

BAB II

TINJAUAN PUSTAKA

2.1 Robot Line Follower

Line Follower Robot adalah sebuah alat yang dapat berjalan secara otomatis mengikuti garis berdasarkan perubahan warna pada garis baik hitam dan putih. Hasil dari perubahan warna tersebut menyebabkan nilai pada photo diode berubah sehingga menyebabkan nilai yang masuk ke dalam port ADC pada mikrokontroller berubah, dan nilai ADC tersebut yang akan kita olah menjadi sebuah input.

Line Follower ini mempunyai dua buah motor DC 6 Volt yang dapat digerakkan maju dan mundur dengan menggunakan *driver motor* L293D, saat sensor mendeteksi adanya garis hitam ditengah dari wilayah sensor maka kedua motor akan berjalan searah jarum jam sehingga robot maju, dan ketika sensor mendeteksi adanya garis hitam dipinggir wilayah sensor maka salah satu motor akan berputar searah jarum jam dan yang satu berlawanan arah jarum jam sehingga robot akan bergerak ke arah kanan atau kiri. Dan ketika sensor tidak mendeteksi adanya garis pada wilayah sensor maka robot akan berjalan mundur.

Robot ini dapat dikembangkan lagi menjadi sebuah aplikasi yang berguna baik di masyarakat maupun di industri, contohnya sebagai sebuah alat pengantar barang secara otomatis atau terprogram. (**Budiharto Widodo, 2014 : 10 mei 2016**)

Ada 2 macam robot line follower yaitu Analog dan Mikrokontroller(Digital). Jika analog menggunakan fungsi-fungsi logika pada IC TTL tapi pada robot mikrokontroller dengan menggunakan program yang dibuat pada software komputer lalu di kirim ke dalam IC mikrokontroller. (**baniexperience, 2013**)



Gambar 2.1 Robot Line Follower Analog

(baniexperience, 2013)



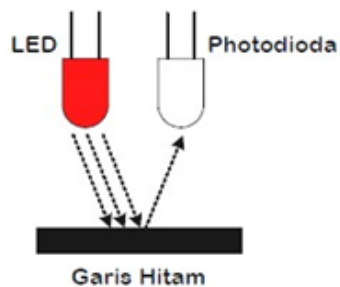
Gambar 2.2 Robot Line Follower Digital

(baniexperience, 2013)

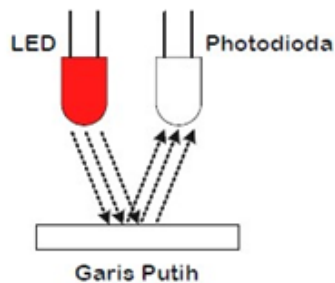
2.2 Sensor

Yang dimaksud sensor garis disini adalah suatu perangkat/alat yang digunakan untuk mendeteksi adanya sebuah garis atau tidak. Garis yang dimaksud adalah garis berwarna hitam diatas permukaan berwarna putih ataupun juga sebaliknya.

Sensor bisa kita buat sendiri. Prinsip kerjanya sederhana, hanya memanfaatkan sifat cahaya yang akan dipantulkan jika mengenai benda berwarna terang dan akan diserap jika mengenai benda berwarna gelap. Sebagai sumber cahaya kita gunakan LED (*Light Emitting Diode*) yang akan memancarkan cahaya merah. Dan untuk menangkap pantulan cahaya LED, kita gunakan photodiode. Jika sensor berada diatas garis hitam maka photodiode akan menerima sedikit sekali cahaya pantulan. Tetapi jika sensor berada diatas garis putih maka photodiode akan menerima banyak cahaya pantulan. Berikut adalah ilustrasinya :



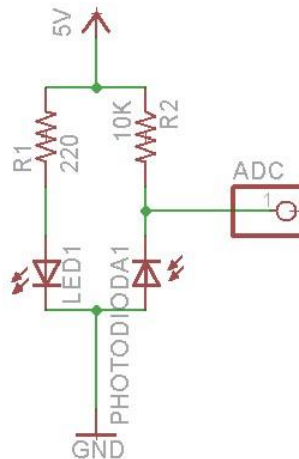
Gambar 2.3 Cahaya pantulan sedikit
(Taufiq Dwi septian suyadhi, 2008)



Gambar 2.4 Cahaya pantulan banyak
(Taufiq Dwi septian suyadhi, 2008)

Sifat dari photodiode adalah jika semakin banyak cahaya yang diterima, maka nilai resistansi diodanya semakin kecil. Dengan melakukan sedikit modifikasi, maka

besaran resistansi tersebut dapat diubah menjadi tegangan. Sehingga jika sensor berada diatas garis hitam, maka tegangan keluaran sensor akan kecil, demikian pula sebaliknya. Berikut adalah gambar rangkaian sensor proximity yang digunakan pada robot ini :



Gambar 2.5 Rangkaian Sensor

(Taufiq Dwi septian suyadhi, 2008)

Agar dapat dibaca oleh mikrokontroler, maka tegangan sensor harus disesuaikan dengan level tegangan TTL yaitu 0 – 1 volt untuk logika 0 dan 3 – 5 volt untuk logika 1. Output dari photodiode yang masuk ke pin ADC pada mikrokontroler, kemudian mikrokontroler akan mengubah nilai tegangan yang masuk ke pin tersebut menjadi sebuah nilai. Dari nilai tersebut terdapat range yang menunjukkan apakah sensor membaca garis hitam atau putih. (**Taufiq Dwi septian suyadhi, 2008**)

2.3 Mikrokontroller

2.3.1 Pengertian Mikrokontroller

Selama 40 tahun sejak pertama kali diperkenalkan, mikrokontroller telah mengalami banyak perkembangan. Berbagai teknologi, fungsi, serta periferai yang diterapkan pada komponen ini menjadikan mikrokontroller yang saat ini beredar memiliki banyak variasi. Sudah tak terhitung pula aplikasi yang dibuat menggunakan mikrokontroller, mulai dari untuk kehidupan sehari-hari hingga skala industri.

