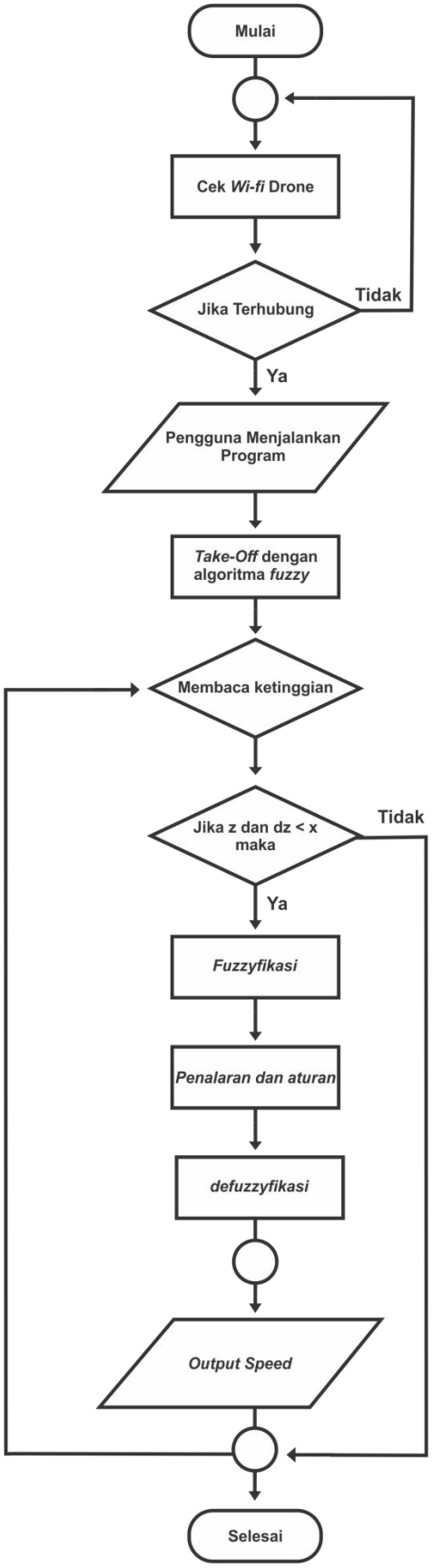
BAB 5 PERANCANGAN DAN IMPLEMENTASI

Pada bab ini akan dijelaskan mengenai proses perancangan pada sistem kendali *take-off quadcoter* menggunakan logika *fuzzy.* Terdapat tiga tahapan seperti ditunjukan Gambar 5.1 tahapan yang pertama adalah tahapan perancangan dan implementasi sensor ultrasonik dan sensor IMU kemudian yang kedua adalah perancangan dan implementasi komunikasi sistem sedangkan yang ketiga adalah perancangan dan implementasi logika *fuzzy*.

Gambar 5.1 Tahapan perancangan sistem

Alur kerja pada sistem ini akan ditunjukkan oleh Gambar 5.2. Pertama akan dilakukan pengecekan koneksi pada *laptop* apakah telah terhubung dengan koneksi *wi-fi* yang di pancarkan oleh *Ar-Drone*, jika koneksi telah terhubung maka *user* dapat menjalankan program pada *laptop* dengan sistem operasi *linux* Ubuntu, jika *laptop* tidak terhubung dengan koneksi *wi-fi* yang di pancarkan oleh *Ar-Drone*, maka *user* harus mengecek kembali koneksi tersebut. Setelah *user* dapat terhubung pada *Ar-Drone*, dan program telah dijalankan maka *Ar-Drone*, akan *take-off* dan sensor ultrasonik akan mulai membaca ketinggian yang telah ditentukan oleh *user*. Jika ketinggian yang dibaca oleh sensor belum sesuai dengan jarak yang ditentukan, maka algoritma *fuzzy* akan terus berjalan untuk menghasilkan *output* berupa pengaturan kecepatan motor hingga *Ar-Drone* berada pada posisi *hover* dengan ketinggian yang telah ditentukan. Proses ini akan terus berulang dikarenakan nilai yang di keluarkan oleh *Ar-Drone* akan terus dibaca dan dijadikan sebagai *input* dari logika *fuzzy.*

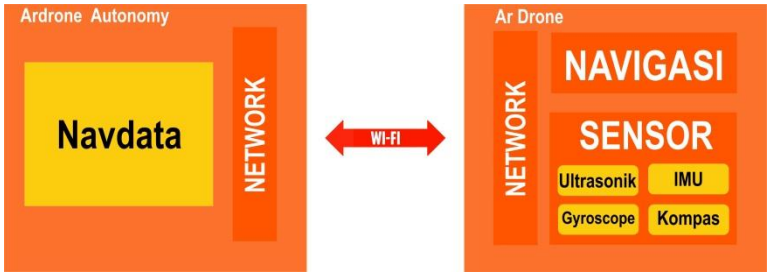


Gambar 5.2 *Flowchart* sistem kendali *take-off quadcopter*

5.1 Perancangan dan Implementasi Sensor Ultrasonik dan IMU

5.1.1 Perancangan Sensor Ultrasonik dan IMU

Pada Gambar 5.3 akan dijelaskan mengenai bagaimana cara perancangan pengambilan data sensor dari *AR-Drone* aktual menggunakan bantuan dari driver *ardrone autonomy*.



Gambar 5.3 Alur pengambilan data sensor

Pada *AR-Drone* terdapat data yang dapat diakses seperti data navigasi dan data sensor seperti yang ditunjukan pada gambar di atas. Pada penelitian ini data sensor yang dibutuhkan adalah data sensor ultrasonic serta sensor IMU untuk dapat mengambil data sensor kita dapat menggunakan bantuan dari *driver ardrone autonomy* sebagai jembatan penghubung antara *quadcopter* aktual dengan *laptop* menggunakan komunikasi *wi-fi* yang ada pada *quadcopter.* Pada *driver ardrone autonomy* terdapat folder navdata. Folder tersebut berisikan data-data yang ada pada *quadcopter.* Data tersebut dapat kita akses untuk dijadikan *input* logika *fuzzy* yang akan kita buat. Untuk cara pengaksesan data pada folder navdata akan dijelaskan pada Subbab 5.1.2 yaitu tahapan implementasi sensor ultrasonik dan sensor IMU.

5.1.2 Implementasi Sensor Ultrasonik dan IMU

Pada penelitian ini dibutuhkan data dari sensor ultrasonik dan sensor IMU pada *AR-Drone.* Data sensor yang telah didapat nantinya akan digunakan sebagai *input* dari *rule* yang akan dibuat pada *fuzzy inference system*. Berikut adalah potongan program untuk pengambilan data sensor ultrasonik dan IMU.

