# BAB IV
# PENGUJIAN DAN ANALISA

## 4.1. Pendahuluan

Pada bab ini akan dilakukan pengujian perangkat lunak dan perangkat keras. Selanjutakn akan diadakan analisis terhadap hasil perancangan untuk melihat apakah hasil yang didapatkan sesuai dengan kinerja yang diinginkan. Prosedurnya akan dilakukan menjadi 2 tahapan yaitu pengujian perangkat lunak dan pengujian peragkat keras. Hasil dari pengujian dan analisa sementara yang telah dilakukan dapat dilihat pada sub bab selanjutnya.

## 4.2. Pengujian *Hardware*

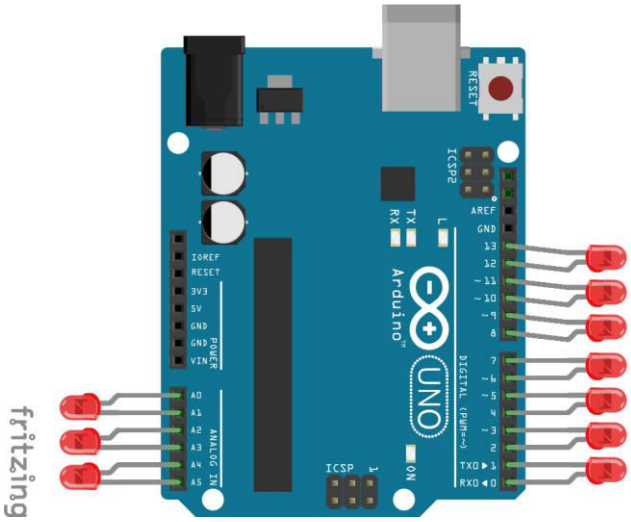
Pengujian perangkat keras dilakukan pada seluruh perangkat keras yang digunakan pada pembuat robot *balancing* atau robot pendulum terbalik, perangkat keras atau modul yang akan diuji yaitu :

1. Mikrokontroler Arduino Uno
2. Modul Sensor MPU 6050
3. Modul Bluetooth HC-05
4. Modul *Driver* Motor DC

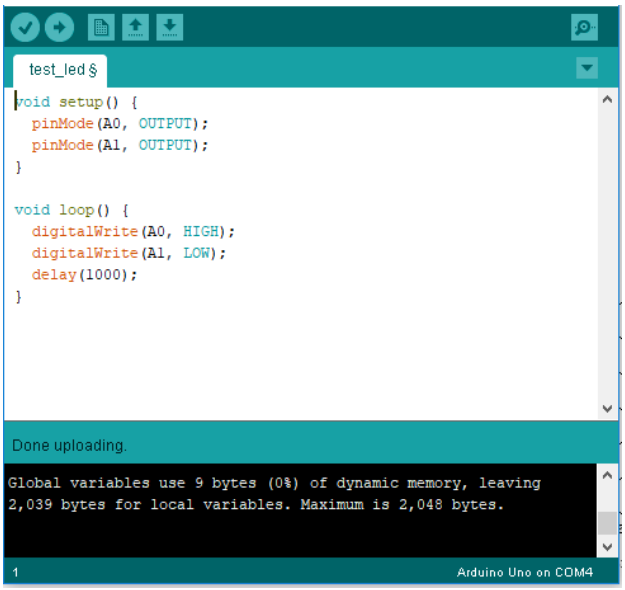
### 4.2.1. Pengujian Mikrokontroler Arduino Uno

Pada pengujian mikrokontroler Arduino Uno terbagi menjadi dua tahap. Pertama, menggunakan bantuan indikator LED untuk mendeteksi pin pin yang akan digunakan pada setiap modul apakah pin pin tersebut sudah bekerja dengan semestinya jika diberi program atau perintah untuk mengaktifkan LED dan apakah ada kerusakan pada salah satu pin mikrokontroler tersebut. Pengujian bisa dilihat pada gambar 4.1 dan 4.2. Selain mendeteksi kerusakan pada pin mikrokontroler yang akan digunakan, pengujian juga dilakukan menggunakan multimeter untuk mengetahui berapa besar tegangan yang dikeluarkan pada pin VCC dan GND apakah sudah seusai atau belum dengan tegangan yang

dibutuhkan untuk mengaktifkan sensor, bluetooth dan driver motor DC yang digunakan.




**Gambar 4.1.** Pengujian Arduino menggunakan LED




**Gambar 4.2.** *Code* untuk pengujian Arduino

### 4.2.2. Pengujian Modul Sensor MPU 6050

Pengujian pada sensor MPU 6050 bertujuan untuk melihat bagaimana kinerja dari sensor ini sebagai sensor yang berfungsi untuk menentukan orientasi suatu benda berdasarkan ketepatan momentum sudut berjalan dengan baik. Pengujian dilakukan dengan menyiapkan alat dan bahan telebih dahulu. Berikut alat dan bahan yang digunakan:

Alat dan Bahan :

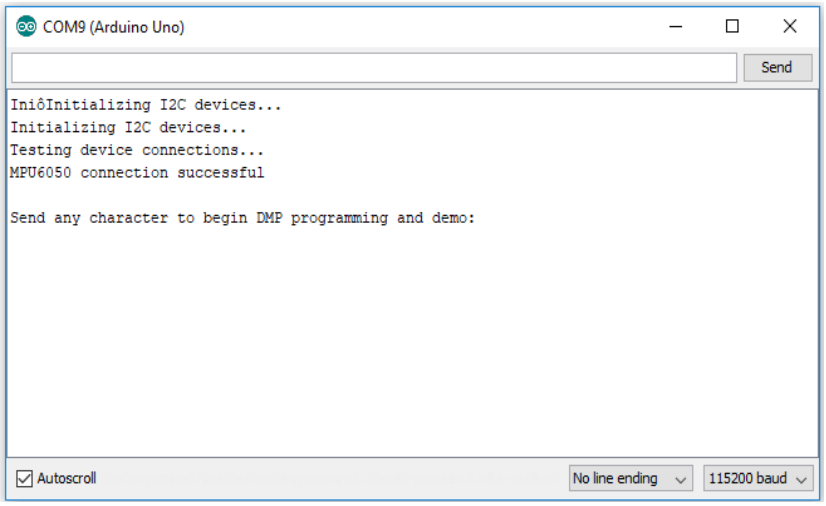
1. Arduino Uno.
2. Sensor MPU 6050 *Accelerometer* and *Gyrometer* 6 DOF.
3. Kabel USB serial untuk mendownloadkan program ke Arduino.
4. Laptop atau PC

Jika alat dan bahan sudah dipersiapkan, berikutnya melakukan instalasi komponen Arduino dan Sensor seperti konfigurasi pada tabel 5.

Jika konfigurasi seperti tabel 5 sudah terhubung dengan benar, berikutnya ke tahap pengetesan I2C hal ini dilakukan agar dapat mengetahui sudahkah sensor terhubung ke Arduino. Langkah untuk pengecekannya yaitu sebagai berikut :

1. Buka aplikasi IDE Arduino
2. Salin kode pengujian sensor sesuai dengan lampiran 1 program
3. Mengupload kode tersebut kedalam Arduino.
4. Setelah selesai, tekan CTRL+SHIFT+M untuk buka *Serial Monitoring***.**

   Jika koneksi I2C dengan sensor terhubung dengan baik maka akan terlihat seperti pada gambar berikut :

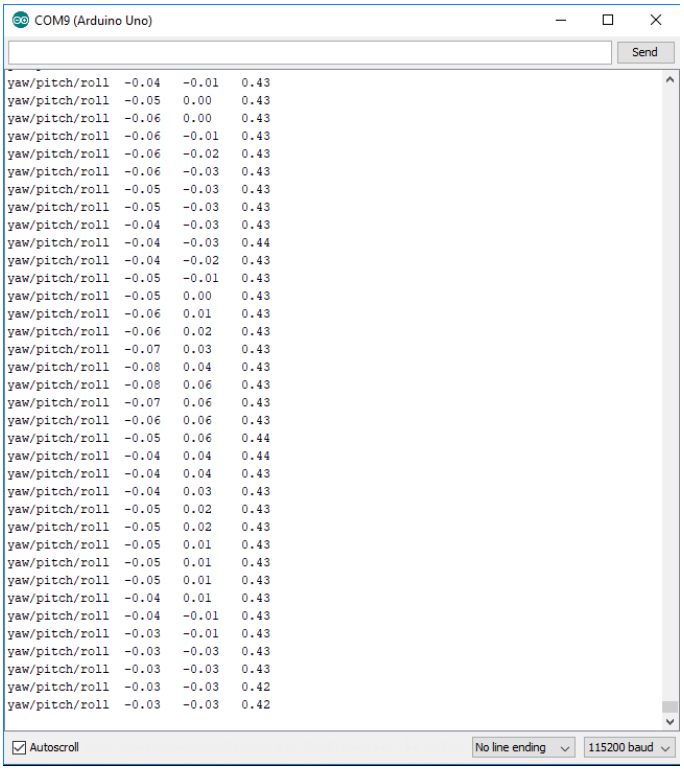



**Gambar 4.3.** Tampilan koneksi I2C terhadap sensor MPU 6050

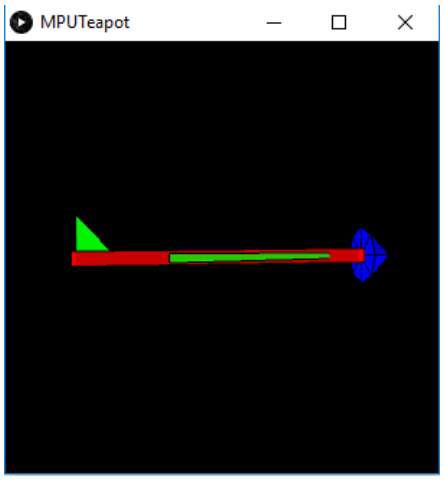
Setelah pengecekan I2C sudah benar selanjutnya melakukan tes modul sensor MPU 6050 apakah data yang yang diberikan oleh sensor benar dan tidak mengalami *error* pada saat pengambilan data secara *realtime* di *Serial Monitoring***.** Untuk menampilkan data yang diberikan oleh sensor yaitu dengan mengetikkan huruf apapun pada *Serial Monitoring* seperti yang terlihat pada Gambar 4.3 dan menekan *ENTER* **.**

MPUTeapot merupakan aplikasi *open source* yang digunakan untuk menampilkan data dari sensor MPU 6050 menjadi bentuk visual. Berikut adalah tampilan data yang dihasilkan sensor MPU 6050 pada *Serial Monitoring* dan Visualisasi dari aplikasi MPUTeapot :

Kondisi sensor datar terhadap lantai :



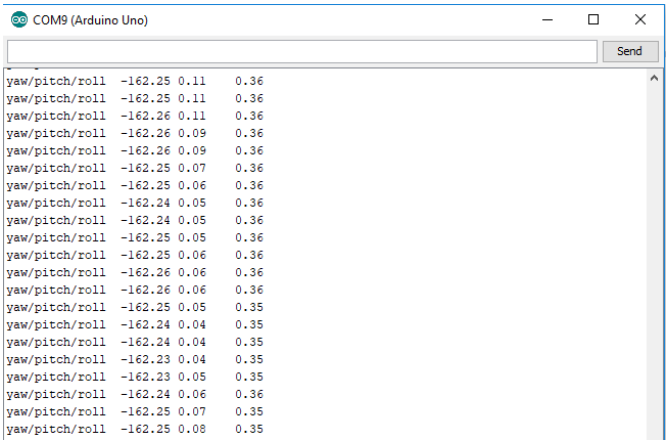

(a)



