BAB 2

LANDASAN TEORI

2.1 Robot

Menurut Thomas Braunl (2008, p3) robot adalah wujud nyata dari hardware dunia nyata, dimana seorang murid dapat berhubungan dengan robot yang jauh lebih baik daripada software. Tugas yang dipecahkan bersama dengan robot bersifat lebih praktis dan langsung mudah dipahami oleh siswa.

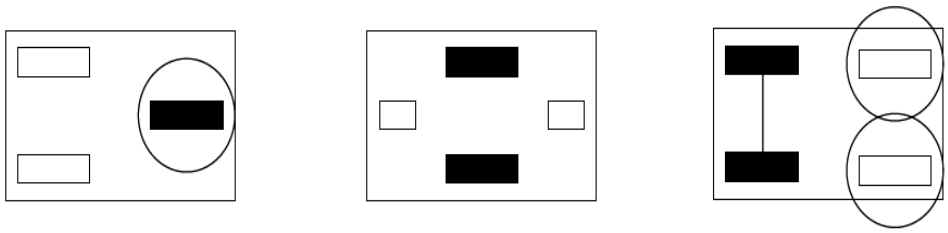
Robotika telah menjalani perjalanan panjang. Khususnya untuk mobile robot, kecenderungan yang sama terjadi seperti yang kita lihat untuk sistem komputer: transisi dari komputing *mainframe* via *workstation* ke *personal computer* (PC), yang mungkin akan terus berkembang dengan perangkat *mobile* untuk banyak aplikasi. Di masa lalu, *mobile* robot dikendalikan oleh sistem komputer yang berat, besar , dan mahal serta tidak bisa dibawa kemana-mana dan harus dihubungkan melalui kabel atau perangkat nirkabel. Namun saat ini, kita dapat membangun *mobile* robot kecil dengan berbagai aktuator dan sensor yang murah, kecil, dan ringan dalam sistem komputer yang terdapat *on-board* di robot . Telah ada peningkatan luar biasa dalam ketertarikan robot *mobile*. Bukan hanya mainan menarik atau terinspirasi oleh cerita-cerita fiksi ilmiah atau film [Asimov 1950], tetapi sebagai alat yang sempurna untuk pendidikan teknik, robot *mobile* yang digunakan hari ini di hampir semua universitas di program sarjana dan pascasarjana di Ilmu Komputer/Teknik Komputer, Teknologi Informasi, Cybernetics, Teknik Elektro, Teknik Mesin, dan Teknik Mekatronika.

Jain dan Saxena. (2011) Robot mampu memahami dan sintesis suara manusia untuk komunikasi. Sebuah unit pengenalan suara dibangun di sekitar prosesor berkecepatan tinggi yang menjamin berbagai macam operasi dari sistem yang akan dilakukan oleh perintah suara. Beberapa perintah

dianjurkan untuk operasi terdaftar sebagai: *START*, *STOP*, MAJU, MUNDUR, KANAN, KIRI, TUTUP, BUKA.

Abdullah dan Awal. (2011) *Robotics Institute of America* (RIA) mendefinisikan robot sebagai *reprogrammable manipulator* multifungsi yang dirancang untuk memindahkan material, suku cadang, peralatan, atau perangkat khusus melalui gerakan yang diprogram untuk variabel kinerja berbagai tugas. RIA membagi robot menjadi empat kelas yaitu perangkat yang memanipulasi objek dengan kontrol manual, perangkat otomatis yang memanipulasi objek dengan siklus yang telah ditentukan, robot diprogram dan servis dikontrol dengan terus-menerus secara *point-to-point* lintasan, dan robot jenis terakhir ini yang juga memperoleh informasi dari lingkungan dan bergerak cerdas dalam merespon.

2.1.1 *Mobile* Robot



Gambar 2.1 *Mobile* Robot

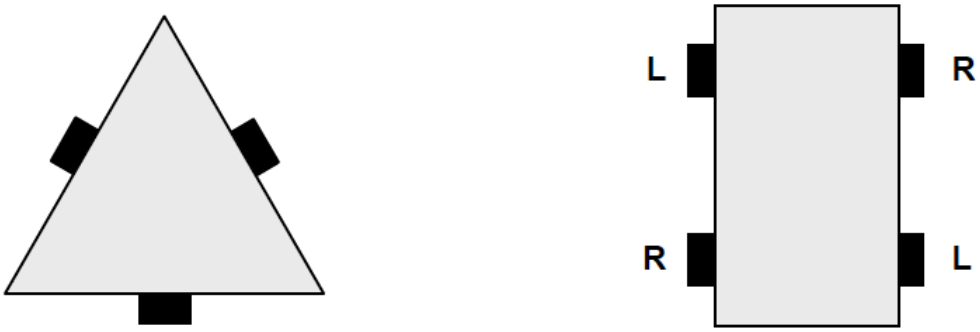
Menurut Thomas Braunl (2008, p5) *Mobile* robot merupakan robot yang dapat bergerak menggunakan roda atau kaki. Pada gambar 2.1 robot beroda yang paling kiri adalah robot dengan roda yang paling dasar. Robot tersebut digerakkan oleh sebuah roda paling depan yang berfungsi sebagai penggerak dan penentu arah. Keunggulannya adalah robot tersebut memiliki dua sistem motor yang berbeda untuk penggerak dan penentu arah. Kerugian yang dimiliki adalah robot tersebut tidak dapat berputar pada satu titik karena roda penggerak tidak terletak di tengah robot.

Robot yang terletak di tengah pada gambar disebut robot *differential drive* merupakan salah satu yang paling umum digunakan dalam desain

mobile robot. Kombinasi dua roda pendorong pada robot ini memungkinkan robot untuk bergerak lurus, berbelok, dan berputar pada satu titik.

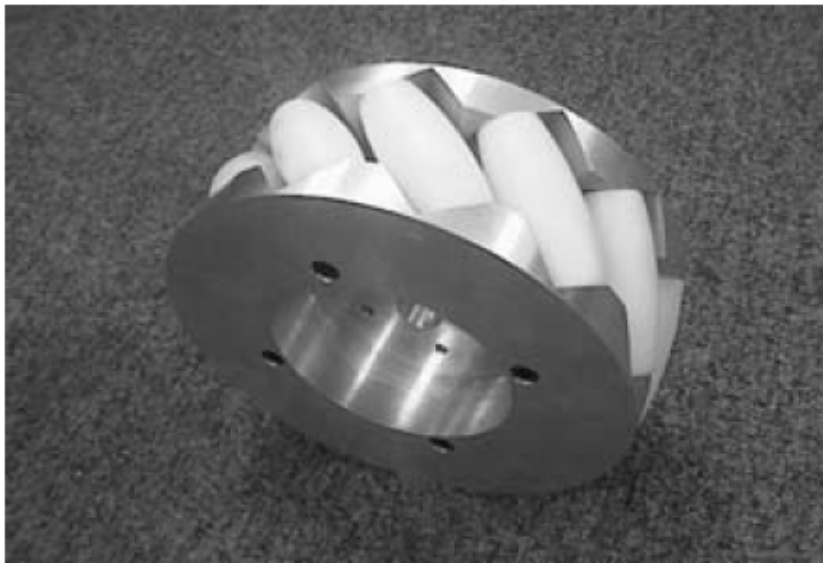
Robot yang terletak di bagian paling kanan pada gambar disebut robot *Ackermann Steering* yang menggunakan satu motor untuk menggerakkan kedua roda belakang dan satu motor untuk mengendalikan arah pada dua roda depan. Semua jenis robot pada gambar di desain sebagai robot yang membutuhkan dua motor, satu motor untuk penggerak dan satu motor untuk mengarahkan robot.