Kode Program 5.1 Pengambil data sensor

```
1  #!/usr/bin/env python
2  import rospy
3  import time
4  import math
5
6  from geometry_msgs.msg import Twist
7  from std_msgs.msg import Empty
8  from ardrone_autonomy.msg import Navdata
```

Pada potongan program di atas dapat dilihat pada baris 8 terdapat perintah `from ardrone_autonomy.msg import Navdata`. Maksud dari baris program tersebut adalah kita dapat menggambil data sensor dengan menggunakan bantuan dari *driver ardrone autonomy* yang lebih jelas nya akan dijelaskan pada subbab perancangan dan implementasi komunikasi sistem. Data yang kita

butuhkan terdapat pada navdata dalam *ardrone autonomy.* Sedangkan pada navdata terdapat data-data yang kita butuhkan seperti data sensor ultrasinik data sensor IMU serta data navigasi dari *quadcopter*. Pada kode program dibawah ini adalah merupakan potongan program untuk mengambil data yang kita perlukan dari navdata yang berada dalam driver *ardrone autonomy*.

```
1   class AutonomousFlight():
2
3      def __init__(self):
4         self.height =0
5         rospy.init_node('forward', anonymous=False)
6         self.rate = rospy.Rate(10)
7         self.pubTakeoff = rospy.Publisher("ardrone/takeoff",Empty,
8   queue_size=10)
9         self.pubLand = rospy.Publisher("ardrone/land",Empty, queue_size=10)
10        self.pubCommand = rospy.Publisher('cmd_vel',Twist, queue_size=10)
11        self.command = Twist()
12        self.subNavdata =
13  rospy.Subscriber('/ardrone/navdata',Navdata,self.ReceiveNavdata)
14        self.state_change_time = rospy.Time.now()
15        rospy.on_shutdown(self.SendLand)
16
17     def ReceiveNavdata(self,navdata):
18        self.height          = navdata.altd
19        self.kecepatanZ      = navdata.vz
20        self.percepatanZ     = navdata.az
21        self.kecepatanmotor1 = navdata.motor1
22        self.kecepatanmotor2 = navdata.motor2
23        self.kecepatanmotor3 = navdata.motor3
24        self.kecepatanmotor4 = navdata.motor4
25
26     def SendTakeOff(self):
27        self.pubTakeoff.publish(Empty())
28        self.rate.sleep()
29
30     def SendLand(self):
31        self.pubLand.publish(Empty())
32
33     def SetCommand(self, linear_x, linear_y, linear_z, angular_x,
34  angular_y, angular_z):
35        self.command.linear.x  = linear_x
36        self.command.linear.y  = linear_y
37        self.command.linear.z  = linear_z
38        self.command.angular.x = angular_x
39        self.command.angular.y = angular_y
40        self.command.angular.z = angular_z
41        self.pubCommand.publish(self.command)
42        self.rate.sleep()
```

Pada baris 1-42 dibuat sebuah *class* dengan nama *AutonomousFlight,* pada *class* tersebut berisi potongan program pengambilan data sensor ketinggian yang ada pada *quadcopter*. Selain data ketinggian pada baris 1-42 terdapat potongan kode program untuk data navigasi *quadcopter* seperti *take-off, landing*, data percepatan sudut x, y, z dan data kecepatan linier x, y, z pada *quadcopter.* Dibawah ini merupakan penjelasan perbaris dari kode program diatas.

a. Baris 3-15 : Digunakan untuk proses inisialisasi printak pada *Ar.Drone* dan juga inisialisasi untuk parameter navdata.

b. Baris 17-24 : Digunakan untuk mengambil atribut ketinggian dari sensor ultrasonik dan data percepatan linear serta kecepatan sudut dari sensor IMU.

c. Baris 26-28 : Digunakan untuk menjalankan perintah *take-off.*

d. Baris 30-31 : Digunakan untuk menjalankan perintah *landing.*

e. Baris 33-42 : Digunakan untuk penggantian nilai parameter kecepatan linear.

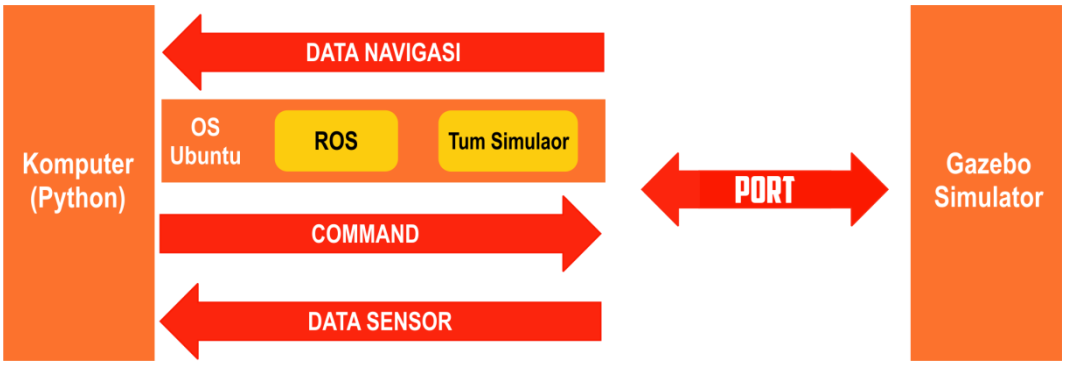
5.2 Perancangan dan Implementasi Komunikasi Sistem

5.2.1 Perancangan Komunikasi Sistem

Pada sistem yang di buat ini terdapat dua proses pertukaran data pada sistem, seperti ditunjukan Gambar 5.4 dan 5.5 dikarenakan sebelum program dijalankan pada *quadcopter* aktual program terlebih dahulu di uji pada *gazebo* simulator dengan menggunakan *driver* tum simulator sedangkan pada *Ar-Drone* aktual *driver* yang digunakan adalah *ardrone autonomy*. Keseluruhan data yang diproses pada *Ar-Drone* aktual dijelaskan sesuai dengan alur yang terdapat Gambar 5.4. Pada *quadcopter* memiliki data navigasi dan juga data sensor seperti yang terlihat pada Gambar 5.4. Data yang dibutuhkan pada penelitian ini adalah data navigasi serta data sensor. Sedangkan data sensor yang dibutuhkan adalah data dari sensor ultrasonik berupa data jarak atau ketinggian,dan juga data dari sensor IMU yang berupa data percepatan serta perubahan percepatan. Data yang telah didapatkan dari *quadcopter* dikirim melalui protokol jaringan UDP dan diproses oleh *Ardrone Autonomy* yang berfungsi sebagai *driver* pada komputer dengan OS Ubuntu. Data navigasi dan juga data sensor yang telah didapat dari *quadcopter* dengan bantuan *driver ardrone autonomy* selanjutnya akan dibuat menjadi suatu program pengolah data dengan bahasa pemograman *python*. Selanjutnya data tersebut diproses pada program yang telah dibuat hingga menghasilkan *output* berupa intruksi agar *quadcoper* dapat mempertahankan posisi *hover* pada ketinggian tertentu. Lalu intruksi yang telah dibuat tersebut digunakan sebagai parameter dalam pengiriman data pada *quadcopter.* Kemudian data yang dikirim dan diterima pada *quadcopter* harus berupa perintah *command* yang dukirim melalui protokol jaringan UDP.

Gambar 5.4 Alur pertukaran data pada sistem

Pada dasarnya *driver* untuk tum simulator dan *ardrone autonomy* adalah sama, sehingga proses pertukaran data serta program yang digunakan pada simulator dan *Ar-Drone* aktual juga sama, perbedaan komunikasi antara tum simulator dan *Ar-Drone* aktual adalah, tum simulator dijalankan di atas *gazebo* dan berkomunikasi dengan *Robot Operating System* (ROS) menggunakan *port,* sedangkan untuk menghubungkan *Ar-Drone* aktual dengan ROS menggunakan koneksi *wi-fi.* Gambar 5.5 merupakan gambaran alur pertukaran data pada simulator.