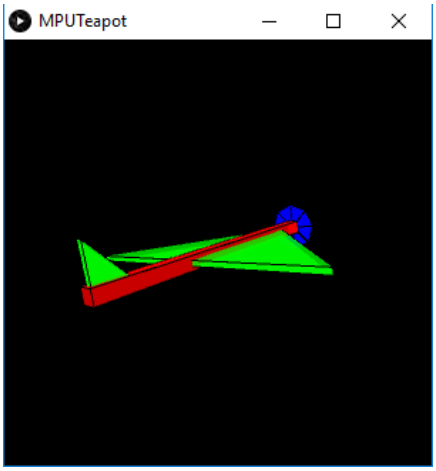

(b)

**Gambar 4.4.** Tampilan kondisi sensor dalam keadaan seimbang (a*) Serial Monitor* (b) Visualisasi

Kondisi sensor dibelok ke kanan :



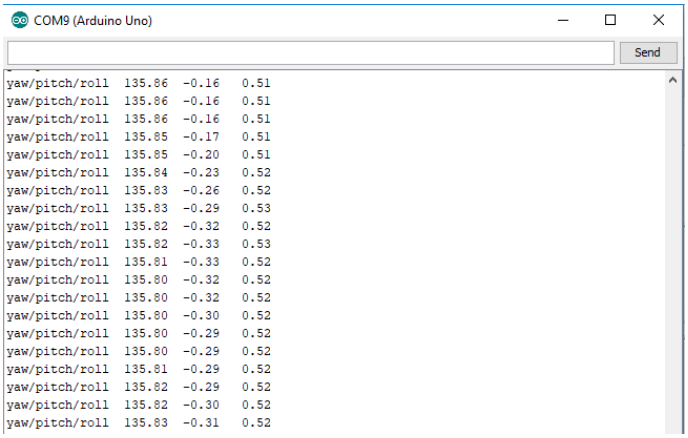

(a)



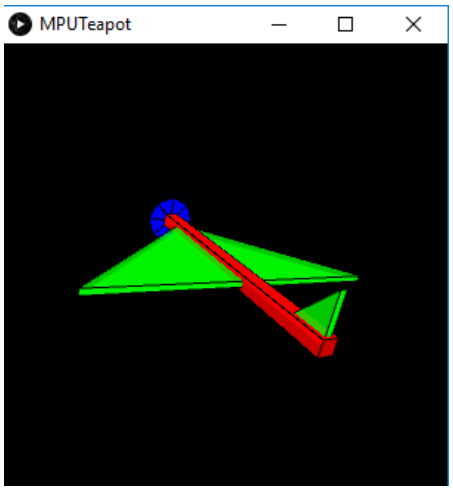

(b)

**Gambar 4.5.** Tampilan kondisi sensor dalam keadaan belok ke kanan (a) *Serial Monitor* (b)Visualisasi

Kondisi sensor dibelok ke kiri :



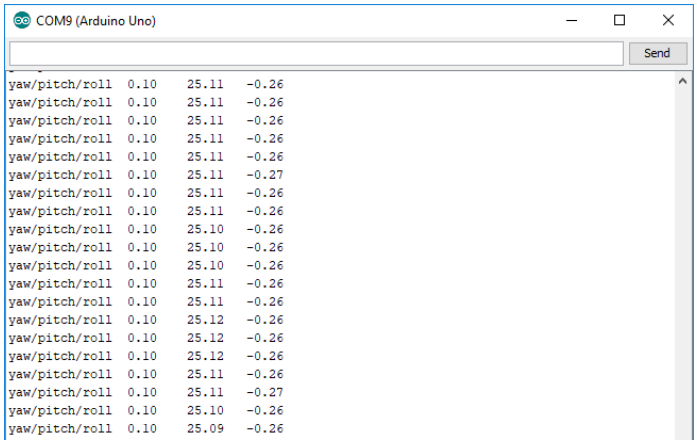

(a)



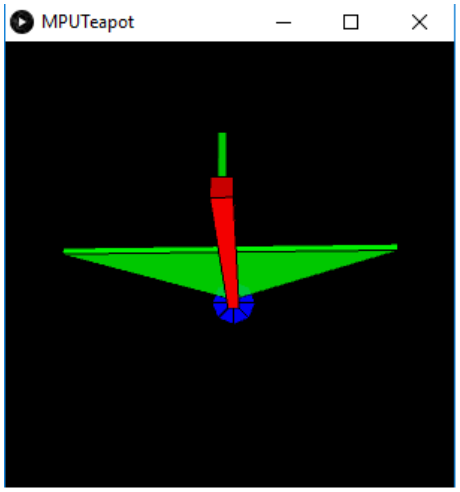

(b)

**Gambar 4.6.** Tampilan kondisi sensor dalam keadaan belok ke kiri (a) *Serial Monitor* (b) Visualisasi

Kondisi sensor miring kedepan :



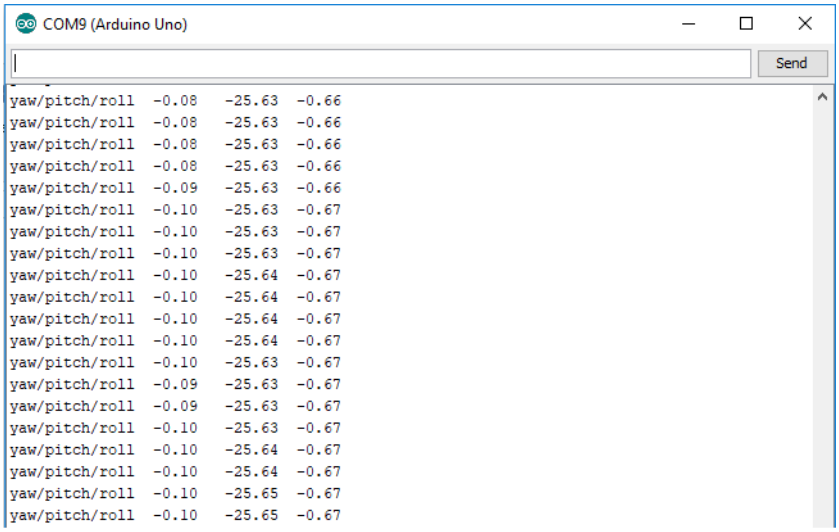

(a)



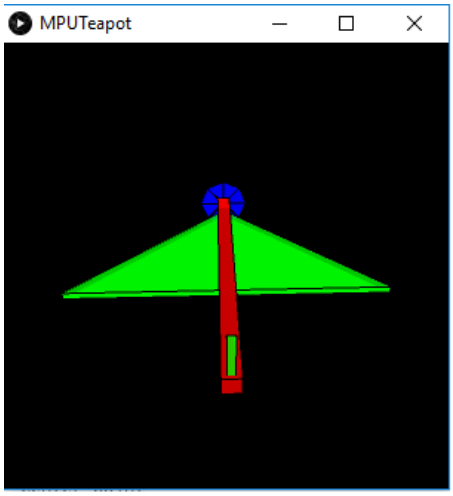

(b)

**Gambar 4.7.** Tampilan kondisi sensor dalam keadaan miring ke depan (a) *Serial Monitor* (b) Visualisasi

Kondisi sensor miring ke belakang :




(a)




(b)

**Gambar 4.8.** Tampilan kondisi sensor dalam keadaan miring ke belakang (a)*Serial Monitor* (b) Visualisasi

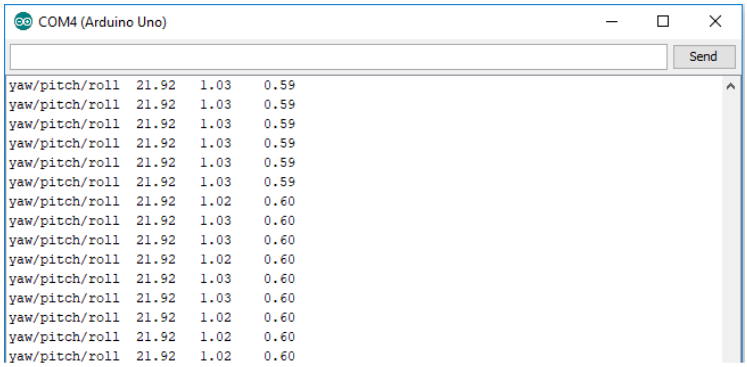
*Yaw* adalah gerakan menggeleng atau bergerak kekanan dan kekiri., *yaw* bergerak pada sumbu vertikal. *Pitch* adalah gerakan menganggu keatas maupun

kebawah, *pitch* bergerak pada sumbu lateral. *Roll* adalah gerakan berguling, *roll* bergerak pada sumbu longitudinal.

Pengujian selanjutnya yaitu membandingkan kondisi sudut sensor yang diukur menggunakan busur dengan data yang ditampilkan pada *Serial Monitor* untuk mendapatkan seberapa besar nilai kesalahan pada sensor.

Pengujian dengan sudut $0^o$:

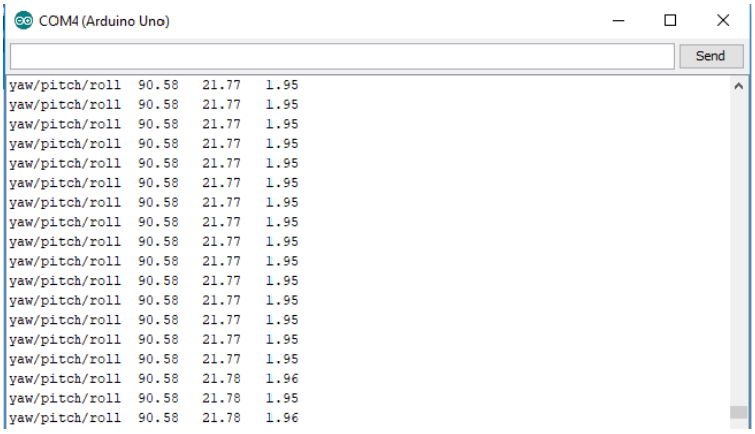
Pada gambar 4.9 dapat dilihat bahwa pada kondisi sensor diberikan sudut $0^o$ (datar terhadap lantai) yang diukur menggunakan busur mendapatkan hasil rata-rata nilai $1{,}03^o$ . Jadi pada pengujian sudut $0^o$ sensor memiliki nilai kesalahan sebesar $1{,}03^o$.




