Laporan Penelitian

WHEELED ROBOT BASED FOR INDONESIAN INTELLIGENT ROBOT CONTEST

ROBOT BERODA UNTUK KONTES ROBOT CERDAS INDONESIA

Disusun oleh:

Peneliti Utama : Novie Theresia Br. Pasaribu, ST., MT. (220924)

Peneliti II : Dr. Erwani Merry Sartika (220148)

Asisten Peneliti: Rocky Anthoni (0822059)

Agus Santoso (0822045)

FAKULTAS TEKNIK JURUSAN TEKNIK ELEKTRO

UNIVERSITAS KRISTEN MARANATHA

BANDUNG

2012

LEMBAR IDENTITAS DAN PENGESAHAN LAPORAN PENELITIAN

Judul Penelitian:

Robot Beroda untuk Kontes Robot Cerdas Indonesia

1. Ketua/Penanggungjawab Pelaksana Kegiatan Penelitian
   - Nama Lengkap dan Gelar : Novie Theresia Br. Pasaribu,S.T.M.T.
   - NIK : 220924
   - Jabatan Akademik/Golongan : Asisten Ahli /IIIB
   - Fakultas/Jurusan : Teknik/Teknik Elektro
2. Pelaksana Kegiatan Penelitian ke II
   - Nama Lengkap dan Gelar : Dr. Erwani Merry Sartika, S.T., M.T.
   - NIK : 220148
   - Jabatan Akademik/Golongan : Lektor /IVA
   - Fakultas/Jurusan : Teknik/Teknik Elektro
3. Jumlah Asisten Peneliti : 2 orang
4. Lokasi Pelaksanaan Penelitian : Universitas Kristen Maranatha
5. Lama Pelaksanaan Penelitian : Januari 2012 – Agustus 2012
6. Sumber Dana Penelitian : Universitas Kristen Maranatha
7. Biaya Penelitian : Rp. 20.365.000,-

Bandung, Agustus 2012

Ketua Peneliti

Menyetujui,
Dekan Fakultas Teknik
Universitas Kristen Maranatha

Novie Theresia Br.Pasaribu.,ST.,MT.

Peneliti II

Ir. Aan Darmawan MT.

Dr. Erwani Merry Sartika, S.T., M.T.

Mengetahui,
Ketua LPPM Universitas Kristen Maranatha

Prof. Dr. Ir. Benjamin Soenarko, MSME.

Robot Beroda untuk Kontes Robot Cerdas Indonesia

Jurusan Teknik Elektro, Fakultas Teknik, Universitas Kristen Maranatha

ABSTRAK

Kontes Robot Cerdas Indonesia (KRCI) merupakan sebuah kompetisi robot yang diadakan setiap tahun. Setiap tahun kompetisi KRCI mempunyai perbedaan tingkat kesulitan yang harus diatasi oleh robot. Robot harus dapat bergerak dengan cepat dan lincah, kembali ke posisi *HOME*, dan memasuki seluruh ruang. Berbagai kelemahan belum dapat diatasi terutama dalam penerapan sistem kontrol yang banyak mengandalkan sistem kontrol *On Off*.

Pada penelitian ini, dirancang robot beroda yang mempunyai dimensi yang lebih kecil dan menggunakan sistem kontrol PID. Metoda tuning PID yang digunakan adalah modifikasi antara metoda Ziegler Nichols II dan *trial and error*. Metoda ini dapat membantu mempercepat perolehan nilai parameter PID.

Berdasarkan percobaan yang dilakukan modifikasi *trial and error* dan Ziegler Nichols II dapat diterapkan untuk kecepatan motor DC rendah (*reference* PWM 25, 50), sedangkan untuk kecepatan motor DC tinggi (75, 125) lebih tepat menggunakan metoda *trial and error* saja. Selain itu desain robot dianjurkan menempatkan posisi sensor lebih ke depan dari *actuator* agar *error* terbaca lebih dahulu dan respon kontroler tidak terlambat.

Kata Kunci : Robot Beroda, KRCI, Sensor UVtron, Sensor Jarak *Ultrasonic*, Pengontrol Mikro Atmega128.

Wheeled Robot Based for Indonesian Intelligent Robot Contest

Electrical Engineering, Maranatha Christian University,

ABSTRACT

Indonesian Intelligent Robot Contest (KRCI) is a robot competition held every year. Each year the competition KRCI have different levels of difficulty to be overcome by the robot. The robot must be able to move quickly and swiftly, back to the HOME position, and go into the whole space. These weaknesses have not been able to overcome, especially in the application of control systems that are relying on the control system On Off.

In this research, designed a wheeled robot that has smaller dimensions and the use of PID control systems. PID tuning method used is a modification of the method of Ziegler and Nichols II trial. This method can help to accelerate to get the PID parameter values .

Based on experiments, modification the method of trial and error and Ziegler Nichols II can be applied to low-speed DC motor (PWM reference 25, 50), while for high-speed DC motor (75, 125) is more appropriate to use the method of trial and error only. In addition it is recommended to put a robot design over the next sensor position of the actuator, so that errors can be read forward and the response of the controller is not too late.

Keywords: Wheeled Robot, KRCI, UVTron Sensor, Ultrasonic Sensor, Micro Controller Atmega 128.

DAFTAR ISI

Halaman

ABSTRAK i

ABSCTRACT ii

DAFTAR ISI iii

DAFTAR TABEL viii

DAFTAR GAMBAR ix

BAB 1 PENDAHULUAN

1.1 Latar Belakang 1

1.2 Perumusan Masalah 2

1.3 Tujuan 2

1.4 Pembatasan Masalah 2

1.5 Spesifikasi Alat yang Digunakan 2

1.6 Sistematika Penulisan 3

BAB 2 TEORI DASAR

II.1 Kontroler 5

II.1.1 *Feedback* Kontrol Sistem *(*Sistem Kontrol *Closed Loop)*..... 5
II.1.2 Sistem Kontrol *Open Loop*..... 6
II.2 .....Tanggap an Peralihan (*Transient Respon)*..... 7
II.3 .....Aksi Kontrol..... 8
II.3.1 Aksi Kontrol *On-Off*..... 8
II.3.2 Aksi Kontrol *Proportional* (P)..... 10
II.3.3 Aksi Kontrol *Integral* (I)..... 11
II.3.4 Aksi Kontrol *Derivative* (D)..... 13
II.3.5 Aksi Kontrol PID..... 14
II.4 .....Tuning PID..... 18
II.4.1 Ziegler Nichols II (Metoda Osilasi)..... 18
II.4.2 *Trial and Error*..... 19
II.5 .....Penganta r Robotika..... 20
II.5.1Robot Beroda..... 20
II.5.1.1 *Diffential Drive*..... 20
II.5.1.2 *Tricycle Drive*..... 20
II.5.1.3 *Synchronous Drive*..... 21
II.5.1.4 *Holonomic Drive*..... 21
II.6 .....Sensor 22
II.6.1 Sensor Jarak Ultrasonik (SRF05)..... 23
II.6.2 Sensor Api UVTron..... 24
II.6.3 Sensor Warna..... 26
II.7 .....Pengenal an Mikro 27
II.7.1 Pengenalan ATMEL AVR RISC..... 27
II.7.2 Pengontrol Mikro ATmega 128..... 28

BAB 3 PERANCANGAN DAN REALISASI

III.1 Sensor dan *Driver* Motor DC..... 29

II.1.1 Sensor Jarak Ultrasonic SRF05**.....** 29

II.1.2 Sensor Api UV-TRON Hamamatsu R2868**.....** 31

II.1.3 Sensor Warna ZX-03**.....** 31

II.1.4 Rangkaian *Driver* Motor DC VNH3SP30 MD01B**.....** 31

III.2 Penghubung Matlab Pada Laptop dengan Mikrokontroller... 33

III.3 Perancangan Sistem Robot Beroda dengan Sistem Kontrol PID..... 34

II.3.1 Diagram Blok Sistem Kontrol Gerak Robot Beroda**.....** 36

II.3.2 Diagram Blok Sistem Pemadam Api Robot Beroda**.....** 37

III.4 Bentuk Robot..... 38

III.5 Perancangan dan Realisasi Robot Beroda Pemadam Api..... 39

III.6 Sistem Gerak..... 41

III.7 Pemutar Kipas Pada Robot..... 41

III.8 Skematik Pengontrol Mikro ATMega128A..... 41

III.9 Metoda Tuning PID..... 43

III.10 *Simulink* Matlab..... 44

III.11 Algoritma Pemrograman pada Robot..... 45

III.11.1 *Flowchart* Utama pada Robot Pemadam Beroda**.....** 46

III.11.2 *Flowchart* untuk *Main Sub Program***.....** 47

III.11.3 *Flowchart* untuk *Sub Program Set* Nilai Kp, Ki, Kd**.....** 50

III.11.4 *Flowchart* untuk Sub Program *Wall Follower* Kanan Menggunakan PID dan Kirim Data ke Matlab**.....** 52

III.11.5 *Flowchart* untuk Sub Program *Wall Follower* Kiri Menggunakan PID**.....** 55

BAB 4 DATA PENGAMATAN DAN ANALISIS

IV.1 Pengujian Sensor Warna..... 57

IV.2 Pengujian Sensor *Ultrasonic* / Sensor Jarak SRF05..... 58

IV.3 *Tuning* PID pada Robot..... 59
IV.3.1 *Tuning* PID untuk *set point* jarak (8cm) pada *Reference* PWM 25 dan 50..... 60
IV.3.1.1 *Tuning* Nilai Parameter *Proportional* (Kp) untuk *Reference* PWM 25 dan *set point* jarak (8cm)..... 61
IV.3.1.2 *Tuning* Nilai Parameter *Integral* (Ki) untuk *Reference PWM* 25 dan *set point* jarak (8cm)..... 62
IV.3.1.3 *Tuning* Nilai Parameter *Derivative* (Kd) untuk *Reference* PWM 25 dan *set point* jarak (8cm)..... 64
IV.3.1.4 Pengujian Hasil *Tuning* Terbaik pada Kontroler P, PI dan PID untuk *Reference* PWM 25 dan *set point* jarak (8cm) 66
IV.3.1.5 Pengujian Pada Arena KRCI untuk *Reference* PWM 25 dan *set point* jarak (8cm)..... 67
IV.3.2 Pengujian *Tuning* PID untuk *Reference* PWM 50 dan *set point* jarak (8cm)..... 71
IV.3.2.1 *Tuning* Nilai Parameter *Proportional* (Kp) untuk *Reference* PWM 50 dan *set point* jarak (8cm)..... 71
IV.3.2.2 *Tuning* Nilai Parameter *Integral* (Ki) untuk *Reference PWM* 50 dan *set point* jarak (8cm)..... 73
IV.3.2.3 *Tuning* Nilai Parameter *Derivative* (Kd) untuk *Reference* PWM 50 dan *set point* jarak (8cm)..... 75
IV.3.2.4 Pengujian Hasil *Tuning* Terbaik pada Kontroler P, PI dan PID untuk *Reference* PWM 50 dan *set point* jarak (8cm)..... 76
IV.3.2.5 Pengujian Pada Arena KRCI untuk *Reference* PWM 50 dan *set point* jarak (8cm)..... 78
IV.3.3 Pengujian *Tuning* PID untuk *Reference* PWM 75 dan *set point* jarak (8cm)..... 81
IV.3.3.1 *Tuning* Nilai Parameter *Proportional* (Kp) untuk *Reference* PWM 75 dan *set point* jarak (8cm)..... 82

IV.3.3.2 *Tuning* Nilai Parameter *Integral* (Ki) untuk *Reference PWM* 75 dan *set point* jarak (8cm)..... 83
IV.3.3.3 *Tuning* Nilai Parameter *Derivative* (Kd) untuk *Reference* PWM 75 dan *set point* jarak (8cm)..... 85
IV.3.3.4 Pengujian Hasil *Tuning* Terbaik pada Kontroler P, PI dan PID untuk *Reference* PWM 75 dan *set point* jarak (8cm)..... 87
IV.3.3.5 Pengujian Pada Arena KRCI untuk *Reference* PWM 75 dan *set point* jarak (8cm)..... 89
IV.4 Pengujian Pembanding Kontoler P, PI dan PID..... 92
IV.5 Pengujian Kontroler PID..... 94

BAB 5 KESIMPULAN DAN SARAN

V.1 Kesimpulan..... 97
V.2 Saran..... 98

DAFTAR PUSTAKA…… 99

LAMPIRAN – A Foto Robot Beroda

LAMPIRAN – B Program pada Pengontrol Mikro ATmega128

LAMPIRAN – C Datasheet

DAFTAR GAMBAR

Halaman

Gambar 2.1 Diagram blok sistem kontrol tertutup..... 5

Gambar 2.2 Diagram blok sistem kontrol terbuka..... 6

Gambar 2.3 Kurva tanggapan sistem dengan karakteristik *transient respon*..... 8

Gambar 2.4 Ilustrasi aksi kontrol *on-off* 9

Gambar 2.5 Ilustrasi aksi kontrol *on-off* dengan histerisis..... 10

Gambar 2.6 Diagram blok sistem kontrol *proportional*..... 10

Gambar 2.7 Diagram blok sistem kontrol *Integral*..... 11

Gambar 2.8 Kurva sinyal kesalahan e(t) terhadap t dan kurva u(t) terhadap t pada pembangkit kesalahan nol..... 12

Gambar 2.9 Diagram blok sistem kontrol *Derivative*..... 13

Gambar 2.10 Kurva waktu hubungan kontroler *input-output diferential* 13

Gambar 2.11 Diagram blok PID *parallel*..... 15

Gambar 2.12 Hubungan dalam fungsi waktu antara sinyal keluaran dengan masukan untuk kontroler PID..... 15

Gambar 2.13 PID dengan hubungan Kp diseri Kd dan Ki..... 16

Gambar 2.14 PID dengan Kp terpisah dengan Kd dan Ki..... 17

Gambar 2.15 Kurva respon *sustain oscillation*..... 18

Gambar 2.16 Sistem gerak *differential drive*..... 20

Gambar 2.17 a.Sistem Gerak *Tricycle Drive* dan b. Kendaraan

dengan Sistem Gerak *Tricycle Drive*..... 21

Gambar 2.18 Sistem gerak *synchronous drive*..... 21

Gambar 2.19 Robot dengan 3 roda *omni-directional*..... 22

Gambar 2.20 Sistem gerak *Holomonic Drive*..... 22

Gambar 2.21a Diagram waktu sensor SRF05 *mode* 1..... 23

Gambar 2.21b Diagram waktu sensor SRF05 *mode* 2..... 24

Gambar 2.22 Sensor Api (a. UVTron R2868 dan
b. Rangkaian Pengaktif) 24

Gambar 2.23 *Spektrum* respon UVTron..... 25

Gambar 2.24 Derajat sensitivitas Hamamtasu R2868..... 25

Gambar 2.25 Sensor Warna (a. Bentuk Sensor TCRT5000
dan b. Koneksi Pin ZX-03) 26

Gambar 3.1 Alokasi pin sensor SRF05..... 30

Gambar 3.2 Diagram alir penggunaan sensor SRF05..... 30

Gambar 3.3 Alokasi pin UVTron Hamamatsu R2868..... 31

Gambar 3.4 Alokasi pin sensor warna..... 31

Gambar 3.5 *Driver* motor DC VNH3SP30 MD01B..... 32

Gambar 3.6 Rangkaian dalam motor *driver*..... 32

Gambar 3.7 Diagram blok penghubung Matlab pada laptop dengan
mikrokontroler..... 33

Gambar 3.8 Penghubung Matlab pada laptop dengan mikrokontroler 34

Gambar 3.9a *Mode non-arbitary start*..... 34

Gambar 3.9b *Mode arbitary start*..... 35

Gambar 3.10 Diagram blok sistem gerak robot..... 36

Gambar 3.11 Diagram blok sistem pemadam menggunakan sensor warna..... 37
Gambar 3.12 Diagram blok sistem pemadam api robot beroda..... 37
Gambar 3.13 Diagram blok sistem robot beroda..... 38
Gambar 3.14 Dimensi robot beroda pemadam api dan penempatan sensor..... 39
Gambar 3.15 Penempatan sensor pada robot 40
Gambar 3.16 Diagram blok pemutar kipas..... 41
Gambar 3.17 Diagram blok pengontrol mikro ATMega128A..... 42
Gambar 3.18 Bagian *main board* ATMega128A..... 42
Gambar 3.19 Bagian *downloader* dari *main board*..... 43
Gambar 3.20 Hubungkan Query dengan Scope, To Workspace dan Display..... 45
Gambar 3.21 *Flowchart* robot menggunakan *system control* PID..... 46
Gambar 3.22a *Flowchart main program* bagian pertama..... 48
Gambar 3.22b *Flowchart main program* bagian kedua..... 49
Gambar 3.23a *Flowchart* set nilai Kp, Ki, Kd bagian pertama..... 50
Gambar 3.23b *Flowchart* set nilai Kp, Ki, Kd bagian kedua..... 51
Gambar 3.24a *Flowchart wall follower* kanan menggunakan PID bagian pertama..... 52
Gambar 3.24b *Flowchart wall follower* kanan menggunakan PID bagian kedua..... 53
Gambar 3.24c *Flowchart wall follower* kanan menggunakan PID bagian ketiga..... 54

Gambar 3.25a *Flowchart wall follower* kiri menggunakan PID bagian pertama**.....** 55

Gambar 3.25b *Flowchart wall follower* kiri menggunakan PID bagian kedua**.....** 56

Gambar 4.1 Posisi robot (proses variable) untuk kontroler *Proportional* dengan *reference* PWM 25 dan *set point* jarak (8 cm) **.....** 61

Gambar 4.2 Sinyal PWM motor DC kiri dari kontroler *Proportional* untuk *reference* PWM 25 dan *set point* jarak (8 cm) **.....** 62

Gambar 4.3 Posisi robot (proses variable) untuk kontroler *Proportional Integral* dengan *reference* PWM 25 dan *set point* jarak (8 cm) **.....** 63

Gambar 4.4 Sinyal PWM motor DC kiri untuk kontroler *Proportional Integral* dengan *reference* PWM 25 dan *set point* jarak (8 cm) **.....** 63

Gambar 4.5 Posisi robot (proses variable) untuk kontroler *Proportional Integral Derivative* dengan *reference* PWM 25 dan *set point* jarak (8 cm) **.....** 65

Gambar 4.6 Sinyal PWM motor DC kiri untuk kontroler *Proportional Integral Derivative* dengan *reference* PWM 25 dan *set point* jarak (8 cm) **.....** 65

Gambar 4.7 Posisi robot (proses variable) P (biru), PI (ungu) dan PID (hijau)

untuk *reference* PWM 25 dan *set point* jarak (8 cm) **.....** 66

Gambar 4.8 Posisi robot (proses variable) P (biru), PI (ungu) dan PID (hijau) untuk *reference* PWM 25 dan *set point* jarak (8 cm) (diperbesar/zoom) **.....** 67

Gambar 4.9 Posisi robot (proses variable) untuk kontroler PID pada uji arena KRCI tanpa *uneven floor* dan *furniture* untuk *reference* PWM 25 dan *set point* jarak (8 cm) **.....** 68

Gambar 4.10 Posisi robot (proses variable) untuk kontroler PID pada uji arena KRCI dengan *uneven floor* dan *furniture* untuk *reference* PWM 25 dan *set point* jarak (8 cm) **.....** 69

Gambar 4.11 Sinyal kontrol untuk kontroler PID pada uji arena KRCI tanpa *uneven floor* dan *furniture* untuk *reference* PWM 25 dan *set point* jarak (8 cm) **.....** 69

Gambar 4.12 Sinyal kontrol untuk kontroler PID pada uji arena KRCI dengan *uneven floor* dan *furniture* untuk *reference* PWM 25 dan *set point* jarak (8 cm) **.....** 70

Gambar 4.13 Posisi robot (proses variable) atau nilai bacaan sensor *ultrasonic* kontroler *Proportional* untuk *reference* PWM 50 dan *set point* jarak (8 cm) **.....** 72

Gambar 4.14 Sinyal PWM motor DC kiri untuk kontroler *Proportional* pada *reference* PWM 50 dan *set point* jarak (8 cm) **.....** 73

Gambar 4.15 Posisi robot (proses variable/nilai bacaan sensor *ultrasonic)* kontroler *Proportional Integral* untuk *reference* PWM 50 dan *set point* jarak (8 cm) **.....** 74

Gambar 4.16 Sinyal PWM motor DC kiri untuk kontroler *Proportional Integral* pada *reference* PWM 50 dan *set point* jarak (8 cm) **.....** 74

Gambar 4.17 Posisi robot (proses variable atau nilai bacaan sensor *Ultrasonic)* kontroler *Proportional Integral Derivative* untuk *reference* PWM 50 dan *set point* jarak (8 cm) **.....** 75

Gambar 4.18 Sinyal PWM motor DC kiri untuk kontroler PID dengan *reference* PWM 50 dan *set point* jarak (8 cm) **.....** 76

Gambar 4.19 Posisi robot (proses variable atau nilai bacaan sensor *ultrasonic)* kontroler P (ungu), PI (biru) dan PID (hijau) untuk *reference* PWM 50 dan *set point* jarak (8 cm) **.....** 77

Gambar 4.20 Sinyal PWM motor DC kiri untuk kontrol P (hijau), PI (ungu) dan PID (merah) pada *reference* PWM 50 dan *set point* jarak (8) **.....** 78

Gambar 4.21 Posisi robot (proses variable) untuk kontroler PID pada uji arena KRCI tanpa *uneven floor* dan *furniture* pada *reference* PWM 50 dan *set point* jarak (8cm) **.....** 79

Gambar 4.22 *Output* proses variabel untuk kontroler PID pada uji arena KRCI menggunakan *uneven floor* dan *furniture* pada *reference* PWM 50 dan *set point* jarak (8 cm) **.....** 80

Gambar 4.23 Sinyal kontrol untuk kontroler PID pada uji arena KRCI tanpa *uneven floor* dan *furniture* dengan *reference* PWM 50 dan *set point* jarak (8 cm) **.....** 80

Gambar 4.24 Sinyal kontrol untuk kontroler PID pada uji arena KRCI menggunakan *uneven floor* dan *furniture* dengan *reference* PWM 50 dan *set point* jarak (8 cm) **.....** 81

Gambar 4.26 Sinyal PWM motor DC kiri untuk kontroler *Proportional* dengan *reference* PWM 75 *set point* jarak (8 cm) **.....** 83

Gambar 4.27 *Output* proses variabel untuk kontroler *Proportional Integral* dengan *reference* PWM 75 dan *set point* jarak (8 cm) **.....** 84

Gambar 4.28 Sinyal PWM motor DC kiri untuk kontroler *Proportional Integral* dengan *reference* PWM 75 dan *set point* jarak (8cm) **.....** 84

Gambar 4.29 Posisi robot (proses variable) untuk kontroler *Proportional Integral Derivative* untuk *reference* PWM 75 dan *set point* jarak (8cm) **.....** 86

Gambar 4.30 Sinyal PWM motor DC kiri untuk kontroler *Proportional Integral Derivative* dengan *reference* PWM 75 dan *set point* jarak (8cm) **.....** 86

Gambar 4.31 Posisi robot (proses variable) kontroler P (hijau), PI (hitam) dan PID (merah) untuk *reference* PWM 75 dan *set point* jarak (8 cm) **.....** 87

Gambar 4.32 Sinyal PWM motor DC kontroler untuk P (hitam), PI (hijau) dan PID (biru) dengan *reference* PWM 75 *set point* jarak (8 cm) **.....** 88

Gambar 4.33 *Output* proses variabel pada uji arena KRCI tanpa *uneven floor* dan *furniture* untuk *reference* PWM 75 dan *set point* jarak (8 cm) **.....** 89

Gambar 4.34 *Output* proses variabel pada uji arena KRCI dengan *uneven floor* dan *furniture* untuk *reference* PWM 75 dan *set point* jarak (8 cm) **.....** 90

Gambar 4.35 Sinyal kontrol untuk kontroler PID pada uji arena KRCI tanpa *uneven floor* dan *furniture* untuk *reference* PWM 75 dan *set point* jarak (8 cm) **.....** 90

Gambar 4.36 Sinyal kontrol untuk kontroler PID pada uji arena KRCI dengan *uneven floor* dan *furniture* untuk *reference* PWM 75 dan *set point* jarak (8 cm) **.....** 91

DAFTAR TABEL

Halaman

Tabel 2.1 Pendekatan nilai parameter PID dengan metoda *oscillation*.. 19

Tabel 3.1 Tabel Kebenaran *Driver* Motor VNH3SP30 MD01B..... 33

Tabel 4.1a Tabel pengukuran sensor warna kiri depan..... 57

Tabel 4.1b Tabel pengukuran sensor warna kiri belakang..... 58

Tabel 4.2 Tabel pengukuran sensor *ultrasonic* SRF05..... 59

Tabel 4.3a Pengujian kontroler P, PI dan PID pada robot untuk memadamkan api (*reference* PWM 25 dan *set point* jarak (8cm)) 92

Tabel 4.3b Pengujian kontroler P, PI dan PID pada robot untuk memadamkan api (*reference* PWM 50 dan *set point* jarak (8cm)) 93

Tabel 4.3c Pengujian kontroler P, PI dan PID pada robot untuk memadamkan api (*reference* PWM 75 dan *set point* jarak (8cm)) 93

Tabel 4.3d Pengujian kontroler P, PI dan PID pada robot untuk memadamkan api (*reference* PWM 125 dan *set point* jarak (8cm)) 93

Tabel 4.4a Pengujian kontroler PID pada arena KRCI dengan posisi titik api di ruang 1 untuk *reference* PWM 125..... 94

Tabel 4.4b Pengujian kontroler PID pada arena KRCI dengan posisi titik api di ruang 2 untuk *reference* PWM 125**.....** 95

Tabel 4.4c Pengujian kontroler PID pada arena KRCI dengan posisi titik api di ruang 3 untuk *reference* PWM 125**.....** 95

Tabel 4.4d Pengujian kontroler PID pada arena KRCI dengan posisi titik api di ruang 4 untuk *reference* PWM 125**.....** 96

BAB I

PENDAHULUAN

Pada Bab ini berisi latar belakang masalah, rumusan masalah, tujuan penelitian, batasan masalah, spesifikasi alat yang digunakan, dan sistematika penulisan.

I.1 Latar Belakang

Dewasa ini penggunaan robot beroda pemadam api telah banyak dilombakan terutama dalam kegiatan Kontes Robot Cerdas Indonesia (KRCI) yang diadakan setiap satu tahun sekali, yang bertujuan untuk mendorong penguasaan teknologi maju bagi para mahasiswa teknik di Indonesia. Berbagai kategori diadakan termasuk *Fire Fighting* Robot *Contest* divisi beroda. Divisi Beroda Robot Cerdas Pemadam Api Beroda merupakan suatu divisi dimana robot menggunakan roda sebagai alat geraknya dengan misi mencari dan memadamkan api di arena lapangan berbentuk simulasi interior suatu rumah. Pada divisi ini yang diutamakan adalah kemampuan robot bernavigasi dan bermanuver serta kecepatan dalam menyelesaikan misinya tersebut. Robot yang berhasil menemukan dan memadamkan api tercepat dinyatakan sebagai pemenang.

Berbagai kelemahan belum dapat diatasi, diantaranya adalah penerapan sistem kontrol yang selama ini banyak mengandalkan sistem kontrol *On Off*. Untuk itu pada penelitian ini, dirancang robot beroda yang mempunyai dimensi yang lebih kecil dan sistem kontrol PID diterapkan untuk menyempurnakan gerakan robot.

I.2 Rumusan Masalah

Masalah-masalah yang akan dibahas dalam penelitian ini yaitu:

1. Bagaimana desain robot yang sesuai dengan arena KRCI?
2. Bagaimana robot beroda dapat bergerak lurus saat berada dilorong menggunakan pengontrol PID?
3. Bagaimana robot dapat bergerak diarena KRCI?

I.3 Tujuan

Tujuan dari penelitian ini, yaitu:

1. Merealisasikan robot beroda yang sesuai untuk arena KRCI
2. Mendapatkan nilai parameter kontroler PID untuk robot beroda agar robot dapat berjalan lurus saat berada di lorong
3. Mendapatkan algoritma yang sesuai untuk menyelesaikan misi di arena KRCI

I.4 Batasan Masalah

Batasan masalah dalam penelitian ini, yaitu:

1. Nilai bacaan sensor *ultrasonic* yang dipengaruhi suhu ruangan dan sudut pantulan pada dinding.

I.5 Spesifikasi Alat yang Digunakan

Alat dan bahan yang digunakan dalam penelitian ini, yaitu:

1. Atmega128A
2. UV-Tron
3. Sensor *ultrasonic*
4. *Sound Activation*
5. Motor DC
6. Motor *Brushless*

7. Servo
8. Driver motor DC
9. Sensor Warna ZX-03
10. LCD 16x2
11. Kipas
12. Batere lippo 4 *cell* dan 2 *cell*
13. Regulator 12V
14. Relay

I.6 Sistematika Penulisan

Laporan penelitian ini disusun dengan sistematika penulisan sebagai berikut:

- Bab I Pendahuluan

Pada bab ini berisi latar belakang masalah, rumusan masalah, tujuan, batasan, masalah, spesifikasi alat yang digunakan, dan sistematika penulisan.

- Bab II Landasan Teori

Pada bab ini berisi teori-teori penunjang, yaitu Mikrokontroler teori PID, ATMega128A, Sensor *ultrasonic*, sensor UV-Tron, motor DC, motor *brushless*, *relay*, sensor garis, servo dan komunikasi serial mikrokontroler dengan laptop.

- Bab III Perancangan dan Realisasi

Pada bab ini dijelaskan tentang perancangan sistem robot beroda pemadam api menggunakan kontroler PID, komunikasi mikrokontroler dengan Matlab, perancangan robot beroda pemadam api, jenis-jenis sensor yang dipakai dan algoritma pemrograman robot beroda pemadam api.

- Bab IV Data Pengamatan dan Analisis Data

Pada bab ini dijelaskan tentang proses pengambilan data pengamatan, pengujian kemampuan robot beroda pemadam api, dan analisisnya.

- Bab V

Pada bab ini berisi kesimpulan dan saran-saran yang perlu dilakukan untuk perbaikan di masa mendatang.

BAB II

TEORI DASAR

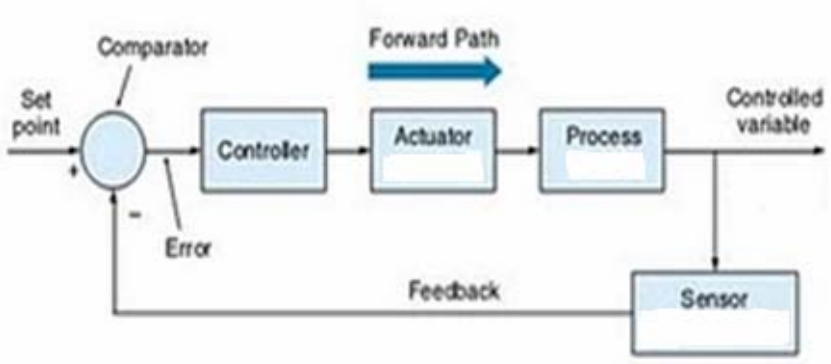
Pada Bab II ini akan dijelaskan mengenai teori tentang kontroler, kontroler PID, dan landasan teori penunjang lain yang diperlukan untuk merancang dan merealisasikan sistem kontrol dalam robot.

II.1 Kontroler

Fungsi dari kontroler secara umum adalah mengendalikan sistem dengan memanipulasi sinyal *error*, sehingga respon sistem (*output*) sama dengan yang diinginkan. Sistem kontrol dapat dibagi menjadi dua, yaitu sistem kontrol tertutup (*closed loop*) dan sistem kontrol terbuka (*open loop*).

II.1.1 *Feedback* Kontrol Sistem (*Closed Loop*)[10]

Sistem kontrol umpan balik merupakan sistem yang menggunakan hubungan antara *output* dan *input* yang diinginkan dengan cara membandingkannya. Dengan sistem kontrol umpan balik, keberadaan gangguan yang menyebabkan *output* menyimpang dari *input* yang diinginkan dapat diantisipasi.

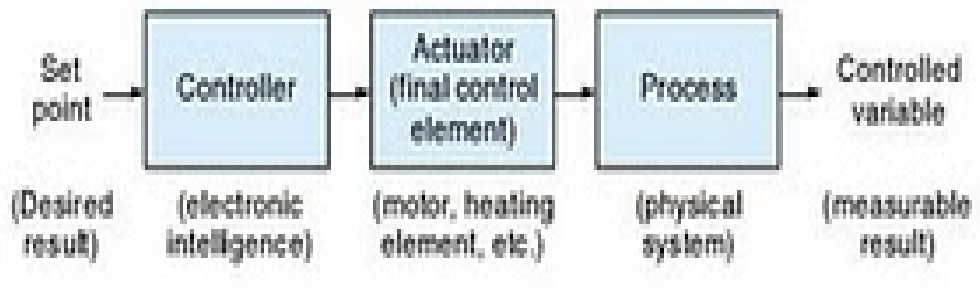


Gambar 2.1 Diagram blok sistem kontrol tertutup

Gambar 2.1 adalah diagram blok dari sistem kontrol umpan balik yang banyak digunakan di industri. Pengontrol akan mendeteksi sinyal *error* (deviasi antara *output* dan *setpoint*). Kontroler memproses sinyal *error* dan menghasilkan sinyal aktuasi yang merupakan aksi kontrol sebagai tanggapan dari *error* tadi. Aksi kontrol menggerakkan *actuator* dan diterapkan pada *plant/*proses sehingga dihasilkan *output.* Elemen sensor akan melihat atau mengukur hasil *output* dan mengkonversikannya ke variabel yang sesuai dengan *input* referensi. Kedua variabel ini dibandingkan dan menghasilkan sinyal *error.* Interaksi ini akan berlangsung terus sampai didapatkan kondisi bahwa *error* menjadi minimum.

II.1.2 Sistem Kontrol Terbuka (*Open Loop*) [10]

Sistem kontrol terbuka adalah sistem dimana *output*-nya tidak memengaruhi aksi kontrol. Pada sistem ini tidak dilakukan perbandingan antara sinyal *output* dan *input*. Performansi dan akurasi dari aksi kontrol sistem ini tergantung dari kalibrasi sistem. Jika terdapat gangguan maka sistem tidak dapat mengantisipasinya sehingga harus dikalibrasi ulang. Gambar 2.2 adalah diagram blok sistem kontrol terbuka.



Gambar 2.2 Diagram blok sistem kontrol terbuka

II.2 Tanggapan Peralihan (*Transient Respon)* [11]

Dalam beberapa kasus praktis, karakteristik performansi sistem kontrol dinyatakan dalam domain waktu. Sistem yang mempunyai elemen penyimpanan energi tidak dapat merespon secara seketika dan akan menunjukkan respon transien jika diberi masukan atau gangguan. Karakteristik performansi sistem kontrol ini dinyatakan dalam bentuk respon transien terhadap masukan *unit step*.

Respon transien dalam sistem kontrol praktis sering menunjukkan osilasi teredam sebelum mencapai keadaan stabil. Berikut adalah parameter karakteristik respon transien terhadap masukan *unit step*:

1. *Delay time* (td) adalah waktu yang diperlukan respon untuk mencapai setengah harga akhir yang pertama kali.

2. *Rise time* (tr) adalah waktu yang diperlukan respon untuk naik dari 10% sampai 90%, 5% sampai 95% atau 0% sampai 100% dari harga akhirnya.

3. *Peak time* (tp)*)* adalah waktu yang diperlukan respon untuk mencapai puncak lewatan yang pertama kali.

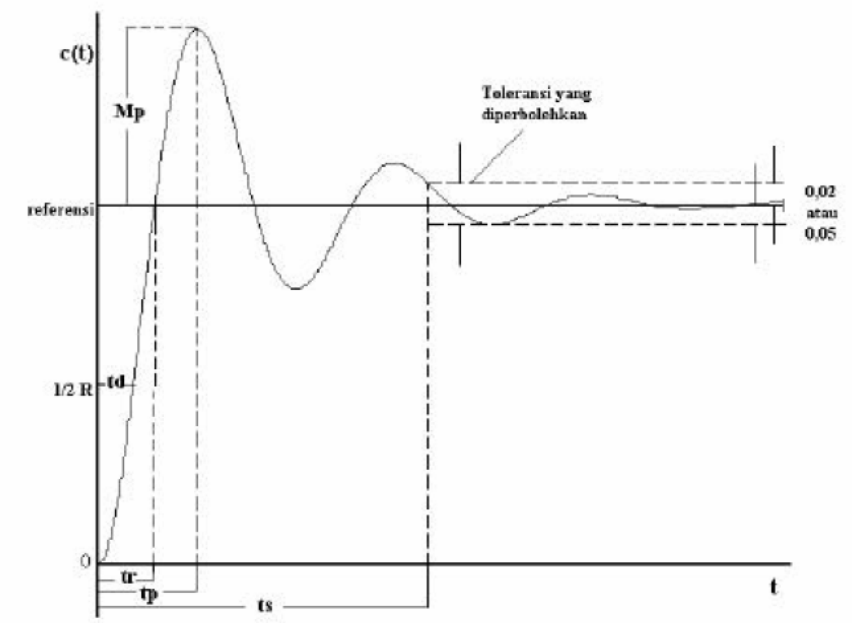
4. *Maximum overshoot* (Mp) adalah persen harga puncak maksimum dari kurva respon yang diukur dari satu. Jika harga keadaan tunak tidak sama dengan satu, maka biasanya digunakan persen lewatan maksimum yang didefinisikan seperti Persamaan 2.1.

$$Mp = \frac{c(tp) - c(\infty)}{c(\infty)} \times 100\% \qquad \textbf{(2.1)}$$

   Besarnya (persen) lewatan maksimum secara langsung menunjukkan kestabilan relative sistem.

5. *Settling time* (ts) adalah waktu yang diperlukan kurva respon untuk mencapai dan menetap dalam daerah disekitar harga akhir yang ukurannya ditentukan

dengan presentase mutlak dari harga akhir (biasanya 2% atau 5%).



Gambar 2.3 Kurva tanggapan sistem dengan karakteristik *transient respon*

II.3 Aksi Kontrol [10]

Aksi kontrol dikenal juga dengan sinyal kontrol yang beraksi berdasarkan *error.* Aksi kontrol ini berusaha mereduksi *error*. Beberapa aksi kontrol dasar yang akan dibahas diantaranya aksi kontrol *on-off* , aksi kontrol *Proportional* (P) , aksi kontrol *Integral* (I) , aksi kontrol *Derivative* (D) , dan aksi kontrol PID.

II.3.1 Aksi Kontreol *On-Off* [10]

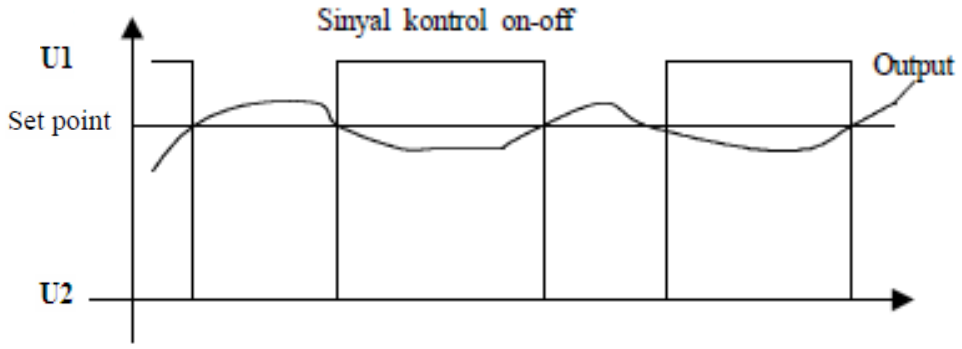
Sistem kontrol ini mempunyai dua posisi yang tetap yaitu *on* atau *off*. Kontrol *on-off* ini banyak digunakan di industri karena murah dan sederhana. Sinyal kontrol akan tetap pada satu keadaan dan akan berubah ke keadaan lainnya bergantung pada nilai *error* positif atau negatif.

u(t) = sinyal kontrol

e(t) = sinyal *error*

u(t) = U1, untuk **e(t) >0** dan **u(t) = U2,** untuk **e(t) <0**

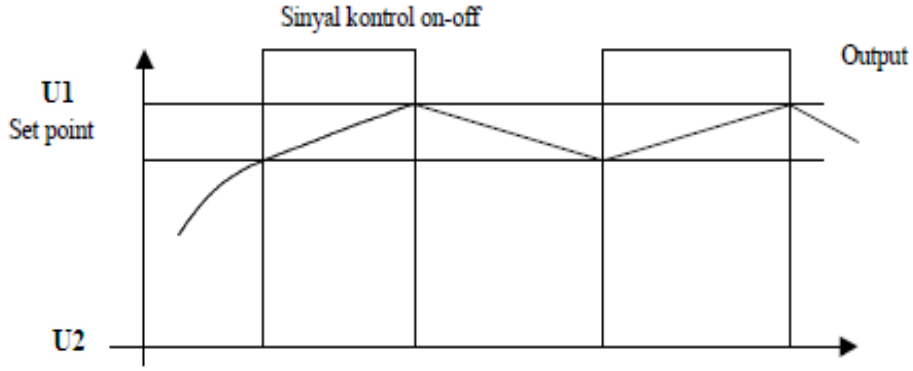
Kontroler dua posisi pada umumnya dijumpai pada komponen elektrik (relay) dan komponen pneumatik (katup dan silinder). Ilustrasi dari *kontroler on-off* dapat dilihat pada Gambar 2.4.



Gambar 2.4 Ilustrasi aksi kontrol *on-off*

Dari Gambar 2.4 dapat diamati bahwa jika *output* lebih besar dari *set point*, *actuator* akan *off. Output* akan turun dengan sendirinya sehingga menyentuh *set point* lagi. Pada saat itu, sinyal kontrol akan kembali *on* (*actuator on*) dan mengembalikan *output* kepada *set point*. Demikian seterusnya sinyal kontrol dan *actuator* akan *on-off* terus menerus.

Kelemahan dari kontroler *on-off* ini adalah jika *output* berosilasi di sekitar *set point* (keadaan yang memang diinginkan) akan menyebabkan *actuator* bekerja keras untuk *on-off* dengan frekuensi yang tinggi. Hal ini akan menyebabkan kontroler akan cepat aus dan memakan energi yang banyak (boros). Untuk sedikit mengatasi hal ini maka dibuat suatu *band* pada *set point* sehingga mengurangi frekuensi *on-off* dari kontroler. Ilustrasi kontroler *on-off* dengan histerisis dapat dilihat pada Gambar 2.5.

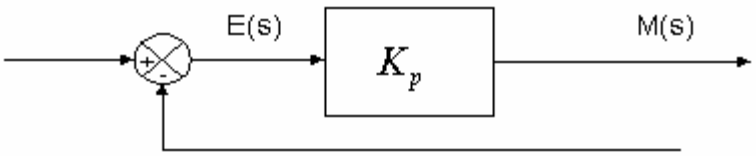


Gambar 2.5 Ilustrasi aksi kontrol *on-off* dengan histerisis

Sinyal kontrol akan *off* ketika *output* menyentuh batas atas dan baru *on* kembali ketika menyentuh batas bawah. *Band* dari *set point* ini disebut juga *diferensial gap*. Dengan keadaan seperti ini serta mengatur besarnya *diferensial gap* maka frekuensi *on-off* dapat dikurangi tetapi berdampak dengan menurunan-nya akurasi terhadap *set point.*

II.3.2 Aksi Kontrol *Proportional* (P)

Proportional kontrol memberi pengaruh sebanding dengan *error.* Semakin besar *error* maka sinyal kontrol yang dihasilkan semakin besar. [11] Gambar 2.6 adalah gambar diagram blok sistem kontrol *Proportional*.



Gambar 2.6 Diagram blok sistem kontrol *Proportional*

Pada keadaan tunak, keluaran sistem dengan pengendali proportional tidak akan sama dengan referensinya. Dengan kata lain pada pengendali *Proportional*

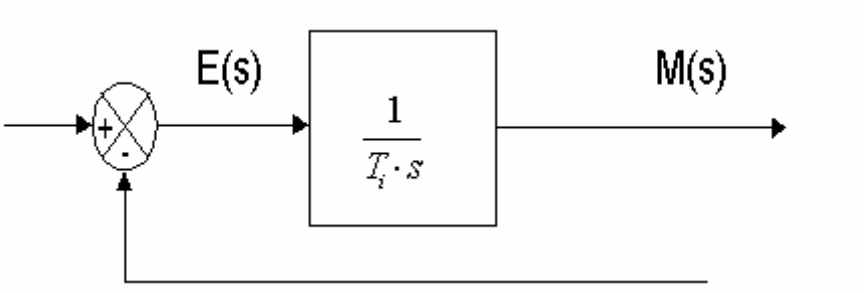
masih terdapat *offset* pada keadaan tunak. *Offset* dapat dihilangkan dengan memberikan harga Kp mendekati tak hingga (Kp ➔ ∞). Akan tetapi hal ini tidak mungkin terjadi, karena harga Kp mempunyai batas maksimal tertentu dan jika diberikan suatu harga Kp melebihi batas maka keluaran akan berosilasi.

Pengaruh kontrol *Proportional* pada suatu sistem, yaitu: [8]

1. Menambah atau mengurangi kestabilan.
2. Memperbaiki *transient respon* khususnya: *rise time* dan *settling time.*
3. Mengurangi (bukan menghilangkan) *error steady state*.

II.3.3 Aksi Kontrol *Integral* (I)

Integral kontrol memberi pengaruh sesuai dengan perubahan *error*, semakin besar *error* maka semakin besar juga sinyal kontrol yang dihasilkan.[11] Kontrol *Integral* memiliki karakteristik seperti halnya sebuah *Integral*. Keluaran kontroler sangat dipengaruhi oleh perubahan yang sebanding dengan nilai sinyal kesalahan. Keluaran kontroler ini merupakan akumulasi yang terus menerus dari perubahan masukannya. Gambar 2.7 adalah diagram blok dari sistem kontrol *Integral*.

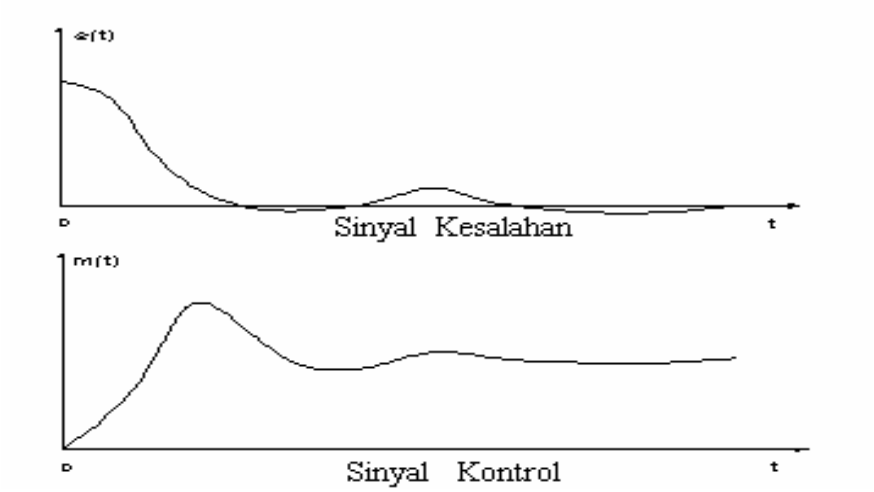


Gambar 2.7 Diagram blok sistem kontrol *Integral*

Jika sinyal kesalahan tidak mengalami perubahan, keluaran akan menjaga keadaan seperti sebelum terjadinya perubahan masukan. Sinyal keluaran pengontrol *Integral* merupakan luas bidang yang dibentuk oleh kurva kesalahan

penggerak. Sinyal keluaran akan berharga sama dengan harga sebelumnya ketika sinyal kesalahan berharga nol. Konstanta *Integral* K_i yang berharga besar akan mempercepat hilangnya *offset*. Tetapi semakin besar nilai konstanta K_i akan mengakibatkan peningkatan osilasi dari sinyal keluaran kontroler. Offset biasanya terjadi pada plant yang tidak mempunyai faktor integrasi $\frac{1}{s}$.

Kurva sinyal kesalahan *error* (e(t)) terhadap waktu (t) dan kurva masukan (u(t)) terhadap waktu (t) pada pembangkit kesalahan nol diperlihatkan pada Gambar 2.8.



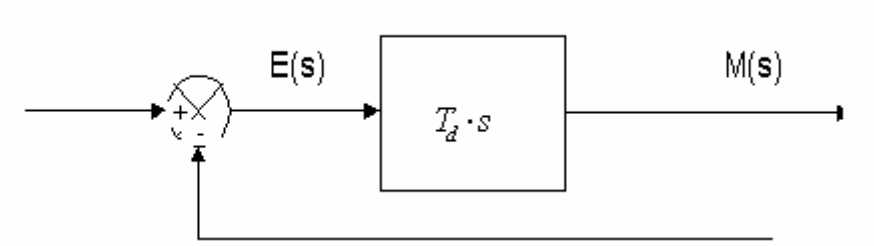
Gambar 2.8 Kurva sinyal kesalahan e(t) terhadap t dan kurva u(t) terhadap t pada pembangkit kesalahan nol

Pengaruh pengontrol *Integral* pada suatu sistem, yaitu: [8]

1. Menghilangkan *error steady state.*
2. Dapat memperbaiki *transient respon*.
3. Dapat menambah ketidakstabilan sistem

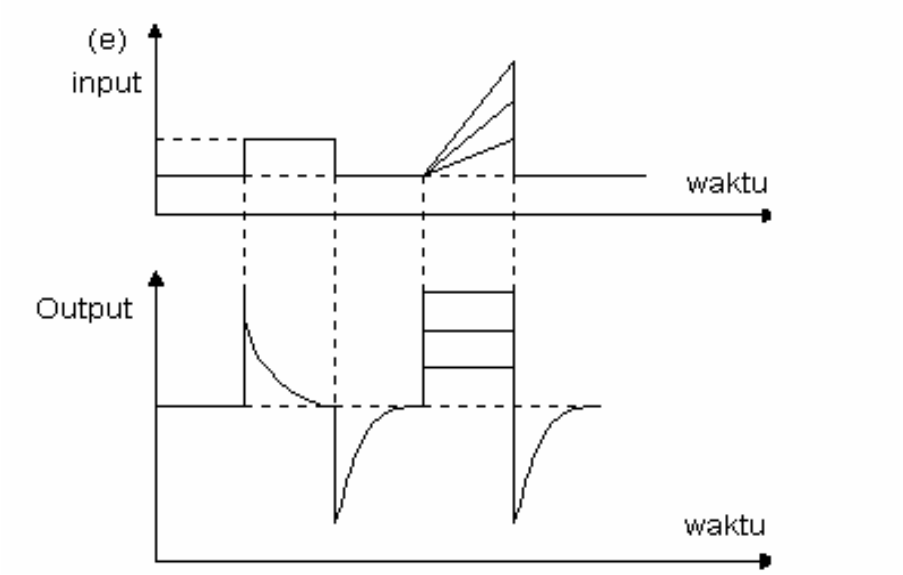
II.3.4 Aksi Kontrol *Derivative* (D)

Derivative kontrol memberi pengaruh terhadap besarnya sinyal kontrol yang dihasilkan sebanding dengan perubahan *error*. [11]Semakin cepat *error* berubah, maka semakin besar sinyal kontrol yang dihasilkan. Keluaran kontroler *diferensial* memiliki sifat seperti halnya suatu operasi *Derivative*. Perubahan yang mendadak pada masukan kontroler, akan mengakibatkan perubahan yang sangat besar.



Gambar 2.9 Diagram blok sistem kontrol *Derivative*

Gambar 2.9 menunjukkan diagram blok yang menggambarkan hubungan antara sinyal kesalahan dengan *output* kontroler.



Gambar 2.10 Kurva waktu hubungan kontroler *input-output diferential*

Gambar 2.10 menyatakan hubungan antara sinyal masukan dengan sinyal keluaran kontroler *diferensial*. Ketika masukannya tidak mengalami perubahan,

keluaran kontroler juga tidak mengalami perubahan, sedangkan apabila sinyal masukan berubah mendadak dan menaik (berbentuk fungsi *step*), keluaran menghasilkan sinyal berbentuk *impuls*. Jika sinyal masukan berubah naik secara perlahan (fungsi *ramp*), keluarannya justru merupakan fungsi *step* yang besar magnitudnya sangat dipengaruhi oleh kecepatan naik dari fungsi *ramp* dan faktor konstanta diferensial $T_{d.}$

Kontroler *Derivative* dapat juga disebut sebagai pengendali laju, karena keluaran kontroler sebanding dengan laju perubahan sinyal kesalahan penggerak. kontroler *Diferential* mempunyai suatu karakter untuk mendahului, sehingga kontroler ini dapat menghasilkan koreksi yang signifikan sebelum pembangkit kesalahan menjadi sangat besar. Jadi kontroler *Diferential* dapat mengantisipasi pembangkit kesalahan, memberikan aksi yang bersifat korektif, dan cenderung meningkatkan stabilitas sistem.

Kontroler *Derivative* ini tidak dapat berdiri sendiri, karena kontroler ini hanya aktif pada waktu transien. Pada waktu transien, pengendali *Derivative* menyebabkan redaman pada sistem sehingga memperkecil lonjakan (*overshoot*).

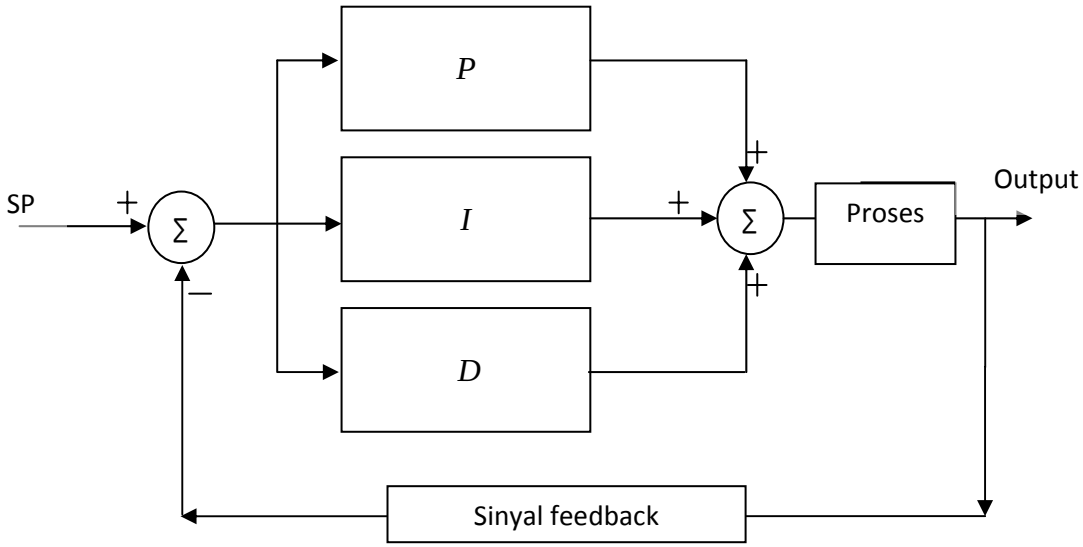
Pengaruh kontroler D terhadap suatu sistem, yaitu: [8]

1. Memberikan efek redaman terhadap sistem yang berosilasi
2. Memperbaiki *transient respon*.
3. Kontroler *Derivative* tidak dapat digunakan sendiri, karena hanya memberikan sinyal kontrol saat terjadi perubahan *error*, sehingga jika ada *error* yang tetap kontroler tidak memberikan aksi.

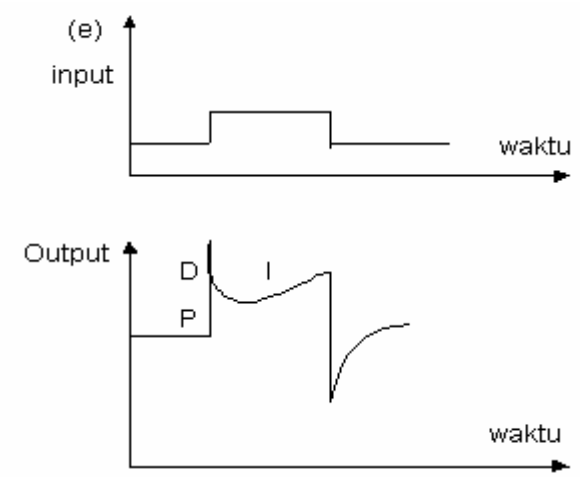
II.3.5 Aksi Kontrol PID

Kontroler PID merupakan kontroler yang cukup kompleks dengan kombinasi kontroler P, kontroler I, dan kontroler D. Kontroler PID dapat menghilangkan *error steady state* dan membuat respon sistem menjadi lebih cepat.

