BAB II

DASAR TEORI

2.1. Definisi Robot

Menurut definisi dari kamus Meriam-Webster, robot adalah mesin yang terlihat seperti manusia dan melakukan berbagai tindakan yang kompleks dari manusia seperti berjalan atau berbicara, atau suatu peralatan yang bekerja secara otomatis. Robot biasanya diprogram untuk melakukan pekerjaan berulang kali dan memiliki mekanisme yang dipandu oleh kontrol otomatis[1].

Terdapat empat karakteristik dasar yang harus dimiliki oleh setiap robot modern. Karakteristik dasar tersebut adalah sebagai berikut[1].

a. Memiliki Sensor

Sensor merupakan peralatan yang berguna untuk mengukur ataupun merasakan sesuatu pada lingkungan di luar robot, layaknya indera pada makhluk hidup, dan memberi laporan hasilnya kepada robot. Dengan adanya sensor, robot bisa memiliki suatu pertimbangan dalam mengambil keputusan[1].

b. Memiliki Sistem Kecerdasan (Kontrol)

Sistem kecerdasan bekerja dengan memproses data masukan berupa keadaan ataupun kejadian yang sedang terjadi dari luar lingkungan. Selanjutnya sistem menghasilkan keluaran berupa instruksi ataupun keputusan pada robot untuk melakukan suatu tindakan tertentu. Sistem ini secara umum memiliki prinsip kerja seperti otak pada makhluk hidup, yang berfungsi untuk berpikir dan memutuskan tindakan apa yang perlu diambil pada suatu waktu tertentu[1].

c. Memiliki Peralatan Mekanik (Aktuator)

Peralatan mekanik berfungsi untuk membuat robot dapat melakukan suatu tindakan tertentu dan berinteraksi dengan lingkungannya. Contohnya seperti adanya roda bermotor untuk bergerak, lengan untuk mengambil objek, dan lain-lain[1].

d. Memiliki Sumber Daya (*Power*)

Sebagaimana pada organisme kehidupan yang membutuhkan makanan untuk hidup, robot juga memerlukan sumber tenaga untuk menggerakkan komponen elektrik dan mekanika yang terpasang. Sumber energi pada robot mencakup penyedia tenaga listrik seperti baterai, dan sistem pengatur transmisi yang bertugas mengonversi tenaga listrik sesuai kebutuhan setiap komponen[1].

2.2. Jenis Robot

Secara umum jenis robot dibedakan menjadi empat kategori, yaitu sebagai berikut[2].

a. *Non-mobile Robot*

Robot ini tidak dapat berpindah posisi dari satu tempat ke tempat lainnya, sehingga robot tersebut hanya dapat menggerakkan beberapa bagian dari tubuhnya dengan fungsi tertentu yang telah dirancang. Contoh dari *non-mobile robot* adalah robot industri. Secara umum perangkat pendukung dari robot industri terdiri dari empat komponen utama, yaitu : manipulator, sensor, aktuator, dan kontroler[2].

b. *Mobile Robot*

Mobile dapat diartikan bergerak, sehingga robot ini dapat memindahkan dirinya dari satu tempat ke tempat lain. Robot ini merupakan robot yang paling populer dalam dunia penelitian robotika. Dari segi manfaat, robot ini diharapkan dapat membantu manusia dalam melakukan otomasi dalam transportasi, *platform* bergerak untuk robot industri , eksplorasi tanpa awak, dan masih banyak lagi. Contoh *mobile robot* adalah *robot line tracker*, *flying robot, under water robot*[2].

c. Kombinasi *Mobile* dan *Non-Mobile Robot*

Robot ini merupakan penggabungan dari fungsi-fungsi pada *mobile* dan *non-mobile robot*. Sehingga keduanya saling melengkapi dimana robot *non-mobile* dapat terbantu fungsinya dengan bergerak dari satu tempat ke tempat yang lain[2].

d. *Humanoid Robot*

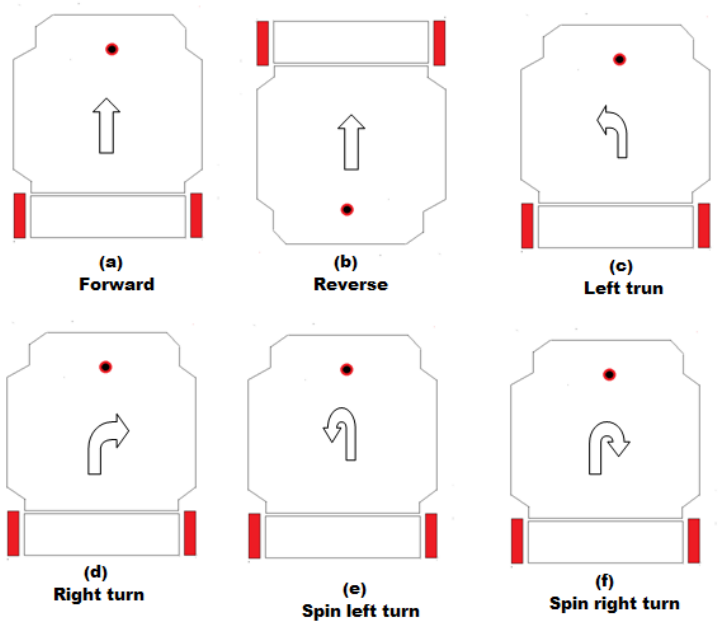
Sebuah *humanoid robot* adalah robot otonom yang dapat beradaptasi dengan perubahan lingkungan atau dirinya sendiri. Ini merupakan perbedaan utama antara jenis *humanoid* dan jenis robot lainnya[2].

2.3. Prinsip Pergerakan *Mobile Robot* dan Sistem Navigasi Robot

Pergerakan merupakan cara berpindah dari satu titik ke titik lain. Terdapat banyak bentuk pergerakan robot, yang paling umum adalah roda, rel, dan kaki. Salah satu contoh sistem kemudi untuk robot beroda adalah *Differential Steering*[3].

a. *Differential Steering Robot*

*Differential Steering Robo*t adalah robot yang dapat bermanuver dengan memanipulasi perbedaan kecepatan kedua roda di sisi kanan dan sisi kiri[3]. Di bawah ini merupakan ilustrasinya yang ditunjukkan pada gambar 2.1.



