BAB II
TEORI PENUNJANG

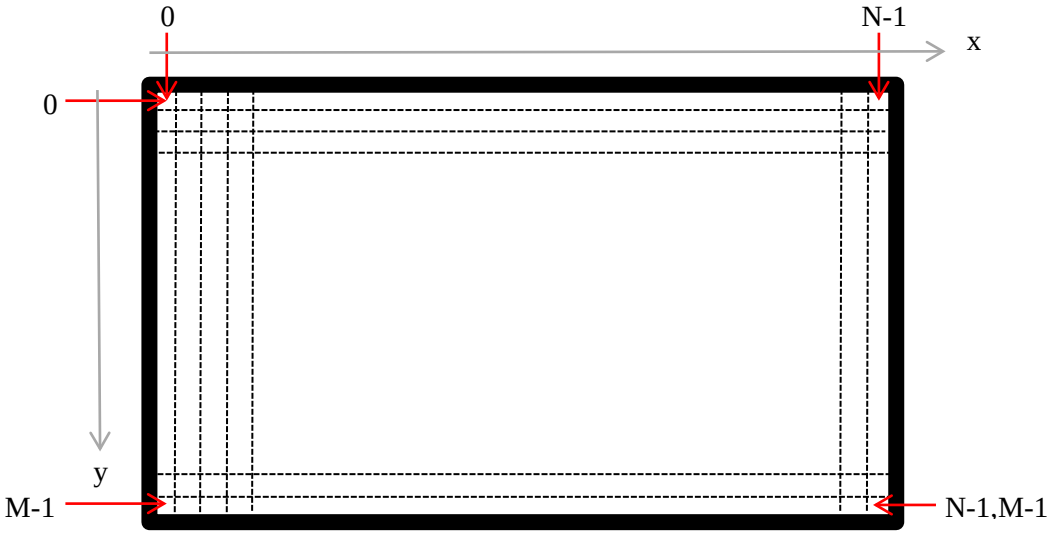
2.1 Pengolahan citra

Pengolahan citra merupakan proses pengolahan gambar untuk mendapatkan informasi yang dibutuhkan pada sebuah citra. Pada sistem ini proses pengolahan citra digunakan untuk mendeteksi informasi ciri dari sebuah objek dan kemudian menggunakan informasi tersebut untuk mendeteksi keberadaan objek yang sama pada gambar yang lainnya.

2.1.1 *Citra digital*

Citra adalah suatu *matriks* dari suatu *pixel (picture elements)* yang diatur dalam suatu kolom dan baris. Terdapat dua jenis citra, yaitu citra analog dan citra digital. Citra analog adalah citra yang dihasilkan dari sistem optik yang menerima sinyal analog dan tidak dapat langsung diolah oleh komputer. Contoh sepeti citra yang tercetak di kertas atau kanvas. Citra digital adalah representasi gambar dua dimensi yang terdiri dari kumpulan titik-titik elemen gambar (*pixel*) [5].

Setiap *pixel* digambarkan sebagai satu kotak kecil. Setiap elemen gambar mempunyai koordinat posisi, sistem koordinat yang digunakan untuk menyatakan citra digital ditunjukkan pada Gambar 2.1.



Gambar 2.1 Sistem koordinat citra berukuran M X N [5]

Dengan sistem koordinat yang mengikuti asas pemindaian pada layar TV standar, sebuah elemen gambar memiliki koordinat berupa sumbu x dan sumbu y,

sumbu x menyatakan posisi kolom sedangkan sumbu y menyatakan posisi baris. Elemen gambar pada pojok kiri atas memiliki koordinat (0, 0) dan elemen gambar pada pojok kanan bawah memiliki koordinat (N-1, M-1) [5].

2.1.2 *Jenis Citra*

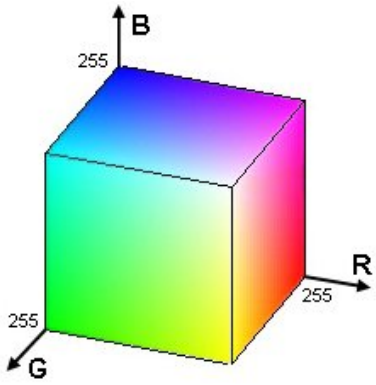
Terdapat tiga jenis citra yang umum digunakan dalam pemrosesan citra. Ketiga citra tersebut yaitu citra berwarna, citra berskala keabuan *(grayscale)* dan citra biner.

2.1.2.1 Citra berwarna

Citra berwarna merupakan jenis citra yang menyajikan warna dalam bentuk komponen R (merah), G (hijau), dan B (biru). Komponen RGB merupakan warna dasar yang dapat diterima oleh mata manusia. Setiap elemen gambar pada citra berwarna membutuhkan data sebesar 3 byte. Setiap warna memiliki intensitas sebesar 8 bit data yaitu dengan nilai minimum nol (0) dan nilai maksimal 255. Dengan demikian, kemungkinan warna yang dapat disajikan mencapai 255 x 255 x 255 atau 16.581.375 warna [2].