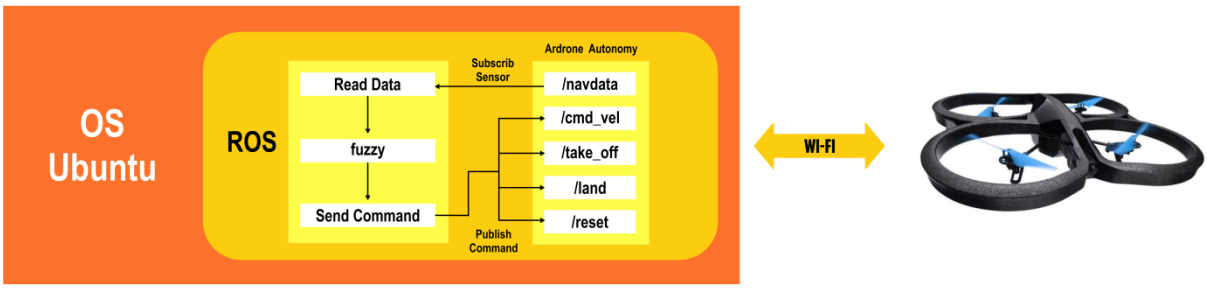
Gambar 5.5 Alur pertukaran data pada simulator

Pada penelitian ini perancangan komunikasi data pada *quadcopter* hanya fokus pada komunikasi antara *Robot Operating System* (ROS) dengan Ubuntu dan komunikasi antara *Robot Operating System* (ROS) dengan *quadcopter*.

5.2.2 Implementasi Komunikasi Sistem

Pada implementasi komunikasi sistem akan dijelaskan bagaimana alur implementasi komunikasi sistem pada penelitian ini. Gambar 5.6 marupakan gambaran dari komunikasi pada sistem dimana *Robot Operating System* (ROS) dijalankan di atas Ubuntu menggunakan terminal dengan sintak *$ roscore.* Data sensor dan data navigasi dari *quadcopter* diproses pada *Robot Operating System* (ROS) dengan *ardrone autonomy* sebagai *driver* perantara nya. Didalam *Robot Operating System* (ROS) data sensor dan data navigasi di ambil dari *ardrone_autonomy.msg import Navdata.* Data sensor yang diperoleh dari navdata diproses pada `sensor_msgs.msg import Range` sedangkan data navigasi atau data gerakan diproses pada `geometry_msgs.msg import Twist`. Selanjutnya data yang diperoleh dari /navdata akan terus dibaca oleh sistem untuk di proses pada logika *fuzzy* dan menghasilkan *command* yang akan di publikasikan pada /cmd_vel, /take_off, /land, /reset, yang nantinya akan

menghasilkan intruksi pada *quadcopter* untuk mempertahankan posisi *hover* pada ketinggian tertentu.



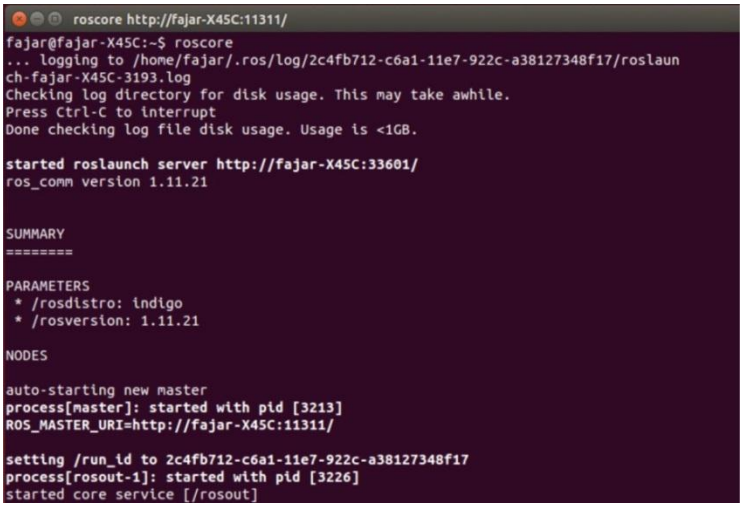
Gambar 5.6 Alur implementasi komunikasi sistem

5.2.2.1 Komunikasi antara *Robot Operating System* (ROS) dengan Ubuntu

Pada komunikasi antara *Robot Operating System* (ROS) dengan ubuntu langkah pertama yang harus kita lakukan adalah kita harus menginstal *Robot Operating System* (ROS) terlebih dahulu pada sistem operasi ubuntu 14.04 dikarenakan *Robot Operating System* (ROS) dijalankan diatas ubuntu menggunakan terminal dengan sintak seperti dibawah ini

```
$ roscore
```

Jika ROS telah terinstal dan telah terpasang dengan baik pada ubuntu maka akan muncul tampilan seperti yang ditunjukkan Gambar 5.7



Gambar 5.7 Tampilan ROS saat dijalankan

5.2.2.2 Komunikasi antara ROS dengan Quadcopter

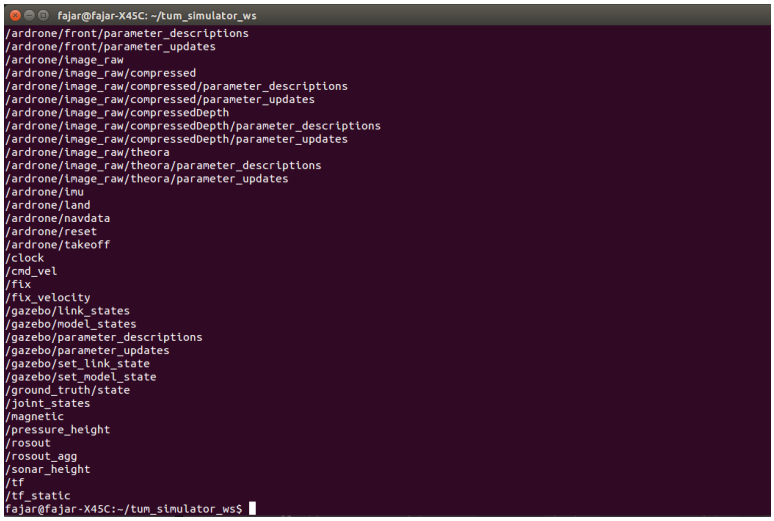
ROS berkomunikasi dengan *quadcopter* menggunakan *driver ardrone autonomy* dengan sintak seperti dibawah ini.

```
-roslaunch ardrone_autonomy ardrone.launch
```

Jika ROS dan *quadcopter* aktual telah terhubung maka data-data yang ada pada *quadcopter* dapat di akses dengan menggunakan sintak seperti dibawah ini.

```
$ rostopic list
```

Akan muncul data-data apa saja yang ada pada *quadcopter* tersebut seperti ditunjukan Gambar 5.8

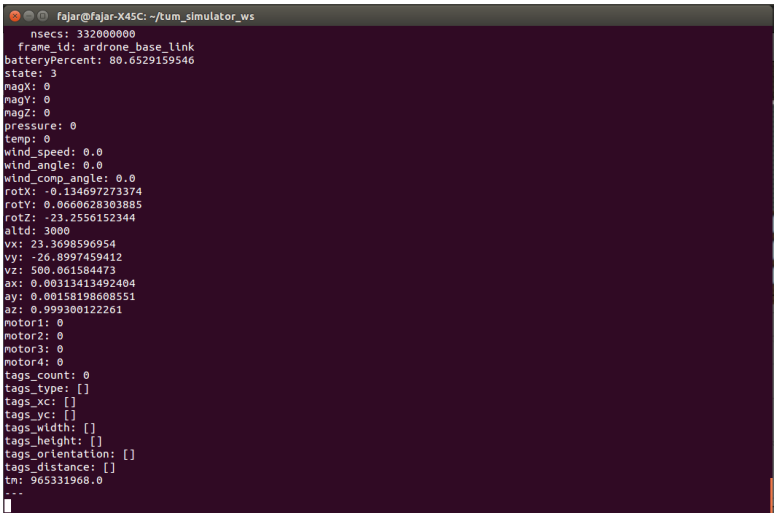


Gambar 5.8 Tampilan data pada *quadcopter*

Salah satu contoh data yang akan kita ambil adalah */ardrone/navdata* yang berfungsi untuk membaca data sensor pada *quadcopte*r dan sintak untuk menampilkan data tersebut adalah

```
$  rostopic echo /ardrone/navdata
```