Gambar 2.15 adalah diagram blok PID[13] dan dalam fungsi waktu antara sinyal keluaran dengan masukan untuk kontroler PID dapat dilihat pada Gambar 2.12. [4]



Gambar 2.11 Diagram blok PID *parallel*

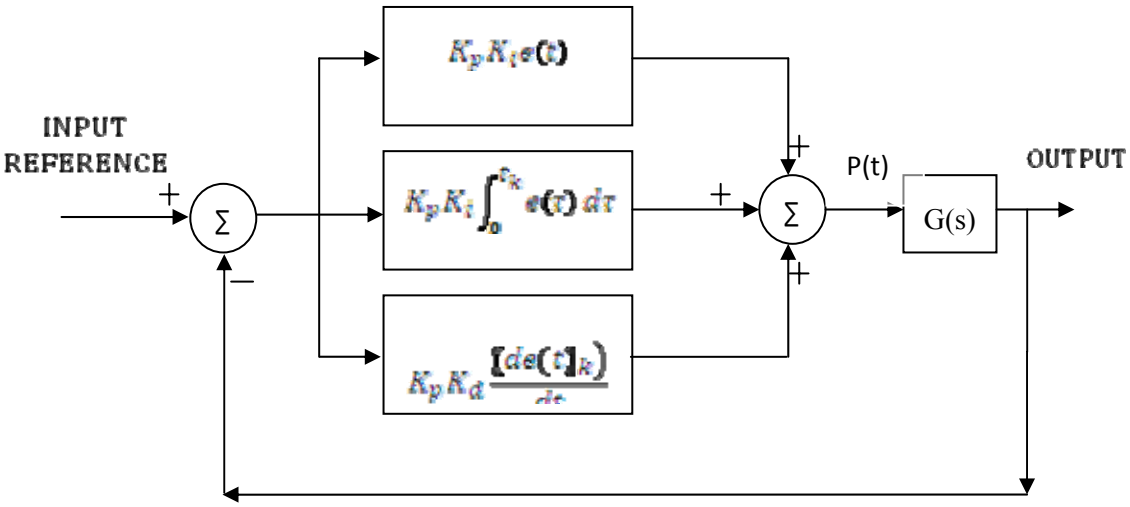


Gambar 2.12 Hubungan dalam fungsi waktu antara sinyal keluaran dengan masukan untuk kontroler PID

PID dapat dikelompokkan menjadi 2 jenis yaitu:

✓ PID dengan nilai Kp yang di pasang secara seri dengan konstanta *Proportional*, *Integral* dan *Derivative*nya, diagram blok jenis PID ini dapat dilihat pada Gambar 2.13 dengan Persamaan 2.3. [13]

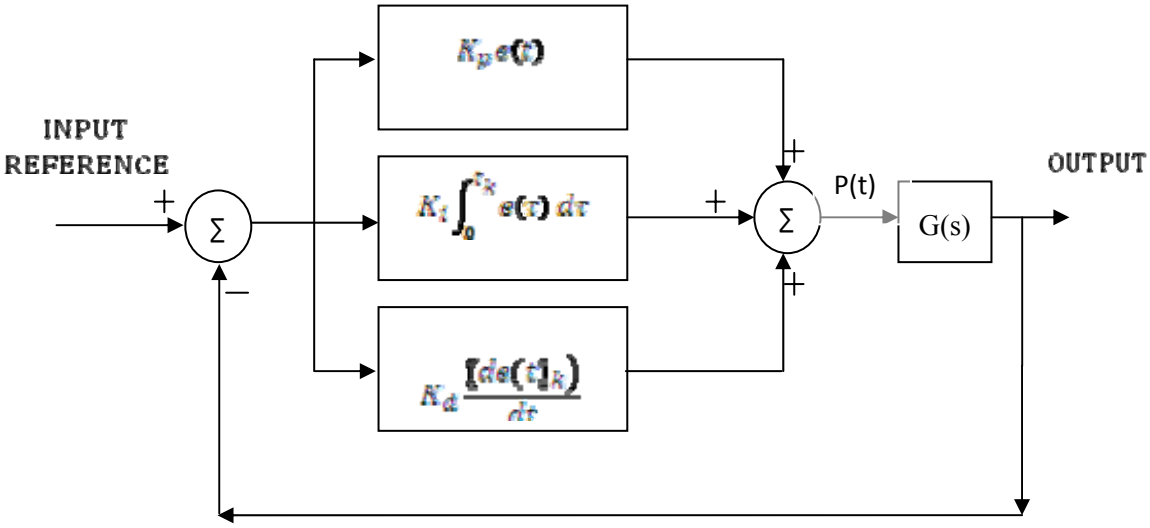
$$p(t) = K_p e_p + K_p K_i \int_0^{t_k} e(\tau)\, d\tau + K_p K_d \frac{d[e(t)_k)}{dt} \qquad \textbf{(2.3)}$$



Gambar 2.13 PID dengan hubungan Kp diseri Kd dan Ki

✓ PID lainnya adalah dengan nilai Kp yang dipasang hanya pada konstanta *Proportional*nya saja, diagram blok PID jenis ini dapat dilihat pada Gambar 2.14 dengan Persamaan dapat dilihat pada Persamaan 2.4. [11]

$$p(t) = K_p e_p + K_i \int_0^{t_k} e(\tau)\, d\tau + K_d \frac{d[e(t]_k)}{dt} \qquad \textbf{(2.4)}$$



Gambar 2.14 PID dengan Kp terpisah dengan Kd dan Ki

Walaupun sistem ini sedikit berbeda, akan tetapi secara matematis sistem ini memiliki fungsi alih yang sama.

Karena dalam realisasinya menggunakan mikroprosesor yang bekerja secara diskrit maka Persamaan 2.3 dan Persamaan 2.4 tentunya harus didiskritisasi. Dalam perancangannya digunakan Persamaan 2.4 sebagai acuan rumus PID yang digunakan. Proses diskritisasi dapat dilihat pada Persamaan 2.5 dan Persamaan 2.6. [6]

$$\int_0^{t_k} e(\tau)\, d\tau \Rightarrow \sum_{i=1}^{k} e(t_i)\Delta t \qquad \textbf{(2.5)}$$

$$\frac{de(t_k)}{dt} \Rightarrow \frac{[e(t]_k) - [e(t]_{k-1})}{\Delta t} \qquad \textbf{(2.6)}$$

Maka Persamaan 2.4 setelah didiskritisasi akan menjadi seperti Persamaan 2.7 [6].

$$p(t_k) = K_p e_k + K_i \sum_{i=1}^{k} e(t_i)\Delta t + K_d \frac{[e(t]_k) - [e(t]_{k-1})}{\Delta t} \qquad \textbf{(2.7)}$$

Di mana :

$$\Delta t = sampling\ \ time$$

II.4 Tuning PID

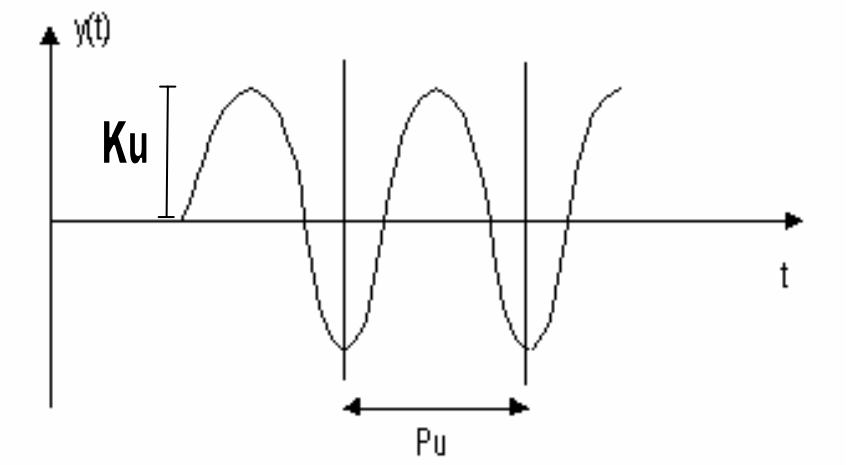
Tuning PID yang digunakan adalah metoda *trial and error yang* dimodifikasi menggunakan metoda Ziegler-Nichols II. Berikut adalah masing-masing metode sebelum dimodifikasi yaitu metoda trial and error dan metoda Ziegler-Nichols II.

II.4.1 Ziegler Nichols II (Metoda Osilasi)

Metode ini didasarkan pada reaksi sistem *closed loop*. Plant disusun serial dengan kontroler PID. Semula parameter integrator dibuat tak berhingga dan parameter diferensial dibuat nol ($T_i = \sim$;$T_d = 0$). Parameter proporsional kemudian dinaikkan bertahap. Mulai dari nol sampai mencapai harga yang mengakibatkan reaksi sistem tepat berosilasi (*Sustain oscillation*).

Nilai penguatan proportional pada saat sistem mencapai kondisi *sustain oscillation* disebut *ultimate gain* K_u. Periode dari *sustained oscillation* disebut *ultimate period* P_u (dapat dilihat pada Gambar 2.15).

Penalaan parameter PID didasarkan terhadap kedua konstanta hasil eksperimen, Ku dan Pu. Ziegler dan Nichols menyarankan penyetelan nilai parameter K_p, T_i, dan T_d berdasarkan rumus yang diperlihatkan pada Tabel 2.1.



Gambar 2.15 Kurva respon *sustain oscillation*

Tabel 2.1 Pendekatan nilai parameter PID dengan metoda *oscillation*

Tipe Kontroler	K_p	Ti	Td
P	$0,5.K_u$		
PI	$0,45.K_u$	$1/2\ P_u$	
PID	$0,6.K_u$	$0,5\ P_u$	$0,125\ P_u$

II.4.2 *Trial and Error*

Metoda *trial and error* disebut juga metoda coba-coba. Metoda ini umumnya cocok digunakan untuk beberapa sistem kontrol, salah satu diantaranya adalah sistem kontrol PID. Dalam merancang sistem kontrol PID *output* yang diharapkan dari metoda *trial and error* harus memenuhi beberapa kriteria seperti[21]:

1. Memiliki *rise time* dan *settling time* yang cepat.
2. Tidak memiliki *steady state error*.

3. *Overshoot* sekecil mungkin.

Langkah-langkah metode *trial and error*:

1. Menentukan nilai Kp yang keluarannya mendekati *set point* dengan nilai Ti=∞ dan nilai Td=0

2. Setelah diperoleh nilai Kp yang sesuai, tentukan nilai Ti agar diperoleh keluaran tunak yang lebih mendekati *set point.*

3. Menentukan nilai Td agar sistem lebih peka terhadap *error*.

II.5 Pengantar Robotika

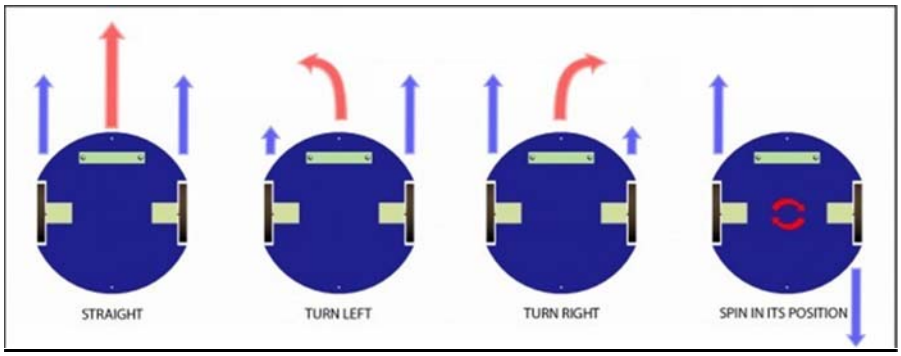
Sebelum merancang dan merealisasikan robot pemadam api diperlukan beberapa pengertian tentang robot. Pada bagian ini akan dijelaskan beberapa hal yang berhubungan dengan robotika.

II.5.1 Robot Beroda[2]

Robot beroda dapat dibagi menurut sistem penggeraknya, yaitu sistem gerak *differential drive, tricycle drive, synchronous drive* dan *holonomic drive.* Jenis konfigurasi sistem gerak robot akan mempengaruhi kemampuan manuver, pengendalian, dan stabilitas robot.

II.5.1.1 *Diffential Drive*

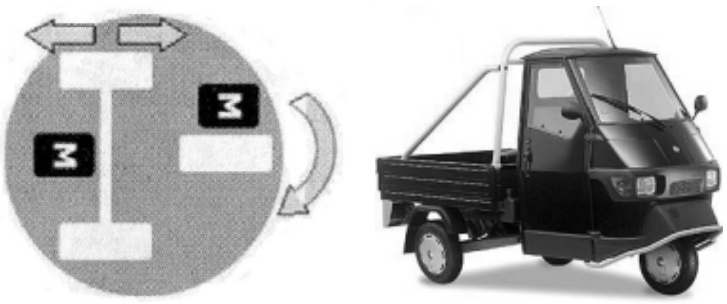
Sistem gerak *differential drive* terdiri dari dua buah roda yang terpasang pada kiri dan kanan robot. Sistem ini memungkinkan robot berputar di tempat dengan cara memutar motor kiri dan kanan dengan arah berlawanan. Ilustrasi sistem gerak *differential drive* dapat dilihat pada Gambar 2.16.



Gambar 2.16 Sistem gerak *differential drive*

II.5.1.2 *Tricycle Drive*

Tricycle drive merupakan sistem gerak dengan tiga buah roda. Dua buah roda dengan satu poros dihubungkan pada sebuah motor penggerak, sedangkan sebuah roda diberlakukan sebagai kemudi yang dapat berputar (setir kemudi), ketika berbelok akan didapatkan radius sepanjang titik pertemuan antara roda depan dengan roda belakang. Ilustrasi sistem gerak *tricycle drive* dapat dilihat pada Gambar 2.17.



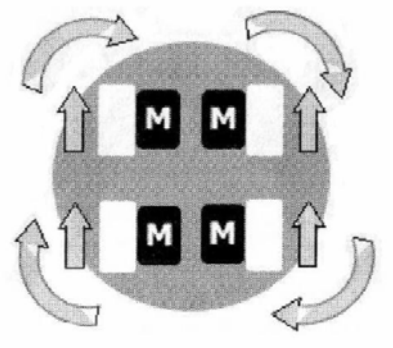
a. **b.**

Gambar 2.17 a.Sistem Gerak *Tricycle Drive* dan b. Kendaraan dengan Sistem Gerak *Tricycle Drive*

II.5.1.3 *Synchronous Drive*

Synchronous drive adalah sistem yang menggunakan semua roda yang terdapat pada robot untuk dapat bergerak. Dengan sistem gerak ini, robot menjadi

lebih stabil. Pada saat robot berjalan pada permukaan yang tidak rata, maka roda yang terpengaruh pada ketidakrataan permukaan akan didukung oleh roda yang tidak terpengaruh, sehingga robot dapat bergerak dengan arah yang tetap. Ilustrasi sistem gerak *synchronous drive* dapat dilihat pada Gambar 2.18.

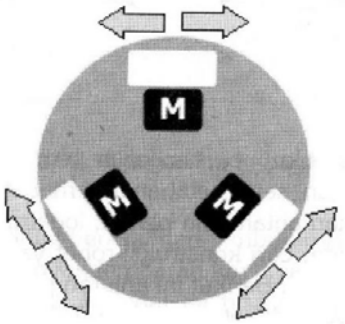


Gambar 2.18 Sistem gerak *synchronous drive*

II.5.1.4 *Holonomic Drive*

Holonomic drive adalah sistem gerak yang memungkinkan robot bergerak ke segala arah (dengan penggunaan roda *omni-directional),* hal tersebut ditunjukan pada Gambar 2.19. Konfigurasi ini memungkinkan gerakan rotasi dan translasi pada *mobile robot.* Ilustrasi sistem gerak *holonomic drive* dapat dilihat pada Gambar 2.20.

Gambar 2.19 Robot dengan 3 roda *omni-directional*



Gambar 2.20 Sistem gerak *Holomonic Drive*

II.6 Sensor

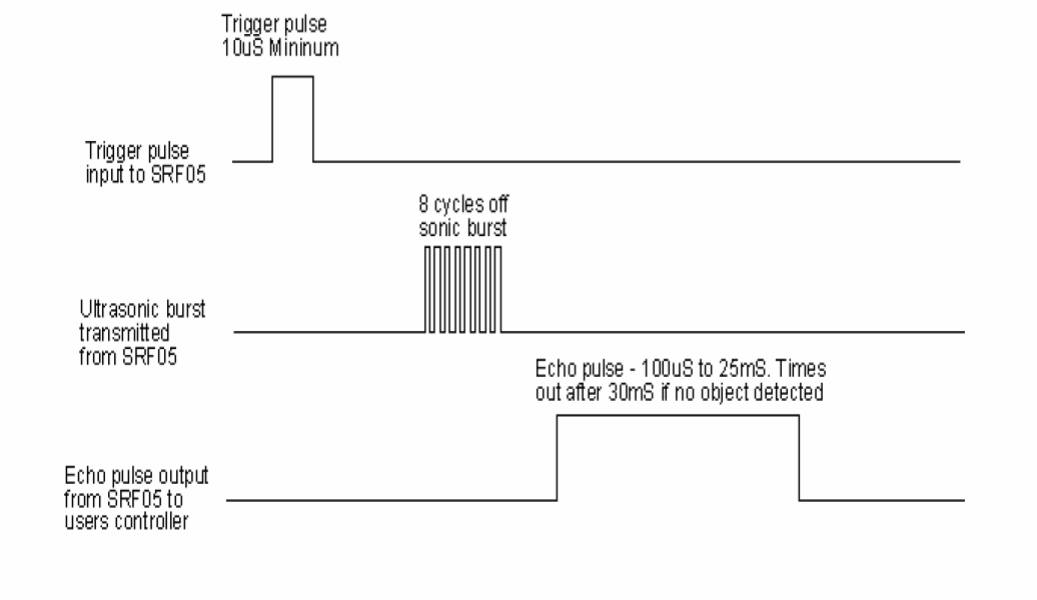
Sensor adalah piranti yang mengubah suatu nilai fisik (*input*) menjadi nilai fisik yang lain (*output*). *Output* yang dihasilkan biasanya berupa sinyal listrik. Pada penelitian ini digunakan beberapa jenis sensor yaitu sensor jarak ultrasonik, sensor keberadaan api (UVTron), dan sensor warna.

II.6.1 Sensor Jarak Ultrasonik (SRF05)[19]

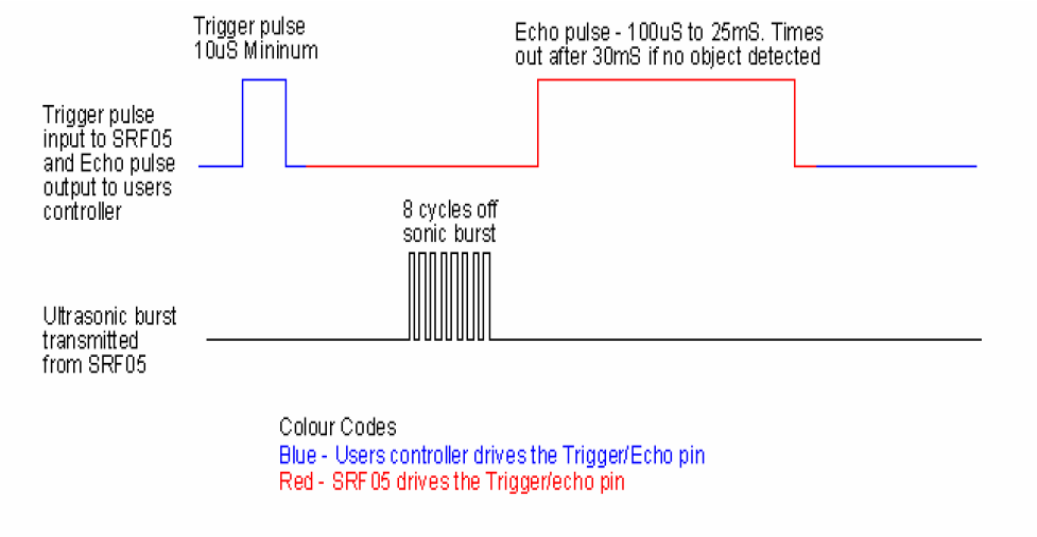
Sensor jarak ultrasonik sangat cocok dipakai untuk aplikasi-aplikasi yang perlu dilakukan pengukuran jarak. Selain itu, sensor ini juga bisa bermanfaat dalam sistem *security* dan dapat digunakan sebagai alternatif pengganti sensor *proximity*. Sensor jarak ultrasonik yang dipakai dalam penelitian ini adalah sensor SRF05. Sensor SRF05 memiliki kemampuan membaca jarak dari 3 cm sampai 4 meter. Sensor SRF05 memiliki 2 *mode* untuk penggunaannya.

Pada Gambar 2.21a dan Gambar 2.21b menunjukkan diagram waktu sensor SRF05 untuk masing-masing *mode*. Untuk mulai menghitung jarak dengan menggunakan SRF05, hanya perlu memberikan pulsa *trigger* minimal 10μS. Kemudian SRF05 akan memancarkan 8 siklus gelombang ultrasonik (40kHz). Sensor SRF05 mengeluarkan pulsa *output high* pada jalur *echo* (atau jalur trigger pada mode 2) setelah memancarkan gelombang ultrasonik. Setelah gelombang pantul terdeteksi, sensor SRF05 mengeluarkan pulsa *output low* pada jalur *echo*

(atau jalur *trigger* pada *mode* 2). Lebar pulsa *high* pada jalur *echo* akan *propotional* dengan jarak kepada objeknya. Maka jarak dalam centimeter dapat dihitung dengan cara lebar pulsa *high* dibagi 29.



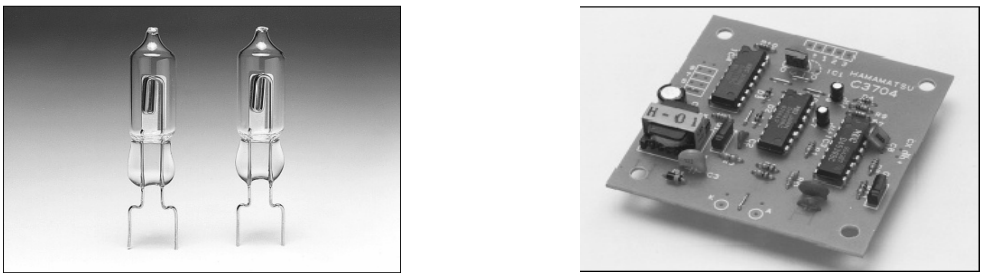
Gambar 2.21a Diagram waktu sensor SRF05 *mode* 1



Gambar 2.21b Diagram waktu sensor SRF05 *mode* 2

II.6.2 Sensor UVTron[18]

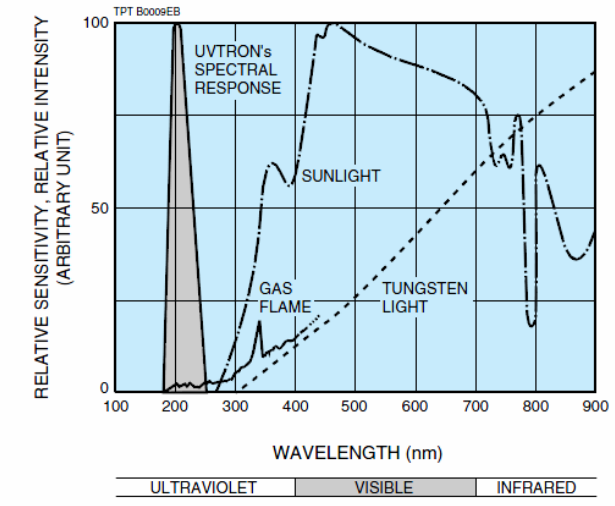
Sensor UVTron digunakan untuk mendeteksi keberadaan api di dalam ruang. Sensor ini bekerja dengan cara mendeteksi ada tidaknya panas api. Sensor ini memberikan sinyal aktif apabila mendeteksi adanya api dalam ruangan. Tipe sensor yang dipakai adalah Hamamatsu *Flame Sensor* UVTron R2868 yang berbentuk bohlam dan dilengkapi dengan rangkaian untuk mengaktifkannya (UVTron *driving circuit*), seperti pada Gambar 2.22a dan Gambar 2.22b.



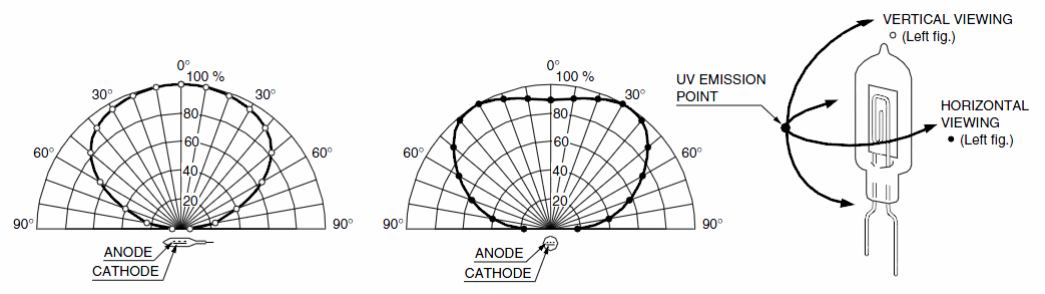
a. **b.**

Gambar 2.22 Sensor Api (a. UVTron R2868 dan b. Rangkaian Pengaktif)

Prinsip kerja Hamamatsu R2868 adalah mendeteksi adanya gelombang ultraviolet yang dihasilkan oleh api, yaitu pada *range* 185-260nm, di area tersebut adalah sinar ultraviolet yang dihasilkan oleh api, ditunjukkan pada Gambar 2.23. Daerah deteksi Hamamatsu R2868, seperti pada Gambar 2.24.



Gambar 2.23 *Spektrum* respon UVTron



Gambar 2.24 Derajat sensitivitas Hamamtasu R2868

Hamamatsu R2868 memerlukan sebuah modul rangkaian untuk mengaktifkannya. Modul ini bernama UVTron *Driving Circuit* dengan seri C3704-02. Bekerja pada tegangan 5-30 volt dengan konsumsi arus sebesar 300 mikro *Ampere* yang akan menghasilkan keluaran digital logika *high* jika terdeteksi api dan logika *low* jika tidak ada api. Dalam perlakuannya, Hamamatsu R2868 tidak boleh dipegang secara langsung oleh tangan manusia karena dapat mengurangi sensitivitas dari sensor api (UVTron) tersebut.

II.6.3 Sensor Warna[19]

Sensor warna *infrared* adalah sensor yang digunakan untuk mendeteksi warna dari lantai robot berada. Sensor warna *infrared* yang digunakan dalam perancangan robot ini adalah ZX-03 *infrared reflector sensor* yang menggunakan TCRT5000 *reflector sensor* sebagai jantung rangkaian.

Sensor TCRT5000 memiliki karakteristik sebagai berikut:

1. Tipe detektor fototransistor.
2. Dimensi sensor 10,2 mm x 5,8 mm x 7 mm.
3. Jarak operasi optimum 2,5 mm.
4. Jangkauan operasi 0,2 mm – 15 mm.
5. Arus keluaran 1 mA.
6. Filter cahaya tampak.
7. Panjang gelombang cahaya 950 nm.

8. Tegangan masukan 5V.
9. Arus keluaran maksimum 100 mA.

Bentuk sensor TCRT5000 ditunjukkan pada Gambar 2.25a., sedangkan koneksi pin ZX-03 ditunjukkan pada Gambar 2.25b.

a. **b.**

Gambar 2.25 Sensor Warna (a. Bentuk Sensor TCRT5000 dan b. Koneksi Pin ZX-03)

II.7 Pengenalan Mikro[3]

Pengontrol mikro dapat diartikan sebagai pengontrol dalam ukuran mikro. Secara umum pengontrol mikro dapat diartikan sebagai komputer dalam sebuah chip. Berbagai unsur seperti prosesor, memori, *input*, dan *output* terintegrasi dalam satu kemasan yang berukuran kecil. Pengontrol mikro membutuhkan daya yang rendah, murah, dan mudah didapatkan. Pengontrol mikro beroperasi dengan kecepatan detak (*clocking*) *megahertz* atau kurang.

II.7.1 Pengenalan ATMEL AVR RISC[3]

Salah satu pengontrol mikro yang banyak digunakan saat ini yaitu pengontrol mikro AVR. AVR adalah pengontrol mikro RISC (*Reduce Instruction Set Computing*) 8 bit berdasarkan arsitektur Harvard, yang dibuat oleh Atmel pada

tahun 1996. AVR mempunyai kepanjangan *Advanced Versatile* RISC atau *Alf and Vegard's RISC processor* yang berasal dari nama penemunya, dua mahasiswa Norwegian Institute of Technology (NTH), yaitu Alf-Egil Bogen dan Vegard Wollan.

AVR memiliki keunggulan dibandingkan dengan pengontrol mikro lain, keunggulan utama pengontrol mikro AVR adalah memiliki laju putaran eksekusi program yang lebih cepat karena sebagian besar instruksi dieksekusi dalam satu siklus *clock*, lebih cepat dibandingkan dengan pengontrol mikro MCS51 yang memiliki arsitektur CISC (*Complex Instruction Set Computing*) karena pengontrol mikro MCS51 membutuhkan 12 siklus *clock* untuk mengeksekusi 1 instruksi. Selain itu pengontrol mikro AVR memiliki fitur yang lengkap (ADC *Internal*, EEPROM *Internal*, *Timer/Counter*, *Watchdog Timer*, PWM, *Port* I/O, komunikasi serial, komparator, I2C, dll.), sehingga dengan fasilitas yang lengkap ini, pemrogram dapat menggunakannya untuk berbagai aplikasi sistem elektronika seperti robot, otomasi industri, peralatan telekomunikasi, dan berbagai keperluan lain. Secara umum pengontrol mikro AVR dapat dikelompokkan menjadi tiga kategori, yaitu keluarga AT90Sxx, ATmega, dan ATtiny.

II.7.2 Pengontrol Mikro ATmega 128

Pengontrol mikro ATmega 128 adalah salah satu pengontrol mikro yang dibuat oleh Atmel Corporation, industri yang bergerak di bidang manufaktur semikonduktor. Pengontrol mikro ini termasuk dalam keluarga AVR 8-bit RISC yang dikategorikan dalam kelas ATmegaAVR. Angka 128 pada ATmega 128 menandakan bahwa pengontrol mikro ini memiliki kapasitas memori flash sebesar 128 KiloByte.

BAB III

PERANCANGAN DAN REALISASI

Pada Bab III ini dijelaskan tentang perancangan sistem robot beroda pemadam api menggunakan kontroler PID, komunikasi mikro dengan Matlab, perancangan robot beroda pemadam api, jenis-jenis sensor yang dipakai dan algoritma pemrograman robot beroda pemadam api.

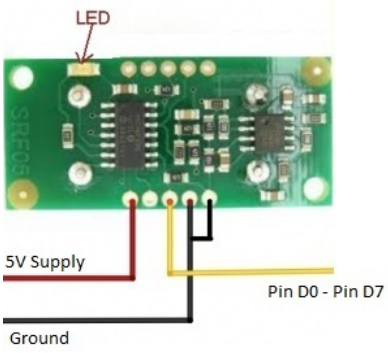
III.1 Sensor dan *Driver* Motor DC

Pada penelitian ini sensor-sensor yang akan dibahas diantaranya sensor jarak ultrasonic SRF05, sensor api UV-TRON, sensor warna ZX-03, dan rangkaian *driver* motor DC VNH3SP30 MD01B.

III.1.1 Sensor Jarak Ultrasonic SRF05

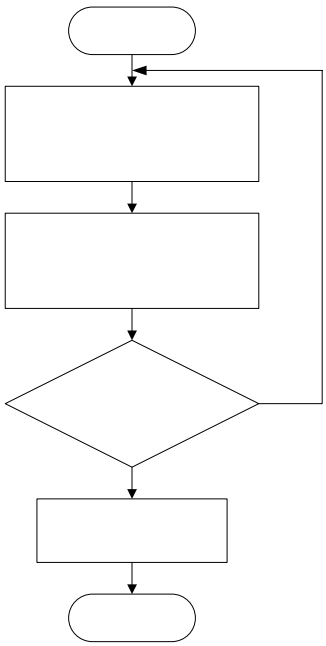
Pada Gambar 3.1 ditunjukkan gambar alokasi pin-pin pada sensor SRF05. Sensor SRF05 mempunyai 5 buah pin. Pada penelitian ini, sensor *ultrasoni*c SRF05 menggunakan *mode* 2, sehingga hanya empat buah pin yang digunakan, yaitu sebagai pin *ground* (Vss), pin tegangan 5V (Vdd), pin sinyal (I/O pin), dan pin *mode*. Pin *mode* harus dihubungkan dengan pin *ground*. Kemudian pin *ground* dan pin tegangan 5 Vdc dihubungkan ke sumber tegangan, sedangkan pin sinyal dihubungkan ke pengontrol mikro. SRF05 yang digunakan berjumlah 8 buah. Setiap sensor memakai 1 buah pin *input* pada ATmega 128 yaitu pin A0–A7.

Masing-masing sensor memiliki fungsi tersendiri. Sensor depan berfungsi untuk mendeteksi benda yang berada di depan robot, sensor kiri depan digunakan sebagai acuan nilai *error* yang dihasilkan, sensor kiri tengah, kanan depan, dan kanan tengah digunakan saat robot bermanuver di dalam *maze* dengan menggunakan metoda *wall follower*.



Gambar 3.1 Alokasi pin sensor SRF05

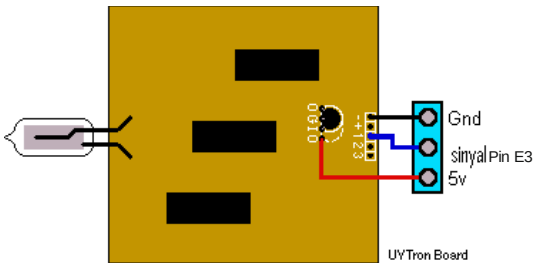
Pada Gambar 3.2 ditunjukkan diagram alir sensor SRF05. Program ini dibuat berdasarkan cara kerja sensor SRF05. Kecepatan gelombang ultrasonik yang dipancarkan sama dengan kecepatan suara yaitu 344 m/s (pada keadaan suhu ruangan).



Gambar 3.2 Diagram alir penggunaan sensor SRF05

III.1.2 Sensor Api UV-TRON Hamamatsu R2868

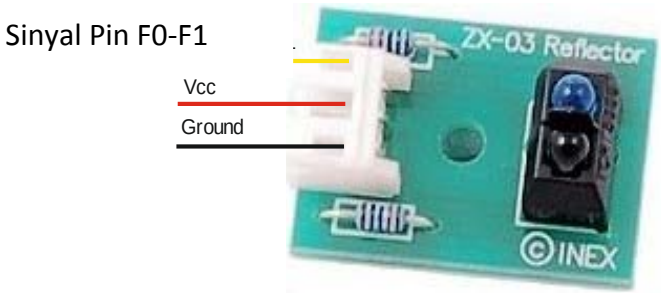
Pada Gambar 3.3 adalah alokasi pin-pin pada sensor UVTron Hamamatsu R2868. Pin 1 digunakan sebagai sinyal dan *supply* yang digunakan adalah 5 Volt. Pin E3 digunakan untuk menerima *output* dari sensor UVTron.



Gambar 3.3 Alokasi pin UVTron Hamamatsu R2868

III.1.3 Sensor Warna ZX-03

Pada Gambar 3.4 menunjukkan alokasi pin sensor warna ZX-03. Sensor ini memiliki 3 buah pin. Pin F0 digunakan untuk menerima *output* dari sensor warna belakang dan F1 digunakan untuk menerima *ouput* dari sensor warna depan.

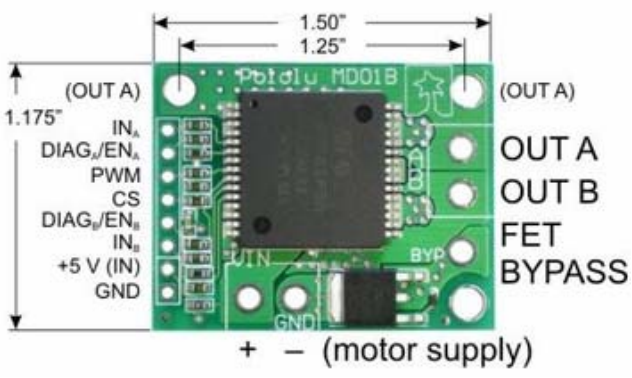


Gambar 3.4 Alokasi pin sensor warna

III.1.4 Rangkaian *Driver* Motor DC VNH3SP30 MD01B

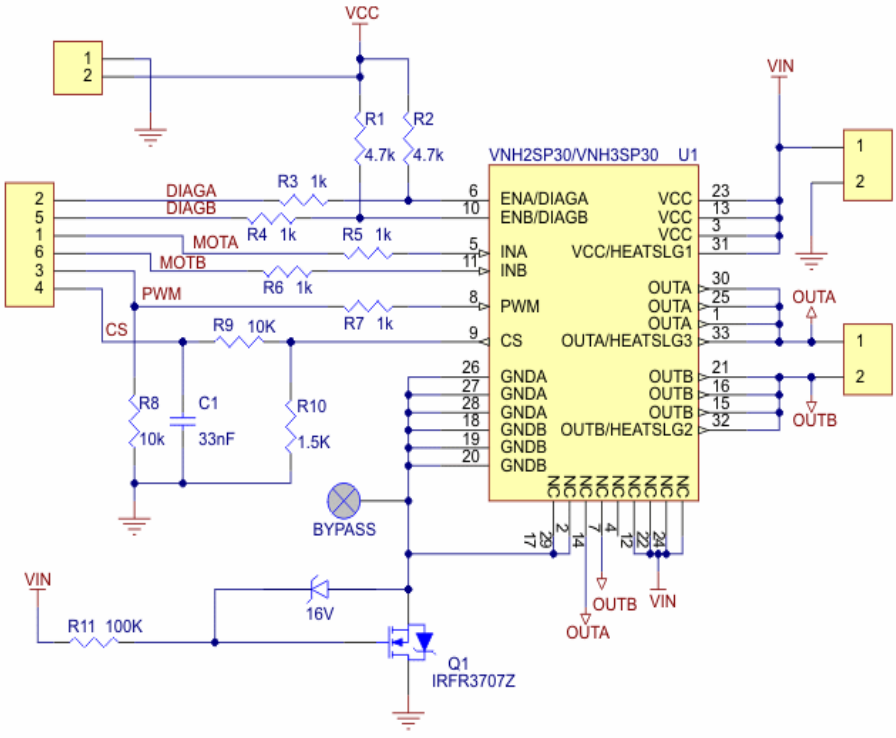
Dalam menggerakkan sebuah motor DC, diperlukan *driver* sehingga motor DC dapat berputar searah dan berlawanan jarum jam. Dalam hal ini digunakan

Driver Motor VNH3SP30 MD01B seperti ditunjukan Gambar 3.5, VNH3SP30 MD01B digunakan sebagai motor *driver*. Dengan motor *driver* ini, mengontrol motor dapat lebih mudah karena hanya memberikan *input* berupa bit – bit logika pada *port output* mikrokontroler.



Gambar 3.5 *Driver* motor DC VNH3SP30 MD01B

Rangkaian *driver* motor VNH3SP30 MD01B yang mempunyai spesifikasi 30A, 40V berisikan rangkaian H-Bridge. Pada Gambar 3.6 ditunjukkan rangkaian dalam motor *driver*.



Gambar 3.6 Rangkaian dalam motor *driver*

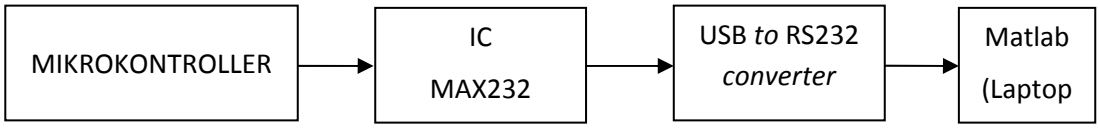
Dalam penggunaannya pin *out* A dan pin *out* B sebagai *output* dihubungkan langsung dengan motor DC. Pin INA dan INB merupakan *input* dari keluaran pengontrol mikro yang mengendalikan arah perputaran dari motor DC. Pin PWM berupa *input* pulsa PWM yang berfungsi sebagai pengontrol kecepatan motor DC. Pin *supply* motor digunakan sebagai *supply* motor DC, dalam hal ini digunakan *supply* sebesar 12volt dc, dan pin GND sebagai *ground.* Pin $DIAG_A/EN_A$ dan $DIAG_B/EN_B$ diberi resistor *pull up* dan dihubungkan ke Vcc. Pada Tabel 3.1 ditunjukkan tabel kebenaran *driver* motor VNH3SP30 MD01B.

Tabel 3.1 Tabel Kebenaran *Driver* Motor VNH3SP30 MD01B

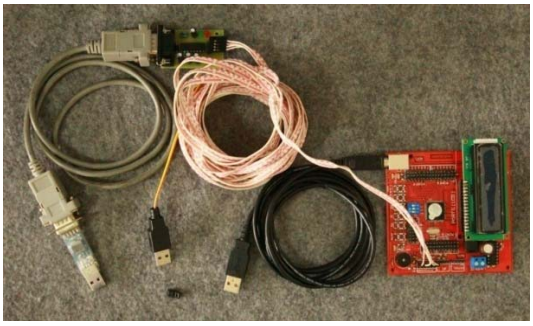
IN_A	IN_B	$DIAG_A/EN_A$	$DIAG_B/EN_B$	OUT_A	OUT_B	Comment
1	1	1	1	H	H	Brake to V_{CC}
1	0	1	1	H	L	Clockwise
0	1	1	1	L	H	Counter cw
0	0	1	1	L	L	Brake to GND

III.2 Penghubung Matlab Pada Laptop dengan Mikrokontroler

Untuk menghubungkan Matlab dengan mikrokontroler dibutuhkan beberapa rangkaian untuk mengubah level tegangan TTL ke level tegangan RS232 yang dimiliki oleh PC. Pada laptop yang tidak dilengkapi dengan *port* serial, dibutuhkan sebuah *converter* dari USB ke RS232 atau yang biasa disebut USB *to* RS232 *converter*. Gambar 3.7 dan Gambar 3.8 menunjukkan gambar dan diagram blok penghubung Matlab dengan mikrokontroler.



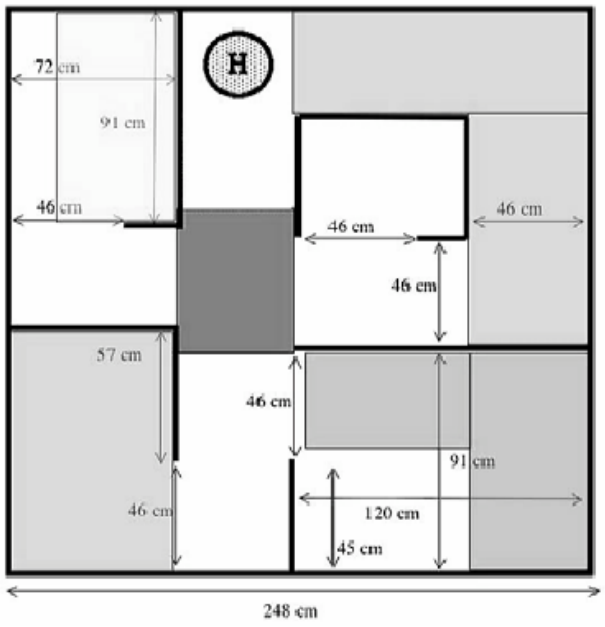
Gambar 3.7 Diagram blok penghubung Matlab pada laptop dengan mikrokontroler



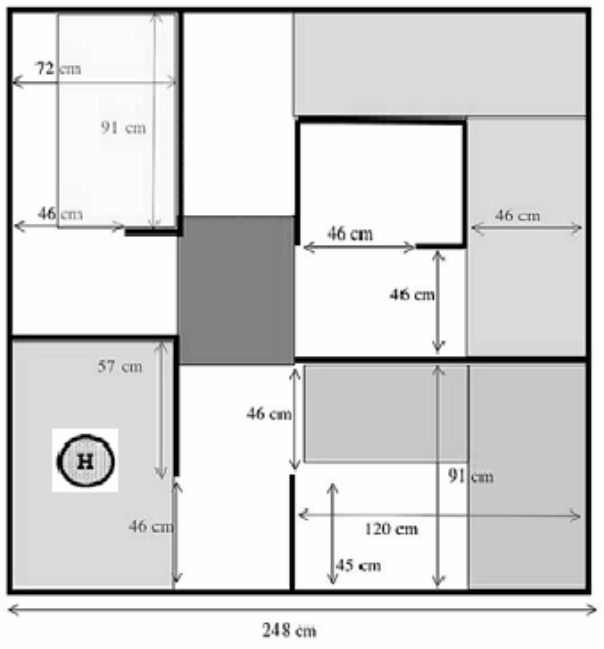
Gambar 3.8 Penghubung Matlab pada laptop dengan mikrokontroler

III.3 Perancangan Sistem Robot Beroda dengan Sistem Kontrol PID

Pada bagian perancangan sistem robot beroda dengan sistem kontrol PID harus menyelesaikan tugasnya sebagai robot permadam. Berdasarkan peletakkannya, ada 2 jenis *home*, yaitu *home* yang diletakkan di lorong dengan posisi tetap disebut *mode Non-Arbitrary Start* yang ditunjukkan pada Gambar 3.9a dan *home* yang diletakkan di dalam salah satu ruang dengan posisi acak disebut *mode Arbitrary Start* seperti yang ditunjukkan pada Gambar 3.9b.



Gambar 3.9a *Mode non-arbitary start*



Gambar 3.9b *Mode arbitary start*

Robot beroda dengan sistem kontrol PID dirancang agar robot dapat melakukan tugas-tugas, yaitu:

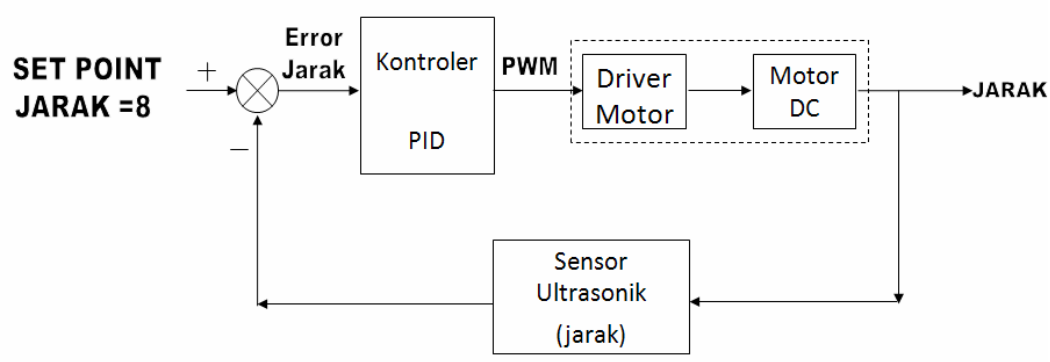
1. *Start* awal dengan *sound activation*.
2. Robot dapat bergerak dengan baik menggunakan sistem kontrol PID untuk menelusuri lorong-lorong lurus dan mengelilingi semua ruang yang ada dalam *maze* dengan menggunakan semua fitur yang tersedia, seperti *variable door location, arbitary start*, *hanging objects* (cermin dan *sound damper*), *uneven floor*, dan *furniture*.
3. Mendeteksi api, mencari posisi api, dan memadamkannya.

Diagram blok sistem kontrol robot terdiri atas sistem *loop* tertutup dan sistem *loop* terbuka. Pendeteksian warna lantai yang dilalui robot dengan menggunakan sensor warna merupakan sistem *loop* terbuka. Sedangkan sistem *loop* tertutup digunakan pada beberapa diagram blok, yaitu:

1. Diagram blok sistem kontrol gerak robot beroda.
2. Diagram blok sistem pemadaman api robot beroda.

III.3.1 Diagram Blok Sistem Kontrol Gerak Robot Beroda

Pada sistem ini digunakan sistem *loop* tertutup. Pada sistem *loop* tertutup, nilai *output* sistem akan dikoreksi kembali terhadap referensi *input*nya *(set point).* Diagram blok sistem ditunjukkan pada Gambar 3.10.



Gambar 3.10 Diagram blok sistem gerak robot beroda

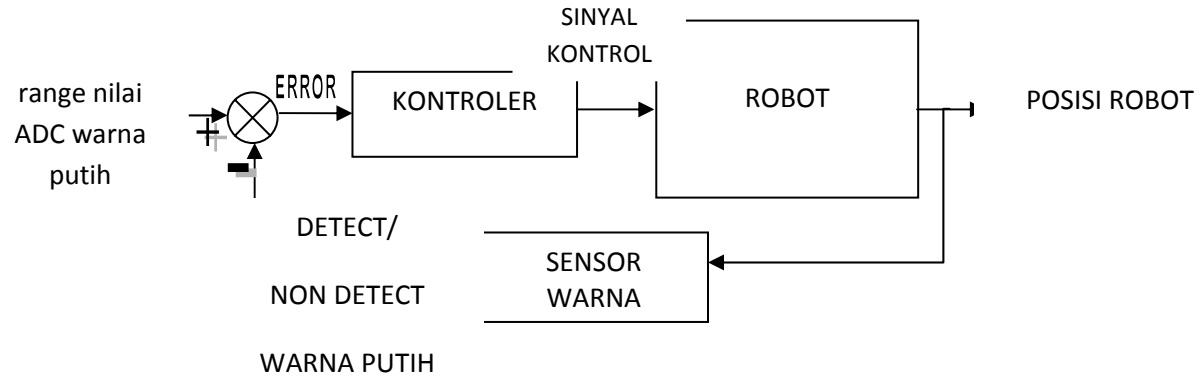
Sistem gerak robot beroda mengandalkan sensor *ultrasonic* sebagai sensor jarak dan untuk menghindari halangan-halangan yang ada. Sistem gerak robot ini adalah merupakan sistem kontrol *loop* tertutup, dengan s*et point* berupa nilai jarak antara robot dengan dinding. *Feedback* negatif dari sistem ini berupa sinyal yang dikirim dari sensor *ultrasonic* dan dikonversi menjadi nilai dalam besaran panjang/jarak. Sensor *ultrasonic* akan mengukur jarak robot dengan dinding dan halangan di sekitarnya. Jarak yang diukur yaitu jarak dinding depan, kiri, dan kanan. Masing-masing pengukuran jarak menggunakan satu sensor *ultrasonic*.

Pada sistem kontrol PID ini ada 2 pokok sensor yang digunakan sebagai acuan yaitu sensor depan dan kiri. Sensor depan dan kiri digunakan karena robot bergerak menyelusuri dinding kiri. Sensor kiri merupakan *feedback* negatif yang selanjutkan akan dibandingkan dengan *set point*, dan sensor depan merupakan referensi untuk navigasi belok kanan saat bagian kiri robot tertutup dinding/halangan. Selisih *feedback* dan *set point* adalah nilai *error* dari sistem yang kemudian diproses didalam pengontrol mikro ATmega 128 yang sudah diprogram dengan algoritma PID. Keluaran pengontrol akan diteruskan sebagai sinyal *input driver* motor DC untuk menggerakan/mengatur kecepatan *actuator*

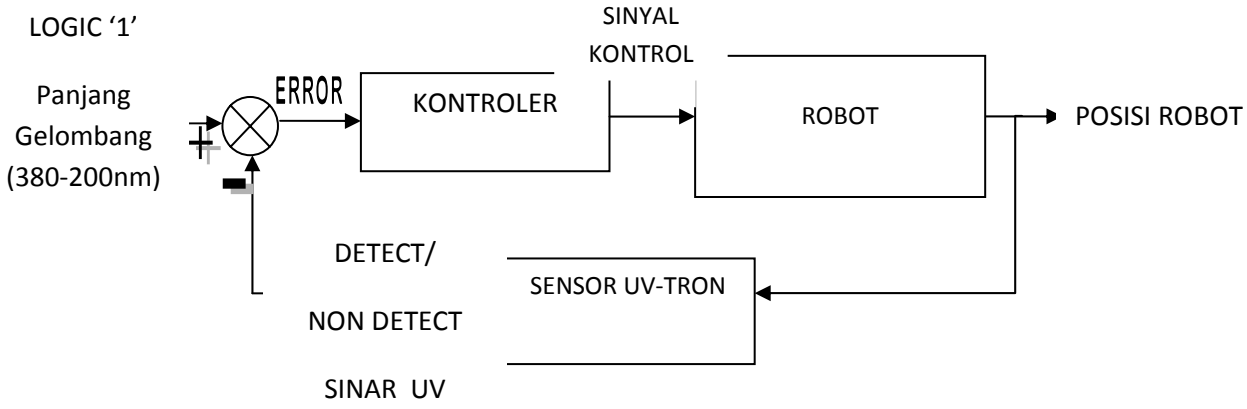
(motor DC). *Output* yang diinginkan dari sistem ini berupa jarak dan posisi robot dalam keadaan *steady* yang diukur sensor *ultrasonic*.

III.3.2 Diagram Blok Sistem Pemadam Api Robot Beroda

Sistem pemadaman api robot beroda mengandalkan sensor UVTron sebagai sensor pendeteksi api. Sistem ini menggunakan sistem *loop* tertutup. Diagram blok sistem ini ditunjukkan pada Gambar 3.11 dan Gambar 3.12.

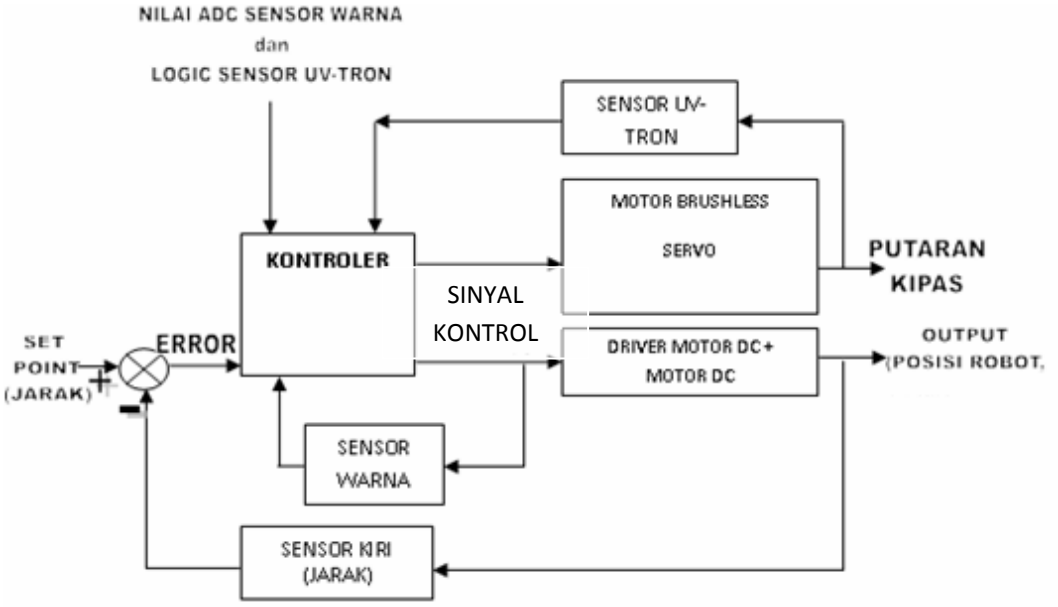


Gambar 3. sor warna sistem robot beroda pemadam



Gambar 3.12 Diagram blok sistem pemadam api robot beroda

Diagram blok sistem robot beroda dapat dilihat pada Gambar 3.13.



Gambar 3.13 Diagram blok sistem robot beroda

Input sistem deteksi api berupa sinyal yang dikirim oleh sensor api ke pengontrol mikro dengan *logic* 1 atau 0, *logic* 1 menandakan tidak ada api sedangkan *logic* 0 menandakan api terdeteksi. Jika sinyal yang dikirim menandakan adanya api, pengontrol mikro akan mengatur pergerakan motor DC dan mengaktifkan sensor warna, sehingga sistem akan mulai mencari posisi api dan memadamkan api. Sensor warna akan mendeteksi warna lantai yang dilalui robot. Ketika sensor warna mendeteksi warna putih, maka pengontrol mikro akan menghentikan gerak motor DC dan mengaktifkan motor *brushless* dan motor servo.