2.1.2 Omni *Directional* Robot

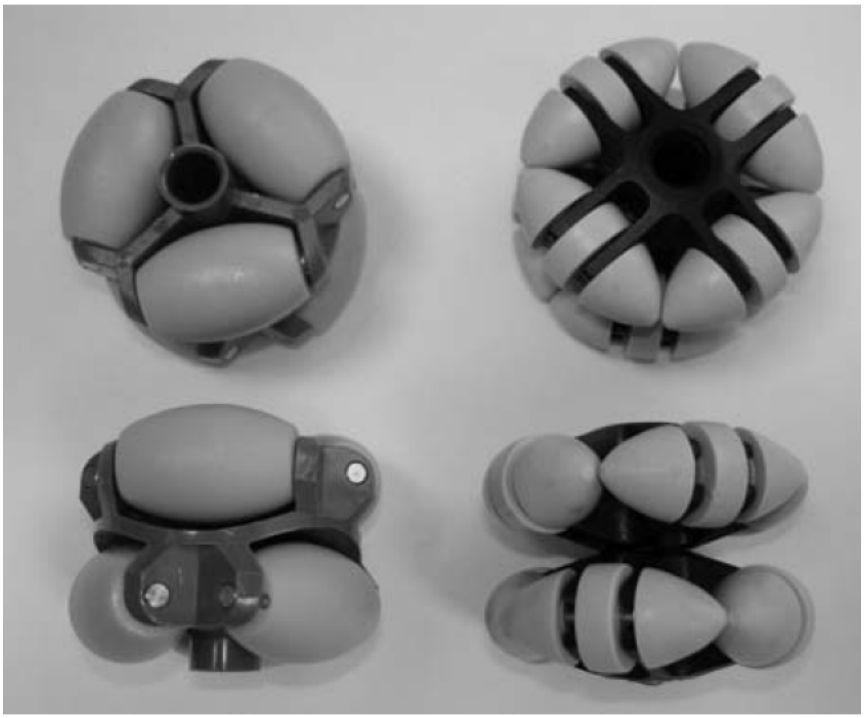


Gambar 2.2 *3-wheel and 4-wheel* omni-*directional vehicles*

Menurut Thomas Braunl (2008, p113-117) Omni *directional* robot adalah robot yang dapat bergerak ke segala arah. Berbeda dengan robot yang dibahas sebelumnya yang hanya bisa begerak maju, berbelok, dan berputar pada satu titik. Robot omni *directional* dapat melakukan gerakan lebih banyak yaitu bergerak maju atau mundur kesamping. Kunci dari omni *directional* adalah *Mecanum wheels* atau roda mecanum. Desain roda mecanum sudah di kembangkan dan dipatenkan oleh Swedish *company* Mecanum AB dengan Bengt Ilon pada 1973. Roda mecanum memiliki beberapa tipe yaitu roda dengan putaran 45 derajat dan 90 derajat.

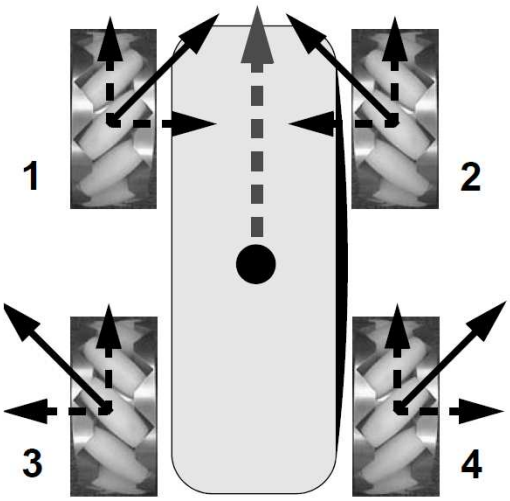


Gambar 2.3 *Mecanum wheel designs with rollers at 45°*



Gambar 2.4 *Mecanum wheel designs with rollers at 90°*

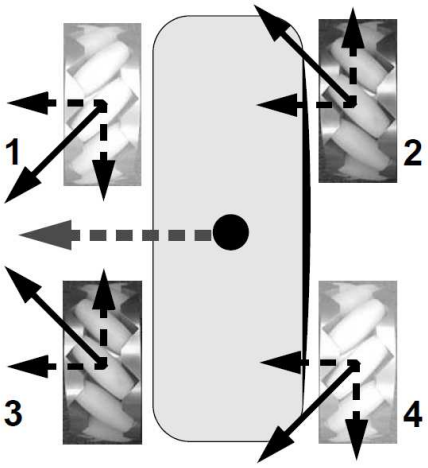
Prinsip cara kerja roda mecanum sebagai berikut :



Gambar 2.5 Roda Mecanum Bergerak Maju

Roda 1 : Bergerak ke serong kanan atas
Roda 2 : Bergerak ke serong kiri atas
Roda 3 : Bergerak ke serong kiri atas
Roda 4 : Bergerak ke serong kanan atas

Pada Gambar dapat dilihat proses bergerak maju sebuah robot yang menggunakan roda mecanum.



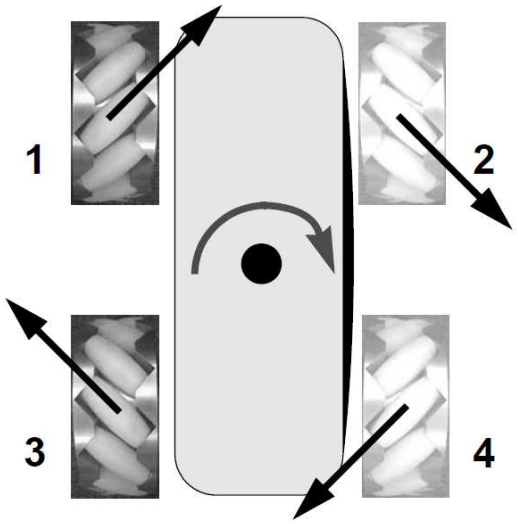
Gambar 2.6 Roda Mecanum bergerak ke samping kiri

Roda 1 : Bergerak ke serong kiri bawah
Roda 2 : Bergerak ke serong kiri atas

Roda 3 : Bergerak ke serong kiri atas

Roda 4 : Bergerak ke serong kiri bawah

Pada Gambar dapat dilihat proses bergerak kesamping kiri sebuah robot yang menggunakan robot mecanum.



