Penerapan Logika *Fuzzy* dan *Pulse Width Modulation* untuk Sistem Kendali Kecepatan Robot *Line Follower*

Applying Fuzzy Logic and Pulse Width Modulation for Speed Control System of Line Follower Robot

Ahyar Supani, Azwardi
Teknik Komputer Politeknik Negeri Sriwijaya,
Jalan Srijaya Negara Bukit Besar Palembang Sumatera Selatan
Email: ahyarsupani@polsri.ac.id

Absract

One obstacle faced by line follower robot (LF) is a drived-motor speed control when it turns to right/left following a sharp turn, a medium turn, a less turn and no line. The obstacle is robot LF that always drive a left or right wheel with maximum speed. This obstacle is overcome by applying a fuzzy logic computing values of on/off motor while turning. And then applying the PWM is to set time of on/off motor. The experiment with angles of 90^0 and 45^0 resulted a maximum wheel speed of robot and the other got two speeds; 14,5 % and 43,1 % of the maximum. The experiment with angle of 10^0 resulted a maximum robot wheel speed and the other is 43,1 % of maximum.

***Keywords** : fuzzy, pulse width modulation, robot wheel speed*

Abstrak

Satu kendala robot *line follower* (LF) yaitu kendali kecepatan putaran saat belok mengikuti garis belok tajam, belok sedang, belok sedikit, dan tidak ada garis. Kendala tersebut adalah robot LF selalu menggerakkan satu roda saja kiri atau kanan dengan kecepatan maksimum. Kendala ini diatasi dengan menerapkan logika *fuzzy* untuk mengkomputasi nilai *on/off* motor saat belokan garis tajam, belokan sedang, dan sedikit. Selanjutnya penerapan *Pulse Width Modulation* untuk mengatur sinyal lamanya waktu *on/off* motor. Pengujian dengan sudut 90^0 dan 45^0 menghasilkan satu roda kecepatan maksimum, satu roda lagi mengalami dua kecepatan 14,5 % dan 43,1 % dari maksimum. Pengujian sudut 10^0 satu roda kecepatan maksimum dan satu roda 43,1% dari maksimum.

Kata kunci : f*uzzy, pulse width modulation*, kecepatan roda robot

1. Pendahuluan

Perkembangan teknologi robot dewasa ini banyak membantu pekerjaan manusia seperti yang dipakai di industri sebagai lengan robot (*robot arm*), namun ada juga robot yang dikembangkan dalam kontes robot misalnya robot seni, robot sepakbola, robot penghindar tabrakan sekaligus pencari dan pemadam api. Dalam perkembangannya, robot semakin kian banyak diteliti di perguruan tinggi untuk meningkatkan kepintaran (*smart*) robot [1], penelitian memformulasikan lintasan robot berdasarkan metode formal *Logic Temporal Linier* (LTL) oleh Irvan Lewenusa, Wisnu Ananta Kusuma [2] dan penelitian Widiyanto [3] LTL diterapkan untuk menyusun spesifikasi *mobile robot* berdasarkan kebutuhan rancangan kendali navigasi Kontes Robot Cerdas Indonesia 2006 lintasannya dalam ruangan, navigasinya berdasarkan jarak dan kompas, ada juga penelitian menerapkan logika *fuzzy* untuk rancangan kontroler untuk robot bergerak yang dibangun oleh Shukla dan Tiwari [4] yang dikembangkan adalah hubungan matematika dan geometri antara koordinat 3 dimensi dan 2 dimensi, untuk peningkatan kepintaran robot bergerak dilakukan oleh Supani [5] dengan navigasi gerak robot berdasarkan jarak penghalang dengan robot tersebut. Pengembangan kendali pintar robot lengan juga telah dikembangkan oleh Bachir Ouamri dan Zubir Ahmed [6] yang menyajikan kendali lengan robot Puma 600 menggunakan *Adaptive Neuro Fuzzy Inference System* (ANFIS) yang berbasiskan *Controller* Torsi Terkomputerisasi. Penerapan *controller fuzzy* pada

Received: 30 Januari 2015; Revised: 14 Februari 2015; Accepted: 15 April 2015 ; Published online: 10 Juli 2015

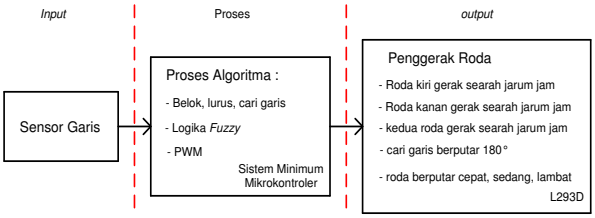
motor DC mesin pembuat gerabah di Desa Nguri Kecamatan Lambeyan Kabupaten Magetan untuk mengatur putaran 100 rpm – 250 rpm motor oleh Resmana et al, [7], dan juga pengaturan motor induksi lebih cepat dan halus [8] dan pengembangan lanjut oleh Andriansyah [9] perancangan robot bergerak berbasis perilaku mahluk hidup menggunakan *Particle Swarm Fuzzy Controller*.

Pada tulisan ini, pengembangan robot LF terus ditingkatkan kepintarannya, karena kelemahan robot LF selama ini adalah tidak ada kendali kecepatan gerak kedua motor ketika belok dan hanya satu roda saja yang bergerak baik belok tajam, sedang, sedikit. Pada tulisan ini telah menerapkan logika *fuzzy* untuk komputasi masukan dari lintasan yang menghasilkan nilai kontrol *on/off* motor, dimana masukan dibagi menjadi beberapa kelompok yaitu "belok kanan tajam", "belok kanan sedang", "belok kanan sedikit", "lurus", "belok kiri tajam", "belok kiri sedang", " belok kiri sedikit", "cari garis". Kemudian *Pulse Width Modulation* (PWM) mengatur lamanya suplai tegangan motor roda robot LF saat belok dan lurus. Luaran selanjutnya mengamati gerak putar roda robot saat belok. Dalam tulisan ini, mikrokontroller yang telah digunakan adalah AT89S52 sebagai mesin robot LF dan *compiler* C [10]. Sedangkan sensor garis masukan yang digunakan adalah *Infra_Red Led* dan *photo* dioda, untuk pengatur penggerak robot LF menggunakan L293D yang dihubungkan ke motor.