III.4 Bentuk Robot

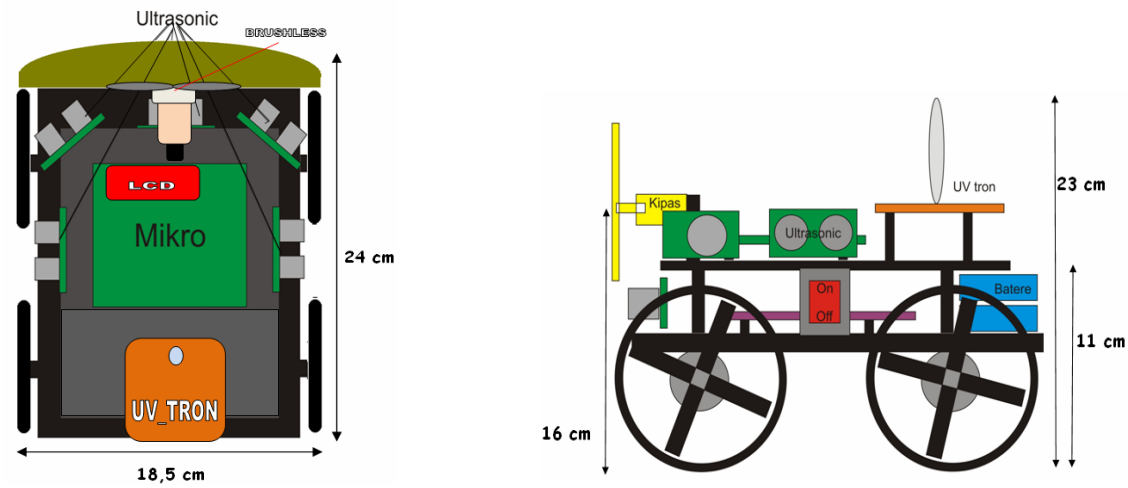
Robot yang dibuat berukuran 24cm x 18.5cm x 23cm. Ukuran ini mengacu pada peraturan KRCI 2012 yaitu ukuran robot tidak boleh melebihi dimensi 31 cm x 31 cm x 27 cm.

Robot dibuat dalam dimensi yang cukup kecil. Dimensi robot yang kecil akan memudahkan robot untuk bergerak di dalam *maze* dan menghindari

rintangan yang ada. Robot menggunakan ban karet buatan *Pololu*. Hal ini dimaksudkan agar robot tidak mudah selip saat bergerak di dalam *maze* dalam kecepatan yang cukup tinggi.

III.5 Perancangan dan Realisasi Robot Beroda Pemadam Api

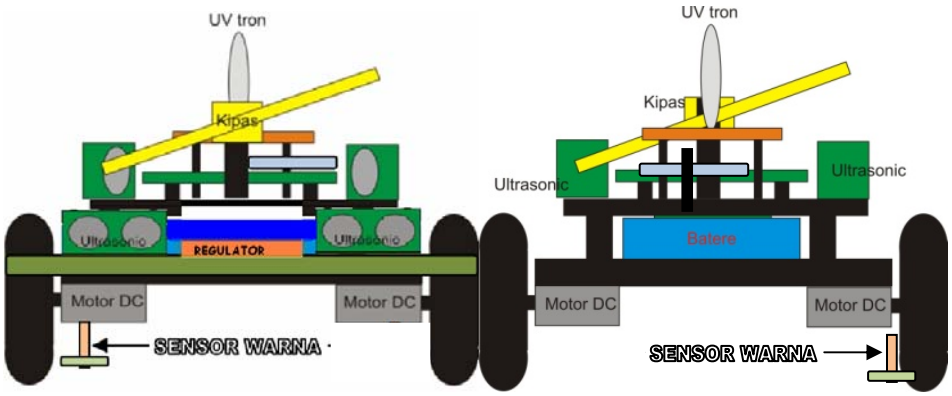
Pada penelitian ini, robot beroda pemadam api dibuat dengan ukuran yang kecil. Robot ini menyerupai sebuah mobil dengan 4 buah roda berdiameter 9 cm dan menggunakan 4 buah motor DC 12 Volt dengan kecepatan putar sekitar 350 rpm sebagai *actuator*. Robot memiliki laju sekitar 70 cm/s. Rangka dasar robot terbuat dari bahan aluminium dan aklirik. Bahan aluminium digunakan untuk menahan beban utama, yaitu 4 buah motor DC beserta rodanya. Bahan aklirik digunakan untuk menyimpan pengontrol mikro, sensor-sensor, dan peralatan ektronik lainnya. Bentuk dan ukuran robot ditunjukkan pada Gambar 3.14.



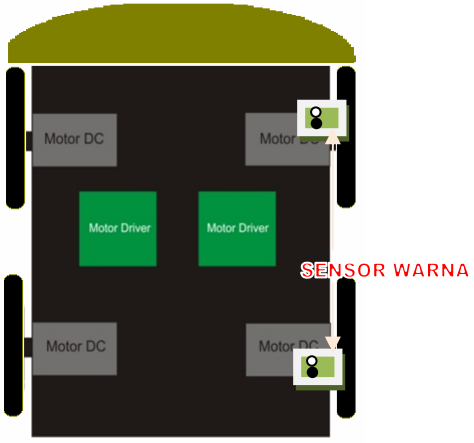
Tampak atas Tampak samping

Gambar 3.14 Dimensi robot beroda pemadam api dan penempatan sensor

Robot ini dilengkapi dengan beberapa jenis sensor. Sensor-sensor yang digunakan adalah sensor jarak ultrasonik, sensor api lilin, dan sensor warna. Posisi penempatan sensor-sensor tersebut diperlihatkan pada Gambar 3.14 dan Gambar 3.15.



Tampak depan Tampak belakang



Tampak bawah

Gambar 3.15 Penempatan sensor pada robot

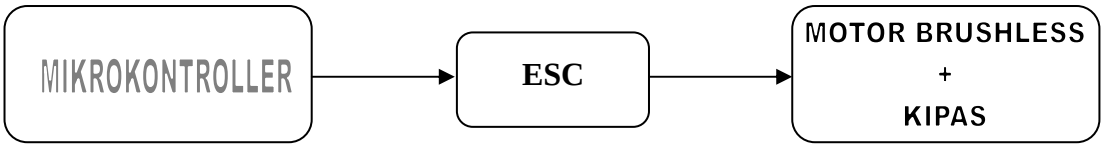
Sensor jarak *ultrasonic* yang digunakan berjumlah 6 buah. Sensor jarak *ultrasonic* digunakan sebagai alat navigasi robot. Pengontrol mikro akan memberikan perintah kepada motor *driver* tergantung pada masukan-masukan dari sensor jarak. Sensor UVTron yang digunakan adalah 1 buah. Sensor warna yang digunakan berjumlah 3 buah. Dua sensor warna terletak di bagian depan dan satu sensor warna terletak dibagian belakang robot. Sensor warna depan berfungsi untuk mendeteksi garis putih yang ada di setiap pintu ruang dan mendeteksi lingkaran atau juring lilin. Selain itu, sensor warna depan akan bekerja bersama dengan sensor warna belakang untuk mendeteksi karpet berwarna abu pada saat robot akan mencari jalan balik ke *home*.

III.6 Sistem Gerak

Robot menerapkan sistem gerak *differential drive*. Sistem ini memungkinkan robot berputar di tempat dengan cara memutar motor dengan arah berlawanan dengan demikian robot tidak memerlukan daerah yang luas untuk bermanuver dan waktu yang lama.

III.7 Pemutar Kipas Pada Robot

Pada penelitian ini digunakan kipas untuk memadamkan api lilin. Untuk memutar kipas diperlukan rangkaian pemutar kipas yang terdiri mikrokontroler, motor *brushless* dan ESC (*Electronic Speed* Kontrol). ESC digunakan sebagai *driver* motor *brushless*. *Output* pada mikrokontroler digunakan untuk menghasilkan *output* sinyal PWM untuk *input* ESC. Dalam hal ini port yang digunakan adalah Port D.5. Gambar 3.16 memperlihatkan diagram blok pemutar kipas.

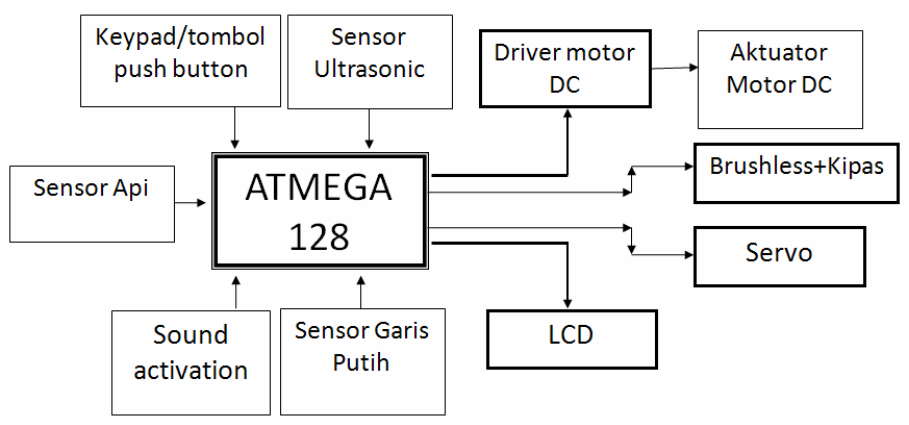


Gambar 3.16 Diagram blok pemutar kipas

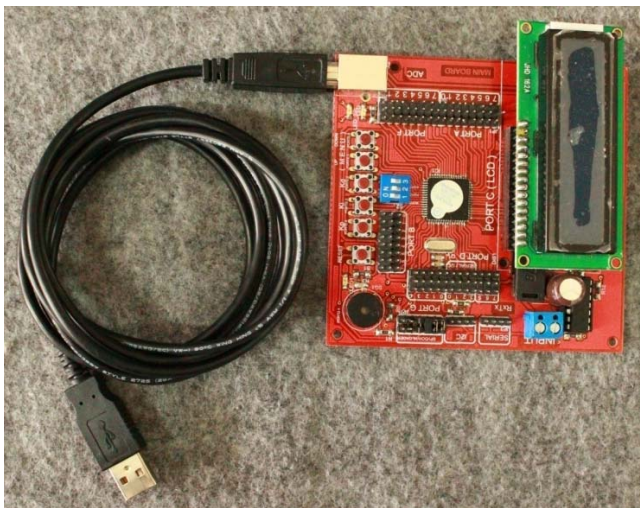
III.8 Skematik Pengontrol Mikro ATMega128A

Pada bagian ini akan dibahas mengenai skematik pengontrol mikro ATmega 128A. Pengontrol mikro ini digunakan untuk mengolah informasi dari sensor-sensor yang akan digunakan untuk mengatur robot agar dapat menjalankan tugasnya dengan baik. Diagram blok pengontrol mikro ATMega128A dapat dilihat pada Gambar 3.17. Sedangkan gambar 3.18 adalah penampang *board* ATmega 128A. Pada rangkaian pengontrol ini digunakan sumber *clock* kristal dengan frekuensi 11.0592 MHz menggunakan kapasitor

dengan nilai 22 pF pada kaki-kaki kristal (C1 dan C2). Rangkaian pengontrol dicatu dengan tegangan yang telah distabilkan dengan regulator tegangan 7805.



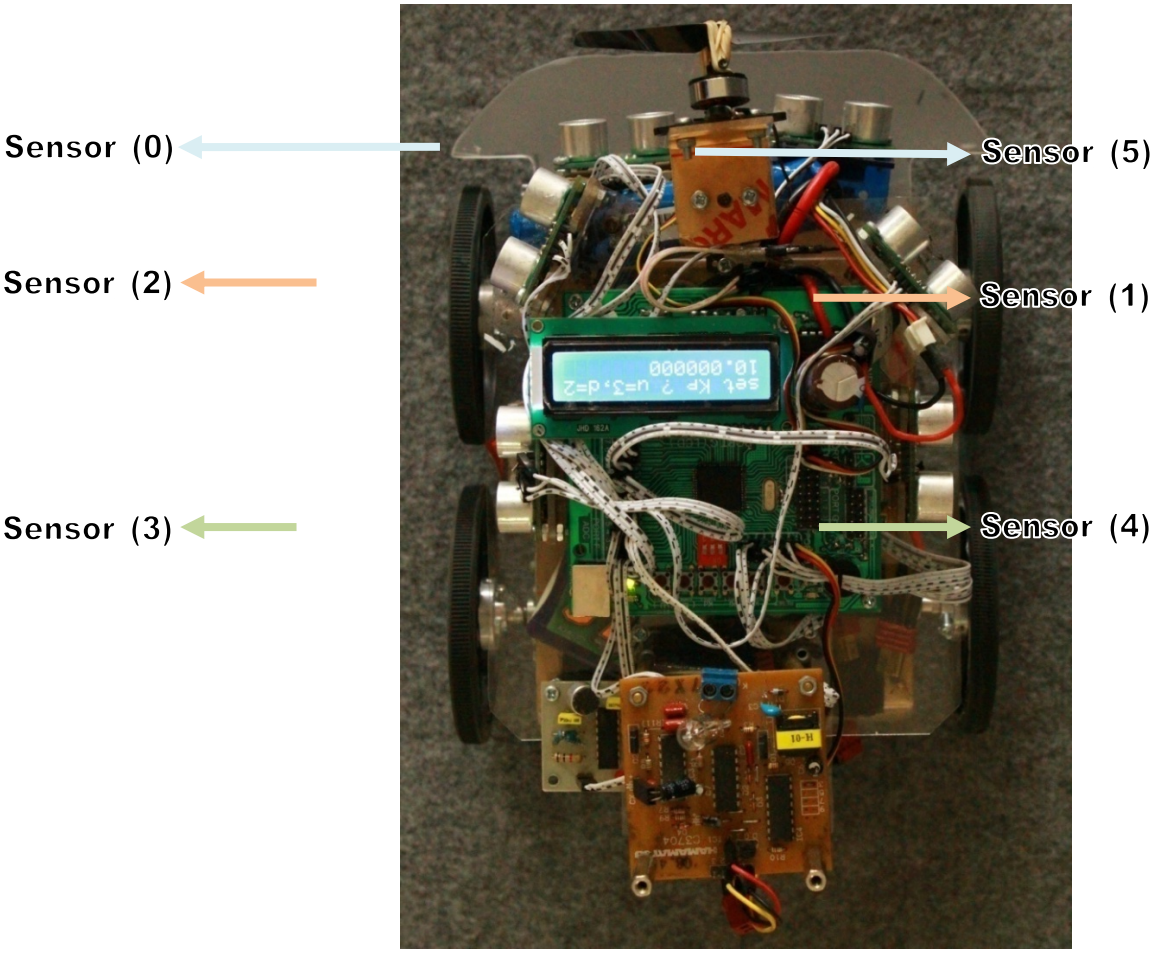
Gambar 3.17 Diagram blok pengontrol mikro ATMega128A



Gambar 3.18 *Board* ATMega128A

Untuk navigasi robot ini, kaki-kaki dari pengontrol mikro ATmega 128A yang digunakan adalah port A.0 sampai A.7 untuk sensor SRF05. Port B.7 digunakan untuk sensor UVTron. Port F.0 dan F.1 digunakan untuk sensor warna. Port F.5, F.6 dan F.7 digunakan untuk *sound activation*. Port D.6 digunakan untuk pengontrol pergerakan servo. Port C digunakan untuk LCD *display*. Port B.1 sampai B.6 digunakan untuk motor *driver* VNH3SP30 MD01B. Port E.0 sampai E.4 digunakan untuk tombol *push button*. Port E5 sampai E.7 digunakan untuk *dip switch*. Port D.4 digunakan untuk *buzzer*. Gambar 3.19

menunjukkan letak posisi sensor SRF05 dan Tabel 3.2 menunjukkan daftar I/O yang digunakan untuk sensor SRF05.



Gambar 3.19 Peletakan sensor-sensor SRF05

III.9 Metoda Tuning PID

Berikut adalah langkah-langkah yang dilakukan dalam *tuning* PID:

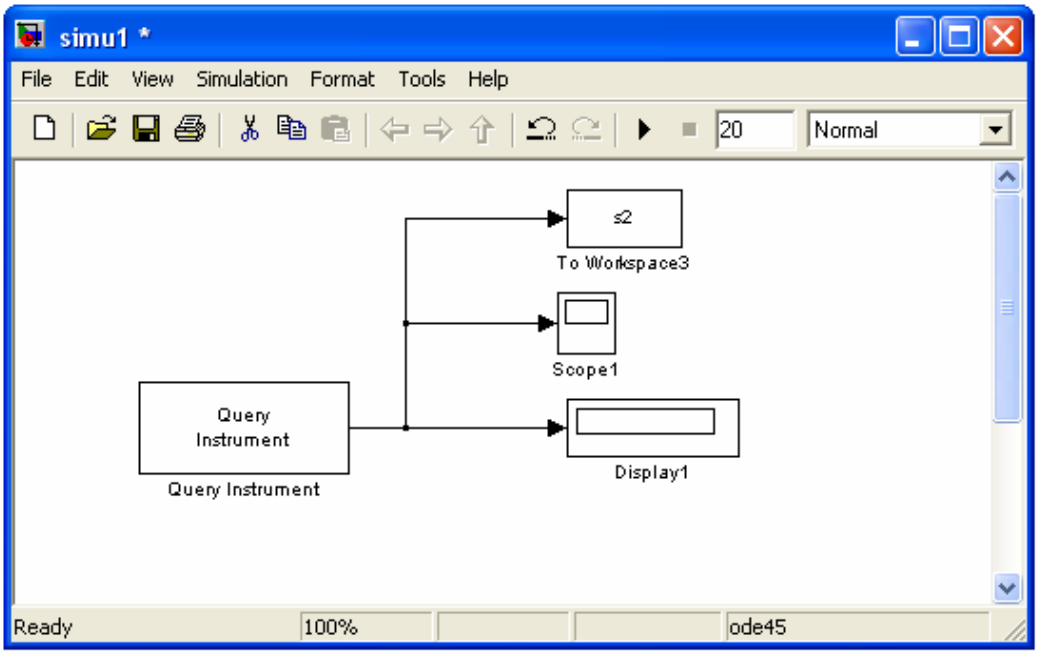
1. Membuat robot berosilasi dengan cara menentukan mengubah parameter gain *proportional* (Kp), mengabaikan parameter *derivative* (Kd) dan *integral* (Ki) terlebih dahulu, dimana gain *proportional* (Kp) menguatkan *output* agar respon mendekati *input (setpoint)* dan memelihara sistem untuk selalu dalam keadaan stabil .

2. Setelah diperoleh nilai parameter Kp yang sesuai, langkah berikutnya adalah menentukan nilai parameter Ki dan mengabaikan nilai Kd. Nilai Ki dituning agar didapatkan keluaran keadaan tunak lebih mendekati *set point*. Dengan penambahan *integral* maka sistem akan tetap dipaksa supaya tetap pada keadaan *steady* dan juga akan mengurangi *error*. Dari nilai Ti yang diperoleh maka nilai Pu dapat dicari berdasarkan hubungan antara Ti dan Pu pada Tabel 2.1 (Ziegler Nichols II) yaitu Ti=0,5 Pu.
3. Langkah selanjutnya adalah menentukan nilai parameter Kd. Melalui nilai Pu yang diperoleh dari langkah no 2, maka nilai parameter Kd (berbanding lurus dengan dengan nilai Td) dapat diperoleh. Penambahan fungsi *derivative* dimaksudkan agar sistem lebih peka terhadap *error* dan mengurangi osilasi, karena dengan penambahan nilai parameter *derivative* berarti menambahkan nilai *zero* sehingga parameter ini mempercepat respon [13].

III.10 *Simulink* Matlab

Simulink pada Matlab adalah salah satu *fitur* dari Matlab untuk mensimulasi suatu desain atau model yang bersifat dinamis ataupun tertanam, simulasi ditujukan untuk mengukur kinerja dari suatu desain atau model system yang telah dirancang yang sesuai hasil yang diinginkan, salah satu kelebihan dari Matlab Simulink ini adalah dapat berkomunikasi serial dengan perangkat kontroler seperti ATmega128, berikut langkah-langkahnya:

1. Pada Matlab, Simulink dibuka dengan mengetik perintah Simulink pada Command Windows Matlab atau klik simbol Simulink pada menu bar. Setelah layar baru terbuka, buka file baru dengan memilih file-new.
2. Pada Simulink Library Browser untuk mencari toolbox instrumen dilakukan dengan mengetik Instrument, klik Icon Search dan Drag Query Instrument ke berkas Model.
3. Kemudian dilakukan dengan cara yang sama untuk mencari Library “Display”, “To Workspace” dan “Scope”, lalu hubungkan kedua item tersebut dengan menarik garis panah seperti pada Gambar 3.20.



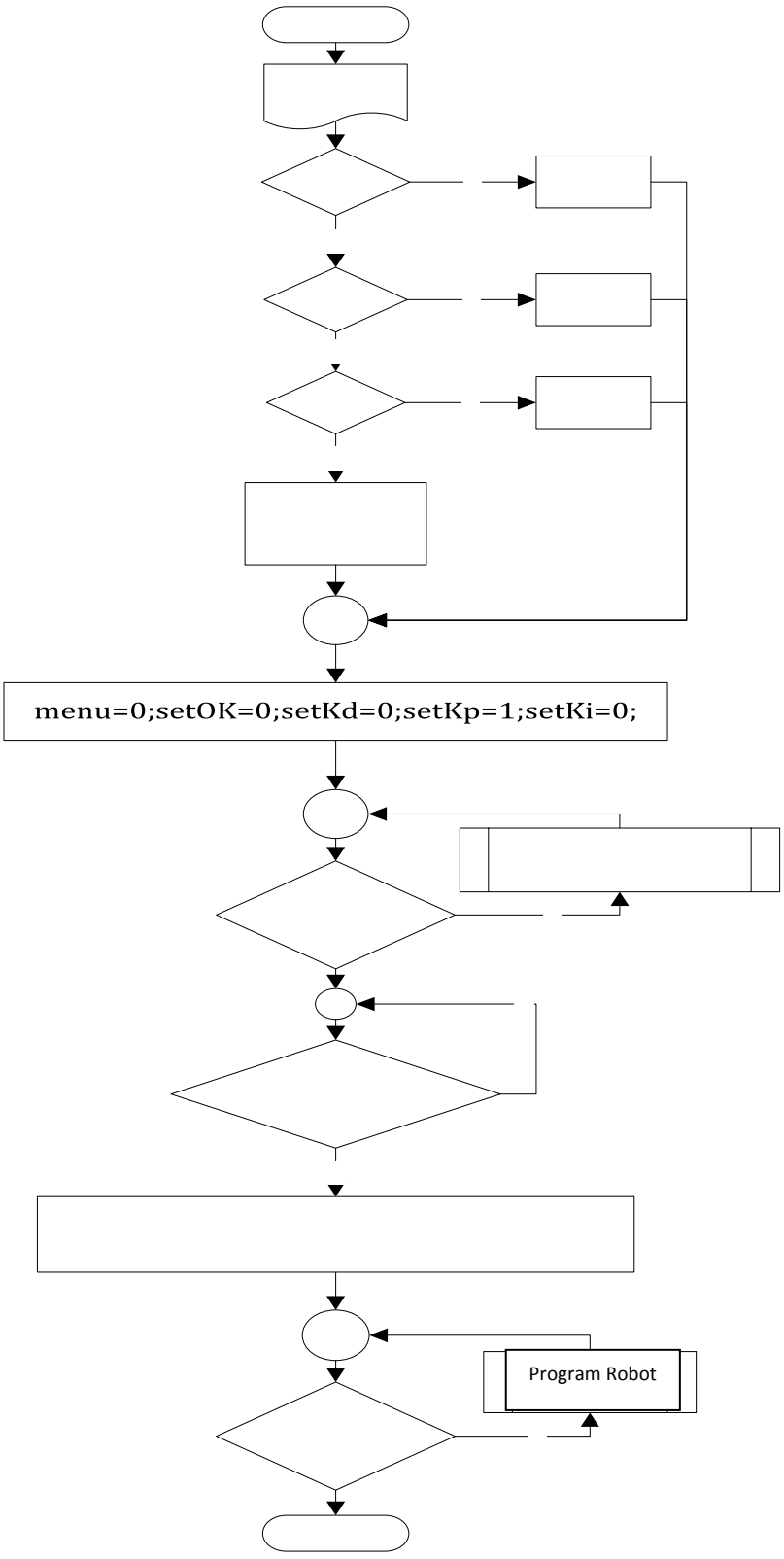
Gambar 3.20 Hubungkan Query dengan Scope, To Workspace dan Display

4. Pengaturan Buffer 16000, Sample Time 0.05 dan Baudrate 9600, dengan cara men-double klik Query Instrument. Baudrate yang digunakan sama dengan Baudrate pada mikrokontroler.
5. Kemudian dilakukan setting pada ATmega128, dengan mengatur Chip sesuai dengan tipe ATmega128 yang dipakai, kemudian Clock yang digunakan adalah 11.059200, lalu meng-klik Tab USART. Beri tanda cek pada Transmintter, kemudian dipilih Baudrate dengan nilai 9600.

III.11 Algoritma Pemrograman pada Robot

Algoritma yang digunakan dalam pengontrolan robot ini berupa diagram alir. Diagram alir dibuat berdasarkan proses pengontrolan robot menggunakan pengontrol mikro ATmega128A.

III.11.1 *Flowchart* Utama pada Robot Pemadam Beroda



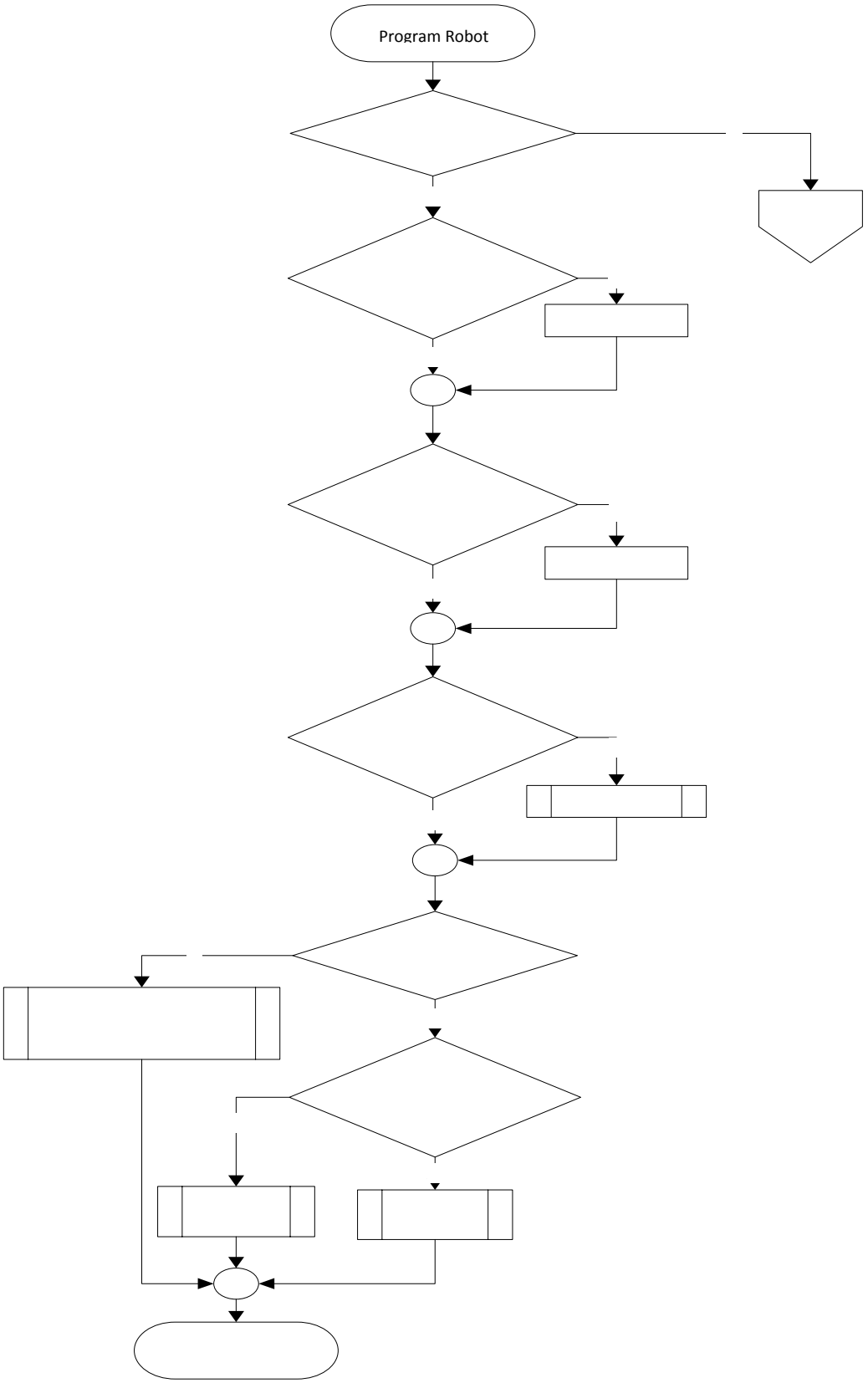
Gambar 3.21 *Flowchart* robot menggunakan sistem kontrol PID

Gambar 3.21 adalah *flowchart*/diagram alir dari program robot *wall follower* menggunakan sistem kontrol PID. Pada awal program terlebih dahulu dilakukan pengecekan pada nilai *variable* Kp, Ki, dan Kd. Setelah dilakukan pengecekan kemudian lanjut ke program berikutnya untuk memasukan nilai Kp, Ki dan Kd menggunakan *push button*. Setelah nilai Kp, Ki dan Kd yang baru dimasukan, maka dengan menggunakan *sound activation* program utama siap dijalankan.

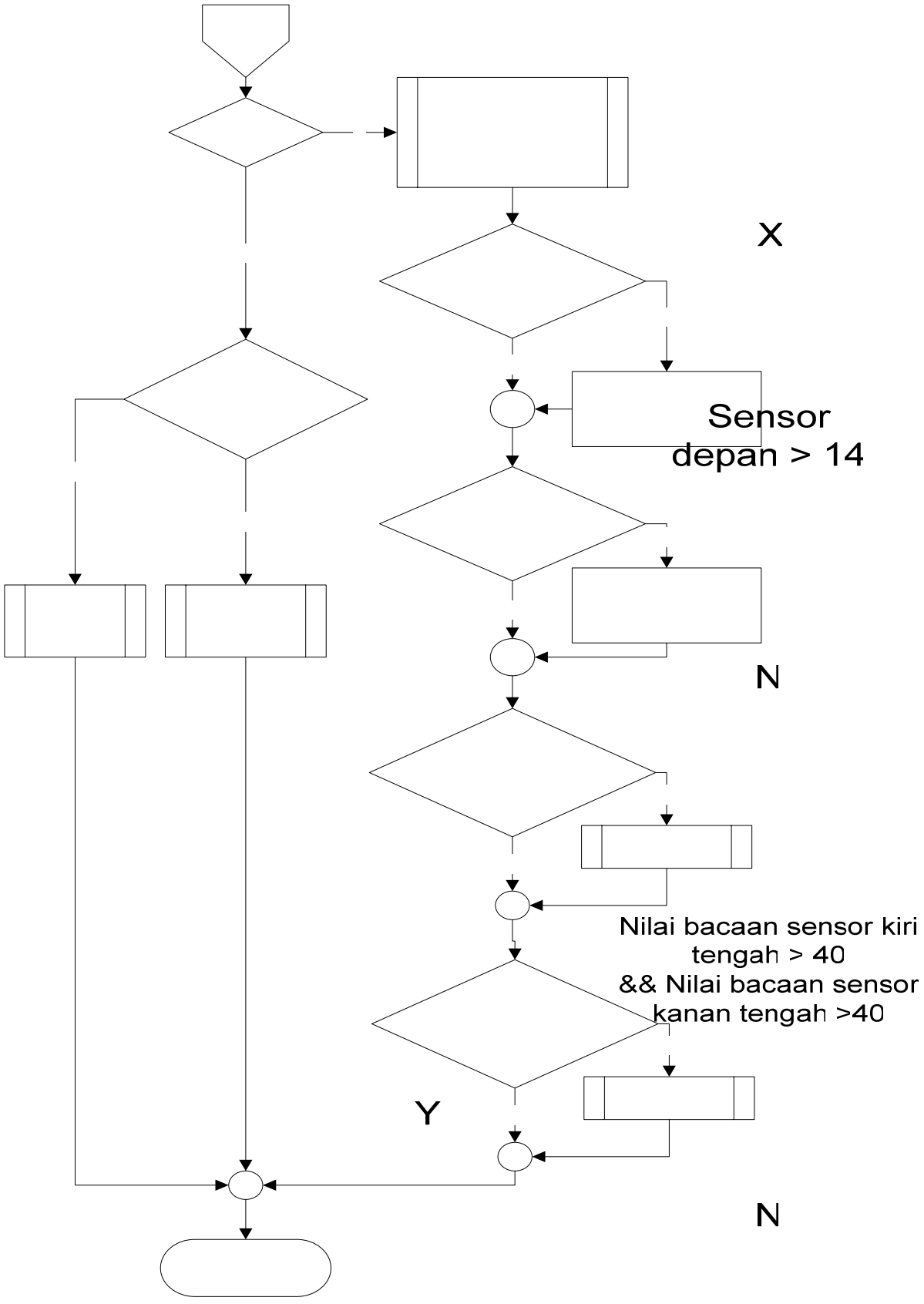
III.11.2 *Flowchart* untuk *Sub* Program Robot

Pada Gambar 3.22a Sub Program ini pertama sensor api mendeteksi keberadaan api jika sensor ini tidak mendeteksi keberadaan api maka program akan lanjut ke konektor X yang akan dibahas pada Gambar 3.22b. Dari diagram alir pada Gambar 3.22a dapat awalnya program mengecek keberadaan titik api dan apabila terdapat keberadaan titik ditemukan maka sensor garis putih akan mengamati kondisi-kondisi seperti keberadaan garis putih pada pintu masuk ruangan dan juring lingkaran yang terdapat titik api sebelum mengaktifkan kipas. Selama robot belum menemukan kondisi-kondisi tersebut maka robot akan tetap menelusuri dinding menggunakan metoda *wall follower* dengan PID.

Gambar 3.22b adalah diagram alir Sub Program pada robot selama robot belum menemukan keberadaan titik api dan juga program untuk balik ke *home*. Pada algoritma ini, selama robot belum menemukan titik api maka robot akan terus menelurusi dinding menggunakan metoda *wall follower* kiri dengan PID dan *up counter* akan tetapi apabila robot telah berhasil memadamkan api maka akan ada *variable* yang berubah untuk menandakan bahwa api telah padam dan robot harus menelusuri dinding untuk kembali ke *home*. Untuk menemukan *home* diperlukan 2 sensor warna untuk mendeteksi warna putih yang besar yang menandakan bahwa itu *home. Up counter* menghitung jumlah ruangan yang dilalui, jika lebih dari 3 ruang dilalui maka robot akan menjalankan instruksi Sub Program *wall follower* kiri.

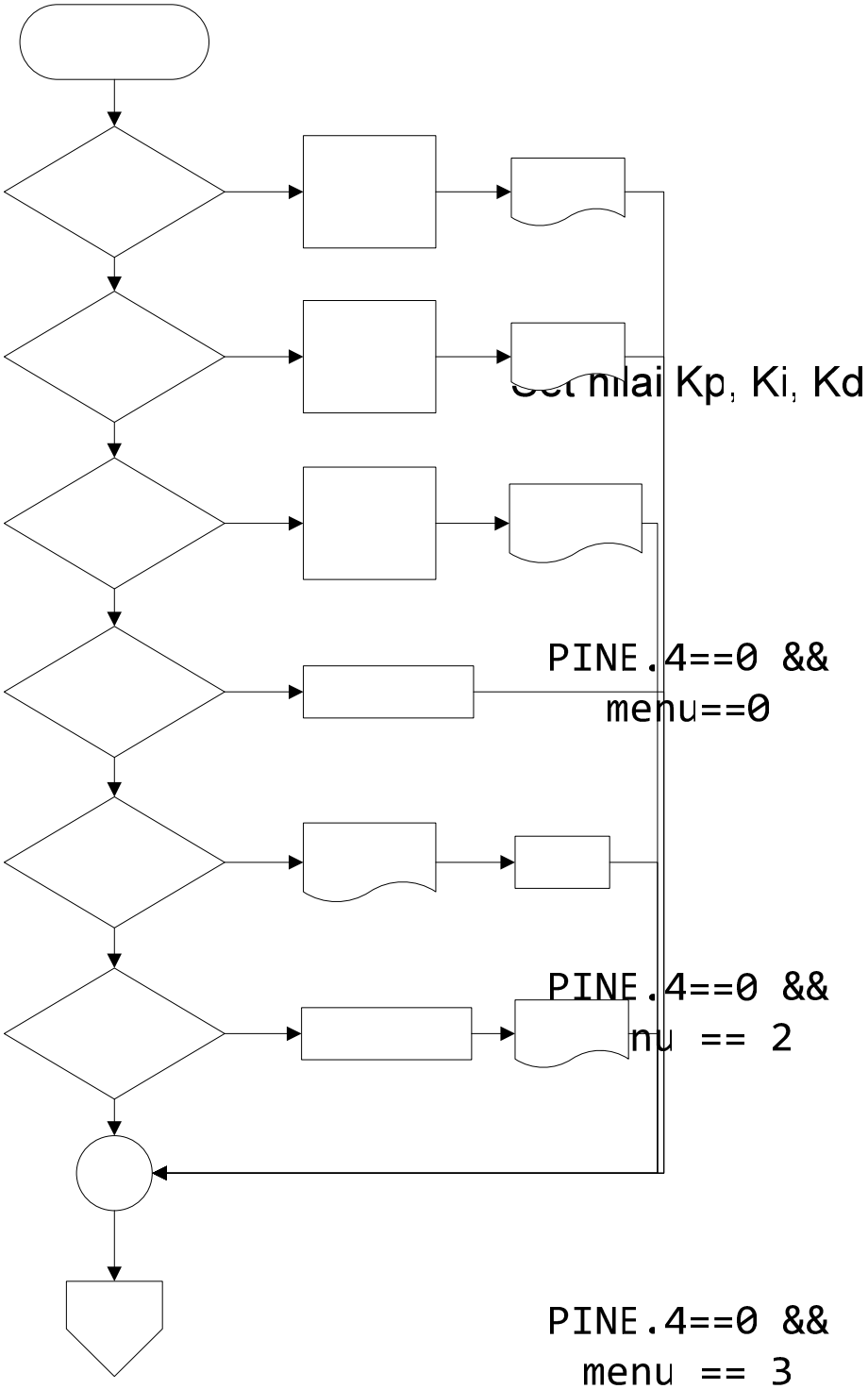


Gambar 3.22a *Flowchart* program robot bagian pertama

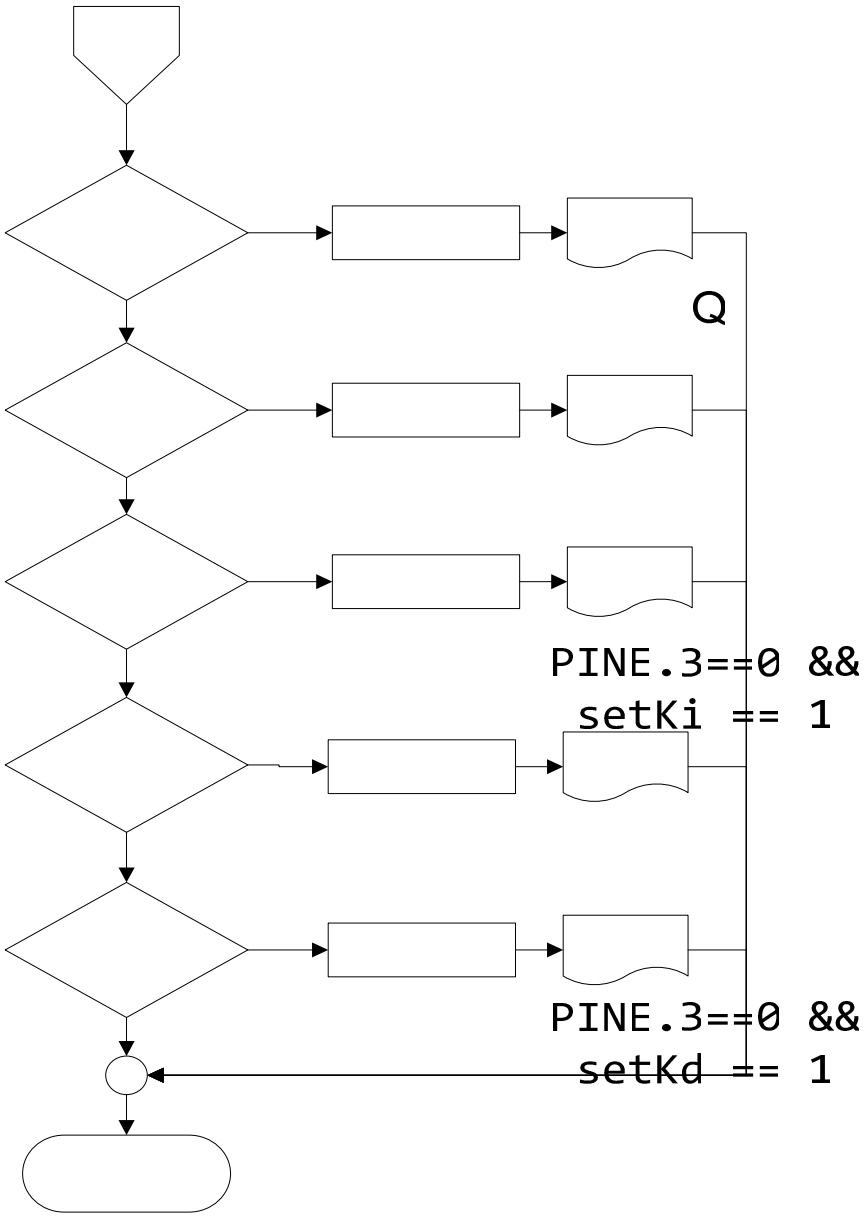


Gambar 3.22b *Flowchart* program robot bagian kedua

III.11.3 *Flowchart* untuk *Sub Program Set* Nilai *Kp, Ki, Kd*



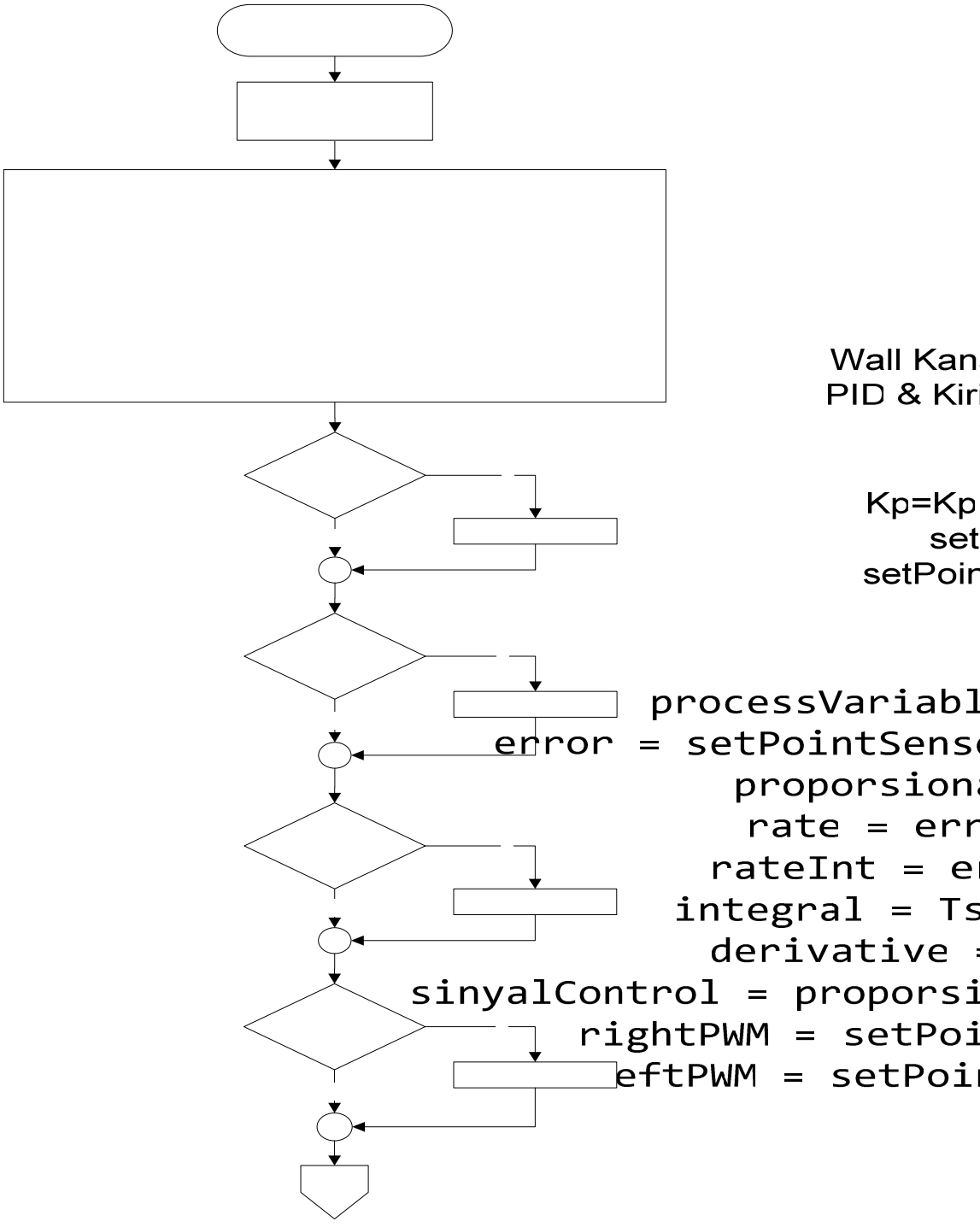
Gambar 3.23a *Flowchart* set nilai Kp, Ki, Kd bagian pertama



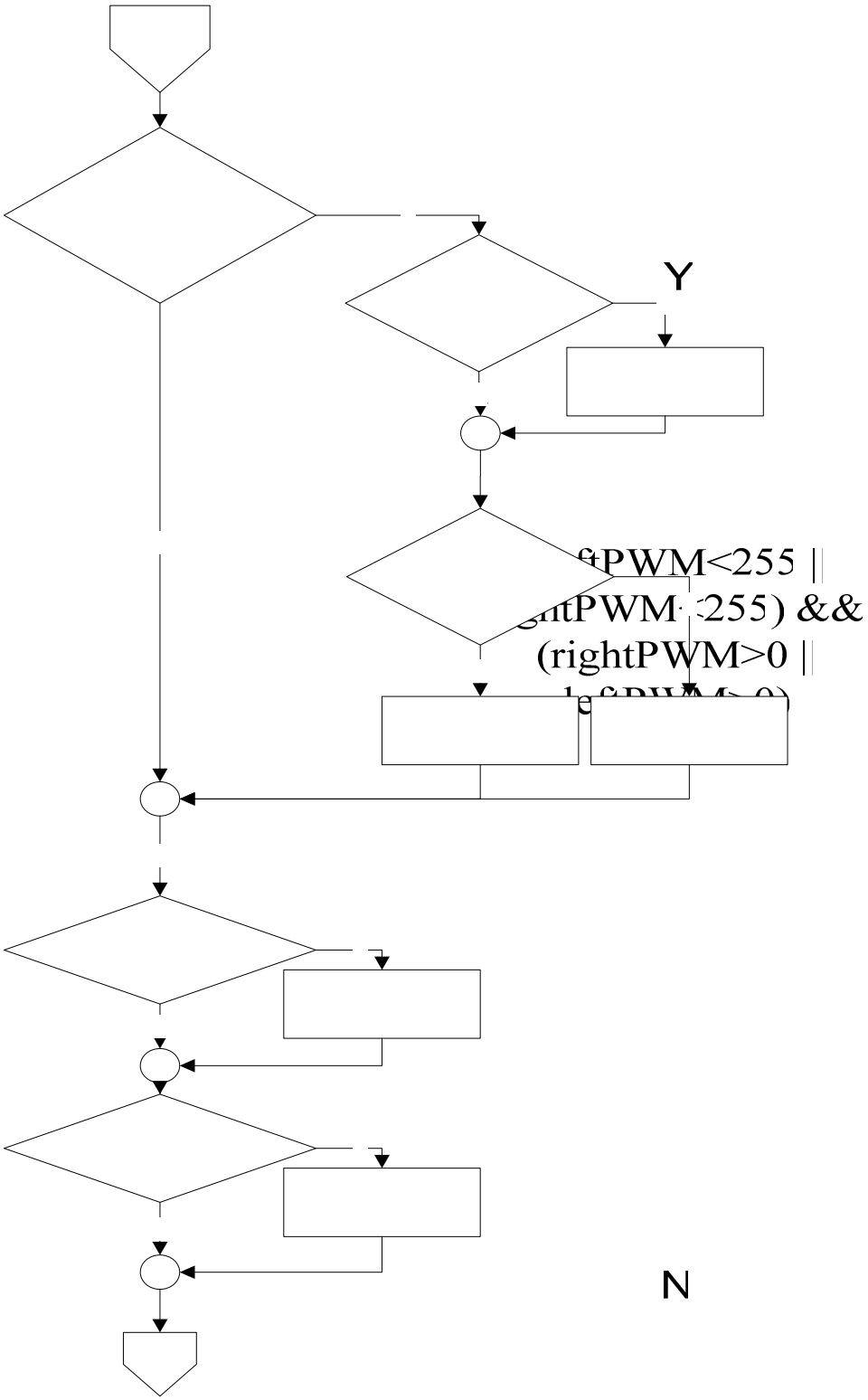
Gambar 3.23b *Flowchart* set nilai Kp, Ki, Kd bagian kedua

Diagram alir untuk Sub Program Set Nilai Kp, Ki dan Kd dapat dilihat pada Gambar 3.23a dan Gambar 3.23b. Gambar 3.23b adalah diagram alir lanjutan dari Gambar 3.23a..Gambar 3.23a adalah program untuk memasukan data secara manual dimana Pin E.4 digunakan untuk *menu* untuk pemilihan nilai Kp, Ki dan Kd yang akan digunakan. Pin E.3 digunakan untuk menambah nilai yang akan digunakan, sedangkan Pin E.2 digunakan untuk mengurangi nilai yang akan digunakan.

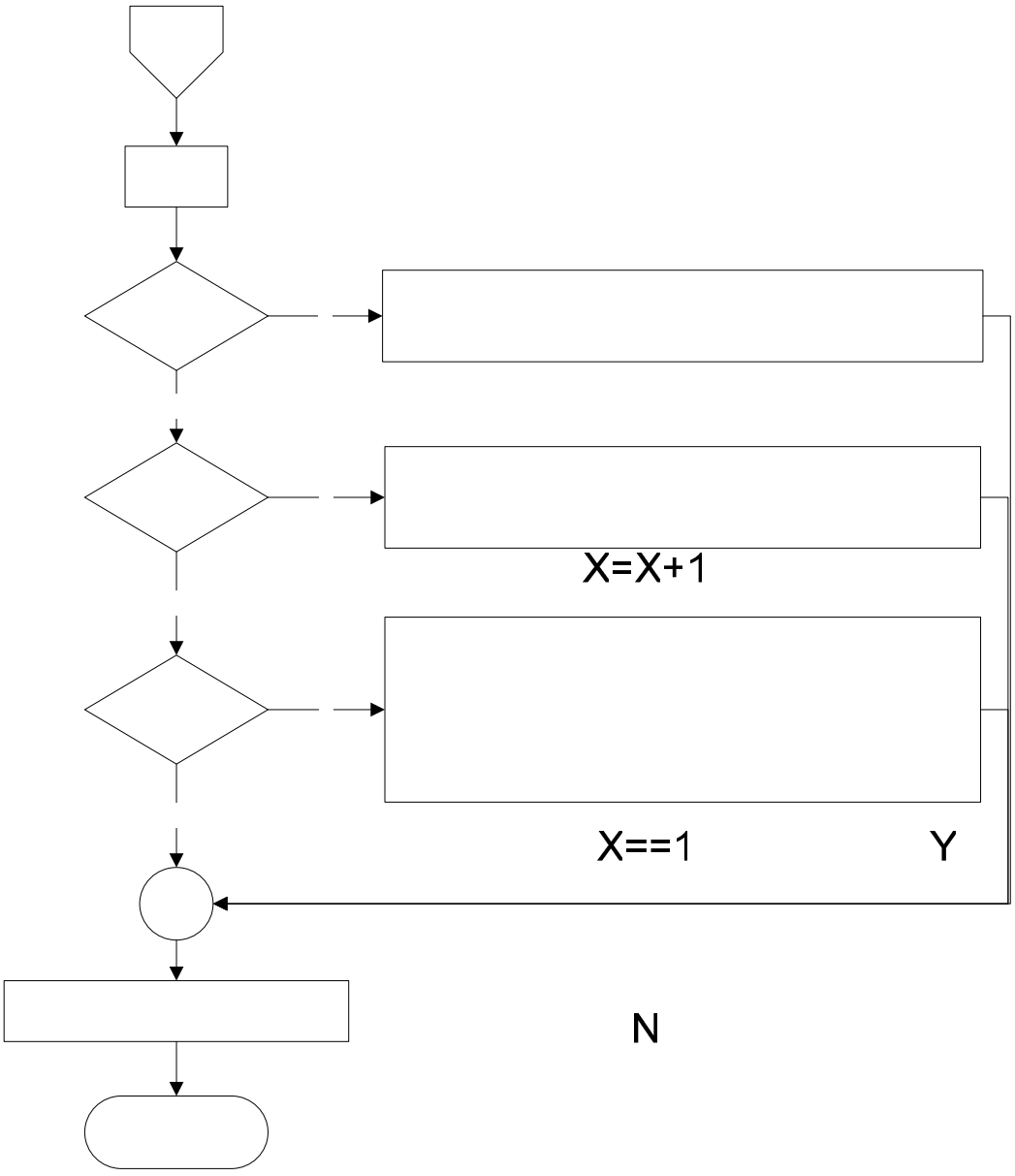
III.11.4 *Flowchart* untuk Sub Program *Wall Follower* Kanan Menggunakan PID dan Kirim Data ke Matlab



Gambar 3.24a *Flowchart wall follower* kanan menggunakan PID bagian pertama



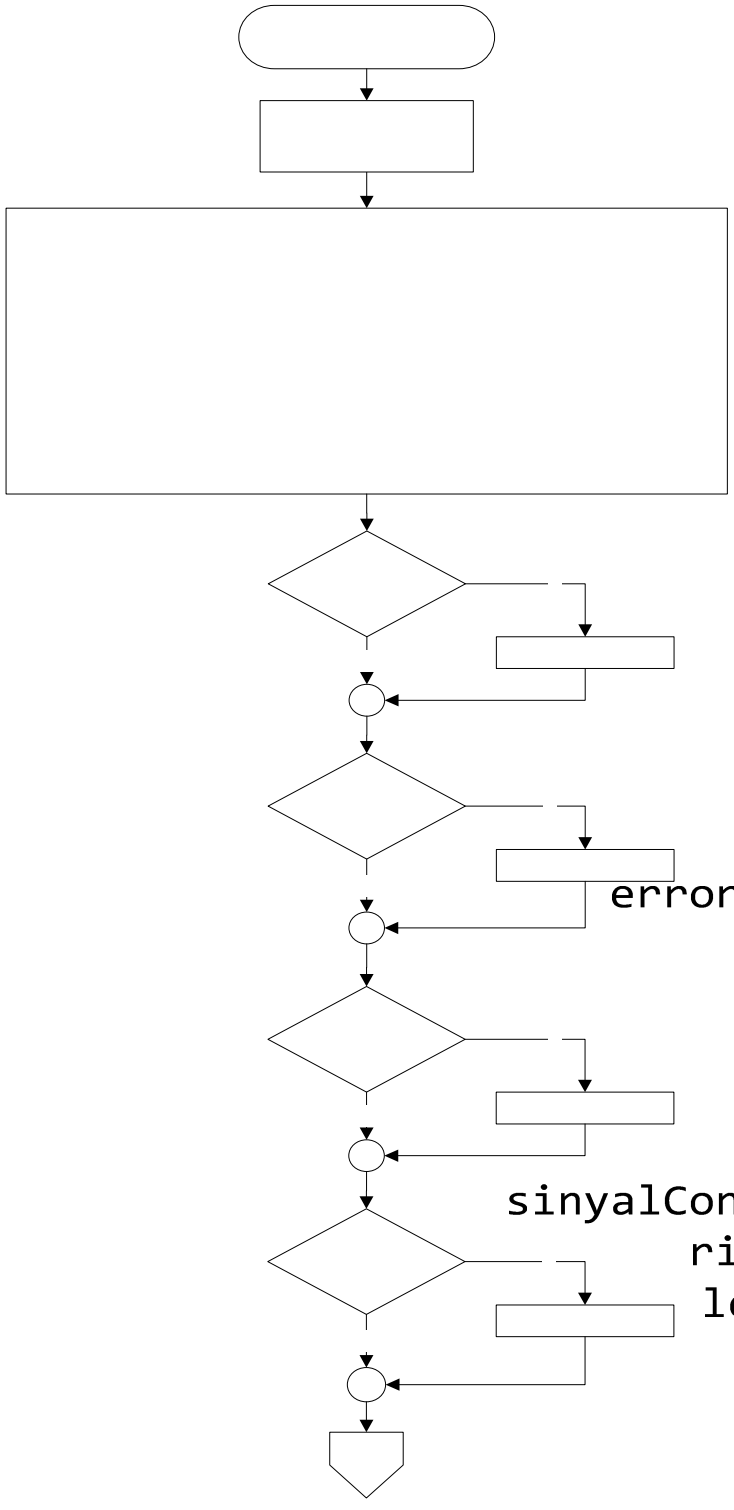
Gambar 3.24b *Flowchart wall follower* kanan menggunakan PID bagian kedua



Gambar 3.24c *Flowchart wall follower* kanan menggunakan PID bagian ketiga

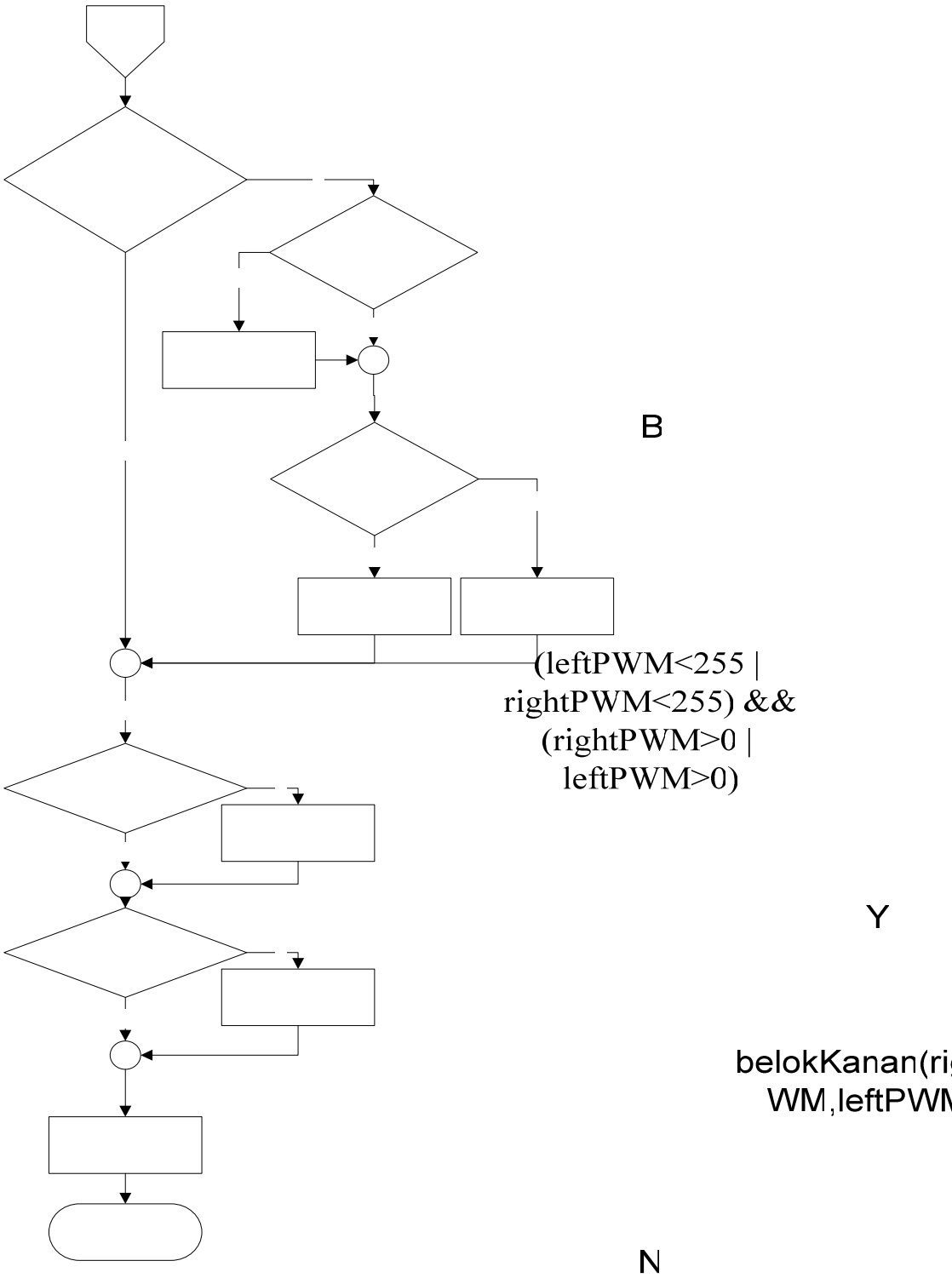
Gambar 3.24a sampai Gambar 3.24c merupakan satu kesatuan gambar yang saling berhubungan. Gambar 3.24a sampai Gambar 3.24c merupakan diagram alir dari program robot untuk berjalan secara *wall follower* kanan dan mengirimkan data pada Matlab.