Tabel 2.1 Tabel warna RGB

Warna	R (merah)	G (hijau)	B (biru)
Merah	255	0	0
Hijau	0	255	0
Biru	0	0	255
Hitam	0	0	0
Putih	255	255	255
Kuning	0	255	255



Gambar 2.2 Pemetaan warna RGB



Gambar 2.3 Contoh citra RGB, sumber (http://www.amazon.com/Apple-Eve-Juice-8-Count-Aseptic/dp/B000VK4D84)

2.1.2.2 Citra berskala keabuan *(grayscale)*

Citra *grayscale* merupakan citra digital yang hanya memiliki satu nilai kanal pada setiap pikselnya, artinya nilai dari *Red = Green = Blue*. Nilai-nilai tersebut digunakan untuk menunjukkan intensitas warna [2].

Citra yang ditampilkan dari citra jenis ini terdiri atas warna abu-abu, bervariasi pada warna hitam pada bagian yang intensitas terlemah dan warna putih pada intensitas terkuat. Citra *grayscale* berbeda dengan citra biner (hitam putih), dimana pada konteks komputer, citra hitam putih hanya terdiri atas 2 warna saja yaitu “hitam” dan “putih” saja. Pada citra *grayscale* warna bervariasi antara hitam dan putih, tetapi variasi warna diantaranya sangat banyak [2].

Citra *grayscale* disimpan dalam format 8 bit untuk setiap elemen gambar, sehingga kemungkinan warna yang dapat disajikan sebanyak 256 warna. Secara umum citra berwarna dapat dikonversi ke citra berskala keabuan melalui rumus:

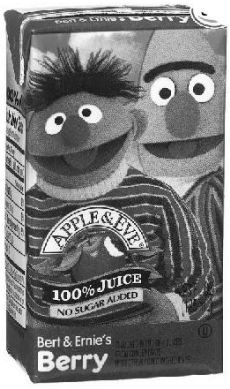
I = a x R + b x G + c x B, dimana a + b + c = 1…….(2.1)

Bobot nilai pada koefisien a, b dan c ditetapkan secara proporsional dengan respon persepsi mata manusia untuk masing-masing warna merah, hijau dan biru, bobot standart yang telah ditentukan oleh NTSC *(National Television System Committee)* pada koefisien a = 0.2989, b = 0.5870, dan c = 0.1140 [2].

Proses perubahan citra berwarna menjadi citra *grayscale* untuk Gambar 2.3 akan menghasilkan gambar seperti pada Gambar 2.5.



Gambar 2.4 Visualisasi 256 aras keabuan



Gambar 2.5 Contoh citra berskala keabuan

2.1.2.3 Citra biner

Citra biner (*binary image*) adalah citra digital yang hanya memiliki 2 kemungkinan warna, yaitu hitam dan putih. Citra biner disebut juga dengan citra W & B (*White & Black*) atau citra monokrom. Hanya dibutuhkan 1 bit untuk mewakili nilai setiap piksel dari citra biner [5].

Pembentukan citra biner memerlukan nilai batas keabuan yang akan digunakan sebagai nilai patokan. *Pixel* dengan derajat keabuan lebih besar dari nilai batas akan diberi nilai 1 dan sebaliknya piksel dengan derajat keabuan lebih kecil dari nilai batas akan diberi nilai 0 [5].

Citra biner sering sekali muncul sebagai hasil dari proses pengolahan citra, seperti segmentasi, pengambangan dan morfologi. Fungsi dari binerisasi sendiri adalah untuk mempermudah proses pengenalan pola, karena pola akan lebih mudah terdeteksi pada citra yang mengandung lebih sedikit warna.

Persamaan untuk binerisasi dapat dilihat pada rumus di bawah ini :

$$f(x,y) = \begin{cases} a_1, & f(x,y) < T \\ a_2, & f(x,y) \geq T \end{cases} \quad \text{..... (2.2)}$$

Keterangan:

- $f(x,y)$ = fungsi gambar
- T = nilai patokan.
- nilai a_1 biasanya 0 dan nilai a_2 = 1.

Proses perubahan citra *grayscale* menjadi citra biner untuk Gambar 2.6(a) akan menghasilkan gambar seperti pada Gambar 2.6(b).



(a) (b)

Gambar 2.6 (a) Contoh citra RGB, (b) Contoh citra biner

2.1.3 *Deteksi Ciri*

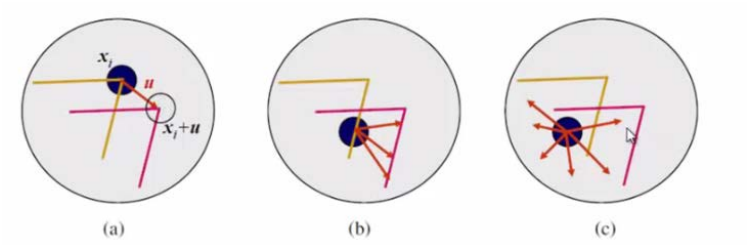
Pada pengolahan citra konsep pendeteksian ciri merupakan metode yang menghitung informasi ciri kemudian memisahkan informasi gambar dan membuat keputusan pada setiap poin dari gambar apakah termasuk dalam kategori ciri

(feature) gambar atau tidak. Hasil dari pengumpulan ciri tersebut akan menjadi bagian dari domain gambar [3].