Gambar 2.7 Roda Mecanum bergerak Berputar Searah Jarum Jam

Roda 1 : Bergerak ke serong kanan atas

Roda 2 : Bergerak ke serong kanan bawah

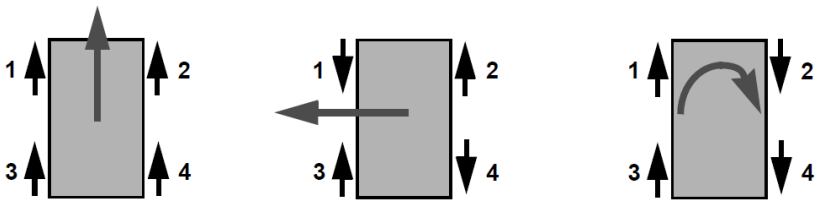
Roda 3 : Bergerak ke serong kiri atas

Roda 4 : Bergerak ke serong kiri bawah

Pada Gambar dapat dilihat proses bergerak berputar searah jarum jam sebuah robot yang menggunakan robot mecanum.

Daftar berikut menunjukkan gerakan dasar, mengemudi maju, mengemudi ke samping, dan memutar di tempat, dengan arah roda sesuai arah gerakan robot :

- Driving forward: all four wheels forward
- Driving backward: all four wheels backward
- Sliding left: 1, 4: backward; 2, 3: forward
- Sliding right: 1, 4: forward; 2. 3: backward
- Turning clockwise on the spot: 1, 3: forward; 2, 4: backward
- Turning counter-clockwise: 1, 3: backward; 2, 4: forward



Gambar 2.8 Gerakan arah roda sesuai gerakan robot

2.2 *Artificial Intelligence*

Artificial Intelligence atau kecerdasan buatan merupakan kecerdasan yang dibuat oleh manusia dan ditanamkan dalam sebuah mesin atau teknologi tertentu sehingga mesin tersebut dapat melakukan suatu pekerjaan seperti yang dapat dilakukan oleh manusia.

Artificial intelligence adalah

- "Kumpulan algoritma yang dapat dengan mudah di kerjakan, pendekatan yang memadai untuk masalah yang ditentukan" (Partridge, 1991)
- "Perusahaan yang membangun sistem simbol fisik yang dipercaya bisa lulus Tes Turing" (Ginsberg, 1993)
- "Bidang ilmu komputer yang mempelajari bagaimana mesin dapat dibuat untuk bertindak cerdas" (Jackson, 1986)
- "Bidang studi yang mencakup teknik komputasi untuk melakukan tugas-tugas yang tampaknya membutuhkan kecerdasan ketika dilakukan oleh manusia" (Tanimoto, 1990)
- "Penyelidikan yang sangat umum dari sifat kecerdasan dan prinsip-prinsip dan mekanisme yang diperlukan untuk memahami atau mereplikasinya" (Sharpies et ai, 1989)
- "Komputer dapat melakukan hal-hal yang tampaknya menjadi cerdas" (Rowe, 1988).

Ada 4 pendekatan mengenai *Artificial Intelligence* (Russell, & Norvig, 2010, p2) yaitu sebagai berikut:

1. Dapat berpikir seperti manusia.
2. Dapat bertindak seperti manusia.
3. Dapat berpikir rasional.
4. Dapat bertindak rasional.

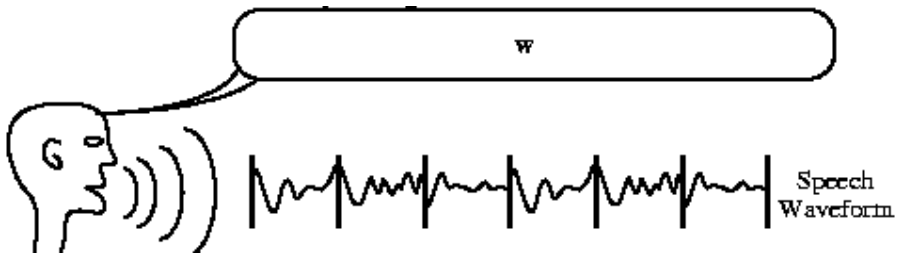
Berdasarkan 4 pendekatan *Artificial Intelligence* di atas maka dapat digolongkan menjadi 2 bagian, yaitu : pendekatan yang berpusat di sekitar manusia dan pendekatan yang berpusat terhadap hal yang rasional.

2.3 *Voice Recognition*

Menurut Baumann. (2008) *Voice recognition* adalah proses pengambilan kata yang di ucapkan sebagai *input* untuk program komputer. Proses ini penting untuk *virtual reality* karena memberikan cara yang cukup *natural* dan *intuitive* untuk mengendalikan simulasi sementara agar *user hand* tetap bebas.

Menurut Rambabu, Naga dan Venkatesh. (2011) *Automatic speech recognition* oleh mesin telah menjadi tujuan dari penelitian untuk waktu yang lama. Pengenal suara yang pertama muncul pada tahun 1952 dan terdiri dari perangkat untuk mengenali suara digit tunggal. *Voice Recognition* adalah proses konversi sinyal akustik, yang ditangkap oleh mikrofon atau telepon, untuk satu set kata-kata. Kata-kata yang dikenali dapat menjadi hasil akhir, seperti untuk aplikasi perintah dan kontrol, entri data, dan persiapan dokumen. Mereka juga dapat berfungsi sebagai input ke proses bahasa yang lebih lanjut untuk mencapai pemahaman pembicaraan. Awalnya pengenalan suara dilaksanakan pada Digital Sinyal Prosesor. Kemudian diimplementasikan pada *Microcontrollers* untuk mengurangi biaya.

Speech recognition adalah suatu teknik yang memungkinkan sistem komputer untuk menerima *input* berupa kata yang diucapkan. Kata-kata tersebut diubah bentuknya menjadi sinyal digital dengan cara mengubah gelombang suara menjadi sekumpulan angka lalu disesuaikan dengan kode-kode tertentu dan dicocokkan dengan suatu pola yang tersimpan dalam suatu perangkat. Hasil dari identifikasi kata yang diucapkan dapat ditampilkan dalam bentuk tulisan atau dapat dibaca oleh perangkat teknologi.