III.11.5 *Flowchart* untuk Sub Program *Wall Follower* Kiri Menggunakan PID



Gambar 3.25a *Flowchart wall follower* kiri menggunakan PID bagian pertama

Gambar 3.25a dan Gambar 3.25b merupakan satu kesatuan gambar yang saling berhubungan. Gambar 3.25a dan Gambar 3.25b merupakan diagram alir dari program robot untuk berjalan secara *Wall Follower* kiri.



Gambar 3.25b *Flowchart wall follower* kiri mengg unakan PID bagian kedua

BAB 4

DATA PENGAMATAN DAN ANALISIS

Pada bab ini dijelaskan tentang proses pengambilan data pengamatan, pengujian kemampuan robot beroda pemadam api, dan analisisnya.

IV.1 Pengujian Sensor Warna

Sensor warna yang digunakan adalah sensor warna ZX-03. Percobaan yang dilakukan adalah pengukuran terhadap warna lantai maze yaitu hitam, abu-abu, dan putih. Pengukuran dilakukan pada beberapa posisi untuk masing-masing warna sehingga diketahui range nilai yang dibaca sensor. Ketinggian sensor depan dari lantai adalah 0.5 cm dan ketinggian sensor belakang dari lantai adalah 0.6 cm. Tabel percobaan yang dilakukan dapat dilihat pada Tabel 4.1a dan Tabel 4.1b.

Tabel 4.1a Tabel pengukuran sensor warna kiri depan

	SENSOR WARNA KIRI DEPAN		
	Hitam	**Abu**	**Putih**
	131	440	975
	108	423	976
	128	400	972
	152	449	983
	135	419	967
	142	429	981
	140	393	982
	88	408	986
	95	430	980
	123	403	967
Range Sensor Warna:	**0-350**	**351-850**	**851-1023**

Tabel 4.1b Tabel pengukuran sensor warna kiri belakang

	SENSOR WARNA BELAKANG		
	Hitam	**Abu**	**Putih**
	425	819	1023
	512	754	995
	376	799	963
	240	646	969
	399	750	936
	467	681	947
	352	672	956
	502	789	942
	373	742	943
	363	818	960
Range Sensor Warna:	**0-600**	**601-850**	**851-1023**

Berdasarkan Tabel 4.1a dan Tabel 4.1b dapat disimpulkan bahwa range nilai sensor warna kiri depan: warna hitam berkisar antara 0-350, warna abu berkisar 351-850, dan untuk warna putih berkisar 851-1023. Sedangkan range nilai sensor warna kiri belakang: warna hitam berkisar antara 0-600, warna abu berkisar 601-850, dan untuk warna putih berkisar 851-1023.

Dari hasil percobaan dapat diketahui bahwa hasil pembacaan sensor dipengaruhi oleh ketinggian sensor dari objek yang akan dideteksi warnanya. Sensor warna harus berada dalam jarak dekat dengan objek yang akan dideteksi agar sensor dapat membedakan warna putih, hitam, dan abu-abu dengan baik.

IV.2 Pengujian Sensor *Ultrasonic* / Sensor Jarak SRF05

Sensor jarak yang digunakan adalah sensor *ultrasonic* SRF05. Percobaan yang dilakukan adalah pengukuran nilai bacaan sensor terhadap sudut sensor *ultrasonic* terhadap bidang/dinding. Tabel percobaan yang dilakukan dapat dilihat pada Tabel 4.2.

Tabel 4.2 Tabel pengukuran sensor *ultrasonic* SRF05

Jarak (cm)	**Sudut Pandang Sensor Terhadap Dinding (Derajat)**	**Nilai Bacaan Sensor Ultrasonic**
8	130	14
8	125	10
8	120	8
8	110	8
8	90	8
8	60	8
8	50	8
8	45	10
8	40	14

Pada Tabel 4.2, hasil bacaan sensor *ultrasonic* mulai kurang akurat pada sudut lebih dari 120º dan kurang dari 50º, yaitu untuk kemiringan mencapai +30º tegak lurus terhadap dinding (dalam hal ini tegak lurus pada saat 90º) dan -40º tegak lurus terhadap dinding, hasil pembacaan sensor mulai kurang akurat.

IV.3 *Tuning* PID pada Robot

Tuning PID pada robot dimaksudkan agar robot memiliki *performance* yang maksimal dalam bergerak. Hasil tuning PID akan digunakan agar jarak robot dengan dinding dijaga berkisar 8cm untuk beberapa nilai PWM kecepatan pergerakan robot (25, 50, 75, dan 125). Maksimal nilai PWM 256 mewakili PWM digital 8 bit (PWM tersebut memiliki resolusi 2^8 = 256). Artinya nilai keluaran PWM memiliki 256 variasi (0 – 255) yang mewakili *duty cycle* 0– 100% dengan periode $\frac{1}{f_{clk}}$ 256 (menggunakan cristal external pada mikrokontroler

dengan nilai f_{clk} = 11,0592MHz). Dengan menghitung *duty cycle*, akan didapat tegangan *output* mikrokontroler yang dihasilkan yaitu $Vout = \frac{Ton}{Ton + Toff} 5volt$. 5volt adalah tegangan dari mikrokontroler. Pada perancangan driver ini, sinyal PWM akan diatur secara digital yang dibangkitkan oleh mikrokontroler ATMEGA128.

Pengujian dilakukan dengan mengamati jarak yang diinginkan robot dengan dinding dan kecepatan pergerakan robot setelah diberi pengontrol PID hingga menuju jarak (8 cm) untuk beberapa kecepatan yang diinginkan (*reference* PWM 25, 50, 75, dan 125). Agar tuning PID dipilih cukup baik maka perlu memperhatikan empat sinyal data yaitu: sinyal kontrol dari kontroler, konversi sinyal kontrol dalam bentuk sinyal PWM untuk *actuator* motor DC kiri, proses variable (nilai bacaan sensor *ultrasonic*) dan *error* jarak yang dihasilkan sensor *ultrasonic*.

IV.3.1 *Tuning* PID untuk *setpoint* jarak (8 cm) pada r*eference* PWM 25 dan 50

Pada pengujian ini akan diambil beberapa data untuk *set point* (8 cm) pada *reference* PWM 25 dan 50 beserta analisisnya:

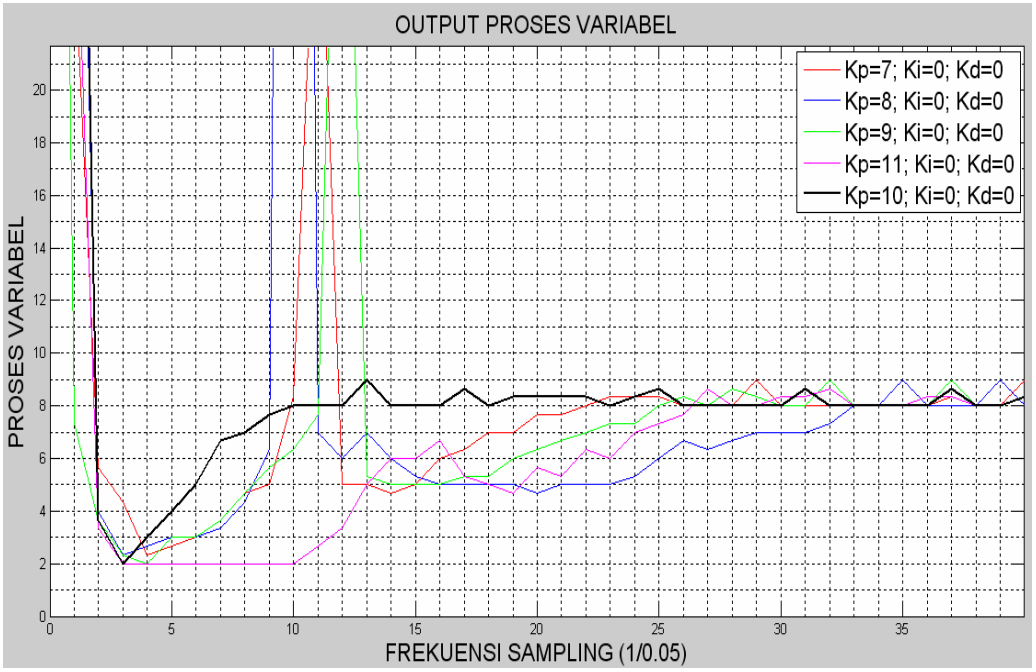
1. Pengujian mencari nilai parameter Kp yang sesuai dengan membuat robot berosilasi dengan cara menentukan mengubah parameter *gain Proportional* (Kp), mengabaikan parameter *Derivative* (Kd) dan *Integral* (Ki) terlebih dahulu. *Gain Proportional* (Kp) menguatkan *output* agar respon mendekati *input (setpoint)* dan memelihara sistem untuk selalu dalam keadaan stabil.
2. Setelah diperoleh nilai parameter Kp yang sesuai pengujian selanjutnya adalah menentukan nilai parameter Ki dan mengabaikan nilai Kd. Nilai Ki dituning agar didapatkan keluaran keadaan tunak lebih mendekati *set point*.
3. Setelah didapat nilai parameter Kp dan Ki pengujian selanjutnya adalah menentukan nilai parameter Kd. Melalui nilai Pu yang diperoleh dari langkah

no 2, maka nilai parameter Kd (berbanding lurus dengan dengan nilai Td) dapat diperoleh.

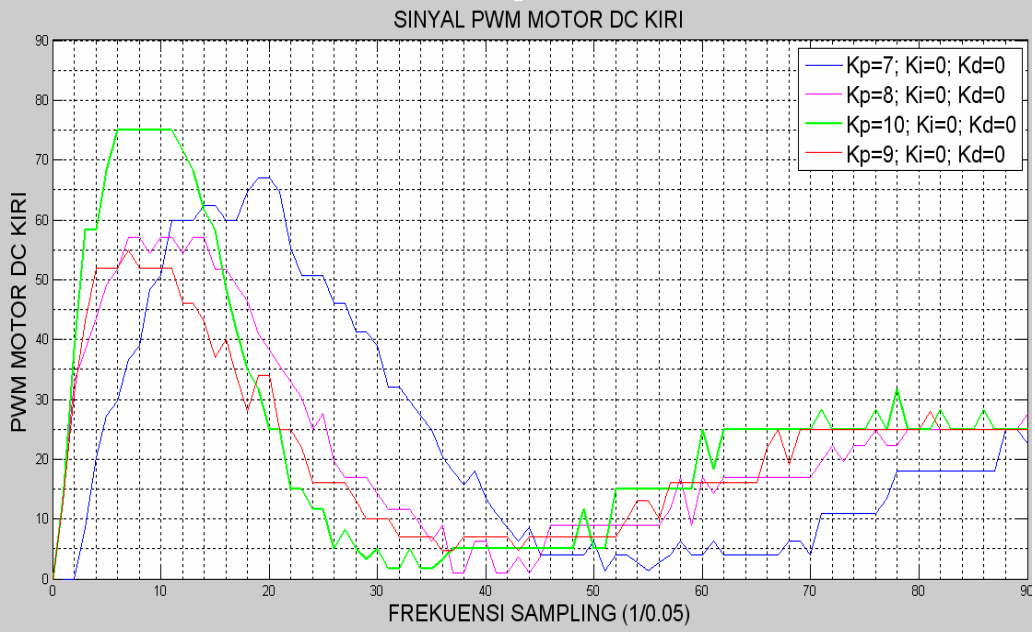
4. Setelah didapat Kp, Ki, Kd pengujian selanjutnya adalah pengujian data-data hasil *tuning* terbaik pada kontroler P, PI, dan PID akan dibandingkan setelah didapatkan nilai parameter yang terbaik. Yang dimaksud terbaik disini adalah yang memiliki *settling time* tercepat dan *error steady state* terkecil.
5. Terakhir adalah pengujian pada arena KRCI.

IV.3.1.1 *Tuning* Nilai Parameter *Proportional* (Kp) untuk *Reference* PWM 25 dengan *set point* jarak (8 cm)

Pada Gambar 4.1 memperlihatkan grafik posisi robot dengan dinding (proses variabel) yang dihasilkan dengan kontroler Proporsional. Grafik dengan nilai Kp=10 memperlihatkan hasil yang terbaik karena memiliki *settling time* yang paling cepat mencapai *set point* jarak (8 cm) dan stabil.



Gambar 4.1 Posisi robot (proses variable) untuk kontroler *Proportional* dengan *reference* PWM 25 dan *set point* jarak (8 cm)



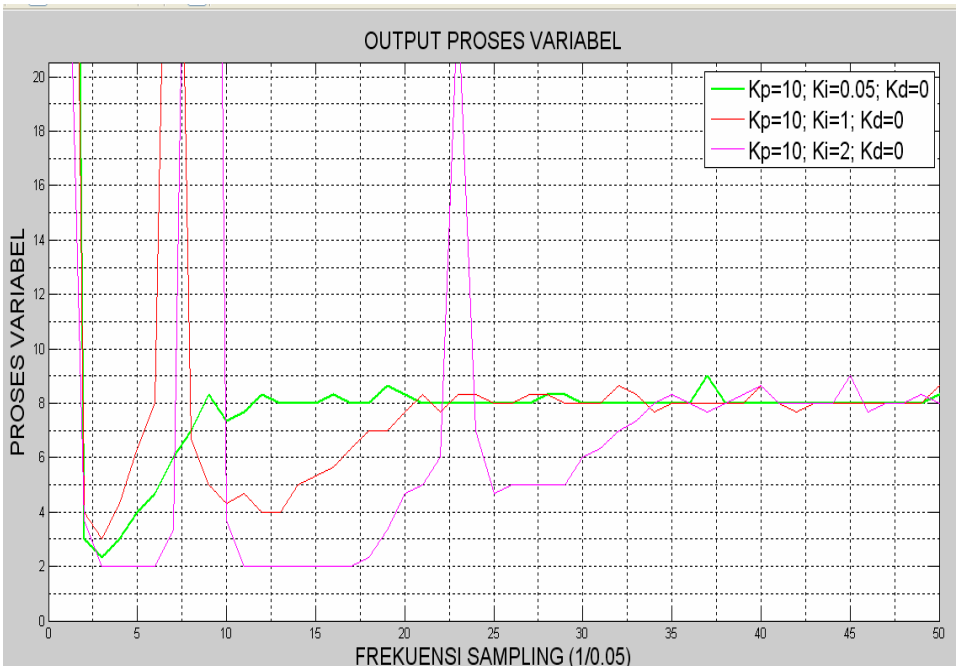
Gambar 4.2 Sinyal PWM motor DC kiri dari kontroler *Proportional* untuk *reference* PWM 25 dan *set point* jarak (8 cm)

Gambar 4.2 memperlihatkan juga bahwa untuk nilai parameter Kp=10 memiliki *settling time* yang cukup cepat mencapai *reference* PWM yang diinginkan dan memiliki osilasi yang cukup stabil juga. Dari gambar tersebut dapat dilihat pemilihan Kp yang semakin besar akan mempercepat *settling time* dalam mencapai *reference* PWM yang diinginkan (25), namun pemilihan parameter Kp yang terlalu besar juga akan membuat sistem justru semakin tidak stabil juga (sistem akan berosilasi terus menerus).

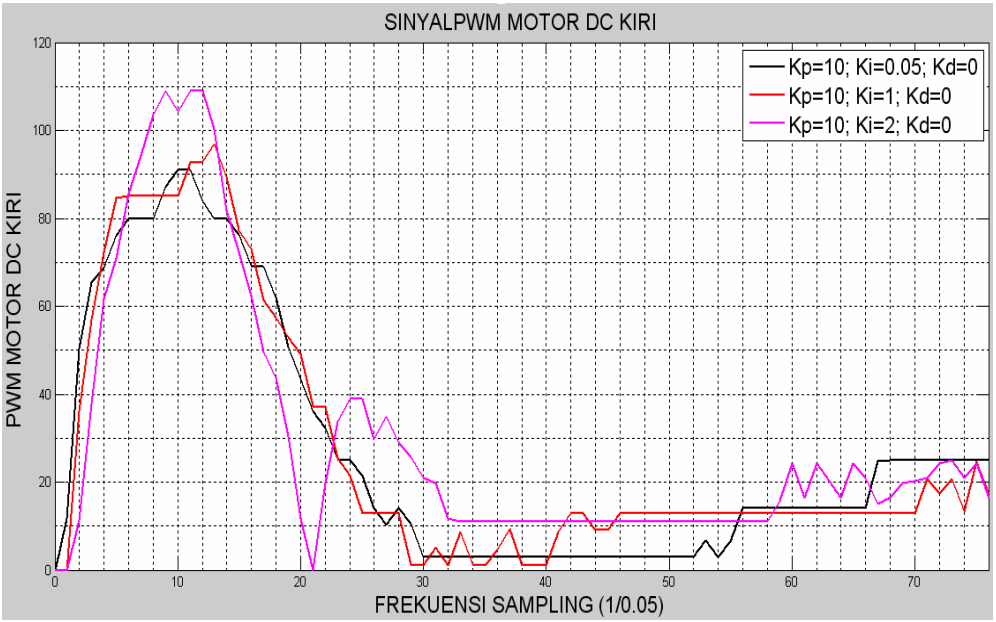
IV.3.1.2 *Tuning* Nilai Parameter *Integral* (Ki) untuk *Reference PWM* 25 dengan *set point* jarak (8 cm)

Setelah didapatkan parameter *gain Proportional* yang baik, berikutnya adalah menentukan parameter *Integral.* Gambar 4.3 memperlihatkan grafik sinyal yang diharapkan yaitu jarak robot dengan dinding yang dihasilkan kontroler *Proportional Integral.* Pemilihan nilai Ki=0.05 memperlihatkan hasil yang baik karena memiliki *settling time* yang paling cepat dan stabil dibanding nilai Ki=1

dan Ki=2. Pada Gambar 4.3, Ki=1 dan Ki=2 terdapat lonjakan. Lonjakan tersebut menandakan bahwa robot mengalami osilasi pada saat bergerak.



Gambar 4.3 Posisi robot **(**proses variable) untuk kontroler *Proportional Integral* dengan *reference* PWM 25 dan *set point* jarak (8 cm)



Gambar 4.4 Sinyal PWM motor DC kiri untuk kontroler *Proportional Integral* dengan *reference* PWM 25 dan *set point* jarak (8 cm)

Pemilihan parameter Ki=0.05 semakin dikuatkan seperti terlihat pada sinyal PWM pada *actuator*. Gambar 4.4 menunjukkan bahwa sinyal PWM yang dihasilkan dapat mencapai *reference* PWM dengan baik tanpa osilasi kembali diakhir. *Rise time* dan *settling time* yang cukup cepat juga mendukung pemilihan parameter Ki=0.05.

IV.3.1.3 *Tuning* Nilai Parameter *Derivative* (Kd) untuk *Reference* PWM 25 dan *Set Point* Jarak (8 cm)

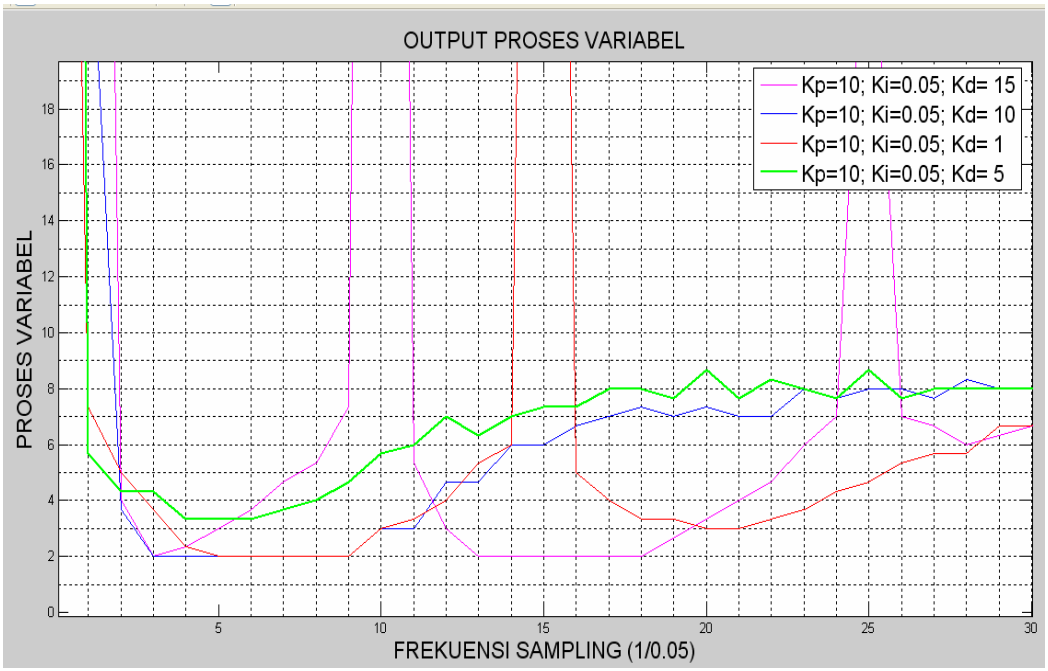
Setelah didapatkan parameter *gain Proportional* dan *Integral* yang baik, berikutnya adalah menentukan parameter *Derivative*. Berbeda dengan *tuning* sebelumnya, pemilihan parameter Kd mengacu pada metoda Ziegler-Nichols II, yaitu setelah mendapatkan nilai parameter Ki maka digunakan persamaan 4.1 sampai persamaan 4.3 sehingga nilai Kd dapat langsung diperoleh tanpa harus *trial and error*.

$$Ki = \frac{1}{Ti} \quad \textbf{(4.1)}$$

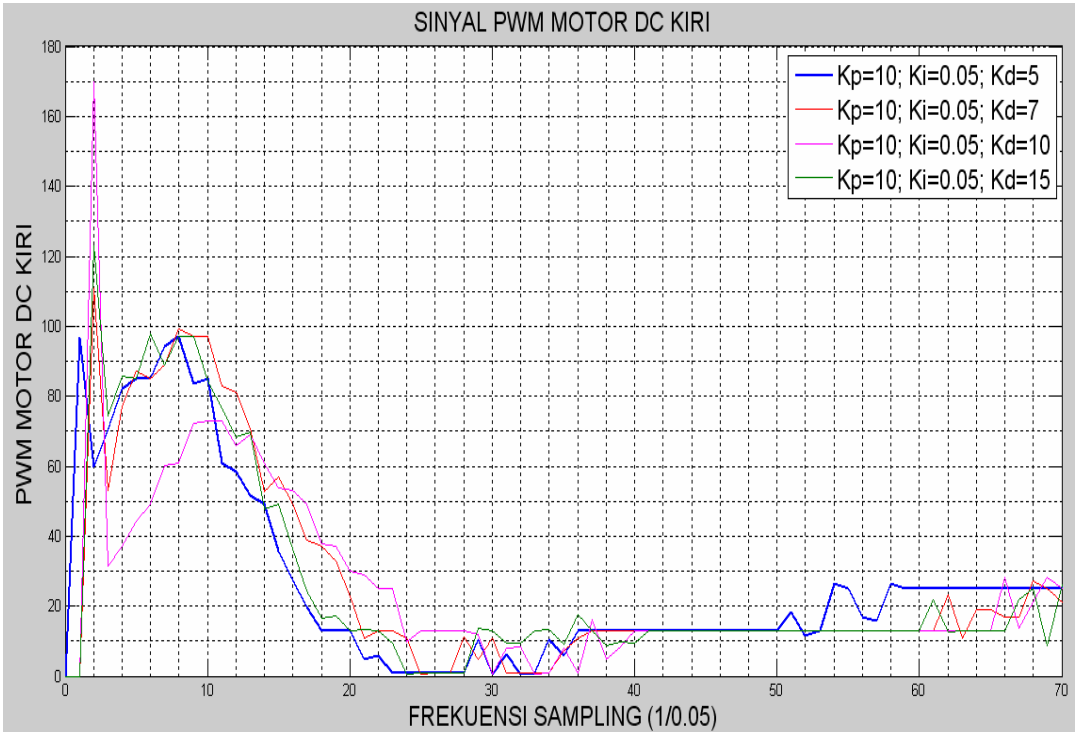
$$Ti = 0.5Pu \quad \textbf{(4.2)}$$

$$Td = Kd = 0.125Pu \quad \textbf{(4.3)}$$

Gambar 4.5 memperlihatkan grafik posisi robot dengan dinding (proses variabel) yang dihasilkan untuk kontroler *Proportional Integral Derivative.* Dengan mengacu pada metoda Ziegler-Nichols II maka kontroler *Proportional Integral Derivative* menghasilkan grafik seperti pada Gambar 4.5 yaitu memiliki nilai parameter *Proportional* (Kp=10), nilai parameter *Integral* (Ki=0.05), dan nilai parameter *Derivative* Kd=5 (menggunakan metoda Ziegler-Nichols II). Pemilihan nilai Kd=5 memperlihatkan hasil *output* yang baik karena memiliki *settling time* yang paling cepat dan stabil dan tidak ada lonjakan. Lonjakan tersebut menandakan bahwa robot mengalami osilasi pada saat bergerak.



Gambar 4.5 Posisi robot (proses variable) untuk kontroler *Proportional Integral Derivative* dengan *reference* PWM 25 dan *set point* jarak (8 cm)



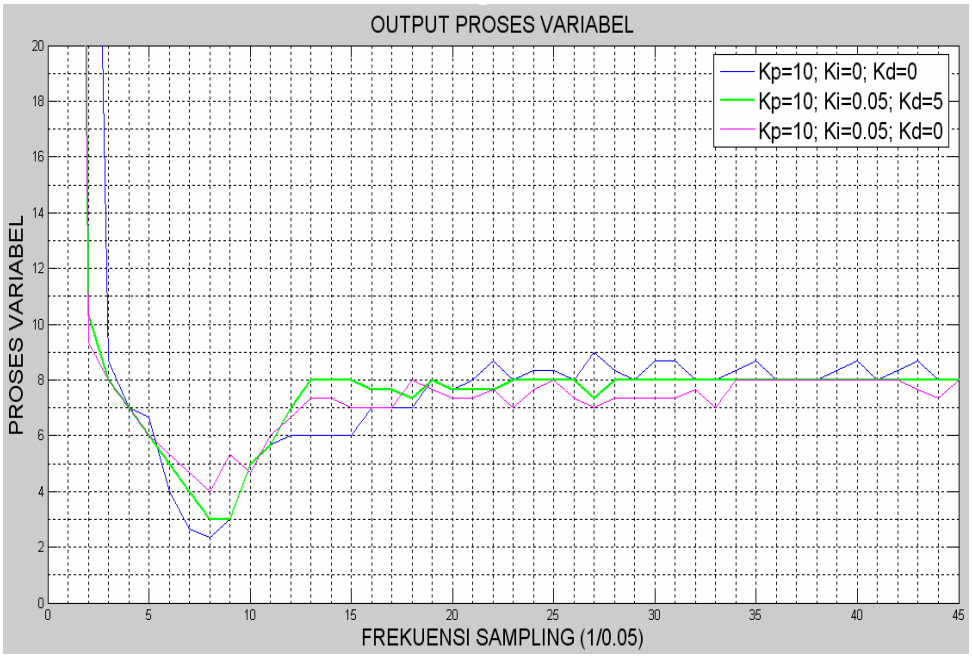
Gambar 4.6 Sinyal PWM motor DC kiri untuk kontroler *Proportional Integral Derivative* dengan *reference* PWM 25 dan *set point* jarak (8 *c*m)

Gambar 4.6 memperlihatkan juga bahwa sinyal PWM pada *actuator* motor DC kiri untuk nilai Kd=5 sangat baik karena memiliki *rise time* dan *settling time* yang cukup cepat.

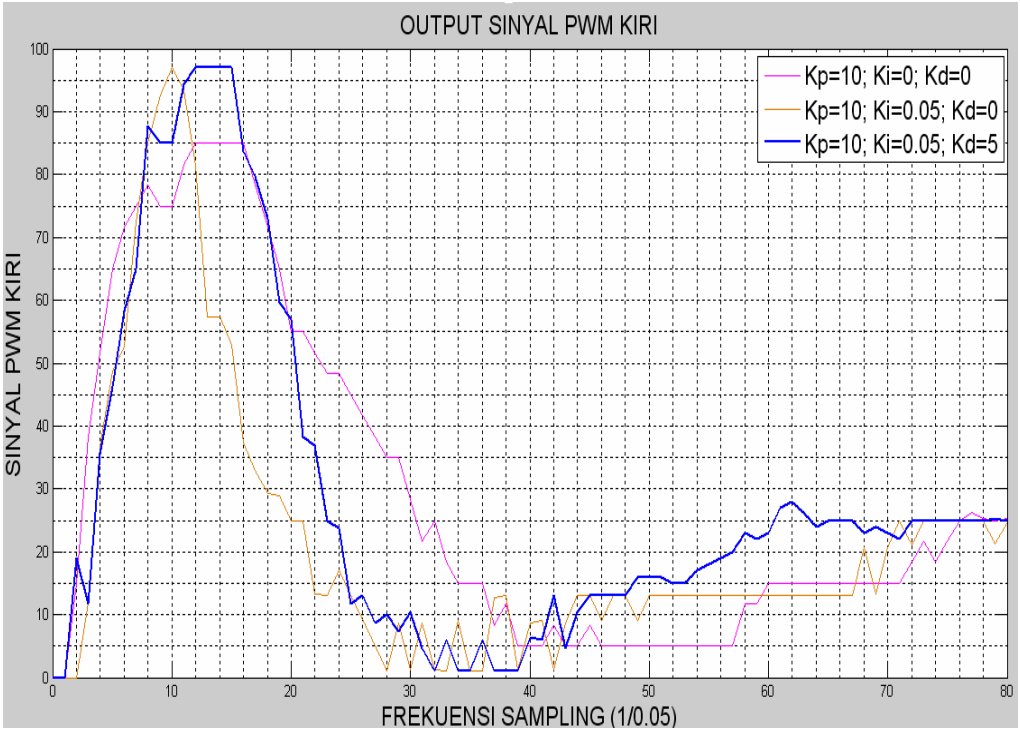
IV.3.1.4 Pengujian Hasil *Tuning* Terbaik pada Kontroler P, PI dan PID untuk *Reference* PWM 25 dan *Set Point* Jarak (8 cm)

Pada Sub Bab IV.3.1.4 akan dibahas mengenai pengujian hasil *tuning* PID, pengujian ini dimaksudkan sebagai pembanding antara kontroler P, PI dan PID untuk *reference* PWM 25 dan *set point* jarak (8 cm).

Gambar 4.7 memperlihatkan perbandingan kontroler P, PI dan PID,dan dapat dimpulkan bahwa kontroler PID memiliki respon yang cukup cepat dibanding kontroler P dan PI (*settling time* dan *rise time* yang dihasilkan). Selain itu kontroler PID memiliki kestabilan yang lebih baik dibanding kontrol P dan PI, karena lonjakan yang dihasilkan paling kecil (berhasil diredam).



Gambar 4.7 Posisi robot (proses variable) P (biru), PI (ungu) dan PID (hijau) untuk *reference* PWM 25 dan *set point* jarak (8 cm)

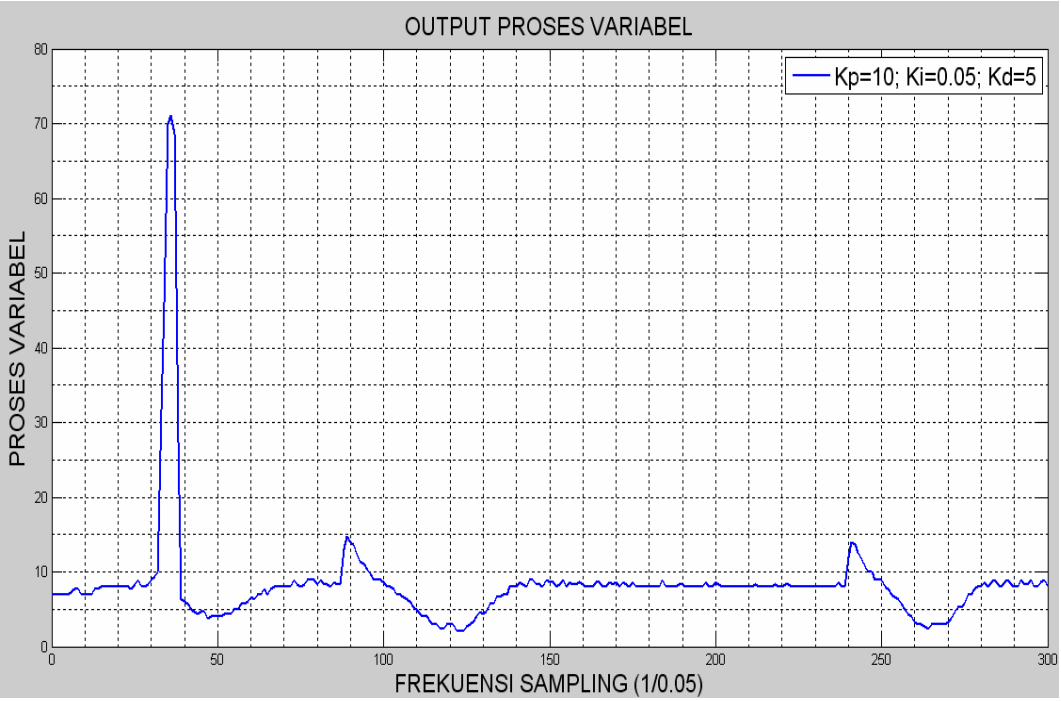


Gambar 4.8 Sinyal PWM motor DC kiri untuk kontroler P (ungu), PI (orange) dan PID (biru) dengan *reference* PWM 25 dan *set point* jarak (8 cm)

Dari Gambar 4.8 dapat dilihat bahwa sinyal PWM motor DC kiri juga memiliki respon/*settling time* yang paling cepat dibanding dengan kontroler P (ungu) dan kontroler PI (orange).

IV.3.1.5 Pengujian Pada Arena KRCI untuk *Reference* PWM 25 dan *Set Point* Jarak (8 cm)

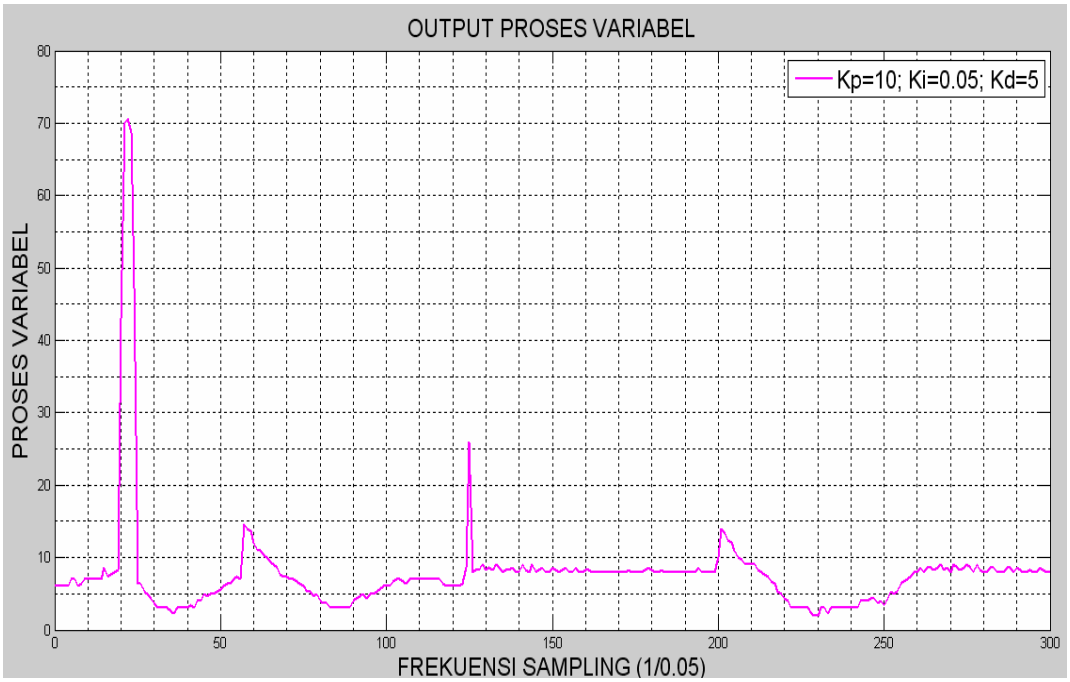
Pada pengujian di arena KRCI untuk *reference* PWM 25 dan *set point* jarak (8 cm) ini pengujian dibagi menjadi 2 kondisi, yaitu kondisi menggunakan *uneven floor* dan *furniture* ataupun juga kondisi saat tidak menggunakan *uneven floor* dan *furniture.*



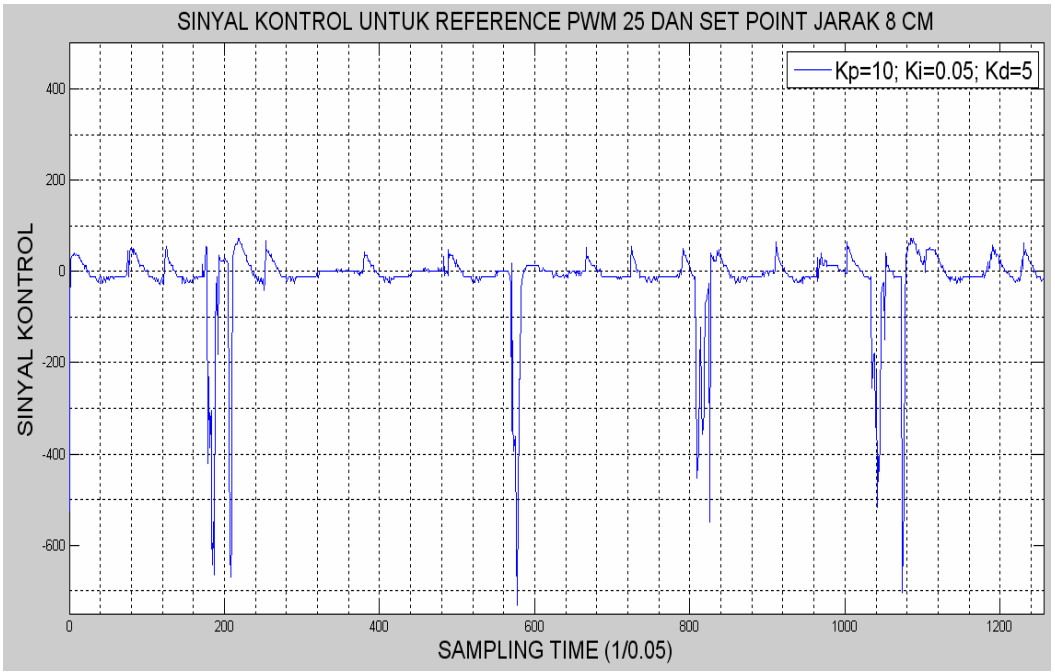
Gambar 4.9 Posisi robot (proses variable) untuk kontroler PID pada uji arena KRCI tanpa *uneven floor* dan *furniture* untuk *reference* PWM 25 dan *set point* jarak (8 cm)

Gambar 4.9 menunjukkan grafik posisi robot terhadap dinding yang diinginkan pada arena KRCI tanpa diberi gangguan berupa *uneven floor* dan *furniture* untuk *reference* PWM untuk pengujian nilai Kp, Ki, Kd yang telah didapat memberikan hasil yang baik. Dapat disimpulkan bahwa pemilihan nilai Kp, Ki, Kd yang telah didapat sanggup diterapkan pada lapangan KRCI. Lonjakan yang terjadi dihasilkan karena adanya tikungan pada arena KRCI.

Gambar 4.10 memperkuat pemilihan nilai Kp, Ki, Kd yang didapatkan, karena walaupun robot diberi gangguan, robot masih dapat mempertahankan *set point* jarak (8 cm) dengan baik.

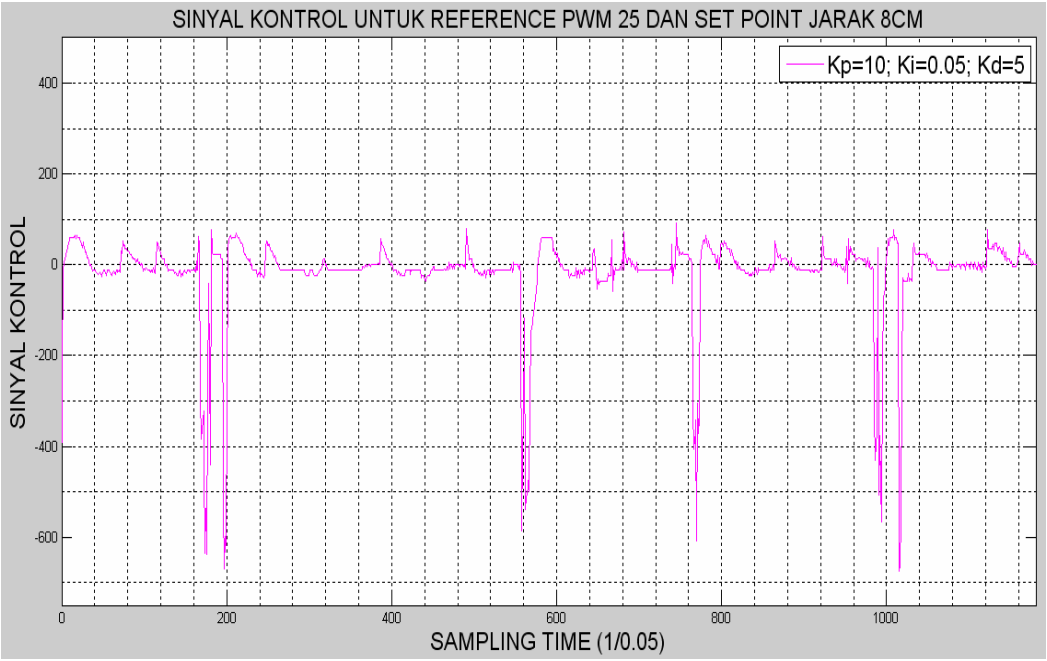


Gambar 4.10 Posisi robot **(**proses variable) untuk kontroler PID pada uji arena KRCI dengan *uneven floor* dan *furniture* untuk *reference* PWM 25 dan *set point* jarak (8 cm)



Gambar 4.11 Sinyal kontrol untuk kontroler PID pada uji arena KRCI tanpa *uneven floor* dan *furniture* untuk *reference* PWM 25 dan *set point* jarak (8 cm)

Gambar 4.11 merupakan hasil tuning PID terbaik dengan nilai parameter Kp=10, Ki=0.05, dan Kd=5 pada arena KRCI tanpa diberi gangguan seperti *uneven floor* dan *furniture*. Dari Gambar 4.11 dapat dilihat sinyal kontrol yang dihasilkan pada sampling waktu sekitar 200, 590, 810, 1200 dan 1400 terdapat osilasi yang sangat tajam kebawah, hal ini dikarenakan ada tikungan ke kiri yang tajam (sehingga robot harus berbelok ke kiri secara tiba-tiba) pada arena KRCI sehingga terdapat *error* yang besar secara tiba-tiba sehingga kontroler menghasilkan aksi kontrol negatif yang besar juga. Lonjakan-lonjakan ke atas pada sampling waktu sekitar 90, 110, 220 dan seterusnya, yang menandakan adanya tikungan kekanan sehingga menimbulkan *error* yang mengharuskan kontroler menghasilkan aksi kontrol positif. Namun apabila dianalisis pada daerah sampling waktu yang tidak terdapat tikungan pada arena, respon PID menghasilkan *output* sinyal kontrol yang baik (*settling time* yang cepat).



Gambar 4.12 Sinyal kontrol untuk kontroler PID pada uji arena KRCI dengan *uneven floor* dan *furniture* untuk *reference* PWM 25 dan *set point* jarak (8 cm)

Gambar 4.12 merupakan hasil tuning PID terbaik dengan nilai parameter Kp=10, Ki=0.05 dan Kd=5 pada arena KRCI dengan diberi gangguan seperti *uneven floor* dan *furniture*. Dari Gambar 4.12 dapat dilihat sinyal kontrol yang dihasilkan pada sampling waktu sekitar 180, 550, 760 dan 1000 terdapat osilasi

yang sangat tajam kebawah, hal ini dikarenakan ada tikungan pada arena KRCI sehingga terdapat *error* yang besar secara tiba-tiba sehingga robot menghasilkan aksi kontrol yang besar juga. Dan untuk respon *uneven floor* dan *furniture* dapat dilihat pada sampling waktu sekitar 650 sampai 700 dan sampling waktu sekitar 720 sampai 750. Dapat dilihat dengan penambahan *uneven floor* dan *furniture* respon dari sistem dapat dikatakan tidak terlalu terpengaruh oleh gangguan untuk *set point* jarak 8cm dan *reference* PWM 25.

Dari uji yang dilakukan untuk *reference* PWM 25 dan *set point* PWM jarak (8cm) dapat disimpulkan pemilihan nilai parameter Kd masih dapat menggunakan metoda *trial and error* dengan mengacu Ziegler Nichols II menggunakan metoda osilasi.

IV.3.2 Pengujian *Tuning* PID untuk *Reference* PWM 50 dan *Set Point* Jarak (8 cm)

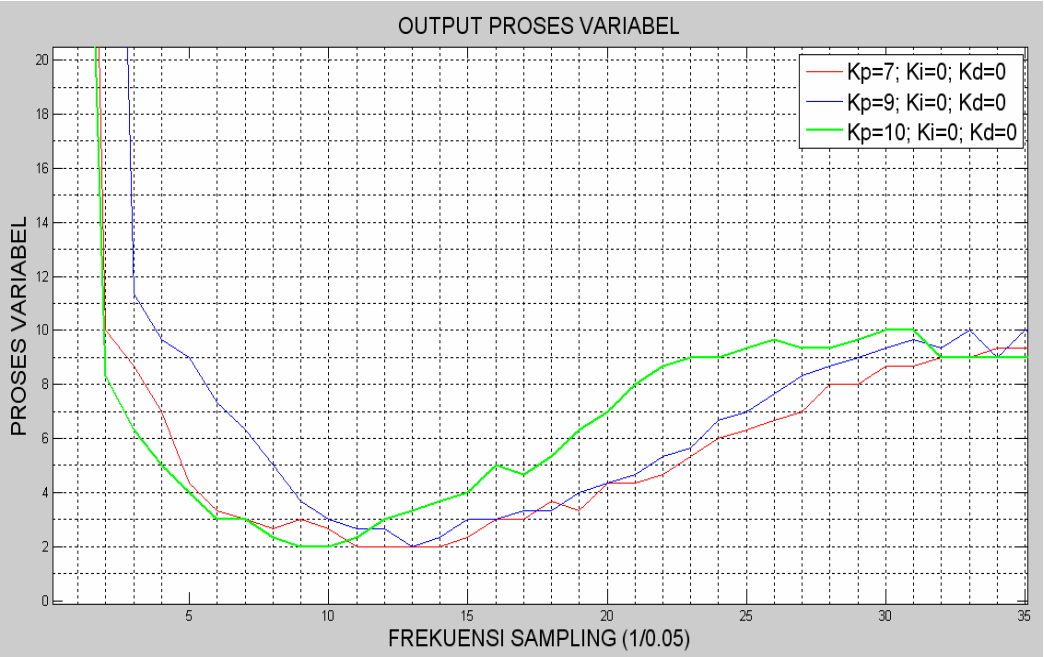
Pada Sub Bab IV.3.2 pengujian data yang diambil dengan nilai *reference* PWM 50, *set point* jarak (8 cm) untuk jenis kontroler P, PI dan PID antara lain adalah *output* proses variable/nilai bacaan sensor dan sinyal PWM untuk *actuator* motor DC kiri sama seperti pada Sub Bab IV.3.1 sebelumnya.

IV.3.2.1 *Tuning* Nilai Parameter *Proportional* (Kp) untuk *Reference* PWM 50 dan *Set Point* Jarak (8 cm)

Pada Sub Bab IV.3.2.1 pengujian data yang diambil untuk nilai *reference* PWM 50 dan *set point* jarak (8 cm) antara lain adalah *output* proses variabel (nilai bacaan sensor), dan sinyal PWM untuk *actuator* motor DC kiri.

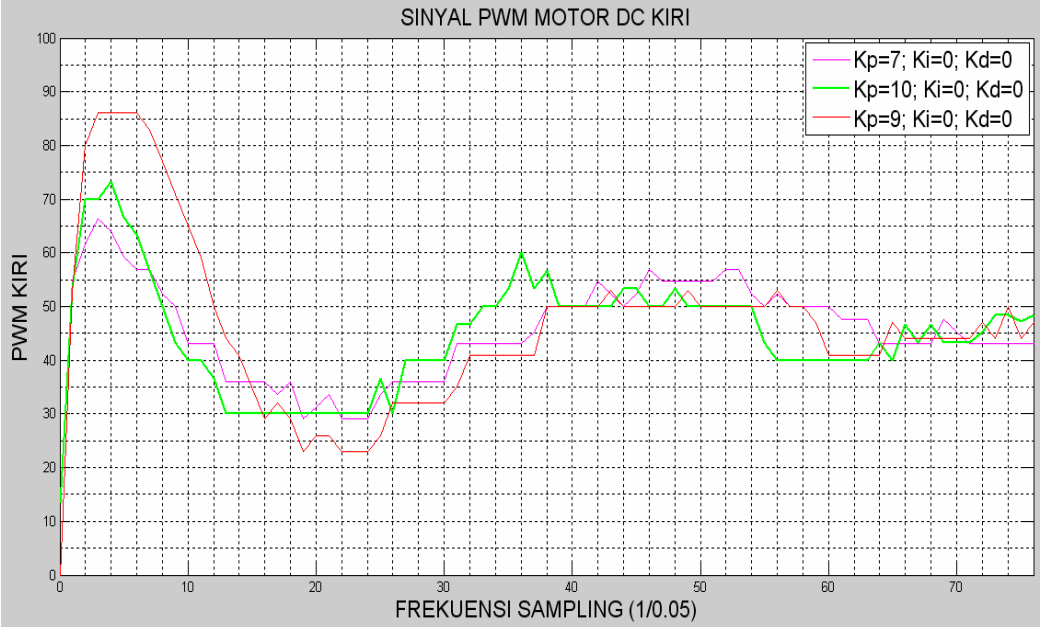
Pada Gambar 4.13 memperlihatkan gambar grafik posisi robot dengan dinding, menunjukkan bahwa penggunaan nilai Kp=10 memperlihatkan hasil

yang terbaik karena memiliki *settling time* yang paling cepat walaupun belum berhasil mencapai *set point* jarak (8 cm) yang diinginkan.



Gambar 4.13 Posisi robot (proses variable) atau nilai bacaan sensor *ultrasonic* kontroler *Proportional* untuk *reference* PWM 50 dan *set point* jarak (8 cm))

Gambar 4.14 menunjukkan bahwa nilai parameter *Proportional* (Kp) yang baik bernilai 10 karena memiliki *settling time* yang cukup cepat, *error steady state* terkecil dan memiliki osilasi yang cukup stabil dibanding yang lainnya. Gambar 4.14 menunjukkan bahwa bila nilai parameter Kp semakin besar maka *settling time* yang dihasilkan semakin cepat dalam mencapai *reference* PWM yang diinginkan, namun pemilihan parameter Kp yang terlalu besar juga akan membuat sistem justru semakin tidak stabil juga (sistem akan berosilasi terus menerus).