2. Sistem robot LF

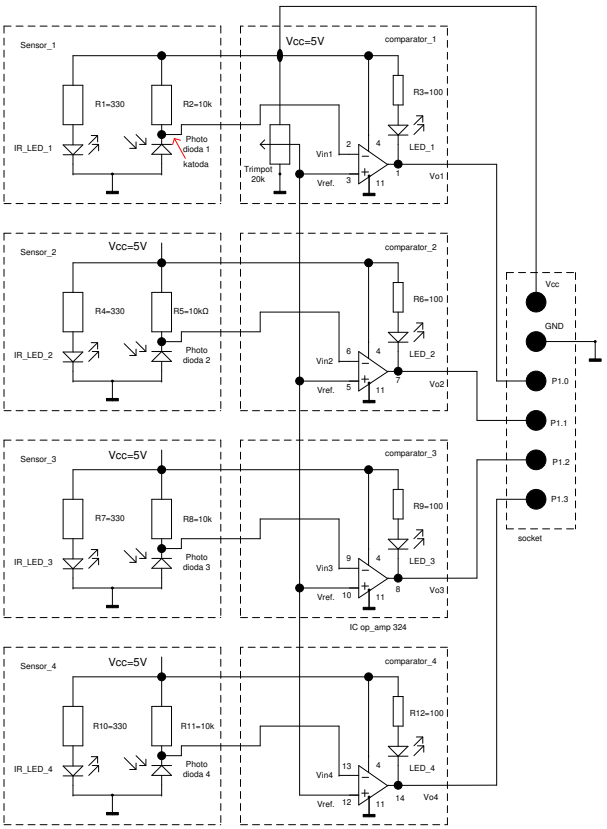
Rancangan diagram blok robot LF telah digambarkan pada Gambar 1, robot LF terdiri atas tiga bagian yaitu masukan (*input*), proses dan keluaran (*output*).

Masukan berupa sensor garis yang menggunakan *infrared* dan *photo diode* yang terdiri atas 4 pasang sensor.



Gambar 1. Diagram blok robot LF

Empat masukan sensor digambarkan pada Gambar 2.



Gambar 2. Empat pasang sensor masukan

Pada Gambar 2 rangkaian sensor dan komparator terdiri atas empat sensor dan empat komparator. Sensor memiliki komponen IR_LED (*infra red*) dan *photo* dioda, dimana fungsi IR_LED memancarkan sinar yang memiliki sinyal dan *photo* dioda berfungsi menerima sinar. Prinsip kedua komponen ini diterapkan pada robot pengikut garis (*line follower*), robot akan mengikuti garis yang diberi warna hitam. Warna hitam ini akan menyerap cahaya, bila garis berbelok maka salah satu sensor tidak mengenai garis mengakibatkan sensor tersebut menerima cahaya pada *photo* dioda dan diteruskan ke masukan sistem robot *line follower*. Sistem robot *line follower* ditanamkan pada mikrokontroller (*embedded system*) yang akan mengatur putaran gerak motor kiri dan kanan.

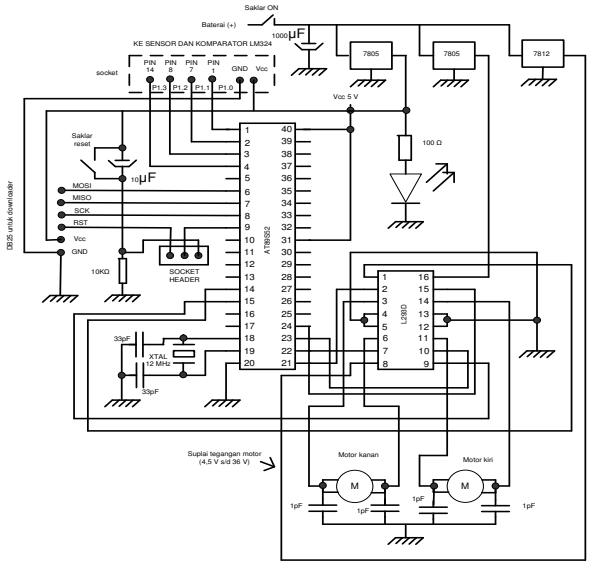
Rangkaian sensor, jika dihalangi maka *photo* dioda tidak mendapatkan cahaya sehingga titik katoda *photo* dioda bertegangan 5 V atau logika 1 pada Gambar 2 sebaliknya tegangan 0 V jika tidak dihalangi. Logika 1 ini diteruskan pada masukan komparator inverting dan dibandingkan dengan tegangan referensi Vref yang tak membalik.

Masukan komparator terhubung langsung dengan katoda *photo* dioda, perubahan logika tinggi dan rendah pada katoda *photo* dioda dibandingkan oleh komparator dengan tegangan referensi trimpot Vref. Adapun persamaan

perbandingan sebagai berikut. Kita ambil contoh sensor 1 dan komparator 1 pada Gambar 2.