Mikrokontroller adalah komputer mikro dalam satu chip tunggal. Mikrokontroller memadukan CPU, ROM, RWM, I/O paralel, I/O seri, *counter-timer*, dan rangkaian *clock* dalam satu chip. Dengan kata lain, mikrokontroller adalah suatu alat elektronika digital yang mempunyai masukan dan keluaran serta kendali dengan program yang bisa ditulis dan dihapus dengan cara khusus. Cara kerja mikrokontroller sebenarnya membaca dan menulis data. (Kaka, 2011)

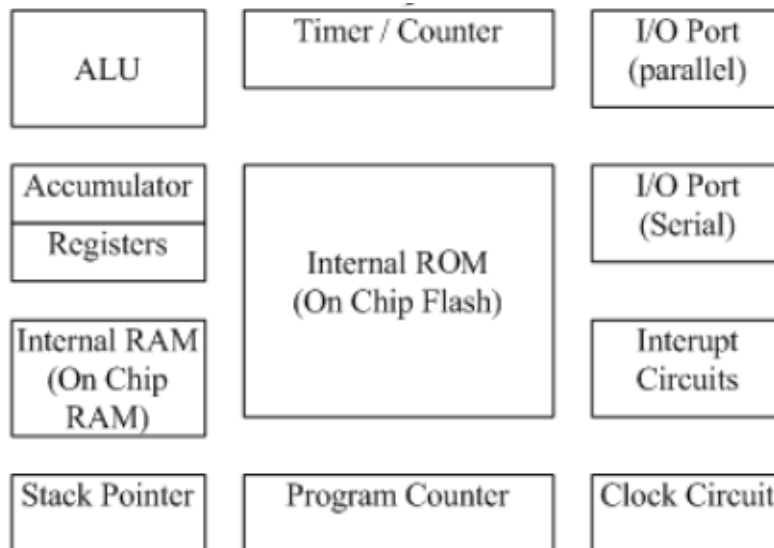
Sama halnya dengan *mikroprocessor*, mikrokontroller adalah piranti yang dirancang untuk kebutuhan umum. Fungsi utama dari mikrokontroller adalah mengontrol kerja mesin atau sistem menggunakan program yang disimpan pada sebuah ROM. Mikrokontroller merupakan komputer didalam chip yang digunakan untuk mengontrol peralatan elektronik, yang menekankan efisiensi dan efektifitas biaya. Secara harfiah dapat disebut sebagai “pengendali kecil” dimana sebuah sistem elektronik yang sebelumnya banyak memerlukan komponen-komponen pendukung seperti IC TTL dan CMOS dapat direduksi/diperkecil dan akhirnya terpusat serta dikendalikan oleh mikrokontroler ini.

Mikrokonktroller digunakan dalam produk dan alat yang dikendalikan secara otomatis, seperti sistem kontrol mesin, *remote control*, mesin kantor, peralatan rumah tangga, alat berat, dan mainan. Dengan mengurangi ukuran, biaya, dan konsumsi tenaga dibandingkan desain menggunakan mikroprosesor memori dan alat input output yang terpisah, kehadiran *mikrokontroller* membuat kontrol elektrik untuk berbagai proses menjadi lebih ekonomis. Dengan penggunaan *mikrokontroller* ini maka:

1. Sistem elektronik akan menjadi lebih ringkas,
2. Rancang bangun sistem elektronik dapat dilakukan lebih cepat karena sebagian besar sistem merupakan perangkat lunak yang mudah dimodifikasi,
3. Gangguan yang terjadi lebih mudah ditelusuri karena sistemnya yang kompak.

Mikrokontroler tidak sepenuhnya dapat mereduksi komponen IC TTL dan CMOS yang seringkali masih diperlukan untuk aplikasi kecepatan tinggi atau sekedar menambah jumlah saluran masukan dan keluaran (I/O). Dengan kata lain, mikrokontroler adalah versi mini atau mikro dari sebuah komputer karena mikrokontroler telah mengandung beberapa periferal yang langsung bisa dimanfaatkan, misalnya port paralel, port serial, komparator, konversi digital ke analog (DAC), konversi analog ke digital dan sebagainya hanya menggunakan sistem minimum yang sederhana.

Agar sebuah mikrokontroler dapat berfungsi, maka mikrokontroler tersebut memerlukan komponen eksternal yang kemudian disebut dengan sistem minimum. Untuk membuat sistem minimum paling tidak dibutuhkan sistem *clock* dan *reset*, walaupun pada beberapa mikrokontroler sudah menyediakan sistem *clock* internal, sehingga tanpa rangkaian eksternal pun mikrokontroler dapat beroperasi. Untuk merancang sebuah sistem berbasis mikrokontroler, kita memerlukan perangkat keras dan perangkat lunak, yaitu sistem minimum mikrokontroler, *software* pemrograman dan *compiler*, serta *downloader*.



Gambar 2.6 Blok Diagram Mikrokontroler Secara Umum

(Sumber : Kaka, 2011)

Dalam gambar 2.6 terlihat bahwa sebuah mikrokontroler terdiri dari beberapa bagian. Bagian-bagian tersebut saling dihubungkan dengan internal dan pada umumnya terdiri dari 3 macam bus yaitu *address bus*, *data bus* dan *control bus*. Masing-masing bagian memiliki fungsi-fungsi sebagai berikut:

1. Register

Register merupakan suatu tempat penyimpanan (variabel) bilangan bulat yang terdiri dari 8 atau 16 bit. Pada umumnya register memiliki jumlah yang banyak, masing-masing ada yang memiliki fungsi khusus dan ada pula yang memiliki fungsi atau kegunaan secara umum. Register yang memiliki fungsi secara khusus misalnya register *timer* yang berisi data penghitungan pulsa untuk *timer*, atau register pengatur mode operasi *counter* (penghitung pulsa). Sedangkan register yang memiliki fungsi umum digunakan untuk menyimpan data sementara yang diperlukan untuk proses penghitungan dan proses operasi mikrokontroler. Register dengan fungsi umum sangat dibutuhkan dalam sistem mikrokontroler karena mikrokontroler hanya mampu melakukan operasi aritmetik atau logika hanya pada satu atau dua operasi

saja, sehingga untuk operasi-operasi yang melibatkan banyak variabel harus dimanipulasi dengan menggunakan variabel-variabel register umum.

2. *Accumulator*

Accumulator merupakan salah satu register khusus yang berfungsi sebagai operasi umum proses aritmetika dan logika.

3. *Program Counter*

Program counter merupakan salah satu register khusus yang berfungsi sebagai pencacah atau penghitung eksekusi program mikrokontroler.

4. *ALU (Arithmetic and Logic Unit)*

ALU memiliki kemampuan dalam mengerjakan proses-proses aritmatika (penjumlahan, pengurangan, perkalian, pembagian) dan operasi logika (AND, OR, XOR, NOT) terhadap bilangan bulat 8 atau 16 bit.

5. *Clock Circuits*

Mikrokontroler merupakan rangkaian logika sekuensial, dimana proses kerjanya berjalan melalui sinkronisasi *clock*. Oleh karena itu, diperlukan *clock circuits* yang menyediakan *clock* untuk seluruh bagian rangkaian.

6. *Internal ROM (Read Only Memory)*

Internal ROM (*Read Only Memory*) merupakan memori penyimpanan data dimana data tersebut tidak dapat diubah atau dihapus (hanya dapat dibaca). ROM biasanya diisi dengan program untuk dijalankan oleh mikrokontroler segera setelah power dihidupkan. Data dalam ROM tidak dapat hilang meskipun power dimatikan.