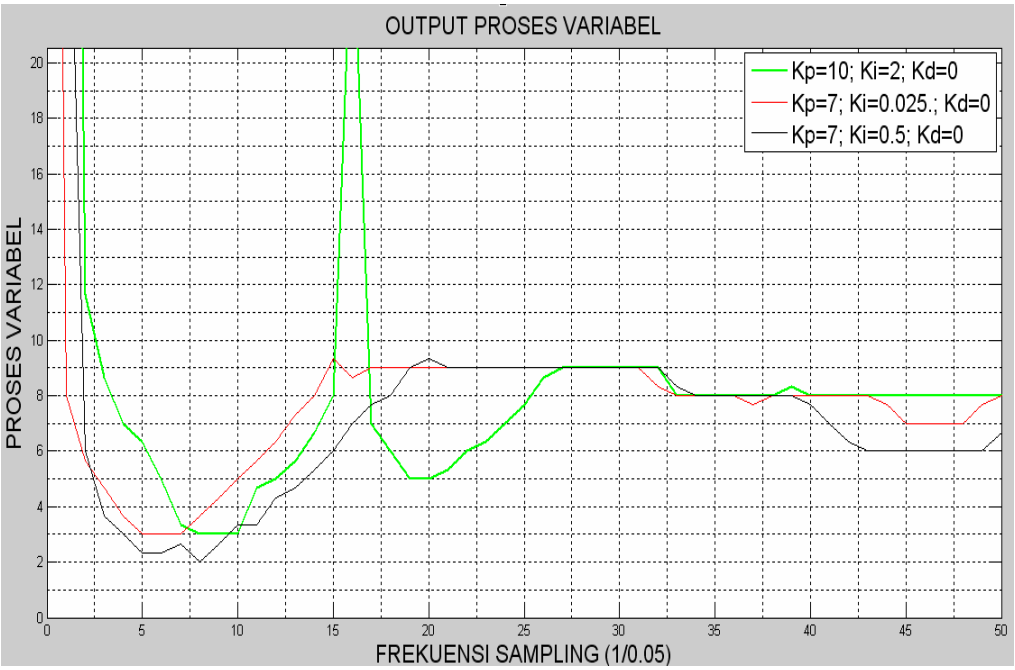


Gambar 4.14 Sinyal PWM motor DC kiri untuk kontroler *Proportional* pada *reference* PWM 50 dan *set point* jarak (8 cm)

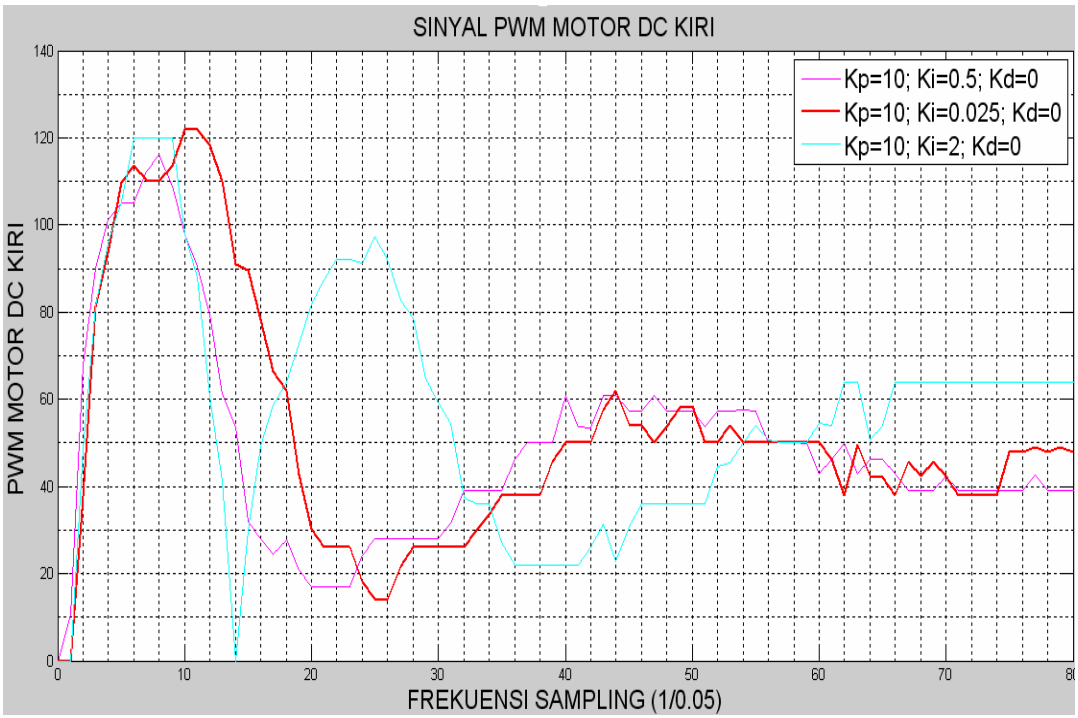
IV.3.2.2 Pengujian Nilai Parameter *Integral* (Ki) untuk *Reference* PWM 50 dan *Set Point* Jarak (8 cm)

Setelah didapatkan parameter *gain Proportional*, berikutnya adalah menentukan parameter *Integral* yang baik juga seperti pada Sub Bab IV.3.1.2 sebelumnya. Pada Sub Bab IV.3.2.2 pengujian data yang diamati untuk nilai *reference* PWM 50 dan *set point* jarak (8 cm) antara lain adalah posisi robot (proses variable), dan sinyal PWM untuk *actuator* motor DC kiri.

Gambar 4.15 adalah posisi robot (proses variable/nilai bacaan sensor *ultrasonic)* kontroler *Proportional Integral* untuk *reference* PWM 50 dan *set point* jarak (8 cm). Dari gambar tersebut diperlihatkan bahwa pemilihan parameter Ki yang baik bernilai 0.025, pemilihan nilai tersebut didasarkan dari *settling time* yang cukup cepat dalam mencapai *set point* jarak (8 cm) dan osilasi yang cukup stabil.



Gambar 4.15 Posisi robot (proses variable/nilai bacaan sensor *ultrasonic*) kontroler *Proportional Integral* untuk *reference* PWM 50 dan *set point* jarak (8 cm) (diperbesar/zoom)

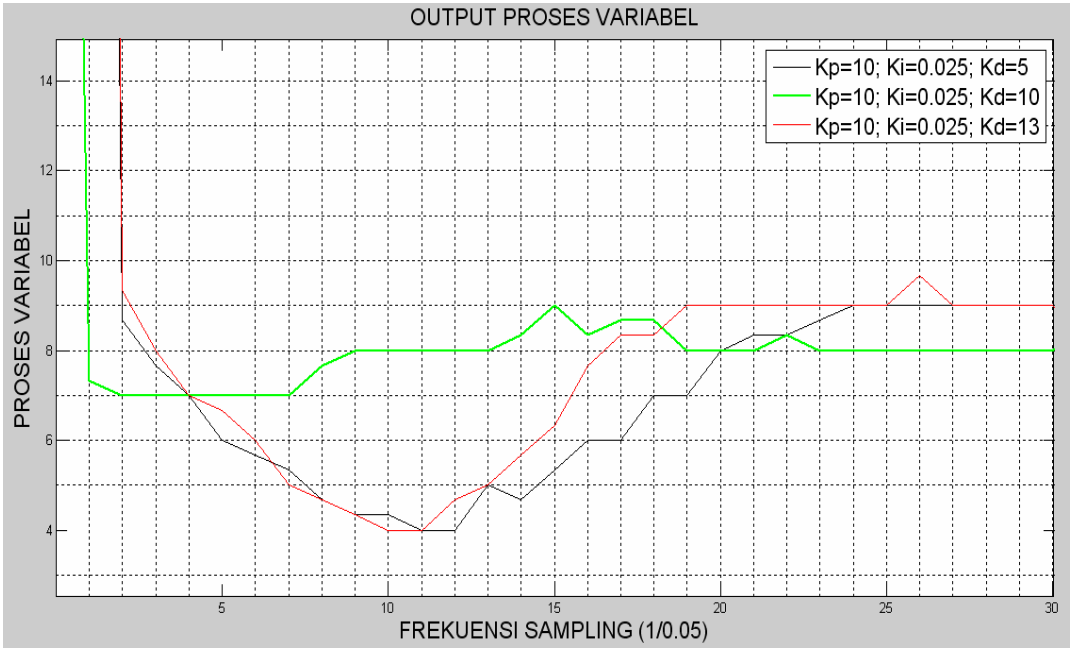


Gambar 4.16 Sinyal PWM motor DC kiri untuk kontroler *Proportional Integral* pada *reference* PWM 50 dan *set point* jarak (8 cm)

Sinyal PWM untuk *actuator* dapat dilihat pada Gambar 4.16. Pemilihan parameter Ki=0.025 semakin dikuatkan oleh hasil Gambar 4.16 yang menunjukkan bahwa respon yang paling mendekati *reference* diperlihatkan oleh nilai parameter Ki=0.025.

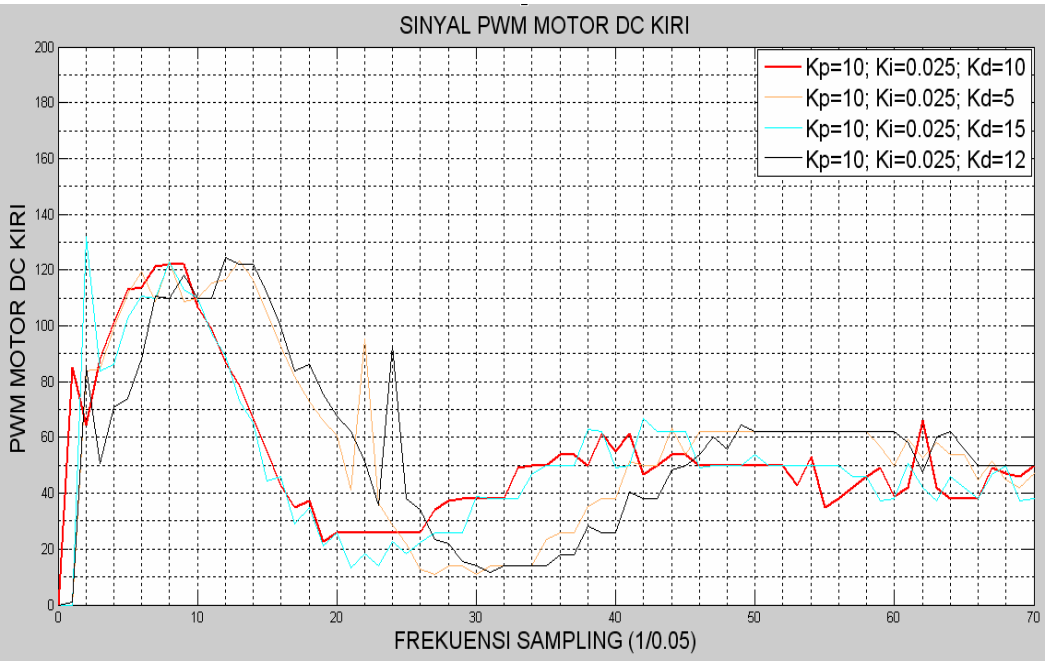
IV.3.2.3 *Tuning* Nilai Parameter *Derivative* (Kd) untuk *Reference* PWM 50 dan *Set Point* Jarak (8 cm)

Setelah didapatkan parameter *gain Proportional* dan *Integral*, berikutnya adalah menentukan parameter *Derivative* mengacu pada metoda Ziegler-Nichols II seperti pada Sub Bab IV.3.1.3 sebelumnya. Persamaan metoda Ziegler-Nichols II ditunjukkan pada Persamaan 4.1 sampai Persamaan 4.3.



Gambar 4.17 Posisi robot **(**proses variabel atau nilai bacaan sensor *ultrasonic*) kontroler *Proportional Integral Derivative* untuk *reference* PWM 50 dan *set point* jarak (8 cm) (perbesar/*zoom*)

Gambar 4.17 adalah sinyal yang dihasilkan dari hasil bacaan sensor *ultrasonic* terhadap dinding. Pada Gambar 4.17 menunjukkan bahwa pemilihan nilai Kd=10 memperlihatkan hasil yang terbaik karena memiliki *settling time* yang paling cepat mencapai *set point* jarak (8 cm).



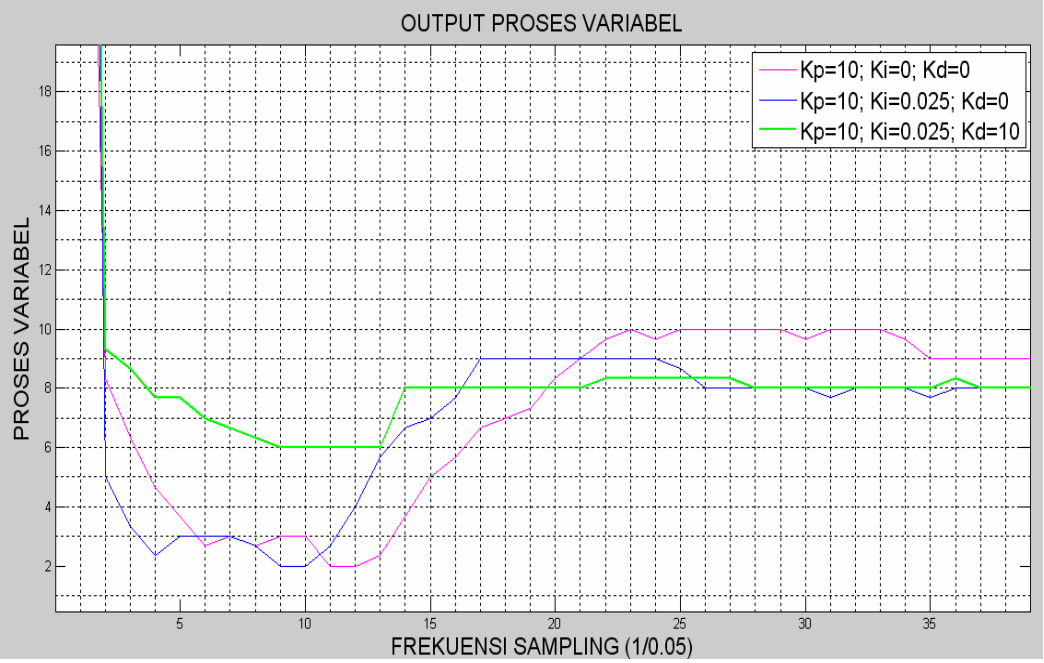
Gambar 4.18 Sinyal PWM motor DC kiri untuk kontroler PID dengan *reference* PWM 50 dan *set point* jarak (8 cm)

Gambar 4.18 semakin menguatkan analisis pemilihan nilai parameter Kd=10. Dapat dilihat untuk nilai parameter Kp=10, Ki, 0.025 dan Kd=10 memiliki *rise time* yang paling cepat dan *settling time* yang paling cepat juga dalam mencapai *reference* PWM 50 dibanding dengan nilai parameter Kd=5, 12 dan 15.

IV.3.2.4 Pengujian Hasil *Tuning* Terbaik pada Kontroler P, PI dan PID untuk *Reference* PWM 50 dan *Set Point* Jarak (8 cm)

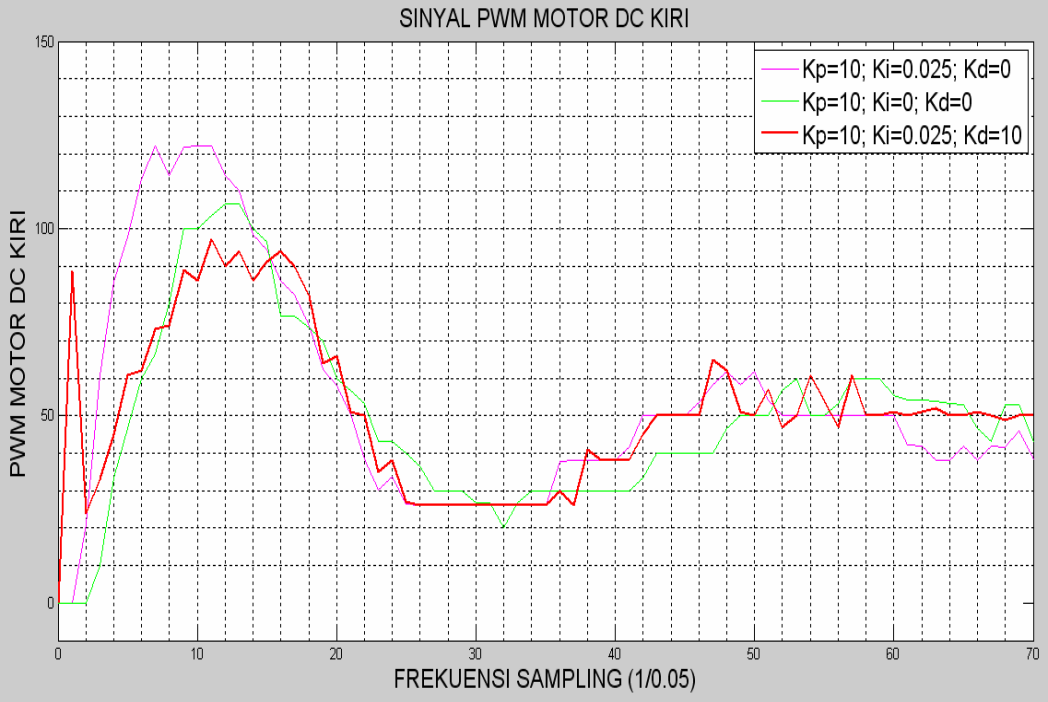
Pada Sub Bab IV.3.2.4 akan dibahas mengenai pengujian hasil *tuning* PID, pengujian ini dimaksudkan sebagai pembanding antara kontroler P, PI dan PID

untuk *reference* PWM 50 dan *set point* jarak (8 cm). Pada pengujian ini diambil dua acuan yang diuji diantaranya posisi robot (proses variable/nilai bacaan sensor *ultrasonic*), dan sinyal kontrol PWM pada *actuator* motor DC kiri untuk kontroler P, PI dan PID.



Gambar 4.19 Posisi robot **(**proses variabel atau nilai bacaan sensor *ultrasonic*) kontroler P (ungu), PI (biru) dan PID (hijau) untuk *reference* PWM 50 dan *set point* jarak (8 cm)

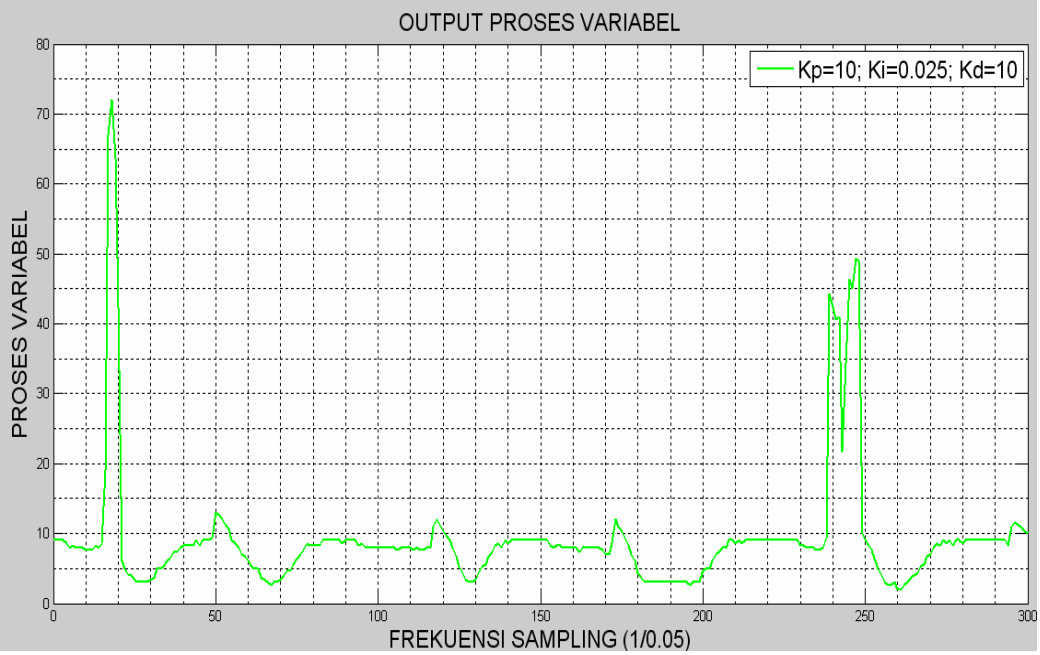
Gambar 4.19 menunjukkan bahwa grafik posisi robot dengan kontroler PID memiliki *settling time* yang cepat (mencapai *set point* jarak (8 cm)). Aksi kontrol pada *actuator* seperti pada Gambar 4.20, kontroler PID (merah) memiliki respon yang paling cepat dibanding dengan kontroler P (hijau) dan kontroler PI (ungu). Respon tersebut dapat dilihat dari redaman osilasi yang dihasilkan oleh masing-masing pengontrol seperti pada Gambar 4.20. Dapat disimpulkan bahwa penggunaan metoda *trial and error* menggunakan acuan Ziegler-Nichols II sebagai acuan *tuning* PID masih dapat diterapkan untuk *reference* PWM 50 dan *set point* jarak (8 cm).



Gambar 4.20 Sinyal PWM motor DC kiri untuk kontrol P (hijau), PI (ungu) dan PID (merah) pada *reference* PWM 50 dan *set point* jarak (8 cm)

IV.3.2.5 Pengujian Pada Arena KRCI untuk *Reference* PWM 50 dan *Set Point* Jarak (8 cm)

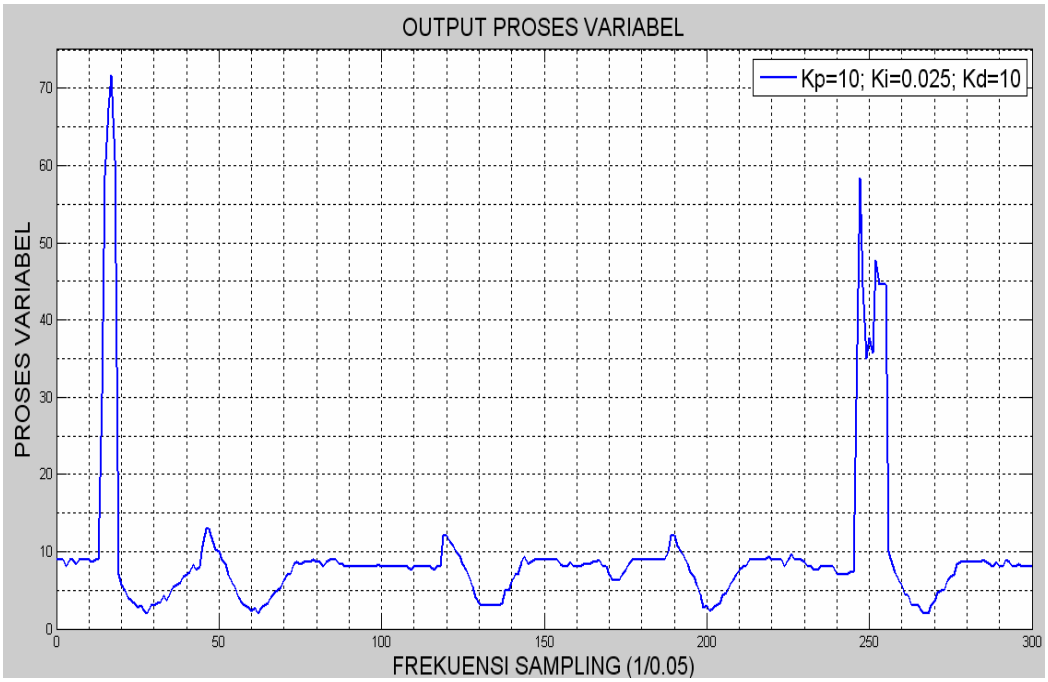
Pada pengujian di arena KRCI untuk *reference* PWM 50 dan *set point* jarak (8cm) ini pengujian dibagi menjadi 2 kondisi, yaitu kondisi menggunakan *uneven floor* dan *furniture* ataupun juga kondisi saat tidak menggunakan *uneven floor* dan *furniture.*



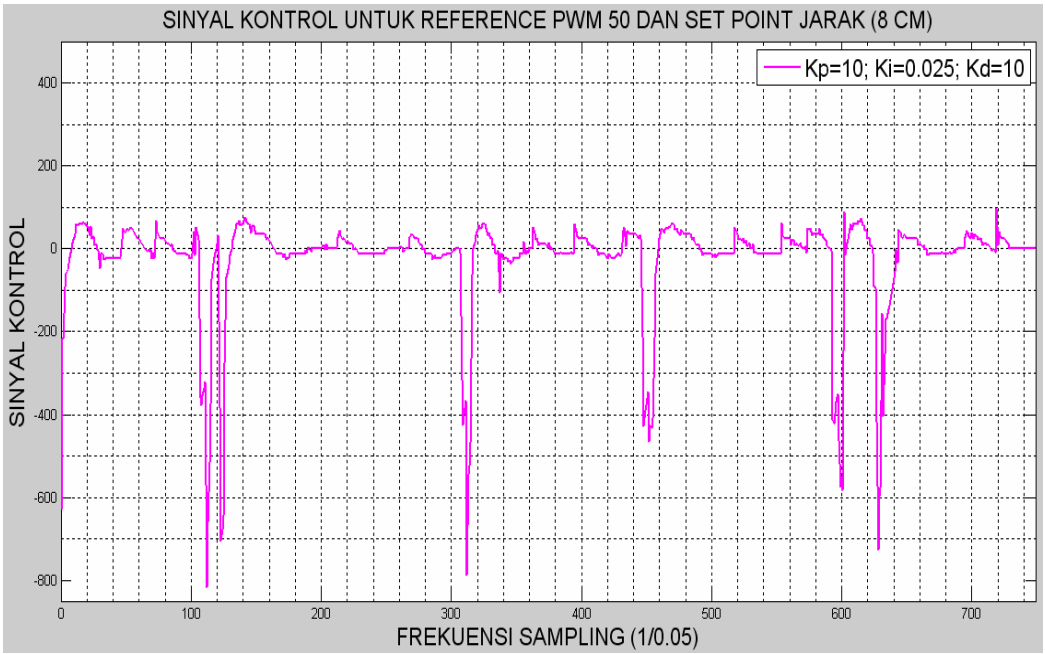
Gambar 4.21 Posisi robot (proses variable) untuk kontroler PID pada uji arena KRCI tanpa *uneven floor* dan *furniture* pada *reference* PWM 50 dan *set point* jarak (8cm)

Pada Gambar 4.21 dapat dilihat posisi robot atau jarak robot terhadap dinding pada arena KRCI tanpa diberi gangguan (*uneven floor* dan *furniture*) untuk *reference* PWM 50 dan *set point* jarak (8 cm) dengan nilai Kp, Ki, Kd yang telah diperoleh, dan dapat disimpulkan bahwa pemilihan nilai Kp, Ki, Kd sanggup diterapkan pada lapangan KRCI karena dapat menanggulangi *error* yang terjadi setiap kali robot berbelok pada tikungan. Lonjakan pada Gambar 4.21 menandakan robot yang berbelok.

Gambar 4.22 memperlihatkan juga bahwa pemilihan Kp, Ki dan Kd yang telah didapat sebelumnya masih dapat digunakan pada lapangan KRCI yang telah diberi gangguan *uneven floor* dan *furniture* untuk *reference* PWM 50 dan *set point* jarak (8 cm).



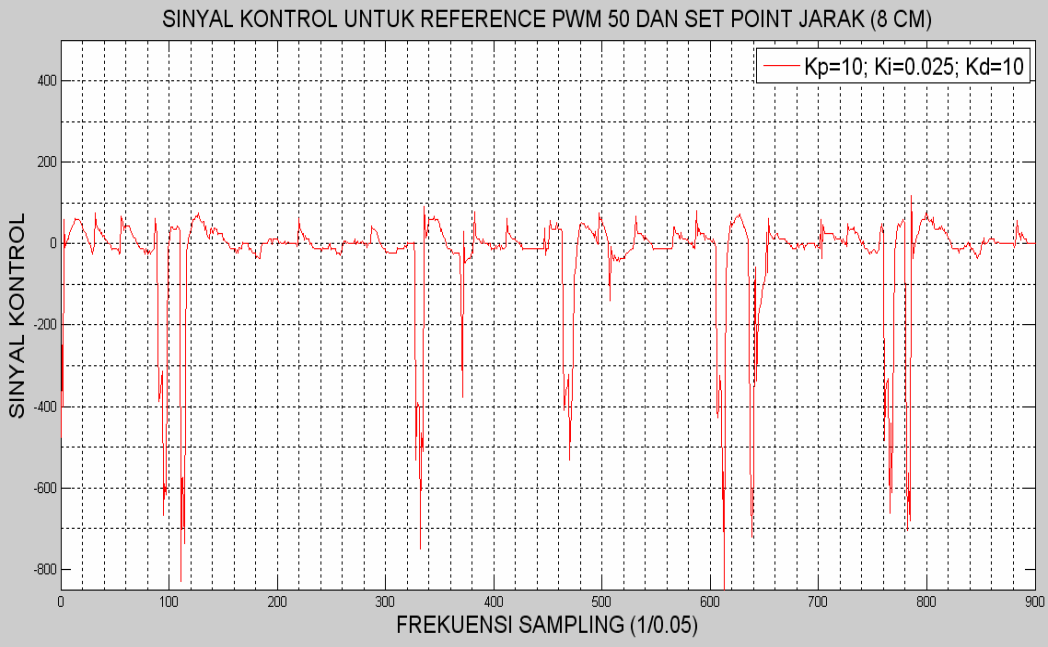
Gambar 4.22 *Output* proses variabel untuk kontroler PID pada uji arena KRCI menggunakan *uneven floor* dan *furniture* pada *reference* PWM 50 dan *set point* jarak (8 cm)



Gambar 4.23 Sinyal kontrol untuk kontroler PID pada uji arena KRCI tanpa *uneven floor* dan *furniture* dengan *reference* PWM 50 dan *set point* jarak (8 cm)

Gambar 4.23 menunjukkan bahwa sinyal kontrol yang dihasilkan kontroler PID tanpa *uneven floor* dan *furniture* mampu meredam osilasi yang dihasilkan pada setiap tikungan pada lapangan KRCI dengan *settling time* yang cepat.

Gambar 4.24 memperkuat kesimpulan yang telah didapat bahwa pemilihan nilai parameter Kp, Ki, Kd yang didapat sudah tepat karena walaupun diberi gangguan berupa *uneven floor* dan *furniture*, sistem masih dapat meredam osilasi yang ditimbulkan oleh *uneven floor* dan *furniture* dengan *settling time* yang cepat.



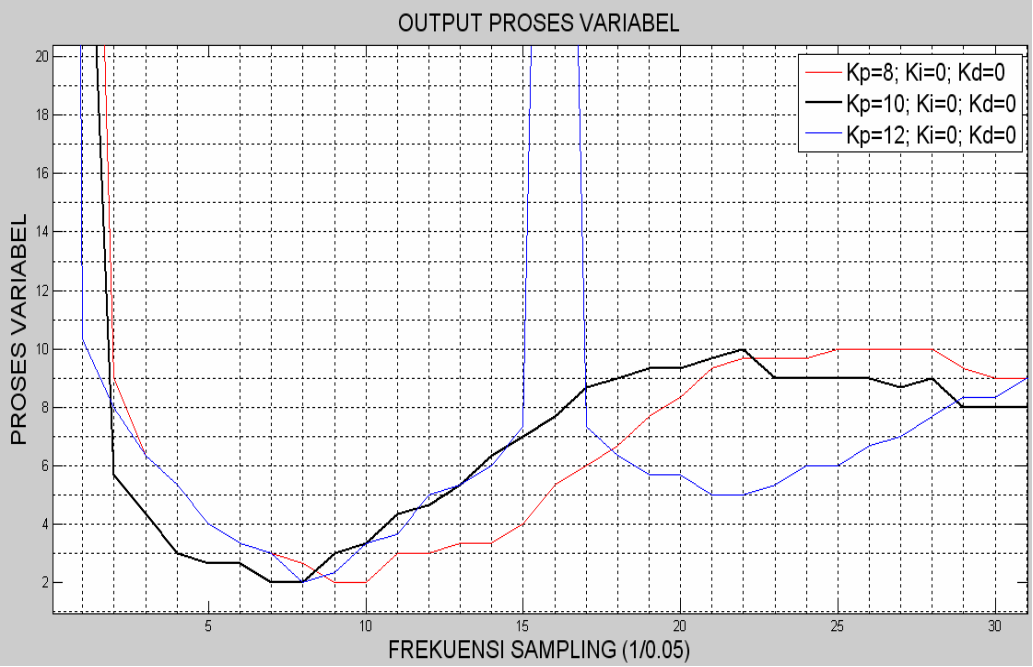
Gambar 4.24 Sinyal kontrol untuk kontroler PID pada uji arena KRCI menggunakan *uneven floor* dan *furniture* dengan *reference* PWM 50 dan *set point* jarak (8 cm)

IV.3.3 Pengujian *Tuning* PID untuk *Reference* PWM 75 dan *Set Point* Jarak (8 cm)

Pada Sub Bab IV.3.3 pengujian data yang diambil untuk nilai *reference* PWM 75 dan *set point* jarak (8 cm) antara lain adalah posisi robot, dan sinyal kontrol untuk *actuator* motor DC kiri.

IV.3.3.1 *Tuning* Nilai Parameter *Proportional* (Kp) untuk *Reference* PWM 75 dengan *set point* jarak (8 cm)

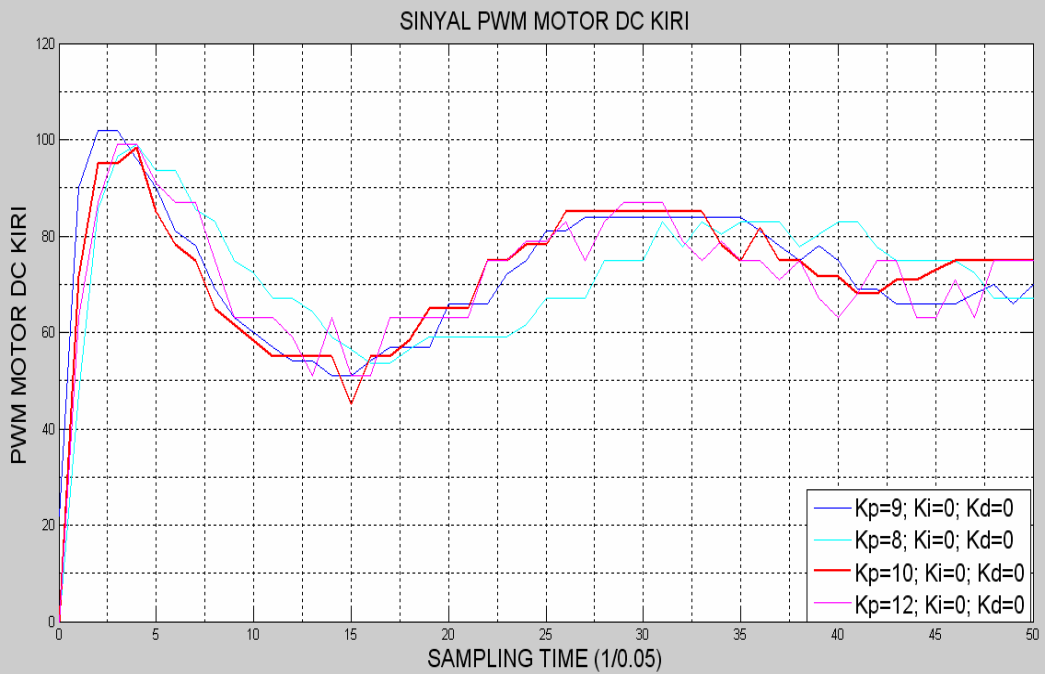
Pada Sub Bab IV.3.3.1 pengujian data yang diambil untuk nilai *reference* PWM 75 dan *set point* jarak (8 cm) antara lain adalah posisi robot (proses variable), dan sinyal kontrol untuk *actuator* motor DC kiri.



Gambar 4.25 Posisi robot **(**proses variable) untuk kontroler *Proportional* dengan *reference* PWM 75 *set point* jarak (8 cm) (diperbesar/*zoom*)

Gambar 4.25 adalah grafik posisi robot, menunjukkan pemilihan nilai Kp yang tepat adalah 10 karena memiliki *settling time* yang paling cepat dalam mencapai *set point* jarak (8 cm).

Gambar 4.26 memperlihatkan sinyal PWM motor kiri dari kontroler *Proportional.* Pemilihan Kp=10 semakin dikuatkan kembali oleh Gambar 4.26 yang memperlihatkan *settling time* yang cepat dalam mencapai *reference* PWM yang diinginkan.

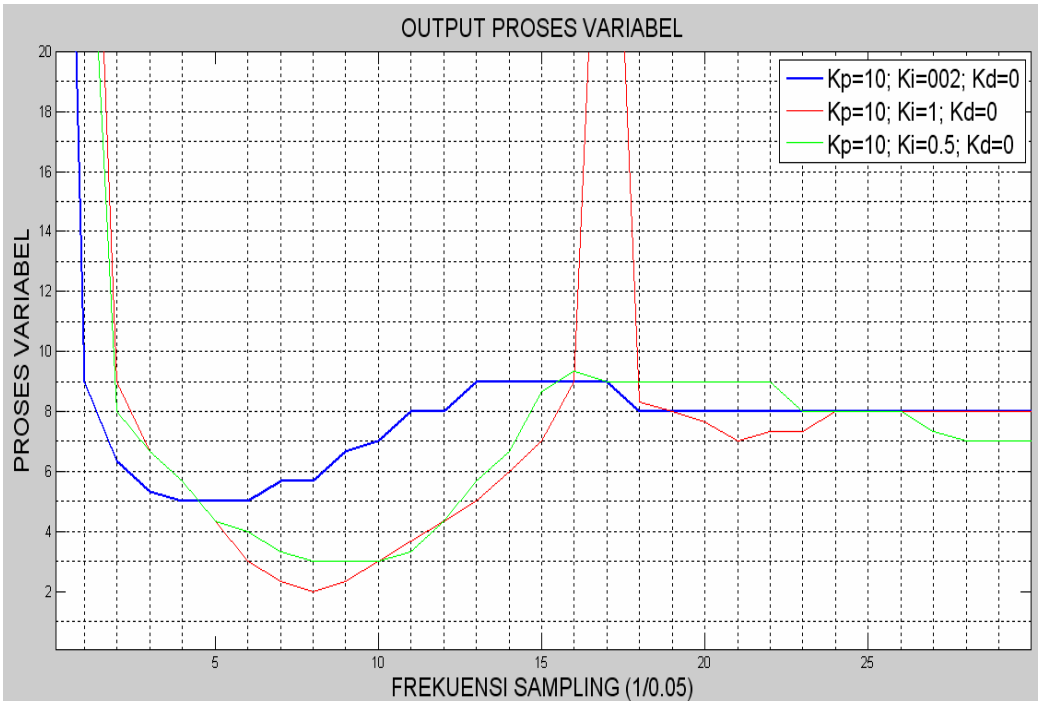


Gambar 4.26 Sinyal PWM motor DC kiri untuk kontroler *Proportional* dengan *reference* PWM 75 *set point* jarak (8 cm)

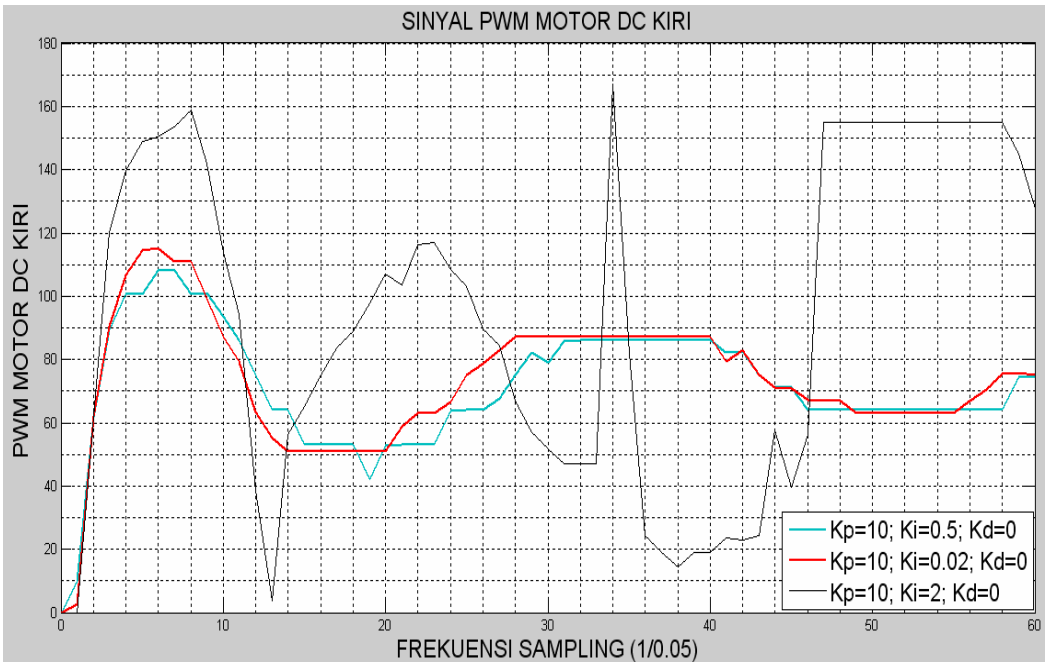
Dari uji yang dilakukan didapatkan kesimpulan kembali untuk nilai parameter Kp semakin besar maka *settling time* yang dihasilkanpun semakin cepat, akan tetapi pemilihan parameter Kp yang terlalu besar juga akan membuat sistem justru semakin tidak stabil juga (sistem akan berosilasi terus menerus).

IV.3.3.2 *Tuning* Nilai Parameter *Integral* (Ki) untuk *Reference* PWM 75 dan *Set Point* Jarak (8 cm)

Setelah didapatkan parameter *gain Proportional*, berikutnya adalah menentukan parameter *Integral.* Pada Sub Bab IV.3.3.2 pengujian data yang diamati untuk nilai *reference* PWM 75 dan *set point* jarak (8 cm) antara lain adalah posisi robot, dan sinyal PWM motor DC kiri.



Gambar 4.27 *Output* proses variabel untuk kontroler *Proportional Integral* dengan *reference* PWM 75 dan *set point* jarak (8 cm) (diperbesar/*zoom*)



Gambar 4.28 Sinyal PWM motor DC kiri untuk kontroler *Proportional Integral* pada *actuator* untuk *reference* PWM 75 dan *set point* jarak (8cm)

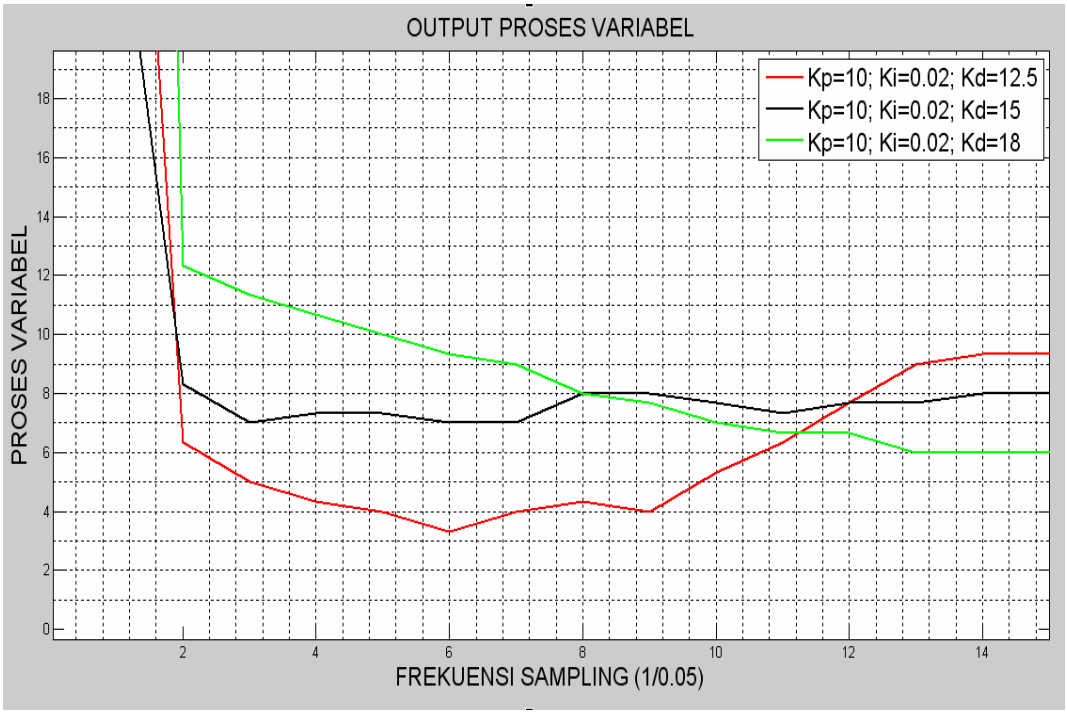
Gambar 4.27 adalah posisi robot (proses variable) dari kontroler, menunjukkan bahwa nilai Ki=0.02 memperlihatkan *settling time* yang paling cepat mencapai *set point.*

Gambar 4.28 memperlihatkan sinyal PWM motor DC kiri untuk kontroler *Proportional Integral* dengan *reference* PWM 75 dan *set point* jarak (8 cm). Pemilihan parameter Ki=0.02 semakin dikuatkan, hal tersebut dapat terlihat pada Gambar 4.28 yang menunjukan gambar dari sinyal kontrol PWM motor DC kiri yang dihasilkan kontroler PI. Dari Gambar 4.28 terlihat bahwa pada Ki=0.02 sinyal kontrol yang dihasilkan lebih stabil dan lebih cepat mencapai *reference* PWM yang diinginkan yaitu 75.

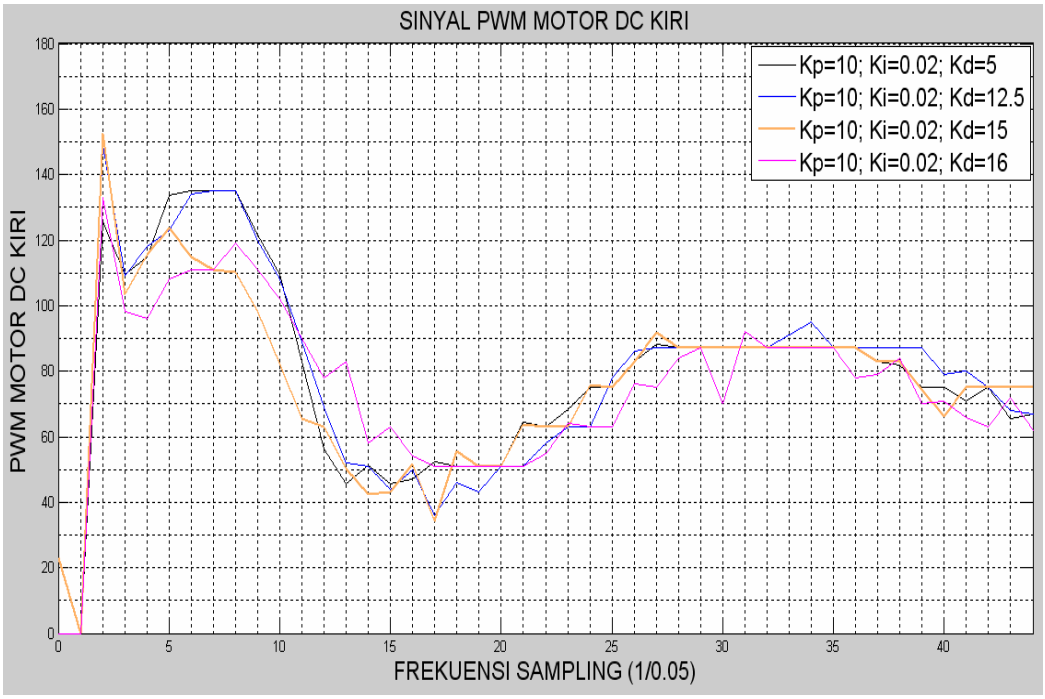
IV.3.3.3 *Tuning* Nilai Parameter *Derivative* (Kd) untuk *Reference* PWM 75 dan *Set Point* Jarak (8 cm)

Setelah didapatkan parameter *gain Proportional* dan *Integral*, berikutnya adalah menentukan parameter *Derivative*. Pada *reference* PWM 75 ini penentuan nilai Kd menggunakan metoda *trial and error* dan tidak lagi mengacu pada metoda Ziegler-Nichols II. Pada Gambar 4.29 menunjukkan hasil dari pemilihan nilai Kd menggunakan acuan Ziegler-Nichols II dan *trial and error.* Apabila nilai Kd dicari berdasar acuan Ziegler-Nichols II untuk nilai Ki=0.02, maka didapatkan hasil Kd=12.5.

Dari Gambar 4.29 terlihat jelas bahwa pemilihan nilai Kd yang baik untuk *reference* PWM 75 tidak lagi mengacu pada metoda Ziegler-Nichols II. Hasil nilai Kd yang terbaik didapat dari metoda *trial and error*, bukan mengacu pada Ziegler-Nichols II, yaitu pada Kd=15. Untuk nilai Kd=15 terlihat jelas pada Gambar 4.29 bahwa posisi robot yang diinginkan berhasil mencapai *set point* jarak (8cm) seperti yang diharapkan.



Gambar 4.29 Posisi robot (proses variable) untuk kontroler *Proportional Integral Derivative* untuk *reference* PWM 75 dan *set point* jarak (8cm)

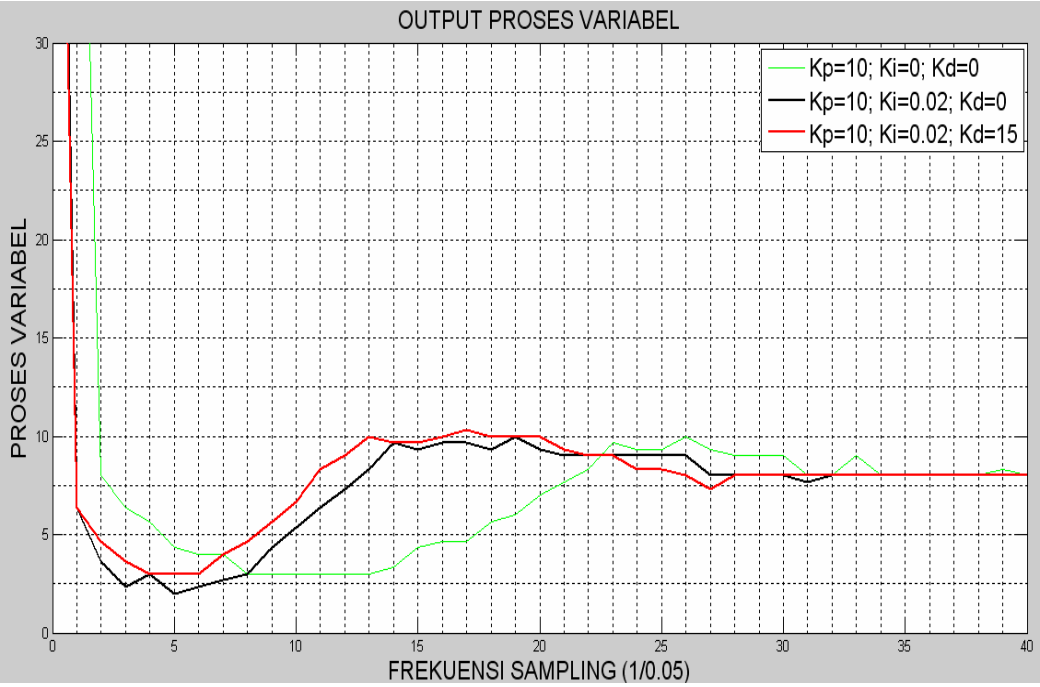


Gambar 4.30 Sinyal PWM motor DC kiri untuk kontroler *Proportional Integral Derivative* dengan *reference* PWM 75 dan *set point* jarak (8cm)

Gambar 4.30 memperlihatkan sinyal PWM motor DC kiri yang dihasilkan kontroler PID. Dari gambar tampak bahwa pada Kd=15, sistem berhasil mencapai *reference* PWM yang diinginkan yaitu 75. Sedangkan untuk Kd=12,5 dan seterusnya sistem masih berosilasi kembali, ini menyatakan bahwa *settling time* tercepat ada pada nilai Kd=15.Dari uji yang dilakukan untuk *reference* PWM 75 dan *set point* PWM jarak (8cm) dapat disimpulkan pemilihan nilai parameter Kd tidak lagi menggunakan metoda *trial and error* dengan mengacu Ziegler Nichols II.

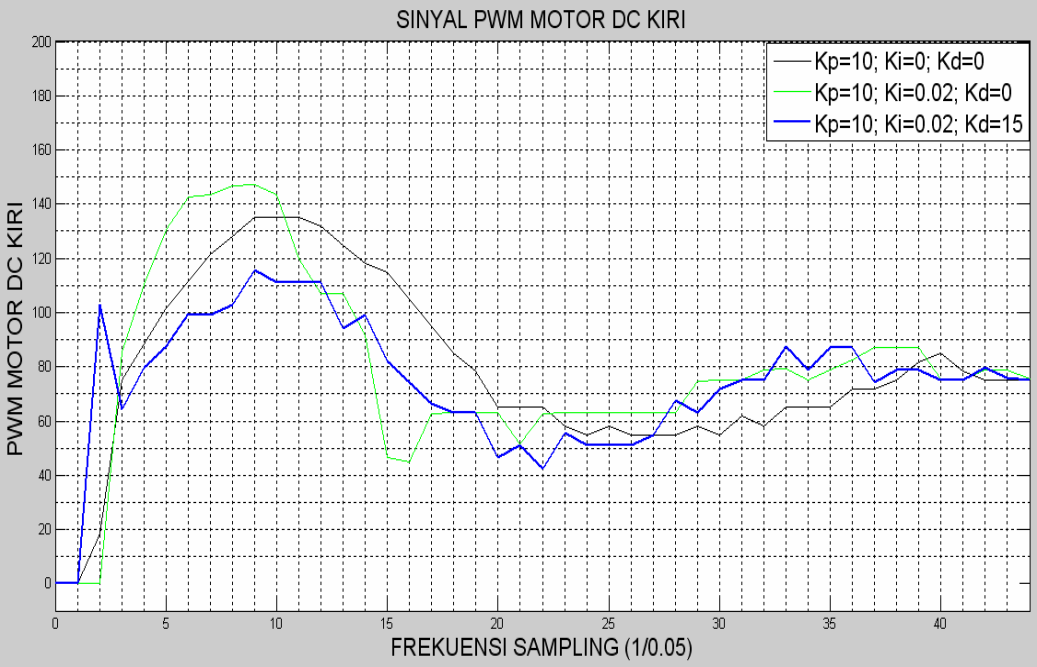
IV.3.3.4 Pengujian Hasil *Tuning* Terbaik pada Kontroler P, PI dan PID untuk *Reference* PWM 75 dan *Set Point* Jarak (8 cm)

Pada Sub Bab IV.3.3.4 akan dibahas mengenai pengujian hasil *tuning* PID, pengujian ini dimaksudkan sebagai pembanding antara kontroler P, PI dan PID untuk *reference* PWM 75 dan *set point* jarak (8 cm).



Gambar 4.31 Posisi robot (proses variable) kontroler P (hijau), PI (hitam), dan PID (merah) untuk *reference* PWM 75 dan *set point* jarak (8 cm)

Gambar 4.31 memperlihatkan grafik posisi berupa jarak antar robot dengan dinding (proses variable), terlihat bahwa posisi robot terbaik adalah kontroler PID karena memiliki *settling time* yang cepat.



