# Laporan FI4171 Proyek Final

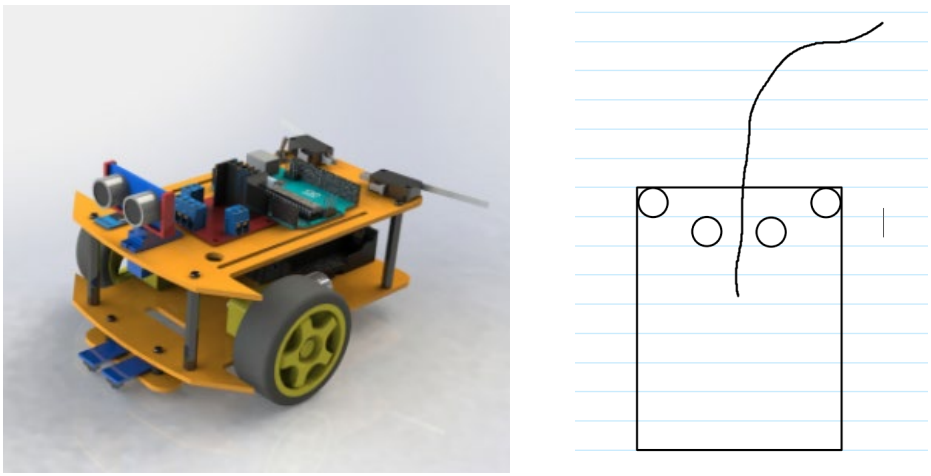
Fadhil Rausyanfikr, Ahmad Fathoni
10217006, 10217037

Proteus adalah sebuah software untuk mendesain PCByang juga dilengkapi dengan simulasi pspice pada level skematik sebelum rangkaian skematik diupgrade ke PCB shingga sebelum PCBnya di cetak kita akan tahu apakah PCB yang akan kita cetak sudah benar atau tidak. Proteus mengkombinasikan program ISIS untuk membuat skematik desain rangkaian dengan program ARES untuk membuat layout PCB dari skematik yang kita buat. Software ini bagus digunakan untuk desain rangkaian mikrokontroller.

Karena keterbatasan waktu dan kesulitan untuk bertemu teman sekelompok pada masa pandemic maka projek ini hanya akan dilakukan simulasi melalui Proteus.

## I. Desain Robot

Didesain suatu robot *line-follower* yang dapat mengikuti garis hitam pada berada pada permukaan lantai putih dengan menggunakan suatu sistem berbasis Arduino Nano. Desain dari robot tersebut dapat dilihat pada gambar berikut



(a) (b)