**Gambar 4.9.** Sensor pada sudut $0^o$

Pengujian dengan sudut $20^o$:

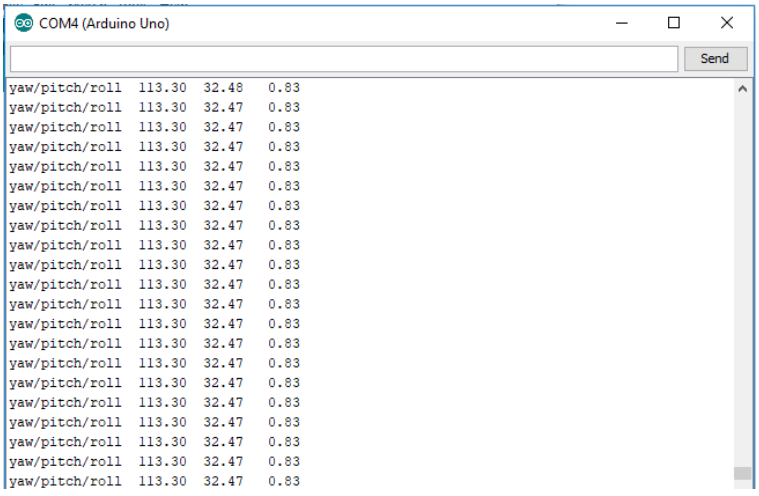
Pada gambar 4.10 dapat dilihat bahwa pada kondisi sensor diberikan sudut $20^o$ yang diukur menggunakan busur mendapatkan hasil rata-rata nilai $21{,}77^o$ . Jadi pada pengujian sudut $20^o$ sensor memiliki nilai kesalahan sebesar $1{,}77^o$.




**Gambar 4.10.** Sensor pada sudut 20$^{o}$

Pengujian dengan sudut 30$^{o}$:

Pada gambar 4.11 dapat dilihat bahwa pada kondisi sensor diberikan sudut 30$^{o}$ yang diukur menggunakan busur mendapatkan hasil rata-rata nilai 32,47$^{o}$ . Jadi pada pengujian sudut 30$^{o}$ sensor memiliki nilai kesalahan sebesar 2,47$^{o}$.




**Gambar 4.11.** Sensor pada sudut 30$^{o}$

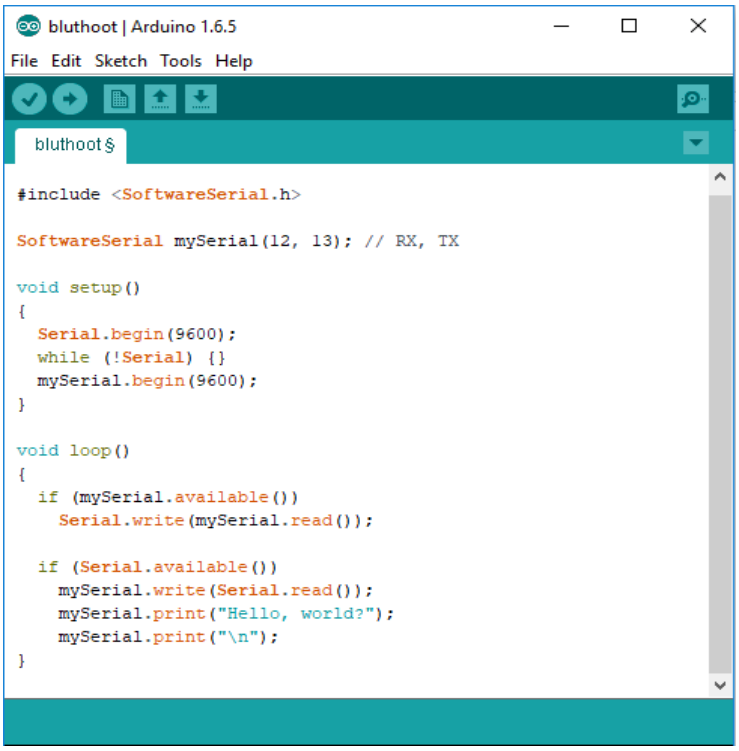
Jadi kondisi sudut sensor yang diukur menggunakan busur dengan data yang ditampilkan pada *Serial Monitor* mendapatkan rentang nilai kesalahan sebesar 1,03$^{o}$ sampai dengan 2,47$^{o}$.

### 4.2.3. Pengujian Modul Bluetooth HC-05

1). Pengujian dengan *Serial Monitoring*

Pada pengujian modul bluetooth HC-05 dilakukan untuk mengetahui apakah bluetooth terkoneksi dengan baik. Sehingga saat nanti digunakan untuk mengirimkan data yang dihasilkan oleh sensor dapat ditampilkan pada *Serial Monitoring* yang tedapat pada IDE Arduino. Berikut langkah yang dilakukan :

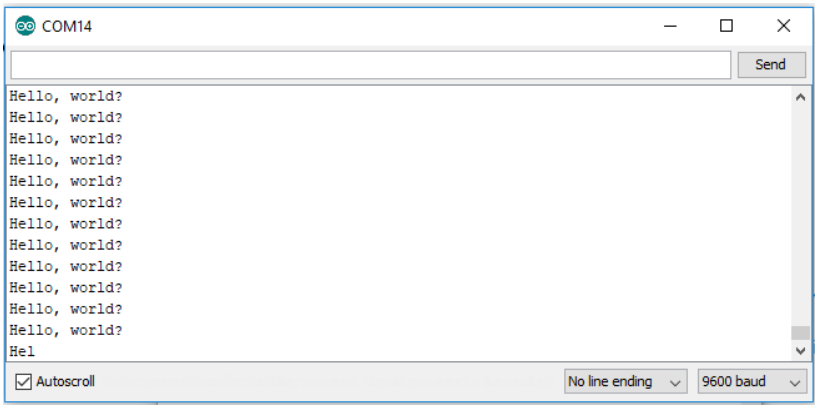
1. Bentuk rangkaian bluetooth HC-05 seperti pada Gambar 3.6
2. *Pairing* bluetooth HC-05 ke PC
3. Temukan pada *port* COM berapa modul terhubung dan catat
4. Mengupload *code* yang terlihat pada Gambar 4.12 melalui usb *serial*.




**Gambar 4.12.** *Code* Pengujian Bluetooth HC-05

Setelah langkah-langkah diatas dilakukan, selanjutnya menampilkan hasil dari *code* program yang diupload ke mikrokontroler Arduino untuk membuktikan apakah modul bluetooth HC-05 berfungsi dengan baik dalam mengirimkan data. Koneksikan bluetooth ke PC, selanjutnya pada IDE Arduino ubah port

sebelumnya yang digunakan usb serial menjadi port yang digunakan bluetooth yang sebelumnya sudah dicatat. Kemudian membuka *Serial Monitoring* untuk mendapatkan hasil tampilan dari code program. Jika modul bluetooth HC-05 bekerja dengan baik maka akan menampilkan hasil seperti yang terlihat pada Gambar 4.13, dimana data yang dikeluarkan akan bersifat kontinu.



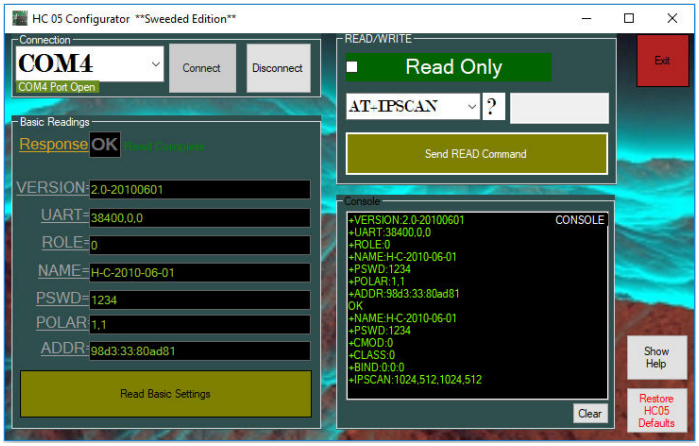

**Gambar 4.13.** Hasil Pengujian bluetooth HC-05

2). Pengujian dengan AT mode

Pada pengujian ini kita menggunakan program GUI HC-05. Tu. Dengan cara berikut :

1. Hubungkan pin VCC, EN, 34 ke +5V Arduino dan GND ke GND arduino
2. Hubungkan pin RX dan TX ke Arduino RX dan TX.
3. Dalam kondisi Arduino tidak tidak terkoneksi ke PC. Tahan push botton pada Bluetooth kemudian hubungkan usb kabel Arduino ke PC kemudian lepaskan.
4. Jalankan program GUI HC-05. konekan port dimana Arduino terkoneksi.
5. Lakukan berbagai macam *commend* seperti terlihat pada gambar 4.14.

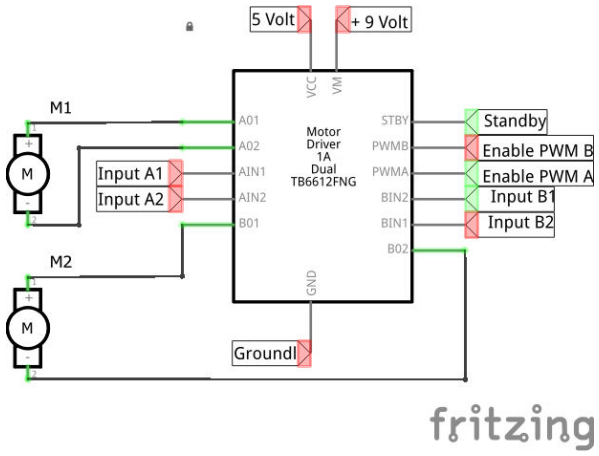
Pengujian ini bertujuan untuk melihat atau mengetahui apakah modul Bluetooth HC-05 dapat melakukan *transmite* dan *receive* data dengan malakukan beberapa perintah yang telah terdapat pada GUI. Apabila setelah dilakukan beberapa perintah dan mendapatkan hasilnya menandakan modul Bluetooth bekerja dengan baik. Dan apabila tidak terjadi apapun maka telah terjadi kesalahan pada pemasang modul atau pun modul dalam kondisi rusak.




**Gambar 4.14.** Pengujian dengan Gui HC-05

### 4.2.4. Pengujian Driver Motor DC

Pengujian driver motor TB6612FNG dilakukan untuk mengetahui apakah IC ini bisa bekerja dengan semestinya. Pengujian driver ini dilakukan dengan cara menguji *Pulse Width Modulation* (PWM) pada *driver* motor. Rangkaian ini terdiri dari IC TB6612FNG. Berikut merupakan rangkaian dari driver TB6612FNG :




**Gambar 4.15.** *Driver* TB6612FNG

Metode pengujian dilakukan dengan memberikan memberikan nilai *duty cycle* PWM secara bertahap mulai dari nilai yang terkecil sampai terbesar. Pengujian dilakukan dengan cara membandingkan hasil pengukuran langsung dengan alat ukur multimeter dengan hasil perhitungan. Pengujian dilakukan sebanyak 5 kali, dengan supplay 12 volt.