7. *Stack Pointer*

Stack pointer merupakan bagian dari RAM yang memiliki metode penyimpanan dan pengambilan data secara khusus. Data yang disimpan dan dibaca tidak dapat dilakukan dengan cara acak karena data yang dituliskan ke dalam *stack* yang berada pada urutan yang terakhir merupakan data yang pertama kali dibaca kembali. *Stack pointer* berisi *offset* dimana posisi data *stack* yang terakhir masuk atau yang pertama kali dapat diambil.

8. *I/O (Input/Output) Ports*

I/O (Input/Output) ports merupakan sarana yang digunakan oleh mikrokontroler untuk mengakses peralatan-peralatan lain di luar sistem. *I/O port* berupa pin-pin yang dapat berfungsi untuk mengeluarkan data digital ataupun sebagai masukan data eksternal.

9. *Interrupt Circuits*

Interrupt circuits merupakan rangkaian yang memiliki fungsi untuk mengendalikan sinyal-sinyal interupsi baik internal maupun eksternal. Adanya sinyal interupsi akan menghentikan eksekusi normal program mikrokontroler untuk selanjutnya menjalankan sub-program untuk melayani interupsi tersebut.

Diagram blok di atas tidak selalu sama untuk setiap jenis mikrokontroler. Beberapa mikrokontroler menyertakan rangkaian ADC (*Analog to Digital Converter*) di dalamnya, ada pula yang menyertakan *port I/O* serial disamping *port I/O* paralel yang sudah ada.

10. *Internal RAM (Random Acces Memory)*

Internal RAM (*Random Acces Memory*) merupakan memori penyimpan data dimana data tersebut dapat diubah atau dihapus. RAM biasanya berisi data-data variabel dan register. Data yang tersimpan pada RAM bersifat *volatile* yaitu akan hilang bila catu daya yang terhubung padanya dimatikan.

2.3.2 Jenis-Jenis Mikrokontroler

Banyak jenis mikrokontroler yang bisa digunakan pada robot line follower, beberapa contoh diantaranya adalah AT89C2051/89S52 (8051 Core), ATmega 8535 (AVR Core), ATmega16 (AVR Core) dan masih banyak lagi. Pada mikrokontroler, program akan dimasukkan sehingga robot mampu mengatur kecepatan rotasi masing-masing motor dan mampu melakukan gerakan seperti yang diinginkan. Karena kecepatan robot line follower cukup tinggi, maka beberapa algoritma kontrol perlu diterapkan agar robot mampu berjalan mulus. Kontrol itu bisa berupa continuous control, PID, fuzzy logic, atau yang lainnya.

Pengaturan kecepatan ini penting terutama jika menghadapi pergantian lintasan, dari lintasan lurus ke tikungan atau sebaliknya dari tikungan ke lintasan lurus. Seperti halnya ketika robot bergerak cepat kemudian menemui tikungan, maka tentu robot akan terpelanting. Untuk itu dibutuhkan rangkaian pengatur kecepatan motor yang dinamis tergantung dari jenis lintasan yang dilalui. Jika robot berjalan lurus, kecepatan robot diusahakan pada tingkat yang maksimal. Jika dalam kondisi tikungan, maka kecepatan dikurangi bergantung pada tingkat ketajaman tikungan. Pada intinya, kecepatan dari robot dibuat fleksibel menurut situasi yang ada dilapangan. Pada robot, pengurangan kecepatan dapat dilakukan dengan menggunakan PWM (Pulse Width Modulation) controller, yaitu pengurangan kecepatan dengan cara mengurangi arus ke motor. (Taufiq Dwi septian suyadhi, 2008 : 75 diakses 25 Mei 2016)

2.3.2.1 ATmega 16

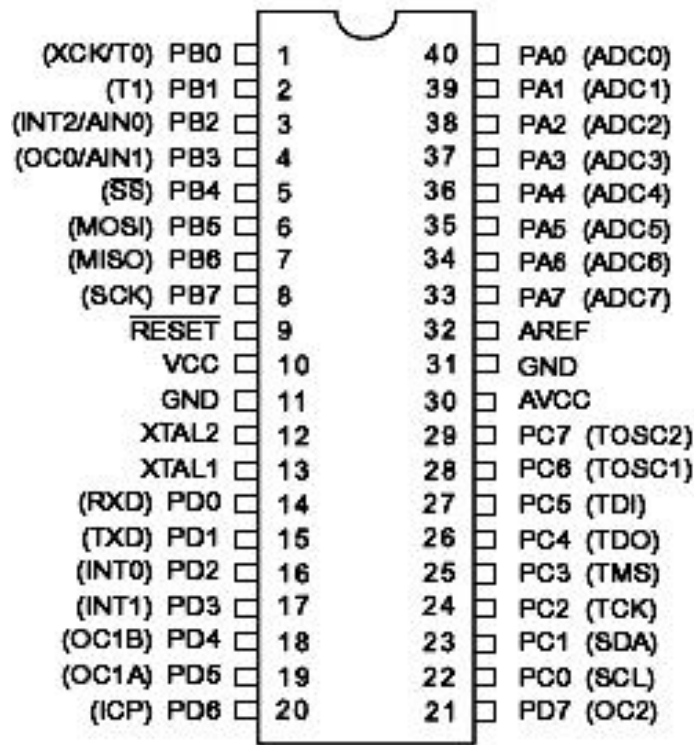
ATmega 16 adalah IC *low-power* yang dibuat berdasarkan arsitektur dari RISC. Dengan memberikan perintah yang tepat dalam satu *single clock cycle*, ATMEGA 16 dapat merespon perintah tersebut 1 MIPS per MHz untuk mengoptimalkan konsumsi tegangan.

Pin-pin pada ATmega16 dengan kemasan 40-pin DIP (*dual in-line package*) ditunjukkan oleh gambar guna memaksimalkan performa, AVR menggunakan

arsitektur *Harvard* (dengan memori dan bus terpisah untuk program dan data).