Gambar 5.9 merupakan tampilan dari hasil pengambilan data yang telah dilakukan pada *quadcopter* dengan menggunakan driver *ardrone autonomy.*



Gambar 5.9 Tampilan data navigasi dari navdata

5.3 Perancangan dan implementasi logika *fuzzy*

Perancangan dan implementasi logika *fuzzy* terdiri dari perancangan pada *fuzzy logic toolbox* dan implementasi pada program. Pada perancangan *fuzzy logic toolbox* akan dibuat dua buah *input* yaitu percepatan serta perubahan percepatan dengan *output* berupa kecepatan menggunakan *rule base* 7x7, yang nantinya akan menghasilkan 49 aturan. Seperti yang dilakukan peneliti sebelumnya oleh Raharja et al. (2015) . Untuk penjelasan lebih lengkap akan dijelaskan pada subbab perancangan pada *fuzzy logic toolbox.*

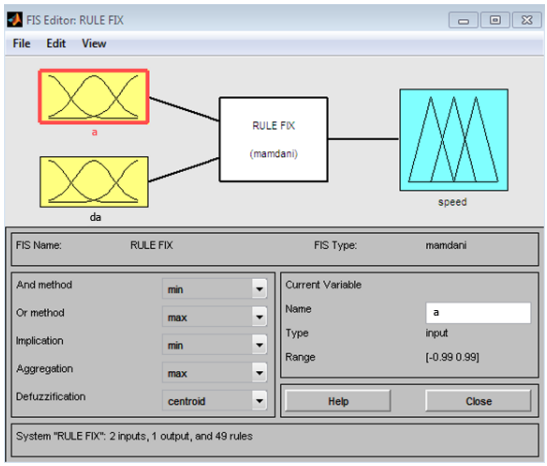
5.3.1 Perancangan Pada *Fuzzy Logic Toolbox*

Dalam penelitian ini digunakan *fuzzy logic toolbox* yang terdapat pada *Matlab*. *Fuzzy logic toolbox* merupakan fasilitas yang disediakan pada *Matlab* untuk membangun suatu sistem *fuzzy* . *Fuzzy logic toolbox* memberikan fasilitas *Graphical User Interface* (GUI) untuk mempermudah dalam membangun suatu sistem *fuzzy*. Terdapat 5 tahapan yang harus dilakukan untuk membangun suatu sistem *fuzzy* yaitu diantaranya :

- *Fuzzy Inference System (FIS) Editor*
- *Membership Function Editor*
- *Rule Editor*
- *Rule Viewer*
- *Surface Viewer*

1. *Fuzzy Inference System* (FIS) Editor

   Untuk memulai membuat suatu sistem *fuzzy* kita cukup menuliskan `fuzzy` pada *command line* yang terdapat di *Matlab*. Kemudian akan muncul GUI untuk FIS Editor seperti yang ditunjukkan Gambar 5.10. Dengan menggunakan FIS Editor kita dapat memulai melakukan editing terhadap sistem *fuzzy* yang ingin kita bangun.



Gambar 5.10 Tampilan dari *Fuzzy Inference System*

Gambar 5.10 terdapat dua *input* dan satu *output* yang digunakan, dimana input yang pertama adalah nilai dari percepatan terhadap sumbu Z (α) dan *input* yang kedua adalah nilai dari perubahan percepatan terhadap sumbu Z (Δα) sedangkan *output* nya adalah kecepatan *(speed).*

2. *Membership Function Editor*

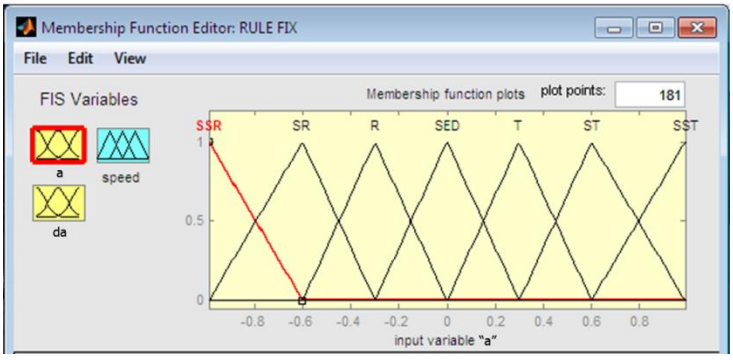
   Membership function editor digunakan untuk merancang fungsi keanggotaan himpunan *fuzzy* untuk setiap variabel *input* dan *output.* Pada *library* dari ardrone _autonomy rentang untuk nilai dari kecepatan sudut dan kecepatan linear adalah antara -1,0 hingga 1,0, sehingga *range* yang digunakan pada penelitian ini berada diantara -1,0 hingga 1,0.

   Dalam penelitian ini terdapat 7 fungsi keanggotaan untuk setiap *input* dan *output* yang digunakan, berikut merupakan gambaran dari fungsi keanggotaan pada penelitian ini.

   Tabel 5.1 adalah Tabel dari fungsi keanggotaan untuk setiap *input* dan *output* yang digunakan pada penelitian ini.

Tabel 5.1 Fungsi keanggotaan *input* dan *output*

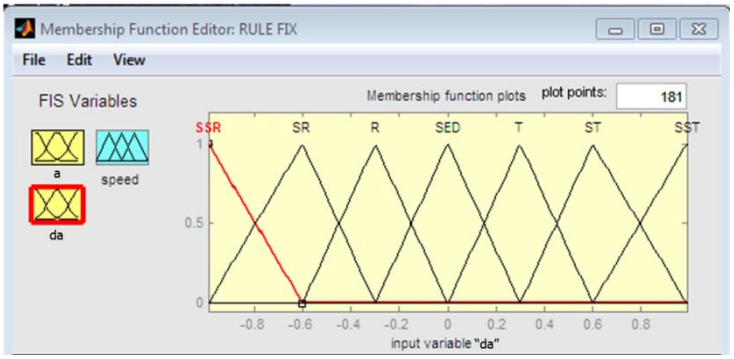
Input Percepatan (α)	***Input Perubahan Percepatan (Δα)***	***Output Kecepatan (Speed)***
Sangat-Sangat Rendah (SSR)	Sangat-Sangat Rendah (SSR)	Sangat-Sangat Lambat (SSL)
Sangat Rendah (SR)	Sangat Rendah (SR)	Sangat Lambat (SL)
Rendah (R)	Rendah (R)	Lambat (L)
Sedang (SED)	Sedang (SED)	Sedang (SED)
Tinggi (T)	Tinggi (T)	Cepat (C)
Sangat Tinggi (ST)	Sangat Tinggi (ST)	Sangat Cepat (SC)
Sangat-Sangat Tinggi (SST)	Sangat-Sangat Tinggi (SST)	Sangat-Sangat Cepat (SSC)



Gambar 5.11 *Membership Function Editor input (α)*

Membership function editor untuk *input* yang pertama yaitu percepatan terhadap sumbu Z ditunjukkan oleh Gambar 5.11. Berikut ini penjelasan mengenai fungsi keanggotaan 7x7 *input* percepatan *(α)* yang dirancang :

- Fungsi keanggotaan Sangat-Sangat Rendah (SSR) : kurva segitiga dengan *range* [-1.0, -0.99, -0.6]
- Fungsi keanggotaan Sangat Rendah (SR) : kurva segitiga dengan *range* [-0.99, -0.6, -0.3]
- Fungsi keanggotaan Rendah (R) : kurva segitiga dengan *range* [-0.6, -0.3, 0]
- Fungsi keanggotaan Sedang (SED) : kurva segitiga dengan *range* [-0.3, 0, 0.3]
- Fungsi keanggotaan Tinggi (T) : kurva segitiga dengan *range* [0, 0.3, 0.6]
- Fungsi keanggotaan Sangat Tinggi (ST) : kurva segitiga dengan *range* [0.3, 0.6, 0,99]
- Fungsi keanggotaan Sangat-Sangat Tinggi (SST) : kurva segitiga dengan *range* [0.6, 0.99, 1.0]