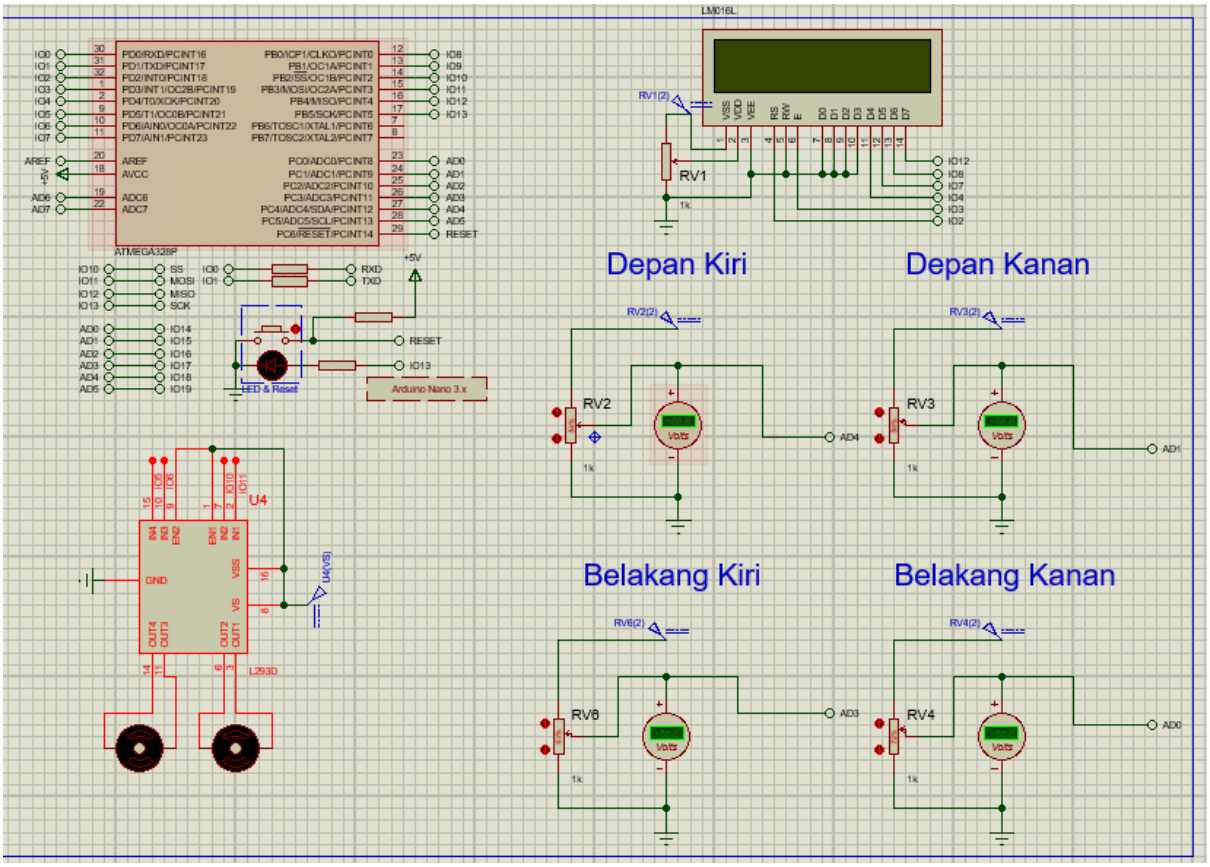
*Gambar 1.1* Model 3d dari Line Follower Robot (a). Layout penempatan sensor ir pada robot (b)

Sensor yang digunakan untuk mendeteksi ada tidaknya garis hitam tersebut adalah sensor warna sebanyak 4 buah, 2 pada sisi kiri dan 2 sisanya pada sisi kanan. Informasi yang diterima oleh keempat sensor tersebut akan diolah dengan menggunakan *fuzzy logic* untuk memperoleh keluaran yang diterima oleh aktuator berupa motor dengan roda yang akan berputar untuk menggerakkan robot mengikuti garis hitam tersebut.

## II. Rangkaian

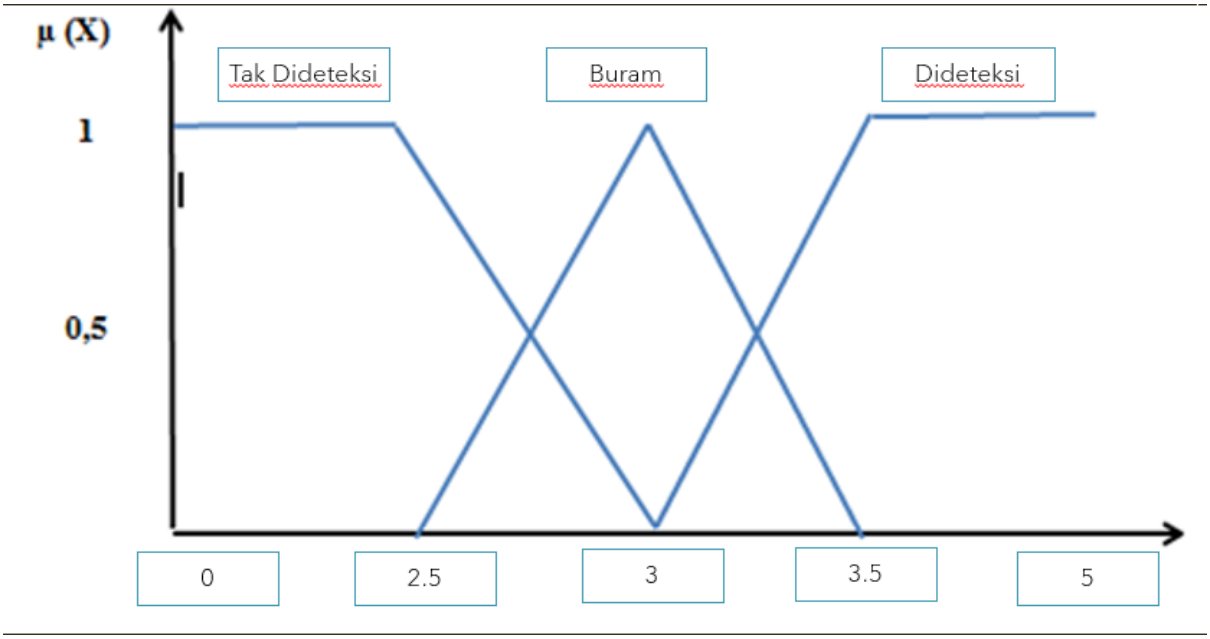
Komponen yang digunakan :