Gambar 2.1 Ilustrasi Sistem Pergerakan *Differential Steering*[3]

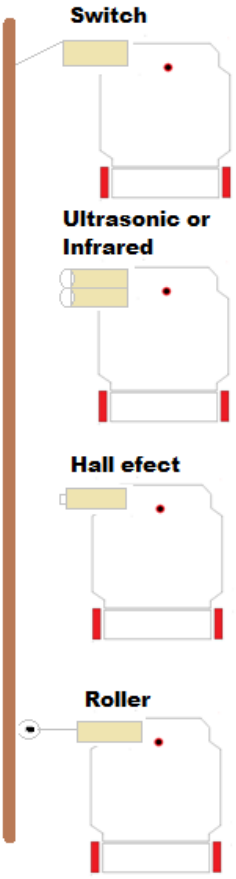
Berdasarkan gambar 2.1, pergerakan *differential steering* dapat dijelaskan sebagai berikut[3].

- Posisi a (*forward*). Jika kedua motor berputar dengan arah dan kecepatan yang sama maka robot akan lurus maju ke depan.
- Posisi b (*reverse*). Jika kedua motor berputar dengan arah dan kecepatan yang sama maka robot akan lurus mundur ke belakang.
- Posisi c (*left turn*). Jika di sisi kanan berputar dan motor di sisi kiri tidak berputar, maka robot akan bergerak/berbelok ke kiri.

- Posisi d (*right turn*). Jika di sisi kiri berputar dan di sisi motor di sisi kanan tidak berputar, maka robot akan bergerak/berbelok ke kanan.
- Posisi e (*spin left turn*). Motor kiri dan kanan berputar dengan arah yang berlawanan. Pada keadaan ini robot akan berbelok/berputar ke kiri/berlawan arah jarum jam dengan pusat putaran pada titik hitam (pivot point). Untuk melakukan maneuver ini, kedua motor harus berputar dengan kecepatan yang sama sehingga pivot point tetap berada di tengah.
- Posisi f (*spin right turn*). Motor kiri dan kanan berputar dengan arah yang berlawanan. Pada keadaan ini robot akan berbelok/berputar ke kanan/searah jarum jam dengan pusat putaran pada titik hitam (*pivot point*). Untuk melakukan maneuver ini, kedua motor harus berputar dengan kecepatan yang sama sehingga *pivot point* tetap berada di tengah.

b. *Wall Following*

Wall Following adalah suatu algoritma untuk menyediakan orientasi navigasi kepada robot dengan menyusuri dinding. Salah satu keuntungannya adalah tidak perlu adanya garis penuntun atau suatu tanda khusus sebagai arahan bagi robot. Cara kerjanya adalah dengan mengatur jarak dinding dengan robot agar tetap konstan. Bila terjadi perubahan, maka robot akan bergerak untuk kemudian menyesuaikan jarak lagi[3]. Proses ini akan dilakukan secara berulang-ulang[3]. Di bawah ini merupakan metode-metode *wall following* yang ditunjukkan pada gambar 2.2.



Gambar 2.2 Metode-metode *Wall Following*[3]

Dari gambar di atas, dapat dijelaskan metode-metode *wall following* sebagai berikut[3].

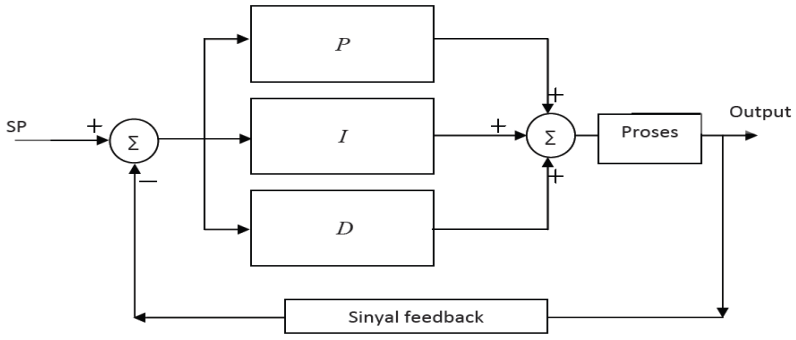
- *Contact*, robot menggunakan saklar mekanik yang merasakan sentuhan dengan dinding. Ini adalah metode yang paling mudah namun saklar akan cenderung mengalami kerusakan mekanis setelah beberapa waktu karena terus bersentuhan dengan objek (dinding).
- *Non-contact* (*active sensor*), robot menggunakan sensor aktif yang beroperasi dalam jarak dekat seperti infra merah atau ultrasonik untuk mengukur jarak antara dinding dengan robot.
- *Non-contact* (*passive sensor*), robot memakai sensor pasif seperti saklar *Hall effect* untuk mengukur jarak antara robot dengan dinding.

Pada kasus ini, dinding harus berbahan logam atau dipasangkan kabel elektrik agar sensor dapat menangkap medan magnetik saat robot mendekati dinding.

- *Soft*-contact, robot menggunakan bahan mekanik untuk mendeteksi sentuhan dengan dinding, namun sentuhan ini diperhalus dengan memasang material lunak atau lentur contohnya roda dari busa atau karet. Kelebihan dari metode ini adalah berkurangnya kerusakan mekanis.

2.4. Pengontrol PID

Kontroler PID merupakan kontroler yang cukup kompleks dengan kombinasi kontroler P (proporsional), kontroler I (integratif), dan kontroler D (derivatif). Kontroler PID dapat menghilangkan *error steady state* dan membuat respon sistem menjadi lebih cepat[4].

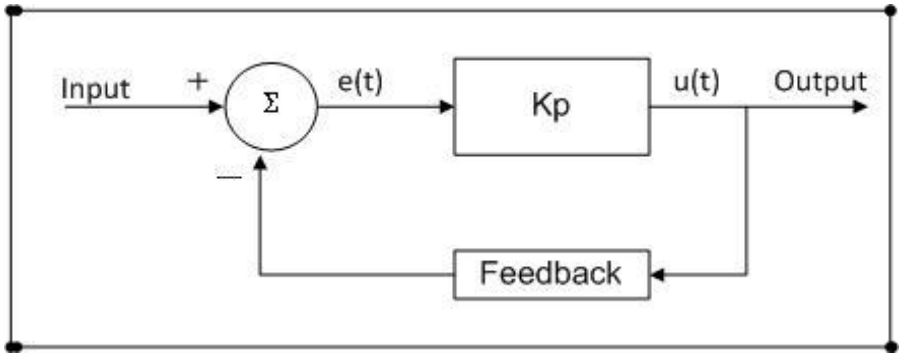


Gambar 2.3 Diagram Blok PID[4]

a. Pengontrol Proporsional (P)

Pengontrol proporsional memberi pengaruh sebanding dengan *error* (selisih antara besaran yang diinginkan dengan harga aktualnya). Semakin besar *error* maka sinyal kontrol yang dihasilkan semakin besar[4].