**Tabel 8**

Pengujian PWM Pada Motor DC

| No | Supply (V) | Duty Cycle PWM (%) | Keluaran | | |
|---|---|---|---|---|---|
| | | | Hasil PWM | Hasil Perhitungan Tegangan (V) | Hasil Pengujian Tegangan (V) |
| 1 | 12 | 0 | 0 | 0 | 0 |
| 2 | | 25 | 63 | 3 | 2,3 |
| 3 | | 50 | 127 | 6 | 5,4 |
| 4 | | 75 | 193 | 9 | 7,8 |
| 5 | | 100 | 255 | 12 | 9,7 |

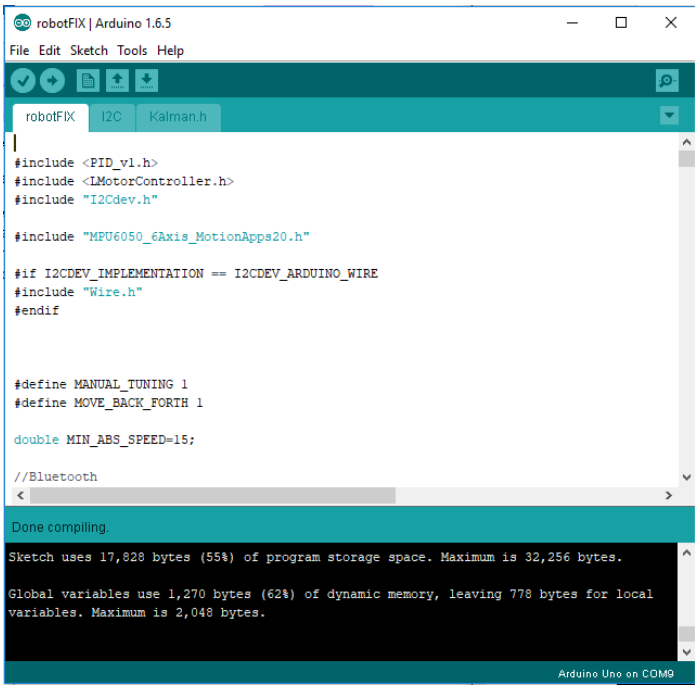
Dari hasil pengujian dapat dinyatakan bahwa semakin besar *duty cycle* pada sinyal PWM yang diberikan, maka tegangan yang masuk ke motor akan semakin besar yang menyebabkan kecepatan motor meningkat. Begitu juga sebaliknya, semakin kecil *duty cycle* pada sinyal PWM, maka tegangan yang masuk ke motor akan semakin kecil yang menyebabkan kecepatan motor melambat.

## 4.3. Pengujian *Software*

### 4.3.1. Pengujian Program Pada Arduino Board

Pengujian ini dilakukan dengan cara mengkompile program yang telah dibuat sebelum didownload kedalam mikrokontroler Arduino. Indikator untuk

melihat keberhasiland dari pengujian ini adalah tidak menghasilkan error maupun warning. Hasil *compile* program dapat dilihat pada gambar 4.16.



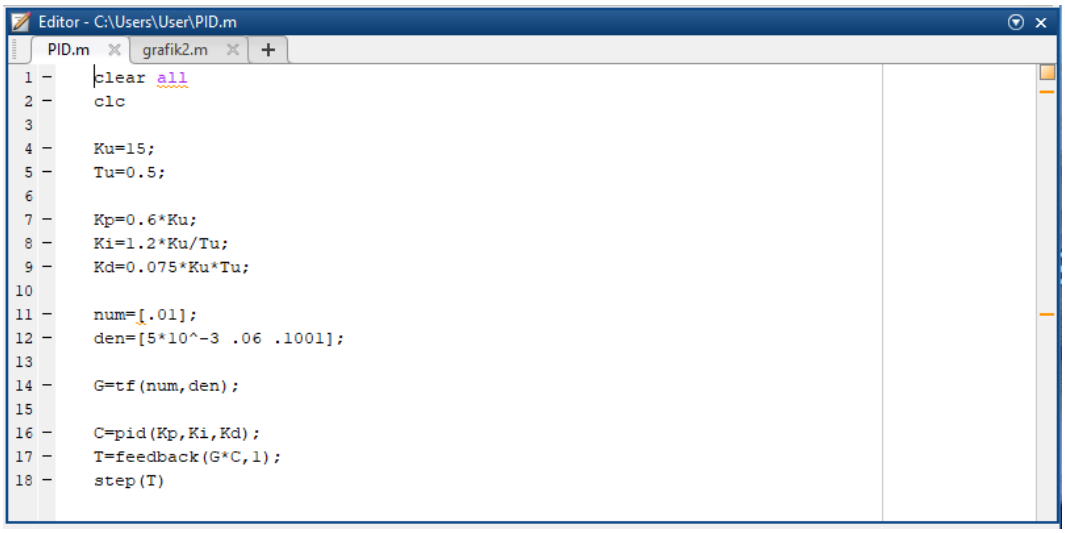

**Gambar 4.16.** Hasil Kompile Program dengan Arduino IDE

Berdasarkan hasil pengujian yang dilakukan pada program menggunakan Arduino IDE, hasil yang didapatkan setelah melakukan *compile* ternya tidak menghasilakan *error* maupun *warning* sehingga dapat dikatakan kalau program telah berhasil dan *resource* program masih dalam keadaan cukup untuk diembedkan kedalam mikrokontroler Arduino.

### 4.3.2. Pengujian Pada Matlab 2017a

Pada pengujian ini akan ditampilkan hasil dari program yang ditanamkan kedalam mikrokontroler dengan cara men-*capture* data hasil pengujian dengan settingan PID Ziegler-Nichols yang sudah di set pada program sebelum diembedkan kedalam mikrokontroler Arduino Uno. Adapun program yang digunakan untuk menampilkan data yang mengkoneksikan robot dengan PC adalah Matlab 2017a.

Pengujian dilakukan dengan menentukan nilai $K_u$ dan $T_u$ dengan metode simulasi pada program Matlab untuk mendapatkan bentuk grafik yang baik. Selanjutnya nilai yag didapatkan pada simulasi pada program Matlab tadi diimplementasikan pada robot. Robot yang telah diinputkan nilai tadi dilakukan pengujian dengan menghubungkan robot dengan Matlab untuk mendapatkan bentuk grafik dari pwm dan sudut yang dihasilkan. Berikut adalah kode program dari simulasi Matlab terlihat pada gambar 4.17.

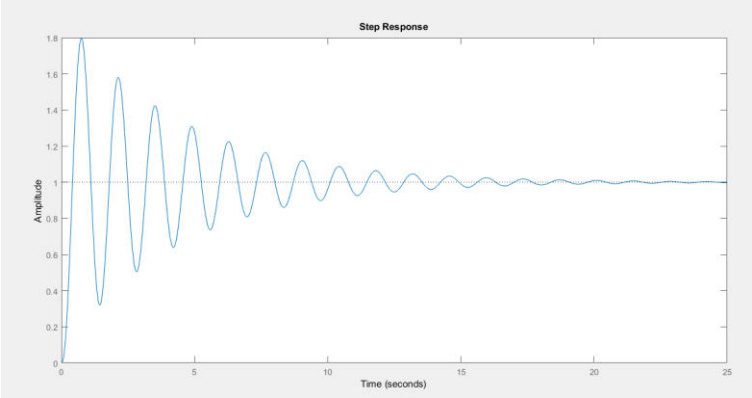



**Gambar 4.17.** Kode program pada simulasi Matlab

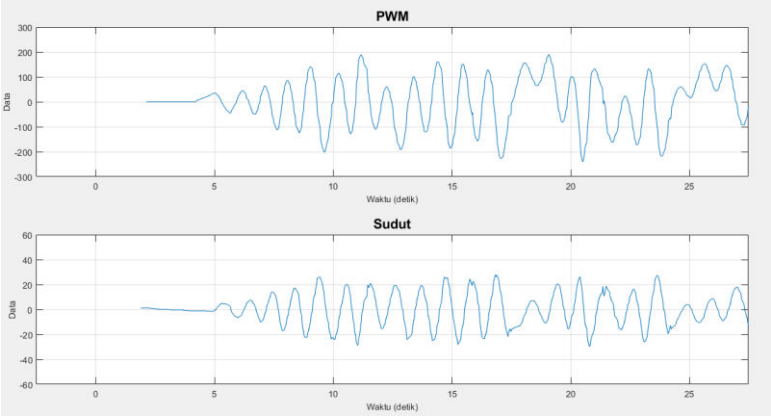
**1. Percobaan dengan nilai parameter $K_u$ =5 , ada tiga perobaan;**

*** Percobaan 1A, $K_u$=5 dan $T_u$=0.05.**

Berikut tampilan grafik pada Matlab 2017a:




(a)



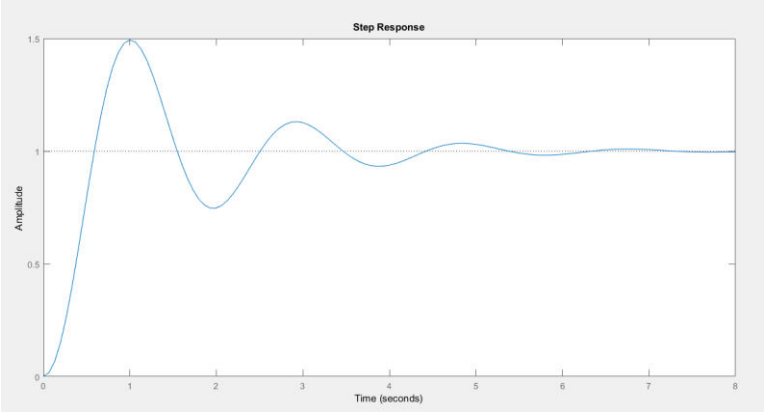

(b)

**Gambar 4.18.** Grafik hasil dengan inputan $K_u$=5 dan $T_u$=0.05: (a)Simulasi (b)Robot

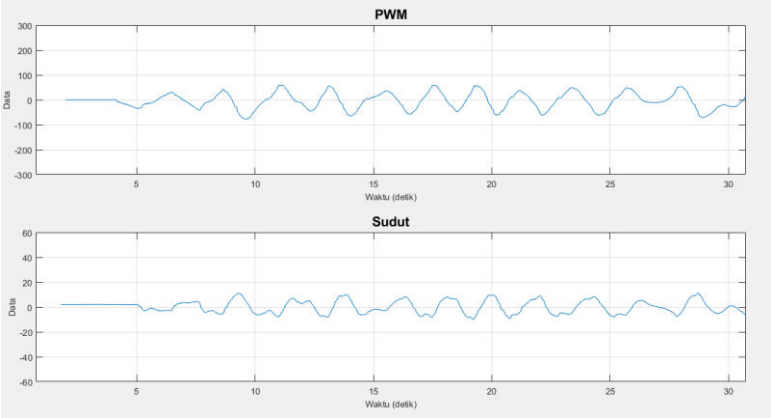
Pada gambar 4.18 didapatkan kondisi saat robot dijalankan selama 25 detik didapatkan rentang besaran sudut antara -30 s/d 30 derajat menghasilkan besaran pwn -240 s/d 200 dan didapatkan grafik pwm dan sudut yang tidak beraturan.