- Arduino Nano v3
- Motor DC (2)
- Driver Motor L293D
- Sensor IR (4)
- LCD 16x2




*Gambar 2.1* Rangkaian skematik pada Proteus

Karena tidak terdapat sensor ir pada proteus maka kami memisalkannya dengan potensiometer sebagai masukan keempat sensor ir. Dengan tegangan berkisar antara 0-5 volt, kami hubungkan dengan pin input ADC dari Arduino. Input dari setiap sensor akan dikategorikan menjadi 3 yaitu terdeteksi, buram, dan tidak terdeteksi berdasarkan tegangan yang masuk sesuai dengan grafik berikut




*Gambar 2.2* Fungsi keanggotaan sensor

## III. Fuzzy Logic

**Fuzzifikasi** merupakan suatu proses pengubahan himpunan non-fuzzy (crisp) kedalam himpunan fuzzy, masukan bukan fuzzy (crisp) dipetakan ke bentuk himpunan fuzzy sesuai dengan variasi semesta pembicaraan masukan. Sementara **Defuzzifikasi** merupakan langkah terakhir dalam suatu sistem logika fuzzy dengan tujuannya mengkonversi setiap hasil dari inference engine yang diekspresikan dalam bentuk fuzzy set kesuatu bilangan real. Hasil konversi tersebut merupakan aksi yang diambil oleh sistem kendali logika fuzzy.

Fuzzy logic yang digunakan pada projek ini yaitu menggunakan metode sugeno. Pada deffuzifikasi dilakukan perhitungan weighted average.

$$WA = \frac{\alpha_{1z_1} + \alpha_2 z_2 + \alpha_3 z_3 + \cdots + \alpha_n z_n}{\alpha_1 + \alpha_2 + \alpha_3 + \cdots + \alpha_n}$$

Keterangan:

WA= Nilai rata-rata, $\alpha n$ = nilai predikat aturan ke-n, dan $zn$ = indeks nilai output (konstanta) ke-n.

Proses fuzzyfikasi pertama2 dilakukan dengan mengubah nilai tegangan masuk pada setiap sensor menjadi *weight* dengan range 0-1 sesuai dengan grafik pada Gambar2.2. Proses tersebut dapat dilihat pada kode berikut.

```
void FuzzyDepanKanan(){
  // untuk tidak terdeteksi
  if (rightlval <= 2.5)
  { depankanan [0] = 1;}
  else if (rightlval > 2.5 && rightlval <= 3)
  {  depankanan [0] = (3 - rightlval)/(3 - 2.5); }
  else
  { depankanan [0] = 0;}

  // untuk buram
  if (rightlval <= 2.5)
  { depankanan [1] = 0;}
  else if (rightlval > 2.5 && rightlval <= 3)
  { depankanan [1] = (rightlval-3)/(3-2.5);}
  else if (rightlval > 3 && rightlval <= 3.5)
  { depankanan [1] = (3.5-rightlval)/(3.5 - 3);}
  else
 { depankanan [1] = 0;}

  // untuk terdeteksi
  if (rightlval <= 3)
  { depankanan [2] = 0;}
  else if (rightlval > 3 && rightlval <= 3.5)
  { depankanan [2] = (rightlval-3)/(3.5-3);}
  else
  { depankanan [2] = 1;}
}
```