Gambar 2.9 Gelombang suara manusia

Voice Recognition System yang digunakan adalah *SmartVR. SmartVR Module* adalah pengembangan sebuah *platform* untuk *speech synthesis* dan *voice recognition*, berdasarkan *sensory* RSC-4128 *mixed signal processor*. Ukurannya sangat kecil (42 x 72 mm) dan memiliki 2 konektor di tepi dengan 2.54 mm pin spasi. *SmartVR Hardware*:

1. RSC-4128 (dengan ROM *Bootloader*)
2. 512KB *Code / Const Flash*
3. 512KB *Data Flash (Serial)*
4. 128KB *External* RAM
5. *Full access to* RSC-4x I/O *pins*

SmartVR Firmware:

1. VeeEm: *Stack based, no Floating Point,* 16*-bit Virtual Machine*
2. *Modified Harvard architecture*:
   - 2.1. 64KB *Code / Near-Const memory*
   - 2.2. 64KB *Data memory*
   - 2.3. 1MB *Far-Const memory*
3. *Native runtime support for*:
   - 3.1. *Support for most of Sensory's FluentChipTM library functions* (T2SI, SD, SV, SX, RPMSG)
   - 3.2. *Some C Runtime functions (integer math, strings)*
   - 3.3. *Serial Flash and EEprom memory access*
   - 3.4. *Fast SPI access to DevBoard memory-card socket* (SD/SDHC/MMC)
   - 3.5. *Generic* I2C *and* SPI *bus access (up to* 5 SPI *slaves)*
   - 3.6. *Access to general purpose I/O pins*

3.7. *Asynchronous serial interface* (9600 – 230400 *baud*) *on any I/O pin*

4. *Programmable in Standard C language (with extensions)*
5. *Max* 64KB *program* / 64KB *volatile data memory*
6. *Up to* 320KB *read-only data (QuickT2SITM, QuickSynthesisTM data)*
7. *Up to* 512KB *read-write data* (SD, SV, RPMSG)

Gambar 2.10 *SmarVR Module RSC-4128*

SmartVR Development Board:

1. *Wide range of power sources (USB, batteries, external power supply)*
2. *On-board USB / Serial adapter and programmer (upgradeable)*
3. *On-board Microphone (can be disabled for external audio input)*
4. *Selectable audio output (mono4, PWM or DAC with on-board amplifier)*
5. *4 push-button inputs and 4 LED outputs for demos and fast prototypes (can be disabled to connect external circuitry)SD/SDHC/MMC compatible socket for extended storage*

Gambar 2.11 *SmartVR Development Board*

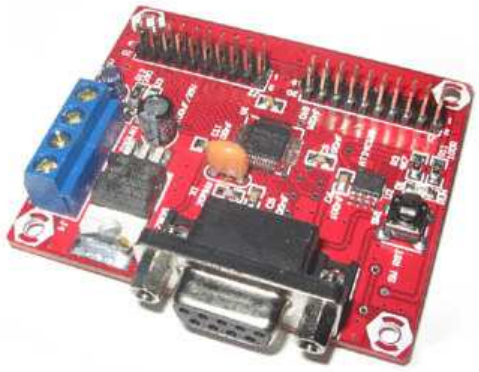
2.4 *Microcontroller*

Microcontroller adalah sebuah *integrated circuit* yang berisi banyak *item* yang sama dengan yang dimiliki *desktop computer*, seperti CPU,

memori, I/O ,dll. Tetapi tidak termasuk perangkat "*human interface*" seperti *monitor, keyboard,* atau *mouse*. *Microcontroller* dirancang untuk mesin *control applications,* daripada interaksi dengan manusia.

Microcontroller yang digunakan adalah DT-BASIC *Mini System Microcontroller*. DT-BASIC *Mini System* merupakan suatu modul *single chip* dengan mikrokontroler *BASIC Stamp*.Modul ini cocok digunakan dalam aplikasi-aplikasi sederhana hingga komplek. Contoh aplikasinya adalah sebagai pengendali *driver motor*, pengendali gerak robot. Spesifikasi *Hardware* :

1. Mikrokontroler *BASIC Stamp* 2P40 *Interpreter Chip* (PBASIC48W/P40)
2. 8 x 2 kbyte EEPROM yang mampu menampung hingga 4.000 instruksi.
3. Kecepatan prosesor 20 MHz *Turbo* dengan kecepatan eksekusi program hingga 12.000 instruksi per detik.
4. RAM sebesar 38 byte (1 2 I/O, 26 variabel) dengan *Scratch Pad* sebesar 128 byte.
5. Jalur input/output sebanyak 32 pin dengan kemampuan *source/sink* arus sebesar 30 mA per pin dan 60 mA per 8 pin.
6. Jumlah instuksi yang didukung : 61
7. Tersedia jalur komunikasi *serial* UART RS-232 dengan konektor DB9.
8. Tegangan input 9 – 12 VDC dan tegangan output 5 VDC.



Gambar 2.12 DT-BASIC *Mini System Microcontroller*.

2.5 *Distance Sensor*

Sensor *ultrasonic* memberikan metode yang mudah pada pengukuran jarak. Sensor ini sangat cocok untuk sejumlah aplikasi yang mengharuskan untuk melakukan pengukuran antara objek yang bergerak atau *stasionary object*. *Interfacing* ke *microcontroller* sangatlah mudah. Sebuah I/O pin yang digunakan untuk memicu ledakan ultrasonik (di atas pendengaran manusia) dan kemudian "*listen*" untuk mengembalikan gema. Sensor mengukur waktuyang dibutuhkan untuk pengembalian gema. Dan mengembalikan nilai ke *microcontroller* sebagai *variable-width pulse* melalui I/O pin yang sama.

Spesifikasi *Hardware* :

1. *Narrow acceptance angle*
2. *Range: approximately* 1 *inch to* 10 *feet* (2 cm to 3 m)
3. 3-*pin male header with* 0.1" *spacing*
4. *Power requirements*: +5 VDC; 35 mA *active*
5. *Communication: positive* TTL *pulse*
6. *Dimensions*: 0.81 x 1.8 x 0.6 in (22 x 46 x 16 mm)
7. *Operating temperature range*: +32 to +158 °F (0 to +70 °C)

Gambar 2.13 *Ultrasonic Distance Sensor*

2.6 *Driver Motor and Motor*

Smart Peripheral Controller/ SPC DC MOTOR merupakan pengontrol motor DC yang menggunakan I2C-*bus* sebagai jalur penyampaian

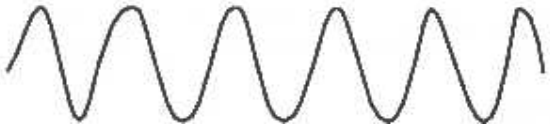
data sehingga dapat lebih menghemat dan mempermudah pengkabelan. SPC DC MOTOR ini dilengkapi dengan prosedur *input* sehingga dapat mengetahui kecepatan motor pada saat tertentu, juga dilengkapi dengan prosedur *brake* yang dapat menghentikan motor secara cepat. Selain itu SPC DC MOTOR dapat digunakan secara paralel. Contoh aplikasi dari SPC DC MOTOR adalah untuk robot, dan sumber gerak lainnya.