Berikut beberapa keistimewaan ATmega 16 yaitu :

1. Saluran *Input/Output* (I/O) ada 32 buah, yaitu PORTA, PORTB, PORTC, PORTD
2. ADC / *Analog to Digital Converter* 10 bit sebanyak 8 channel pada PORTA
3. 2 buah *timer/counter* 8-bit dan 1 buah *timer/counter* 16-bit dengan *prescalers* dan kemampuan pembandingan
4. *Watchdog timer* dengan osilator internal
5. Tegangan operasi 2,75 - 5,5 V pada ATmega16L dan 4,5 - 5,5 V pada ATmega16
6. EEPROM sebesar 512 byte yang dapat diprogram saat operasi
7. Antarmuka komparator analog
8. 4 *channel* PWM
9. kecepatan nilai (*speed grades*) 0 - 8 MHz untuk ATmega16L dan 0 – 16 MHz untuk ATmega16

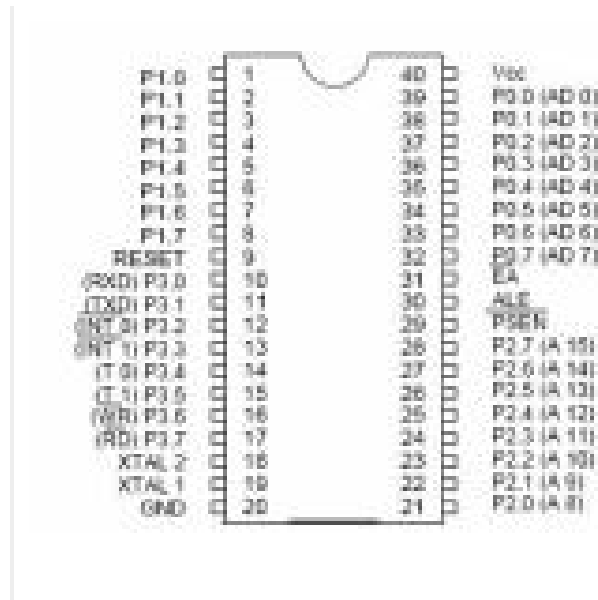


Gambar 2.7 Mikrokontroler ATmega 16
(Datasheet ATmega 16)

2.3.2.2 Mikrokontroler 89S52

Mikrokontroler AT89S52 merupakan pengembangan dari mikrokontroler MCS-51. Mikrokontroler ini biasa disebut juga dengan mikrokomputer CMOS 8 bit dengan 8 Kbyte yang dapat diprogram sampai 1000 kali pemograman. Selain itu AT89S52 juga mempunyai kapasitas RAM sebesar 256 bytes, 32 saluran I/O, Watchdog timer, dua pointer data, tiga buah timer/counter 16-bit, Programmable UART (Serial Port). Memori Flash digunakan untuk menyimpan perintah (instruksi) berstandar MCS-51, sehingga memungkinkan mikrokontroler ini bekerja sendiri tanpa diperlukan tambahan chip lainnya (single chip operation), mode operasi keping tunggal yang tidak memerlukan external memory dan memori flashnya mampu

diprogram hingga seribu kali. Hal lain yang menguntungkan adalah sistem pemrograman menjadi lebih sederhana dan tidak memerlukan rangkaian yang rumit. (wordpress, 2013 diakses 05 Juni 2016)



Gambar 2.8 Mikrokontroler 89S52

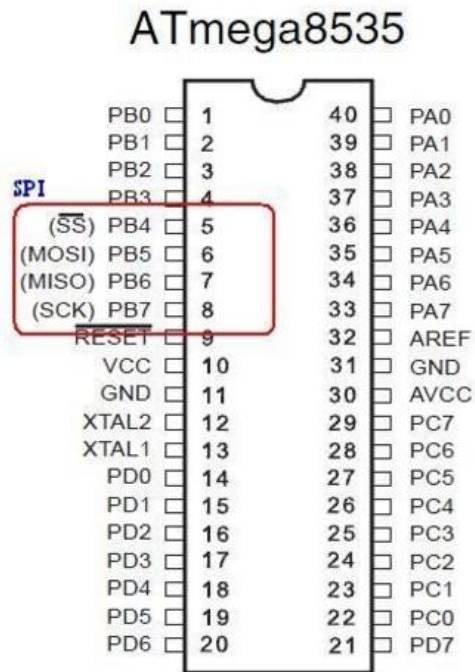
(wordpress, 2013)

2.3.2.3 Mikrokontroler ATMEGA 8535

ATMega8535 merupakan salah satu mikrokontroler 8 bit buatan Atmel untuk keluarga AVR. Karena merupakan keluarga AVR, maka ATMega8535 juga menggunakan arsitektur RISC. Secara singkat, ATMega8535 memiliki beberapa kemampuan:

Jalur I/O 32 buah yang terbagi dalam PORT A, PORT B, PORT C, PORT D dengan masing-masing PORT ada 8 pin. ADC 10 bit sebanyak 8 input, 2 buah Timer/Counter dengan kemampuan pembanding. CPU 8 bit yang terdiri dari 32 register, Frekuensi clock maksimum 16 MHz. Watchdog Timer dengan osilator internal. SRAM sebesar 512 byte. Memori flash sebesar 8 KB dengan kemampuan read while write. Interrupt internal dan eksternal. PORT komunikasi SPI. EEPROM sebesar 512 byte yang dapat

diprogram saat operasi .Analog Comparator dan komunikasi serial standart USART dengan kecepatan maksimal 2,5Mbps. (Sumardi, 2013)

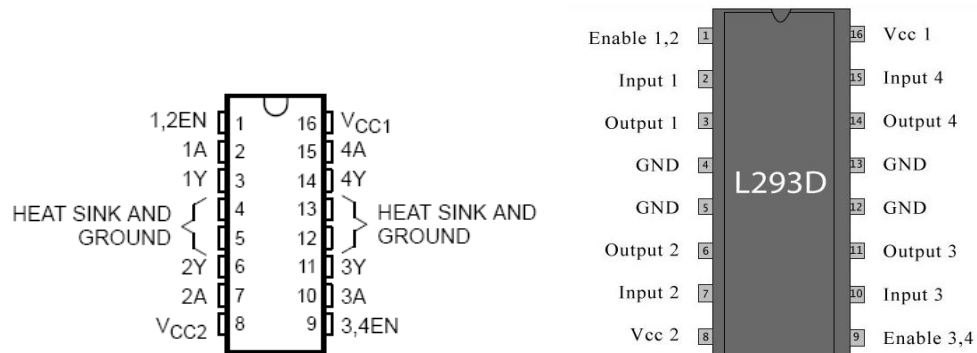


Gambar 2.9 Atmega 8538

(Sumardi, 2013)

2.4 Driver Motor

Driver Motor L293D adalah driver yang digunakan untuk arus tinggi. L293 dibuat untuk menyediakan arus yang dapat diarahkan hingga 1 A pada tegangan 4,5 Volt 36 volt. Sedangkan L293D dibuat untuk menyediakan arus yang dapat diarahkan hingga 600 mA pada tegangan 4,5 volt hingga 36 volt. Kedua driver tersebut didisain untuk dapat mengendalikan beban yang bersifat induktif seperti relay, solenoid, dc, dan motor stepping *bipolar*, maupun beban yang mempunyai tegangan tinggi dan arus tinggi dalam penggunaan positif-supply. (Taufiq Dwi Septian Suyadhi, 2008 : 40 diakses 10 juni 2016)