- Kondisi Vin1 logika 1 (tinggi), maka Vin1 > Vref., maka Vo1 berlogika 1 (tinggi)
- Kondisi Vin1 logika 0 (rendah), maka Vin1 < Vref, maka Vo1 berlogika 0 (rendah)
- Kondisi Vin1 = Vref, maka Vo1 berlogika 0 (rendah)

Bagian proses adalah untuk memproses algoritma gerak robot LF dengan menerapkan logika *Fuzzy* dan PWM. Logika *fuzzy* dan PWM ditanamkan di mikrokontroler yang berupa perintah program. Bagian proses ini adalah perangkat keras yang berupa sistem minimum mikrokontroler dimana mikrokontroler yang digunakan adalah AT89S52, maka sistem minimumnya adalah AT89S52 seperti Gambar 3.



Gambar 3. Sistem minimum AT89S52, *driver-motor*

IC AT89S52 pada Gambar 3 mempunyai empat buah *port* yang dapat digunakan sebagai masukan dan keluaran. Sebelum menggunakan IC Mikrokontroler AT89S52 ini langkah yang harus dipersiapkan adalah membuat rangkaian sistem minimum AT89S52 yang terdiri atas mikrokontroller dan osilator, osilator dirangkai dengan menggunakan kristal (XTAL) 12 MHz dan dua kapasitor yang masing-masing 30pF. Sistem minimum ini berguna untuk menanamkan kepintaran robot dalam algoritma *Fuzzy* dan PWM yang diilustrasikan dalam program (*source code*). Pada Gambar 3 sistem minimumnya adalah hilangkan IC *driver* L293D dan motor DC, rangkaian reset boleh ada atau tidak. Setelah mendapatkan sebuah rangkaian sistem minimum yang lengkap, sistem minimum AT89S52 ini akan dioperasikan sebagai *input* sekaligus sebagai *output* pada keseluruhan rangkaian mikrokontroler AT89S52. Pin 31 ($\overline{EA}$/Vpp) dihubungkan ke Vcc untuk menggunakan memori internal sebaliknya jika dihubungkan ke *ground* (GND) maka mikrokontroller menggunakan memori eksternal dari alamat 0000h sampai dengan FFFFh.

Driver yang dipakai pada robot LF ini pada Gambar 3 yang bergabung dengan sistem minimum menggunakan tipe IC L293D, yang memiliki 16 pin, sebenarnya *driver-motor* ada dua tipe IC yaitu L293D dengan 16 pin dan L293DD dengan 20 *pin*. Untuk menyederhanakan pemakaian sebagaimana dua *bridge* yang masing-masing pasangan *chanel* dilengkapi sebuah *input enable*. Sebuah *input supply* terpisah yang disediakan untuk logik, yang memperbolehkan operasi pada tegangan rendah dan termasuk dioda *clamp* internal. Perangkat ini cocok untuk pemakaian aplikasi pensaklaran pada frekuensi 5 k Hz.

Tabel 1 adalah tabel kebenaran L293D yang merupakan panduan untuk menggerakkan motor.

Tabel 1. Tabel kebenaran *drive-motor* L293D

Enable 1=pin 1, enable 2= pin9

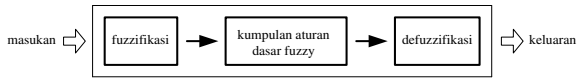
in1	in2	in3	in4	aksi motor	
				kanan	kiri
1	0	0	1	cw	cw
0	0	0	1	*off*	cw
1	0	0	0	cw	*off*
0	0	0	0	*off*	*off*
0	1	1	0	ucw	ucw
1	0	1	0	cw	ucw
0	1	0	1	ucw	cw

keterangan: cw:*clockwise*, ucw:*unclockwise*, off:motor tidak bergerak

3. Algoritma logika *fuzzy* untuk robot LF

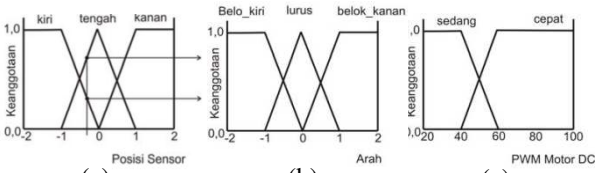
Lofti Zadeh mengembangkan *logika fuzzy* pada tahun 1964, dasar pemikirannya adalah tidak ada keadaan yang hanya selalu bernilai "benar" dan "salah" atau "*on*" dan "*off*", tetapi ada gradasi nilai diantara dua nilai ekstrim tersebut. Dengan memperhatikan kenyataan ini, kita memerlukan penggeseran skala variabel yang dapat diukur sebagai bagian dari "*on*" dan sebagian dari "*off*" atau sebagian besar "benar" dan sebagian "salah". Teori himpunan klasik berdasarkan pada logika ekstrem yang menetapkan objek sebagai anggota atau bukan anggota himpunan. Sebaliknya, pada logika *fuzzy*, suatu objek dapat menjadi anggota pada banyak himpunan dengan derajat keanggotaan berbeda-beda pada masing-masing himpunan. Derajat keanggotaan pada suatu himpunan didasarkan pada skala 0 sampai dengan 1 dan menetapkan 1 sebagai keanggotaan lengkap dan 0 sebagai tidak ada keanggotaan.

Pelopor aplikasi logika *fuzzy* dalam bidang kontrol, yang merupakan aplikasi pertama dan utama dari logika *fuzzy* adalah Prof. Ebrahim Mamdani dkk dari *Queen Mary College* London. Penerapan kontrol logika *fuzzy* secara nyata di industri banyak dipelopor para ahli dari jepang, misalnya Prof. Sugeno dari Tokyo *Institute of Technology*. Aplikasi logika *fuzzy* hampir tak terbatas, misalnya untuk kontrol proses, proses produksi, robotika, manajemen skala besar, teknik sipil, kimia, transportasi, kedokteran maupun ekonomi. Pengaturan (*control*) sistem non linier yang mengandung sejumlah informasi padat memerlukan pengintegrasian sistem secara cepat dan dapat diterapkan dengan menggunakan logika *fuzzy*. Suatu keluaran dihitung berdasarkan nilai keanggotaan yang diberikan oleh masukkan sesudah dikonfigurasikan dalam kumpulan aturan *fuzzy*. Sebelum menjadi keluaran sistem, sistem memerlukan tiga transformasi untuk masukan sistem Gambar 4.