Spesifikasi *Hardware* :

1. Kompatibel penuh dengan DT-51 *Minimum System* Ver 3.0.
2. Hanya perlu 2 jalur kabel untuk *interface* dengan mikroprosesor / mikrokontroler lain.
3. Mempunyai 2 buah pengontrol motor DC yang dapat bekerja secara bersama-sama.
4. Masing-masing pengontrol motor DC dilengkapi dengan prosedur *input* dan *brake*.
5. Dapat dikontrol secara I2C-*bus* maupun paralel.
6. Pengaturan kecepatan motor menggunakan metode *Pulse Width Modulation* (PWM).
7. Semua pin–pin kontrol paralel diakses dengan taraf logika TTL.
8. Dilengkapi dengan jumper untuk setting alamat, sehingga bila menggunakan I2C *bus* dapat di-ekspan sampai 8 *board* (16 buah motor DC) tanpa tambahan perangkat keras.
9. Tersedia prosedur siap pakai untuk aplikasi SPC DC MOTOR.

Gambar 2.14 SPC DC Motor

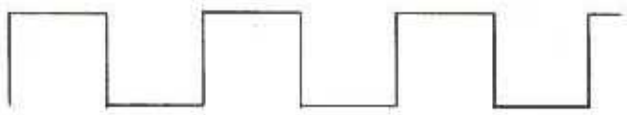
2.7 Sinyal Analog dan Sinyal Digital



Gambar 2.15 Sinyal analog

Menurut Russel dan Norvig. (2010) Sebuah sinyal analog atau sinyal waktu kontinu adalah sinyal dengan variabel apapun yang merupakan sinyal kontinu dalam hal waktu, sehingga untuk setiap nilai waktu dapat diambil nilai-nilai dalam selang kontinu (*a*,*b*), dengan *a* dapat menjadi -∞ dan *b* dapat menjadi ∞. Perbedaannya dengan sinyal digital adalah pada sinyal analog, pergerakan naik turun dari sinyal – walaupun kecil – sangatlah berarti. Sinyal analog menggunakan beberapa sifat dari perantara untuk menyampaikan informasi sinyal.

Kerugian dari penggunaan sinyal analog adalah di setiap sistem terdapat gangguan (*noise*), sebagai contoh adalah variasi acak. Ketika sinyal ditransmisikan dengan jarak jauh, *noise* akan menjadi dominan. Efek dari *noise* akan membuat sinyal menghilang dan menyimpang, dan untuk mengembalikannya sangat susah, karena ketika sinyal dikuatkan untuk mengembalikan bagian sinyal yang lemah, maka *noise* juga akan dikuatkan.



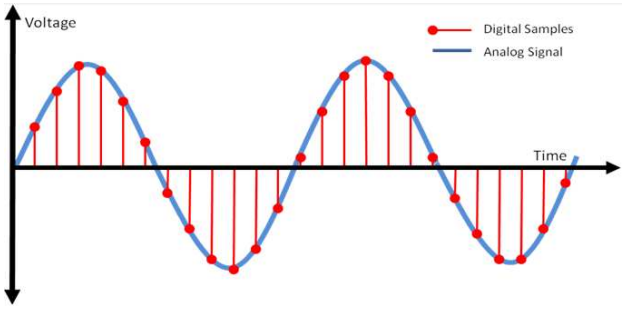
Gambar 2.16 Sinyal Digital

Sinyal digital adalah suatu sinyal waktu-diskrit yang mempunyai himpunan nilai- nilai diskrit. Biasanya, nilai-nilai sinyal dari suatu himpunan terbatas ini seimbang dan karena itu dapat dinyatakan sebagai suatu kelipatan integer dari jarak antara dua nilai berurutan. Suatu sistem digital yang dapat diprogram memiliki keluwesan untuk mengkonfigurasi ulang operasi-operasi sinyal digital secara sederhana dengan mengubah program. Sedangkan konfigurasi ulang suatu sistem analog biasanya menuntut mendesain ulang perangkat keras yang diikuti dengan pengkajian dan pembuktian untuk

melihat apakah sistem itu beroperasi dengan baik.

Pertimbangan keakuratan juga memainkan peranan penting dalam penentuan bentuk prosesor sinyal. Toleransi pada komponen rangkaian analog membuat hal itu sangat sulit bagi pendesain sistem untuk mengontrol ketepatan suatu sistem pemrosesan sinyal analog. Dengan kata lain,suatu sistem digital menyediakan kontrol yang lebih baik untuk syarat keakuratan. Sinyal digital mudah disimpan pada media magnetik tanpa mengalami penurunan atau kehilangan keaslian sinyalnya. Sehingga sinyal-sinyal itu menjadi mudah dipindah- pindah dan dapat diproses secara tidak terhubung (*offline).*

2.8 *Analog To Digital Converter*



Gambar 2.17 *Analog to Digital*

Menurut Russel dan Norvig. (2010) Sebagian besar sinyal-sinyal, seperti sinyal biologis, sinyal seismik, sinyal radar, dan khususnya adalah suara adalah sinyal analog. Untuk memproses sinyal analog dengan alat digital tidak dapat langsung diproses begitu saja. Tentunya, pertama kali adalah harus dikonversi dulu menjadi sinyal digital, yaitu mengkonversi menjadi suatu deret angka yang mempunyai presisi terbatas. Prosedur ini dinamakan konversi analog ke digital (A/D) dengan alat yang dinamakan pengkonversi A/D. Secara Konsepsi kita memandang konversi A/D sebagai proses 3 langkah, yaitu :

1. Pencuplikan (*Sampling*)

   Ini adalah konversi suatu sinyal waktu-kontinu menjadi suatu sinyal waktu diskrit yang diperoleh dengan mengambil cuplikan (*sample*) sinyal

waktu kontinu pada saat waktu diskrit. Diketahui hubungan :

$$x(n) = x_a(nT), \qquad -\infty < n < \infty \quad (2.1)$$

dengan*x(n)* adalah sinyal waktu diskrit yang diperoleh dengan mengambil cuplikan-cuplikan sinyal analog $x_a(t)$ setiap *T* detik. Selang waktu *T* antara cuplikan yang berurutan dinamakan periode pencuplikan dan kebalikannya $1/T = F_S$ dinamakan laju pencuplikan.

2. Kuantisasi

   Ini adalah konversi sinyal yang bernilai kontinu waktu diskrit menjadi sinyal bernilai diskrit, waktu diskrit (digital). Nilai setiap cuplikan sinyal digambarkan dengan suatu nilai terpilihdari himpunan berhingga nilai-nilai yang mungkin. Selisih antara cuplikan *x(n)* yang tidak terkuantisasi dan keluaran $x_q(n)$ yang terkuantisasi dinamakan Galat Kuantisasi (*Quantization Error*).

3. Pengkodean

   Dalam proses pengkodean, setiap nilai diskrit $x_q(n)$ digambarkan dengan suatu barisan biner-b.

2.9 *Hidden Markov Model*

Menurut Mark Gales dan Steve Young. (2007) *Hidden Markov Model* (HMM) adalah suatu model statistic dari sebuah sistem yang diasumsikan sebuah proses *Markov* dengan parameter yang tak diketahui. Kita harus menentukan parameter-parameter tersembunyi (*state*) dari parameter-parameter yang dapat diamati. Parameter-parameter yang ditentukan kemudian dapat digunakan untuk analisis yang lebih jauh,misalnya untuk aplikasi *pattern recognition.*

Dalam HMM, *state* tidak dapat diamati secara langsung, akan tetapi yang dapat diamati adalah variabel-variabel yang terpengaruh oleh *state*. Setiap *state* memiliki distribusi probabilitas atas *token-token output* yang mungkin muncul. Oleh karena itu rangkaian *token* yang dihasilkan oleh HMM memberikan sebagian informasi tentang sekuens *state-state*.