*** Percobaan 1B, $K_u$=5 dan $T_u$=0.1.**

Berikut tampilan grafik pada Matlab 2017a:




(a)



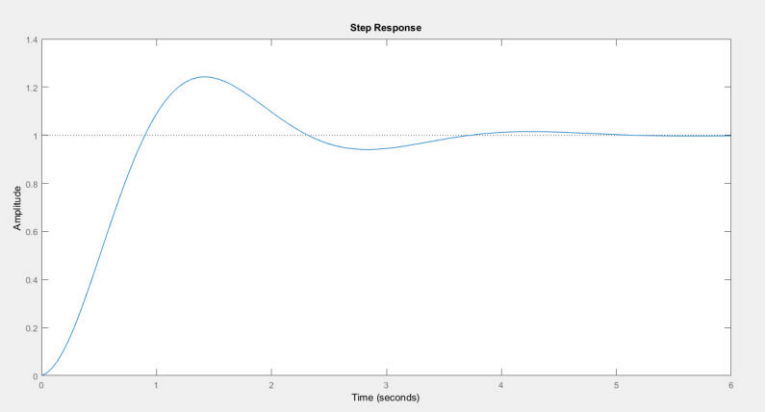

(b)

**Gambar 4.19.** Grafik hasil dengan inputan $K_u$=5 dan $T_u$=0.1: (a)Simulasi (b)Robot

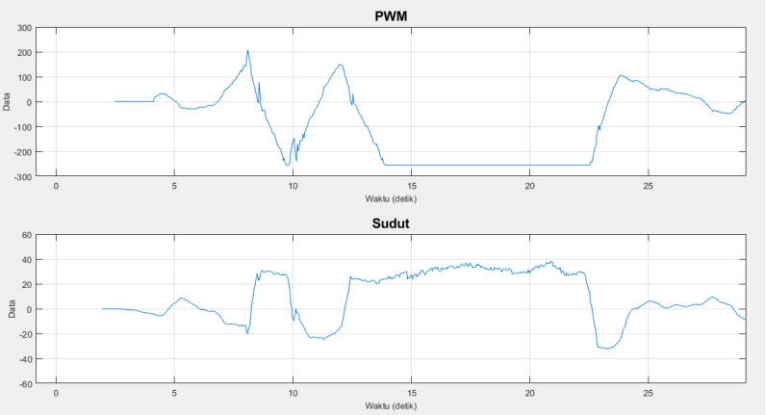
Pada gambar 4.19 didapatkan kondisi saat robot dijalankan selama 25 detik didapatkan rentang besaran sudut antara -10 s/d 16 derajat menghasilkan besaran pwn -90 s/d 80 dan didapatkan grafik pwm dan sudut yang beraturan dan berkesinambungan.

*** Percobaan 1C, $K_u$=5 dan $T_u$=0.2.**

Berikut tampilan grafik pada Matlab 2017a:




(a)



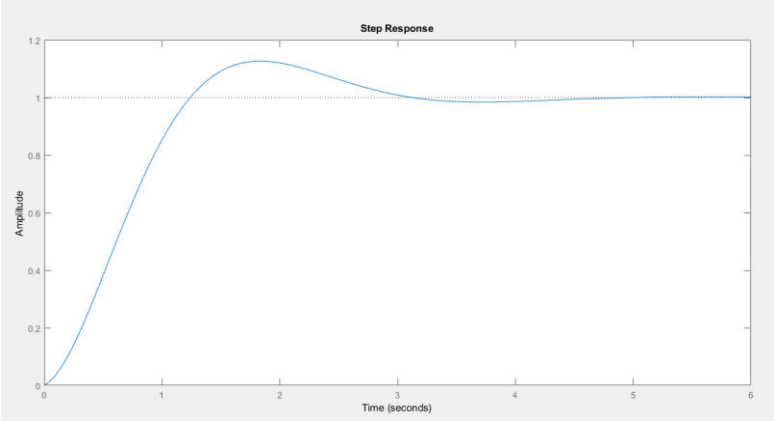

(b)

**Gambar 4.20.** Grafik hasil dengan inputan $K_u$=5 dan $T_u$=0.2: (a)Simulasi (b)Robot

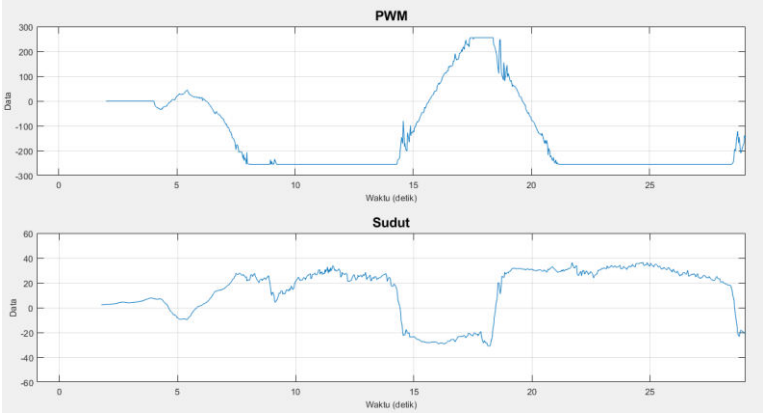
Pada gambar 4.20 didapatkan kondisi saat robot dijalankan selama 25 detik didapatkan rentang besaran sudut antara -35 s/d 38 derajat menghasilkan besaran pwn -255 s/d 210 dan didapatkan grafik pwm dan sudut yang tidak beraturan.

*** Percobaan 1D, $K_u$=5 dan $T_u$=0.3.**

Berikut tampilan grafik pada Matlab 2017a:




(a)



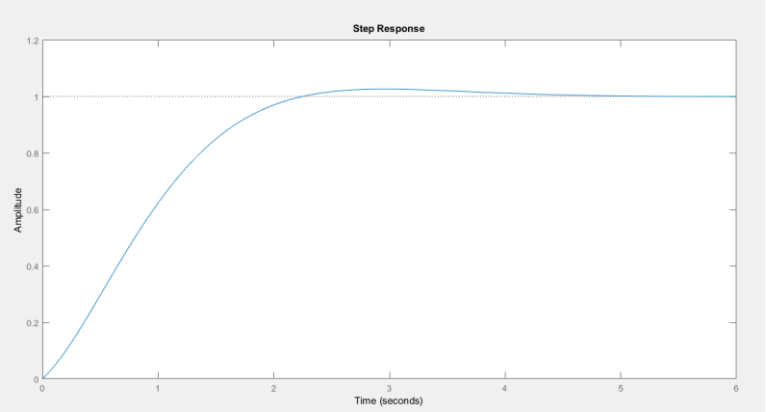

(b)

**Gambar 4.21.** Grafik hasil dengan inputan $K_u$=5 dan $T_u$=0.3: (a)Simulasi (b)Robot

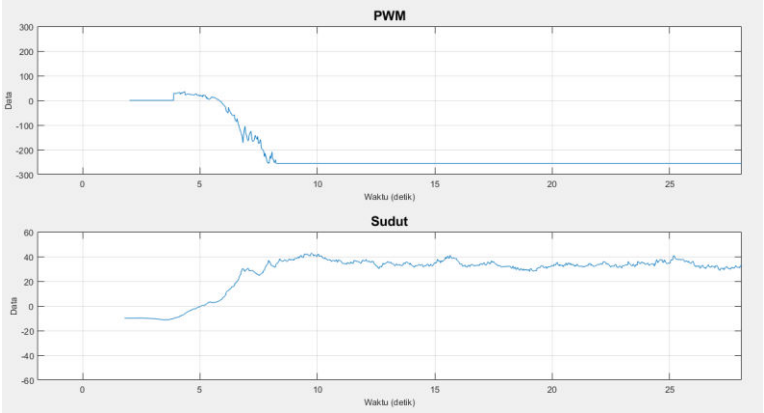
Pada gambar 4.21 didapatkan kondisi saat robot dijalankan selama 25 detik didapatkan rentang besaran sudut antara -35 s/d 39 derajat menghasilkan besaran pwn -255 s/d 255 dan didapatkan grafik pwm dan sudut yang tidak beraturan.

*** Percobaan 1E, $K_u$=5 dan $T_u$=0.5.**

Berikut tampilan grafik pada Matlab 2017a:




(a)




(b)

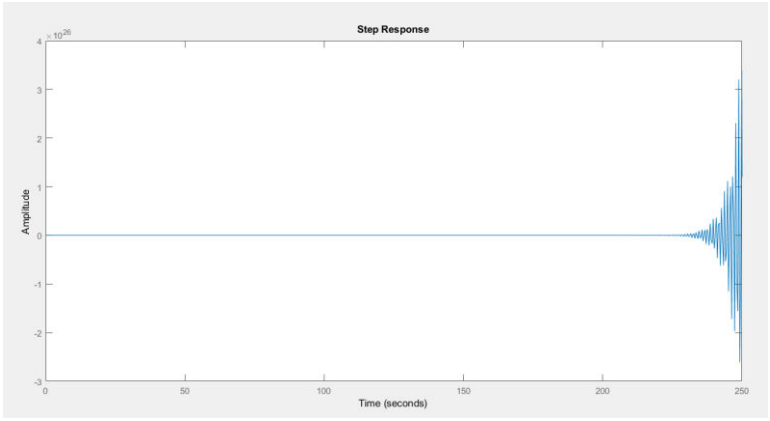
**Gambar 4.22.** Grafik hasil dengan inputan $K_u$=5 dan $T_u$=0.5: (a)Simulasi (b)Robot

Pada gambar 4.22 didapatkan kondisi saat robot dijalankan selama 25 detik didapatkan rentang besaran sudut antara -10 s/d 45 derajat menghasilkan besaran pwn -255 s/d 40 dan didapatkan grafik pwm dan sudut yang tidak beraturan.

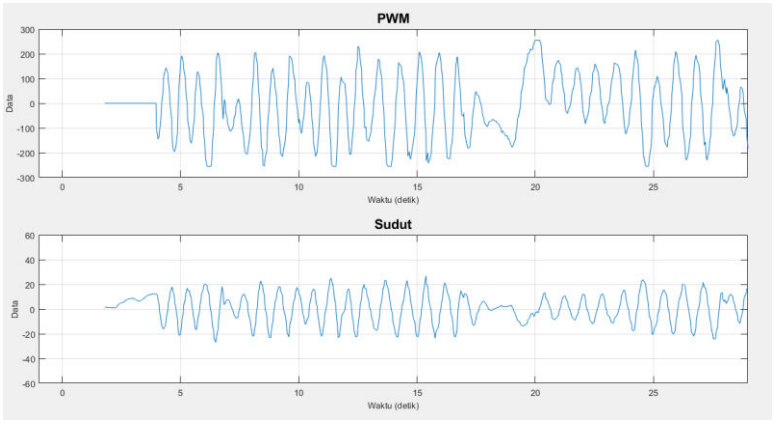
**2. Percobaan dengan nilai parameter $K_u$ =10 , ada tiga perobaan;**

*** Percobaan 2A, $K_u$=10 dan $T_u$=0.05.**

Berikut tampilan grafik pada Matlab 2017a:




(a)



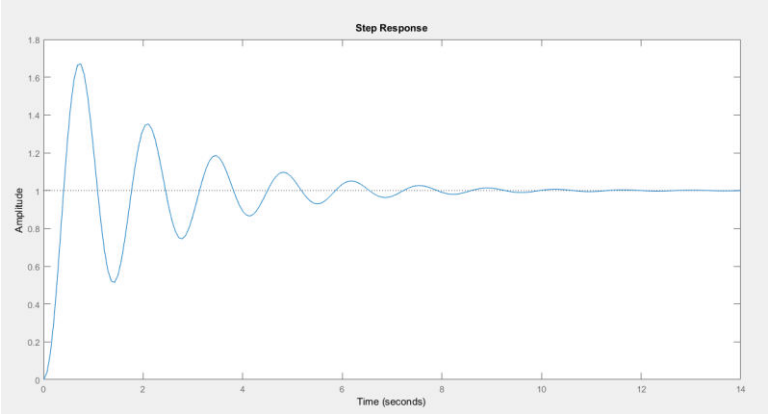

(b)

**Gambar 4.23.** Grafik hasil dengan inputan $K_u$=10 dan $T_u$=0.05: (a)Simulasi (b)Robot

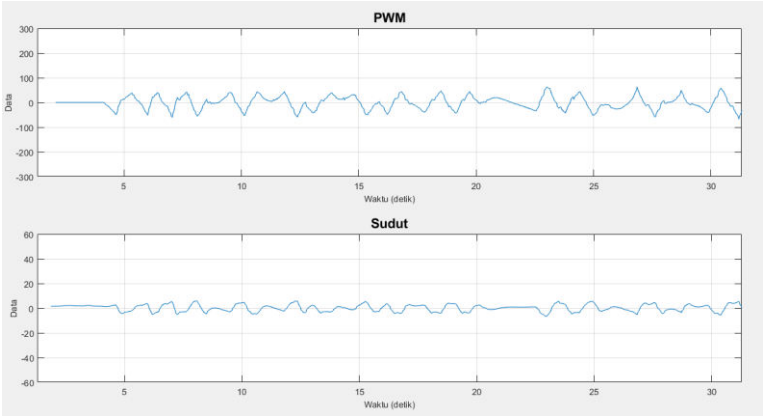
Pada gambar 4.23 didapatkan kondisi saat robot dijalankan selama 25 detik didapatkan rentang besaran sudut antara -28 s/d 28 derajat menghasilkan besaran pwn -255 s/d 255 dan didapatkan grafik pwm dan sudut yang tidak beraturan.