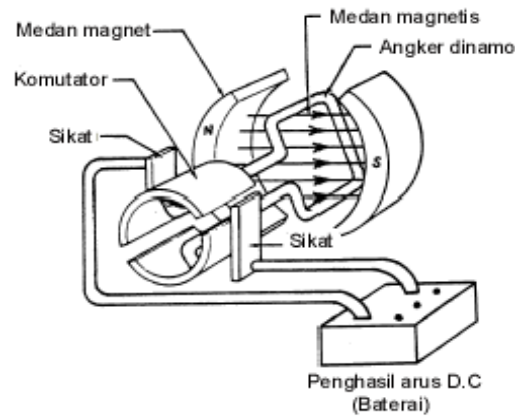
Gambar 2.10 Tampak atas LM293
(Taufiq Dwi Septian Suyadhi, 2008)

2.5 Motor DC

Motor DC atau mesin adalah perangkat mesin pertama yang mengkonversi besaran listrik menjadi besaran mekanik. Putaran dan torsi pada motor Dc yang dihasilkan dari gaya tarik-menarik dan gaya dorong yang dihasilkan oleh medan magnetik pada motor DC tersebut. Berdasarkan pada gambar, Motor DC terdiri dari 6 bagian utama antara lain : Axis atau poros motor Dc, bagian yang berputar yang disebut rotor, bagian yang tetap disebut stator, komuntator, Field magnets, dan brushes.

Mengendalikan kecepatan putaran motor DC ada 3 cara antara lain :

1. Metode ON Off.
2. Menggunakan variabel tegangan
3. Menggunakan PWM (Pulse Width Modulation).



Gambar 2.11 Motor DC Sederhana
(Sumardi, 2013)

Catu tegangan dc dari baterai menuju kelilitan melalui sikat yang menyentuh komutator, dua segmen yang terhubung dengan dua ujung ilitan. Kumparan satulilitan pada gambar di atas disebut angker dinamo. Angker dinamo adalah sebutan untuk komponen yang berputar di antara medan magnet. (Sumardi, 2013)

2.6 Rangkaian LCD

LCD berfungsi sebagai media penampil untuk menyeting robot *line follower*. (Datashhet LCD)

2.7 Power Supply

Dalam pembuatan rangkaian power supply, kita akan memanfaatkan IC regulator 7805 sebagai regulator tegangan yang akan di input kan pada seluruh rangkaian pembangun line follower. Tujuan penggunaan ic regulator 7805 adalah mendapatkan tegangan +5 volt yang stabil. Dengan demikian, rangkaian elektronik line follower yang digunakan bisa bekerja secara normal. (Taufiq Dwi septian suyadhi, 2008)

2.8 Bahasa Pemrograman

Bahasa C adalah bahasa pemrograman yang dapat dikatakan berada di antara bahasa beraras rendah dan beraras tinggi. Bahasa beraras rendah artinya bahasa yang berorientasi pada mesin, sedangkan beraras tinggi berorientasi pada manusia. Bahasa beraras rendah misalnya, bahasa assembler, ditulis dengan sandi yang hanya dimengerti oleh mesin sehingga hanya digunakan bagi yang memrogram mikroprosesor. Bahasa beraras rendah merupakan bahasa yang membutuhkan kecermatan tinggi bagi pemrogram karena perintahnya harus rinci, ditambah lagi masing-masing pabrik mempunyai sandi perintah sendiri. Bahasa tinggi relatif mudah digunakan karena ditulis dengan bahasa manusia sehingga mudah dimengerti dan tidak tergantung mesinnya. Bahasa beraras tinggi umumnya digunakan pada komputer.

Pencipta bahasa C adalah Brian W. Kernighan dan Denis M. Ritchi sekitar 1972. Penulisan program dalam bahasa C dilakukan dengan membagi dalam blok-blok sehingga bahasa C disebut bahasa terstruktur. Bahasa C dapat digunakan mesin dengan mudah, mulai dari PC sampai *mainframe*, serta menggunakan berbagai sistem operasi misalnya DOS, UNIX, VMS, dan lain-lain.

2.8.1 Penulisan Bahasa C

Program Bahasa C tidak mengenal aturan penulisan di kolom tertentu sehingga bisa dimulai dari kolom manapun. Namun demikian, untuk mempermudah pembacaan program dan keperluan dokumentasi, sebaiknya penulisan bahasa C diatur sedemikian rupa sehingga mudah dan enak dibaca.

Berikut contoh penulisan program bahasa C:

```
#include <atmega16.h>
```

```
Main()
```

```
{
```

```
.....}
```

Program dalam bahasa C selalu berbentuk fungsi seperti ditunjukkan main(). Program yang dijalankan berada dalam tubuh program dan dimulai dengan tanda kurung buka “{“ dan diakhiri tanda kurung tutup “}”. Semua yang tertulis didalam tubuh program disebut blok.

Tanda () digunakan untuk mengapit argumen suatu fungsi. Argumen adalah suatu nilai yang akan digunakan dalam fungsi tersebut. Setiap pernyataan dalam bahasa pemrograman C harus diakhiri dengan tanda titik koma “;”.

Baris pertama #include<...> bukanlah pernyataan sehingga tidak memerlukan titik koma pada akhir penulisannya. Baris pertama tersebut meminta compiler untuk menyertakan file yang namanya ada diantara tanda <...> dalam proses kompilasi. File ini berekstensi .h berisi deklarasi fungsi ataupun variable. File ini disebut header dan digunakan semacam perpustakaan (*library*) untuk pernyataan yang ada di tubuh program.

2.8.2 Tipe Data

Tipe data merupakan bagian program paling penting karena mempengaruhi setiap instruksi yang akan dilaksanakan oleh komputer. Misalnya, 7 dibagi 2 bisa menghasilkan hasil yang berbeda tergantung tipe datanya. Jika 7 dan 2 bertipe integer maka akan menghasilkan nilai 3, tetapi jika keduanya bertipe float maka akan menghasilkan nilai 3.500000. Pemilihan tipe data yang tepat akan membuat proses operasi data menjadi lebih efisien dan efektif. (Yolan,2015)

Tabel 2.1 Tipe Data

Tipe Data	Byte	Bit	Minimum	Maximum
Char	1	8	-128	127
Signed char	1	8	-128	127
Unsigned char	1	8	0	255
Int	2	16	-32768	32767

Signed int	2	16	-32768	32767
Unsigned int	2	16	0	65536
Signed long	4	32	-2147483648	2147483647
Unsigned long	4	32	0	4294967295
Float	4	32	1.28E-38	3.4E3.8