Secara lebih sederhana dapat dikatakan bahwa keluaran pengontrol proporsional merupakan perkalian antara konstanta proporsional dengan masukannya. Perubahan pada sinyal masukan akan segera menyebabkan sistem secara langsung mengeluarkan *output* sinyal sebesar konstanta pengalinya. Gambar 2.4 menunjukan blok diagram yang menggambarkan hubungan antara besaran *setpoint*, besaran aktual dengan keluaran pengontrol proporsional.



Gambar 2.4 Diagram Blok Pengontrol Proporsional[5]

Diagram blok di atas dapat diwujudkan dalam bentuk persamaan matematis dan laplace seperti pada persamaan berikut dimana K_P adalah konstanta proporsional[5].

$$u(t) = K_P . e(t) \quad \text{..... (1)}$$

Pengaruh kontrol Proporsional pada suatu sistem, yaitu[4]:

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

Ciri-ciri pengontrol proporsional harus diperhatikan ketika pengontrol tersebut diterapkan pada suatu sistem. Secara eksperimen, pengguna

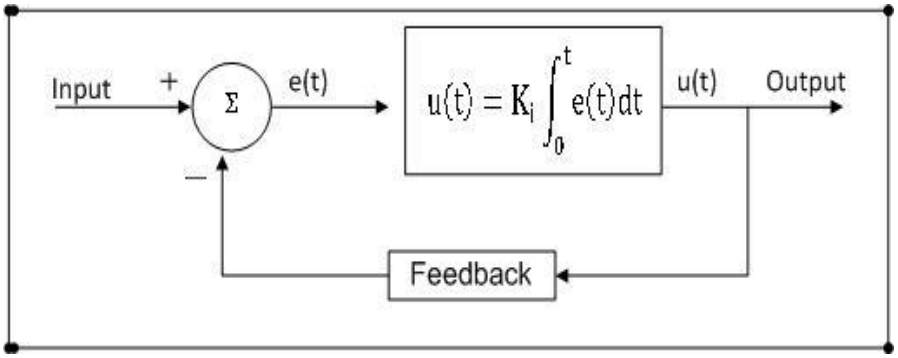
pengontrol proporsional harus memperhatikan ketentuan-ketentuan berikut ini[6]:

- Jika nilai K_P kecil, maka pengontrol proporsional hanya mampu melakukan koreksi kesalahan yang kecil, sehingga akan menghasilkan sistem respon yang lambat.
- Jika nilai K_P dinaikkan, respon sistem menunjukkan semakin cepat mencapai keadaan mantapnya.
- Jika nilai K_P diperbesar mencapai harga yang berlebihan, akan mengakibatkan sistem bekerja tidak stabil, atau sistem akan berosilasi.

b. Pengontrol Integratif (I)

Pengontrol integratif berfungsi menghasilkan respon sistem yang memiliki kesalahan keadaan stabil nol (*error steady state = 0*). Jika sebuah pengontrol tidak memiliki unsur *integrator*, pengontrol proporsional tidak akan mampu menjamin keluaran sistem dengan kesalahan keadaan stabilnya nol[6]. Dengan pengontrol integratif, respon sistem dapat diperbaiki, yaitu mempunyai kesalahan keadaan stabilnya nol.

Kontrol integratif memberi pengaruh sesuai dengan perubahan *error*, semakin besar *error* maka semakin besar juga sinyal kontrol yang dihasilkan. Kontrol integratif memiliki karakteristik seperti halnya sebuah integratif. Keluaran kontroler sangat dipengaruhi oleh perubahan yang sebanding dengan nilai sinyal kesalahan. Keluaran kontroler ini merupakan akumulasi yang terus menerus dari perubahan masukannya[4]. Gambar 2.5 adalah diagram blok dari sistem kontrol integratif.



Gambar 2.5 Diagram Blok Pengontrol Integratif[5]

Diagram blok di atas dapat diwujudkan dalam bentuk persamaan matematis dan laplace seperti pada persamaan berikut dimana K_i adalah konstanta integrative[5].

$$u(t) = K_i \int_0^t e(t)dt \quad \text{..... (2)}$$

Pengaruh pengontrol Integratif pada suatu sistem, yaitu[4]:

- Menghilangkan *error steady state.*
- Dapat memperbaiki *transient respon.*
- Dapat menambah ketidakstabilan sistem

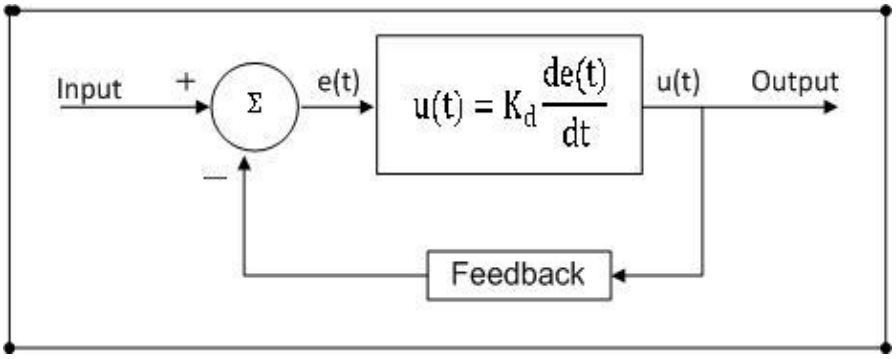
Ketika digunakan, pengontrol integratif mempunyai beberapa karakteristik berikut ini[6]:

- Keluaran kontroler membutuhkan selang waktu tertentu sehingga pengontrol integratif cenderung memperlambat respon.
- Ketika sinyal kesalahan berharga nol, keluaran kontroler akan bertahan pada nilai sebelumnya.
- Jika sinyal kesalahan tidak berharga nol, keluaran akan menunjukan sinyal kesalahan dan nilai K_i.

- Pengontrol integratif K_i yang berharga besar akan mempercepat hilangnya *offset*. Tetapi semakin besar nilai konstanta K_i akan mengakibatkan peningkatan osilasi dari sinyal keluaran kontroler.

c. Pengontrol Derivatif (D)

Pengontrol derivatif memberi pengaruh terhadap besarnya sinyal kontrol yang dihasilkan sebanding dengan perubahan *error*. Semakin cepat *error* berubah, maka semakin besar sinyal kontrol yang dihasilkan. Keluaran kontroler differensial memiliki sifat seperti halnya suatu operasi derivatif. Perubahan yang mendadak pada masukan kontroler, akan mengakibatkan perubahan yang sangat besar[4]. Gambar 2.6 adalah diagram blok dari sistem kontrol derivatif.