*** Percobaan 2B, $K_u$=10 dan $T_u$=0.1.**

Berikut tampilan grafik pada Matlab 2017a:




(a)



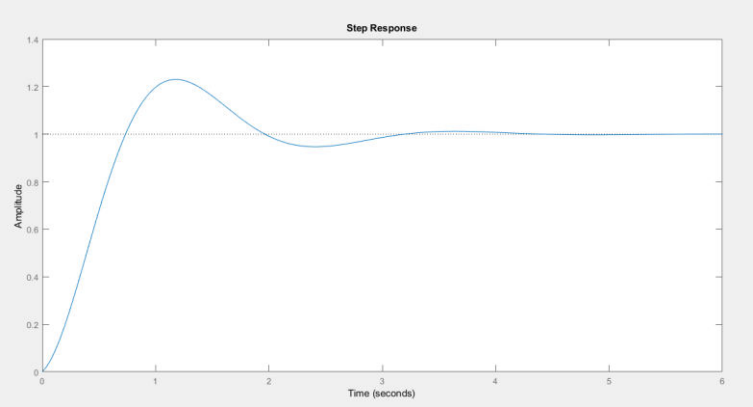

(b)

**Gambar 4.24.** Grafik hasil dengan inputan $K_u$=10 dan $T_u$=0.1: (a)Simulasi (b)Robot

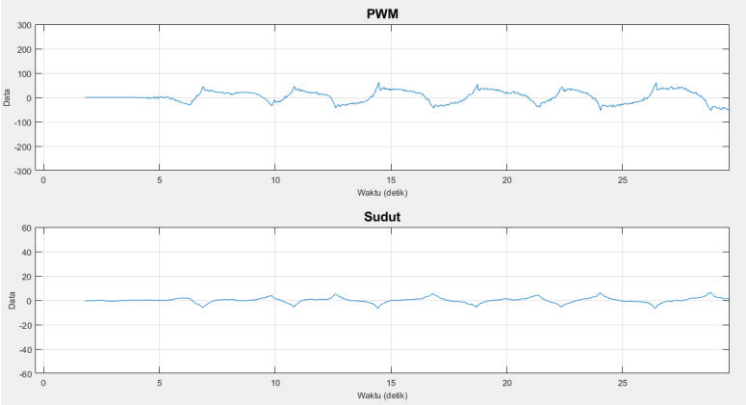
Pada gambar 4.24 didapatkan kondisi saat robot dijalankan selama 25 detik didapatkan rentang besaran sudut antara -8 s/d 8 derajat menghasilkan besaran pwn -60 s/d 60 dan didapatkan grafik pwm dan sudut yang beraturan dan berkesinambungan.

*** Percobaan 2C, $K_u$=10 dan $T_u$=0.2.**

Berikut tampilan grafik pada Matlab 2017a:




(a)



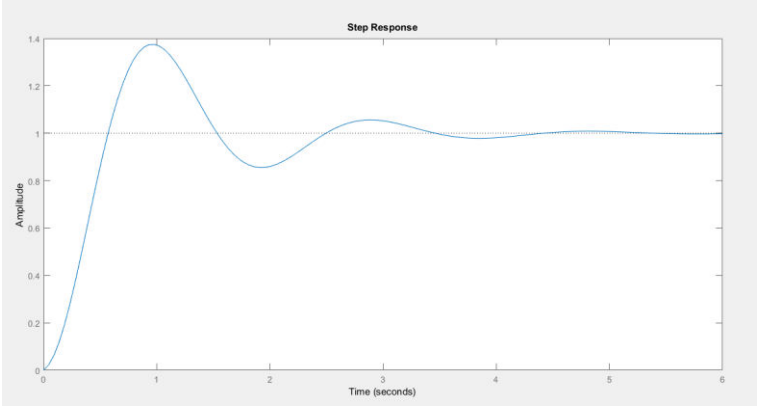

(b)

**Gambar 4.25.** Grafik hasil dengan inputan $K_u$=10 dan $T_u$=0.2: (a)Simulasi (b)Robot

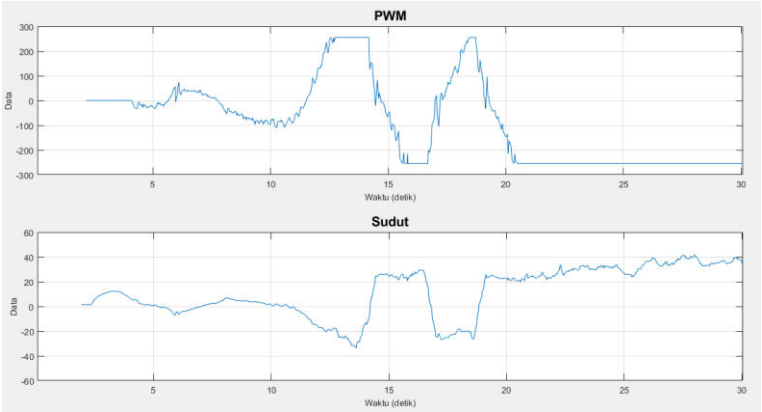
Pada gambar 4.25 didapatkan kondisi saat robot dijalankan selama 25 detik didapatkan rentang besaran sudut antara -8 s/d 8 derajat menghasilkan besaran pwn -60 s/d 70 dan didapatkan grafik pwm dan sudut yang beraturan dan berkesinambungan.

*** Percobaan 2D, $K_u$=10 dan $T_u$=0.3.**

Berikut tampilan grafik pada Matlab 2017a:




(a)



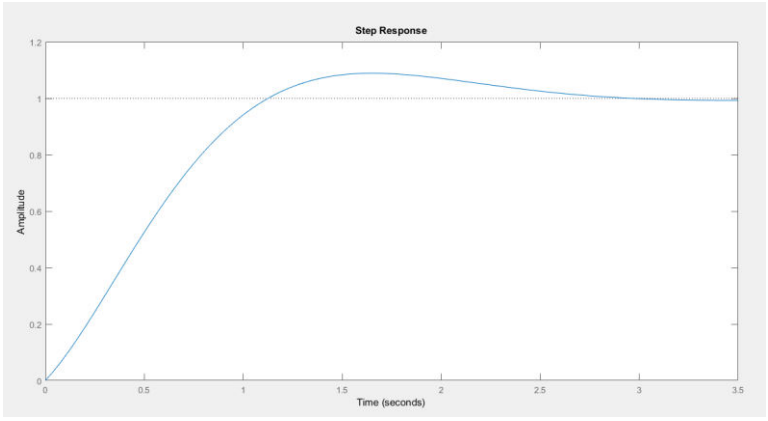

(b)

**Gambar 4.26.** Grafik hasil dengan inputan $K_u$=10 dan $T_u$=0.3: (a)Simulasi (b)Robot

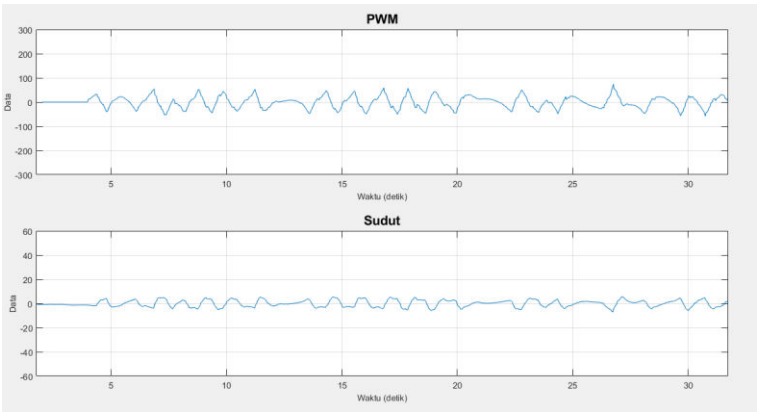
Pada gambar 4.26 didapatkan kondisi saat robot dijalankan selama 25 detik didapatkan rentang besaran sudut antara -38 s/d 38 derajat menghasilkan besaran pwn -255 s/d 255 dan didapatkan grafik pwm dan sudut yang tidak beraturan.

*** Percobaan 2E, $K_u$=10 dan $T_u$=0.5.**

Berikut tampilan grafik Simulasi pada Matlab 2017a:




(a)




(b)

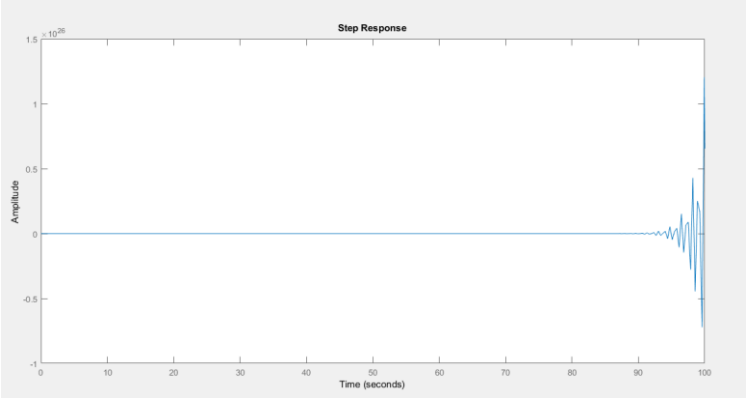
**Gambar 4.27.** Grafik hasil dengan inputan $K_u$=10 dan $T_u$=0.5: (a)Simulasi (b)Robot

Pada gambar 4.27 didapatkan kondisi saat robot dijalankan selama 25 detik didapatkan rentang besaran sudut antara -8 s/d 9 derajat menghasilkan besaran pwn -60 s/d 75 dan didapatkan grafik pwm dan sudut beraturan dan berkesinambungan.