2.1.3.1 Ciri (feature)

Feature (ciri) adalah bagian yang menarik pada sebuah gambar, dan poin-poin *features* digunakan sebagai langkah awal dalam pembuatan algoritma *computer vision* [3].

Terdapat dua pendekatan yang digunakan untuk mencari poin dari sebuah ciri, pertama untuk mencari sebuah ciri pada gambar dengan akurat metode yang digunakan adalah teknik pencarian lokal *(local search technique)*, seperti teknik *correlation* atau *least square*. Pendekatan kedua adalah dengan mendeteksi ciri gambar pada seluruh gambar kemudian mencocokkannya. Pendekatan pertama cocok digunakan pada gambar yang diambil dari sudut pandang terdekat atau pada gambar yang berganti-ganti secara cepat (seperti video), sedangkan pendekatan kedua lebih cocok digunakan untuk gambar yang berukuran besar atau pada gambar yang perubahannya dapat diatur sesuai dengan keinginan, seperti penyatuan gambar sehingga berbentuk panorama [3].



Gambar 2.7 Jenis-jenis ciri [3]

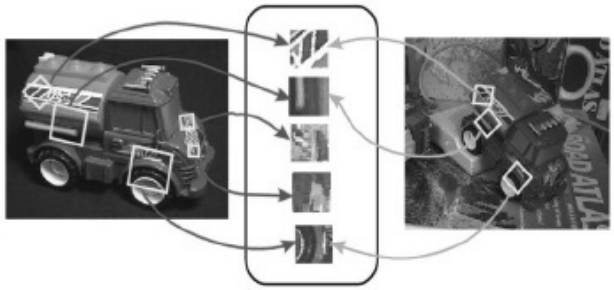
Bagian pada gambar yang dapat disebut sebuah ciri adalah bagian yang memiliki keunikan tersendiri. Pada Gambar 2.7 menunjukkan jenis penentuan ciri pada sebuah gambar, pada Gambar 2.7(a) poin Xi pada garis berwarna kuning memiliki keunikan tersendiri sehingga ketika dicocokkan dengan garis yang berwarna merah poin tersebut hanya cocok dengan satu poin yang berada pada garis berwarna merah, sehingga poin tersebut dapat didefinisikan sebagai ciri yang tepat. Sedangkan pada Gambar 2.7(b) dan Gambar 2.7(c) poin yang terdapat pada garis yang berwarna kuning memiliki banyak kecocokan apabila di cocokkan

dengan garis yang berwarna merah, sehingga poin tersebut didefinisikan sebagai ciri yang tidak tepat.

2.1.3.2 *SIFT*

SIFT (Scale Invariant Feature Transform) merupakan metode yang digunakan untuk mendeteksi dan mendeskripsikan ciri pada gambar. Metode ini dipublikasikan oleh David Lowe pada 1999 [1].

Secara garis besar, algoritma ini menyediakan sebuah metode untuk mendeteksi titik-titik fitur pada sebuah citra 2-D yang dianggap memiliki karakter invariant (kuat dan tidak terpengaruh) terhadap transformasi geometris (scaling, rotation, translation), noise, dan perubahan kadar pencahayaan. Titik-titik ini yang memiliki karakter khusus ini sangat bermanfaat untuk mendeteksi sebuah objek yang terekam dalam berbagai posisi [1]. Gambar 2.8 menunjukkan bagaimana *SIFT* mampu mendeteksi titik-titik fitur pada sebuah objek, kemudian menyimpan data fitur tersebut dan menggunakannya untuk mendeteksi objek yang sama dengan posisi pengambilan gambar yang berbeda.



Gambar 2.8 Proses pendeteksian SIFT

Selain itu, karakteristik lain yang sangat penting pada algoritma *SIFT* adalah posisi relatif dari fitur-fitur yang terdeteksi pada sebuah objek tidak boleh berubah antara satu citra dengan citra yang lain. Sebagai contoh, apabila untuk mendeteksi objek jendela digunakan empat titik sudut jendela, maka algoritma *SIFT* akan bekerja dengan baik meskipun objek jendela direkam dalam posisi yang berbeda. Namun demikian, apabila titik fitur pada daun jendela dilibatkan dalam proses tersebut, ada kemungkinan *SIFT* akan gagal mendeteksi objek jendela dalam citra lain apabila pada citra tersebut daun jendela dibuka atau

ditutup. *SIFT* bekerja dengan menerapkan operasi filtering bertingkat, dimana operasi filtering yang memiliki kompleksitas lebih tinggi akan diterapkan pada titik-titik fitur yang telah diseleksi oleh serangkaian pengujian awal.