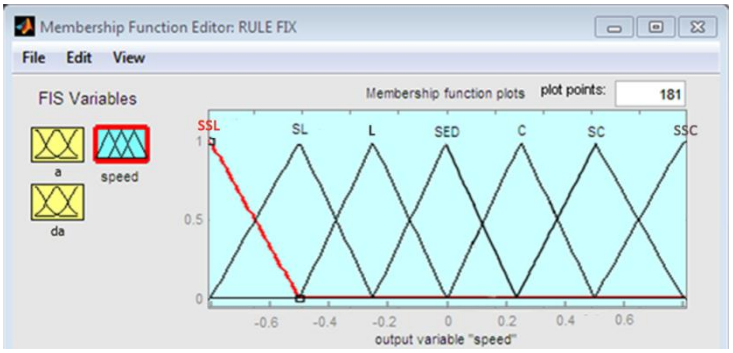


Gambar 5.12 *Membership Function Editor input (Δα)*

Membership function editor untuk *input* yang kedua yaitu perubahan percepatan terhadap sumbu Z ditunjukkan oleh Gambar 5.12. Berikut ini penjelasan mengenai fungsi keanggotaan 7x7 *input perubahan percepatan (Δα)* yang dirancang :

- Fungsi keanggotaan Sangat-Sangat Rendah (SSR) : kurva segitiga dengan *range* [-1.0, -0.99, -0.6]
- Fungsi keanggotaan Sangat Rendah (SR) : kurva segitiga dengan *range* [-0.99, -0.6, -0.3]
- Fungsi keanggotaan Rendah (R) : kurva segitiga dengan *range* [-0.6, -0.3, 0]

- Fungsi keanggotaan Sedang (SED) : kurva segitiga dengan *range* [-0.3, 0, 0.3]
- Fungsi keanggotaan Tinggi (T) : kurva segitiga dengan *range* [0, 0.3, 0.6]
- Fungsi keanggotaan Sangat Tinggi (ST) : kurva segitiga dengan *range* [0.3, 0.6, 0,99]
- Fungsi keanggotaan Sangat-Sangat Tinggi (SST) : kurva segitiga dengan *range* [0.6, 0.99, 1.0]



Gambar 5.13 *Membership Function Editor output (Speed)*

Membership function editor untuk *output* berupa kecepatan Ditunjukkan oleh Gambar 5.13. Berikut ini penjelasan mengenai fungsi keanggotaan 7x7*output (Speed)* yang dirancang :

- Fungsi keanggotaan Sangat-Sangat Lambat (SSL) : kurva segitiga dengan *range* [-1.0, -0.8, -0.5]
- Fungsi keanggotaan Sangat Lambat (SL) : kurva segitiga dengan *range* [-0.8, -0.5, -0.25]
- Fungsi keanggotaan Lambat (L) : kurva segitiga dengan *range* [-0.5, -0.25, -0]
- Fungsi keanggotaan Sedang (SED) : kurva segitiga dengan *range* [-0.25, -0, 0.15]
- Fungsi keanggotaan Cepat (C) : kurva segitiga dengan *range* [-0, 0.25, 0.5]
- Fungsi keanggotaan Sangat Cepat (SC) : kurva segitiga dengan *range* [0.15, 0.5, 0,8]

- Fungsi keanggotaan Sangat-Sangat Cepat (SSC) : kurva segitiga dengan *range* [0.5, 0.8, 1.0]

Nilai dari *range* yang digunakan diatas adalah nilai dari hasil penyesuaian ulang selama beberapa kali untuk menghasilkan *output* yang sesuai dengan penelitian ini.

Setelah melakukan proses penentuan fungsi keanggotaan untuk setiap variable *input* dan *output* maka selanjutnya akan dilanjutkan dengan pembuatan aturan dasar *fuzzy* yang digunakan untuk menentukan aturan yang nantinya akan menghasilkan keluaran berupa kecepatan motor. Tabel 5.2 merupakan tabel dari aturan dasar pada penelitian ini. Cara pembacaan dari basis aturan adalah IF <fungsi keanggotaan 1> AND <fungsi keanggotaan 2> THEN kecepatan <fungsi keanggotaan 3>.

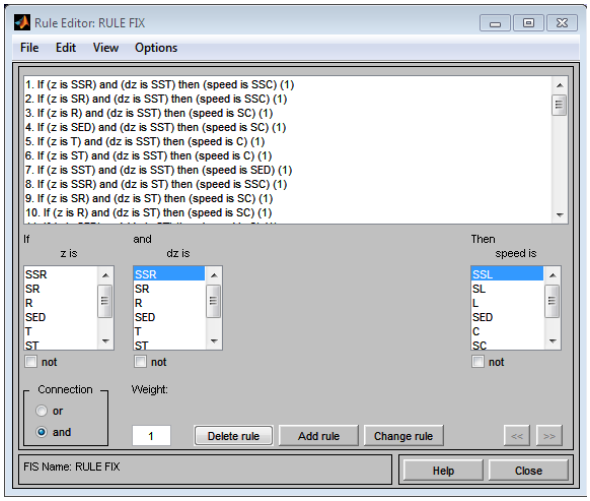
Tabel 5.2 Basis Aturan 7x7

(α) / *(Δα)*	SSR	SR	R	SED	T	ST	SST
SST	SSC	SSC	SC	SC	C	C	SED
ST	SSC	SC	SC	C	C	SED	L
T	SC	SC	C	C	SED	L	L
SED	SC	C	C	SED	L	L	SL
R	C	C	SED	L	L	SL	SL
SR	C	SED	L	L	SL	SL	SSL
SSR	SED	L	L	SL	SL	SSL	SSL

Fuzzy inference sistem yang digunakan adalah Mamdani dengan *defuzifikasi* yang digunakan adalah metode *centroid*. Untuk mengetahui 49 *rule* yang terapat pada penelitian ini dapat dilihat pada *rule* editor.

3. *Rule Editor*

Setelah selesai menentukan fungsi keanggotaan selanjutnya kita masuk tahapan pembuatan *rule* pada *rule editor.* Fungsi dari *rule editor* sendiri adalah untuk merancang maupun menampilkan aturan yang akan atau telah kita buat. Pada penelitian ini digunakan basis aturan 7x7 dan menghasilkan 49 rule yang akan dibuat pada rule editor. Gambar 5.14 merupakan gambaran dari *rule editor.*



Gambar 5.14 Tampilan dari *Rule Editor*

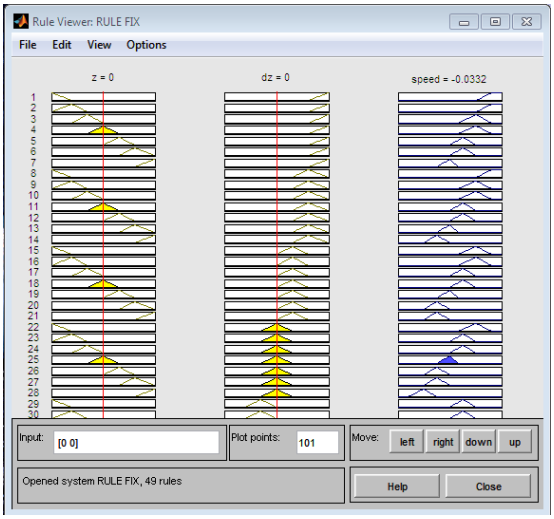
Di bawah ini merupakan 49 *rule* yang di gunakan yaitu;