Gambar 4. Sistem himpunan *fuzzy*

Fuzzifikasi adalah proses dekomposisi suatu masukan dan atau keluaran sistem kedalam satu atau lebih himpunan *fuzzy*. Fungsi keanggotaan yang dapat digunakan berbentuk macam-macam jenis kurva, tetapi bentuk segitiga pada Gambar 5 berikut adalah bentuk paling umum yang digunakan untuk sistem pengaturan.



(a) (b) (c)

Gambar 5. Himpunan *fuzzy*

Gambar 5 diatas menunjukkan suatu sistem himpunan *fuzzy* untuk system navigasi *mobile robot*, dengan satu masukan (a) dan dua keluaran (b) dan (c).

Fuzzifikasi adalah proses pembuatan besaran *fuzzy* dari besaran *crisp* yang dapat dilakukan secara sederhana, yakni dengan menandai banyaknya besaran yang dianggap *crisp* dan tertentu. Sebenarnya, tidak semua besaran tertentu, tetapi ada besaran yang tidak tentu. Jika ketidaktentuan muncul karena ketidakpresisian, kerancuan, atau ketidaksengajaan, maka kemungkinan besarannya adalah *fuzzy* dan dapat dinyatakan oleh fungsi keanggotaan.

Setelah masukan dan keluaran didekomposisikan ke dalam himpunan *fuzzy*, kita memerlukan basis aturan yang mengatur tingkah laku sistem tiap kombinasi masukan. Masing-masing aturan terdiri atas satu kondisi dan satu tindakan. Kondisi diintepretasikan dari masukkan himpunan *fuzzy* dan tindakan ditentukan oleh keluaran himpunan *fuzzy*. Suatu himpunan aturan yang mempresentasikan semua kombinasi masukkan bisa di-*set up* dalam suatu matriks yang disebut *Fuzzy Associative Memory* (FAM) atau system *fuzzy* berbasis aturan.

Aturan suatu sistem logika *fuzzy* sesungguhnya disusun sebagai suatu aturan yang mewakili pengetahuan sistem tersebut. Agar dapat menyatakan pengetahuan, pengaturan berbasis *fuzzy logic* menggunakan variabel linguistik dalam menuliskan aturan yang diperlukan. Pada kecerdasan buatan, ada berbagai cara untuk mempresentasikan pengetahuan. Bentuk dari *Switch-case* merupakan pernyataan yang dirancangan khusus untuk menangani pengambilan keputusan yang melibatkan sejumlah atau banyak alternatif penyelesaian. Pernyataan *switch-case* ini memiliki kegunaan sama seperti *if–else* bertingkat, meskipun *Switch* didesain untuk mengganti *If-Else*, akan tetapi *Switch* memiliki batasan:

1. Data yang diperiksa haruslah bertipe *Integer* (*int*) atau Karakter (*char*).
2. *Range* data yang diperiksa bernilai 0 s/d 255.

Bentuk penulisan perintah ini sebagai berikut :

```
Switch (value)
{
case constanta 1// akan dicocokkan dengan isi value: statement 1; //pernyataan yang akan di kerjakan jika value cocok dengan salah satu data dari constanta break; //perintah untuk mengakhiri statement....etc
default : statement x;// bersifat optional. dieksekusi jika value tidak cocok dengan salah satu constanta yang tersedia}
```

Secara umum, ada tiga bentuk umum untuk setiap variabel linguistik, yaitu :

a. Pernyataan penunjukan (*assignment statement*)

   x = kiri warna = biru
   x adalah tidak besar dan tidak kecil

b. Pernyataan kondisional (*conditional statement*)
   Keluaran Suhu = panas
   Putaran motor = cepat

c. Perrnyataan bukan kondisional (*Unconditional statement*)

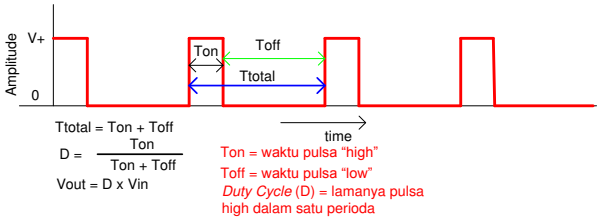
JIKA sensor kiri MAKA belok kiri DAN putaran motor sedang, *IF x is large THEN y is small ELSE y is not small*

Pada pengaturan suatu sistem, keluaran pengendali haruslah suatu nilai diskrit. Kita memerlukan defuzzifikasi untuk mengubah hasil *fuzzy* kedalam nilai keluaran yang tepat. Nilai keluaran dihitung dengan menjumlahkan hasil perkalian keanggotaan himpunan masukkan untuk tiap masukkan dengan nilai keluaran. Hasil perkalian dibagi dengan jumlah dari perkalian keanggotaan himpunan masukkan untuk tiap masukkan.

Defuzzifikasi adalah perubahan dari suatu besaran *fuzzy* ke suatu besaran numerik, sedangkan fuzzifikasi adalah perubahan dari suatu besaran numerik ke suatu besaran *fuzzy*. Keluaran proses *fuzzy* dapat berupa satuan logika dari dua atau lebih fungsi keanggotaan *fuzzy* dan didefinisikan dalam himpunan semesta keluaran.