*Gambar 3.1* Kode proses Fuzzyfikasi untuk input dari sensor depan kanan

Adapun rule yang dirancang untuk salah satu sisi sebagai berikut :

| Belakang → /Depan ↓ | Tidak terdeteksi | Buram | Terdeteksi |
|---|---|---|---|
| Tidak terdeteksi | Cepat | Lambat | Lambat |
| Buram | Cepat | Sedang | Lambat |
| Terdeteksi | Cepat | Cepat | Lambat |

Table diatas merupakan rule base yang akan di program pada Arduino. Dari parameter tersebut, terdapat 3 himpunan fungsi keanggotaan output motor yaitu lambat,sedang dan cepat. Proses evaluasi rule base pada hasil fuzzyfikasi dapat dilihat pada kode berikut.

```
void RuleKiri (){
 int i, j;
 float temp;
 for ( i=0; i<=2; i=i+1)
 {
   for ( j=0; j<=2; j=j+1)
   {
     temp = min(depankiri[i], belakangkiri[j]);
     rulekiri [i][j] = temp;
   }
 }
 rulekiri00 = rulekiri [0][0]; // (Cepat)
 rulekiri01 = rulekiri [0][1]; // (Cepat)
 rulekiri02 = rulekiri [0][2]; // (Cepat)

 rulekiril0 = rulekiri [1][0]; // (Lambat)
 rulekiril1 = rulekiri [1][1]; // (Sedang)
 rulekiril2 = rulekiri [1][2]; // (Cepat)

 rulekiri20 = rulekiri [2][0]; // (Lambat)
 rulekiri21 = rulekiri [2][1]; // (Lambat)
 rulekiri22 = rulekiri [2][2]; // (Lambat)
}
```

*Gambar 3.2* Kode proses evaluasi Rule base sisi kiri berdasarkan hasil fuzzyfikasi input sensor depan kiri dan belakang kiri.

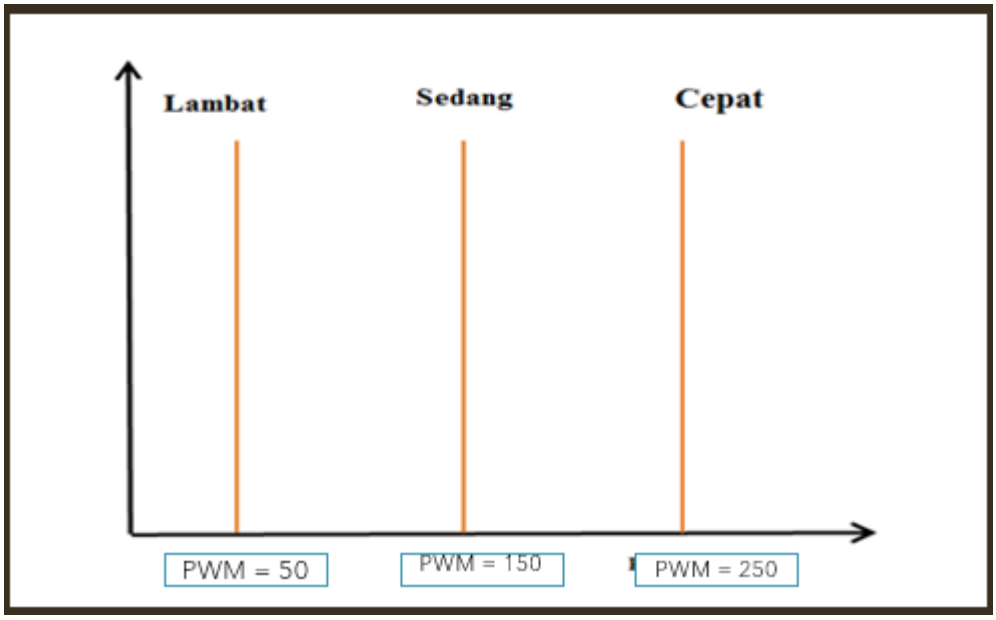
Pada defuzzifikasi yaitu tahap terakhir pada *Fuzzy Logic Controller* (FLC). Penyelesaian defuzzifikasi ini menggunakan metode Sugeno tipe WA (*Weighted Average*). Proses dapat dilihat pada kode berikut.

```
float DefuzzyKiri () {
  // metode sugeno (weighted average)
  float lambat = 50;
  float sedang = 150;
  float cepat = 250;
  float defuz,pwm;
  RuleKiri();
  pwm = (rulekiri00 * cepat) + (rulekiri01 * cepat)+ (rulekiri02 *cepat)+ (rulekiril0 * lambat)+ (rulekiril1 * sedang)+ (rulekiril2 * cepat) + (rulekiri20 * lambat)+

  defuz = 0;
  int i, j;
  for ( i=0; i<=2; i=i+1)
  {
    for ( j=0; j<=2; j=j+1)
    {
      defuz = defuz + rulekiri [i][j];
    }
  }
  if (defuz != 0){
  pwm = pwm / defuz;
  } else{
    pwm = sedang;
  }
  return pwm;
}
```

