan-pada-mobil-680181.html

[3] Accelerator and gyroscope. [Online]. Available:http://siska-theanalyst.blogspot.com/2012/05/accelerator-and-gyroscop.html

[4] Keseimbangan Robot Menggunakan Sensor *Gyro* GS-12 dan *Accelerometer* DE-ACCM3D

[5] Axis gyroscope-accelerometer [Online]. Available:http://www.vcc2gnd.com/2014/02/mpu-6050-6-axis-gyroscope accelerometer.html