Transisi pada rantai *Markov* yaitu:

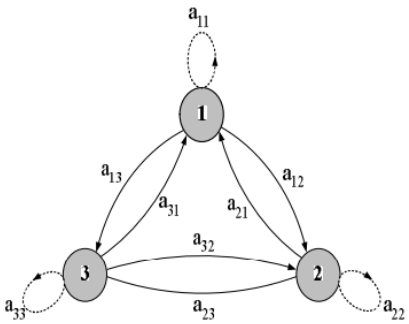
- Transisi dari suatu keadaan tergantung pada keadaan sebelumnya.

$$P[q_t = j | q_{t-1} = i,\ q_{t-2} = k.....] = P[q_t = j \mid q_{t-1} = i] \quad (2.2)$$

- Transisi keadaan bebas terhadap waktu.

$$a_{ij} = P[q_t = j \mid q_{t-1} = i] \quad (2.3)$$

Berikut ini adalah contoh gambar rantai *Markov*.



Gambar 2.18 Rantai Markov

Elemen pada HMM yaitu:

1. N, jumlah keadaan (*state*) dalam model.
2. M, jumlah symbol observasi yang berbeda tiapkeadaan.
3. Distribusi keadaan transisi A = $\{a_{ij}\}$ dengan

$$a_{ij} = P[q_{t+1} = j \mid q_t = j], \qquad 1 \le i, j \le N \quad (2.4)$$

4. Distribusi probabilitas symbol observasi B=$\{b_j(k)\}$ dengan

$$b_j(k) = P(o_t = v_k \mid q_t = j), 1 \le J \le N,\ 1 \le k \le M \quad (2.5)$$

5. Distribusi keadaan awal $\pi = \{\pi_i\}$

$$\pi_i = P[q_t = i] \qquad 1 \le i \le N \quad (2.6)$$

Speech recognition (pengenalan ucapan) merupakan pengenalan sebuah kata berurutan dari ucapan manusia. Otak manusia berfungsi sebagai generator teks yang menghasilkan kumpulan kata *W* . Kumpulan kata berasal dari saluran akustik yang terdiri dari artikulasi pembicara dan proses akustik lainnya yang mengkonversi kumpulan teks kedalam suatu bentuk gelombang akustik yang dapat didengar. Saluran akustik dalam komunikasi verbal bertindak sebagai *data transducer* dan komposer. *Speech recognizer* adalah dekoder yang menampilkan proses pembalikkan dari pesan kebentuk gelombang suara. Oleh karena itu, decoder menunjukkan keputusan akhir

maksimum dari pemeriksaan urutan kata Ŵ sebagai berikut :

$$\widehat{W} = \underset{w}{argmax}\, P(W \mid X) = \underset{w}{argmax}\, P(X \mid W)P(W) \quad (2.7)$$

Dimana P(X|W) adalah nilai dari pemodelan akustik dan P(W) adalah nilai dari model bahasa.

Sistem pengenalan ucapan terdiri dari beberapa komponen dasar, diantaranya:

- *Acoustic feature extraction*: berfungsi untuk mengekstraksi pengenalan suara dari gelombang suara.Biasanya termasuk analisis *cepstral* jangka pendek yang mengolah suatu fitur vector dari frekuensi rendah (10-16) koefisien cepstral untuk setiap 10ms. $X = (x_1, \ldots, x_T)$ untuk merepresentasikan fitur observasi akustik urutan vector.

- *Acoustic modeling*: berfungsi untuk menyediakan pemodelan statistic bagi observasi akustik *X*.
- *Language modeling*: berfungsi untuk menyediakan linguistic dan batasan *grammar* (tata bahasa) untuk urutan teks *W*. *Language model* (model bahasa) sering kali dibuat berdasarkan *N-gram language model*. Sebuah N- gram model bahasa dibentuk dari $P(W_n|W_1, \ldots, W_{n-1})$ dimana peluang dari kata yang diamati W_n diberikan deretan kata $W_1, \ldots, W_{n-1}$
- *Decoding engine*: berfungsi untuk mencari urutan kata terbaik yang diberikan fitur dan model. Untuk pengenalan suara berdasarkan pemodelan HMM, maka *decoding* menggunakan *Viterbi*. Untuk peluang observasi diskrit, kumpulan kata *W* diperoleh dari:

  $$\widehat{W} = \underset{w}{argmax}\, P(X, \Lambda w_Q | W) \quad (2.8)$$

  Dimana Λw_Q adalah *state* terbaik dari W, X dan model Λ. Untuk densitas yang bekelanjutan,

  $$\widehat{W} = \underset{w}{argmax} \log f(X, \Lambda w_Q | W) \quad (2.9)$$

  Dimana rumus diatas berdasarkan nilai *log-likelihood* sepanjang urutan *state* terbaik W_Q.

Pemodelan HMM adalah pemodelan statistik sinyal suara yang populer dan bagus. Diberikan suatu ucapan, misalkan $X = (x_1, x_2, \dots, x_T)$ menjadi fitur urutan vektor terekstraksi dari gelombang suara, dimana x_T menunjukkan suatu pengukuran vektor jangka pendek dan secara konvensional sebuah vektor cepstral.

Suatu permintaan awal N-state rantai Markov diperintah oleh suatu transisi state peluang matriks A= $\{a_{ij}\}$, dimana a_{ij} adalah peluang pembuatan transisi dari state I kestate j. Asumsikan bahwa pada t=0, *state* pada sistem q_0 dikhususkan oleh suatu inisial *state* dengan peluang $\pi_i = P(q_0=i)$. Lalu, untuk setiap urutan *state* $q = (q_1, q_2, \dots, q_T)$, peluang q yang dihasilkan dari rantai *Markov* yaitu:

$$P(q \mid A, \pi) = \pi_{q_0} a_{q_0 q_1} a_{q_1 q_2} \dots a_{q_T - 1 q_T} \qquad (2.10)$$

Anggap pada sistem, ketika di*state* q_T, keluarkan observasi x_T menurut distribusi $b_{qt}(X_t) = P(X_t \mid q_t)$, $q_t = 1,2, \dots, N$. HMM digunakan sebagai distribusi ungkapan suara X lalu didefinisikan sebagai:

$$P\left(X \mid \pi, A, \{b_j\}_{j=1}^{N}\right) = P(X \mid \Lambda) = \sum_{\mathbf{q}} P(X, \mathbf{q} \mid \Lambda)$$
$$= \sum_{\mathbf{q}} P(X \mid \mathbf{q}, \Lambda)\, P(\mathbf{q} \mid \Lambda) = \sum_{\mathbf{q}} \pi_{q_0} \prod_{t=1}^{T} a_{q_{t-1} q_t} b_{q_t}(\mathbf{x}_t) \qquad (2.11)$$

Dimana $\Lambda = \left(\pi, A, \{b_j\}_{j=1}^{N}\right)$ adalah himpunan parameter untuk model. Seperti pada persamaan 2.5, $\{b_{qt}\}$ mendefinisikan distribusi untuk observasi jangka pendek dan A mencirikan sifat dan hubungan diantara *state* yang berbeda dari proses penghasilan suara. Secara umum, N (jumlah total *state*) lebih kecil dari T (waktu selama pengucapan suara). Urutan *state* q menunjukkan derajat stabilitas tertentu diantara *adjacent* q_tS.