2.8.3 Variabel

Variabel adalah tempat untuk menyimpan dan mengakses data yang mewakili memori dalam mikrokontroler. Variabel harus dideklarasikan (memberitahu *compiler*) dengan tipe data beserta nama variabel yang akan digunakan. Bahasa C bersifat *case sensitive* dimana huruf kapital dan huruf kecil dibedakan. Tiap tipe data mempunyai jangkauan bilangan yang dapat disimpan, hal ini akibat dari memori yang dipesan dan bentuk bilangan bertanda atau tidak. Misalnya unsigned char oleh *compiler* disediakan 1 byte memori RAM sehingga hanya bisa menampung bilangan dari 0 s.d. 255 sedangkan jika bertanda -128 s.d. 127. (Yolan,2015)

2.8.4 Deklarasi

Deklarasi diperlukan bila kita akan menggunakan pengenal (*identifier*) dalam program. *Identifier* dapat berupa variabel, konstanta, dan fungsi.

a. Deklarasi Variabel

Bentuk umum pendeklarasian suatu variabel adalah:

Nama_tipe nama_variabel;

Contoh:

```
int x; //deklarasi x bertipe integer
```

```
char y; //deklarasi y bertipe char
```

b. Deklarasi Konstanta

Dalam bahasa C, konstanta dideklarasikan menggunakan *preprocessor* `#define`. Contohnya:

```
#define phi 3.14
#define nim "03101004096"
#define nama "abdi prasetyo"
```

c. Deklarasi Fungsi

Fungsi adalah bagian terpisah dari program dan dapat diaktifkan atau dipanggil dimanapun dalam program. Ada fungsi dalam bahasa C yang sudah disediakan sebagai fungsi pustaka, seperti `printf()`, `scanf()`, dan `getch()`. Kemudian, fungsi tersebut tidak perlu dideklarasikan untuk menggunakannya.

Fungsi yang perlu dideklarasikan terlebih dahulu adalah fungsi yang dibuat oleh *programmer*. Bentuk umum deklarasi sebuah fungsi adalah:

```
Tipe_fungsi nama_fungsi (parameter fungsi);
```

Contohnya:

```
float luas_lingkaran(int jari);
```

```
void tampil();
```

```
int tambah(int x,int y);
```

2.8.5 Operator

Operator adalah karakter-karakter khusus yang digunakan untuk memanipulasi variabel. Operator aritmatika, penugasan (*assignment*), logika dan bit, relasi, pointer.

a. Operator Aritmatika

Tabel 2.2 Operator Aritmatika

Binary operator	Keterangan	Contoh
+	Penambahan	$A=b+c$
-	Pengurangan	$A=b-c$
*	Perkalian	$A=b*c$
/	Pembagian	$A=b/c$
%	Modulo(sisabagi)	$A=b\%c$
Unery operator	Keterangan	Contoh
++	Increment	A++
--	Decreament	A--

b. Operator Penugasan (*assignment*)

Operator ini digunakan untuk member atau memasukkan nilai suatu variabel.

Tabel 2.3 Operator Penugasan

Penugasan	Arti	Contoh	Keterangan
=	Penugasansederhana	$X=8+y$	-
+=	Penugasan +	$X+=8$	$X=X+8$
-=	Penugasan -	$X-=8$	$X=X-8$
*=	Penugasan *	$X*=8$	$X=X*8$
/=	Penugasan /	$X/=8$	$X=X/8$
%=	Penugasan %	$X%=8$	$X=X58$
<<=	Penugasan<<	$X<<=2$	$X=X<<2$
>>=	Penugasan>>	$X>>=2$	$X=X>>2$

c. Operator Relasional

Operator ini berfungsi untuk menguji benar atau tidaknya hubungan dua *operand*, jika hubungannya benar maka akan menghasilkan 1 dan jika salah akan menghasilkan 0.

Tabel 2.4 Operator Relasional

Operator	Operasi	Contoh
==	Samadengan	if(x==1){x++;};
!=	Tidaksamadengan	if(x!=1){x++;};
>	Lebihbesar	if(x>1){x++;};
<	Lebihkecil	if(x<1){x++;};
>=	Lebihbesarsamadengan	if(x>=1){x++;};
<=	Lebihkecilsamadengan	if(x<=1){x++;};

d. Operator Bit

Berfungsi untuk mengoperasikan antar bit variabel.

Tabel 2.5 operator Bit

Operator	Operasi
&	Meng-and-kantiappasangan bit danmenghasilkan 1 jikakeduanyaberisi 1, dan 0 untuk yang lainnya
	Meng-or-kantiappasangan bit danmenghasilkan 0 jikakeduanyaberisi 0, dan 1 untuk yang lainnya
^	Meng-exor-kantiappasangan bit danmenghasilkan 1 jikakeduanyaberbedaisi, dan 0 untuk yang lainnya
~	Membalik(menegasikan) isitiap bit
>>	Menggesertiap bit kekanan
<<	Menggesertiap bit kekiri

Tabel 2.6 Operasi logika Dasar

A	B	A&B	A B	A^B	~B
0	0	0	0	0	1
0	1	0	1	1	0
1	0	0	1	1	1
1	1	1	1	0	0

2.9 Komentar Program

Komentar hanya diperlukan untuk memudahkan pembacaan dan pemahaman suatu program (untuk keperluan dokumentasi program). Dengan kata lain, komentar program hanya merupakan keterangan atau penjelasan program. Cara memberikan

komentar atau penjelasan dalam bahasa C adalah menggunakan pembatas `/*` dan `*/` atau menggunakan tanda `//` untuk komentar yang hanya terdiri atas satu baris. Komentar program tidak akan ikut diproses dalam program (akan diabaikan).

Contoh pertama:

```
//Author : Abdi Prasetyo
```

```
//Company : Abdi Prasetyo
```

Di belakang tanda `//` tak akan diproses dalam kompilasi. Tanda ini hanya untuk satu baris kalimat.

```
/* Program Eksekusi
```

```
Simpang */
```

Bentuk ini berguna kalau pernyataannya berupa kalimat panjang sampai beberapa baris.

2.9.1 Array

Array merupakan kumpulan nilai data bertipe sama dalam urutan tertentu yang menggunakan nama sama. Letak atau posisi elemen array ditunjukkan oleh suatu indeks. Dilihat dari dimensinya array dapat dibagi menjadi array dimensi satu, array dimensi dua, dan array multi dimensi.

a. Array Dimensi Satu

Setiap elemen *array* dapat diakses melalui indeks. Indeks *array* secara default dimulai dari deklarasi *Array*. Bentuk umumnya adalah:

Tipe_array nama_array[ukuran];

b. Array Dimensi Dua

Array dua dimensi merupakan *array* yang terdiri atas m baris dan n kolom. Bentuknya dapat berupa matriks atau tabel.