Berikut ini adalah langkah-langkah yang dilakukan untuk pendeteksian objek menggunakan metode *SIFT* [1]:

1. Konstruksi *scale space*
2. *Laplacian of Gaussian*
3. Pencarian *keypoints*
4. Eliminasi *keypoin* yang tidak sesuai
5. Kalkulasi *keypoin* yang tidak terpengaruh perputaran
6. Hasilkan *SIFT features*

2.2 Perangkat Keras

Komponen perangkat keras pada sistem ini merupakan komponen perangkat keras yang digunakan untuk mendeteksi dan mengambil objek, perangkat keras pada sistem ini meliputi kamera, mikrokontroler, motor DC, motor servo, driver motor dan catu daya.

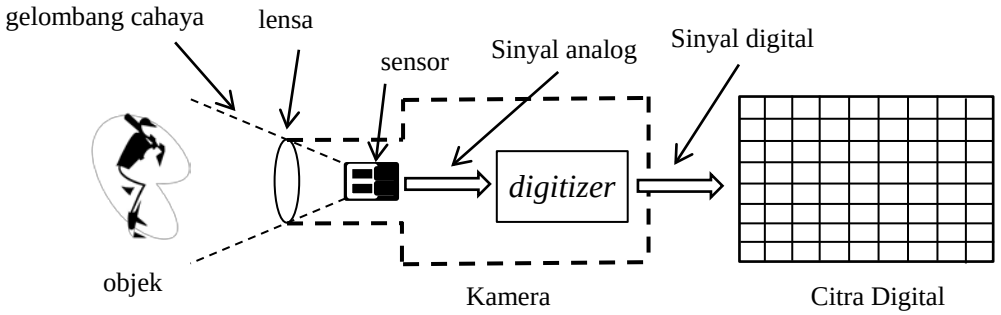
2.2.1 *Kamera*

Kamera adalah alat untuk mengakusisi citra digital. Citra digital yang ditangkap oleh kamera tersebut dapat diproses sehingga didapatkan banyak parameter yang dibutuhkan oleh sistem pengendalian robot, diantaranya parameter posisi, parameter bentuk, bahkan parameter kecepatan [7].

Kamera adalah alat untuk menangkap gelombang cahaya yang dipantulkan oleh objek, sehingga objek tersebut dapat dipetakan dalam bentuk citra. Pada kamera digital, citra yang dihasilkan merupakan citra dalam bentuk digital. Secara umum kamera digital terdiri dari lensa, sensor cahaya, dan *digitizer* seperti yang diperlihatkan pada Gambar 2.9.

Lensa pada kamera berguna untuk memfokuskan cahaya yang diterima sehingga tepat mengenai bidang sensor cahaya. Sensor cahaya yang terdapat pada kamera merupakan kumpulan dari banyak sensor cahaya (phototransistor /

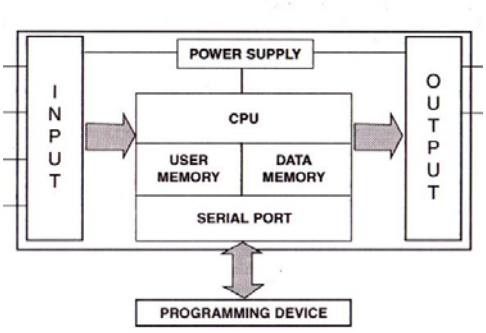
photodiode) yang disusun dalam bentuk matriks. Sinyal analog yang dihasilkan oleh sensor lalu dikonversikan ke dalam bentuk digital oleh *digitizer* sehingga menghasilkan citra digital.



Gambar 2.9 Kamera dan citra digital

2.2.2 *Mikrokontoler*

Mikrokontroler dapat didefinisikan sebagai sistem komputer yang memiliki CPU, *osilator*, *clock*, dan piranti *Input / Output* dalam satu sirkuit terpadu yang digunakan untuk menerapkan sisem kendali. Memori program dalam bentuk flash NOR atau ROM OTP juga sering disertakan pada chip, serta biasanya RAM dengan kapasitas kecil. Mikrokontroller dirancang untuk aplikasi *embedded*, berbeda dengan mikroprosesor yang digunakan dalam komputer pribadi atau aplikasi untuk tujuan umum lainnya [4].