1. If (a is SSR) and (da is SST) then (speed is SSC) (1)
2. If (a is SR) and (da is SST) then (speed is SSC) (1)
3. If (a is R) and (da is SST) then (speed is SC) (1)
4. If (a is SED) and (da is SST) then (speed is SC) (1)
5. If (a is T) and (da is SST) then (speed is C) (1)
6. If (a is ST) and (da is SST) then (speed is C) (1)
7. If (a is SST) and (da is SST) then (speed is SED) (1)
8. If (a is SSR) and (da is ST) then (speed is SSC) (1)
9. If (a is SR) and (da is ST) then (speed is SC) (1)
10. If (a is R) and (da is ST) then (speed is SC) (1)
11. If (a is SED) and (da is ST) then (speed is C) (1)
12. If (a is T) and (da is ST) then (speed is C) (1)
13. If (a is ST) and (da is ST) then (speed is SED) (1)
14. If (a is SST) and (da is ST) then (speed is L) (1)
15. If (a is SSR) and (da is T) then (speed is SC) (1)
16. If (a is SR) and (da is T) then (speed is SC) (1)
17. If (a is R) and (da is T) then (speed is C) (1)
18. If (a is SED) and (da is T) then (speed is C) (1)
19. If (a is T) and (da is T) then (speed is SED) (1)
20. If (a is ST) and (dz is T) then (speed is L) (1)

21. If (a is SST) and (da is T) then (speed is L) (1)
22. If (a is SSR) and (da is SED) then (speed is SC) (1)
23. If (a is SR) and (da is SED) then (speed is C) (1)
24. If (a is R) and (da is SED) then (speed is C) (1)
25. If (a is SED) and (da is SED) then (speed is SED) (1)
26. If (a is T) and (da is SED) then (speed is L) (1)
27. If (a is ST) and (da is SED) then (speed is L) (1)
28. If (a is SST) and (da is SED) then (speed is SL) (1)
29. If (a is SSR) and (da is R) then (speed is C) (1)
30. If (a is SR) and (da is R) then (speed is C) (1)
31. If (a is R) and (da is R) then (speed is SED) (1)
32. If (a is SED) and (da is R) then (speed is L) (1)
33. If (a is T) and (da is R) then (speed is L) (1)
34. If (a is ST) and (da is R) then (speed is SL) (1)
35. If (a is SST) and (da is R) then (speed is SL) (1)
36. If (a is SSR) and (da is SR) then (speed is C) (1)
37. If (a is SR) and (da is SR) then (speed is SED) (1)
38. If (a is R) and (da is SR) then (speed is L) (1)
39. If (a is SED) and (da is SR) then (speed is L) (1)
40. If (a is T) and (da is SR) then (speed is SL) (1)
41. If (a is ST) and (da is SR) then (speed is SL) (1)
42. If (a is SST) and (da is SR) then (speed is SSL) (1)
43. If (a is SSR) and (da is SSR) then (speed is SED) (1)
44. If (a is SR) and (da is SSR) then (speed is L) (1)
45. If (a is R) and (da is SSR) then (speed is L) (1)
46. If (a is SED) and (da is SSR) then (speed is SL) (1)
47. If (a is T) and (da is SSR) then (speed is SL) (1)
48. If (a is ST) and (da is SSR) then (speed is SSL) (1)
49. If (a is SST) and (da is SSR) then (speed is SSL) (1)

4. *Rule Viewer*

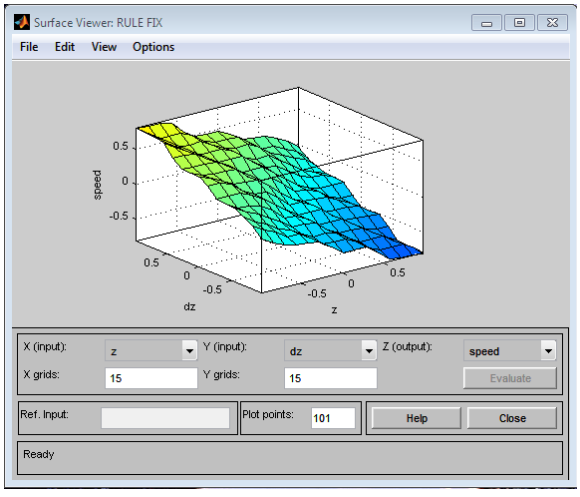
   Tahapan selanjutnya adalah *rule viewer* fungsi dari *rule viewer* adalah untuk melihat alur penalaran *fuzzy* pada sistem. Gambar 5.15 merupakan gambaran dari *rule viewer.*



Gambar 5.15 Tampilan dari *Rule Viewer*

5. *Surface Viewer*

   Tahapan ahir dalam pembuatan suatu sistem *fuzzy* yaitu *surface viewer*. *Surface viewer* sendiri berguna untuk melihat gambar pemetaan variabel-variabel *input* dan variabel-variabel *output*. Gambaran dari *surface viewer* dapat dilihat pada Gambar 5.16.



Gambar 5.16 Tampilan dari *Surface Viewer*

5.3.2 Implementasi Pada Kode Program

Setelah proses pembuatan rule pada *fuzzy inference system* telah selesai pada penelitian ini akan dilanjutkan dengan pengimplementasian *rule* yang telah dibuat menggunakan bahasa pemograman *python*. Adapun beberapa tahapan