Deklarasi *array* dimensi dua:

Tipe_array nama_array[baris][kolom];

c. Array Multidimensi

Array multidimensi merupakan *array* yang mempunyai ukuran lebih dari dua. Bentuk pendeklarasian *array* sama dengan *array* dimensi satu maupun *array* dimensi dua.

Bentuk umumnya adalah:

Tipe_array nama_array[ukuran1][ukuran2]...[ukuranN];

2.9.2 Fungsi

Fungsi merupakan bagian program yang bertujuan mengerjakan tugas tertentu dan letaknya terpisah dari program yang memanggilnya. Fungsi merupakan elemen utama dalam bahasa C karena bahasa C sendiri terbentuk dari kumpulan fungsi.

Dalam setiap program bahasa C, minimal terdapat satu fungsi, yaitu fungsi `main()`. Fungsi banyak diterapkan dalam program-program C yang terstruktur. Keuntungan penggunaan fungsi dalam program adalah program akan memiliki struktur yang jelas (memiliki *readability* tinggi) serta akan menghindari penulisan bagian program yang sama.

Contoh:

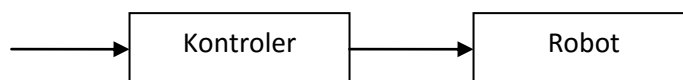
```
intjumlah(int bil1,int bil2)
{
    inthasil;
    hasil = bil1 + bil2;
    return(hasil);
}
```

(Arandhita. 2013)

2.10 Sistem Kontrol PID Pada Robot Lne Follower

Sistem kontrol robotik pada dasarnya terbagi dua kelompok, yaitu sistem kontrol loop terbuka (*open loop*) dan loop tertutup (*close loop*). Diagram kontrol loop terbuka pada sistem robot dapat dinyatakan dalam gambar 2.10 berikut ini.

Referensi gerak



Gambar 2.12 Kontrol Robot Loop Terbuka

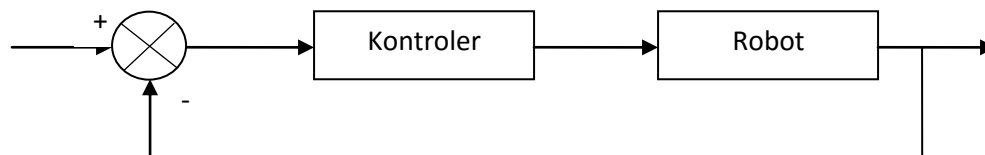
(fahmizal, 2013)

Kontrol loop terbuka atau umpan maju (*feedforward*) dapat dinyatakan sebagai sistem kontrol yang outputnya tidak diperhitungkan ulang oleh kontroler. Keadaan apakah robot benar-benar telah mencapai target seperti yang dikehendaki sesuai referensi, adalah tidak dapat mempengaruhi kinerja kontroler. Kontrol ini sesuai untuk sistem operasi robot yang memiliki aktuator yang beroperasi

berdasarkan umpan logika berbasis konfigurasi langkah sesuai urutan, misalnya *stepper* motor. *Stepper* motor tidak perlu dipasang sensor pada porosnya untuk mengetahui posisi akhir. Jika dalam keadaan berfungsi baik dan tidak ada masalah beban lebih maka *stepper* motor akan berputar sesuai dengan perintah kontroler dan mencapai posisi target dengan tepat.

Kontrol robot loop tertutup dapat dinyatakan seperti dalam gambar

Referensi gerak



Gambar 2.13 Kontrol Robot Loop Tertutup

(fahmizal, 2013)

Pada gambar diatas, jika hasil gerak aktual telah sama dengan referensi maka input kontroler akan nol. Artinya kontroler tidak lagi memberikan sinyal aktuasi kepada robot karena target akhir perintah gerak telah diperoleh. Makin kecil *error* terhitung maka makin kecil pula sinyal pengemudian kontroler terhadap robot, sampai akhirnya mencapai kondisi tenang (*steady state*).

2.10.1 Kontrol Proporsional

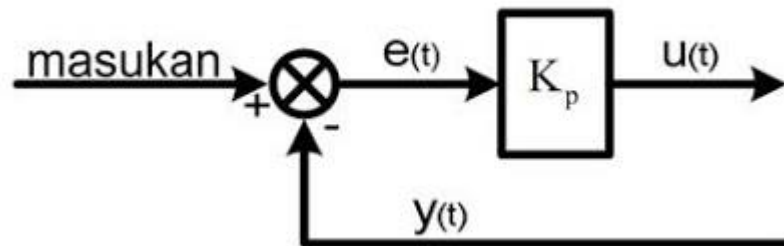
Kontrol proporsional berfungsi untuk memperkuat sinyal kesalahan penggerak (sinyal *error*), sehingga akan mempercepat keluaran sistem mencapai titik referensi.

Bentuk umum dari kontrol proporsional dapat dinyatakan sebagai berikut:

$$P_{\text{out}} = K_p e(t) \dots \dots \dots (2.1)$$

Dengan K_p adalah konstanta proporsional. K_p berlaku sebagai *gain* (penguat) saja tanpa memberikan efek dinamik kepada kinerja kontroler. Penggunaan kontrol P memiliki berbagai keterbatasan karena sifat kontrol yang tidak dinamik ini. Walaupun

demikian dalam aplikasi-aplikasi dasar yang sederhana kontrol P ini cukup mampu untuk mencapai konvergensi meskipun *error steady state* relatif besar. Sebagai materi pembelajaran, kontrol P dianggap sangat baik untuk permulaan.



Gambar 2.14 Diagram blok kontrol proporsional

(fahmizal, 2013)

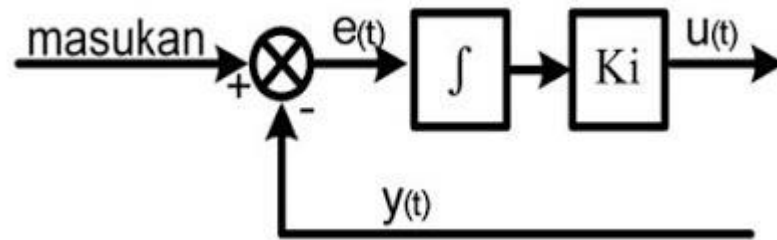
2.10.2 Kontrol Integral

Kontrol integral pada prinsipnya bertujuan untuk menghilangkan kesalahan keadaan tunak (offset) yang biasanya dihasilkan oleh kontrol proporsional.