Tiga masalah dasar dalam HMM yang harus dikaji ulang yaitu masalah evaluasi, masalah *decoding* dan masalah estimasi. Masalah evaluasi adalah memperkirakan peluang $P(X|\Lambda)$ dari pengamatan fitur suara urutan vector *X* yang diberikan *Hidden Markov Model*. Masalah *decoding* yaitu untuk menemukan urutan *state* terbaik *q* yang optimal dari urutan fitur suara *X*. Karena *state* dalam HMM terhubung kekata dan kelas kata, urutan kata dalam pengucapan suara dapat diidentifikasikan dengan menelusuri *label* kata pada urutan state q.Masalah yang ketiga yaitu estimasi yang diperlukan untuk memperkirakan parameter HMM Λ dari himpunan sampel yang sudah

dilatih. Pendekatan biasa yaitu berdasarkan prinsip *MaximumLikelihood* (ML) dan himpunan parameter model Λ diestimasi sehingga *likelihood* pada pelatihan data menjadi maksimal. Beragam ML yang efisien tergantung pada algoritma yang dikembangkan dalam pengenalan suara untuk HMM, seperti *Baum-Welch* dan algoritma segmental*k-means*. (Chou,Wu.,BiingH. Juang, 2003, p 23-26).

2.10 *Neural Network*

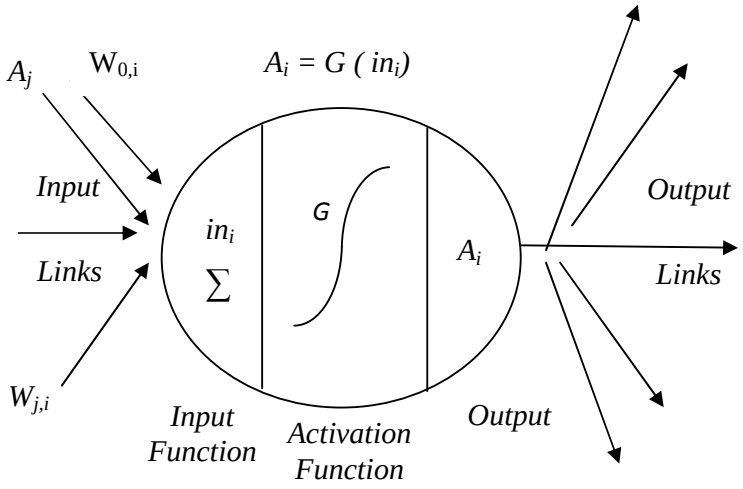
Menurut Demuth, Beale dan Hagan (2009, p1-2), *Neural Network* tersusun atas elemen-elemen dasar yang beroperasi secara paralel dimana elemen-elemen tersebut diilhami oleh sistem saraf biologis manusia. Pada bagian ini, akan dibahas tentang konsep biologi *Neural Network*, konsep dasar permodelan *Neural Network*, proses aktivasi *Neural Network*, model pelatihan *Neural Network* dan algoritma pelatihan *Backpropagation* .

Menurut Russel dan Norvig (2003, p737), *Neural Network* adalah suatu metode komputasi yang meniru sistem jaringan saraf biologi. Metode ini menggunakan elemen perhitungan dasar non-*linier* yang disebut *neuron* yang diorganisasikan sebagai jaringan yang saling berhubungan, sehingga mirip dengan jaringan saraf manusia. *Neural Network* dengan kemampuannya yang luar biasa dibentuk untuk memecahkan suatu masalah tertentu seperti pengenalan pola, klasifikasi proses pembelajaran dan mendeteksi tren yang terlalu kompleks untuk diperhatikan oleh manusia atau teknik komputer lainnya. Layaknya *neuron* biologi, *Neural Network* juga merupakan sistem yang bersifat *fault tolerant* dalam dua hal. Pertama, dalam hal mengenali sinyal *input* yang agak berbeda dari yang pernah diterima sebelumnya. Kedua, dalam hal mampu tetap bekerja meskipun beberapa *neuron*nya tidak mampu bekerja dengan baik. Bila sebuah *neuron* rusak, *neuron* lain dapat dilatih untuk menggantikan fungsi *neuron* yang rusak tersebut.

Neural Network , seperti manusia, belajar dari suatu contoh karena mempunyai karakteristik yang adaptif, yaitu dapat belajar dari data-data sebelumnya dan mengenal pola data yang selalu berubah. Selain itu, *Neural Network* merupakan sistem yang tak terprogram. Artinya, semua keluaran atau kesimpulan yang ditarik oleh jaringan didasarkan pada pengalamannya

selama mengikuti proses pembelajaran atau pelatihan. *Neural Network* yang telah terlatih dan berpengalaman dapat dianggap sebagai pakar apabila dapat digunakan untuk menyediakan proyeksi yang diberi *training* dari situasi yang baru.

Model tiruan sebuah *neuron* dapat dilatih untuk melakukan fungsi tertentu dengan menyesuaikan nilai dari koneksi antar elemen. Umumnya *Neural Network* dilatih sehingga *input* tertentu menghasilkan *output* tujuan yang spesifik. Walaupun masih jauh dari sempurna, namun kinerja dari tiruan *neuron* ini identik dengan kinerja dari sel biologi yang kita kenal saat ini.



Gambar 2.19 Model Tiruan Sebuah *Neuron*

(Sumber :Russel dan Norvig, 2003, p737)

Dimana :

A_j : Nilai aktivasi dari unit *j*

$W_{j,i}$: Bobot dari unit *j* ke unit *i*

in_i : Penjumlahan bobot dan masukan ke unit *i*

G : Fungsi aktivasi

A_i : Nilai aktivasi dari unit *i*

$W_{0,i}$:Bobot bias ke unit *i*

Bias dapat didefinisikan sebagai sebuah bobot koneksi dari suatu unit tambahan dengan nilai yang konstan dan bukan nol. Unit bias terhubung ke setiap unit di *hidden* dan *output layer*, sehingga bobot yang menghubungkan

unit bias dengan unit lainnya dapat dikoreksi, sama seperti bobot lainnya. Tanpa bias, kemampuan belajar dari suatu *neural network* menjadi terbatas. Nilai pada bias memungkinkan suatu *neural network* menghasilkan *output* selain 0 (0.5 jika menggunakan fungsi sigmoid biner) ketika inputnya bernilai 0.