Gambar 2.10 Diagram blok mikrokontroler

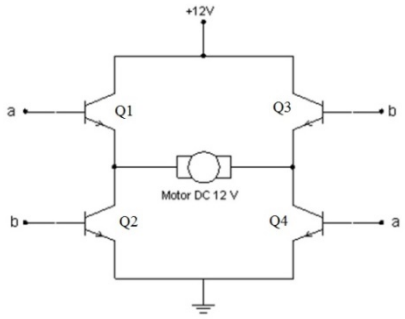
Mikrokontoler digunakan pada produk atau perangkat yang dikontrol secara otomatis, seperti sistem kontrol mesin mobil, perangkat medis, *remote* kendali, alat-alat listrik dan *embedded* sistem lainnya. Kelebihan utama dari mikrokontroler ialah tersedianya RAM dan peralatan I/O pendukung sehingga

ukuran *board* mikrokontroler menjadi sangat ringkas dan lebih ekonomis. Berikut ini adalah komponen-komponen yang terdapat pada mikrokontroler [4]:

1. CPU *(Central Processing Unit)* atau unit pengolahan keseluruhan sistem, yaitu perangkat yang menjalankan perangkat lunak yang telah disimpan pada memori program, mengatur jalur pengiriman data dari atau ke piranti-piranti lain (memori, I/O, dan sebagainya), mengolah data-data yang ada pada perangkat lunak, dan lain sebagainya.
2. ROM *(Read Only Memory)* digunakan untuk menyimpan perangkat lunak yang akan dijalankan oleh mikrokontroler.
3. RAM *(Random Access Memory)* digunakan untuk menyimpan data sementara yang mungkin diperlukan oleh mikrokontroler sewaktu menjalankan perangkat lunak, misalnya menyimpan nilai-nilai pada variable.
4. Piranti I/O digunakan untuk menghubungkan sistem dengan piranti diluar sistem mikrokontroler. Sehingga tugas dari piranti ini adalah untuk menerima data dari luar sistem atau mengirimkan data keluar sistem.
5. *Clock* atau denyut berguna untuk menyinkronkan kerja piranti-piranti dalam sistem mikrokontoler. Pembangkitnya berupa osilator kristal yang telah terpasang pada setiap mikrokontroler.

2.2.3 *Driver Motor*

Driver motor digunakan untuk mengontrol modul penggerak (motor DC), rangkaian driver motor menggunakan konfigurasi transistor yang dikenal dengan istilah *H-bridge.* Konfigurasi ini menggunakan 4 buah transistor NPN atau dua transistor NPN dan dua buah transistor PNP.



Gambar 2.11 Konfigirasi transistor sebagai driver motor

Tabel 2.2 Tabel kebenaran driver motor

Table kebenaran		
b	**a**	Kondisi Motor DC
0	0	Berhenti
0	1	Berputar searah jarum jam
1	0	Berputar berlawanan arah jarum jam
1	1	Berhenti

Pada Gambar 2.11 menunjukkan konfigurasi transistor NPN yang digunakan sebagai pengontrol motor DC. Polaritas arus yang mengalir ke motor DC dapat diatur dengan memberikan logika ke pin a dan pin b. Pada gambar diatas transistor Q1 dan Q2 tidak tersambung ke satu pin, karena apabila kedua transistor tersebut disambungkan ke satu pin maka akan menyebabkan *short circuit* terhadap baterai.

L298 adalah contoh IC yang dapat digunakan sebagai *driver* motor DC. IC ini menggunakan prinsip kerja yang sama dengan prinsip kerja *H-Bridge.* Tiap *H-Bridge* dikontrol menggunakan level TTL yang berasal dari *output* mikrokontroler. IC L298 dapat digunakan untuk mengontrol 2 buah motor DC. IC tersebut dapat mensuplai tegangan hingga 46 VDC dan mampu mengalirkan arus hingga 2 A untuk setiap kanalnya.

Pengaturan kecepatan kedua motor dilakukan dengan cara pengontrolan pulsa (metode PWM *Pulse Width Modulation*) yang dialirkan ke rangkaian *driver* motor oleh modul pengendali. *Duty cycle* PWM digunakana untuk mengatur dan

merubah kecepatan motor DC. Gambar 2.12 menunjukkan bentuk IC L298 yang digunakan sebagai *driver* motor.



Gambar 2.12 IC L298

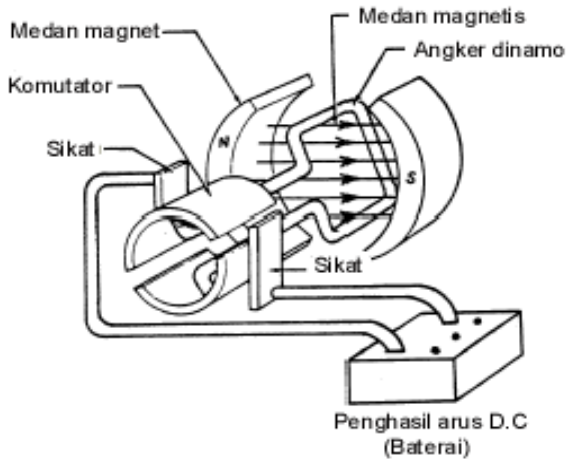
2.2.4 *Motor DC*

Untuk menggerakkan sistem pengambilan makanan atau minuman dibutuhkan modul penggerak, modul penggerak yang digunakan adalah motor DC.