Gambar 2.6 Diagram Blok Pengontrol Derivatif[5]

Blok diagram di atas diwujudkan dalam bentuk persamaan laplace seperti pada persamaan berikut dimana K_d adalah konstanta derivatif.

$$u(t) = K_d \frac{de(t)}{dt} \quad \text{..... (3)}$$

Kontroler derivatif ini tidak dapat berdiri sendiri, karena kontroler ini hanya aktif pada waktu transien. Pada waktu transien, pengontrol derivatif

menyebabkan redaman pada sistem sehingga memperkecil lonjakan (*overshoot*)[4].

Pengaruh kontroler derivatif terhadap suatu sistem, yaitu[4]:

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

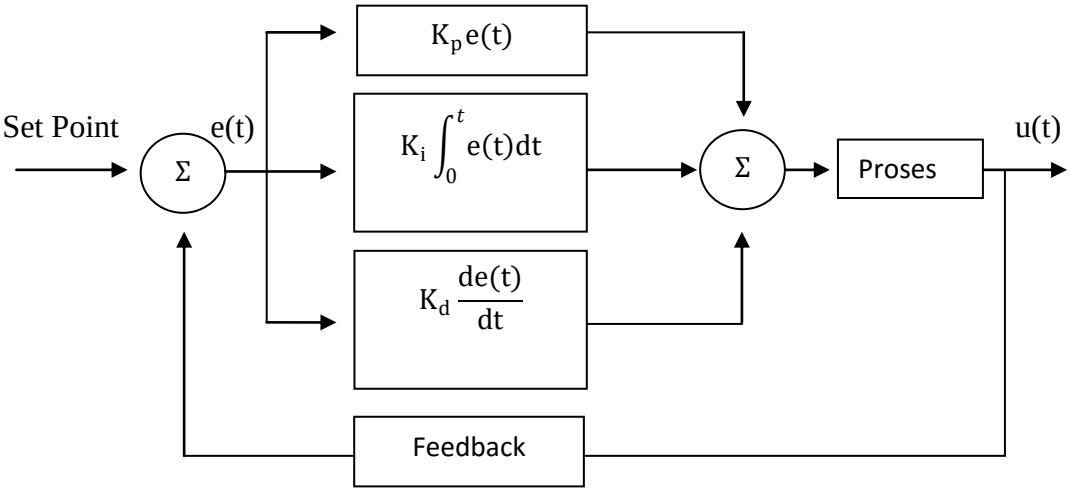
Karakteristik pengontrol derivatif adalah sebagai berikut[6]:

- Pengontrol ini tidak dapat menghasilkan keluaran bila tidak ada perubahan pada masukannya (berupa sinyal kesalahan).
- Jika sinyal kesalahan berubah terhadap waktu, maka keluaran yang dihasilkan kontroler tergantung pada nilai K_d dan laju perubahan sinyal kesalahan.
- Pengontrol derivatif mempunyai karakteristik untuk mendahului, sehingga kontroler ini dapat menghasilkan koreksi yang signifikan sebelum pembangkit kesalahan menjadi sangat besar. Jadi pengontrol derivatif dapat mengantisipasi pembangkit kesalahan, memberikan aksi yang bersifat korektif, dan cenderung meningkatkan stabilitas sistem.

Berdasarkan karakteristik pengontrol ini umumnya dipakai untuk mempercepat respon awal suatu sistem, tetapi tidak memperkecil kesalahan pada keadaan tunaknya. Kerja pengontrol derivatif hanyalah efektif pada

lingkup yang sempit, yaitu pada periode peralihan. Oleh karena itu pengontrol ini tidak pernah digunakan tanpa ada kontroler lainnya[6].

Setiap kekurangan dan kelebihan dari masing-masing pengontrol P, I dan D dapat saling menutupi dengan menggabungkan ketiganya secara paralel menjadi pengontrol proporsional plus integratif plus derivatif (pengontrol PID). Elemen-elemen pengontrol P, I dan D masing-masing secara keseluruhan bertujuan untuk mempercepat reaksi sebuah sistem, menghilangkan *offset* dan menghasilkan perubahan awal yang besar dan mengurangi *overshoot*[6].



Gambar 2.7 Blok Diagram Kontrol PID Analog[7]

Dari blok diagram di atas dapat diperoleh persamaan sebagai berikut[5]:

$$u(t) = K_p e(t) + K_i \int_0^t e(t) + K_d \frac{de(t)}{dt} \quad \text{..... (4)}$$

Karakteristik kontroler PID sangat dipengaruhi oleh kontribusi besar dari ketiga parameter P, I dan D. Penyetelan konstanta K_p, K_i, dan K_d akan mengakibatkan penonjolan sifat dari masing-masing elemen. Satu atau dua dari ketiga konstanta tersebut dapat disetel lebih menonjol dibanding yang lain.

Konstanta yang menonjol itulah akan memberikan kontribusi pengaruh pada respon sistem secara keseluruhan[7].

Dalam pengontrol PID dikenal metode *tuning trial and error* (*Heuristic Methode*), dimana tuning parameter pengendali dimulai dengan hanya menggunakan pengendali P, kemudian baru ditambahkan pengendali I dan terakhir ditambahkan dengan pengendali D. Pemberian nilai parameter disesuaikan dengan karakteristik respon sistem yang diperoleh[8].

2.5. Perangkat Keras

Berikut di bawah ini akan dijelaskan mengenai landasan teori dan fungsi komponen-komponen yang digunakan pada perangkat yang akan dirancang.

a. Sensor Ultrasonik PING Parallax

Sensor Ultrasonik PING Parallax merupakan sensor jarak yang dapat mengukur jarak dari sekitar 2 cm sampai dengan 3 meter. Sensor Ultrasonik PING Parallax juga sangat mudah untuk dikoneksikan dengan mikrokontroler, misalnya seperti BASIC Stamp, Propeller chip, atau Arduino, dan hanya membutuhkan satu pin I/O. Gambar 2.8 merupakan bentuk dari sensor Ultrasonik PING Parallax[9].