*Gambar 3.3* Kode proses defuzzyfikasi sisi kiri.

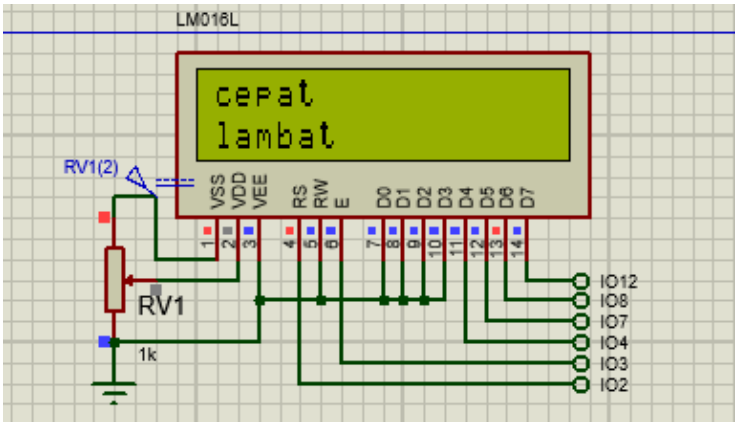
Hasil dari proses defuzzyfikasi ini menghasilkan nilai yang akan dikeluarkan sebagai output untuk menggerakan motor dc sesuai dengan keanggotaan.