4. Algoritma *pulse width modulation* (PWM)

Pulse Width Modulation (PWM) secara umum [11] adalah sebuah cara memanipulasi lebar sinyal yang dinyatakan dengan pulsa dalam suatu perioda seperti Gambar 6, untuk mendapatkan tegangan rata-rata yang berbeda. Beberapa contoh aplikasi PWM adalah pemodulasian data untuk telekomunikasi, pengontrolan daya atau tegangan yang masuk ke beban, regulator tegangan, *audio effect* dan penguatan, serta aplikasi-aplikasi lainnya. Aplikasi PWM berbasis mikrokontroler biasanya berupa pengendalian kecepatan motor DC, Pengendalian Motor Servo, Pengaturan nyala terang LED.



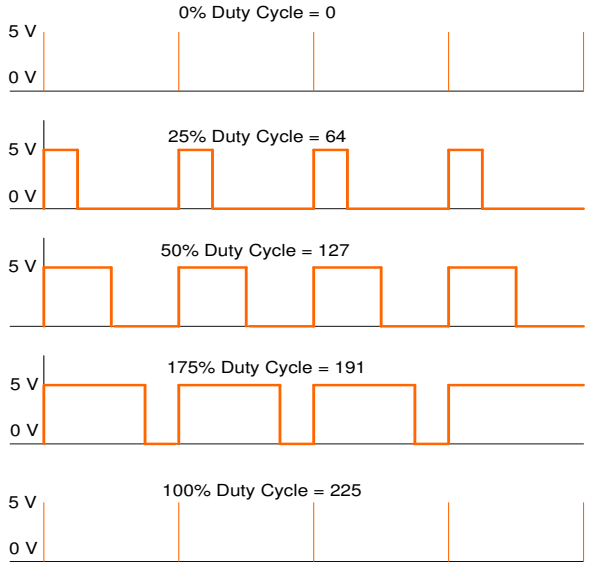
Gambar 6. Sinyal PWM, Vout PWM

Pada metode digital setiap perubahan PWM dipengaruhi oleh resolusi dari PWM itu sendiri. Misalkan PWM digital 8 bit berarti PWM tersebut memiliki resolusi 2^8 = 256, maksudnya nilai keluaran PWM ini memiliki 256 variasi, variasinya mulai dari 0 – 255 yang mewakili *duty cycle* 0 – 100% dari keluaran PWM tersebut seperti Gambar 7.

Dengan cara mengatur lebar pulsa "*on*" dan "*off*" dalam satu perioda gelombang melalui pemberian besar sinyal referensi *output* dari suatu PWM akan didapat *duty cycle* yang diinginkan. *Duty cycle* dari PWM dapat dinyatakan persamaan 1 sebagai berikut.

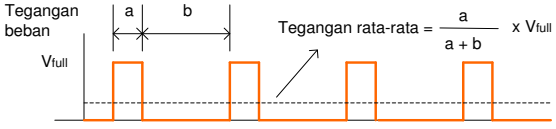
$$dutycycle = \frac{t_{on}}{t_{on}+t_{off}} x100\% \qquad (1)$$

Duty cycle 100% berarti sinyal tegangan pengatur motor dilewatkan seluruhnya. Jika tegangan catu 12V, maka motor akan mendapat tegangan 12V, pada *duty cycle* 50%, tegangan pada motor hanya akan diberikan 50% dari total tegangan yang ada, begitu seterusnya.



Gambar 7. *Duty cycle* dan nilai PWM

Perhitungan pengontrolan tegangan *output motor* dengan metode PWM cukup sederhana. Dengan menghitung *duty cycle* yang diberikan, akan didapat tegangan *output* yang dihasilkan. Sesuai dengan persaman 2 yang telah dijelaskan pada Gambar 8.



Gambar 8. Sinyal PWM pengontrolan motor

$$Averagevoltage = \frac{a}{a+b} xV_{full} \qquad (2)$$

Keterangan: tegangan rata-rata = rata-rata tegangan pada motor, a = lamanya sinyal "*on*", b = lamanya sinyal "*off*", Vfull= tegangan sumber motor.

Tegangan rata-rata merupakan tegangan *output* pada motor yang dikontrol oleh sinyal PWM, a adalah nilai *duty cycle* saat kondisi sinyal "*on*". b adalah nilai *duty cycle* saat kondisi sinyal "*off*". Vfull adalah tegangan maksimum pada motor. Dengan menggunakan rumus diatas, maka akan

didapatkan tegangan *output* sesuai dengan sinyal kontrol PWM yang dibangkitkan.

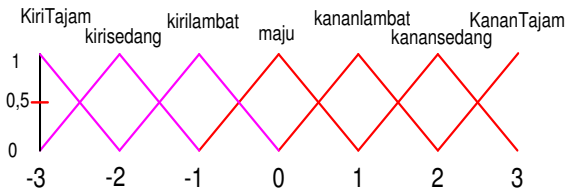
5. Metodologi implementasi robot LF

Tahapan-tahapan pembuatan robot LF dengan menerapkan algoritma *fuzzy* dan PWM yaitu pembuatan perangkat keras yang mengikuti Gambar 2 dan Gambar 3. Selanjutnya tahapan perangkat lunak dimana logika *fuzzy* dan PWM yang dimasukkan dalam program robot LF.

Penentuan aturan dan anggota masukan robot LF dengan *fuzzy* dilakukan dengan simulasi program Matlab. Hasil yang didapat seperti Gambar 9 aturan berdasarkan *fuzzy* dan Gambar 10 masukan anggota *fuzzy*.