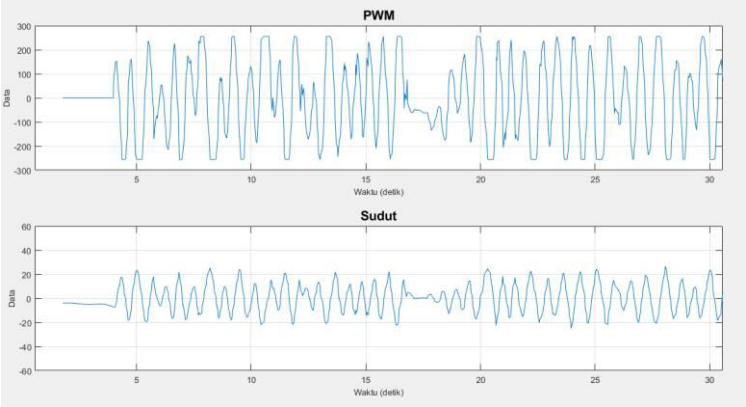
**3. Percobaan dengan nilai parameter $K_u$ =15 , ada tiga perobaan;**

*** Percobaan 3A, $K_u$=15 dan $T_u$=0.05.**

Berikut tampilan grafik pada Matlab 2017a:




(a)



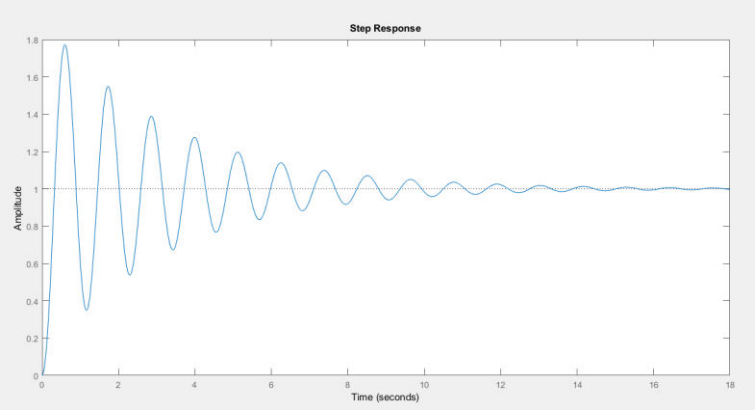

(b)

**Gambar 4.28.** Grafik hasil dengan inputan $K_u$=15 dan $T_u$=0.05: (a)Simulasi (b)Robot

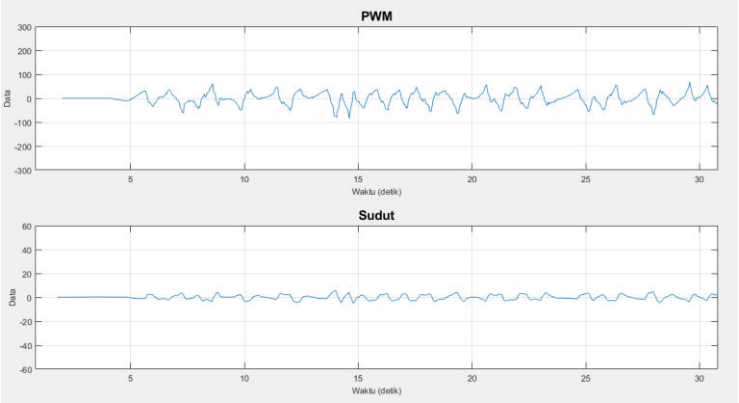
Pada gambar 4.28 didapatkan kondisi saat robot dijalankan selama 25 detik didapatkan rentang besaran sudut antara -28 s/d 30 derajat menghasilkan besaran pwn -255 s/d 255 dan didapatkan grafik pwm dan sudut yang tidak beraturan.

*** Percobaan 3B, $K_u$=15 dan $T_u$=0.1.**

Berikut tampilan grafik Simulasi pada Matlab 2017a:




(a)



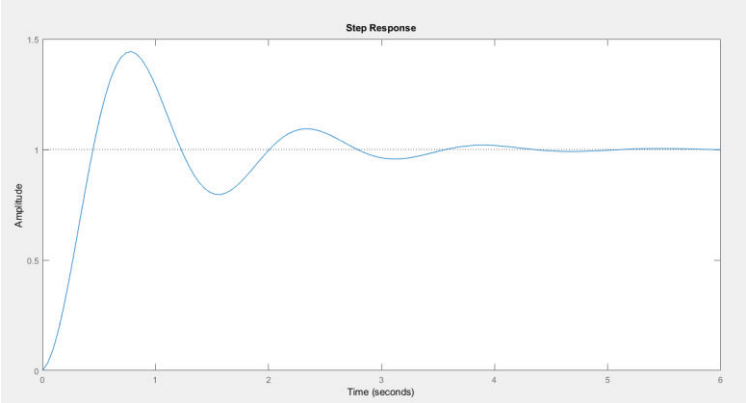

(b)

**Gambar 4.29.** Grafik hasil dengan inputan $K_u$=15 dan $T_u$=0.1: (a)Simulasi (b)Robot

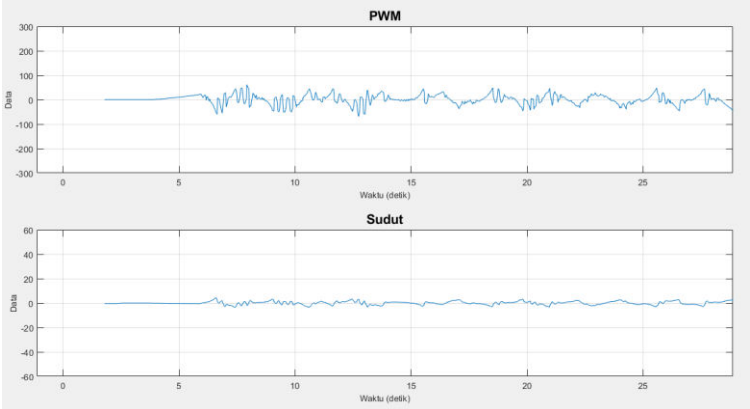
Pada gambar 4.29 didapatkan kondisi saat robot dijalankan selama 25 detik didapatkan rentang besaran sudut antara -6 s/d 7 derajat menghasilkan besaran pwn -90 s/d 80 dan didapatkan grafik pwm dan sudut beraturan dan berkesinambungan.

*** Percobaan 3C, $K_u$=15 dan $T_u$=0.2.**

Berikut tampilan grafik Simulasi pada Matlab 2017a:




(a)



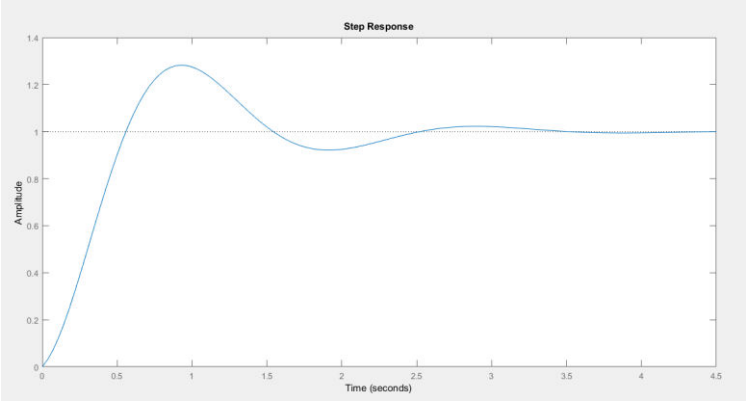

(b)

**Gambar 4.30.** Grafik hasil dengan inputan $K_u$=15 dan $T_u$=0.2: (a)Simulasi (b)Robot

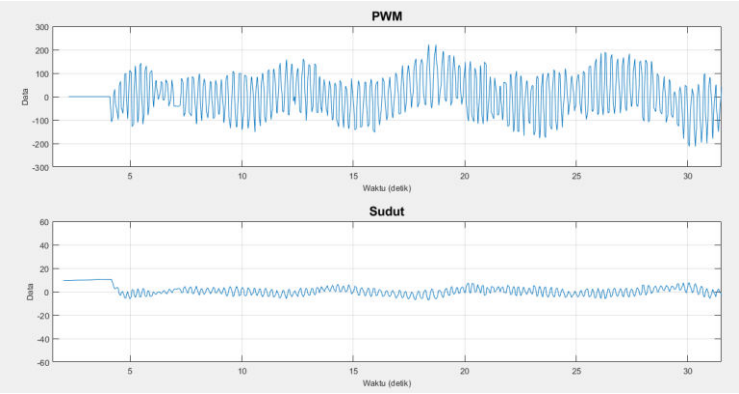
Pada gambar 4.30 didapatkan kondisi saat robot dijalankan selama 25 detik didapatkan rentang besaran sudut antara -5 s/d 5 derajat menghasilkan besaran pwn -70 s/d 70 dan didapatkan grafik pwm dan sudut beraturan dan berkesinambungan.

*** Percobaan 3D, $K_u$=15 dan $T_u$=0.3.**

Berikut tampilan grafik Simulasi pada Matlab 2017a:




(a)



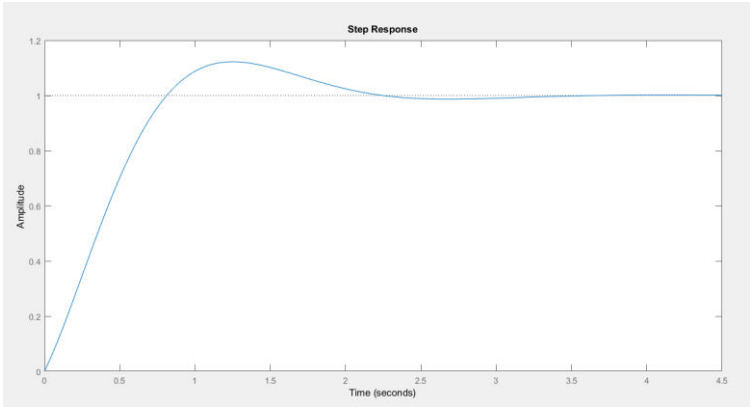

(b)

**Gambar 4.31.** Grafik hasil dengan inputan $K_u$=15 dan $T_u$=0.3: (a)Simulasi (b)Robot

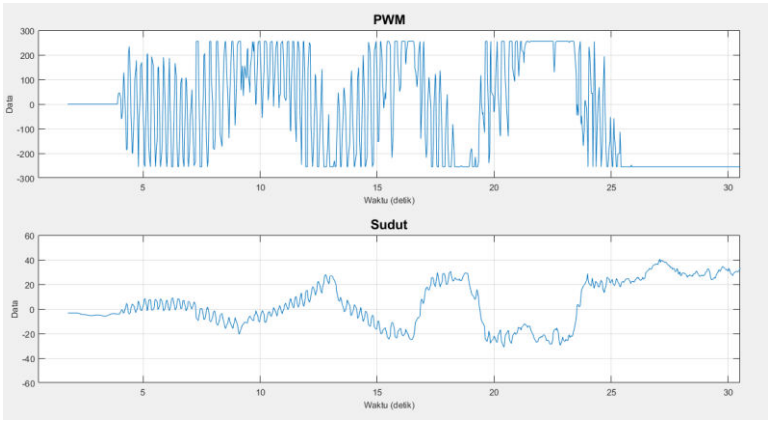
Pada gambar 4.31 didapatkan kondisi saat robot dijalankan selama 25 detik didapatkan rentang besaran sudut antara -10 s/d 10 derajat menghasilkan besaran pwn -180 s/d 210 dan didapatkan grafik pwm dan sudut beraturan dan berkesinambungan.