Misalkan ada *n* buah sinyal masukan dan *n* buah penimbang, fungsi keluaran dari *neuron* adalah seperti persamaan berikut:

$$in_i = \sum_j W_{ji} * A_j \qquad (2.12)$$

2.11 Bahasa C

Menurut Kernighan dan Ritchie (1988, pl), bahasa C merupakan bahasa pemrograman yang dipakai untuk keperluan umum. Bahasa C berhubungan erat dengan sistem UNIX dimana ia dikembangkan karena baik sistem maupun kebanyakan program yang dijalankan pada sistem UN1X ditulis dalam bahasa C. Namun bahasa C tidak terikat pada sistem operasi atau mesin apapun. Bahasa C disebut juga bahasa pemrograman sistem karena berguna unruk menulis compiler dan sistem operasi.

Menurut Deitel dan Deitel (2001, p8) bahasa C dikembangkan dari bahasa sebelumya yaitu BCPL dan B.BCPL dikembangkan pada tahun 1967 oleh Martin Richards sebagai sebuah bahasa untuk menulis sistem operasi dan compiler. Ken Thompson merancang banyak fitur dalam bahasa B setelah fitur-fitur tersebut terlibat dalam BCPL dan menggunakan B untuk membuat versi awal dari sistem operasi UNIX di laboratorium Bell dengan menggunakan komputer DEC PDP-7.

Bahasa C dlkembangkan oleh Dennis Ritchie di laboratorium Bell dan diimplementasikan dalam sebuah komputer DEC PDP-11 pada tahun1972. Kemudian C dikenal sebagai bahasa pengembang sistem operasi UNIX. Saat ini hampir semua sistem operasi utama ditulis dalam bahasa C atau C++.

Berdasarkan pendapat Kernighan dan Ritchie (1988, pi), C menyediakan konstruksi *control flow* yang diperlukan untuk program terstruktur : pengelompokan *statement*, pengambilan keputusan ("*if..else..*"), memilih satu dari banyak kasus yang mungkin ("*switch*"),

perulangan dengan pengujian kondisi di awal ("while", "for") atau di akhir ("*do*"), dan keluar dari perulangan secara langsung ("*break*").

Fungsi dapat mengembalikan nilai berupa tipe dasar, struktur, union, atau pointer. Fungsi apapun dapat dipanggil secara rekursif. Definisi fungsi tidak boleh bersarang tetapi variabel boleh dideklarasikan di dalam struktur blok tersebut. Fungsi dari program C dapat berada dalam file-file sumber yang terpisah yang di- compile secara terpisah.Variabel dapat berada dalam sebuah fungsi atau terlihat oleh keseluruhan program.

2.12 Diagram UML (Unified Modelling Languange)

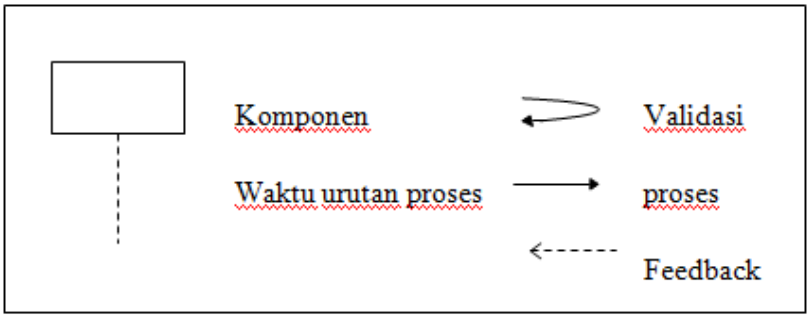
Pengertiam UML oleh Martin Fowler (2003: p1) *the unified modeling language* (UML) adalah keluarga dari notasi grafis, didukung oleh meta-model tunggal, yang membantu dalam menjelaskan dan merancang perangkat lunak sistem, khususnya sistem perangkat lunak yang dibangun dengan menggunakan metode *object-oriented programming*(OOP).

Berdasarkan pendapat yang dikemukakan oleh Martin Fowler diatas maka dapat ditarik kesimpulan bahwa " *Unified Modelling Languange* " (UML) adalah sebuah bahasa yang standart yang terdiri dari kumpulan-kumpulan diagram, grafik atau gambar yang membantu para pengembang dalam menganalisis sistem dan software yang berbasis objek.

Menurut Martin Fowler (2003: p99) usecase adalah suatu teknik untuk menangkap kebutuhan fungsional sistem. Usecase bekerja dengan menggambarkan interaksi khas antara pengguna sistem dan sistem itu sendiri, memberikan narasi tentang bagaimana sistem yang digunakan.

Class Diagram adalah diagram yang menggambarkan *class-class* yang digunakan dalam perancangan program dan hubungan antar *class*.

Sequence diagram adalah diagram yang menunjukkan urutan peristiwa oleh sejumlah objek dalam melakukan tugas tertentu. Pada bagian ini menjelaskan tentang urutan proses *voice recognition* pada robot *guidance* di pasar swalayan. Untuk mempermudah memahami diagram *voice recognition system* ini, terdapat legenda dimana menjelaskan simbol – simbol yang ada didalam diagram proses voice recognition ini.



Gambar 2.20 Legenda untuk Sequence Diagram

2.13 Keterbaruan

Pada pembuatan robot *guide* swalayan dengan *system voice recognition* ini akan dilakukan beberapa keterbaruan yang membuat aplikasi ini berbeda dengan robot *voice recognition* yang pernah dibuat sebelumnya. Robot *voice recognition* yang pernah dibuat sebelumnya antara lain :

1. Pembuatan *voice recognition robotic car* yang berguna sebagai mobil yang dapat mengarahkan kita ke tujuan yang diinginkan.
2. *Android robot with voice recognition* yang digunakan sebagai aplikasi *voice recognition* yang digunakan di OS android.
3. *Voice recognition* pada *mobile phone* yang digunakan untuk menulis *Short Message Service* atau biasa disebut SMS.
4. Tomy I-SOBOT robot yang bisa berbicara, menendang, *split,* karate, dan dapat memainkan *air guitar*.

Berdasarkan robot *voice recognition* yang telah dibuat sebelumnya, maka kali ini penulis akan melakukan beberapa keterbaruan yang akan membedakan robot guide swalayan *voice recognition* kami dengan robot *voice recognition* yang sudah ada sebelumnya :

1. Robot *guide* swalayan dengan *system voice recognition* belum pernah ada di Indonesia, jadi penelitian ini adalah sebuah terobosan baru yang dapat menjadi suatu daya tarik.

2. Robot *guide* swalayan ini menerapkan sistem T2SI (*Text To Speaker Independent*) yang menerapkan penggabungan metode *Hidden Markov Model* (HMM) dan *Neural Network.*
3. Robot *guide* swalayan ini dapat membantu menunjukkan area dari sebuah jenis barang.