Gambar 2.8 Sensor Ultrasonik PING Parallax[9]

Pada modul Sensor Ultrasonik PING Parallax terdapat 3 pin yang digunakan untuk jalur *power supply* (+5V), *ground* dan *signal*. Pin *signal* dapat langsung dihubungkan dengan mikrokontroler tanpa tambahan komponen apapun. Berdasarkan *datasheet*, sensor ultrasonik PING Parallax memiliki spesifikasi sebagai berikut[9]:

- *Supply Voltage – 5 VDC*
- *Supply Current – 30 mA typ; 35 mA max*
- *Range – 2 cm to 3 m (0.8 in to 3.3 yrds)*
- *Input Trigger – positive TTL pulse, 2 uS min, 5 µs typ.*
- *Echo Pulse – positive TTL pulse, 115 uS to 18.5 ms*
- *Echo Hold-off – 750 µs from fall of Trigger pulse*
- *Burst Frequency – 40 kHz for 200 µs*
- *Delay before next measurement – 200 µs*
- *Size – 22 mm H x 46 mm W x 16 mm D (0.84 in x 1.8 in x 0.6 in)*

Sinyal ultrasonik berfrekuensi diatas 20KHz, biasanya yang digunakan untuk mengukur jarak benda adalah 40KHz. Sinyal tersebut di bangkitkan oleh rangkaian pemancar ultrasonik. Sinyal yang dipancarkan tersebut kemudian akan merambat sebagai sinyal atau gelombang bunyi dengan kecepatan bunyi yang berkisar 340 m/s. Sinyal tersebut kemudian akan dipantulkan dan akan diterima kembali oleh bagian penerima ultrasonik. Setelah sinyal tersebut sampai di penerima ultrasonik, kemudian sinyal tersebut akan diproses untuk menghitung jaraknya. Jarak dihitung berdasarkan rumus[10]:

$$S = \frac{340 \ .t}{2} \quad \text{..... (5)}$$

Dimana S adalah jarak antara sensor ultrasonik dengan bidang pantul, dan t adalah selisih waktu antara pemancaran gelombang ultrasonik sampai diterima kembali oleh bagian penerima ultrasonic[10]. Sensor ultrasoni terdiri dari pemancar ultrasonik (*transmitter*) dan penerima ultrasonik (*receiver*).

Pemancar Ultrasonik ini berupa rangkaian yang memancarkan sinyal sinusoidal berfrekuensi di atas 20 KHz menggunakan sebuah *transducer transmitter* ultrasonik. Penerima Ultrasonik ini akan menerima sinyal ultrasonik yang dipancarkan oleh pemancar ultrasonik dengan karakteristik frekuensi yang sesuai.

Sinyal yang diterima tersebut akan melalui proses filterisasi frekuensi dengan menggunakan rangkaian band pass filter (penyaring pelewat pita), dengan nilai frekuensi yang dilewatkan telah ditentukan. Kemudian sinyal keluarannya akan dikuatkan dan dilewatkan ke rangkaian komparator (pembanding) dengan tegangan referensi ditentukan berdasarkan tegangan keluaran penguat pada saat jarak antara sensor kendaraan mini dengan sekat/dinding pembatas mencapai jarak minimum untuk berbelok arah. Dapat dianggap keluaran komparator pada kondisi ini adalah high, Sedangkan jarak yang lebih jauh adalah low. Logika-logika biner ini kemudian diteruskan ke rangkaian pengendali (mikrokontroler).

b. Flame Sensor Modul

Sensor ini merupakan jenis sensor yang digunakan untuk mendeteksi api jarak dekat. Sensor ini dapat mengukur dengan kisaran jarak antara 20 cm (4.8 V) - 100 cm (1V). *Flame sensor modul* sangat sensitif terhadap cahaya dengan panjang gelombang 760 nm – 1100 nm. Sensor ini terdiri dari

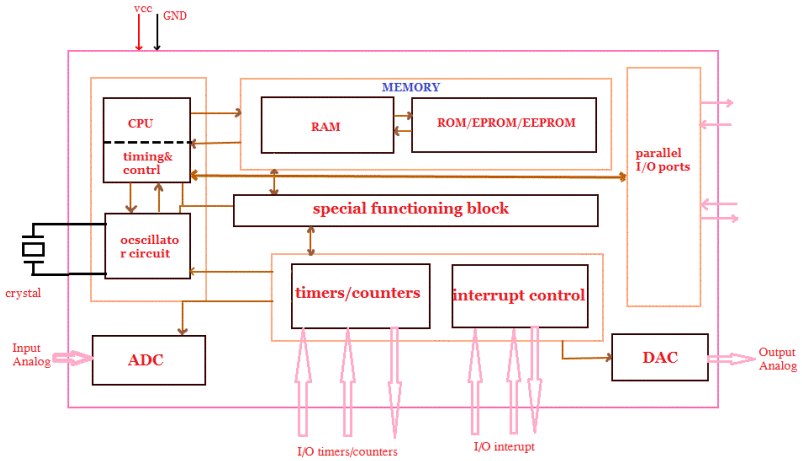
beberapa buah pin, yaitu *power* (3.3V-5V), *ground*, *analog output*, *digital output* (0 dan 1). Berikut bentuk *flame sensor modul* ditunjukan pada gambar 2.9[6].



Gambar 2.9 *Flame Detector Sensor*[6]

c. Mikrokontroler (Arduino MEGA 2560)

Microcontroller adalah sebuah komputer kecil yang terdapat di dalam sebuah rangkaian terintegrasi. *Microcontroller* pada dasarnya terdiri CPU (*Central Processing Unit*), *memory*, *input/output ports*, *timers and counters*, *interrupt controls*, *analog to digital converters*, *serial interfacing ports* dan *oscillatory circuits*[12].



Gambar 2.10 Blok Diagram *Microcontroller*[12]

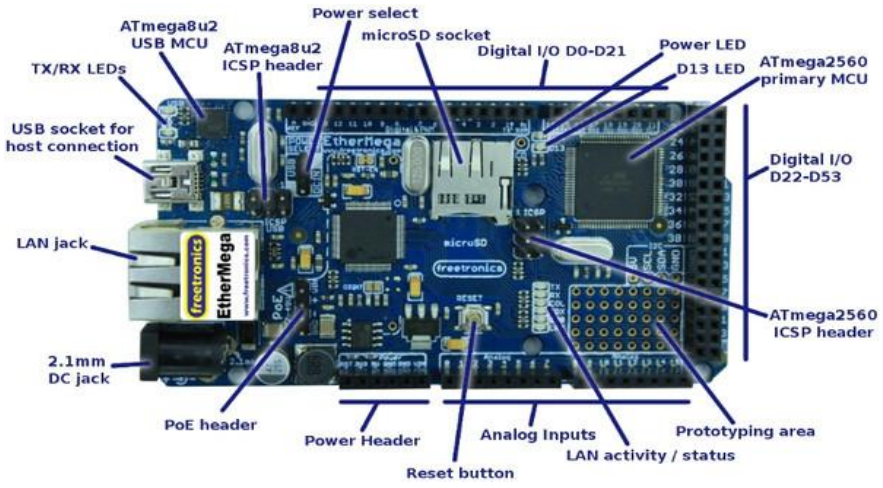
Penjelasan singkat mengenai blok diagram *microcontroller*[12] :