*** Percobaan 3E, $K_u$=15 dan $T_u$=0.5.**

Berikut tampilan grafik Simulasi pada Matlab 2017a:




(a)




(b)

**Gambar 4.32.** Grafik hasil dengan inputan $K_u$=15 dan $T_u$=0.5: (a)Simulasi (b)Robot

Pada gambar 4.32 didapatkan kondisi saat robot dijalankan selama 25 detik didapatkan rentang besaran sudut antara -30 s/d 40 derajat menghasilkan besaran pwn -255 s/d 255 dan didapatkan grafik pwm dan sudut beraturan dan berkesinambungan.

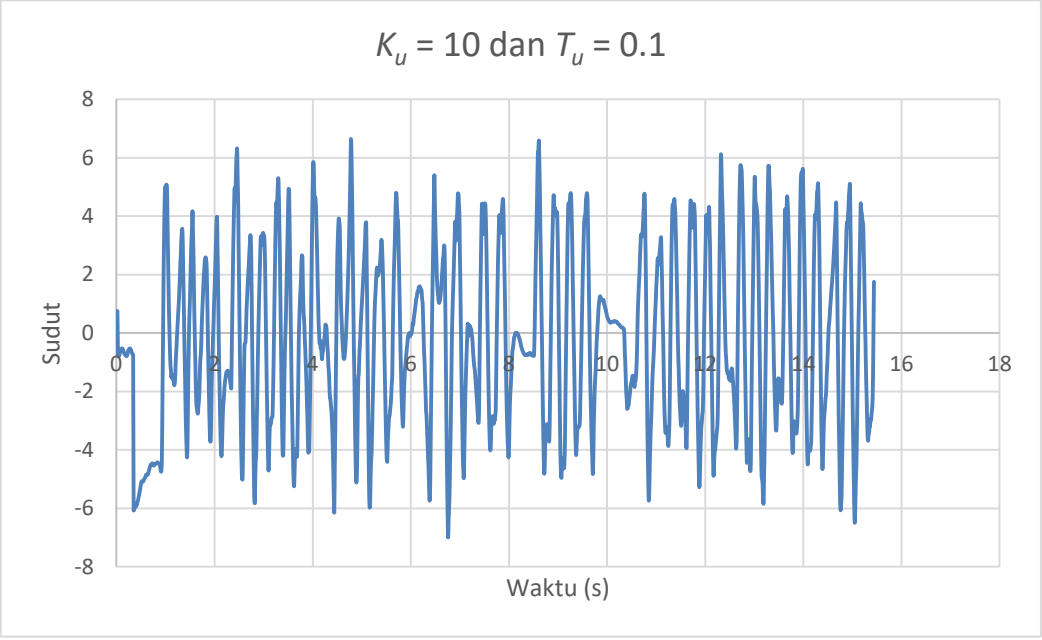
**Tabel 9**

Perbandingan hasil pengujian robot

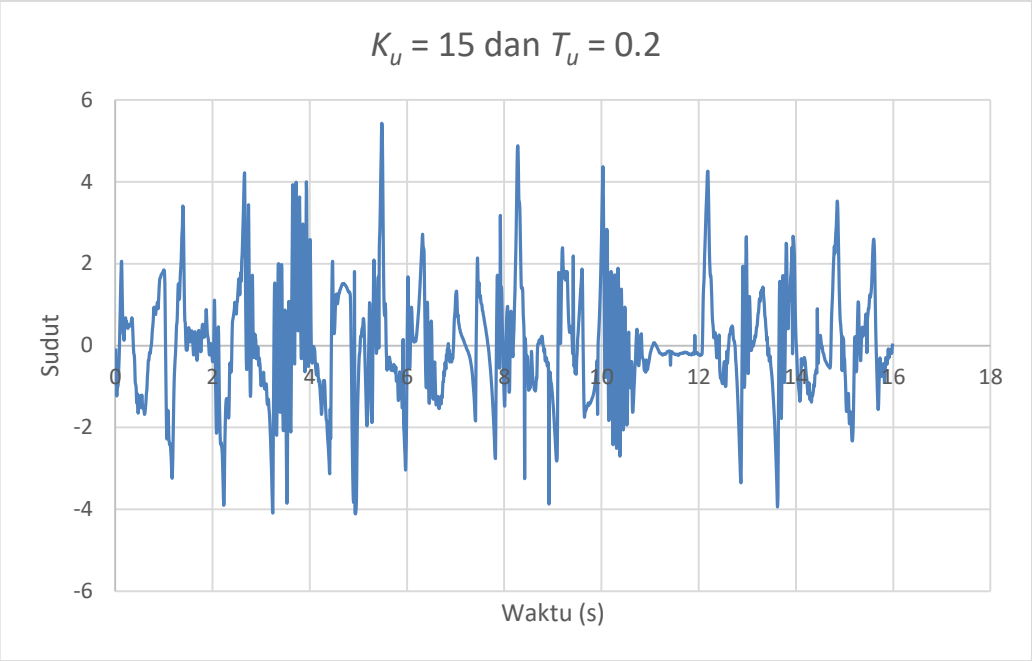
| **Parameter $K_u$ dan $T_u$** | **Besaran Sudut** ($^o$) | **Nilai PWM** | **Kondisi Robot** |
|---|---|---|---|
| Percobaan 1A, $K_u$=5 dan $T_u$=0.05. | -30 s/d 30 | -240 s/d 200 | Robot tidak dapat seimbang |
| Percobaan 1B, $K_u$=5 dan $T_u$=0.1. | -10 s/d 16 | -90 s/d 80 | Robot dapat seimbang |
| Percobaan 1C, $K_u$=5 dan $T_u$=0.2. | -35 s/d 38 | -255 s/d 255 | Robot tidak dapat seimbang |
| Percobaan 1D, $K_u$=5 dan $T_u$=0.3. | -35 s/d 39 | -255 s/d 255 | Robot tidak dapat seimbang |
| Percobaan 1E, $K_u$=5 dan $T_u$=0.5. | -10 s/d 45 | -255 s/d 40 | Robot tidak dapat seimbang |
| Percobaan 2A, $K_u$=10 dan $T_u$=0.05. | -28 s/d 28 | -255 s/d 255 | Robot tidak dapat seimbang |
| Percobaan 2B, $K_u$=10 dan $T_u$=0.1. | -8 s/d 8 | -60 s/d 60 | Robot dapat seimbang |
| Percobaan 2C, $K_u$=10 dan $T_u$=0.2. | -8 s/d 8 | -60 s/d 70 | Robot dapat seimbang |
| Percobaan 2D, $K_u$=10 dan $T_u$=0.3. | -38 s/d 38 | -255 s/d 255 | Robot tidak dapat seimbang |
| Percobaan 2E, $K_u$=10 dan $T_u$=0.5. | -8 s/d 9 | -60 s/d 75 | Robot dapat seimbang |
| Percobaan 3A, $K_u$=15 dan $T_u$=0.05. | -25 s/d 30 | -255 s/d 255 | Robot tidak dapat seimbang |
| Percobaan 3B, $K_u$=15 dan $T_u$=0.1. | -6 s/d 7 | -90 s/d 80 | Robot dapat seimbang |
| Percobaan 3C, $K_u$=15 dan $T_u$=0.2. | -5 s/d 5 | -70 s/d 70 | Robot dapat seimbang |
| Percobaan 3D, $K_u$=15 | -10 s/d 10 | -180 s/d 210 | Robot dapat |

| | | | |
|---|---|---|---|
| dan $T_u$=0.2. | | | seimbang |
| Percobaan 3E, *Ku*=15 dan $T_u$=0.5. | -30 s/d 40 | -255 s/d 255 | Robot tidak dapat seimbang |

Berdasarkan keseluruhan dari pengujian yang dilakukan ini dapat dinyatakan bahwa *balancing* robot dapat berjalan dengan baik. Proses penggunaan kontrol PID Ziegler-Nichols dengan posisi yang paling mendekati titik seimbang adalah pada percobaan 2B dengan *setting* nilai $K_u$ = 10 dan $T_u$ = 0.1 serta 3C dengan *setting* nilai $K_u$ = 15 dan $T_u$ = 0.2 dimana *system* mampu menentukan keputusan dalam pengaturan arah dan kecepatan motor DC sebagai usaha untuk mempertahankan posisi seimbang robot dengan tampilan grafik yang lebih stabil antara sudut terhadap pwm motor secara berkelanjutan melalui grafik yang dihasilkan. Data hasil *setting* dapat dilihat pada lampiran 2 dan ditunjukan pada gambar 4.33.




(a)




(b)

**Gambar 4.33.** Grafik Data Hasil Setting (a) $K_u$ = 10 dan $T_u$ = 0.1 (b) $K_u$ = 15 dan $T_u$ = 0.2

## 4.4. Perbandingan Percobaan

Perbandingan percobaan ini dilakukan dengan membandingkan percobaan yang dilakukan pada tugas akhir ini dengan penelitian sebelumnya [3]. Berikut merupakan hasil dari perbandingan :

**Tabel 10**

Perbandingan Percobaan

| Tugas Akhir ini | Penelitian Sebelumnya [3] |
|---|---|
| Percobaan dilakukan dalam waktu 25 detik | Percobaan dilakukan dalam waktu 15 detik |
| Penentuan nilai PID menggunakan metode PID Ziegler-Nichols | Penentuan nilai PID secara Manual |
| Menggunakan Simulasi program | Tanpa simulasi program |

| | |
|---|---|
| Percobaan dilakukan sebanyak 15 percobaan | Percobaan dilakukan sebanyak 18 percobaan |
| Tanpa menggunakan nilai referensi | Menggunakan nilai referensi |
| Menggunakan Batasan besaran sudut -90 s/d 90 dan pwm sebesar 0 – 255 | Tanpa menggunakan Batasan besaran sudut dan pwm. |
| Nilai yang mendekati keseimbangan<br>$K_u$ = 10 dan $T_u$ = 0,1.<br>$K_u$= 15 dan $T_u$ = 0,2. | Nilai yang mendekati keseimbangan<br>$K_p$ = 0,9<br>$K_i$ = 0,9<br>$K_d$ = 0.001 |

Pada penelitian sebelumnya [3] untuk mendapatkan keadaan seimbang robot dilakukan penyetingan nilai PID secara manual sehingga membutuhkan waktu dan ketelitian untuk menemukan keadaan nilai yang dibutuhkan agar robot dapat seimbang. Berdasarkan hasil percobaan didapatkan bentuk grafik dari besaran sudut dan pwm yang sebagian besar tidak beraturan dan tidak berulang.

Pada tugas akhir ini menggunakan *tuning* Ziegler-Nichols untuk menentukan nilai PID dimana *tuning* ini bertujuan untuk mencapai nilai maksimum *overshoot* berkisar atau kurang dari 25%. Sehingga pada percobaan tidak membutuhkan waktu yang lama untuk mendapatkan nilai keseimbangan serta memiliki ketelitian yang lebih baik. Pada percobaan tugas akhir ini dapat dilihat bentuk grafik dari besaran sudut dan pwm sebagian besar beraturan dan berulang.