dalam pengimplementasian kode program tersebut diantaranya yaitu *fuzzyfikasi,* aturan dasar, penalaran, dan *defuzzyfikasi*.

```
1     def FuzzyfyMIN(BB, BA, input) :
2        if (input <= BB) :
3           u = 1
4        elif (input >= BA):
5           u = 0
6        else :
7           u = (BA - input) / (BA - BB)
8        return u
9
10    def FuzzyfyMED(BB, BA, BT, input) :
11       if  (input <= BB) or (input >= BA) :
12          u = 0
13       elif  input > BB and input <=BT :
14          u = (input-BB)/(BT-BB)
15       elif input > BT and input <=BA  :
16          u = (BA-input)/(BA-BT)
17       else:
18          u = 1
19       return u
20
21    def FuzzyfyMAX(BB, BA, input):
22       if (input <= BB):
23          u = 0
24       elif (input >= BA):
25          u = 1;
26       else:
27          u = (input - BB) / (BA - BB)
28       return u;
29
30    def hitungFuzzy(Nilai_A, Nilai_DA):
31       ret = 0;
32
33       # [-6.35 -5.15 -4.55 -2.15]
34       BatasBawah_Nilai = -0.99  ;
35       BatasTengah_Nilai = -0.6   ;
36       BatasAtas_Nilai = -0.3 ;
37       input = Nilai_A;
38
39       #[-4.5 -3 -1.5]
40       A_SSR = FuzzyfyMIN(BatasBawah_Nilai, BatasTengah_Nilai, input);
41       A_SR = FuzzyfyMED(BatasBawah_Nilai,BatasAtas_Nilai,
42    BatasTengah_Nilai, input);
43
44       # [-3 -1.5 0]
45       BatasBawah_Nilai = BatasTengah_Nilai;
46       BatasTengah_Nilai = BatasAtas_Nilai;
47       BatasAtas_Nilai = 0 ;
48       A_R = FuzzyfyMED(BatasBawah_Nilai,BatasAtas_Nilai,
49    BatasTengah_Nilai, input);
50
51       # [-1.5 0 1.5]
52       BatasBawah_Nilai = BatasTengah_Nilai ;
53       BatasTengah_Nilai = BatasAtas_Nilai   ;
54       BatasAtas_Nilai = 0.3 ;
55       A_SED = FuzzyfyMED(BatasBawah_Nilai,BatasAtas_Nilai,
56    BatasTengah_Nilai, input);
57
58       # [0.02646 1.526 3.026]
59       BatasBawah_Nilai = BatasTengah_Nilai ;
60       BatasTengah_Nilai = BatasAtas_Nilai   ;
61       BatasAtas_Nilai = 0.6;
62       A_T = FuzzyfyMED(BatasBawah_Nilai,BatasAtas_Nilai,
```

```
63      BatasTengah_Nilai, input);
64
65         # [1.5 3 4.5]
66         BatasBawah_Nilai = BatasTengah_Nilai ;
67         BatasTengah_Nilai = BatasAtas_Nilai   ;
68         BatasAtas_Nilai = 0.99 ;
69         A_ST = FuzzyfyMED(BatasBawah_Nilai,BatasAtas_Nilai,
70      BatasTengah_Nilai, input);
71         A_SST = FuzzyfyMAX(BatasTengah_Nilai, BatasAtas_Nilai, input);
72
73         #  [-9.7 -7.8 -5.7 -3.8]
74         BatasBawah_Nilai = -0.99 ;
75         BatasTengah_Nilai = -0.6   ;
76         BatasAtas_Nilai = -0.3 ;
77
78         input = Nilai_DA;
79
80         # [-5.7 -3.8 -1.9]
81         dA_SSR = FuzzyfyMIN(BatasBawah_Nilai, BatasTengah_Nilai, input);
82         dA_SR = FuzzyfyMED(BatasBawah_Nilai,BatasAtas_Nilai,
83      BatasTengah_Nilai, input);
84
85         #  [-3.8 -1.95 0]
86         BatasBawah_Nilai = BatasTengah_Nilai;
87         BatasTengah_Nilai = BatasAtas_Nilai;
88         BatasAtas_Nilai = 0 ;
89         dA_R = FuzzyfyMED(BatasBawah_Nilai,BatasAtas_Nilai,
90      BatasTengah_Nilai, input);
91
92         #  [-1.9 0 1.9]
93         BatasBawah_Nilai = BatasTengah_Nilai ;
94         BatasTengah_Nilai = BatasAtas_Nilai   ;
95         BatasAtas_Nilai = 0.3;
96         dA_SED = FuzzyfyMED(BatasBawah_Nilai,BatasAtas_Nilai,
97      BatasTengah_Nilai, input);
98
99         # [-0.0317 1.87 3.77]
100        BatasBawah_Nilai = BatasTengah_Nilai ;
101        BatasTengah_Nilai = BatasAtas_Nilai   ;
102        BatasAtas_Nilai = 0.6 ;
103        dA_T = FuzzyfyMED(BatasBawah_Nilai,BatasAtas_Nilai,
104     BatasTengah_Nilai, input);
105
106        #  [1.9 3.8 5.7]
107        BatasBawah_Nilai = BatasTengah_Nilai ;
108        BatasTengah_Nilai = BatasAtas_Nilai   ;
109        BatasAtas_Nilai = 0.99 ;
110        dA_ST = FuzzyfyMED(BatasBawah_Nilai,BatasAtas_Nilai,
111     BatasTengah_Nilai, input);
112        dA_SST = FuzzyfyMAX(BatasTengah_Nilai, BatasAtas_Nilai, input);
```

Pada potongan kode program di atas merupakan tahapan dari *fuzzyfikasi.* Pada tahapan *fuzzyfikasi* akan dibuat tiga metod rumus untuk mempermudah pemetaan nilai input yang akan di jadikan masukan dalam fungsi keanggotaan himpunan *fuzzy*. Pada baris 30 dibuat metod dengan nama `def hitungFuzzy`. Metode ini digunakan untuk pengisian nilai *input* pada *fuzzyfikasi* yang akan dijadikan sebagai fungsi kenggotaan. Terdapat 7 fungsi keanggotaan diantaranya yaitu sangat-sangat rendah (SSR), sangat rendah (SR), rendah(R), sedang (SED), tinggi (T), sangat tinggi (ST), dan sangat-sangat tinggi (SST). Berikut merupakan penjelasan perbaris dari potongan kode program diatas.