1. if (Sensor is KiriTajam) then (PwmKiri is SangatLambat)(PwmKanan is Cepat)(1)
2. If (Sensor is KiriSedang) then (PwmKiri is Lambat)(PwmKanan is cepat)(1)
3. If (Sensor is KiriLambat) then (PwmKiri is sedang)(PwmKanan is cepat)(1)
4. If (Sensor is maju) then (PwmKiri is cepat)(PwmKanan is cepat)(1)
5. If (Sensor is KananLambat) then (PwmKiri is cepat)(PwmKanan is sedang)(1)
6. If (Sensor is KananSedang) then (PwmKiri is cepat)(PwmKanan is Lambat)(1)
7. If (Sensor is KananTajam) then (PwmKiri is cepat)(PwmKanan is SangatLambat(1)

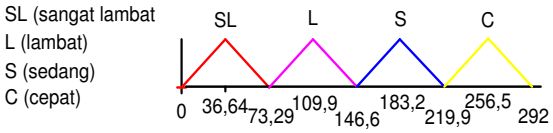
Gambar 9. Aturan berdasakan *fuzzy*



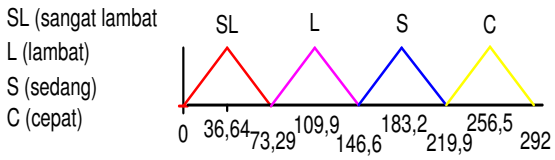
Gambar 10. Masukan anggota *fuzzy*

Keluaran Pwm_kanan, Pwm_kiri dan nilai PWM berturut-turut Gambar 11, Gambar 12 dan Gambar 13.

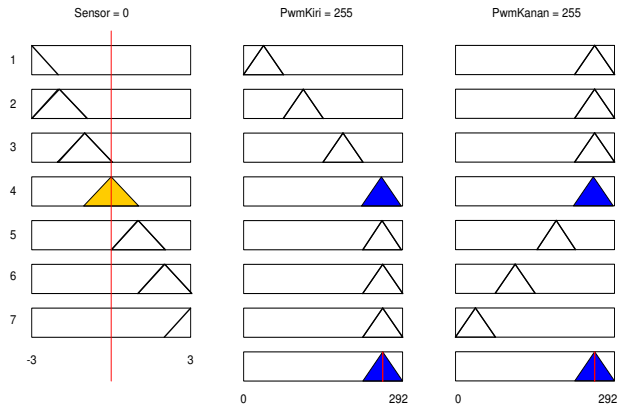
Proses defuzifikasi dengan metode centroid yang didapatkan Gambar 13 untuk menghasilkan nilai PWM.



Gambar 11. Nilai Pwm_kanan



Gambar 12. Nilai Pwm_kiri



Gambar 13. Hasil nilai PWM

Tabel 2 adalah hasil pengujian proses defuzzifikasi, dimana *range* sensor [-3 3] dan *range* Pwm_kiri dan Pwm_kanan [0 292].

Tabel 2. Defuzifikasi nilai Pwm

No.	*Input* variabel sensor	Defuzifikasi nilai Pwm		Keterangan belok
		Pwm_kiri	Pwm_kanan	
1	-3	36,7	255	Kiri tajam
2	-2	110	255	Kiri sedang
3	-1	183	255	Kiri sedikit
4	0	255	255	lurus
5	1	255	183	Kanan sedikit
6	2	255	110	Kanan sedang
7	3	255	36,7	Kanan tajam

Source code yang telah dituliskan sintaknya adalah sebagai berikut.

- Pembacaan masukan dengan aturan *fuzzy*

```
switch(sensor)
{
case 0b11111000: kanan_tajam();break;
                 //belok kanan tajam
case 0b11111100: kanan_tajam();break;
case 0b11111110: kanan_sedang();
                 break; //belok kanan
                 setengah tajam
case 0b11111101: kanan_sedikit();
                 break; // belok
                 kanan lebih sedikit
case 0b11111001: maju();break;
case 0b11110111: kiri_tajam();break;
case 0b11110001: kiri_tajam();break;
case 0b11110011: kiri_sedang();break;
case 0b11111011: kiri_sedikit();
                 break;
case 0b11110000: kiri_tajam();break;
                 //jika tidak ada
                garis, maka belok 180
                derajat
case 0b11111111: P2=0x00;break;
}
```

- Kendali motor kanan dan kiri

```
void RPWM(unsigned char
cent_per,unsigned char reg_duty)
//kendali motor kanan
{
```

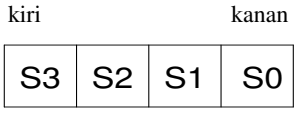
```
      count++;
      if(count<reg_duty)
      {
            en2=1;
      }
      else
            en2=0;
      if(count==cent_per)
      count=0;
}

void LPWM(unsigned char
cent_per,unsigned char reg_duty)
//kendali motor kiri
{
      count++;
      if(count<reg_duty)
      {
            en1=1;
      }
      else
            en1=0;
      if(count==cent_per)
      count=0;
}
```

- Keluaran dengan nilai PWM

```
void maju()
{
kiri1=1;
kiri2=0;
kanan1=0;
kanan2=1;
RPWM(0,255); //nilai PWM motor kanan
LPWM(0,255); //nilai PWM motor kiri
delay(50);
}

void kiri_tajam()
{
kiri1=0;
kiri2=1;
kanan1=0;
kanan2=1;
RPWM(0,255); //nilai PWM motor kanan
LPWM(218,37); //nilai PWM motor kiri
}

void kiri_sedang()
{
kiri1=0;
kiri2=1;
kanan1=0;
kanan2=1;
RPWM(0,255); //nilai PWM motor kanan
LPWM(145,110); //nilai PWM motor kiri
}
void kiri_lambat()
{
kiri1=0;
kiri2=1;
kanan1=0;
kanan2=1;
RPWM(0,255); //nilai PWM motor kanan
LPWM(72,183); //nilai PWM motor kiri
}

void kanan_tajam()
{
kiri1=1;
kiri2=0;
kanan1=1;
kanan2=0;
RPWM(218,37); //nilai PWM motor kanan
LPWM(0,255); //nilai PWM motor kiri
}

void kanan_sedang()
{
kiri1=1;
kiri2=0;
kanan1=1;
kanan2=0;
RPWM(145,110);//nilai PWM motor kanan
LPWM(0,255);  //nilai PWM motor kiri
}

void kanan_lambat()
{
kiri1=1;
kiri2=0;
kanan1=1;
kanan2=0;
RPWM(72,183); //nilai PWM motor kanan
LPWM(0,255); //nilai PWM motor kiri
}
```

6. Hasil dan pembahasan

Pengamatan gerak robot LF yang diuji dengan posisi sensor S0, S1, S2 dan S3 Gambar 14.