Gambar 2.13 Motor DC

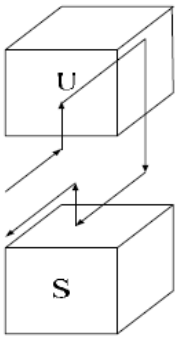
Motor DC merupakan perangkat elektromagnetis yang mengubah energi listrik menjadi energi mekanik. Kumparan medan pada motor DC disebut *stator* (bagian yang tidak berputar) dan kumparan jangkar disebut *rotor* (bagian yang berputar). Jika terjadi putaran pada kumparan jangkar dalam pada medan magnet, maka akan timbul tegangan (GGL) yang berubah-ubah arah pada setiap setengah putaran, sehingga menyebabkan tegangan bolak-balik. Prinsip kerja dari arus searah adalah membalik phasa tegangan dari gelombang yang mempunyai nilai positif dengan menggunakan komutator, dengan demikian arus akan berbalik arah dengan kumparan jangkar yang berputar dalam medan magnet. Bentuk motor DC paling sederhana memiliki kumparan satu lilitan yang bisa berputar bebas di antara kutub-kutub magnet permanen [9].



Gambar 2.14 Diagram motor DC [9]

Catu tegangan DC dari baterai menuju ke lilitan melalui sikat yang menyentuh komutator, dua segmen yang terhubung dengan dua ujung lilitan. Kumparan satu lilitan pada gambar di atas disebut angker dinamo. Angker dinamo adalah sebutan untuk komponen yang berputar di antara medan magnet [9].

Pada motor DC, daerah kumparan medan yang dialiri arus listrik akan menghasilkan medan magnet yang melingkupi kumparan jangkar dengan arah tertentu. Konversi dari energi listrik menjadi energi mekanik (motor) maupun sebaliknya berlangsung melalui medan magnet, dengan demikian medan magnet disini selain berfungsi sebagai tempat untuk menyimpan energi, sekaligus sebagai tempat berlangsungnya proses perubahan energi, daerah tersebut dapat dilihat pada Gambar 2.15 [9].



Gambar 2.15 Medan magnet motor DC [9]

Agar proses perubahan energi mekanik dapat berlangsung secara sempurna, maka tegangan sumber harus lebih besar daripada tegangan gerak yang disebabkan reaksi lawan. Dengan memberi arus pada kumparan jangkar yang dilindungi oleh medan maka menimbulkan perputaran pada motor DC.

Dalam memahami sebuah motor, penting untuk mengerti apa yang dimaksud dengan beban motor. Beban dalam hal ini mengacu kepada keluaran tenaga putar / *torque* sesuai dengan kecepatan yang diperlukan. Beban pada motor DC mempengaruhi penggunaan arus sehingga untuk menggerakkan motor DC harus menggunakan *driver* motor yang dapat mengalirkan arus yang sesuai dengan kebutuhan.

2.2.5 *Motor servo*

Motor servo adalah sebuah motor DC yang dilengkapi rangkaian kendali dengan sistem *closed feedback* yang terintegrasi dalam motor tersebut. Pada motor servo posisi putaran sumbu (axis) dari motor akan diinformasikan kembali ke rangkaian kontrol yang ada di dalam motor servo [8].



Gambar 2.16 Motor servo hitec HS-5645 [8]

Motor servo disusun dari sebuah motor DC, *gearbox,* variabel resistor (VR) atau potensiometer dan rangkaian kontrol. Potensiometer berfungsi untuk menentukan batas maksimum putaran sumbu (axis) motor servo. Sedangkan sudut dari sumbu motor servo diatur berdasarkan lebar pulsa yang pada pin kontrol motor servo [8].

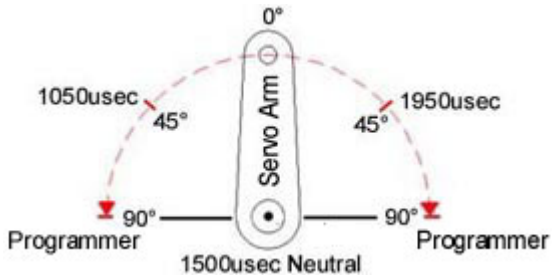


Gambar 2.17 Konstruksi motor servo [8]

Motor servo adalah motor yang mampu bekerja dua arah (CW dan CCW) dimana arah dan sudut pergerakan rotornya dapat dikendalikan dengan memberikan variasi lebar pulsa (*duty cycle*) sinyal PWM pada bagian pin kontrolnya.

Motor Servo Standar 180° Motor servo jenis ini hanya mampu bergerak dua arah (CW dan CCW) dengan defleksi masing-masing sudut mencapai 90° sehingga total defleksi sudut dari kanan – tengah – kiri adalah 180°. Motor Servo Continuous Motor servo jenis ini mampu bergerak dua arah (CW dan CCW) tanpa batasan defleksi sudut putar (dapat berputar secara terus menerus) [8].

Untuk menggerakkan motor servo ke posisi netral maka mikrokontroler harus mengirim pulsa selama 1500uS, sedangkan untuk menggerakkan 45^{o} kearah kanan maka mikrokontroler harus mengirim pulsa selama 1950uS. Gambar 2.18 menunjukkan jumlah waktu pengiriman pulsa yang harus dikirim agar motor servo bergerak sesuai dengan yang diinginkan.