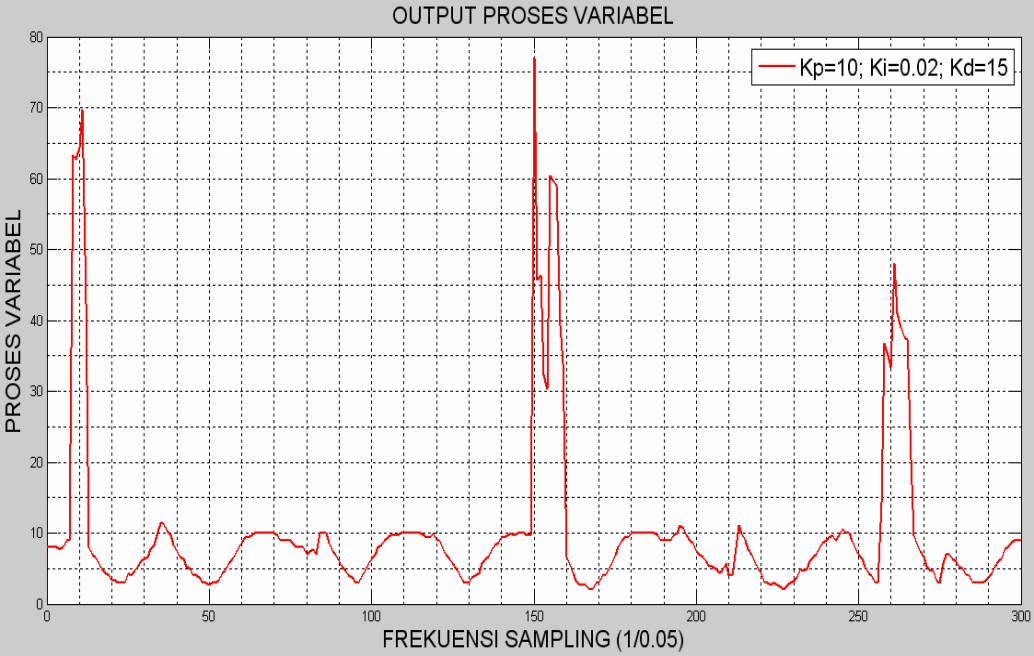
Gambar 4.32 Sinyal PWM motor DC kontroler untuk P (hitam), PI (hijau) dan PID (biru) dengan *reference* PWM 75 *set point* jarak (8 cm)

Respon sinyal PWM motor DC kiri seperti pada Gambar 4.32 kontroler PID (biru) juga memiliki respon yang paling cepat dibanding dengan kontroler P (hitam) dan kontroler PI (hijau). Respon tersebut dapat dilihat dari redaman osilasi yang dihasilkan oleh masing-masing pengontrol seperti pada Gambar 4.32.

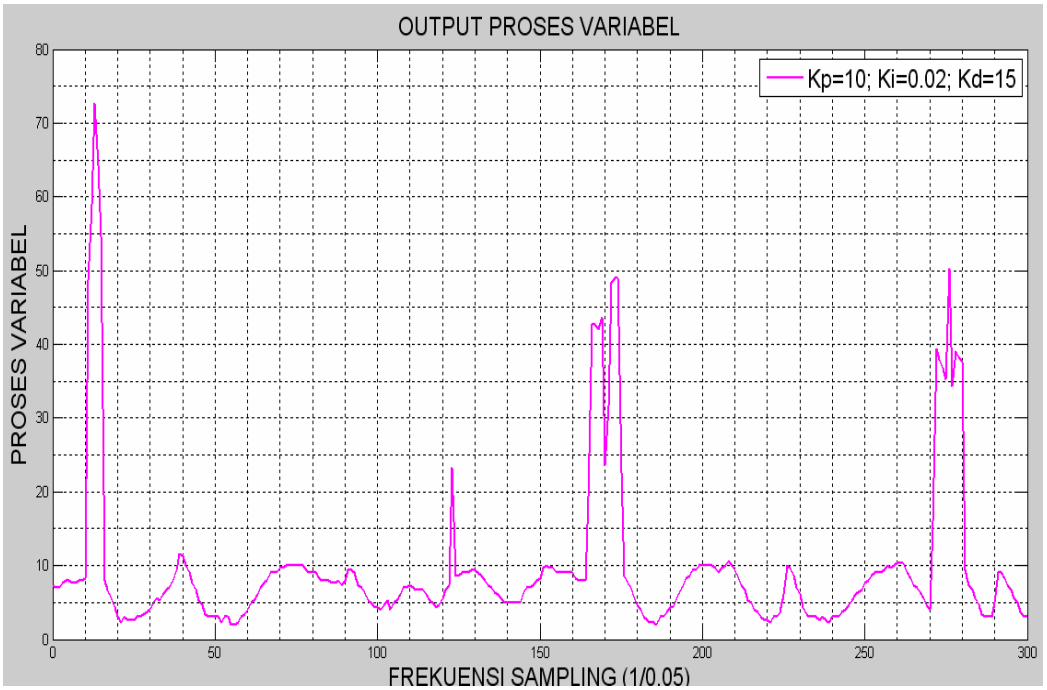
IV.3.3.5 Pengujian Pada Arena KRCI untuk *Reference* PWM 75 dan *Set Point* Jarak (8cm)

Pada pengujian di arena KRCI untuk *reference* PWM 75 ini pengujian dibagi menjadi 2 kondisi, yaitu kondisi menggunakan *uneven floor* dan *furniture* ataupun juga kondisi saat tidak menggunakan *uneven floor* dan *furniture*.

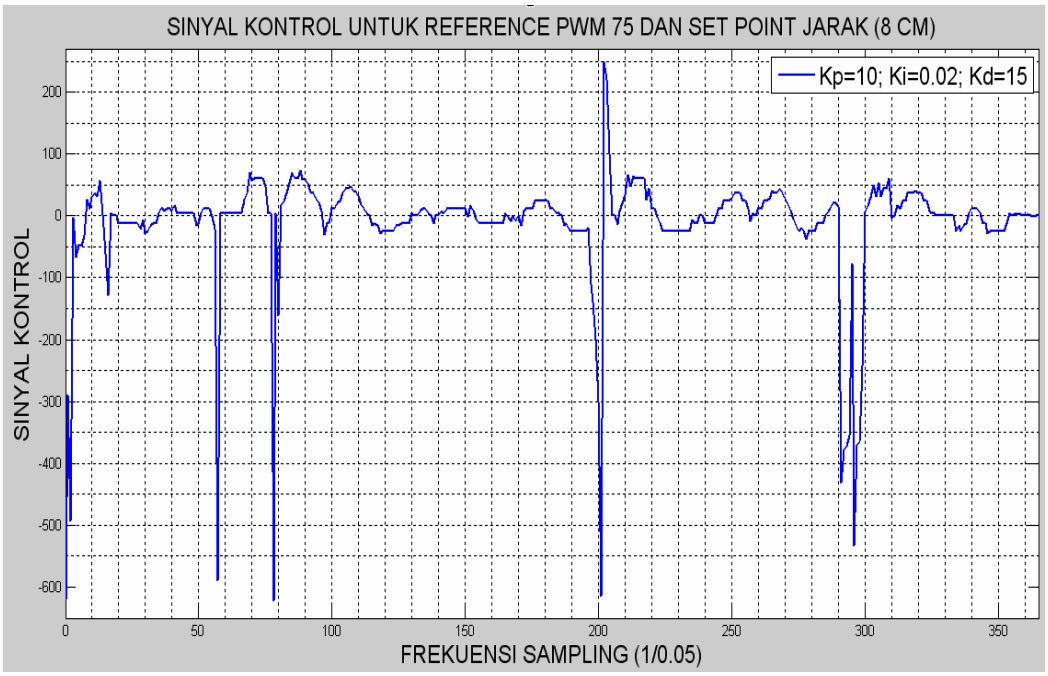
Gambar 4.33 dan Gambar 4.34 memperlihatkan *output* yang diharapkan. Dari gambar tersebut dapat disimpulkan bahwa pemilihan nilai Kp, Ki dan Kd yang telah didapatkan sebelumnya mampu digunakan pada arena KRCI walaupun diberi gangguan berupa *uneven floor* dan *furniture*.



Gambar 4.33 *Output* proses variabel pada uji arena KRCI tanpa *uneven floor* dan *furniture* untuk *reference* PWM 75 dan *set point* jarak (8 cm)

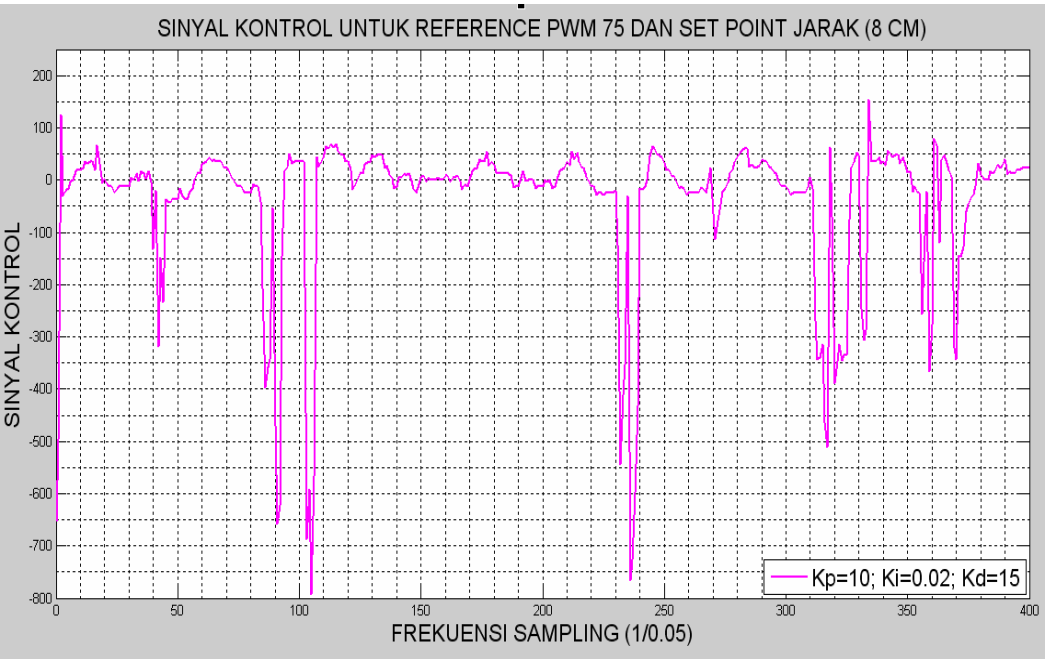


Gambar 4.34 *Output* proses variabel pada uji arena KRCI dengan *uneven floor* dan *furniture* untuk *reference* PWM 75 dan *set point* jarak (8 cm)



Gambar 4.35 Sinyal kontrol untuk kontroler PID pada uji arena KRCI tanpa *uneven floor* dan *furniture* untuk *reference* PWM 75 dan *set point* jarak (8 cm)

Gambar 4.35 merupakan hasil tuning PID terbaik dengan nilai parameter Kp=10, Ki=0.02 dan Kd=15 pada arena KRCI tanpa diberi gangguan seperti *uneven floor* dan *furniture*. Dari Gambar 4.65 dapat dilihat sinyal kontrol yang dihasilkan pada sampling waktu sekitar 55, 75, 200 dan 290 terdapat osilasi yang sangat tajam kebawah, hal ini dikarenakan ada tikungan ke kiri yang tajam (sehingga robot harus berbelok ke kiri secara tiba-tiba) pada arena KRCI, sehingga terdapat *error* yang besar secara tiba-tiba sehingga kontroler menghasilkan aksi kontrol negatif yang besar juga. Lonjakan-lonjakan ke atas pada sampling waktu 10, 202, 250 dan seterusnya, menandakan ada tikungan kekanan sehingga menimbulkan *error* yang mengharuskan kontroler menghasilkan aksi kontrol untuk positif. Akan tetapi apabila dianalisis menurut kawasan waktu yang tidak terdapat tikungan pada arena, respon PID yang dihasilkan bisa dibilang menghasilkan sinyal kontrol yang baik karena masih dapat meredam *error* yang timbul.



Gambar 4.36 Sinyal kontrol untuk kontroler PID pada uji arena KRCI dengan *uneven floor* dan *furniture* untuk *reference* PWM 75 dan *set point* jarak (8 cm)

Gambar 4.36 merupakan hasil tuning PID terbaik dengan nilai parameter Kp=10, Ki=0.02 dan Kd=15 pada arena KRCI dengan diberi gangguan seperti

uneven floor dan *furniture*. Dari Gambar 4.66 dapat dilihat sinyal kontrol yang dihasilkan pada sampling waktu sekitar 90, 105, 235 dan 315 terdapat osilasi yang sangat tajam kebawah, hal ini dikarenakan ada tikungan ke kiri yang tajam (sehingga robot harus berbelok ke kiri secara tiba-tiba) pada arena KRCI, sehingga terdapat *error* yang besar secara tiba-tiba sehingga kontroler menghasilkan aksi kontrol negatif yang besar juga. Respon *uneven floor* dan *furniture* dapat dilihat pada sampling waktu sekitar 45, 270 dan 335, dan sistem masih sanggup mengatasi gangguan yang diberikan.

IV.4 Pengujian Pembanding Kontoler P, PI dan PID

Pada Sub Bab IV.4 dilakukan pengujian terhadap kontroler P, PI dan PID pada robot untuk memadamkan api. Dari Tabel 4.3a, 4.3b, dan 4.3c dapat disimpulkan bahwa kontroler PID memiliki respon yang cepat daripada kontroler P dan PI, walaupun ketiganya dapat memadamkan api dan balik *home* baik untuk *reference* PWM 25, 50 dan 75, akan tetapi kontroler PID membutuh waktu yang lebih sedikit daripada kontroler P dan PI.

Tabel 4.3a Pengujian kontroler P, PI dan PID pada robot untuk memadamkan api (*reference* PWM 25 dan *set point* jarak (8cm))

Titik Api	Home	Waktu		Keterangan		Kontroler (reference PWM 25)
		Memadamkan Api	Balik Home	Balik Home	Memadamkan Api	
Ruang 1	*Non Arbitary Start*	28.54	86.31	Berhasil	Berhasil	P
Ruang 1	*Non Arbitary Start*	27.01	85.21	Berhasil	Berhasil	PI
Ruang 1	*Non Arbitary Start*	24.44	86.12	Berhasil	Berhasil	PID

Tabel 4.3b Pengujian kontroler P, PI dan PID pada robot untuk memadamkan api (*reference* PWM 50 dan *set point* jarak (8cm))

Titik Api	Home	Waktu		Keterangan		Kontroler (reference PWM)
		Memadamkan Api	Balik Home	Balik Home	Memadamkan Api	
Ruang 1	*Non Arbitary Start*	24.24	77.23	Berhasil	Berhasil	P
Ruang 1	*Non Arbitary Start*	23.48	75.58	Berhasil	Berhasil	PI
Ruang 1	*Non Arbitary Start*	22.03	69.09	Berhasil	Berhasil	PID

Tabel 4.3c Pengujian kontroler P, PI dan PID pada robot untuk memadamkan api (*reference* PWM 75 dan *set point* jarak (8cm))

Titik Api	Home	Waktu		Keterangan		Kontroler (reference PWM)
		Memadamkan Api	Balik Home	Balik Home	Memadamkan Api	
Ruang 1	*Non Arbitary Start*	21.57	66.48	Berhasil	Berhasil	P
Ruang 1	*Non Arbitary Start*	21.09	65.08	Berhasil	Berhasil	PI
Ruang 1	*Non Arbitary Start*	20.47	62.25	Berhasil	Berhasil	PID

Tabel 4.3d Pengujian kontroler P, PI dan PID pada robot untuk memadamkan api (*reference* PWM 125 dan *set point* jarak (8cm))

Titik Api	Home	Waktu		Keterangan		Kontroler (reference PWM)
		Memadamkan Api	Balik Home	Balik Home	Memadamkan Api	
Ruang 1	*Non Arbitary Start*	19.57	-	Gagal	Berhasil	P
Ruang 1	*Non Arbitary Start*	19.44	-	Gagal	Berhasil	PI
Ruang 1	*Non Arbitary Start*	18.58	40.05	Berhasil	Berhasil	PID

Pada Tabel 4.3d data yang didapat sedikit berbeda untuk *reference* PWM 125, dapat dilihat pada kontroler P dan PI, robot tidak berhasil balik ke *home* dikarenakan osilasi yang cukup besar untuk setiap tikungan yang tajam (robot menyimpang jauh dari rute dan tidak dapat kembali ke *home*).Akan tetapi penyimpangan jauh tersebut dapat ditanggulangi dengan menambahkan *Derivative*, kontroler ini sangat efektif ditambahkan karena dapat meredam osilasi yang ditimbulkan akibat tikungan tajam pada arena KRCI.

IV.5 Pengujian Kontroler PID

Pada Sub Bab IV.5 dilakukan pengujian kontroler PID pada lapangan KRCI. Pengujian ini lakukan dengan mengubah-ubah posisi *home* dan posisi titik api.

Pada Tabel 4.4a didapatkan dapat yang baik, dan tingkat kegagalan yang kecil untuk peletakan titik api pada ruang 1. Kegagalan terjadi untuk posisi *home* diruang 1, kegagalan tersebut terjadi karena sensor warna tidak bekerja maksimal.

Tabel 4.4a Pengujian kontroler PID pada arena KRCI dengan posisi titik api di ruang 1 untuk *reference* PWM 125

Titik Api	Home	Waktu (s)		Keterangan	
		Memadamkan Api	Balik Home	Balik Home	Memadamkan Api
Ruang 1	*Non Arbitary Start*	19.44	39.11	Berhasil	Berhasil
Ruang 1	*Arbitary Start* (Ruang 2)	33.31	38.26	Berhasil	Berhasil
Ruang 1	*Arbitary Start* (Ruang 3)	19.59	37.42	Berhasil	Berhasil
Ruang 1	*Arbitary Start* (Ruang 4)	14.22	-	Gagal	Berhasil
Ruang 1	*Arbitary Start* (Ruang 3)	26.3	47.33	Berhasil	Berhasil

Tabel 4.4b Pengujian kontroler PID pada arena KRCI dengan posisi titik api di ruang 2 untuk *reference* PWM 125

Titik Api	Home	Waktu (s)		Keterangan	
		Memadamkan Api	Balik Home	Balik Home	Memadamkan Api
Ruang 2	*Non Arbitary Start*	26.53	33.31	Berhasil	Berhasil
Ruang 2	*Arbitary Start* (Ruang 1)	9.33	40.56	Berhasil	Berhasil
Ruang 2	*Arbitary Start* (Ruang 3)	27.21	37.42	Berhasil	Berhasil
Ruang 2	*Arbitary Start* (Ruang 4)	34.42	-	Gagal	Berhasil
Ruang 2	*Arbitary Start* (Ruang 3)	29.3	37.33	Berhasil	Berhasil

Pada Tabel 4.4b didapatkan dapat yang baik, dan tingkat kegagalan yang kecil untuk peletakan titik api pada ruang 2. Kegagalan terjadi untuk posisi *home* diruang 2, kegagalan tersebut terjadi karena sensor warna tidak bekerja maksimal.

Tabel 4.4c Pengujian kontroler PID pada arena KRCI dengan posisi titik api di ruang 3 untuk *reference* PWM 125

Titik Api	Home	Waktu (s)		Keterangan	
		Memadamkan Api	Balik Home	Balik Home	Memadamkan Api
Ruang 3	*Non Arbitary Start*	42.1	50.03	Berhasil	Berhasil
Ruang 3	*Non Arbitary Start*	-	-	Gagal	Gagal
Ruang 3	*Arbitary Start* (Ruang 1)	22.52	42.12	Berhasil	Berhasil
Ruang 3	*Arbitary Start* (Ruang 4)	45.21	-	Gagal	Berhasil
Ruang 3	*Arbitary Start* (Ruang 2)	11.3	37.38	Berhasil	Berhasil

Tabel 4.4d Pengujian kontroler PID pada arena KRCI dengan posisi titik api di ruang 4 untuk *reference* PWM 125

Titik Api	Home	Waktu (s)		Keterangan	
		Memadamkan Api	Balik Home	Balik Home	Memadamkan Api
Ruang 4	*Non Arbitary Start*	42.1	50.03	Berhasil	Berhasil
Ruang 4	*Non Arbitary Start*	-	-	Gagal	Gagal
Ruang 4	*Arbitary Start* (Ruang 1)	22.52	42.12	Berhasil	Berhasil
Ruang 4	*Arbitary Start* (Ruang 4)	45.21	-	Gagal	Berhasil
Ruang 4	*Arbitary Start* (Ruang 2)	11.3	37.38	Berhasil	Berhasil

Pada Tabel 4.4c didapatkan dapat yang baik, dan tingkat kegagalan yang kecil untuk peletakan titik api pada ruang 3. Kegagalan terjadi untuk posisi *home* diruang 2, kegagalan tersebut terjadi karena sensor warna tidak bekerja maksimal. Pada pengujian ini terdapat kegagalan lain yaitu tidak berhasil memadamkan api dan balik ke *home* dikarenakan robot *error* akibat pemasangan kabel pada sensor yang tidak kencang dan mengakibatkan *error* pada kontroler.

Pada Tabel 4.4d juga terdapat kegagalan lain yaitu robot tidak berhasil memadamkan api dan balik ke *home* dikarenakan sensor warna yang kurang optimal dalam mendapatkan range warna.

BAB 5

KESIMPULAN

Pada Bab ini berisi kesimpulan dari Tugas Akhir dan saran-saran yang perlu dilakukan untuk perbaikan di masa mendatang.

V.1 Kesimpulan

Dengan memperhatikan data pengamatan dan analisis pada bab sebelumnya, dapat disimpulkan bahwa:

1. Berdasarkan data pengamatan peranan nilai parameter Kp memiliki peran yang baik terutama untuk memelihara sistem agar selalu dalam keadaan stabil dan berguna untuk mendekatkan keadaan sistem supaya lebih cepat mencapai *setpoint*.

2. Berdasarkan data pengamatan peranan nilai parameter *integral* (Ki) pada *reference* PWM 75 masih cukup berperan walaupun kecil, karena akan berpengaruh pada nilai parameter Kd.

3. Berdasarkan data pengamatan nilai parameter *derivative* (Kd) sangat berperan khususnya untuk kecepatan robot tinggi karena membantu manuver robot saat berada pada tikungan tajam dan meredam osilasi dengan cepat.

4. Modifikasi *trial and error* dan Ziegler Nichols II dapat digunakan untuk kecepatan motor DC rendah (*reference* PWM 25, 50), sedangkan untuk kecepatan motor DC tinggi (75, 125) lebih tepat menggunakan metoda *trial and error* saja.

5. Kegagalan banyak terjadi pada ruang 4, diakibatkan oleh pembacaan sensor warna pada karpet yang tidak akurat sehingga robot tidak dapat kembali ke HOME.

6. Kontroler P dan PI lebih tepat digunakan untuk kecepatan motor rendah (25, 50) karena robot mengalami *loss* saat tikungan tajam. Sedangkan kontroler PID masih sanggup digunakan untuk kecepatan *reference* PWM 125

V.2 Saran

Saran-saran yang dapat diberikan untuk perbaikan dan pengembangan dari Tugas Akhir ini adalah sebagai berikut:

1. Dianjurkan posisi sensor lebih kedepan dari *actuator* agar *error* terbaca lebih dahulu dan respon kontroler tidak terlambat.

2. Desain robot yang cocok untuk kontroler PID adalah *Tricycle Drive* dengan 2 buah roda kiri dan kanan sebagai kemudinya di bagian belakang dan satu roda statis dibagian depan karena membuat respon kontroler lebih cepat.

3. Sensor *ultrasonic* yang digunakan hendaknya memiliki kecepatan membaca jarak dengan cepat untuk menanggulangi perubahan *error* yang extreme.

4. Untuk kecepatan tinggi *deadzone* diperlukan agar robot dapat berputar dengan baik saat terdapat tikungan.

5. *Reference* PWM pada yang disarankan pada robot sebaik tidak terlalu besar karena *reference* PWM yang terlalu besar dapat menyebabkan sinyal kontrol yang dihasilkan akan saturasi karena *reference* PWM dibatasi hanya 255.

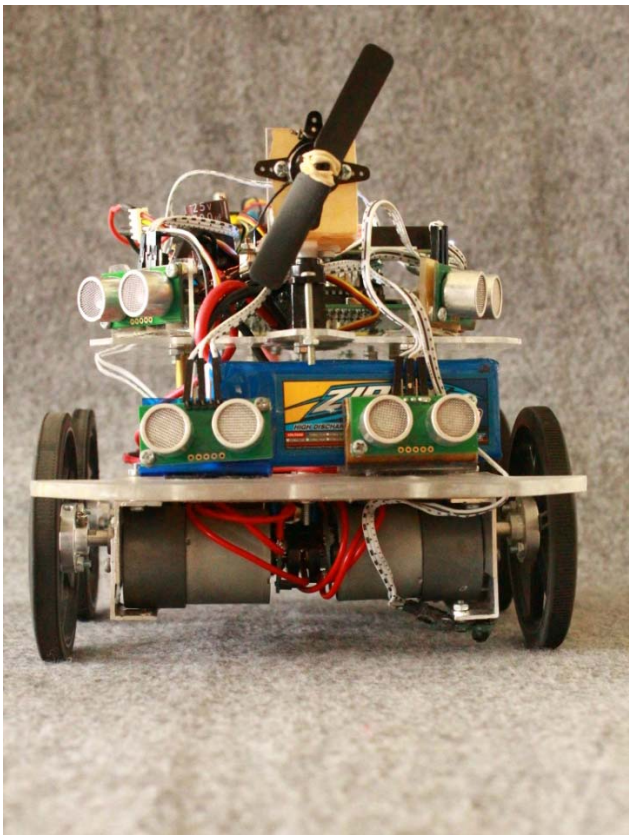
DAFTAR PUSTAKA

1. Ogata, K., *Buku Modern Control Engineering, Third Edition*, Jakarta : Gramedia, 1996.
2. Goris, Kristof. 2004. Autonomous Mobile Robot Mechanical Design. *Thesis*. Vrije Universiteit Brussel.
3. Andrianto, H., *Buku Panduan : Pelatihan Mikrokontroler AVR ATmega16*, 2008.
4. http://www.elektroindonesia.com/elektro/tutor12.html (12 Desember 2011)
5. http://k12008.widyagama.ac.id/ske/diktatpdf/BabVIII_Pengendalian_Proporsional_PID.pdf (13 Desember 2011)
6. http://en.wikipedia.org/wiki/PID_controller#PID_tuning_software (1 Januari 2012)
7. http://fahmizaleeits.wordpress.com/2011/02/25/definisi-kontroler/ (3 Januari 2012)
8. http://menanamilmu.blogspot.com/2010/09/teori-kontrol-pid-proportionalintegrald.html (3 Januari 2012)
9. http://www.scribd.com/doc/36370655/Slide-Kontroler-PID-3 (4 Januari 2012)
10. http://www.reocities.com/al_dodi/kerja/kp3.pdf (15 Februari 2012)
11. http://www.elektro.undip.ac.id/el_kpta/upload/L2F097650_MTA.pdf (15 Februari 2012)
12. http://viri-electronics.blogspot.com/2010/07/modul-proportional-integrator.html (15 Februari 2012)
13. http://ebookbrowse.com/gdoc.php?id=304307628&url=af6806c832a7b06e218be9ebc7c1b40c (15 Februari 2012)
14. http://repo.eepis-its.edu/1430/1/[A-D206-10]_pp.49-55_Pengaturan_Gerak_Dan_Keseimbangan_Robot_Line_Tracer_Dua_Roda.pdf (16 Februari 2012)

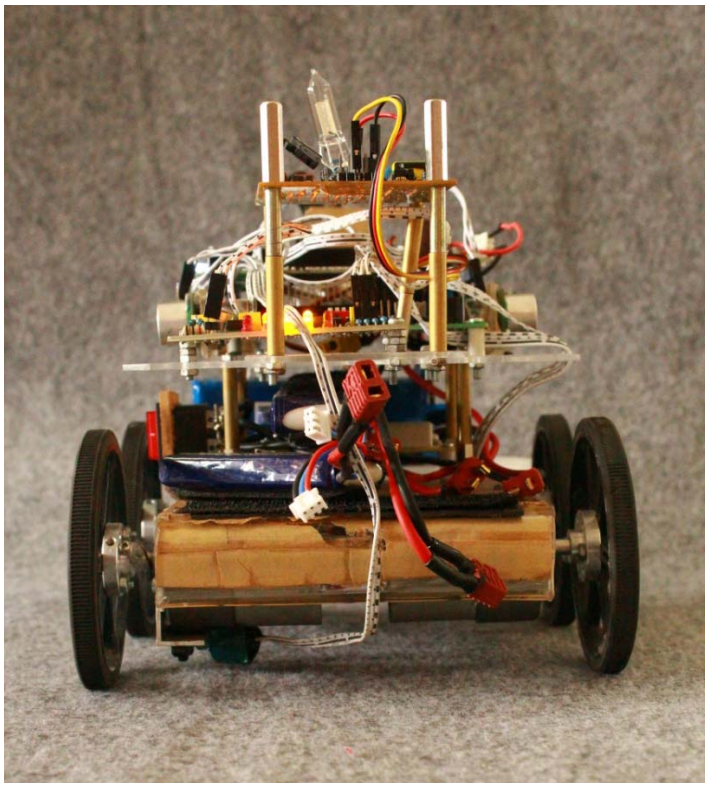
15. http://digilib.its.ac.id/public/ITS-Undergraduate-16550-Paper-pdf.pdf (19 Februari 2012)
16. http://www.atmel.com/dyn/resources/prod_documents/doc2467.pdf (19 Februari 2012)
17. http://www.robot-electronics.co.uk/htm/srf05tech.html (24 Februari 2012)
18. http://www.robotstorehk.com/R2868.pdf (24 Februari 2012)
19. http://www.robokits.co.nz/ZX-03_IR_Sensor_Resource_Page (27 Februari 2012)
20. http://www.pololu.com/catalog/product/705 (17 Maret 2012)
21. http://124it.blogspot.com/2010/11/sistem-kontrol-pid.html (18 Maret 2012)
22. http://www.google.co.id/url?sa=t&rct=j&q=ziegler%20nichols%20ii%20adalah&source=web&cd=2&ved=0CFUQFjAB&url=http://www.freewebs.com/kapeha/dsp.doc&ei=0vPCT97hJIHRrQfYt-TUCQ&usg=AFQjCNH_EYX4IzUHDYN8_YlRyx2SMr3ZyA (1 April 2012)
23. http://digilib.ittelkom.ac.id/index.php?option=com_content&view=article&id=820:pwm-pulse-width-modulation&catid=15:pemrosesan-sinyal&Itemid=14 (1 April 2012)