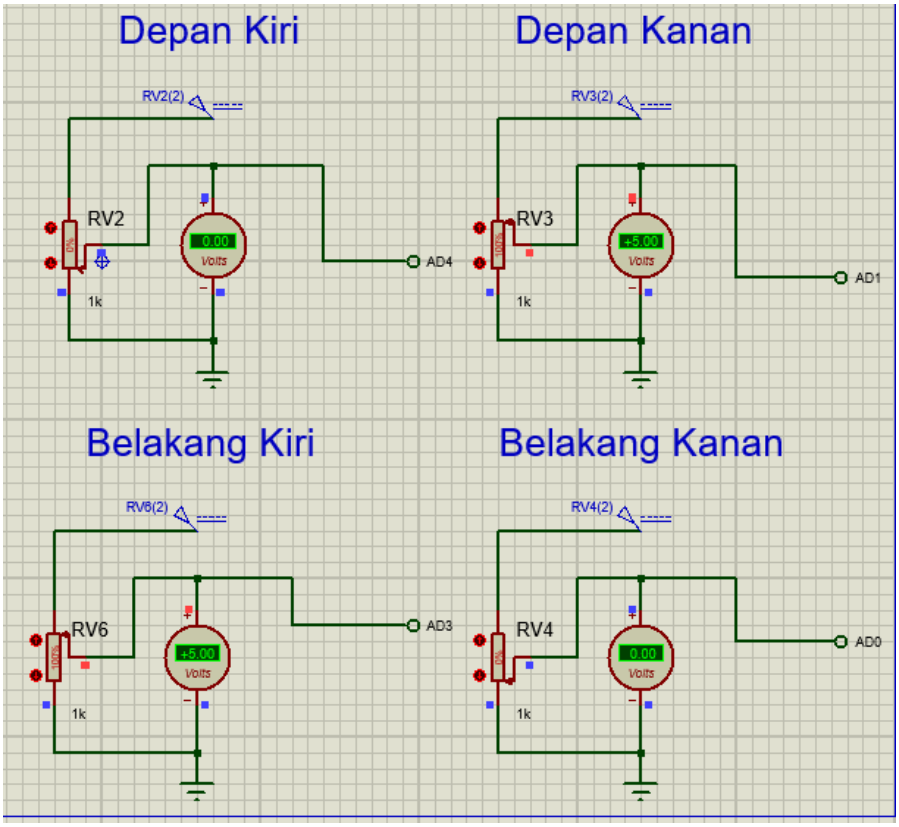



*Gambar 3.4* Fungsi keanggotaan kecepatan motor.

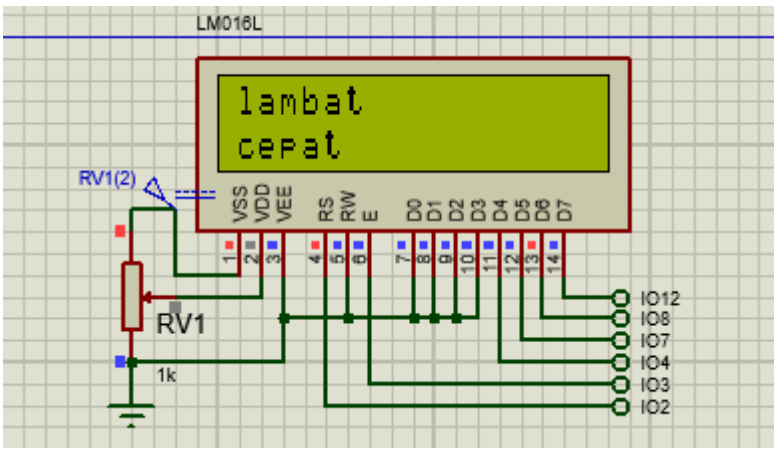
## IV. Hasil Simulasi

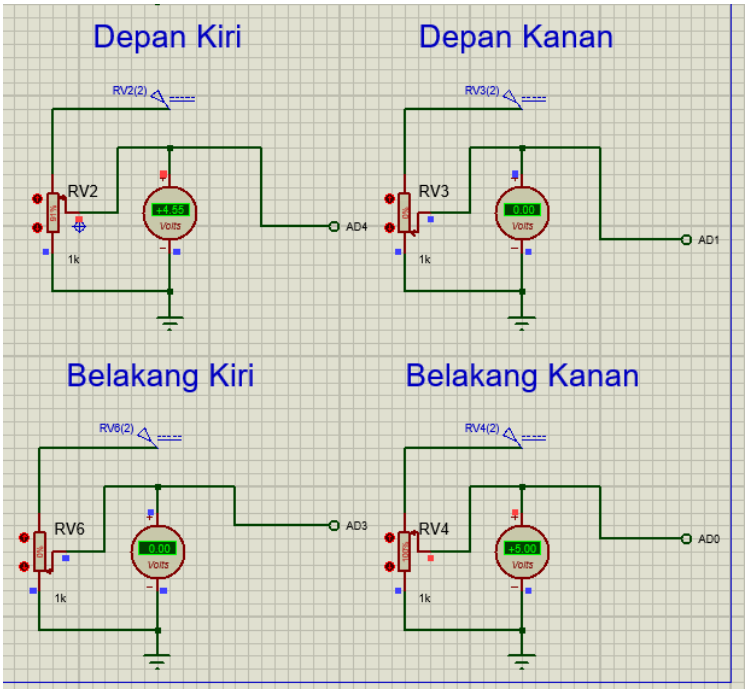
Contoh kasus apabila robot menemui belokan ke kanan. Maka sensor kanan depan akan mendeteksi terlebih dahulu sementara sensor kanan belakang belum mendeteksi. Kombinasi tersebut (sesuai dengan table rule base) akan mengeluarkan output pada sisi kanan dengan pwm = 50 (lambat). Sementara motor sisi kiri masih dalam keadaan cepat (pwm = 250) sehingga badan robot akan membelok ke arah kanan dan menyebabkan sensor kiri belakang mendeteksi line sementara sensor kiri depan tidak. Hasil simulasi dapat dilihat pada gambar berikut.