- CPU (*Central Processing Unit*) adalah otak dari sebuah *microcontroller*. CPU bertugas untuk mengambil setiap intruksi dalam bentuk kode dan melakukan *decode* (menterjemahkan intruksi) ke dalam bahasa mesin untuk selanjutnya dilakukan eksekusi. CPU juga bertugas untuk menghubungkan setiap bagian dari *microcontroller* ke dalam sebuah sistem.
- *Memory* berfungsi menyimpan setiap intruksi dan data dari sebuah program. *Microcontroller* biasanya memiliki sejumlah memori seperti SRAM, ROM/EPROM/EEPROM dan *flash memory*.
- *Parallel Input*/*Output Ports* digunakan untuk melakukan *interface* dengan perangkat lain yang dibutuhkan untuk keperluan sebuah sistem seperti LED, LCD, motor servo, ultrasonik, dan lain-lain.
- *Serial Ports* digunakan untuk keperluan berbagai *interface* serial antara mikrokontroler dan perangkat lain seperti halnya *parallel port*.
- *Timers / counters* merupakan salah satu fungsi yang sangat berguna dari mikrokontroler. Mikrokontroler memiliki lebih dari satu *timer* dan *counter*. Operasi utama dari *timer* dan *counter* adalah melakukan *clock function* , modulasi, *pulse generations*, pengukuran frekuensi, membuat osilasi, dan lain-lain.
- ADC (*Analog to Digital Converter*) berfungsi untuk mengkonversi sinyal analog menjadi digital. Konversi ini dilakukan dengan tujuan mendapatkan output berupa sinyal digital, salah satu contohnya adalah pengukuran suhu yang hasilnya ditampilkan pada sebuah LCD.

- *Interrupt control* digunakan untuk melakukan interupsi pada sebuah program ketika dieksekusi. *Interrupt control* memiliki dua jenis, yaitu *interrupt control internal* (memakai intruksi interupsi) dan *interrupt control eksternal* (memakai akses pin interupsi).
- *Special functioning block* merupakan bagian tambahan pada sebuah *mikrokontroller* untuk kebutuhan tertentu. Tidak semua *mikrokontroller* menggunakan *special functioning block.*

Salah satu *microcontroller* yang paling banyak digunakan pada jaman sekarang adalah arduino. Arduino adalah sebuah platform komputasi fisik yang bersifat *open-source* berbasis papan/*board microcontroller* sederhana. *Hardware*-nya menggunakan prosesor *atmel* AVR dan *software*-nya memiliki bahasa pemrograman sendiri yaitu C++ arduino[12].

Salah satu alasan kenapa arduino banyak dipakai adalah karena sistem pengoperasiannya yang cukup mudah. Bahasa pemograman yang mudah dipahami dan perangkat tambahan yang mudah didapatkan menjadi daya tarik tersendiri[12].



Gambar 2.11 Arduino MEGA 2560[13]

Arduino MEGA 2560 adalah modul mikrokontroler dengan IC ATmega2560. Modul ini memiliki 54 digital *input/output* pin yaitu 14 pin untuk *output PWM*, *16-pin analog input*, *4 UART* ,*16 MHz* osilator kristal, koneksi USB, sumber tegangan, *ICSP header*, dan tombol *reset*. memiliki spesifikasi kerja sebagai berikut[14]:

- *Operating 5V*
- *Input Voltage* (rekomendasi) *7-12V*
- *Input Voltage* (batas) *6-20V*
- *Digital I/O Pins54* termasuk *14 pin output pwm*
- *Analog Input Pins 16*
- *DC Current per I/O Pin 40 mA*
- *DC Current for 3.3V Pin 50 mA*
- *Flash Memory 8 KB dari 256 KB* digunakan oleh *bootloader*
- *SRAM 8 KB*
- *EEPROM 4 KB*
- *Clock Speed 16 MHz*

d. *Driver* Motor L298

IC L298 adalah sebuah IC H-bridge yang mampu mengendalikan beban-beban induktif seperti relay, selenoid, motor DC dan motor stepper. IC L298 mempunyai dua buah H-bridge di dalamnya sehingga bisa mengendalikan kecepatan dan arah dua buah motor DC dengan arus 2A setiap H-bridge nya[15].



Gambar 2.12 *Driver* Motor DC IC L298[15]

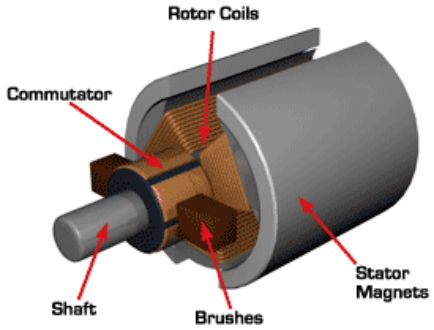
Kelebihan dari *driver* motor DC L298 ini adalah cukup presisi dalam mengontrol motor. Selain itu, kelebihan *driver* motor L298 adalah mudah untuk dikontrol[15].

Untuk mengontrol driver L298 ini dibutuhkan enam buah pin mikrokontroler. Dua buah untuk pin *enable* (satu buah untuk motor satu dan satu buah untuk motor dua). *Input* 1, 2, 3, 4 dihubungkan ke mikrokontroler sebagai input direksi. *Enable A, enable B* dihubungkan ke *port* PWM pada mikrokontroler[15].

e. Motor DC

Motor DC adalah piranti elektronik yang mengubah energi listrik menjadi energi mekanik berupa gerak rotasi. Pada motor DC terdapat jangkar dengan satu atau lebih kumparan terpisah. Tiap kumparan berujung pada cincin belah yang berperan sebagai komutator. Dengan adanya isulator antara komutator, cincin belah dapat berperan sebagai saklar kutub ganda. Atau sering disebut *double pole, double throw switch*[16].

Motor DC bekerja berdasarkan prinsip gaya Lorentz. Prinsip yang menyatakan ketika sebuah konduktor bernilai arus diletakan pada medan magnet, maka sebuah gaya yang dikenal dengan gaya Lorentz, akan tercipta secara *orthogonal* diantara medan magnet dan arah aliran arus[16].