Gambar 14. Posisi sensor

Tabel 3 menunjukkan hasil gerak robot yang telah diuji baik di lintasan garis maupun simulasi pada program proteus.

Tabel 3. Hasil pengamatan gerak motor

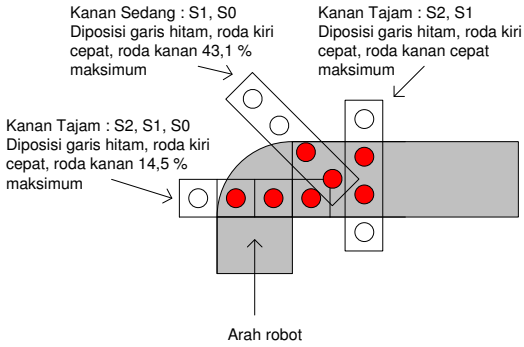
Belokan	Gerak motor		Kondisi masukan			
	kiri	kanan	S3	S2	S1	S0
Kanan tajam	cepat	lambat	1	0	0	0
Kanan sedang	cepat	sedang	1	1	0	0
Kanan sedikit	cepat	Agak cepat	1	1	0	1
lurus	cepat	cepat	1	0	0	1
Kiri sedikit	Agak cepat	cepat	1	0	1	1
Kiri sedang	sedang	cepat	0	0	1	1
Kiri tajam	lambat	cepat	0	1	1	1

Adapun nilai pwm yang didapat dari hasil simulasi *fuzzy* di matlab pada Tabel 4.

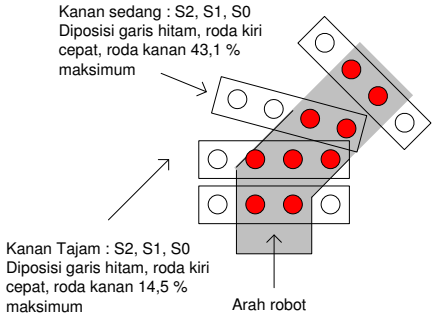
Tabel 4. Nilai PWM simulasi *fuzzy* di matlab

Belokan	Nilai PWM	
	LPWM	RPWM
Kanan tajam	(0, 255)	(218, 37)
Kanan sedang	(0, 255)	(145, 110)
Kanan sedikit	(0, 255)	(72, 183)
lurus	(0, 255)	(0, 255)
Kiri sedikit	(72, 183)	(0, 255)
Kiri sedang	(145, 110)	(0, 255)
Kiri tajam	(218, 37)	(0, 255)

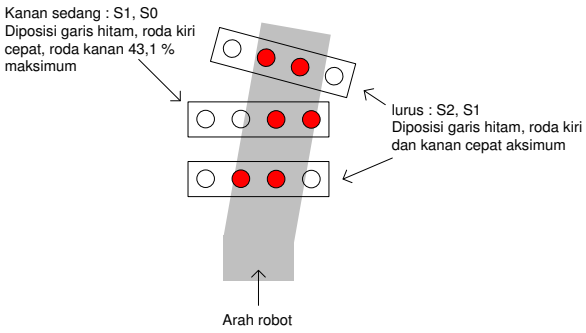
Simulasi lintasan garis belok kanan Gambar 15, Gambar 16 dan Gambar 17 dengan bulatan merah sensor diatas garis.



Gambar 15. Lintasan belok tajam



Gambar 16. Lintasan belok sedang



Gambar 17. Lintasan belok sedikit

Robot LF yang telah diuji seperti tertera pada Tabel 2 hasil yang diamati langsung baik melalui lintasan garis maupun simulasi proteus di komputer menunjukkan perubahan gerak baik motor kiri maupun motor kanan.

Perubahan gerak pada Tabel 2, belokan kanan tajam dibaca sensor S0=0, S1=0, S2=0, S3=1 artinya robot LF belok kanan antara sudut 60^o – 90^o yang menyebabkan motor kiri robot LF bergerak cepat (maksimum) dan motor kanan robot LF bergerak lambat dengan nilai LPWM(0,255) dan RPWM(218,37) seperti Tabel 4. Belokan kanan sedang dengan S0=0, S0=1, S1=1, S1=1 antara sudut 30^o – 60^o menyebabkan motor kiri robot LF bergerak cepat dan motor kanan robot LF bergerak sedang dengan nilai LPWM(0,255) dan RPWM(145,110). Belokan kanan sedikit dengan S0=1, S1=0, S2=1, S3=1 antara sudut 0^o – 30^o menyebabkan motor kiri robot LF bergerak cepat dan motor kanan robot LF bergerak agak cepat, sebaliknya sama dengan belok kiri, gerak motor kiri yang berubah-ubah dan motor kanan kontan cepat serta nilai LPWM yang berubah dan RPWM tetap konstan.