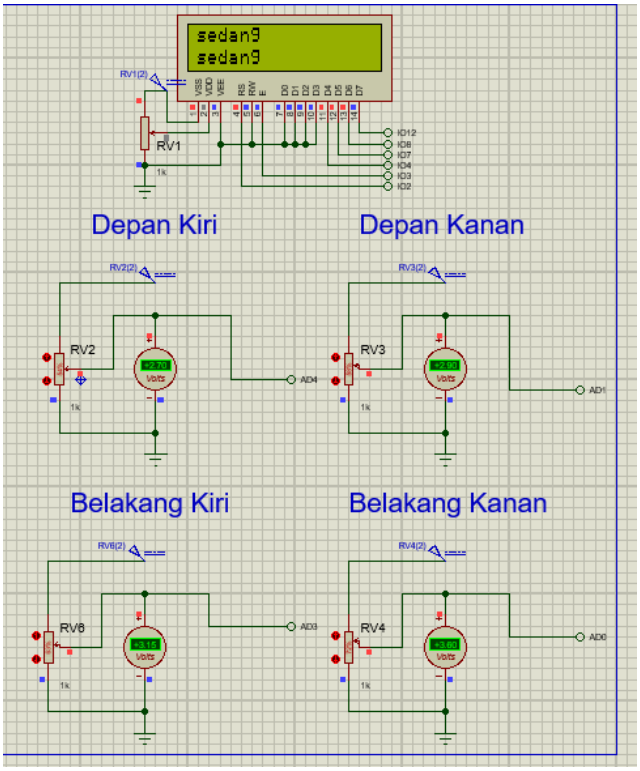

Contoh kasus apabila robot menemui belokan ke **kiri.** Maka sensor kiri depan akan mendeteksi terlebih dahulu sementara sensor kiri belakang belum mendeteksi. Kombinasi tersebut (sesuai dengan table rule base) akan mengeluarkan output pada sisi kiri dengan pwm = 50 (lambat). Sementara motor sisi kanan masih dalam keadaan cepat (pwm = 250) sehingga badan robot akan membelok ke arah kiri dan menyebabkan sensor kanan belakang mendeteksi line sementara sensor kanan depan tidak. Hasil simulasi dapat dilihat pada gambar berikut.






Contoh kasus apabila semua sensor membaca buram sehingga motor pada kedua sisi memiliki kecepatan sedang. Kasus ini kemungkinan terjadi sangat kecil pada kejadian nyata dan hanya mungkin apabila menemui line yang bercabang sehingga membentuk segitiga. Kejadian yang mungkin terjadi yaitu salah satu sisi memiliki kecepatan sedang sementara sisi lain lambat/cepat. Salah satu kejadian yang dapat menyebabkan kombinasi tersebut yaitu tepat saat salah satu sensor depan mendeteksi garis yang belok sehingga masih terbaca buram.

Untuk kepentingan simulasi maka ditunjukan kejadian saat semua sensor membaca buram



## V. Kesimpulan

- Simulasi sudah dapat merekayasa system robot line follower dengan cukup baik terutama saat akan berbelok kanan dan kiri.
- Sistem fuzzy sudah dapat mengkategorikan output berdasarkan dua parameter input keanggotaan yang berbeda.
- Untuk kedepannya dapat diperbaik dengan cara menambah keanggotaan pada rule base output. Misalnya menambahkan lambat sedang saat sensor depan buram dan sensor belakang tidak terdeteksi. Selain itu dapat juga ditambahkan cepat sedang saat sensor depan tidak terdeteksi sementara sensor belakang buram.

## VI. Referensi

**[1]** Jurnal Ilmu Matematika dan Terapan , Desember 2015 , Volume 9 Nomor 2 , Hal. 121 – 134

**[2]**

https://sonoku.com/implementasi-fuzzy-logic-controller-untuk-kontrol-kecepatan-motor-dc-pada-prototype-kipas-angin/