Gambar 2.13 Motor DC[17]

Motor DC memerlukan masukan tegangan yang searah pada kumparan medan untuk diubah menjadi energi mekanik. Kumparan medan pada motor DC dinamakan *stator* (tidak berputar). Sedangkan kumparan jangkar disebut dengan *rotor* (bagian berputar). Jika terjadi putaran pada kumparan jangkar dalam pada medan magnet, maka akan timbul tegangan yang berubah-ubah arah pada setiap setengah putaran. Sehingga menjadi tegangan yang bolak-balik. Bentuk motor paling sederhana memiliki kumparan satu lilitan yang bisa berputar bebas di antara kutub-kutub magnet permanen[16].

f. Catu Daya

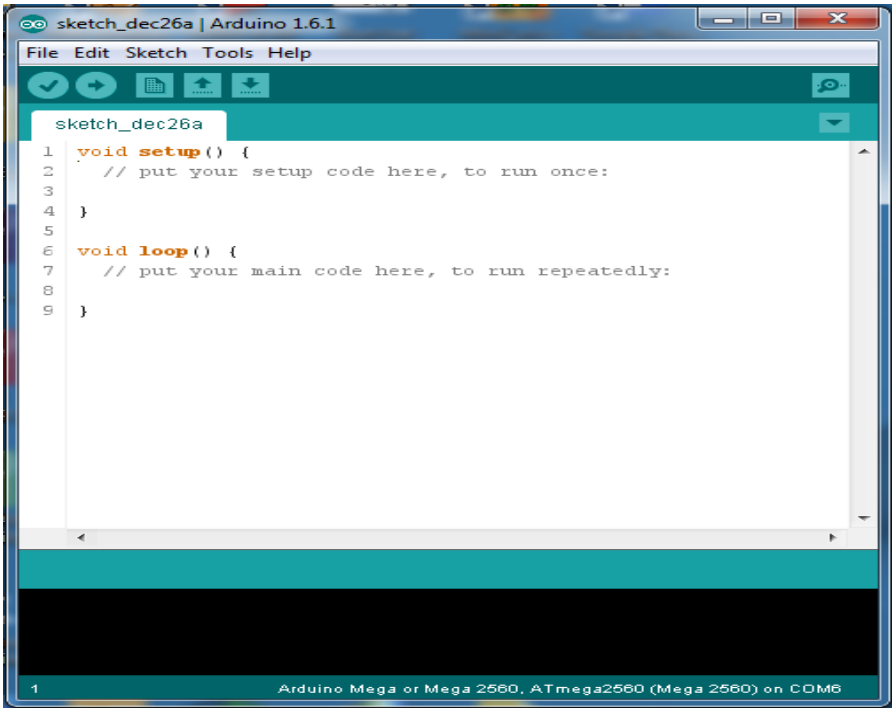
Catu daya memegang peranan yang sangat penting dalam hal perancangan sebuah *mobile robot.* Pemilihan catu daya yang tepat akan menghasilkan *mobile robot* yang bekerja dengan baik. Penentuan sistem

catu daya yang akan digunakan ditentukan oleh banyak faktor, diantaranya[18]:

- Tegangan, setiap aktuator tidak memiliki tegangan yang sama. Hal ini akan berpengaruh terhadap disain catu daya. Tegangan tertinggi dari salah satu aktuator akan menentukan nilai tegangan catu daya.
- Arus, arus memiliki satuan Ah (*Ampere-hour*). Semakin besar Ah, semakin lama daya tahan baterai bila digunakan pada beban yang sama.
- Teknologi Baterai, baterai isi ulang ada yang dapat diisi hanya apabila benar-benar kosong, dan ada pula yang dapat diisi ulang kapan saja tanpa harus menunggu baterai tersebut benar-benar kosong.

2.6. Arduino IDE (*Integrated Development Environment*)

Arduino merupakan *software* editor yang berbasiskan *open source*, yaitu cara penggunaannya dapat dilihat oleh orang banyak. *Software* ini digunakan untuk mikrokontroler Arduino, dengan bahasa yang digunakan yaitu menggunakan bahasa C. *Software* ini dapat menampilkan data *interface* antara mikrokontroler dan *software*. Berikut tampilan program *software* dari Arduino versi 1.6.1[7].



Gambar 2.14 Arduino IDE

Software Arduino ini dapat di-*install* di berbagai *operating system* (OS) seperti LINUX, Mac OS, Windows. *Software IDE* Arduino terdiri dari tiga bagian:

- *Editor* program, untuk menulis dan mengedit program dalam bahasa *processing*. *Listing* program pada Arduino disebut *sketch*.
- *Compiler*, modul yang berfungsi mengubah bahasa *processing* (kode program) ke dalam kode biner karena kode biner adalah satu-satunya bahasa program yang dipahami oleh mikrokontroller.
- *Uploader*, modul yang berfungsi memasukkan kode biner ke dalam mikrokontroller Struktur perintah pada Arduino secara garis besar terdiri dari dua bagian yaitu *void setup* dan *void loop*. *Void setup* berisi perintah yang akan dieksekusi hanya satu kali sejak Arduino dihidupkan

sedangkan *void loop* berisi perintah yang akan dieksekusi berulang ulang selama Arduino dinyalakan.

2.7. Bahasa Pemograman Arduino

Bahasa Pemograman arduino sering disebut C++ arduino karena struktur pemograman dan tipe datanya mirip dengan C++ pada umumnya. C++ arduino terbagi menjadi 3 bagian utama, *structure*, *variables* dan *function*[12].

a. *Structure*

- *Main Structure*

  void setup( ), nama fungsi yang telah disediakan oleh arduino untuk menyatakan fungsi yang akan dijalankan pertama kali. Fungsi ini berisi kode-kode untuk kepentingan inisialisasi.

  void loop( ) , fungsi yang secara otomatis dijalankan oleh arduino setelah fungsi *setup* ( ) dieksekusi. Seluruh kode yang ada di fungsi dengan sendirinya akan diulang terus-menerus. Satu-satunya yang bisa menghentikan eksekusi *loop* ( ) adalah berhentinya input catu daya ke papan arduino.

- *Control Structures*

  If….else, format pemogramannya seperti

  if (kondisi)

  {aksi A}

  Else

  {aksi B}

  Jika kondisi *true* maka output akan melakukan aksi A, jika kondisi *false* output akan melakukan aksi B.

for, contoh format pemogramannya seperti di bawah ini :

for (inisialisasi ; kondisi ; penambahan / pengurangan)

contohnya : for (int i=0; i <= 255; i++){analogWrite(PWMpin, i); delay(10); }

loop akan terus berlangsung sampai kondisi terpenuhi, selama loop berlangsung kondisi akan terus dilakukan penambahan/pengurangan.