*“Integral (\int) adalah suatu operator matematis dalam kawasan kontinyu jika didiskretisasi maka akan menjadi sigma (Σ), yang merupakan operator matematis dalam kawasan diskret. Dimana fungsi dari operator sigma adalah menjumlahkan nilai ke i sampai dengan nilai ke k . Berdasarkan perhitungan diatas variabel error (e) yang di integralkan sehingga dalam kawasan diskret menjadi $e(0)+e(1)+\dots+e(k-1)+e(k)$, atau dengan kata lain **error yang sebelumnya dijumlahkan dengan error-error yang sebelumnya hingga error yang sekarang.**”*

Bentuk umum dari kontrol integral dapat dinyatakan sebagai berikut:

$$I_{\text{out}} = K_i \int_0^t e(\tau) d\tau \dots\dots\dots(2.2)$$



Gambar 2.15 Diagram Blok Kontrol Integral

(fahmizal, 2013)

Jika $e(t)$ mendekati konstan (bukan nol) maka I_{out} akan menjadi sangat besar sehingga diharapkan dapat memperbaiki *error*. Jika $e(t)$ mendekati nol maka efek kontrol I ini semakin kecil. Kontrol I dapat memperbaiki respon *steady-state*, namun pemilihan K_i yang tidak tepat dapat menyebabkan respon transien yang tinggi sehingga justru dapat menyebabkan ketidakstabilan sistem. Pemilihan K_i yang sangat tinggi justru dapat menyebabkan output berosilasi.

2.10.3 Kontrol Derivatif

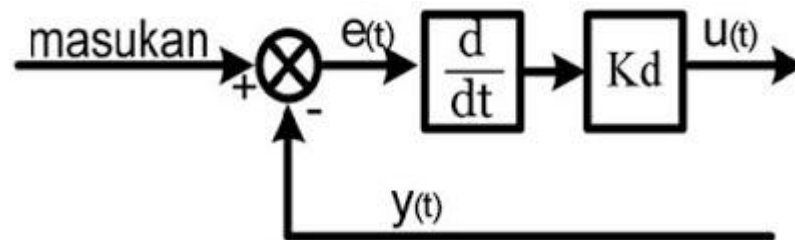
Kontrol derivatif dapat disebut pengendali laju, karena output kontroler sebanding dengan laju perubahan sinyal error.

*“Derivatif (de/dt) adalah suatu operator matematis dalam kawasan kontinyu jika didiskretisasi maka akan menjadi limit, yang merupakan operator matematis dalam kawasan diskret. Dimana fungsi dari operator limit adalah mengurangi nilai ke k dengan nilai ke $k-1$. Berdasarkan perhitungan diatas variabel error (e) yang di derivatifkan, atau dengan kata lain **error yang sekarang dikurangi error yang sebelumnya.**”*

Waktu sampling adalah lamanya waktu yang digunakan untuk mencuplik atau mensampling nilai dari sensor. Nilai dari sensor ini berguna untuk mendapatkan sinyal error ($error(e) = \text{set point} - \text{nilai sensor}$). Dimana waktu sampling ini sangat berpengaruh pada kesensitifan sistem yang akan dikontrol.

Bentuk umum dari kontrol derivative dapat dinyatakan sebagai berikut:

$$D_{\text{out}} = K_d \frac{d}{dt} e(t) \dots \dots \dots (2.3)$$

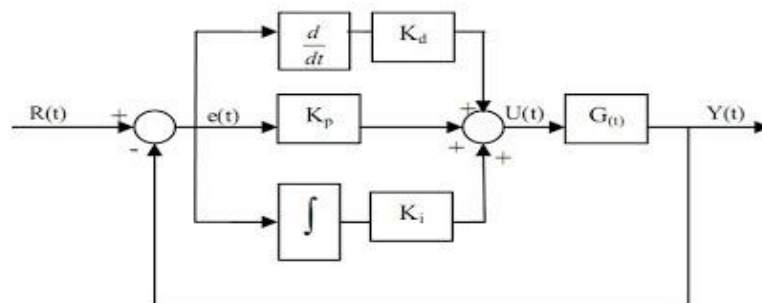


Gambar 2.16 Diagram Blok Kontrol Derivatif

(fahmizal, 2013)

Dari persamaan diatas terlihat bahwa sifat dari kontrol D ini berhubungan dengan konteks “kecepatan” atau rate dari *error*. Dengan sifat ini kontrol D dapat digunakan untuk memprediksi *error* yang akan terjadi. Umpan balik yang diberikan adalah sebanding dengan kecepatan perubahan $e(t)$ sehingga kontroler dapat mengantisipasi *error* yang akan terjadi.

Gabungan dari ketiga kontroler tersebut menjadi **kontrol PID**.



Gambar 2.17 Diagram Blok Kontrol PID

Sehingga persamaan untuk kontrol PID adalah:

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

atau

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

Dengan:

$u(t)$ = sinyal output pengendali PID

K_p = konstanta proporsional

T_i = waktu integral

T_d = waktu derivatif

K_i = konstanta integral (K_p/T_i)

K_d = konstanta derivatif ($K_p.T_d$)

$e(t)$ = sinyal error = referensi – keluaran plant = set point – nilai sensor


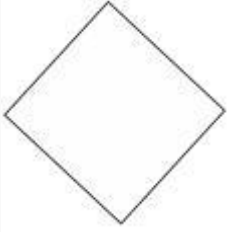



T_c = waktu sampling atau waktu cuplik (*Sampling time*)


2.11 Flowchat

Merupakan sebuah diagram dengan simbol-simbol grafis yang menyatakan aliran algoritma atau proses yang menampilkan langkah-langkah yang disimbolkan dalam bentuk kotak, beserta urutannya dengan menghubungkan masing masing langkah tersebut menggunakan tanda panah. Diagram ini bisa memberi solusi

selangkah demi selangkah untuk penyelesaian masalah yang ada di dalam proses atau algoritma tersebut.

Gambar berikut adalah simbol flowchart yang umum digunakan.

Gambar	Simbol untuk	Keterangan
	Proses / Langkah	Menyatakan kegiatan yang akan ditampilkan dalam diagram alir.
	Titik Keputusan	Proses / Langkah di mana perlu adanya keputusan atau adanya kondisi tertentu. Di titik ini selalu ada dua keluaran untuk melanjutkan aliran kondisi yang berbeda.
	Masukan / Keluaran Data	Digunakan untuk mewakili data masuk, atau data keluar.
	Terminasi	Menunjukkan awal atau akhir sebuah proses.
	Garis alir	Menunjukkan arah aliran proses atau algoritma.

	Kontrol / Inspeksi	Menunjukkan proses / langkah di mana ada inspeksi atau pengontrolan.
---	--------------------	--

-
- a. *Diagram Alir Dokumen*, menunjukkan kontrol dari sebuah sistem aliran dokumen.
 - b. *Diagram Alir Data*, menunjukkan kontrol dari sebuah sistem aliran data.
 - c. *Diagram Alir Sistem*, menunjukkan kontrol dari sebuah sistem aliran secara fisik.
 - d. *Diagram Alir Program*, menunjukkan kontrol dari sebuah program dalam sebuah sistem. (**wikipedia**)