LAMPIRAN A
FOTO ROBOT BERODA

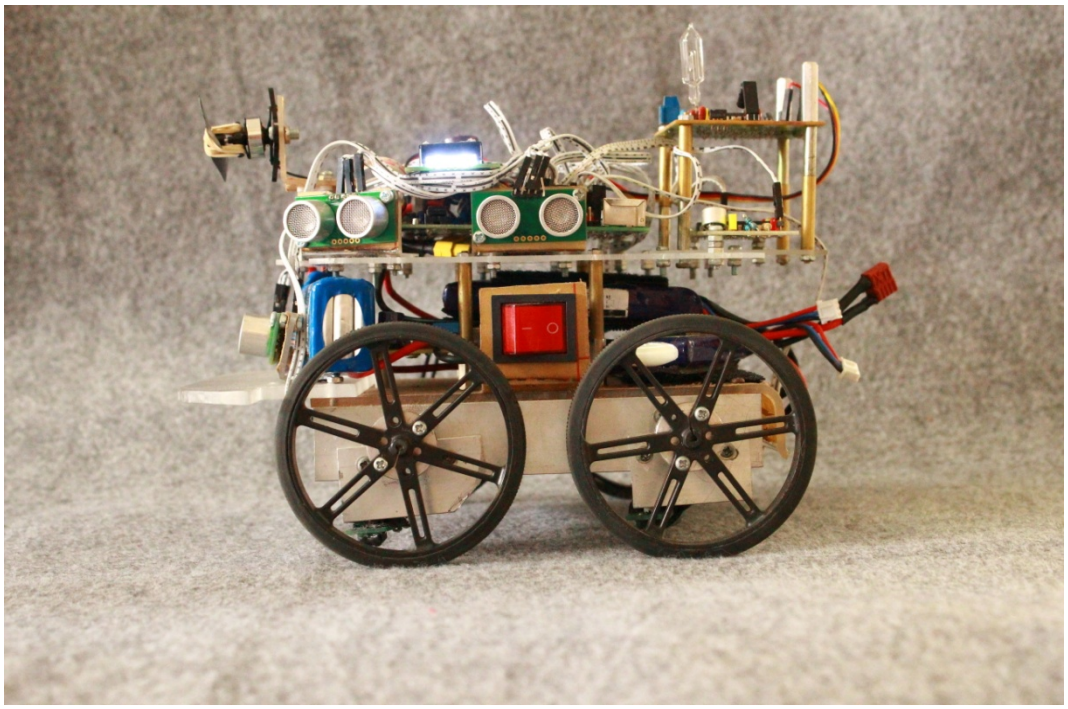
TAMPAK DEPAN



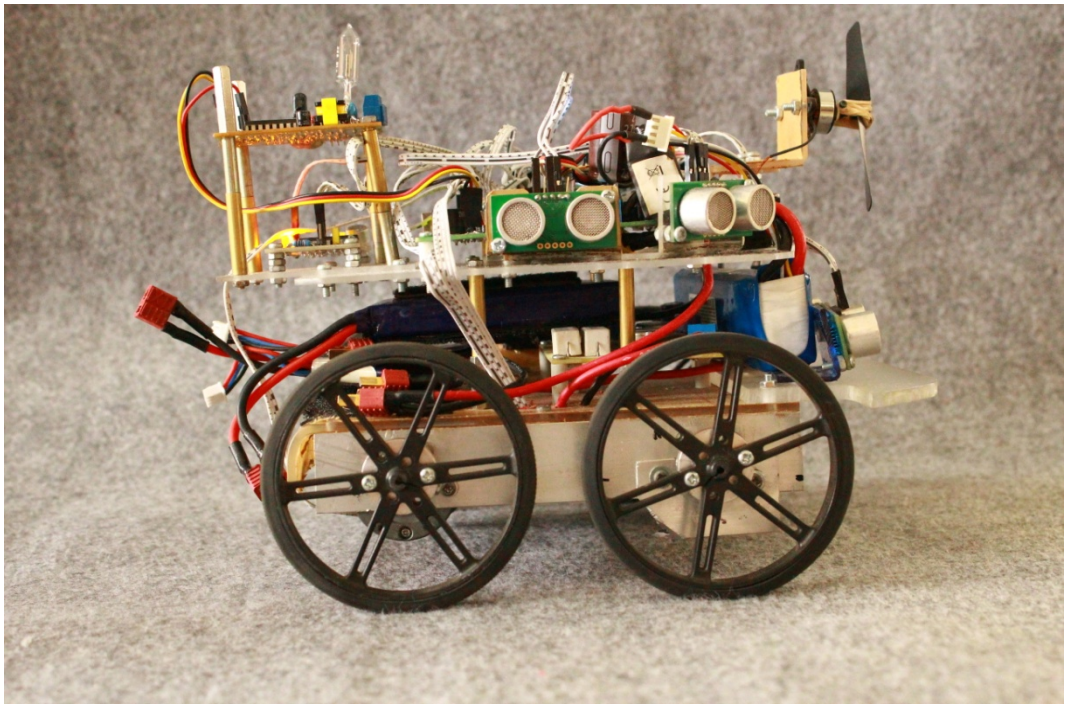
TAMPAK BELAKANG



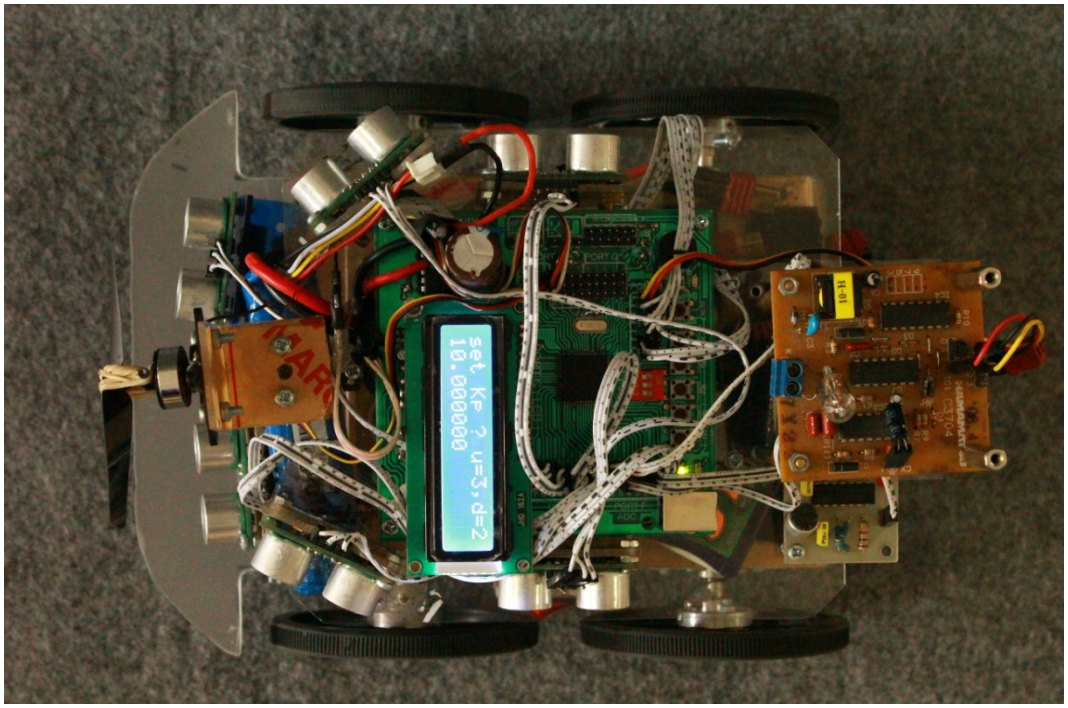
TAMPAK SAMPING KIRI



TAMPAK SAMPING KANAN



TAMPAK ATAS



LAMPIRAN B

PROGRAM PID PADA ROBOT BERODA

PROGRAM UTAMA

```
/*****************************************************
This program was produced by the
CodeWizardAVR V1.25.3 Standard
Automatic Program Generator
© Copyright 1998-2007 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com

Project :
Version :
Date    : 1/5/2012
Author  : ROCKY ANTHONI
Company :
Comments:


Chip type           : ATmega128
Program type        : Application
Clock frequency     : 11.059200 MHz
Memory model        : Small
External SRAM size  : 0
Data Stack size     : 1024
*****************************************************/

#include <mega128.h>
#include <delay.h>
#include <stdio.h>
#include <math.h>



#define RXB8 1
#define TXB8 0
#define UPE 2
#define OVR 3
#define FE 4
#define UDRE 5
#define RXC 7

#define FRAMING_ERROR (1<<FE)
#define PARITY_ERROR (1<<UPE)
#define DATA_OVERRUN (1<<OVR)
#define DATA_REGISTER_EMPTY (1<<UDRE)
#define RX_COMPLETE (1<<RXC)

// USART0 Transmitter buffer
#define TX_BUFFER_SIZE0 8
char tx_buffer0[TX_BUFFER_SIZE0];

#if TX_BUFFER_SIZE0<256
unsigned char tx_wr_index0,tx_rd_index0,tx_counter0;
#else
unsigned int tx_wr_index0,tx_rd_index0,tx_counter0;
#endif

// USART0 Transmitter interrupt service routine
interrupt [USART0_TXC] void usart0_tx_isr(void)
{
if (tx_counter0)
   {
   --tx_counter0;
```

```
  UDR0=tx_buffer0[tx_rd_index0];
  if (++tx_rd_index0 == TX_BUFFER_SIZE0) tx_rd_index0=0;
  };
}

#ifndef _DEBUG_TERMINAL_IO_
// Write a character to the USART0 Transmitter buffer
#define _ALTERNATE_PUTCHAR_
#pragma used+
void putchar(char c)
{
while (tx_counter0 == TX_BUFFER_SIZE0);
#asm("cli")
if (tx_counter0 || ((UCSR0A & DATA_REGISTER_EMPTY)==0))
  {
  tx_buffer0[tx_wr_index0]=c;
  if (++tx_wr_index0 == TX_BUFFER_SIZE0) tx_wr_index0=0;
  ++tx_counter0;
  }
else
  UDR0=c;
#asm("sei")
}
#pragma used-
#endif


// #define RXB8 1
// #define TXB8 0
// #define UPE 2
// #define OVR 3
// #define FE 4
// #define UDRE 5
// #define RXC 7
//
// #define FRAMING_ERROR (1<<FE)
// #define PARITY_ERROR (1<<UPE)
// #define DATA_OVERRUN (1<<OVR)
// #define DATA_REGISTER_EMPTY (1<<UDRE)
// #define RX_COMPLETE (1<<RXC)
//
// // USART0 Transmitter buffer
// #define TX_BUFFER_SIZE0 8
// char tx_buffer0[TX_BUFFER_SIZE0];
//
// #if TX_BUFFER_SIZE0<256
// unsigned char tx_wr_index0,tx_rd_index0,tx_counter0;
// #else
// unsigned int tx_wr_index0,tx_rd_index0,tx_counter0;
// #endif
//
// // USART0 Transmitter interrupt service routine
// interrupt [USART0_TXC] void usart0_tx_isr(void)
// {
// if (tx_counter0)
//    {
//    --tx_counter0;
//    UDR0=tx_buffer0[tx_rd_index0];
//    if (++tx_rd_index0 == TX_BUFFER_SIZE0) tx_rd_index0=0;
//    };
// }
//
// #ifndef _DEBUG_TERMINAL_IO_
// // Write a character to the USART0 Transmitter buffer
```

```
// #define _ALTERNATE_PUTCHAR_
// #pragma used+
// void putchar(char c)
// {
// while (tx_counter0 == TX_BUFFER_SIZE0);
// #asm("cli")
// if (tx_counter0 || ((UCSR0A & DATA_REGISTER_EMPTY)==0))
//    {
//    tx_buffer0[tx_wr_index0]=c;
//    if (++tx_wr_index0 == TX_BUFFER_SIZE0) tx_wr_index0=0;
//    ++tx_counter0;
//    }
// else
//    UDR0=c;
// #asm("sei")
// }
// #pragma used-
// #endif


// I2C Bus functions
#asm
   .equ __i2c_port=0x12 ;PORTD
   .equ __sda_bit=1
   .equ __scl_bit=0
#endasm
#include <i2c.h>

// Alphanumeric LCD Module functions
#asm
   .equ __lcd_port=0x15 ;PORTC
#endasm
#include <lcd.h>

#define RXB8 1
#define TXB8 0
#define UPE 2
#define OVR 3
#define FE 4
#define UDRE 5
#define RXC 7

#define FRAMING_ERROR (1<<FE)
#define PARITY_ERROR (1<<UPE)
#define DATA_OVERRUN (1<<OVR)
#define DATA_REGISTER_EMPTY (1<<UDRE)
#define RX_COMPLETE (1<<RXC)



#define ADC_VREF_TYPE 0x00

// Read the AD conversion result
unsigned int read_adc(unsigned char adc_input)
{
ADMUX=adc_input | (ADC_VREF_TYPE & 0xff);
// Start the AD conversion
ADCSRA|=0x40;
// Wait for the AD conversion to complete
while ((ADCSRA & 0x10)==0);
ADCSRA|=0x10;
return ADCW;
}
```

```
// Get a character from the USART1 Receiver
#pragma used+
char getchar1(void)
{
char status,data;
while (1)
      {
      while (((status=UCSR1A) & RX_COMPLETE)==0);
      data=UDR1;
      if ((status & (FRAMING_ERROR | PARITY_ERROR | DATA_OVERRUN))==0)
        return data;
      };
}
#pragma used-

// Write a character to the USART1 Transmitter
#pragma used+
void putchar1(char c)
{
while ((UCSR1A & DATA_REGISTER_EMPTY)==0);
UDR1=c;
}
#pragma used-

// Declare your global variables here



unsigned char matlab,x,y;
unsigned char text[32];
unsigned char varKipas,varUlang,varServo1,varServo2,varServo3,a,detect,apiPadam;
unsigned char counter,findHome,apiDead,i,juringTengah, juringPinggir, varJuring;
unsigned char Ts,menu,setOK,setKp,setKi,setKd;
eeprom float kp,kd,ki;
float proporsional,dataLeftPWM[5],dataRightPWM[5],dataSinyalControl[5];
float integral,derivative,processVariable,error,setPointSensorKiri,setPointPWM,rate,rateInt,lastError;
float
sinyalControl,leftPWM,prosesLeftPWM,rightPWM,dataTerakhir,prosesSinyalControl,prosesRightPWM;
unsigned int nilaiSrf08Kiri, range2;
unsigned int srfKanan, srfKiri;
unsigned char cariRuang4, counterRuang4, varRuang4, varHomeRuang4, nilaiCounterRuang4,varLoop;
unsigned char  count, count2, count3, counterLagiCui, counterCariHomeDariRuang4;
unsigned char posisi, varPosisiDepan, varPosisiKanan, varPos;

void soundAktive()
{
ulang:
if(PINA.3 == 0){
      goto lanjut;
      }
if(PINF.5==1 & PINF.6==0 & PINF.7==0){
if(PINA.3 == 0){
      goto lanjut;
      }
            while(PINF.5==1 & PINF.6==0 & PINF.7==0){delay_us(2);
if(PINA.3 == 0){
      goto lanjut;
      }}
}
else{

if(PINA.3 == 0){
      goto lanjut;
      }goto ulang;
```

```
}
if(PINF.5==0 & PINF.6==1 & PINF.7==0){
if(PINA.3 == 0){
	goto lanjut;
	}
		while(PINF.5==0 & PINF.6==1 & PINF.7==0) {
if(PINA.3 == 0){
	goto lanjut;
	}delay_us(2);}
if(PINF.5==1 & PINF.6==1 & PINF.7==0){
if(PINA.3 == 0){
	goto lanjut;
	}
	while(PINF.5==1 & PINF.6==1 & PINF.7==0) {
if(PINA.3 == 0){
	goto lanjut;
	}delay_us(2);}
//		if(PINF.5==0 & PINF.6==0 & PINF.7==1)
//		{
//		while(PINF.5==0 & PINF.6==0 & PINF.7==1) {delay_us(2);}

if(PINA.3 == 0){
	goto lanjut;
	}goto lanjut;
//		}
//		else
//		{
//		goto ulang;
//		}
	}
	else
	{

if(PINA.3 == 0){
	goto lanjut;
	}goto ulang;
	}
}
else
{

if(PINA.3 == 0){
	goto lanjut;
	}goto ulang;
}
lanjut:
}

void startranging(unsigned char address)
{
i2c_start();
i2c_write(address);
i2c_write(0x00);
i2c_write(0x51);
i2c_stop();
delay_ms(70);
}

void setAddress(unsigned char old, unsigned char new)
{
i2c_start();
i2c_write(old);
i2c_write(0x00);
i2c_write(0xA0);
```

```
i2c_stop();
delay_ms(70);
i2c_start();
i2c_write(old);
i2c_write(0x00);
i2c_write(0xAA);
i2c_stop();
delay_ms(70);
i2c_start();
i2c_write(old);
i2c_write(0x00);
i2c_write(0xA5);
i2c_stop();
delay_ms(70);
i2c_start();
i2c_write(old);
i2c_write(0x00);
i2c_write(new);
i2c_stop();
delay_ms(70);
}

unsigned int getRange(unsigned char address)
{
unsigned int data, data1, data2;

i2c_start();
i2c_write(address);
i2c_write(0x02);
i2c_start();
i2c_write(address|1);
data1 = i2c_read(0);
i2c_stop();

data1 = data1 << 8;

i2c_start();
i2c_write(address);
i2c_write(0x03);
i2c_start();
i2c_write(address|1);
data2 = i2c_read(0);
i2c_stop();

data = data1+data2;

return data;
}

unsigned int getSrf(unsigned char i)
{
      int jarak = 0, bacaJarak;
      switch(i)
      {
            case 0 :
                  {
                        PORTA.0 = 0;
                        DDRA.0 = 1;
                        PORTA.0 = 1;
                        delay_us(15);
                        PORTA.0 = 0;
                        DDRA.0 = 0;
                        delay_us(750);
                        while(PINA.0 == 0){delay_us(1);};
```

```
            while(PINA.0 == 1)
            {
                jarak++;
                delay_us(1);
                if(jarak>25000)   break;
            }
            delay_us(5);
        }
        break;
case 1 :
        {
            PORTA.1 = 0;
            DDRA.1 = 1;
            PORTA.1 = 1;
            delay_us(15);
            PORTA.1 = 0;
            DDRA.1 = 0;
            delay_us(750);
            while(PINA.1 == 0){delay_us(1);};
            while(PINA.1 == 1)
            {
                jarak++;
                delay_us(1);
                if(jarak>25000)  break;
            }
            delay_us(5);
        }
        break;
case 2 :
        {
            PORTA.2 = 0;
            DDRA.2 = 1;
            PORTA.2 = 1;
            delay_us(15);
            PORTA.2 = 0;
            DDRA.2 = 0;
            delay_us(750);
            while(PINA.2 == 0){delay_us(1);};
            while(PINA.2 == 1)
            {
                jarak++;
                delay_us(1);
                if(jarak>25000)  break;
            }
            delay_us(5);
        }
        break;
case 3 :
        {
            PORTB.0 = 0;
            DDRB.0 = 1;
            PORTB.0 = 1;
            delay_us(15);
            PORTB.0 = 0;
            DDRB.0 = 0;
            delay_us(750);
            while(PINB.0 == 0){delay_us(1);};
            while(PINB.0 == 1)
            {
                jarak++;
                delay_us(1);
                if(jarak>25000)  break;
            }
            delay_us(5);
```

```
        }
        break;
case 4 :
        {
                PORTA.4 = 0;
                DDRA.4 = 1;
                PORTA.4 = 1;
                delay_us(15);
                PORTA.4 = 0;
                DDRA.4 = 0;
                delay_us(750);
                while(PINA.4 == 0){delay_us(1);};
                while(PINA.4 == 1)
                {
                        jarak++;
                        delay_us(1);
                        if(jarak>25000)  break;
                }
                delay_us(5);
        }
        break;
case 5 :
        {
                PORTA.5 = 0;
                DDRA.5 = 1;
                PORTA.5 = 1;
                delay_us(15);
                PORTA.5 = 0;
                DDRA.5 = 0;
                delay_us(750);
                while(PINA.5 == 0){delay_us(1);};
                while(PINA.5 == 1)
                {
                        jarak++;
                        delay_us(1);
                        if(jarak>25000)  break;
                }
                delay_us(5);
        }
        break;
case 6 :
        {
                PORTA.6 = 0;
                DDRA.6 = 1;
                PORTA.6 = 1;
                delay_us(15);
                PORTA.6 = 0;
                DDRA.6 = 0;
                delay_us(750);
                while(PINA.6 == 0){delay_us(1);};
                while(PINA.6 == 1)
                {
                        jarak++;
                        delay_us(1);
                        if(jarak>25000)  break;
                }
                delay_us(5);
        }
        break;
case 7 :
        {
                PORTA.7 = 0;
                DDRA.7 = 1;
                PORTA.7 = 1;
```

```
                        delay_us(15);
                        PORTA.7 = 0;
                        DDRA.7 = 0;
                        delay_us(750);
                        while(PINA.7 == 0){delay_us(1);};
                        while(PINA.7 == 1)
                        {
                            jarak++;
                            delay_us(1);
                            if(jarak>25000)  break;
                        }
                        delay_us(5);
                    }
                    break;
                default :
                    {
                        return 0;
                    }
        }
        bacaJarak = jarak/29.034;
        return bacaJarak;
}

void maju(char speedA, char speedB)
{
 OCR1A = speedA;
 OCR1B = speedB;
 PORTB.1 = 1;
 PORTB.2 = 0;
 PORTB.3 = 1;
 PORTB.4 = 0;
}
void berenti(char speedA, char speedB)
{
 OCR1A = speedA;
 OCR1B = speedB;
 PORTB.1 = 1;
 PORTB.2 = 1;
 PORTB.3 = 1;
 PORTB.4 = 1;
}

void belokKiri(char speedA, char speedB)      // ban kanan maju, kiri diem
{
 OCR1A = speedA;
 OCR1B = speedB;
 PORTB.1 = 1;
 PORTB.2 = 1;
 PORTB.3 = 1;
 PORTB.4 = 0;
}
void belokKanan(char speedA, char speedB)      // ban kiri maju, kanan diem
{
 OCR1A = speedA;
 OCR1B = speedB;
 PORTB.1 = 1;
 PORTB.2 = 0;
 PORTB.3 = 1;
 PORTB.4 = 1;
}
void kiri(char speedA, char speedB)          // ban kiri mundur, kanan diem
{
 OCR1A = speedA;
 OCR1B = speedB;
```

```
 PORTB.1 = 0;
 PORTB.2 = 1;
 PORTB.3 = 1;
 PORTB.4 = 1;
}
void kanan(char speedA, char speedB)         //ban kanan mundur, kiri diem
{
 OCR1A = speedA;
 OCR1B = speedB;
 PORTB.1 = 1;
 PORTB.2 = 1;
 PORTB.3 = 0;
 PORTB.4 = 1;
}
void bantingKiri(char speedA, char speedB)  //ban kiri mundur, kanan maju
{
 OCR1A = speedA;
 OCR1B = speedB;
 PORTB.1 = 0;
 PORTB.2 = 1;
 PORTB.3 = 1;
 PORTB.4 = 0;
}
void bantingKanan(char speedA, char speedB) //ban kanan mundur, kiri maju
{
 OCR1A = speedA;
 OCR1B = speedB;
 PORTB.1 = 1;
 PORTB.2 = 0;
 PORTB.3 = 0;
 PORTB.4 = 1;
}
void kipas(void){

      for( varServo1=0;varServo1<=10;varServo1++ ){
          PORTD.6=1;
          delay_us(1500);
          PORTD.6=0;
          delay_us(18500);
      }
      for( varKipas=0;varKipas<=50;varKipas++ ){
          PORTD.5=1;
          delay_us(1500);
          PORTD.5=0;
          delay_us(18500);
          }
      for( varServo1=0;varServo1<=10;varServo1++ ){
          PORTD.6=1;
          delay_us(1200);
          PORTD.6=0;
          delay_us(18800);
      }
      for( varKipas=0;varKipas<=50;varKipas++ ){
          PORTD.5=1;
          delay_us(2000);
          PORTD.5=0;
          delay_us(18000);
          }
      for( varServo1=0;varServo1<=10;varServo1++ ){
          PORTD.6=1;
          delay_us(1800);
          PORTD.6=0;
          delay_us(18200);
      }
```

```
        for( varKipas=0;varKipas<=50;varKipas++ ){
            PORTD.5=1;
            delay_us(2000);
            PORTD.5=0;
            delay_us(18000);
            }
        for( varServo1=0;varServo1<=10;varServo1++ ){
            PORTD.6=1;
            delay_us(1500);
            PORTD.6=0;
            delay_us(18500);
        }
        for( varServo1=0;varServo1<=10;varServo1++ ){
            PORTD.6=1;
            delay_us(1500);
            PORTD.6=0;
            delay_us(18500);
        }

}

unsigned int srf08Kiri() {
        startranging(0xE0);
        nilaiSrf08Kiri = getRange(0xE0);
        return nilaiSrf08Kiri;
        }

void countingHome(void){
        if(read_adc(0) >= 410){
                count2=1;
                }
        if(count2 == 1 && read_adc(0)<410 && apiPadam == 1){
                counterCariHomeDariRuang4 = counterCariHomeDariRuang4 + 1;
                count2 =0;
                }
 }
void counting(void){
        if(read_adc(0) >= 410){
                count=1;
                }
        if(count == 1 && read_adc(0)<410 && apiPadam == 0){
                counterRuang4 = counterRuang4 + 1;
                count =0;
                }
 }
void countingLagi(void){
        if(read_adc(0) >= 410){
                count3=1;
                }
        if(count3 == 1 && read_adc(0)<410 && apiPadam == 0){
                counterLagiCui = counterLagiCui + 1;
                count3 =0;
                }
 }

 void cariPosisiAwal(){
         while(1){
                cariPosisi:
                varPosisiDepan=getSrf(6);
                varPosisiKanan=getSrf(4);
                if(varPosisiDepan > 14){
                        if(varPosisiKanan > 8 && getSrf(1) > 10){
                                bantingKanan(120,120);
                                delay_ms(75);
```

```
                varPos=varPos+1;
                if(varPos>=19){
                    break;
                    }
                goto cariPosisi;
                }
            else{
                varPos=19;
                break;
                }
            goto cariPosisi;
            }

        else if( varPosisiDepan <=14){
            if(varPosisiKanan > 8 && getSrf(1) > 10){
                bantingKiri(120,120);
                delay_ms(75);
                varPos=varPos+1;
                if(varPos>=19){
                    break;
                    }
                goto cariPosisi;
                }
            else{
                varPos=19;
                break;
                }
            goto cariPosisi;
            }
        berenti(255,255);
        PORTD.4=1;
        delay_ms(20);
        PORTD.4=0;
        delay_ms(20);
        PORTD.4=1;
        delay_ms(20);
        PORTD.4=0;
        delay_ms(20);
        PORTD.4=1;
        delay_ms(20);
        PORTD.4=0;
        delay_ms(20);
        PORTD.4=1;
        delay_ms(20);
        PORTD.4=0;
        delay_ms(20);
        }
}


void wallKananMulus(void){

//=============================================================================
    if(PINB.7 == 1){                                    //   lagi gak ada api
            juringTengah=0;
            juringPinggir=0;                            //  reset juring tengah, klo juring tgh =1 muter
kiri trs

            if( getSrf(6) > 14) {                               //14

                kp = 10;                                        //10
                kd = 15;                                        //15
                ki = 0.005;                                     // 0.025
                setPointPWM =110;
```

```
                    processVariable = getSrf(1);                        // proporsional program
                    setPointSensorKiri = 8;
                    error = setPointSensorKiri - processVariable;
                    proporsional = kp * error;                          //proporsional end
                    rate = error - lastError;                           //derivative program
                    rateInt = error + lastError;
                    Ts=1;                                               //Ts = time sampling
                    integral = Ts * (ki * rateInt);
                    derivative = (kd/Ts) * rate;                        //derivative
                    sinyalControl = proporsional + derivative + integral;       //sinyal control
                    rightPWM = setPointPWM + sinyalControl;
                    leftPWM = setPointPWM - sinyalControl;
                    if(rightPWM >= 255) {
                          rightPWM = 130;
                          }
                    if(leftPWM >= 255) {
                          leftPWM = 130;
                          }
                    if(rightPWM <= 0) {
                          rightPWM = 0;
                          }
                    if(leftPWM <= 0) {
                          leftPWM = 0;
                          }
                    leftPWM;
                    rightPWM;
                    if(leftPWM == 0 && rightPWM == 200){
                           bantingKiri(leftPWM,rightPWM);
                           lcd_clear();
                           lcd_gotoxy(0,0);
                           lcd_putsf("belok kiri");
                          }
                    if(rightPWM == 0 && leftPWM == 200){
                           bantingKanan(leftPWM,rightPWM);
                           lcd_clear();
                           lcd_gotoxy(0,0);
                           lcd_putsf("belok kanan");
                          }
                    if( (rightPWM <200 || leftPWM < 200)  && (rightPWM > 0 || leftPWM > 0) ){

                           if( (leftPWM < rightPWM) && (leftPWM < 40)){
                                belokKiri(leftPWM,rightPWM);
                                }
                           if( (leftPWM > rightPWM) && (rightPWM < 40) ){
                                belokKanan(leftPWM,rightPWM);
                                }
                           else{
                                maju(leftPWM,rightPWM);
                               }
                           lcd_clear();
                           lcd_gotoxy(0,0);
                           sprintf(text,"S.K=%3d ",processVariable);
                           lcd_puts(text);
                           lcd_gotoxy(0,1);
                           sprintf(text,"r=%3d l=%3d",rightPWM,leftPWM);
                           lcd_puts(text);
                           }//end  if( (rightPWM<170||leftPWM<170)&& (rightPWM>0||leftPWM>0) ){

//************************************COUNTER  ****************************************
                    counting();
                    if( counterRuang4 >= 4 && apiPadam == 0 && varRuang4 == 0 ){
                          if( (read_adc(0) > 140 && read_adc(0) < 300) &&  (read_adc(1) > 70 &&
read_adc(1) < 300) ){          // CEK WARNA ABU
                                srfKiri=getSrf(2);
```

```
                srfKanan=getSrf(4);
                if( srfKiri < 20 && srfKiri > 1 && srfKanan < 10 && srfKanan > 1){

                        PORTD.4 = 1;
                        varRuang4=1;
                        lcd_clear();
                        lcd_gotoxy(0,1);
                        sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                        lcd_puts(text);
                        delay_ms(50);
                        }
                else {
                    lcd_clear();
                    lcd_gotoxy(0,1);
                    sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                    lcd_puts(text);
                    PORTD.4 = 0;
                    goto start;
                    }
                }
            else {
                goto start;
                }
            }
        else {
            goto start;
            }
        if ( counterRuang4 >= 4 && apiPadam == 0 && varRuang4 == 1){

            if( (read_adc(0) > 140 && read_adc(0) < 300) &&  (read_adc(1) > 70 &&
read_adc(1) < 300) ){

                srfKiri=getSrf(2);
                srfKanan=getSrf(4);
                if( srfKiri < 20 && srfKiri > 1 && srfKanan < 10 && srfKanan > 1){
                        PORTD.4 = 0;
                        varRuang4=2;
                        lcd_clear();
                        lcd_gotoxy(0,1);
                        sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                        lcd_puts(text);
                        delay_ms(50);
                        }
                else {
                        lcd_clear();
                        lcd_gotoxy(0,1);
                        sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                        lcd_puts(text);
                        PORTD.4 = 0;
                        varRuang4=0;
                        goto start;
                        }
                }
            else  {
                    lcd_clear();
                    lcd_gotoxy(0,1);
                    sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                    lcd_puts(text);
                    PORTD.4 = 0;
                    varRuang4=0;
                    goto start;
                    }
                }
        else {
```

```
                varRuang4=0;
                goto start;
                }
            if ( counterRuang4 >= 4 && apiPadam == 0 && varRuang4 == 2){

                if( (read_adc(0) > 140 && read_adc(0) < 300) &&  (read_adc(1) > 70 &&
read_adc(1) < 300) ){

                    srfKiri=getSrf(2);
                    srfKanan=getSrf(4);
                    if( srfKiri < 20 && srfKiri > 1 && srfKanan < 10 && srfKanan > 1){
                            PORTD.4 = 0;
                            varRuang4=3;
                            lcd_clear();
                            lcd_gotoxy(0,1);
                            sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                            lcd_puts(text);
                            delay_ms(40);
                            }
                    else {
                        lcd_clear();
                        lcd_gotoxy(0,1);
                        sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                        lcd_puts(text);
                        PORTD.4 = 0;
                        varRuang4=0;
                        goto start;
                        }
                    }
                else  {
                    lcd_clear();
                    lcd_gotoxy(0,1);
                    sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                    lcd_puts(text);
                    PORTD.4 = 0;
                    varRuang4=0;
                    goto start;
                   }
                }
            else {
                varRuang4=0;
                goto start;
                }
            if ( counterRuang4 >= 4 && apiPadam == 0 && varRuang4 == 3){

                if( (read_adc(0) > 140 && read_adc(0) < 300) &&  (read_adc(1) > 70 &&
read_adc(1) < 300) ){

                    srfKiri=getSrf(2);
                    srfKanan=getSrf(4);
                    if( srfKiri < 20 && srfKiri > 1 && srfKanan < 10 && srfKanan > 1){
                            PORTD.4 = 0;
                            varRuang4=4;
                            lcd_clear();
                            lcd_gotoxy(0,1);
                            sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                            lcd_puts(text);
                            delay_ms(30);
                            }
                    else {
                        lcd_clear();
                        lcd_gotoxy(0,1);
                        sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                        lcd_puts(text);
```

```
                                        PORTD.4 = 0;
                                        varRuang4=0;
                                        goto start;
                                        }
                                }
                            else   {
                                        lcd_clear();
                                        lcd_gotoxy(0,1);
                                        sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                        lcd_puts(text);
                                        PORTD.4 = 0;
                                        varRuang4=0;
                                        goto start;
                                       }
                            }
                        else {
                            varRuang4=0;
                            goto start;
                            }
                        if ( counterRuang4 >= 4 && apiPadam == 0 && varRuang4 == 4){

                             if( (read_adc(0) > 140 && read_adc(0) < 300) &&  (read_adc(1) > 70 &&
read_adc(1) < 300) ){
                                srfKiri=getSrf(2);
                                srfKanan=getSrf(4);
                                if( srfKiri < 20 && srfKiri > 1 && srfKanan < 10 && srfKanan > 1){
                                            PORTD.4 = 0;
                                            varRuang4=5;
                                            lcd_clear();
                                            lcd_gotoxy(0,1);
                                            sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                            lcd_puts(text);
                                            delay_ms(15);
                                            }
                                else {
                                            lcd_clear();
                                            lcd_gotoxy(0,1);
                                            sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                            lcd_puts(text);
                                            PORTD.4 = 0;
                                            varRuang4=0;
                                            goto start;
                                            }
                                }
                            else   {
                                        lcd_clear();
                                        lcd_gotoxy(0,1);
                                        sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                        lcd_puts(text);
                                        PORTD.4 = 0;
                                        varRuang4=0;
                                        goto start;
                                       }
                            }
                        else {
                            varRuang4=0;
                            goto start;
                            }
                        if ( counterRuang4 >= 4 && apiPadam == 0 && varRuang4 == 5){

                             if( (read_adc(0) > 140 && read_adc(0) < 300) &&  (read_adc(1) > 70 &&
read_adc(1) < 300) ){
                                srfKiri=getSrf(2);
                                srfKanan=getSrf(4);
```

```
                                    if( srfKiri < 20 && srfKiri > 1 && srfKanan < 10 && srfKanan > 1){
                                                PORTD.4 = 0;
                                                varRuang4=6;
                                                //wallKiri();
                                                lcd_clear();
                                                lcd_gotoxy(0,1);
                                                sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                                lcd_puts(text);
                                                }
                                    else {
                                                lcd_clear();
                                                lcd_gotoxy(0,1);
                                                sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                                lcd_puts(text);
                                                PORTD.4 = 0;
                                                varRuang4=0;
                                                goto start;
                                                }
                                    }
                                else  {
                                            lcd_clear();
                                            lcd_gotoxy(0,1);
                                            sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                            lcd_puts(text);
                                            PORTD.4 = 0;
                                            varRuang4=0;
                                            goto start;
                                           }
                                }
                        else {
                             varRuang4=0;
                             goto start;
                             }

//*************************************END COUNTER  ***************************
start:
                        if( apiPadam == 1 && read_adc(0) < 345 && getSrf(1) > 12){  // CEK WARNA !!   abu
terbesar
                             findHome = 1;
                             apiPadam = 0;
                             delay_ms(200);
                             }
                        if( findHome == 1 && read_adc(0) < 130){                        // CEK WARNA !!
item
                             findHome = 2;
                             apiPadam = 0;
                             delay_ms(200);
                             }
                        if(varJuring == 2 && read_adc(0) > 140 && read_adc(0) < 300 && findHome == 2){
// CEK WARNA !!    abu terkecil dan terbesar

                             findHome = 3;
                             apiPadam = 0;
                             delay_ms(200);
                            }
                        if(varJuring == 2 && read_adc(0) < 140 && findHome == 3){              // CEK
WARNA !!      item terbesar

                             findHome = 4;
                             apiPadam = 0;
                             delay_ms(200);
                            }
                        if( (read_adc(2) > 61 && read_adc(0) > 380) && findHome == 4 && read_adc(1) > 300
&& varJuring == 2){     // CEK WARNA !! putih
```

```
                        berenti(255,255);
                        for( i=0;i<=10;i++ ){
                              lcd_clear();
                              lcd_gotoxy(0,0);
                              lcd_putsf("HOME!!");
                              berenti(255,255);
                              delay_ms(200);
                              lcd_clear();
                              lcd_putsf(" HOME!!");
                              berenti(255,255);
                              delay_ms(200);
                              lcd_clear();
                              lcd_putsf("  HOME!!");
                              berenti(255,255);
                              delay_ms(200);
                              lcd_clear();
                              lcd_putsf("   HOME!!");
                              delay_ms(200);
                              lcd_clear();
                              lcd_gotoxy(0,0);
                              lcd_putsf("  HOME!!");
                              delay_ms(200);
                              berenti(255,255);
                              lcd_clear();
                              lcd_putsf(" HOME!!");
                              delay_ms(200);
                              berenti(255,255);
                              lcd_clear();
                              lcd_putsf("HOME!!");
                              delay_ms(200);
                              berenti(255,255);
                             }
                        delay_ms(10000);
                        }
                     if( read_adc(0) > 380 && findHome == 2 && read_adc(1) > 300 && varJuring == 1){
// CEK WARNA !! putih

                        berenti(255,255);
                        for( i=0;i<=10;i++ ){
                              lcd_clear();
                              lcd_gotoxy(0,0);
                              lcd_putsf("HOME!!");
                              berenti(255,255);
                              delay_ms(200);
                              lcd_clear();
                              lcd_putsf(" HOME!!");
                              berenti(255,255);
                              delay_ms(200);
                              lcd_clear();
                              lcd_putsf("  HOME!!");
                              berenti(255,255);
                              delay_ms(200);
                              lcd_clear();
                              lcd_putsf("   HOME!!");
                              delay_ms(200);
                              lcd_clear();
                              lcd_gotoxy(0,0);
                              lcd_putsf("  HOME!!");
                              delay_ms(200);
                              berenti(255,255);
                              lcd_clear();
                              lcd_putsf(" HOME!!");
                              delay_ms(200);
```

```
                        berenti(255,255);
                        lcd_clear();
                        lcd_putsf("HOME!!");
                        delay_ms(200);
                        berenti(255,255);
                       }
                    delay_ms(10000);
                    }
                if(detect==2){
                    detect=0;
                    }
                lastError = error;                          // error sebelumnya
                }                                           //end   if(getSrf(0)>15){
            else if(getSrf(6) <= 14){
                    if( getSrf(2) > 30 && getSrf(4) > 30){
                      //if(getSrf(4) > 40){
                            bantingKanan (150,150);
                      //        }
                       }
                    else {
                        bantingKiri(190,190);
                        }
                }                                           //end   else if(getSrf(0) <= 14){

            }
        else{                                               //lagi ada api

            if( (read_adc(1) > 350 ) && detect == 0){       // CEK WARNA !! nilai putih terkecil, nemu api,
tpi kena garis putih pintu masuk dulu

                    lcd_clear();
                    lcd_gotoxy(0,0);
                    lcd_putsf("ada API !!!");
                    maju(255,255);
                    detect=1;
                }                                           // end nemu api, tpi kena garis putih pintu masuk
dulu
            else if( read_adc(1) < 200 && detect ==1 ){     // CEK WARNA !!  abu terbesar
                    detect=2;
                }


            else if(  (read_adc(2) > 71 && detect == 2) && juringPinggir == 0){     // CEK WARNA !!
putih terkecil, dapat juring lingkaran putih (putih>500)

                        apiPadam=1;
                        berenti(255,255);
                        kipas();
                        berenti(255,255);
                        juringPinggir=1;
                        varJuring=1;
                    }
            else if (juringPinggir == 1 && varJuring == 1){
                    bantingKanan(255,255);
                    delay_ms(75);
                    berenti(255,255);
                    kipas();
                    berenti(255,255);
                    apiPadam=1;
                    }
            else if( (read_adc(1) > 300 && detect == 2 ) && juringTengah == 0){     // CEK WARNA putih
terkecil !! dapat juring lingkaran putih di tengah (putih>800)

                    if( juringTengah == 1 ){                // blm mati tuw juring di tengah nya coi
```

```
                bantingKiri(255,255);
                delay_ms(50);
                berenti(255,255);
                kipas();
                berenti(255,255);
                apiPadam=1;
                }
            else {
                apiPadam=1;
                berenti(255,255);
                kipas();
                berenti(255,255);
                juringTengah=1;
                varJuring=2;
                }
            }

else if (juringTengah == 1 && varJuring == 2){
            bantingKiri(255,255);
            delay_ms(75);
            berenti(255,255);
            kipas();
            berenti(255,255);
            apiPadam=1;
            }
else {                      //  else klo blm dapat juring, jalan trs nyari juring putih+klo ada api
    if( getSrf(6) > 14) {                                   //14
            kp = 10;                                        //10
            kd = 15;                                        //15
            ki = 0.005;                                     //0.025
            setPointPWM =50;
            processVariable = getSrf(1);                    //proporsional program
            setPointSensorKiri = 8;
            error = setPointSensorKiri - processVariable;
            proporsional = kp * error;                      // proporsional end
            rate = error - lastError;                       //derivative program
            rateInt = error + lastError;
            Ts=1;                                           //Ts = time sampling
            integral = Ts * (ki * rateInt);
            derivative = (kd/Ts) * rate;                    //derivative
            sinyalControl = proporsional + derivative + integral;  //sinyal control , derivative end
            rightPWM = setPointPWM + sinyalControl;
            leftPWM = setPointPWM - sinyalControl;
            if(rightPWM >= 255) {
                rightPWM = 150;
                }
            if(leftPWM >= 255) {
                leftPWM = 150;
                }
            if(rightPWM <= 0) {
                rightPWM = 0;
                }
            if(leftPWM <= 0) {
                leftPWM = 0;
                }
            leftPWM;
            rightPWM;
            if(leftPWM == 0 && rightPWM == 200){
                bantingKiri(leftPWM,rightPWM);
                lcd_clear();
                lcd_gotoxy(0,0);
                lcd_putsf("belok kiri");
                }
            if(rightPWM == 0 && leftPWM == 200){
```

```
                        bantingKanan(leftPWM,rightPWM);
                        lcd_clear();
                        lcd_gotoxy(0,0);
                        lcd_putsf("belok kanan");
                        }
                    if( (rightPWM <200 || leftPWM < 200)  && (rightPWM > 0 || leftPWM > 0) ){

                        if( (leftPWM < rightPWM) && (leftPWM < 10)){
                            belokKiri(leftPWM,rightPWM);
                            }
                        if( (leftPWM > rightPWM) && (rightPWM < 10) ){
                            belokKanan(leftPWM,rightPWM);
                            }
                        else{
                            maju(leftPWM,rightPWM);
                           }
                        lcd_clear();
                        lcd_gotoxy(0,0);
                        sprintf(text,"S.K=%3d ",processVariable);
                        lcd_puts(text);
                        lcd_gotoxy(0,1);
                        sprintf(text,"r=%3d l=%3d",rightPWM,leftPWM);
                        lcd_puts(text);
                        }  //end   if( (rightPWM<170||leftPWM<170)&&(rightPWM>0||leftPWM>0) ){

                    lastError = error;                              // error sebelumnya
                    }
                else if(getSrf(6) <= 14){
                    if( getSrf(2) > 40 && getSrf(4) > 40){
                        //  if(getSrf(4) > 40){
                            bantingKanan (150,150);
                        //        }
                        }
                    else {
                        bantingKiri(190,190);
                        }
                    }
                }   //  end klo dari else klo blm dapat juring, jln trs nyari juring putih+klo ada api
            }      //   end klo lagi ada api

//=============================================================================
}



void wallKiri(void){
    while(1){
        if(PINB.7 == 1){                                            // ================   lagi gak
ada api
            juringTengah=0;
            juringPinggir=0;
            varRuang4=6;
            if( getSrf(0) > 14 ) {                                      //14
                kp = 10;                                        //10
                kd = 15;                                            //15
                ki = 0.001667;                                          //0.0167
                setPointPWM = 100;
                error = setPointSensorKiri - processVariable;
                processVariable = getSrf(2);
                setPointSensorKiri = 8;
                rate = error - lastError;                               //================== derivative
program
                rateInt = error + lastError;
                Ts=1;
```

```
                proporsional = kp * error;                              //================== proporsional
end                                             //===============>>>> Ts = time sampling
                integral = Ts * (ki * rateInt);
                derivative = (kd/Ts) * rate;                            //===============>>>> derivative
                sinyalControl = proporsional + derivative + integral;           //==================
sinyal control , derivative end
                rightPWM = setPointPWM - sinyalControl;
                leftPWM = setPointPWM + sinyalControl;
                if(rightPWM >= 255) {
                    rightPWM = 130;
                    }
                if(leftPWM >= 255) {
                    leftPWM = 130;
                    }
                if(rightPWM <= 0) {
                    rightPWM = 0;
                    }
                if(leftPWM <= 0) {
                    leftPWM = 0;
                    }
                leftPWM;
                rightPWM;
                if(leftPWM == 0 && rightPWM == 200){
                     bantingKiri(leftPWM,rightPWM);
                     lcd_clear();
                     lcd_gotoxy(0,0);
                     lcd_putsf("belok kiri");
                    }
                if(rightPWM == 0 && leftPWM == 200){
                     bantingKanan(leftPWM,rightPWM);
                     lcd_clear();
                     lcd_gotoxy(0,0);
                     lcd_putsf("belok kanan");
                    }
                if( (rightPWM <200 || leftPWM < 200)  && (rightPWM > 0 || leftPWM > 0) ){
                     if( (leftPWM > rightPWM) && (rightPWM < 40) ){
                          belokKanan(leftPWM,rightPWM);
                          }
                     else if( (leftPWM < rightPWM) && (leftPWM < 40)){
                          belokKiri(leftPWM,rightPWM);
                          }
                     else{
                          maju(leftPWM,rightPWM);
                         }
                     lcd_clear();
                     lcd_gotoxy(0,0);
                     sprintf(text,"S.K=%3d S.D=%3d ",processVariable,getSrf(0));
                     lcd_puts(text);
                     lcd_gotoxy(0,1);
                     sprintf(text,"r=%3d l=%3d",rightPWM,leftPWM);
                     lcd_puts(text);
                     }                                          //end   if( (rightPWM <170 || leftPWM < 170)  &&
(rightPWM > 0 || leftPWM > 0) ){

                if( varRuang4 != 6 && apiPadam == 1 && read_adc(0) < 350 && processVariable > 15){
//CEK WARNA!!  abu terbesar   jgn lupa varRuang4!!!
                     findHome = 1;
                     apiPadam = 0;
                     delay_ms(200);
                     }
                if( findHome == 1 && read_adc(0) < 130){                        //CEK WARNA!! item
terbesar
                     findHome = 2;
                     apiPadam = 0;
```

```
                        delay_ms(200);
                        }

                if(varJuring == 2 && read_adc(0) > 150 && read_adc(0) < 350 && findHome == 2){
//CEK WARNA!! abu terkecil dan terbesar
                        findHome = 3;
                        apiPadam = 0;
                        delay_ms(200);
                        }
                if(varJuring == 2 && read_adc(0) < 150 && findHome == 3){               //CEK WARNA!!
abu terkecil
                        findHome = 4;
                        apiPadam = 0;
                        delay_ms(200);
                        }
                if( varRuang4 == 6 &&  apiPadam == 1){
                        countingHome();
                        if( counterCariHomeDariRuang4 >= 1 && varHomeRuang4 == 0 ){
                                if( (read_adc(0) > 140 && read_adc(0) < 550) &&  (read_adc(1) > 70 &&
read_adc(1) < 600) ){
                                        srfKiri=getSrf(2);
                                        srfKanan=getSrf(4);
                                        if( srfKiri < 20 && srfKiri > 1 && srfKanan < 20 && srfKanan > 1){
                                                        PORTD.4 = 1;
                                                        varHomeRuang4=1;
                                                        lcd_clear();
                                                        lcd_gotoxy(0,1);
                                                        sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                                        lcd_puts(text);
                                                        delay_ms(10);
                                                        }
                                        else {
                                                lcd_clear();
                                                lcd_gotoxy(0,1);
                                                sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                                lcd_puts(text);
                                                PORTD.4 = 0;
                                                goto startKiri;
                                                }
                                        }
                                else {

                                        goto startKiri;
                                        }
                                }
                        else {
                                goto startKiri;
                                }
                        if ( counterCariHomeDariRuang4 >= 1 && varHomeRuang4 == 1){
                                 if( (read_adc(0) > 140 && read_adc(0) < 550) &&  (read_adc(1) > 70 &&
read_adc(1) < 600) ){
                                        srfKiri=getSrf(2);
                                        srfKanan=getSrf(4);
                                        if( srfKiri < 20 && srfKiri > 1 && srfKanan < 20 && srfKanan > 1){
                                                        PORTD.4 = 1;
                                                        varHomeRuang4=2;
                                                        lcd_clear();
                                                        lcd_gotoxy(0,1);
                                                        sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                                        lcd_puts(text);
                                                        delay_ms(10);
                                                        }
                                        else {
                                                        lcd_clear();
```

```
                                        lcd_gotoxy(0,1);
                                        sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                        lcd_puts(text);
                                        PORTD.4 = 0;
                                        varHomeRuang4=0;
                                        goto startKiri;
                                        }
                                }
                            else  {
                                    lcd_clear();
                                    lcd_gotoxy(0,1);
                                    sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                    lcd_puts(text);
                                    PORTD.4 = 0;
                                    varHomeRuang4=0;
                                    goto startKiri;
                                   }
                            }
                        else {
                            varHomeRuang4=0;
                            goto startKiri;
                            }
                        if ( counterCariHomeDariRuang4 >= 1 && varHomeRuang4 == 2){
                                if( (read_adc(0) > 140 && read_adc(0) < 550) &&  (read_adc(1) > 70 &&
read_adc(1) < 600) ){
                                    srfKiri=getSrf(2);
                                    srfKanan=getSrf(4);
                                    if( srfKiri < 20 && srfKiri > 1 && srfKanan < 20 && srfKanan > 1){
                                                PORTD.4 = 1;
                                                varHomeRuang4=3;
                                                lcd_clear();
                                                lcd_gotoxy(0,1);
                                                sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                                lcd_puts(text);
                                                delay_ms(15);
                                                }
                                    else {
                                                lcd_clear();
                                                lcd_gotoxy(0,1);
                                                sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                                lcd_puts(text);
                                                PORTD.4 = 0;
                                                varHomeRuang4=0;
                                                goto startKiri;
                                                }
                                    }
                                else  {
                                        lcd_clear();
                                        lcd_gotoxy(0,1);
                                        sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                        lcd_puts(text);
                                        PORTD.4 = 0;
                                        varHomeRuang4=0;
                                        goto startKiri;
                                       }
                                }
                        else {
                            varHomeRuang4=0;
                            goto startKiri;
                            }
                        if ( counterCariHomeDariRuang4 >= 1 && varHomeRuang4 == 3){
                                if( (read_adc(0) > 140 && read_adc(0) < 550) &&  (read_adc(1) > 70 &&
read_adc(1) < 600) ){
                                    srfKiri=getSrf(2);
```

```
                                srfKanan=getSrf(4);
                                if( srfKiri < 20 && srfKiri > 1 && srfKanan < 20 && srfKanan > 1){
                                        PORTD.4 = 0;
                                        varHomeRuang4=4;
                                        while(1){
                                            wallKananMulus();
                                            }
                                        lcd_clear();
                                        lcd_gotoxy(0,1);
                                        sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                        lcd_puts(text);
                                        }
                                else {
                                        lcd_clear();
                                        lcd_gotoxy(0,1);
                                        sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                        lcd_puts(text);
                                        PORTD.4 = 0;
                                        varHomeRuang4=0;
                                        goto startKiri;
                                        }
                                }
                            else  {
                                    lcd_clear();
                                    lcd_gotoxy(0,1);
                                    sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                    lcd_puts(text);
                                    PORTD.4 = 0;
                                    varHomeRuang4=0;
                                    goto startKiri;
                                   }
                            }
                        else {
                            varHomeRuang4=0;
                            goto startKiri;
                            }
                        }

                    if( varRuang4 == 6 &&  apiPadam == 0){
                        countingLagi();
                        if( counterLagiCui >= 1 && varHomeRuang4 == 0 ){
                             if( (read_adc(0) > 140 && read_adc(0) < 550) &&  (read_adc(1) > 70 &&
read_adc(1) < 600) ){
                                srfKiri=getSrf(2);
                                srfKanan=getSrf(4);
                                if( srfKiri < 20 && srfKiri > 1 && srfKanan < 20 && srfKanan > 1){
                                        PORTD.4 = 1;
                                        varHomeRuang4=1;
                                        lcd_clear();
                                        lcd_gotoxy(0,1);
                                        sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                        lcd_puts(text);
                                        delay_ms(10);
                                        }
                                else {
                                        lcd_clear();
                                        lcd_gotoxy(0,1);
                                        sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                        lcd_puts(text);
                                        PORTD.4 = 0;
                                        goto startKiri;
                                        }
                                }
                             else {
```

```
                        goto startKiri;
                        }
                    }
                else {
                    goto startKiri;
                    }
                if ( counterLagiCui >= 1 && varHomeRuang4 == 1){
                        if( (read_adc(0) > 140 && read_adc(0) < 550) &&  (read_adc(1) > 70 &&
read_adc(1) < 600) ){
                            srfKiri=getSrf(2);
                            srfKanan=getSrf(4);
                            if( srfKiri < 20 && srfKiri > 1 && srfKanan < 20 && srfKanan > 1){
                                    PORTD.4 = 1;
                                    varHomeRuang4=2;
                                    lcd_clear();
                                    lcd_gotoxy(0,1);
                                    sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                    lcd_puts(text);
                                    delay_ms(10);
                                    }
                            else {
                                lcd_clear();
                                lcd_gotoxy(0,1);
                                sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                lcd_puts(text);
                                PORTD.4 = 0;
                                varHomeRuang4=0;
                                goto startKiri;
                                }
                            }
                        else  {
                                lcd_clear();
                                lcd_gotoxy(0,1);
                                sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                lcd_puts(text);
                                PORTD.4 = 0;
                                varHomeRuang4=0;
                                goto startKiri;
                               }
                        }
                else {
                    varHomeRuang4=0;
                    goto startKiri;
                    }
                if ( counterLagiCui >= 1 && varHomeRuang4 == 2){
                        if( (read_adc(0) > 140 && read_adc(0) < 550) &&  (read_adc(1) > 70 &&
read_adc(1) < 600) ){
                            srfKiri=getSrf(2);
                            srfKanan=getSrf(4);
                            if( srfKiri < 20 && srfKiri > 1 && srfKanan < 20 && srfKanan > 1){
                                    PORTD.4 = 1;
                                    varHomeRuang4=3;
                                    lcd_clear();
                                    lcd_gotoxy(0,1);
                                    sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                    lcd_puts(text);
                                    delay_ms(15);
                                    }
                            else {
                                lcd_clear();
                                lcd_gotoxy(0,1);
                                sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                lcd_puts(text);
```

```
                                        PORTD.4 = 0;
                                        varHomeRuang4=0;
                                        goto startKiri;
                                        }
                                }
                            else  {
                                        lcd_clear();
                                        lcd_gotoxy(0,1);
                                        sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                        lcd_puts(text);
                                        PORTD.4 = 0;
                                        varHomeRuang4=0;
                                        goto startKiri;
                                       }
                            }
                        else {
                            varHomeRuang4=0;
                            goto startKiri;
                            }
                        if ( counterLagiCui >= 1 && varHomeRuang4 == 3){
                                if( (read_adc(0) > 140 && read_adc(0) < 550) &&  (read_adc(1) > 70 &&
read_adc(1) < 600) ){
                                    srfKiri=getSrf(2);
                                    srfKanan=getSrf(4);
                                    if( srfKiri < 20 && srfKiri > 1 && srfKanan < 20 && srfKanan > 1){
                                                PORTD.4 = 0;
                                                varHomeRuang4=4;
                                                while(1){
                                                    wallKananMulus();
                                                    }
                                                lcd_clear();
                                                lcd_gotoxy(0,1);
                                                sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                                lcd_puts(text);
                                                }
                                    else {
                                                lcd_clear();
                                                lcd_gotoxy(0,1);
                                                sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                                lcd_puts(text);
                                                PORTD.4 = 0;
                                                varHomeRuang4=0;
                                                goto startKiri;
                                                }
                                    }
                                else  {
                                        lcd_clear();
                                        lcd_gotoxy(0,1);
                                        sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                        lcd_puts(text);
                                        PORTD.4 = 0;
                                        varHomeRuang4=0;
                                        goto startKiri;
                                       }
                                }
                            else {
                                varHomeRuang4=0;
                                goto startKiri;
                                }
                            }

                    startKiri:
                    if( (read_adc(2) > 135 && read_adc(0) > 450) && findHome == 4 && read_adc(1) > 520
&& varJuring == 2){       //CEK WARNA!!  putih
```

```
                berenti(255,255);
                kipas();
                for( i=0;i<=10;i++ ){
                      lcd_clear();
                      lcd_gotoxy(0,0);
                      lcd_putsf("HOME!!");
                      berenti(255,255);
                      delay_ms(200);
                      lcd_clear();
                      lcd_putsf(" HOME!!");
                      berenti(255,255);
                      delay_ms(200);
                      lcd_clear();
                      lcd_putsf("  HOME!!");
                      berenti(255,255);
                      delay_ms(200);
                      lcd_clear();
                      lcd_putsf("   HOME!!");
                      delay_ms(200);
                      lcd_clear();
                      lcd_gotoxy(0,0);
                      lcd_putsf("  HOME!!");
                      delay_ms(200);
                      berenti(255,255);
                      lcd_clear();
                      lcd_putsf(" HOME!!");
                      delay_ms(200);
                      berenti(255,255);
                      lcd_clear();
                      lcd_putsf("HOME!!");
                      delay_ms(200);
                      berenti(255,255);
                     }
                delay_ms(10000);
                }

          if( read_adc(0) > 450 && findHome == 2 && read_adc(1) > 520 && varJuring == 1){
//CEK WARNA!!  putih

                berenti(255,255);
                kipas();
                for( i=0;i<=10;i++ ){
                      lcd_clear();
                      lcd_gotoxy(0,0);
                      lcd_putsf("HOME!!");
                      berenti(255,255);
                      delay_ms(200);
                      lcd_clear();
                      lcd_putsf(" HOME!!");
                      berenti(255,255);
                      delay_ms(200);
                      lcd_clear();
                      lcd_putsf("  HOME!!");
                      berenti(255,255);
                      delay_ms(200);
                      lcd_clear();
                      lcd_putsf("   HOME!!");
                      delay_ms(200);
                      lcd_clear();
                      lcd_gotoxy(0,0);
                      lcd_putsf("  HOME!!");
                      delay_ms(200);
                      berenti(255,255);
```

```
                        lcd_clear();
                        lcd_putsf(" HOME!!");
                        delay_ms(200);
                        berenti(255,255);
                        lcd_clear();
                        lcd_putsf("HOME!!");
                        delay_ms(200);
                        berenti(255,255);
                        }
                    delay_ms(10000);
                    }                                   // error sebelumnya
                lastError = error;
                }                                       //end   if(getSrf(0)>15){
          else if(getSrf(0) <= 14){
                    if( getSrf(2) > 40 && getSrf(4) > 40 && getSrf(6) < 9){
                        bantingKiri(150,150);
                        }
                    else {
                        bantingKanan(190,190);
                        }
                }                                                   //end   else if(getSrf(0) <= 14){
          }                                                         // end if lg gk ada api
    else{                                                           // lagi ada api

        if( read_adc(1) > 500 && detect == 0){                      //CEK WARNA!!  putih   // nemu api,
tpi kena garis putih pintu masuk dulu

                lcd_clear();
                lcd_gotoxy(0,0);
                lcd_putsf("ada API !!!");
                maju(255,255);
                detect=1;
           }                                                        // end nemu api, tpi kena garis putih
pintu masuk dulu
        else if( read_adc(1) < 300 && detect ==1 ){                 //CEK WARNA!!  abu terbesar
                detect=2;
           }
        else if(  (read_adc(0) > 550 && detect == 2) && juringPinggir == 0){        //CEK WARNA!!  putih
terkecil  adc 0, dapat juring lingkaran putih (putih>500)

                    apiPadam=1;
                    berenti(255,255);
                    kipas();
                    berenti(255,255);
                    juringPinggir=1;
                    varJuring=1;
                }
        else if (juringPinggir == 1 && varJuring == 1){
                bantingKiri(255,255);
                delay_ms(75);
                berenti(255,255);
                kipas();
                berenti(255,255);
                apiPadam=1;
                }
        else if( (read_adc(2) > 135 && detect == 2 ) && juringTengah==0){       //CEK WARNA!!  putih
terkecil    // ====================================== dapat juring lingkaran putih di tengah
(putih>800)

              if( juringTengah == 1 ){                              //  blm mati, juring di tengah
                    bantingKanan(255,255);
                    delay_ms(50);
                    berenti(255,255);
                    kipas();
```

```
                berenti(255,255);
                apiPadam=1;
                }
            else {
                apiPadam=1;
                berenti(255,255);
                kipas();
                berenti(255,255);
                juringTengah=1;
                varJuring=2;
                }
            }

else if (juringTengah == 1 && varJuring == 2){
            bantingKanan(255,255);
            delay_ms(75);
            berenti(255,255);
            kipas();
            berenti(255,255);
            apiPadam=1;
            }                       //  end klo dapat juring lingkaran putih (putih>800)

else {                              //  else klo blm dapat juring, jalan trs nyari juring putih+klo ada api
    if( getSrf(0) > 14) {
            kp = 10;                                    //10
            kd = 10;                                    //15
            ki = 0.025;
            setPointPWM = 50;
            processVariable = getSrf(2);                // proporsional program
            setPointSensorKiri = 8;
            error = setPointSensorKiri - processVariable;
            proporsional = kp * error;                  // proporsional end
            rate = error - lastError;                   // derivative program
            rateInt = error + lastError;
            Ts=1;                                       // Ts = time sampling
            integral = Ts * (ki * rateInt);
            derivative = (kd/Ts) * rate;                // derivative
            sinyalControl = proporsional + derivative + integral;       // sinyal control
            rightPWM = setPointPWM - sinyalControl;
            leftPWM = setPointPWM + sinyalControl;
            if(rightPWM >= 255) {
                rightPWM = 140;
                }
            if(leftPWM >= 255) {
                leftPWM = 140;
                }
            if(rightPWM <= 0) {
                rightPWM = 0;
                }
            if(leftPWM <= 0) {
                leftPWM = 0;
                }
            leftPWM;
            rightPWM;
            if(leftPWM == 0 && rightPWM == 200){
                bantingKiri(leftPWM,rightPWM);
                lcd_clear();
                lcd_gotoxy(0,0);
                lcd_putsf("belok kiri");
                }
            if(rightPWM == 0 && leftPWM == 200){
                bantingKanan(leftPWM,rightPWM);
                lcd_clear();
                lcd_gotoxy(0,0);
```

```
                    lcd_putsf("belok kanan");
                    }
                if( (rightPWM <200 || leftPWM < 200)  && (rightPWM > 0 || leftPWM > 0) ){
                    if( (leftPWM > rightPWM) && (rightPWM < 30) ){
                        belokKanan(leftPWM,rightPWM);
                        }
                    if( (leftPWM < rightPWM) && (leftPWM < 30)){
                        belokKiri(leftPWM,rightPWM);
                        }
                    else{
                        maju(leftPWM,rightPWM);
                        }
                    lcd_clear();
                    lcd_gotoxy(0,0);
                    sprintf(text,"S.K=%3d S.D=%3d ",processVariable,getSrf(0));
                    lcd_puts(text);
                    lcd_gotoxy(0,1);
                    sprintf(text,"r=%3d l=%3d",rightPWM,processVariable);
                    lcd_puts(text);
                    }   //end   if( (rightPWM <170 || leftPWM < 170)  && (rightPWM > 0 || leftPWM > 0) ){
                lastError = error;                          // error sebelumnya
                }                                           // end   if(getSrf(0)>15){
            else if(getSrf(0) <= 14){
                    if( getSrf(2) > 40 && getSrf(4) > 40  && getSrf(6) < 9){
                        bantingKiri(150,150);
                        }
                    else {
                        bantingKanan(190,190);
                        }
                }                   //end   else if(getSrf(0) <= 14){
             }                      // end klo dari else klo blm dapat juring, jln trs nyari juring putih+klo ada api
         }                          // end klo lagi ada api

    }
}                                   //end void wallKiri


void setKeypad(void){
      b:
      lcd_clear();
      lcd_gotoxy(0,0);
      sprintf(text,"set Kp ? u=3,d=2");
      lcd_puts(text);
      lcd_gotoxy(0,1);
      sprintf(text,"%f",kp);
      lcd_puts(text);
      delay_ms(200);
     if( kp != 0 ){
          kp=kp;
          }
     else if( ki != 0 ){
          ki=ki;
          }
     else if( kd != 0 ){
          kd=kd;
          }
     else{
          kp=0;
          ki=0;
          kd=0;
          }
     menu=0;
     setOK=0;
     setKd=0;
```

```
setKp=1;
setKi=0;
while(1){
    if(PINE.4 == 0 && menu == 0){
            lcd_clear();
            lcd_gotoxy(0,0);
            sprintf(text,"set Kd ? u=3,d=2");
            lcd_puts(text);
            lcd_gotoxy(0,1);
            sprintf(text,"%f",kd);
            lcd_puts(text);
            menu=2;
            setKd=1;
            setKp=0;
            setKi=0;
            delay_ms(200);
           }
    else if(PINE.4 == 0 && menu == 2){
            lcd_clear();
            lcd_gotoxy(0,0);
            sprintf(text,"set Ki? u=3,d=2");
            lcd_puts(text);
            lcd_gotoxy(0,1);
            sprintf(text,"%f",ki);
            lcd_puts(text);
            menu=3;
            setKd=0;
            setKp=0;
            setKi=1;
            delay_ms(200);
           }
    else if(PINE.4 == 0 && menu == 3 ){
            lcd_clear();
            lcd_gotoxy(0,0);
            sprintf(text,"YES=PORTE.3");
            lcd_puts(text);
            lcd_gotoxy(0,1);
            lcd_putsf("NO=PORTE.2");
            delay_ms(200);
            setOK=1;
            setKp=0;
            setKd=0;
            setKi=0;
           }

    else if(PINE.2 == 0 && setOK == 1 ){
            goto b;
           }
    else if(PINE.3 == 0 && setOK == 1 ){

            kp=kp;
            ki=ki;
            kd=kd;
            lcd_clear();
            lcd_gotoxy(0,0);
            sprintf(text," 3 ");
            lcd_puts(text);
            PORTD.4 = 1;
            delay_ms(200);
            PORTD.4 = 0;
            delay_ms(200);
            lcd_clear();
            lcd_gotoxy(0,0);
            sprintf(text," 2 ");
```

```
            lcd_puts(text);
            PORTD.4 = 1;
            delay_ms(200);
            PORTD.4 = 0;
            delay_ms(200);
            lcd_clear();
            lcd_gotoxy(0,0);
            sprintf(text," 1 ");
            lcd_puts(text);
            PORTD.4 = 1;
            delay_ms(200);
            PORTD.4 = 0;
            delay_ms(200);
            lcd_clear();
            lcd_gotoxy(0,0);
            sprintf(text," START ");
            lcd_puts(text);
            PORTD.4 = 1;
            delay_ms(200);
            PORTD.4 = 0;
            delay_ms(200);
            PORTD.4 = 1;
            delay_ms(150);
            PORTD.4 = 0;
            delay_ms(100);
            PORTD.4 = 1;
            delay_ms(150);
            PORTD.4 = 0;
            delay_ms(100);
            PORTD.4 = 1;
            delay_ms(100);
            PORTD.4 = 0;
            delay_ms(50);
            break;
           }

    else if(PINE.3 == 0 && setKp == 1){
            lcd_clear();
            lcd_gotoxy(0,0);
            sprintf(text,"set Kp ? u=3,d=2");
            lcd_puts(text);
            lcd_gotoxy(0,1);
            kp=kp+0.5;
            sprintf(text,"%f",kp);
            lcd_puts(text);
            delay_ms(200);
           }
    else if(PINE.3 == 0 && setKi == 1){
            lcd_clear();
            lcd_gotoxy(0,0);
            sprintf(text,"set Ki ? u=3,d=2");
            lcd_puts(text);
            lcd_gotoxy(0,1);
            ki=ki+0.5;
            sprintf(text,"%f",ki);
            lcd_puts(text);
            delay_ms(200);
           }
    else if(PINE.3 == 0 && setKd == 1){
            lcd_clear();
            lcd_gotoxy(0,0);
            sprintf(text,"set Kd ? u=3,d=2");
            lcd_puts(text);
            lcd_gotoxy(0,1);
```

```
                sprintf(text,"%f",kd);
                lcd_puts(text);
                kd=kd+0.5;
                delay_ms(200);
                }
        else if(PINE.2 == 0 && setKp == 1){
                lcd_clear();
                lcd_gotoxy(0,0);
                sprintf(text,"set Kp ? u=3,d=2");
                lcd_puts(text);
                lcd_gotoxy(0,1);
                kp=kp-0.5;
                sprintf(text,"%f",kp);
                lcd_puts(text);
                delay_ms(200);
                }
        else if(PINE.2 == 0 && setKi == 1){
                lcd_clear();
                lcd_gotoxy(0,0);
                sprintf(text,"set Ki ? u=3,d=2");
                lcd_puts(text);
                lcd_gotoxy(0,1);
                ki=ki-0.5;
                sprintf(text,"%f",ki);
                lcd_puts(text);
                delay_ms(200);
                }
        else if(PINE.2 == 0 && setKd == 1){
                lcd_clear();
                lcd_gotoxy(0,0);
                sprintf(text,"set Kd ? u=3,d=2");
                lcd_puts(text);
                lcd_gotoxy(0,1);
                kd=kd-0.5;
                sprintf(text,"%f",kd);
                lcd_puts(text);
                delay_ms(200);
                }
      }

}

void main(void)
{
// Declare your local variables here

// Input/Output Ports initialization
// Port A initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=P State2=T State1=T State0=T
PORTA=0x08;
DDRA=0x00;

// Port B initialization
// Func7=In Func6=Out Func5=Out Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=0 State5=0 State4=T State3=T State2=T State1=T State0=T
PORTB=0x00;
DDRB=0x60;



// Port C initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
```

```
PORTC=0x00;
DDRC=0x00;

// Port D initialization
// Func7=In Func6=In Func5=Out Func4=Out Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=0 State4=0 State3=T State2=T State1=T State0=T
PORTD=0x00;
DDRD=0x30;

// Port E initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=P State3=P State2=P State1=P State0=P
PORTE=0x1F;
DDRE=0x00;


// Port F initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTF=0x00;
DDRF=0x00;

// Port G initialization
// Func4=In Func3=In Func2=In Func1=In Func0=In
// State4=T State3=T State2=T State1=T State0=T
PORTG=0x00;
DDRG=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=FFh
// OC0 output: Disconnected
ASSR=0x00;
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: 11059.200 kHz
// Mode: Ph. correct PWM top=00FFh
// OC1A output: Non-Inv.
// OC1B output: Non-Inv.
// OC1C output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer 1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
// Compare C Match Interrupt: Off
TCCR1A=0xA1;
TCCR1B=0x01;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;
OCR1CH=0x00;
OCR1CL=0x00;
```

```
// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer 2 Stopped
// Mode: Normal top=FFh
// OC2 output: Disconnected
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;


// Timer/Counter 3 initialization
// Clock source: System Clock
// Clock value: Timer 3 Stopped
// Mode: Normal top=FFFFh
// Noise Canceler: Off
// Input Capture on Falling Edge
// OC3A output: Discon.
// OC3B output: Discon.
// OC3C output: Discon.
// Timer 3 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
// Compare C Match Interrupt: Off
TCCR3A=0x00;
TCCR3B=0x00;
TCNT3H=0x00;
TCNT3L=0x00;
ICR3H=0x00;
ICR3L=0x00;
OCR3AH=0x00;
OCR3AL=0x00;
OCR3BH=0x00;
OCR3BL=0x00;
OCR3CH=0x00;
OCR3CL=0x00;

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
// INT3: Off
// INT4: Off
// INT5: Off
// INT6: Off
// INT7: Off
EICRA=0x00;
EICRB=0x00;
EIMSK=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x00;
ETIMSK=0x00;

 // USART0 initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART0 Receiver: On
// USART0 Transmitter: On
// USART0 Mode: Asynchronous
// USART0 Baud rate: 9600
UCSR0A=0x00;
UCSR0B=0x58;
```

```
UCSR0C=0x06;
UBRR0H=0x00;
UBRR0L=0x47;

// // USART0 initialization
// // Communication Parameters: 8 Data, 1 Stop, No Parity
// // USART0 Receiver: On
// // USART0 Transmitter: On
// // USART0 Mode: Asynchronous
// // USART0 Baud rate: 9600
// UCSR0A=0x00;
// UCSR0B=0x58;
// UCSR0C=0x06;
// UBRR0H=0x00;
// UBRR0L=0x47;

// USART1 initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART1 Receiver: On
// USART1 Transmitter: On
// USART1 Mode: Asynchronous
// USART1 Baud rate: 9600
UCSR1A=0x00;
UCSR1B=0x18;
UCSR1C=0x06;
UBRR1H=0x00;
UBRR1L=0x47;


// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
SFIOR=0x00;

// ADC initialization
// ADC Clock frequency: 691.200 kHz
// ADC Voltage Reference: AREF pin
ADMUX=ADC_VREF_TYPE & 0xff;
ADCSRA=0x84;

// I2C Bus initialization
i2c_init();

// LCD module initialization
lcd_init(16);




//================================================================================
//setKeypad();
//==============================VARIABLE==========================================
x=0;
matlab=1;
detect=0;
apiPadam=0;
counter=0;
findHome=0;
juringTengah=0;
juringPinggir=0;
varJuring=0;
```

```
cariRuang4=0;
varRuang4=0;
nilaiCounterRuang4=0;
counterRuang4=0;
count=0;
count2=0;
varHomeRuang4=0;
varPos=0;

//============================SOUND ACTIVATION=======================
//============================SET ADDRESS I2C SRF08====================
//setAddress(0xE4,0xE0);
      menu=0;
      while(1){
      if(PINE.4==0 && menu==0){
                        while(1){
                              lcd_clear();
                              lcd_gotoxy(0,0);
                              sprintf(text,"%d %d %d ", read_adc(0), read_adc(1), read_adc(2));
                              lcd_puts(text);
                              lcd_gotoxy(0,1);
                              sprintf(text,"%3d %3d %d",getSrf(2),getSrf(4),PINB.7);         //  srf4 tuw kanan
                              lcd_puts(text);
                              menu=1;
                              delay_ms(150);
                              if(PINE.3==0 && menu==1){break;}
                        }
                  }delay_ms(150);
      if(PINE.4==0 && menu==1){
                         while(1){
                              lcd_clear();
                              lcd_gotoxy(0,0);
                              sprintf(text,"%3d %3d %3d %3d ", getSrf(1),getSrf(2),getSrf(4),getSrf(0));
                              lcd_puts(text);
                              lcd_gotoxy(0,1);
                              sprintf(text,"%3d  ",getSrf(6),);
                              lcd_puts(text);
                              menu=2;
                              delay_ms(150);
                              if(PINE.3==0 && menu==2){break;}
                        }
                  }delay_ms(150);
      if(PINE.4==0 && menu==2){
                         while(1){
                               counting();
                               countingLagi();
                               countingHome();
                              lcd_clear();
                              lcd_gotoxy(0,0);
                              sprintf(text," %3d %3d %3d ",
counterCariHomeDariRuang4,counterRuang4,counterLagiCui);
                              lcd_puts(text);
                              lcd_gotoxy(0,1);
                              sprintf(text,"%3d %3d ",getSrf(4),getSrf(2));
                              lcd_puts(text);
                              menu=3;
                              delay_ms(200);
                              if(PINE.3==0 && menu==3){break;}
                               }
                        }delay_ms(150);
       if(PINE.4==0 && menu==3){break;}delay_ms(150);
       }
PORTD.4=1;delay_ms(100);
PORTD.4=0;delay_ms(100);
```

```
PORTD.4=1;delay_ms(100);
PORTD.4=0;delay_ms(100);
PORTD.4=1;delay_ms(100);
PORTD.4=0;delay_ms(100);
soundAktive();
cariPosisiAwal();
while (1)  {
      if(PINB.7 == 1){                          //   lagi gak ada api
            juringTengah=0;
            juringPinggir=0;                    //  reset juring tengah, klo juring tgh =1 muter kiri trs

            if( getSrf(6) > 14) {                           //14
                  kp = 10;                                  //10
                  kd = 15;                                  //15
                  ki = 0.005;                               //0.025
                  setPointPWM =110;
                  processVariable = getSrf(1);              // proporsional program
                  setPointSensorKiri = 8;
                  error = setPointSensorKiri - processVariable;
                  proporsional = kp * error;                // proporsional end
                  rate = error - lastError;                 // derivative program
                  rateInt = error + lastError;
                  Ts=1;                                     // Ts = time sampling
                  integral = Ts * (ki * rateInt);
                  derivative = (kd/Ts) * rate;              //  derivative
                  sinyalControl = proporsional + derivative + integral;       // sinyal control
                  rightPWM = setPointPWM + sinyalControl;
                  leftPWM = setPointPWM - sinyalControl;
                  if(rightPWM >= 255) {
                        rightPWM = 130;
                        }
                  if(leftPWM >= 255) {
                        leftPWM = 130;
                        }
                  if(rightPWM <= 0) {
                        rightPWM = 0;
                        }
                  if(leftPWM <= 0) {
                        leftPWM = 0;
                        }
                  leftPWM;
                  rightPWM;
                  if(leftPWM == 0 && rightPWM == 200){
                        bantingKiri(leftPWM,rightPWM);
                        lcd_clear();
                        lcd_gotoxy(0,0);
                        lcd_putsf("belok kiri");
                        }
                  if(rightPWM == 0 && leftPWM == 200){
                        bantingKanan(leftPWM,rightPWM);
                        lcd_clear();
                        lcd_gotoxy(0,0);
                        lcd_putsf("belok kanan");
                        }
                  if( (rightPWM <200 || leftPWM < 200)  && (rightPWM > 0 || leftPWM > 0) ){

                        if( (leftPWM < rightPWM) && (leftPWM < 40)){
                              belokKiri(leftPWM,rightPWM);
                              }
                        if( (leftPWM > rightPWM) && (rightPWM < 40) ){
                              belokKanan(leftPWM,rightPWM);
                              }
                        else{
                              maju(leftPWM,rightPWM);
```

```
                    }
                lcd_clear();
                lcd_gotoxy(0,0);
                sprintf(text,"S.K=%3d ",processVariable);
                lcd_puts(text);
                lcd_gotoxy(0,1);
                sprintf(text,"r=%3d l=%3d",rightPWM,leftPWM);
                lcd_puts(text);
                }                                   //end   if( (rightPWM <170 || leftPWM < 170)
&& (rightPWM > 0 || leftPWM > 0) ){

//**********************************KIRIM KE MATLAB*****************************
            x=x+1;
//*************PWM KIRI ******
//              if(x==1){
//                  dataLeftPWM[0]=leftPWM;
//                  dataTerakhir=dataLeftPWM[0];
//                  }
//              if(x==2){
//                  dataLeftPWM[1]=dataLeftPWM[0]+leftPWM;
//                  dataTerakhir=dataLeftPWM[1];
//                  }
//              if(x==3){
//                  dataLeftPWM[2]=dataLeftPWM[1]+leftPWM;
//                  dataTerakhir=dataLeftPWM[2]/2;
//                  }
//              if(x==4){
//                  dataLeftPWM[3]=dataLeftPWM[2]+leftPWM;
//                  prosesLeftPWM =dataLeftPWM[3] / 3;
//                  dataTerakhir=prosesLeftPWM;
//                  printf("%f",prosesLeftPWM);
//                  putchar(0x0A);
//                  x=1;
//                  }
//************SINYAL KONTROL*****
//
//              if(x==1){
//                  dataSinyalControl[0]=sinyalControl;
//                  }
//                          else if(x==2){
//                  dataSinyalControl[1]=dataSinyalControl[0]+sinyalControl;
//
//                  }
//              else if(x==3){
//                  dataSinyalControl[2]=dataSinyalControl[1]+sinyalControl;
//                  prosesSinyalControl =dataSinyalControl[2] / 3;
//                  printf("%f",prosesSinyalControl);
//                  putchar(0x0A);
//
//                  x=0;
//                  }

//************ERROR*****************
            if(x==1){
                dataError[0]=error;
                }
            else if(x==2){
                dataError[1]=dataError[0]+error;

                }
             else if(x==3){
                dataError[2]=dataError[1]+error;
                prosesSinyalControl =dataError[2] / 3;
                printf("%f",prosesSinyalControl);
```

```
                    putchar(0x0A);
                    x=0;
                   }

//*************************************************************************************************
//***********PROSES VARIABEL*****
//
//                  if(x==1){
//                      dataPV[0]=processVariable;
//                      }
//                              else if(x==2){
//                      dataPV[1]=dataPV[0]+processVariable;
//
//                      }
//                  else if(x==3){
//                      dataPV[2]=dataPV[1]+processVariable;
//                      prosesSinyalControl =dataPV[2] / 3;
//                      printf("%f",prosesSinyalControl);
//                      putchar(0x0A);
//                      x=0;
//                      }

//*************************************************************************************************
////////////////////  COUNTER  ////////////////////////
                    counting();
                    if( counterRuang4 >= 4 && apiPadam == 0 && varRuang4 == 0 ){
                        if( (read_adc(0) > 140 && read_adc(0) < 550) &&  (read_adc(1) > 70 &&
read_adc(1) < 600) ){          // CEK WARNA ABU

                            srfKiri=getSrf(2);
                            srfKanan=getSrf(4);
                            if( srfKiri < 20 && srfKiri > 1 && srfKanan < 13 && srfKanan > 1){

                                        PORTD.4 = 1;
                                        varRuang4=1;
                                        lcd_clear();
                                        lcd_gotoxy(0,1);
                                        sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                        lcd_puts(text);
                                        delay_ms(10);
                                        }
                            else {
                                lcd_clear();
                                lcd_gotoxy(0,1);
                                sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                lcd_puts(text);
                                PORTD.4 = 0;
                                goto start;
                                }
                            }
                        else {
                            goto start;
                            }
                        }
                    else {
                            goto start;
                            }
                    if ( counterRuang4 >= 4 && apiPadam == 0 && varRuang4 == 1){

                         if( (read_adc(0) > 140 && read_adc(0) < 550) &&  (read_adc(1) > 70 &&
read_adc(1) < 600) ){
                                srfKiri=getSrf(2);
                                srfKanan=getSrf(4);
                                if( srfKiri < 20 && srfKiri > 1 && srfKanan < 13 && srfKanan > 1){
```

```
                                        PORTD.4 = 0;
                                        varRuang4=2;
                                        lcd_clear();
                                        lcd_gotoxy(0,1);
                                        sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                        lcd_puts(text);
                                        delay_ms(10);
                                        }
                                else {
                                        lcd_clear();
                                        lcd_gotoxy(0,1);
                                        sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                        lcd_puts(text);
                                        PORTD.4 = 0;
                                        varRuang4=0;
                                        goto start;
                                        }
                                }
                        else  {
                                lcd_clear();
                                lcd_gotoxy(0,1);
                                sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                lcd_puts(text);
                                PORTD.4 = 0;
                                varRuang4=0;
                                goto start;
                               }
                        }
                else {
                     varRuang4=0;
                     goto start;
                     }
                if ( counterRuang4 >= 4 && apiPadam == 0 && varRuang4 == 2){

                   if( (read_adc(0) > 140 && read_adc(0) < 550) &&  (read_adc(1) > 70 &&
read_adc(1) < 600) ){
                                srfKiri=getSrf(2);
                                srfKanan=getSrf(4);
                                if( srfKiri < 20 && srfKiri > 1 && srfKanan < 13 && srfKanan > 1){
                                        PORTD.4 = 0;
                                        varRuang4=3;
                                        lcd_clear();
                                        lcd_gotoxy(0,1);
                                        sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                        lcd_puts(text);
                                        delay_ms(10);
                                        }
                                else {
                                        lcd_clear();
                                        lcd_gotoxy(0,1);
                                        sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                        lcd_puts(text);
                                        PORTD.4 = 0;
                                        varRuang4=0;
                                        goto start;
                                        }
                                }
                        else  {
                                lcd_clear();
                                lcd_gotoxy(0,1);
                                sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                lcd_puts(text);
                                PORTD.4 = 0;
                                varRuang4=0;
```

```
                                goto start;
                                }
                        }
                else {
                    varRuang4=0;
                    goto start;
                    }
                if ( counterRuang4 >= 4 && apiPadam == 0 && varRuang4 == 3){

                    if( (read_adc(0) > 140 && read_adc(0) < 550) &&  (read_adc(1) > 70 &&
read_adc(1) < 600) ){
                            srfKiri=getSrf(2);
                            srfKanan=getSrf(4);
                            if( srfKiri < 20 && srfKiri > 1 && srfKanan < 13 && srfKanan > 1){
                                    PORTD.4 = 0;
                                    varRuang4=4;
                                    lcd_clear();
                                    lcd_gotoxy(0,1);
                                    sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                    lcd_puts(text);
                                    delay_ms(10);
                                    }
                            else {
                                    lcd_clear();
                                    lcd_gotoxy(0,1);
                                    sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                    lcd_puts(text);
                                    PORTD.4 = 0;
                                    varRuang4=0;
                                    goto start;
                                    }
                            }
                        else  {
                                lcd_clear();
                                lcd_gotoxy(0,1);
                                sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                lcd_puts(text);
                                PORTD.4 = 0;
                                varRuang4=0;
                                goto start;
                                }
                        }
                else {
                    varRuang4=0;
                    goto start;
                    }
                if ( counterRuang4 >= 4 && apiPadam == 0 && varRuang4 == 4){

                    if( (read_adc(0) > 140 && read_adc(0) < 550) &&  (read_adc(1) > 70 &&
read_adc(1) < 600) ){
                            srfKiri=getSrf(2);
                            srfKanan=getSrf(4);
                            if( srfKiri < 20 && srfKiri > 1 && srfKanan < 13 && srfKanan > 1){
                                    PORTD.4 = 0;
                                    varRuang4=5;
                                    lcd_clear();
                                    lcd_gotoxy(0,1);
                                    sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                    lcd_puts(text);
                                    delay_ms(15);
                                    }
                            else {
                                    lcd_clear();
                                    lcd_gotoxy(0,1);
```

```
                                    sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                    lcd_puts(text);
                                    PORTD.4 = 0;
                                    varRuang4=0;
                                    goto start;
                                    }
                              }
                        else  {
                                    lcd_clear();
                                    lcd_gotoxy(0,1);
                                    sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                    lcd_puts(text);
                                    PORTD.4 = 0;
                                    varRuang4=0;
                                    goto start;
                                   }
                        }
                  else {
                        varRuang4=0;
                        goto start;
                        }
                  if ( counterRuang4 >= 4 && apiPadam == 0 && varRuang4 == 5){

                     if( (read_adc(0) > 140 && read_adc(0) < 550) &&  (read_adc(1) > 70 &&
read_adc(1) < 600) ){
                              srfKiri=getSrf(2);
                              srfKanan=getSrf(4);
                              if( srfKiri < 20 && srfKiri > 1 && srfKanan < 13 && srfKanan > 1){
                                          PORTD.4 = 0;
                                          varRuang4=6;
                                          //wallKiri();
                                          lcd_clear();
                                          lcd_gotoxy(0,1);
                                          sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                          lcd_puts(text);
                                          }
                              else {
                                          lcd_clear();
                                          lcd_gotoxy(0,1);
                                          sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                          lcd_puts(text);
                                          PORTD.4 = 0;
                                          varRuang4=0;
                                          goto start;
                                          }
                              }
                        else  {
                                    lcd_clear();
                                    lcd_gotoxy(0,1);
                                    sprintf(text,"%3d %3d ",srfKiri,srfKanan);
                                    lcd_puts(text);
                                    PORTD.4 = 0;
                                    varRuang4=0;
                                    goto start;
                                   }
                        }
                  else {
                        varRuang4=0;
                        goto start;
                        }

//////////////////////END COUNTER ////////////////////
                  start:
```

```
                    if( apiPadam == 1 && read_adc(0) < 500 && getSrf(1) > 12){ // CEK WARNA!!abu
terbesar
                        findHome = 1;
                        apiPadam = 0;
                        delay_ms(200);
                        }
                    if( findHome == 1 && read_adc(0) < 200){                    // CEK WARNA, item
                        findHome = 2;
                        apiPadam = 0;
                        delay_ms(200);
                        }
                    if(varJuring == 2 && read_adc(0) > 140 && read_adc(0) < 600 && findHome == 2){
// CEK WARNA !!    abu terkecil dan terbesar

                        findHome = 3;
                        apiPadam = 0;
                        delay_ms(200);
                       }
                    if(varJuring == 2 && read_adc(0) < 140 && findHome == 3){  // CEK WARNA !!
item terbesar

                        findHome = 4;
                        apiPadam = 0;
                        delay_ms(200);
                       }
                    if( (read_adc(2) > 135 && read_adc(0) > 380) && findHome == 4 && read_adc(1) >
350 && varJuring == 2){     // CEK WARNA !! putih

                        berenti(255,255);
                        for( i=0;i<=10;i++ ){
                            lcd_clear();
                            lcd_gotoxy(0,0);
                            lcd_putsf("HOME!!");
                            berenti(255,255);
                            delay_ms(200);
                            lcd_clear();
                            lcd_putsf(" HOME!!");
                            berenti(255,255);
                            delay_ms(200);
                            lcd_clear();
                            lcd_putsf("  HOME!!");
                            berenti(255,255);
                            delay_ms(200);
                            lcd_clear();
                            lcd_putsf("   HOME!!");
                            delay_ms(200);
                            lcd_clear();
                            lcd_gotoxy(0,0);
                            lcd_putsf("  HOME!!");
                            delay_ms(200);
                            berenti(255,255);
                            lcd_clear();
                            lcd_putsf(" HOME!!");
                            delay_ms(200);
                            berenti(255,255);
                            lcd_clear();
                            lcd_putsf("HOME!!");
                            delay_ms(200);
                            berenti(255,255);
                           }
                        delay_ms(10000);
                        }
  ////////////////////////////////////////////////////////////////
```

```
                    if( read_adc(0) > 380 && findHome == 2 && read_adc(1) > 350 && varJuring == 1){
// CEK WARNA !!        putih

                        berenti(255,255);
                        for( i=0;i<=10;i++ ){
                            lcd_clear();
                            lcd_gotoxy(0,0);
                            lcd_putsf("HOME!!");
                            berenti(255,255);
                            delay_ms(200);
                            lcd_clear();
                            lcd_putsf(" HOME!!");
                            berenti(255,255);
                            delay_ms(200);
                            lcd_clear();
                            lcd_putsf("  HOME!!");
                            berenti(255,255);
                            delay_ms(200);
                            lcd_clear();
                            lcd_putsf("   HOME!!");
                            delay_ms(200);
                            lcd_clear();
                            lcd_gotoxy(0,0);
                            lcd_putsf("  HOME!!");
                            delay_ms(200);
                            berenti(255,255);
                            lcd_clear();
                            lcd_putsf(" HOME!!");
                            delay_ms(200);
                            berenti(255,255);
                            lcd_clear();
                            lcd_putsf("HOME!!");
                            delay_ms(200);
                            berenti(255,255);
                           }
                        delay_ms(10000);
                        }
                    if(detect==2){
                        detect=0;
                        }
                    lastError = error;                          // error sebelumnya
                    }                                           //  end   if(getSrf(0)>15){
                else if(getSrf(6) <= 14){
                        if( getSrf(2) > 30 && getSrf(4) > 30){
                            //if(getSrf(4) > 40){
                                bantingKanan (150,150);
                            //        }
                             }
                        else {
                            bantingKiri(190,190);
                             }
                    }                                           //end   else if(getSrf(0) <= 14){

                }
            else{                                               // lagi ada api

                if( (read_adc(1) > 500 ) && detect == 0){       // CEK WARNA !! nilai putih terkecil, nemu
api, tpi kena garis putih pintu masuk dulu

                    lcd_clear();
                    lcd_gotoxy(0,0);
                    lcd_putsf("ada API !!!");
                    maju(255,255);
                    detect=1;
```

```
            }                                                   // end nemu api, tpi kena garis putih pintu masuk
dulu

        else if( read_adc(1) < 400 && detect ==1 ){     // CEK WARNA !!  abu terbesar
                detect=2;
            }


        else if(  (read_adc(2) > 130 && detect == 2) && juringPinggir == 0){     // CEK WARNA !!
putih terkecil, dapat juring lingkaran putih (putih>500)
                    apiPadam=1;
                    berenti(255,255);
                    kipas();
                    berenti(255,255);
                    juringPinggir=1;
                    varJuring=1;
                }
        else if (juringPinggir == 1 && varJuring == 1){
                bantingKanan(255,255);
                delay_ms(75);
                berenti(255,255);
                kipas();
                berenti(255,255);
                apiPadam=1;
                }
        else if( (read_adc(1) > 500 && detect == 2 ) && juringTengah == 0){     // CEK WARNA putih
terkecil, dapat juring lingkaran putih di tengah (putih>800)

                if( juringTengah == 1 ){                // blm mati tuw juring di tengah
                    bantingKiri(255,255);
                    delay_ms(50);
                    berenti(255,255);
                    kipas();
                    berenti(255,255);
                    apiPadam=1;
                    }
                else {
                    apiPadam=1;
                    berenti(255,255);
                    kipas();
                    berenti(255,255);
                    juringTengah=1;
                    varJuring=2;
                    }
                }

        else if (juringTengah == 1 && varJuring == 2){
                bantingKiri(255,255);
                delay_ms(75);
                berenti(255,255);
                kipas();
                berenti(255,255);
                apiPadam=1;
                }
        else {                      // else klo blm dapat juring, jalan trs nyari juring putih+klo ada api

            if( getSrf(6) > 14) {                               //14
                kp = 10;                                        //10
                kd = 15;                                        //15
                ki = 0.005;                                     //0.025
                setPointPWM =50;
                processVariable = getSrf(1);                    // proporsional program
                setPointSensorKiri = 8;
                error = setPointSensorKiri - processVariable;
```

```
                    proporsional = kp * error;                          // proporsional end
                    rate = error - lastError;                           // derivative program
                    rateInt = error + lastError;
                    Ts=1;                                               // Ts = time sampling
                    integral = Ts * (ki * rateInt);
                    derivative = (kd/Ts) * rate;                                // derivative
                    sinyalControl = proporsional + derivative + integral;  // sinyal control
                    rightPWM = setPointPWM + sinyalControl;
                    leftPWM = setPointPWM - sinyalControl;
                    if(rightPWM >= 255) {
                          rightPWM = 150;
                          }
                    if(leftPWM >= 255) {
                          leftPWM = 150;
                          }
                    if(rightPWM <= 0) {
                          rightPWM = 0;
                          }
                    if(leftPWM <= 0) {
                          leftPWM = 0;
                          }
                    leftPWM;
                    rightPWM;
                    if(leftPWM == 0 && rightPWM == 200){
                           bantingKiri(leftPWM,rightPWM);
                           lcd_clear();
                           lcd_gotoxy(0,0);
                           lcd_putsf("belok kiri");
                          }
                    if(rightPWM == 0 && leftPWM == 200){
                           bantingKanan(leftPWM,rightPWM);
                           lcd_clear();
                           lcd_gotoxy(0,0);
                           lcd_putsf("belok kanan");
                          }
                    if( (rightPWM <200 || leftPWM < 200)  && (rightPWM > 0 || leftPWM > 0) ){

                           if( (leftPWM < rightPWM) && (leftPWM < 10)){
                                belokKiri(leftPWM,rightPWM);
                                }
                           if( (leftPWM > rightPWM) && (rightPWM < 10) ){
                                belokKanan(leftPWM,rightPWM);
                                }
                           else{
                                maju(leftPWM,rightPWM);
                               }
                           lcd_clear();
                           lcd_gotoxy(0,0);
                           sprintf(text,"S.K=%3d ",processVariable);
                           lcd_puts(text);
                           lcd_gotoxy(0,1);
                           sprintf(text,"r=%3d l=%3d",rightPWM,leftPWM);
                           lcd_puts(text);
                           }                                            //end   if( (rightPWM <170 || leftPWM < 170)
&& (rightPWM > 0 || leftPWM > 0) ){

                    lastError = error;                                  //error sebelumnya
                              }
              else if(getSrf(6) <= 14){
                    if( getSrf(2) > 40 && getSrf(4) > 40){
                              bantingKanan (150,150);
                          }
                    else {
                          bantingKiri(190,190);
```

```
                        }
                    }
                }                                   // end klo dari else klo blm dapat juring, jln trs
nyari juring putih+klo ada api
            }                                       // end klo lagi ada api

//=============================================================================
    };                          //end while(1)

}
```