Gambar 2.18 Derajat pergerakan motor servo

2.2.6 *Catu daya*

Catu daya memegang peranan yang sangat penting dalam hal perancangan sebuah robot. Tanpa bagian ini robot tidak akan berfungsi. Begitu juga bila pemilihan catu daya tidak tepat, maka robot tidak akan berfungsi dengan semestinya.

Penentuan sistem catu daya yang akan digunakan ditentukan oleh banyak faktor, diantaranya :

1. Tegangan

   Setiap modul sensor atau *actuator* tidak memiliki tegangan yang sama. Hal ini akan berpengaruh terhadap desain catu daya. Tegangan tertinggi dari salah satu modul sensor atau *actuator* akan menetukan nilai tegangan catu daya.
2. Arus

   Arus memiliki satuan Ah (*Ampere-hour).* Semakin besar Ah, maka semakin lama daya tahan baterai bila digunakan pada beban yang sama.
3. 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 batrei tersebut benar-benar kosong.

Secara umum, ada beberapa jenis dan bentuk baterai yang dapat digunakan sebagai catu daya pada sebuah robot, diantaranya adalah baterai LiPo *(Lithium Polimer).* Baterai LiPo adalah generasi terbaru dari baterai isi ulang, keunggulannya adalah ramah lingkungan, sedang kemampuan lainnya sama persis dengan Battery Lithium Ion. Selain ramah lingkungan baterai LiPo memiliki bobot yang ringan dan tersedia dalam berbagai bentuk dan ukuran.



Gambar 2.19 Baterai lithium polimer

2.3 Perangkat Lunak

Perangakat lunak merupakan aplikasi pendukung yang digunakan untuk memproses pendeteksian dan memprogram mikrokontroler pada sistem. Aplikasi matlab digunakan untuk pemrosesan pendeteksian objek, aplikasi BASCOM-AVR digunakan untuk memprogram mikrokontroler.

2.3.1 *Matrix Laboratory (MATLAB)*

MATLAB (*Matrix Laboratory)* adalah suatu bahasa tingkat tinggi yang digunakan untuk komputasi teknik. Bahasa ini mengintegrasikan proses komputasi, visualisasi, dan pemrograman dengan *environment* yang mudah digunakan dengan mengekspresikan masalah dan solusi kedalam notasi-notasi matematika [11].

Kegunaan umum dari MATLAB diantaranya untuk matematika dan komputasi, pengembangan algoritma, akusisi data, pemodelan dan simulasi, pembuatan prototype, analisis data, eksplorasi, visualisasi dan pengembangan aplikasi termasuk GUI.

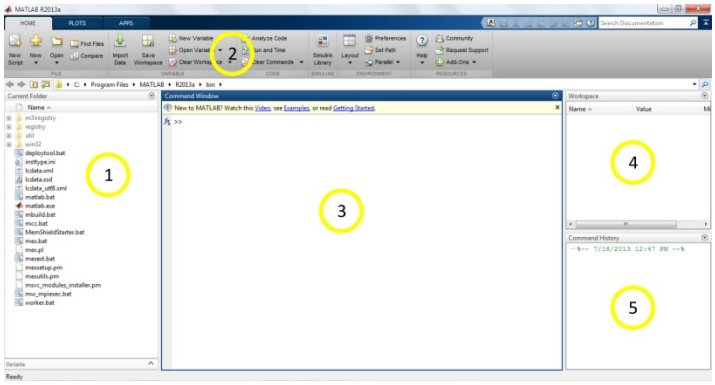
MATLAB adalah sebuah sistem interaktif yang elemen data dasarnya adalah *array* yang tidak membutuhkan pengaturan dimensi. Hal ini memungkinkan penyelesaian banyak maslah komputasi teknik, terutama yang berhubungan dengan formulasi matriks dan vector. Dalam beberapa kasus, dapat juga dilakukan penulisan program dalam bahasa seperti C dan Fortan.

Nama MATLAB merupakan singkatan dari *Matrix Laboratory.* Awalnya, MATLAB dibuat untuk menyediakan akses yang mudah untuk pengembangan *software* matriks dalam proyek LINPACK dan EISPACK. Saat ini, MATLAB menggunakan LAPACK dan BLAS *library* untuk komputasi matriks.

Di lingkunagan universitas, MATLAB digunakan sebagai alat pembelajaran standart untuk tahap pengenalan dan lanjut pada bidang matematika, teknik dan sains. Pada bidang industri MATLAB adalah salah satu alat yang dapat dipilih untuk penelitian, pengembangan dan analisa.

MATLAB juga memiliki fungsi *toolbox* yang memungkinkan pengguna untuk belajar menerapkan teknologi khusus seperti pemrosesan sinyal, sistem kontrol, pengolahan citra, simulasi dan banyak teknologi lainnya.