- Aturan *Syntax*

Tabel 2.1 Aturan *Syntax*

Syntax	**Keterangan**
;	Akhir dari setiap *statement*
{}	Batas awal dan akhir program.
//	Memberi komentar pendek
/* */	Memberi komentar panjang
#define	Mendeskripsikan sebuah variabel
#include	Memasukan *library*

- Operator Aritmatika

Tabel 2.2 Operator Aritmatika

Operator	**Keterangan**
+	Penjumlahan dan tanda positif.
-	Pengurangan dan tanda negatif
/	Pembagian.
*	Perkalian.
%	Sisa pembagian

- Operator Pembanding

Tabel 2.3 Operator Pembanding

Operator	**Keterangan**
==	Operator sama dengan
!=	Operator tidak sama dengan
>	Operator lebih dari
>=	Operator lebih dari atau sama dengan
<	Operator kurang dari

- Operator *Boolean*

Tabel 2.4 Operator *Boolean*

Operator	Keterangan
&&	Operator “AND”
||	Operator “OR”
!	Operator “NOT”

b. *Variable*

- Tipe Data

Tabel 2.5 Tipe Data C++ *Arduino*

Tipe Data	Keterangan	Kebutuhan Memori
boolean	Hanya dapat digunakan untuk menampung dua nilai saja *true* atau *false*	1 byte
char	Berguna untuk menyimpan sebuah nilai karakter seperti ‘A’,’9’ dan ‘*’.	1 byte
unsigned char	Sama dengan tipe char, tetapi bilangan akan dikodekan dalam bentuk bilangan positif antara 0 sampai dengan 255.	1 byte
byte	Berguna untuk menampung bilangan bulat yang berkisar antara 0 sampai 255.	1 byte
int	Berguna untuk menampung bilangan bulat yang berkisar antara -32768 sampai 32767.	2 byte
unsigned nt	Berguna untuk menampung bilang- an bulat yang berkisar antara 0 sampai 65535.	2 byte
word	Tipe word identik dengan usigned int.	2 byte
long	Berguna untuk menampung bilang-an bulat yang berkisar antara -2,147,483, 648 sampai 2,147,483,647	4 byte
unsigned long	Tipe data ini berguna untuk menampung bilang-an bulat yang berkisar antara 0 sampai 4,294,967,295 (2^{23-1})	4 byte
Float	Tipe data ini berguna untuk menampung bilang-an real. Angka yang disimpan dari -3,4028235 E+38 sampai 3,4028235 E+38.	4 byte

- Konversi Tipe Data

Tabel 2.6 Fungsi-fungsi untuk konversi data

Konversi	Keterangan
char( )	Konversi nilai argument tipe char( ) menjadi tipe char.
byte( )	Konversi nilai argument tipe byte( ) menjadi tipe byte.
int( )	Konversi nilai argument tipe int( ) menjadi tipe int.
word( )	Konversi nilai argument tipe word( ) menjadi tipe word.
long( )	Konversi nilai argument tipe long( ) menjadi tipe long.
float( )	Konversi nilai argument tipe float( ) menjadi tipe float.

c. *Function*

- Digital I/O

Tabel 2.7 Konfigurasi Digital I/O

Digital I/O	Keterangan	Format Program
pinMode( )	Mengkonfigurasi mode pin tertentu pada arduino. Mode yang dipilih bisa *imput* dan *output.*	pinMode(Pin,Mode)
digitalWrite( )	Mengkondisikan *high* atau *low* pada sebuah pin digital.	digitalWrite(Pin,Value)
digitalRead( )	Membaca *high* atau *low* pada sebuah pin digital.	digitalWrite(Pin)

- Analog I/O

Tabel 2.8 Konfigurasi Analog I/O

Analog I/O	Keterangan	Format Program
analogReference( )	Mengkonfigurasi mo- detegangan referensi untuk input analog. Mode yang bisa dipilih adalah DEFAULT, INTERNAL, INTERNAL1V1, INTERNAL 2V56 dan EXTERNAL	analogReference(Type)
analogWrite()	Menulis nilai analog pada sebuah pin analog.	analogWrite(Pin,Value)
analogRead()	Membaca nilai pada sebuah pin analog.	analogWrite(Pin)

- Fungsi Matematika

Tabel 2.9 Fungsi Matematika

Fungsi	Keterangan
min(x,y)	Menghitung nilai minimum dari x dan y.
max(x,y)	Menghitung nilai maximum dari x dan y.
abs(x)	Memberikan nilai absolut x.
pow(x,y)	Memberikan nilai balik berupa x^y. Hasilnya mempunyai tipe float.
sin(x)	Memberika nilai balik berupa sinus x. Argumen x dinyatakan dalam radian. Hasilnya mempunyai tipe float.
cos(x)	Memberika nilai balik berupa cosinus x. Argumen x dinyatakan dalam radian. Hasilnya mempunyai tipe float.
tan(x)	Memberika nilai balik berupa tangent x. Argumen x dinyatakan dalam radian. Hasilnya mempunyai tipe float.

2.8. *Error* Pengukuran

Error adalah sebuah nilai variasi dalam sebuah pengukuran di mana nilai variasi tersebut merupakan selisih yang diperoleh dari nilai terukur dan nilai sebenarnya. Besarnya nilai *error* pada pengukuran dapat dihitung dengan beberapa rumus, yaitu[19].

$$Absolute\ error = |Nilai\ hasil\ pengukuran - Nilai\ sebenarnya| \quad \text{.....(6)}$$

$$Error\ rata - rata = \frac{\sum Absolute\ error}{Jumlah\ pengukuran} \quad \text{.....(7)}$$

$$Persentase\ keberhasilan = \frac{Jumlah\ keberhasilan}{Jumlah\ pengujian}\ x\ 100\% \quad \text{.....(8)}$$

2.9. Membaca Tegangan Analog dengan Arduino

Pembacaan tegangan analog yang masuk pada pin analog Arduino harus dikonversi terlebih dahulu menjadi data digital. Pin analog Arduino dapat menerima nilai hingga 10 bit sehingga dapat mengkonversi data analog menjadi 1024 keadaan (2^10= 1024). Artinya nilai 0 merepresentasikan tegangan 0 volt

dan nilai 1023 merepresentasikan tegangan 5 volt. Proses konversi dari nilai analog menjadi digital dapat dicari dengan persamaan sebagai berikut[20].

$$ADC = Nilai\ analog\ yang\ terbaca \times \frac{5\ volt}{1023} \quad (9)$$

2.10. Rata-rata Hitung (*Mean*)

Rata-rata merupakan ukuran statistik kecenderungan terpusat yang paling sering digunakan. Penghitungan rata-rata dilakukan dengan menjumlahkan seluruh nilai data suatu kelompok sampel, kemudian dibagi dengan jumlah sampel tersebut. Jadi jika suatu kelompok sampel acak dengan jumlah sampel *n*, maka bisa dihitung rata-rata dari sampel tersebut dengan rumus sebagai berikut[21].

$$\bar{x} = \frac{x_1 + x_2 + \cdots + x_n}{n} \quad (10)$$