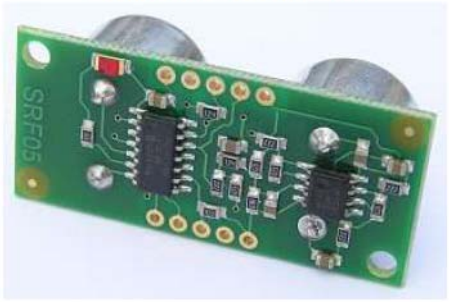
LAMPIRAN C

DATASHEET

Sensor Ultrasonik (SRF05)…….. C-1
Sensor Api (UVTron)…… C-4
Modul C3704…….. C-6
Sensor Warna (TCRT5000)…….. C-8

SRF05 - Ultra-Sonic Ranger

Technical Specification

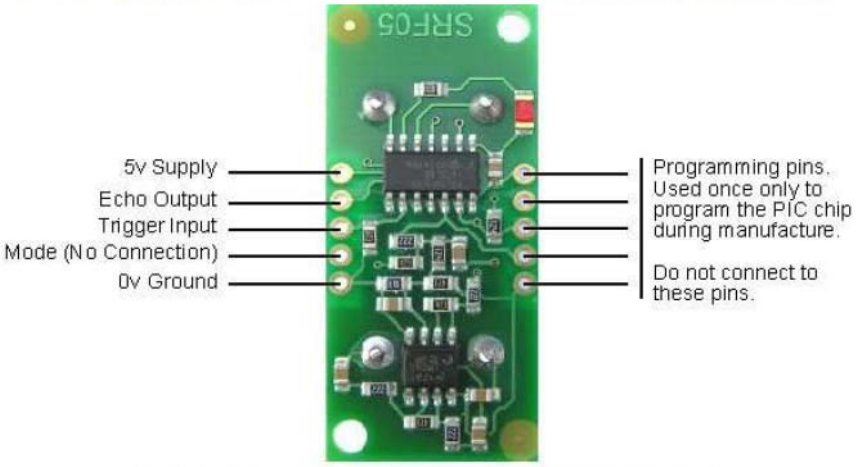


Introduction

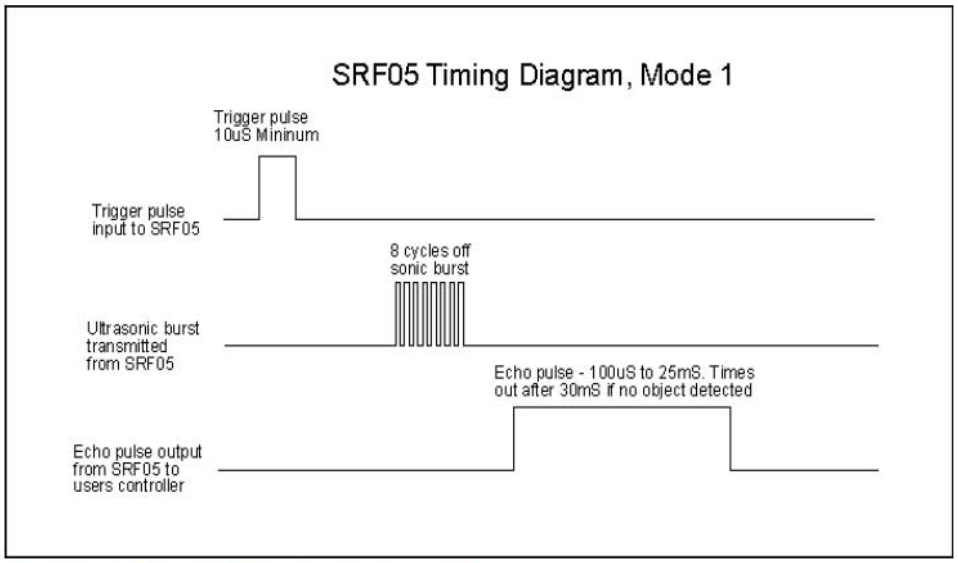
The SRF05 is an evolutionary step from the SRF04, and has been designed to increase flexibility, increase range, and to reduce costs still further. As such, the SRF05 is fully compatible with the SRF04. Range is increased from 3 meters to 4 meters. A new operating mode (tying the mode pin to ground) allows the SRF05 to use a single pin for both trigger and echo, thereby saving valuable pins on your controller. When the mode pin is left unconnected, the SRF05 operates with separate trigger and echo pins, like the SRF04. The SRF05 includes a small delay before the echo pulse to give slower controllers such as the Basic Stamp and Picaxe time to execute their pulse in commands.

Mode 1 - SRF04 compatible - Separate Trigger and Echo

This mode uses separate trigger and echo pins, and is the simplest mode to use. All code examples for the SRF04 will work for the SRF05 in this mode. To use this mode, just leave the mode pin unconnected - the SRF05 has an internal pull up resistor on this pin.

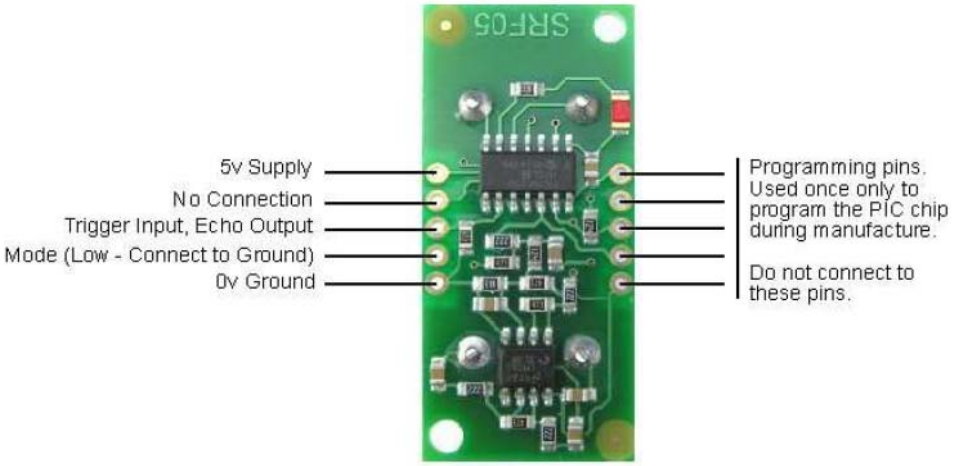


Connections for 2-pin Trigger/Echo Mode (SRF04 compatible)

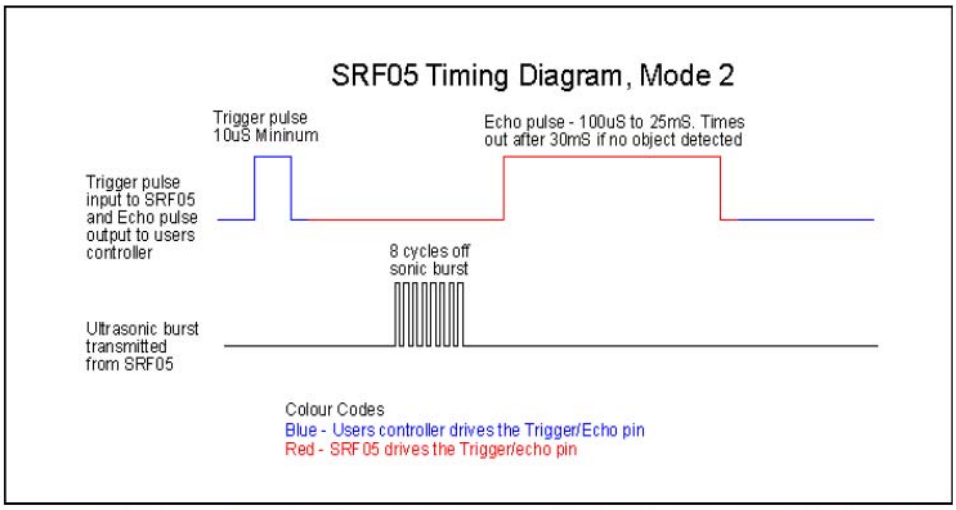


Mode 2 - Single pin for both Trigger and Echo

This mode uses a single pin for both Trigger and Echo signals, and is designed to save valuable pins on embedded controllers. To use this mode, connect the mode pin to the 0v Ground pin. The echo signal will appear on the same pin as the trigger signal. The SRF05 will not raise the echo line until 700uS after the end of the trigger signal. You have that long to turn the trigger pin around and make it an input and to have your pulse measuring code ready. The PULSIN command found on many popular controllers does this automatically.



Connections for single pin Trigger/Echo Mode



To use mode 2 with the Basic Stamp BS2, you simply use PULSOUT and PULSIN on the same pin, like this:

```
SRF05 PIN 15              ' use any pin for both trigger and echo
Range VAR Word            ' define the 16 bit range variable

SRF05 = 0                 ' start with pin low
PULSOUT SRF05, 5          ' issue 10uS trigger pulse (5 x 2uS)
PULSIN SRF05, 1, Range    ' measure echo time
Range = Range/29          ' convert to cm (divide by 74 for inches)
```

Calculating the Distance

The SRF05 Timing diagrams are shown above for each mode. You only need to supply a short 10uS pulse to the trigger input to start the ranging. The SRF05 will send out an 8 cycle burst of ultrasound at 40khz and raise its echo line high (or trigger line in mode 2). It then listens for an echo, and as soon as it detects one it lowers the echo line again. The echo line is therefore a pulse whose width is proportional to the distance to the object. By timing the pulse it is possible to calculate the range in inches/centimeters or anything else. If nothing is detected then the SRF05 will lower its echo line anyway after about 30mS.

The SRF04 provides an echo pulse proportional to distance. If the width of the pulse is measured in uS, then dividing by 58 will give you the distance in cm, or dividing by 148 will give the distance in inches. uS/58=cm or uS/148=inches.

The SRF05 can be triggered as fast as every 50mS, or 20 times each second. You should wait 50ms before the next trigger, even if the SRF05 detects a close object and the echo pulse is shorter. This is to ensure the ultrasonic "beep" has faded away and will not cause a false echo on the next ranging.

The other set of 5 pins

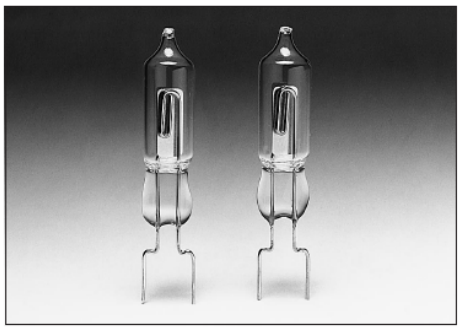
The 5 pins marked "programming pins" are used once only during manufacture to program the Flash memory on the PIC16F630 chip. The PIC16F630's programming pins are also used for other functions on the SRF05, so make sure you don't connect anything to these pins, or you will disrupt the modules operation.

HAMAMATSU

FLAME SENSOR
UV TRON® R2868

Quick Detection of Flame from Distance, Compact UV Sensor with High Sensitivity and Wide Directivity, Suitable for Flame Detectors and Fire Alarms.

Hamamatsu R2868 is a UV TRON ultraviolet detector that makes use of the photoelectric effect of metal and the gas multiplication effect. It has a narrow spectral sensitivity of 185 to 260 nm, being completely insensitive to visible light. Unlike semiconductor detectors, it does not require optical visible-cut filters, thus making it easy to use.
In spite of its small size, the R2868 has wide angular sensitivity (directivity) and can reliably and quickly detect weak ultraviolet radiations emitted from flame due to use of the metal plate cathode (eg. it can detect the flame of a cigarette lighter at a distance of more than 5 m.).
The R2868 is well suited for use in flame detectors and fire alarms, and also in detection of invisible discharge phenomena such as corona discharge of high-voltage transmission lines.



APPLICATIONS

- **Flame detectors for gas/oil lighters and matches**
- **Fire alarms**
- **Combustion monitors for burners**
- **Inspection of ultraviolet leakage**
- **Detection of discharge**
- **Ultraviolet switching**

GENERAL

Parameters	Rating	Units
Spectral Response	185 to 260	nm
Window Material	UV glass	—
Weight	Approx. 1.5	g
Dimensional Outline	See Fig. 3	—

MAXIMUM RATINGS

Parameters	Rating	Units
Supply Voltage	400	Vdc
Peak Current [1]	30	mA
Average Discharge Current [2]	1	mA
Operating Temperature	-20 to +60	°C

CHARACTERISTICS (at 25°C)

Parameters	Rating	Units
Discharge Starting Voltage (with UV radiation)	280	Vdc Max.
Recommended Operating Voltage	325±25	Vdc
Recommended Average Discharge Current	100	µA
Background [3]	10	cpm Max
Sensitivity [4]	5000	cpm Typ.

NOTES:
1) This is the maximum momentary current that can be handled if its full width at half maximum is less than 10 µs.
2) If the tube is operated near this or higher, the service life is noticeably reduced. Use the tube within the recommended current values.
3) Measured under room illuminations (approximately 500 lux) and recommended operating conditions. Note that these values may increase if the following environmental factors are present.
   1. Mercury lamps, sterilization lamps, or halogen lamps are located nearby.
   2. Direct or reflected sunlight is incident on the tube.
   3. Electrical sparks such as welding sparks are present.
   4. Radiation sources are present.
   5. High electric field (including static field) generates across the tube.
4) These are representative values for a wavelength of 200 nm and a light input of 10 pW/cm². In actual use, the sensitivity will vary with the wavelength of the ultraviolet radiation and the drive circuitry employed.

Figure 1: UV TRON's Spectral Response and Various Light Sources

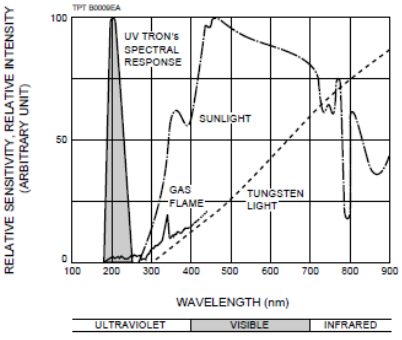


Figure 2: Angular Sensitivity (Directivity)

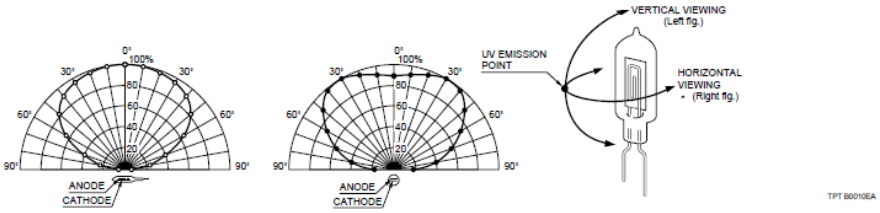


Figure 3: Dimensional Outline (Unit: mm)

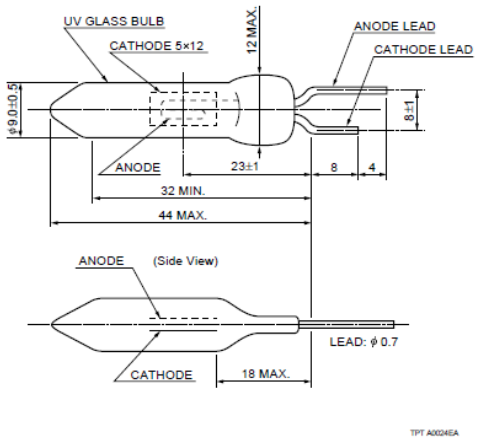
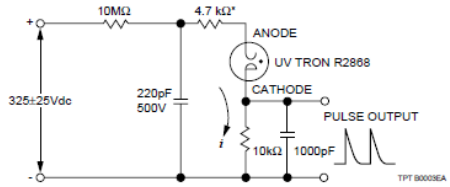
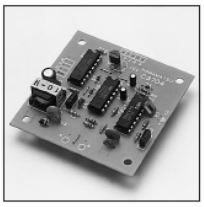


Figure 4: Recommended Operating Circuit



* Be sure to connect the 4.7 kΩ resistor within 2.5 cm from the anode lead end of UV TRON.

- UV TRON Driving Circuit C3704 series (Option)



Hamamatsu also provide the driving circuit C3704 series for R2868 operation.
C3704 series include a high voltage power supply and a signal processing circuit in printed circuit board, which allows to operate R2868 easily as a flame sensor with the low input voltage (DC 6 to 30 V) only.
For the details, please refer to the datasheet of C3704 series.

PRECAUTIONS FOR USE

- **Ultraviolet Radiation**

The UV TRON itself emits ultraviolet radiation in operation. When using two or more UV TRONs at the same time in close position, care should be taken so that they do not optically interfere with each other.

- **Vibration and Shock**

The UV TRON is designed in accordance with the standards of MIL-STD-202F (Method 204D/0.06 inch or 10g, 10- 500Hz, 15 minutes, 1 cycle) and MIL-STD-202F (Method 213B/100g, 11ms, Half-sine, 3 times). However, should a strong shock be sustained by the UV TRON (e.g. if dropped), the glass bulb may crack or the internal electrode may be deformed, resulting in deterioration of electrical characteristics. So extreme care should be taken in handling the tube.

- **Polarity**

Connect the UV TRON with correct polarity. Should it be connected with reverse polarity, operating errors may occur.

WARRANTY.

The UV TRON is covered by a warranty for a period of one year after delivery. The warranty is limited to replacement of any defective tube due to defects traceable to the manufacturer.

HAMAMATSU
HAMAMATSU PHOTONICS K.K., Electron Tube Center
314-5, Shimokanzo, Toyooka-village, Iwata-gun, Shizuoka-ken, 438-0193, Japan, Telephone: (81)539/62-5248, Fax: (81)539/62-2205,
U.S.A.: Hamamatsu Corporation: 360 Foothill Road, Bridgewater, N.J. 08807-0910, U.S.A., Telephone: (1)908-231-0960, Fax: (1)908-231-1218
Germany: Hamamatsu Photonics Deutschland GmbH: Arzbergerstr. 10, D-82211 Herrsching am Ammersee, Germany, Telephone: (49)8152-375-0, Fax: (49)8152-2658
France: Hamamatsu Photonics France S.A.R.L.: 8, Rue du Saule Trapu, Parc du Moulin de Massy, 91882 Massy Cedex, France, Telephone: (33)1 69 53 71 00, Fax: (33)1 69 53 71 10
United Kingdom: Hamamatsu Photonics UK Limited: Lough Point, 2 Gladbeck Way, Windmill Hill, Enfield, Middlesex EN2 7JA, United Kingdom, Telephone: (44)181-367-3560, Fax: (44)181-367-6384
North Europe: Hamamatsu Photonics Norden AB: Färögatan 7, S-164-40 Kista, Sweden, Telephone: (46)8-703-29-50, Fax: (46)8-750-58-95
Italy: Hamamatsu Photonics Italia S.R.L.: Via Della Mola, 1/E, 20020 Arese, (Milano), Italy, Telephone: (39)2-935 81 733, Fax: (39)2-935 81 741

TPT 1008E01
MAR.1998 CR
Created in Japan

HAMAMATSU

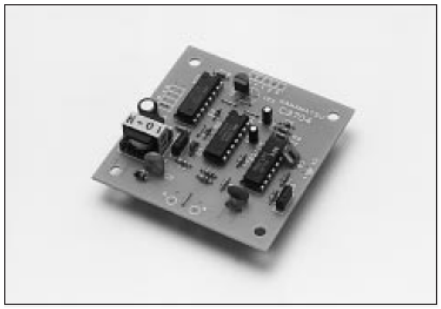
UV TRON® DRIVING CIRCUIT C3704 SERIES

Compact, Lightweight, Low Current Consumption, Low Cost Operates as High Sensitivity UV Sensor with UV TRON Suitable for Flame Detectors and Fire Alarms

Hamamatsu C3704 series UV TRON driving circuits are low current consuming, signal processing circuits for the UV TRON, well known as a high sensitivity ultraviolet detecting tube. The C3704 series can be operated as a UV sensor by connecting the UV TRON and applying DC low voltage, as they have both a high-voltage power supply and a signal processing circuit on the same printed circuit board.

Since background discharges of the UV TRON caused by natural excitation lights (such as a cosmic ray, scattered sunlight, etc.) can be cancelled in the signal processing circuit, the output signals from the C3704 series can be used without errors.

When the high sensitivity sensor "UV TRON R2868" (sold separately) is used, the flame from a cigarette lighter (flame length: 25mm) can be detected even from a distance of more than 5m.



APPLICATIONS

- **Flame detectors for gas and oil lighters**
- **Fire alarms**
- **Combustion monitors for burners**
- **Electric spark detector**
- **UV photoelectric counter**

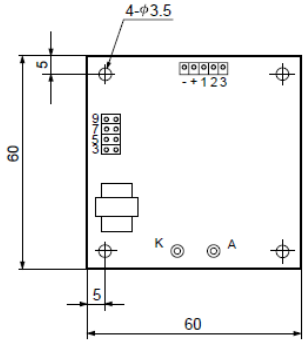
SPECIFICATIONS

Dimensional outline ..... Figure 1
Weight ..... Approx. 20g
Output signal ..... Open collector Output (50 V, 100 mA Max.)
10 ms width pulse output (Note : 1)
UV TRON supply voltage ..... DC 350 V (Note : 2)
Quenching time ..... Approx. 50 ms
Operating temperature ..... -10 to +50°C
(with no condensation)
Suitable UV TRON ..... Low voltage operation UV TRON
(such as R2868)

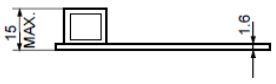
	C3704	C3704-02	C3704-03
Input Voltage	10 to 30 Vdc	5Vdc ± 5%	6 to 9 Vdc
Current consumption	3 mA Max.	300µA Max.	300µA Max.

Note 1: The output pulse width can be extended up to about 100s by adding a capacitor to the circuit board.
Note 2: Since the output impedance of this power supply is extremely high, an ordinary voltmeter cannot be used. Use a voltmeter that has an input impedance of more than 10 GΩ.

Figure 1: Dimensional Outline (Unit : mm)



<Top View>



<Side View>

TPT A0024EA

Figure 2: Schematic Diagram

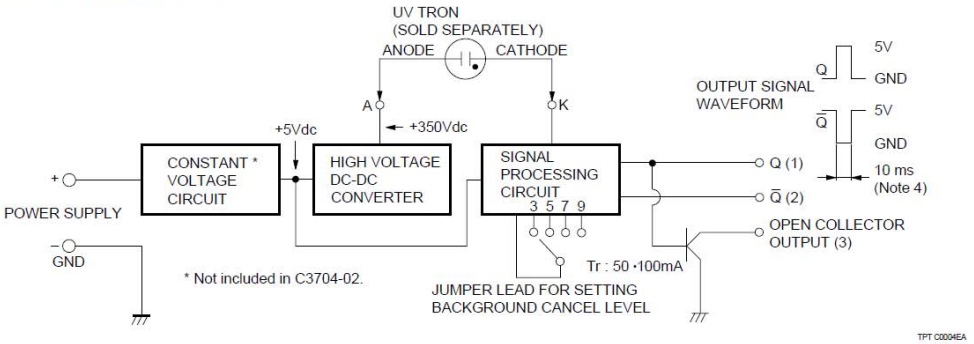
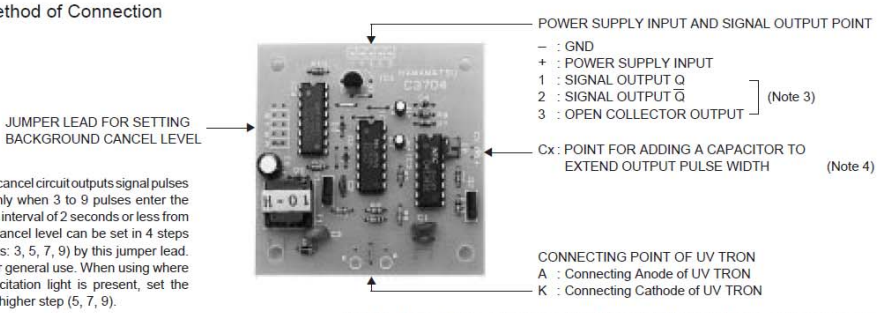


Figure 3: Method of Connection



This background cancel circuit outputs signal pulses of 10ms width only when 3 to 9 pulses enter the circuit with a time interval of 2 seconds or less from the UV TRON. Cancel level can be set in 4 steps (number of pulses: 3, 5, 7, 9) by this jumper lead. Connect to "3" for general use. When using where much natural excitation light is present, set the cancel level at a higher step (5, 7, 9).

Note 3: No load can be driven by an output from points "1" and "2" because these signals are output from the only C-MOS IC directly. When a load such as a buzzer and a relay is connected to this circuit, it should be connected to the point open collector output. The transistor ratings of the open collector is 50V, 100mA. Be careful not to exceed the ratings.

Note 4: The output pulse width is set to 10ms at shipping. If the pulse width needs to be extended, add a capacitor to this point. (When using an electrolytic condenser, make sure the polarity is correct.)
e.g. CX = 1 µF: Pulse Width ≃ 1s, CX=10 µF : Pulse Width ≃ 10s

PRECAUTIONS FOR USE

- Since the operation impedance is extremely high, the UV TRON should be connected as close as possible to the circuit board within 5 cm.
- Take care to avoid external noise since a C-MOS IC is used in the circuit. It is recommended that the whole PC board be put in the shield box when it is used.
- To reduce current consumption, oscillating frequency is very low (approx. 20 Hz) in this DC-DC converter. Thus, the output impedance of the high voltage power supply is extremely high. If the surrounding humidity is high, electrical leakage on the PC board surface may lead to a drop in the supply voltage to the UV TRON. This voltage drop may result in lowered detection performance, so a moistureproof material (silicone compound, etc.) should be applied at the connecting point of the UV TRON, etc., if using the unit in a humid environment.

- A model equipped with a flame sensor (R2868) is also available.

HAMAMATSU

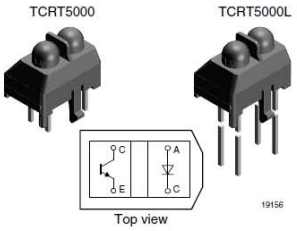
HAMAMATSU PHOTONICS K.K., Electron Tube Center
314-5, Shimokanzo, Toyooka-village, Iwata-gun, Shizuoka-ken, 438-0193, Japan, Telephone: (81)539/62-5248, Fax: (81)539/62-2205, Telex: 4225-186HAMAHQ
U.S.A.: Hamamatsu Corporation: 360 Foothill Road, Bridgewater, N.J. 08807-0910, U.S.A., Telephone: (1)908-231-0960, Fax: (1)908-231-1218
Germany: Hamamatsu Photonics Deutschland GmbH: Arzbergerstr. 10, D-82211 Herrsching am Ammersee, Germany, Telephone: (49)8152-375-0, Fax: (49)8152-2658
France: Hamamatsu Photonics France S.A.R.L.: 8, Rue du Saule Trapu, Parc du Moulin de Massy, 91882 Massy Cedex, France, Telephone: (33)1 69 53 71 00, Fax: (33)1 69 53 71 10
United Kingdom: Hamamatsu Photonics UK Limited: Lough Point, 2 Gladbeck Way, Windmill Hill, Enfield, Middlesex EN2 7JA, United Kingdom, Telephone: (44)181-367-3560, Fax: (44)181-367-6384
North Europe: Hamamatsu Photonics Norden AB: Färögatan 7, S-164-40 Kista, Sweden, Telephone: (46)8-703-29-50, Fax: (46)8-750-58-95
Italy: Hamamatsu Photonics Italia S.R.L.: Via Della Moia, 1/E 20020 Arese, (Milano), Italy, Telephone: (39)2-935 81 733, Fax: (39)2-935 81 741

TPT1007E01
JUL.1997 CR
Created in Japan

Reflective Optical Sensor with Transistor Output

Description

The TCRT5000 and TCRT500L are reflective sensors which include an infrared emitter and phototransistor in a leaded package which blocks visible light. The package includes two mounting clips. TCRT5000L is the long lead version.



Features

- Package type: Leaded
- Detector type: Phototransistor
- Dimensions:
  L 10.2 mm x W 5.8 mm x H 7.0 mm
- Peak operating distance: 2.5 mm
- Operating range: 0.2 mm to 15 mm
- Typical output current under test: I_C = 1 mA
- Daylight blocking filter
- Emitter wavelength 950 nm
- Lead (Pb)-free soldering released
- Lead (Pb)-free component in accordance to RoHS 2002/95/EC and WEEE 2002/96/EC

Applications

- Position sensor for shaft encoder
- Detection of reflective material such as paper, IBM cards, magnetic tapes etc.
- Limit switch for mechanical motions in VCR
- General purpose - wherever the space is limited

Order Instructions

Part Number	Remarks	Minimum Order Quantity
TCRT5000	3.5 mm lead length	4500 pcs, 50 pcs/tube
TCRT5000L	15 mm lead length	2400 pcs, 48 pcs/tube

Absolute Maximum Ratings

T_{amb} = 25 °C, unless otherwise specified

Input (Emitter)

Parameter	Test condition	Symbol	Value	Unit
Reverse voltage		V_R	5	V
Forward current		I_F	60	mA
Forward surge current	$t_p \leq 10$ µs	I_{FSM}	3	A
Power dissipation	$T_{amb} \leq 25$ °C	P_V	100	mW
Junction temperature		T_j	100	°C

Output (Detector)

Parameter	Test condition	Symbol	Value	Unit
Collector emitter voltage		V_{CEO}	70	V
Emitter collector voltage		V_{ECO}	5	V
Collector current		I_C	100	mA
Power dissipation	$T_{amb} \leq 55$ °C	P_V	100	mW
Junction temperature		T_j	100	°C

Sensor

Parameter	Test condition	Symbol	Value	Unit
Total power dissipation	$T_{amb} \leq 25$ °C	P_{tot}	200	mW
Operation temperature range		T_{amb}	- 25 to + 85	°C
Storage temperature range		T_{stg}	- 25 to + 100	°C
Soldering temperature	2 mm from case, t ≤ 10 s	T_{sd}	260	°C

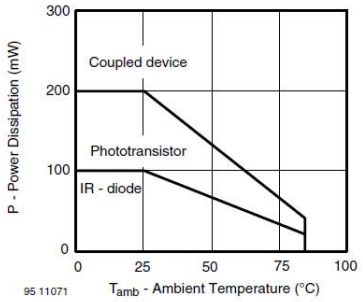


Figure 1. Power Dissipation Limit vs. Ambient Temperature

Electrical Characteristics

T_{amb} = 25 °C, unless otherwise specified

Input (Emitter)

Parameter	Test condition	Symbol	Min	Typ.	Max	Unit
Forward voltage	I_F = 60 mA	V_F		1.25	1.5	V
Junction capacitance	V_R = 0 V, f = 1 MHz	C_j		17		pF
Radiant intensity	I_F = 60 mA, t_P = 20 ms	I_E			21	mW/sr
Peak wavelength	I_F = 100 mA	λ_P	940			nm
Virtual source diameter	Method: 63 % encircled energy	Ø		2.1		mm

Output (Detector)

Parameter	Test condition	Symbol	Min	Typ.	Max	Unit
Collector emitter voltage	I_C = 1 mA	V_{CEO}	70			V
Emitter collector voltage	I_E = 100 µA	V_{ECO}	7			V
Collector dark current	V_{CE} = 20 V, I_F = 0, E = 0	I_{CEO}		10	200	nA

Sensor

Parameter	Test condition	Symbol	Min	Typ.	Max	Unit
Collector current	V_{CE} = 5 V, I_F = 10 mA, D = 12 mm	I_C [1,2)]	0.5	1	2.1	mA
Collector emitter saturation voltage	I_F = 10 mA, I_C = 0.1 mA, D = 12 mm	V_{CEsat} [1,2)]			0.4	V

1) See figure 3

2) Test surface: Mirror (Mfr. Spindler a. Hoyer, Part No 340005)

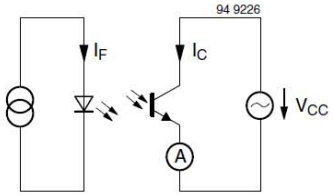


Figure 2. Test Circuit

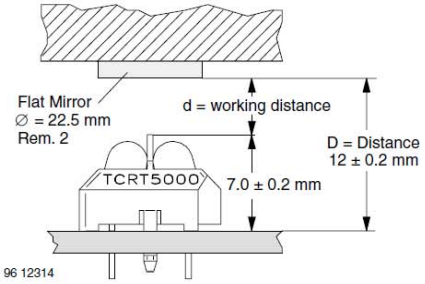


Figure 3. Test Circuit

Typical Characteristics

T_{amb} = 25 °C, unless otherwise specified

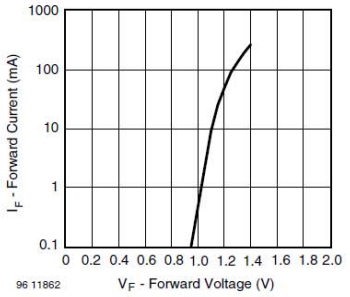


Figure 4. Forward Current vs. Forward Voltage

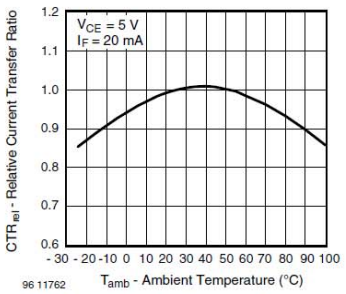


Figure 5. Relative Current Transfer Ratio vs. Ambient Temperature

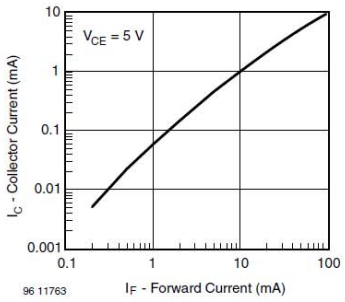


Figure 6. Collector Current vs. Forward Current

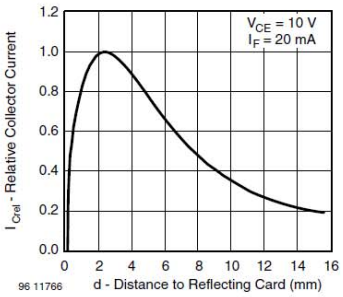


Figure 9. Relative Collector Current vs. Distance

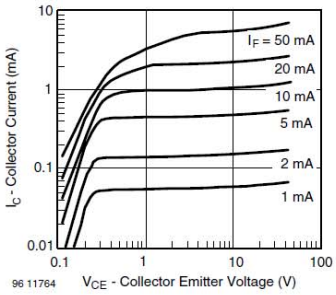


Figure 7. Collector Emitter Saturation Voltage vs. Collector Current

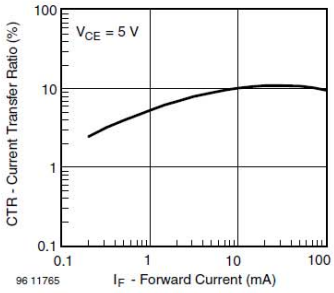
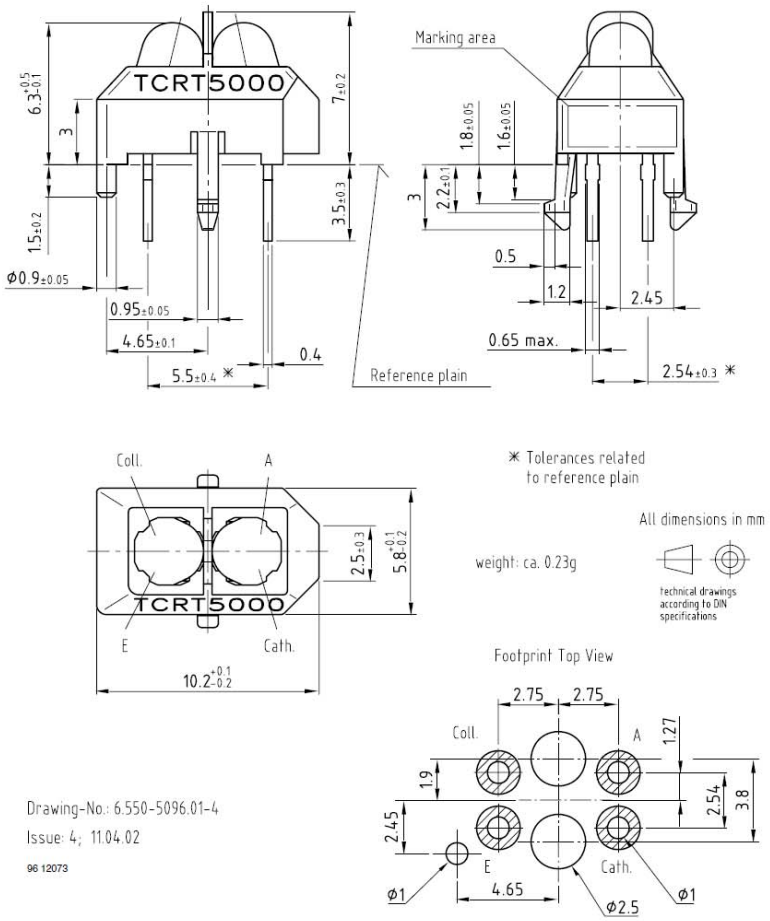
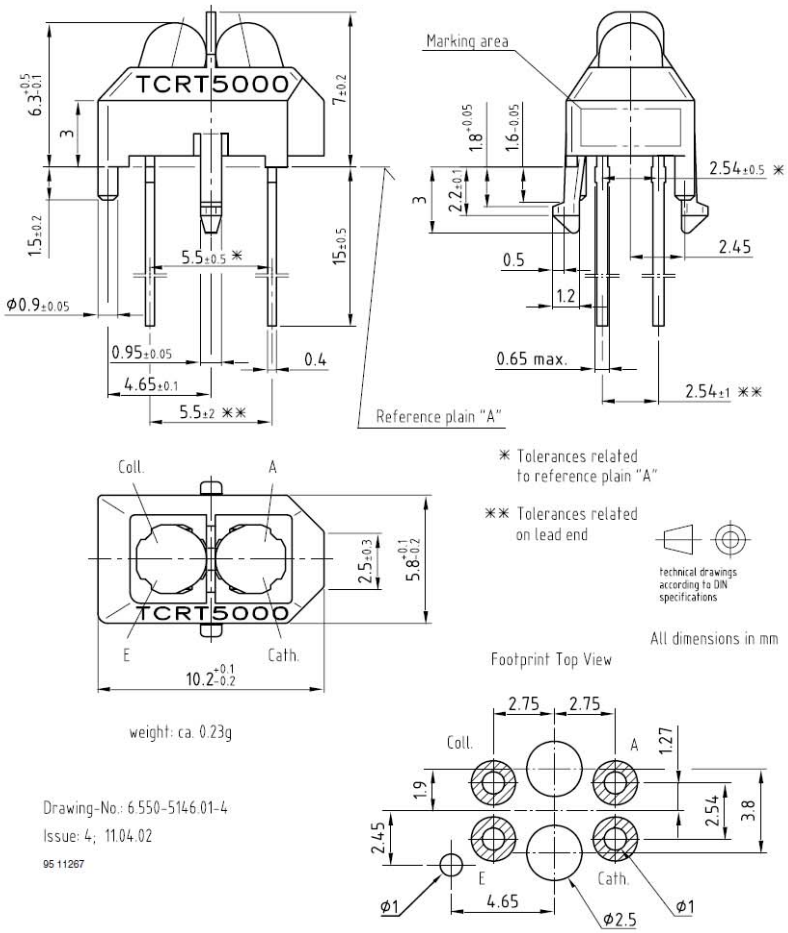


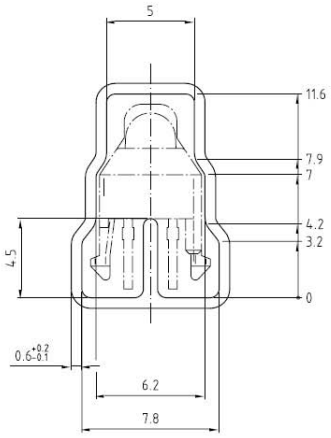
Figure 8. Current Transfer Ratio vs. Forward Current

Package Dimensions in mm




TCRT5000
Marking area
Reference plain "A"
* Tolerances related to reference plain "A"
** Tolerances related on lead end
technical drawings according to DIN specifications
All dimensions in mm
Coll.
A
E
Cath.
weight: ca. 0.23g
Footprint Top View
Drawing-No.: 6.550-5146.01-4
Issue: 4; 11.04.02
95 11267

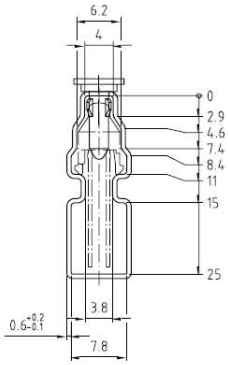
TCRT5000, Tube Dimensions



With rubber stopper
Tolerance: ±0.5mm
Length: 575±1mm
All dimensions in mm

Drawing-No.: 9.700-5139.01-4
Issue: 1; 10.05.00
20298

TCRT5000L, Tube Dimensions



With stopper pins
Tolerance: ±0.5mm
Length: 575±1mm

All dimensions in mm

Drawing-No.: 9.700-5178.01-4
Issue: 1; 25.02.00
20299