Nilai RPWM dan LPWM Tabel 3 adalah untuk mengatur suplai tegangan motor dimana pengaturan kendali motor berdasarkan lamanya logik "1" yang terhubung pada *enable driver motor* L293D, jika *enable* berlogik "1" maka suplai tegangan tersalurkan pada motor (*on*), jika logik "0" maka suplai tegangan motor tidak tersalurkan (*off*). LPWM(0,255) artinya logik "1" terus menerus (100%) terhubungkan ke *enable motor* kiri dengan prioda takhingga sehingga motor kiri bergerak cepat maksimum karena suplai tegangan ke motor tidak terputus, sedangkan RPWM(218,37) maka logik "1" berbanding nilai (37/255) x 100 % = 14,5%, sisanya berlogik "0" 85,5%, jadi motor kanan selama 14,5 % logik "1" mendapat suplai tegangan sisanya selama 85 % berlogik "0" motor kanan tidak mendapat suplai tegangan dalam satu prioda sehingga rata-rata

tegangan suplai motor sebesar 14,4 % dari tegangan penuh sehingga motor bergerak lambat menyebabkan arah robot LF belok kanan tajam. Nilai LPWM(0,255) dan RPWM(145,110) menghasilkan motor kiri bergerak cepat maksimum tetapi motor kanan bergerak sedang, lamanya logik "1" motor kanan sebesar (110/255) x 100 % = 43,1 % jadi rata-rata tegangan suplai motor kanan 43,1 % dari tegangan penuh sehingga motor bergerak sedang menyebabkan arah robot LF belok kanan sedang. Nilai LPWM(0,255) dan RPWM(72,183) menghasilkan motor kiri gerak cepat maksimum dan motor kanan bergerak agak cepat karena rata-rata tegangan suplai motor sebesar (183/255) x 100 % = 71,8%. Sebaliknya sama jika robot LF belok kiri.

Simulasi uji Gambar 15 dengan sudut belok 90^0 ke kanan, jejak pertama kali ada 3 sensor di posisi atas lintasan (S2, S1, S0), kemudian jejak kedua ada 2 sensor di posisi atas lintasan (S1, S0), selanjutnya lurus. Pengujian Gambar 16 dengan sudut belok 45^0 jejak pertama ada tiga sensor (S2, S1, S0) diatas lintasan kemudian jejak kedua ada dua sensor (S1, S0) di atas lintasan selanjutnya lurus. Pengujian Gambar 17 dengan sudut 10^0 ada jejak dua sensor (S1, S0) di atas lintasan kemudian lurus.

7. Kesimpulan

Beberapa kesimpulan yang diperoleh dalam penelitian ini yaitu sudut belok 90^0 salah satu roda kecepatan maksimum satu roda yang lain mengalami dua kecepatan 14,5% dan 43,1 % dari maksimum. Sudut belok 45^0 satu roda kecepatan maksimum dan satu roda lainnya mengalami dua kecepatan 14,5 % dan 43,1 %. Pengujian sudut belok 10^0 satu roda kecepatan maksimum dan roda satu lainnya 43,1 %.

Ucapan terimakasih

Ucapan terima kasih penulis sampaikan kepada Pusat Penelitian Pengabdian Kepada Masyarakat dan Laboratorium Jurusan Teknik Komputer Politeknik Negeri Sriwijaya dukungan dana dan laboratoirum penelitian melalui Daftar Isian Pelaksanaan Anggaran (DIPA) 2014.

Daftar Pustaka

[1] Pitowarno E., Robotika : *Desain, Kontrol, dan Kecerdasan Buatan*, Andi Offset, Yogyakarta, 2006.

[2] Irvan Lewenusa, Wisnu Ananta Kusuma, "*Autonomous Mobile Robot* Menggunakan Metode Formal Logika Temporal Linier". *Seminar Nasional Informatika 2008 (semnasIF 2008)* ISSN: 1979-2328, UPN "Veteran" Yogyakarta, hal.:145-152, 2008.

[3] Widodo Romy Budhi, *Embedded System menggunakan Mikrokontroler dan Pemrograman C*, Andi Offset, Yogyakarta, 2009.

[4] Shukla, S., Tiwari Mukesh, "Fuzzy Logic of Speed and Steering Control system for Three Dimensional Line Following of An Autonomus Vehicle", *International Journal of Computer Science and Information Security (IJCSIS)*, Vol. 7, No. 3, hal:101-108 , 2010.

[5] Supani, A., 2011, "Pelacakan Jarak Untuk Navigasi Gerak Robot", *Prosiding KNTIA 2011*, ISSN:2088-9658, Fasilkom Universitas Sriwijaya, 21-22 hal. A50-A53, Oktober 2011.

[6] Bachir O., Zoubir, A., "Adaptive Neuro-fuzzy Inference System Based Control of Puma 600 Robot Manipulator", *International Journal of Electrical and Computer Engineering* (IJECE), Vol.2, No.1, ISSN: 2088-8708, pp. 90~97, 2012.

[7] Resmana, et. al., "Implementasi *Fuzzy Logic* Pada *Microcontoller* Untuk Kendali Putaran Motor DC", *Proceedings Industrial Electronic Seminar* 1999 (IES'99), Graha Institut Teknologi Sepuluh Nopember, Surabaya, 1999.

[8] Era Purwanto, M. Ashary, Subagio, Mauridhi Herry P., "Pengembangan *Inverter Fuzzy Logic Control* untuk Pengendalian Motor Induksi Sebagai Penggerak Mobil Listrik dengan Metoda Vector Kontrol", *Jurnal Makara Teknologi*, vol. 12, no. 1, pp. 1-6, April 2008.

[9] Adriansyah A., "Perancangan Pengendali Robot Bergerak Berbasis Perilaku Menggunakan *Particle Swarm Fuzzy Controller"*, *Jurnal Ilmu Komputer dan Informasi,* vol. 3, No. 1, pp. 1-9, 2010.

[10] Widiyanto, D., Supriyo, P.T, Kusuma, W. A., "Formalisasi Navigasi Mobile robot". *Prosiding Seminar Nasional Teknologi dan Rekayasa Industri*. Fakultas Teknologi Industri, Institut Teknologi Indonesia, 2008.

[11] Fadhlan Nuran Ghani, 2012, *Pulse Width Modulation*. http://robotic_electric.blogspot.com diakses tanggal 10 November 2014.