Ketika memulai MATLAB, desktop MATLAB akan ditampilkan sepaket dengan *Graphical User Interface* yang dapat digunakan untuk mengatur *file*, variable, dan aplikasi yang akan diolah oleh matlab



Gambar 2.20 Tampilan jendela matlab

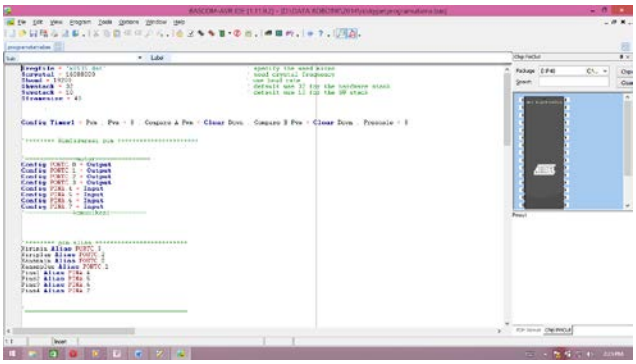
Tampilan desktop Matlab terdiri dari *Current Folder, Toolstrip, Command Window, Workspace, dan Command History*.

1. Current Folder menampilkan lokasi folder yang digunakan
2. Toolstrip berisi fungsi serta tombol-tombol perintah
3. Command Window merupakan jendela perintah utama
4. Workspace berisi tampilan variable-variabel yang sedang digunakan
5. Command History berisi rekaman perintah yang telah dilakukan

2.3.2 *BASCOM-AVR*

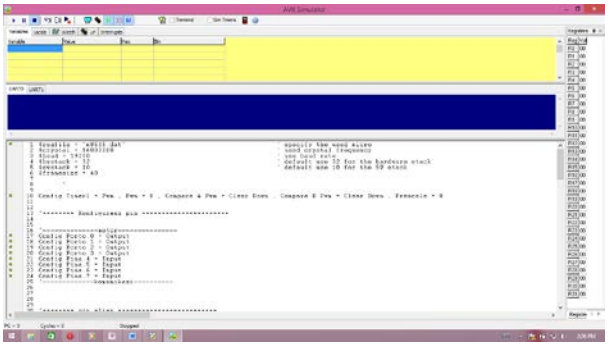
BASCOM-AVR adalah program *basic compiler* berbasis windows untuk mikrokontroler keluarga AVR dan merupakan pemrograman dengan bahasa

tingkat tinggi “BASIC” yang dikembangkan dan dikeluarkan oleh MCS elektronika sehingga dapat dengan mudah dimengerti atau diterjemahkan [1]. Ketika program BASCOM-AVR dijalankan dengan mengklik ikon BASCOM-AVR, maka jendela berikut akan tampil :



Gambar 2.21 Tampilan jendela program Bascom AVR

BASCOM-AVR menyediakan pilihan yang dapat mensimulasikan program. Program simulasi ini bertujuan untuk menguji suatu aplikasi yang dibuat berjalan sesuai dengan yang diinginkan.



Gambar 2.22 Tampilan jendela simulator Bascom AVR

Instruksi yang digunakan pada *editor* Bascom-AVR relatif cukup banyak dan bergantung dari tipe dan jenis AVR yang digunakan. Berikut ini beberapa instruksi-instruksi dasar yang dapat digunakan pada mikrokontroler ATmega.

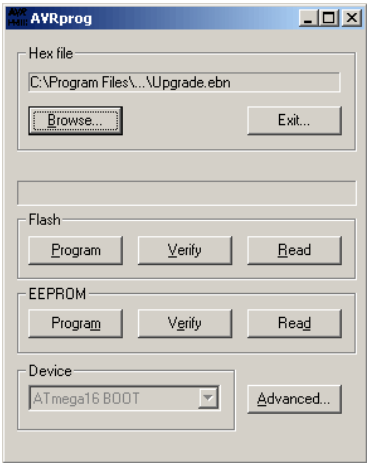
Tabel 2.3 Tabel instruksi pada Bascom AVR

Instruksi	**Keterangan**
DO..LOOP	Perulangan
GOSUB	Memanggil Prosedur

IF…THEN	Percabangan
FOR…NEXT	Perulangan
WAIT	Delay atau waktu tunda dalam satuan detik
WAITMS	Delay atau waktu tunda dalam satuan milidetik
WAITUS	Delay atau waktu tunda dalam satuan mikrodetik
SELECT…CASE	Percabangan
PULSOUT	Membangkitkan pulsa
PULSIN	Menerima atau membaca pulsa yang masuk
SERIN	Menerima atau membaca data serial yang masuk
SEROUT	Mengirimkan data serial
GOTO	Menuju atau loncat ke alamat memori tertentu

2.3.3 *AVRprog*

AVRprog merupakan perangkat lunak yang digunakan untuk memprogram mikrokontroler ATmega. Berikut adalah tampilan jendela program AVRprog.



Gambar 2.23 Tampilan jendela AVRprog