1. Baris 1-8 : Digunakan untuk *fuzzifikasi* nilai minimum
2. Baris 10-19 : Digunakan untuk *fuzzifikasi* nilai medium
3. Baris 21-28 : Digunakan untuk *fuzzifikasi* nilai maximum
4. Baris 31-112 : Digunakan untuk untuk proses penginputan parameter pada *fuzzy*

```
1     #1. If (A is SSR) and (dA is SST) then (speed is SSC) (1)
2        p1 = min(round(A_SSR, 2), round(dA_SST, 2))
3
4     #2. If (A is SR) and (dA is SST) then (speed is SSC) (1)
5        p2 = min(round(A_SR, 2), round(dA_SST, 2))
6
7     #3. If (A is R) and (dA is SST) then (speed is SC) (1)
8        p3 = min(round(A_R, 2), round(dA_SST, 2))
9
10    #4. If (A is SED) and (dA is SST) then (speed is SC) (1)
11       p4 = min(round(A_SED, 2), round(dA_SST, 2))
12
13    #5. If (A is T) and (dA is SST) then (speed is C) (1)
14       p5 = min(round(A_T, 2), round(dA_SST, 2))
15
16    #6. If (A is ST) and (dA is SST) then (speed is C) (1)
17       p6 = min(round(A_ST, 2), round(dA_SST, 2))
18
19    #7. If (A is SST) and (dA is SST) then (speed is SED) (1)
20       p7 = min(round(A_SST, 2), round(dA_SST, 2))
21
22    #8. If (A is SSR) and (dA is ST) then (speed is SSC) (1)
23       p8 = min(round(A_SSR, 2), round(dA_ST, 2))
24
25    #9. If (A is SR) and (dA is ST) then (speed is SC) (1)
26       p9 = min(round(A_SR, 2), round(dA_ST, 2))
27
28    #10. If (A is R) and (dA is ST) then (speed is SC) (1)
29       p10 = min(round(A_R, 2), round(dA_ST, 2))
30
31    #11. If (A is SED) and (dA is ST) then (speed is C) (1)
32       p11 = min(round(A_SED, 2), round(dA_ST, 2))
33
34    #12. If (A is T) and (dA is ST) then (speed is C) (1)
35       p12 = min(round(A_T, 2), round(AZ_ST, 2))
36
37    #13. If (A is ST) and (dA is ST) then (speed is SED) (1)
38       p13 = min(round(A_ST, 2), round(dA_ST, 2))
39
40    #14. If (A is SST) and (dA is ST) then (speed is L) (1)
41       p14 = min(round(A_SST, 2), round(dA_ST, 2))
42
43    #15. If (A is SSR) and (dA is T) then (speed is SC) (1)
44       p15 = min(round(A_SSR, 2), round(dA_T, 2))
45
46    #16. If (A is SR) and (dA is T) then (speed is SC) (1)
47       p16 = min(round(A_SR, 2), round(dA_T, 2))
48
49    #17. If (A is R) and (dA is T) then (speed is C) (1)
50       p17 = min(round(A_R, 2), round(dA_T, 2))
51
52    #18. If (A is SED) and (dA is T) then (speed is C) (1)
53       p18 = min(round(A_SED, 2), round(dA_T, 2))
54
```

```
    #19. If (A is T) and (dA is T) then (speed is SED) (1)
       p19 = min(round(A_T, 2), round(dA_T, 2))

    #20. If (A is ST) and (dA is T) then (speed is L) (1)
       p20 = min(round(A_ST, 2), round(dA_T, 2))

    #21. If (A is SST) and (dA is T) then (speed is L) (1)
       p21 = min(round(Z_SST, 2), round(dA_T, 2))

    #22. If (A is SSR) and (dA is SED) then (speed is SC) (1)
       p22 = min(round(A_SSR, 2), round(dA_SED, 2))

    #23. If (A is SR) and (dA is SED) then (speed is C) (1)
       p23 = min(round(A_SR, 2), round(dA_SED, 2))

    #24. If (A is R) and (dA is SED) then (speed is C) (1)
       p24 = min(round(A_R, 2), round(dA_SED, 2))

    #25. If (A is SED) and (dA is SED) then (speed is SED) (1)
       p25 = min(round(A_SED, 2), round(dA_SED, 2))

    #26. If (A is T) and (dA is SED) then (speed is L) (1)
       p26 = min(round(A_T, 2), round(dA_SED, 2))

    #27. If (A is ST) and (dA is SED) then (speed is L) (1)
       p27 = min(round(A_ST, 2), round(dA_SED, 2))

    #28. If (A is SST) and (dA is SED) then (speed is SL) (1)
       p28 = min(round(A_SST, 2), round(dA_SED, 2))

    #29. If (A is SSR) and (dA is R) then (speed is C) (1)
       p29 = min(round(A_SSR, 2), round(dA_R, 2))

    #30. If (A is SR) and (dA is R) then (speed is C) (1)
       p30 = min(round(A_SR, 2), round(dA_R, 2))

    #31. If (A is R) and (dA is R) then (speed is SED) (1)
       p31 = min(round(A_R, 2), round(dA_R, 2))

    #32. If (A is SED) and (dA is R) then (speed is L) (1)
       p32 = min(round(A_SED, 2), round(dA_R, 2))

    #33. If (A is T) and (dA is R) then (speed is L) (1)
       p33 = min(round(A_T, 2), round(dA_R, 2))

    #34. If (A is ST) and (dA is R) then (speed is SL) (1)
       p34 = min(round(A_ST, 2), round(dA_R, 2))

    #35. If (A is SST) and (dA is R) then (speed is SL) (1)
       p35 = min(round(A_SST, 2), round(dA_R, 2))

    #36. If (A is SSR) and (dA is SR) then (speed is C) (1)
       p36 = min(round(A_SSR, 2), round(dA_SR, 2))

    #37. If (A is SR) and (dA is SR) then (speed is SED) (1)
       p37 = min(round(A_SR, 2), round(dA_SR, 2))

    #38. If (A is R) and (dA is SR) then (speed is L) (1)
       p38 = min(round(A_R, 2), round(dA_SR, 2))

    #39. If (A is SED) and (dA is SR) then (speed is L) (1)
       p39 = min(round(A_SED, 2), round(dA_SR, 2))

    #40. If (A is T) and (dA is SR) then (speed is SL) (1)
       p40 = min(round(A_T, 2), round(dA_SR, 2))
```

```
120
121     #41. If (A is ST) and (dA is SR) then (speed is SL) (1)
122        p41 = min(round(A_ST, 2), round(dA_SR, 2))
123
124     #42. If (A is SST) and (dA is SR) then (speed is SSL) (1)
125        p42 = min(round(A_SST, 2), round(dA_SR, 2))
126
127     #43. If (A is SSR) and (dA is SSR) then (speed is SED) (1)
128        p43 = min(round(A_SSR, 2), round(dA_SSR, 2))
129
130     #44. If (A is SR) and (dA is SSR) then (speed is L) (1)
131        p44 = min(round(A_SR, 2), round(dA_SSR, 2))
132
133     #45. If (A is R) and (dA is SSR) then (speed is L) (1)
134        p45 = min(round(A_R, 2), round(dA_SSR, 2))
135
136     #46. If (A is SED) and (dA is SSR) then (speed is SL) (1)
137        p46 = min(round(A_SED, 2), round(dA_SSR, 2))
138
139     #47. If (A is T) and (dA is SSR) then (speed is SL) (1)
140        p47 = min(round(A_T, 2), round(dA_SSR, 2))
141
142     #48. If (A is ST) and (dA is SSR) then (speed is SSL) (1)
143        p48 = min(round(A_ST, 2), round(dA_SSR, 2))
144
145     #49. If (A is SST) and (dA is SSR) then (speed is SSL) (1)
146        p49 = min(round(A_SST, 2), round(dA_SSR, 2))
147
148        Min1 = min(p1, p2)
149        Min2 = min(p3, p4)
150        Min3 = min(p5, p6)
151        Min4 = min(p7, p8)
152        Min5 = min(p9, p10)
153        Min6 = min(p11, p12)
154        Min7 = min(p13, p14)
155        Min8 = min(p15, p16)
156        Min9 = min(p17, p18)
157        Min10= min(p19, p20)
158        Min11= min(p21, p22)
159        Min12= min(p23, p24)
160        Min13= min(p25, p26)
161        Min14= min(p27, p28)
162        Min15= min(p29, p30)
163        Min16= min(p31, p32)
164        Min17= min(p33, p34)
165        Min18= min(p35, p36)
166        Min19= min(p37, p38)
167        Min20= min(p39, p40)
168        Min21= min(p41, p42)
169        Min22= min(p43, p44)
170        Min23= min(p45, p46)
171        Min24= min(p47, p48)
172
173        MIN = min(Min24, p49)
174        MIN = min(MIN, Min1 )
175        MIN = min(MIN, Min2 )
176        MIN = min(MIN, Min3 )
177        MIN = min(MIN, Min4 )
178        MIN = min(MIN, Min5 )
179        MIN = min(MIN, Min6 )
180        MIN = min(MIN, Min7 )
181        MIN = min(MIN, Min8 )
182        MIN = min(MIN, Min9 )
183        MIN = min(MIN, Min10)
184        MIN = min(MIN, Min11)
```

```
185         MIN = min(MIN, Min12)
186         MIN = min(MIN, Min13)
187         MIN = min(MIN, Min14)
188         MIN = min(MIN, Min15)
189         MIN = min(MIN, Min16)
190         MIN = min(MIN, Min17)
191         MIN = min(MIN, Min18)
192         MIN = min(MIN, Min19)
193         MIN = min(MIN, Min20)
194         MIN = min(MIN, Min21)
195         MIN = min(MIN, Min22)
196         MIN = min(MIN, Min23)
197
198         Max1 = max(p1, p2)
199         Max2 = max(p3, p4)
200         Max3 = max(p5, p6)
201         Max4 = max(p7, p8)
202         Max5 = max(p9, p10)
203         Max6 = max(p11, p12)
204         Max7 = max(p13, p14)
205         Max8 = max(p15, p16)
206         Max9 = max(p17, p18)
207         Max10= max(p19, p20)
208         Max11= max(p21, p22)
209         Max12= max(p23, p24)
210         Max13= max(p25, p26)
211         Max14= max(p27, p28)
212         Max15= max(p29, p30)
213         Max16= max(p31, p32)
214         Max17= max(p33, p34)
215         Max18= max(p35, p36)
216         Max19= max(p37, p38)
217         Max20= max(p39, p40)
218         Max21= max(p41, p42)
219         Max22= max(p43, p44)
220         Max23= max(p45, p46)
221         Max24= max(p47, p48)
222         MAX = max(Max24, p49)
223         MAX = max(MAX, Max1 )
224         MAX = max(MAX, Max2 )
225         MAX = max(MAX, Max3 )
226         MAX = max(MAX, Max4 )
227         MAX = max(MAX, Max5 )
228         MAX = max(MAX, Max6 )
229         MAX = max(MAX, Max7 )
230         MAX = max(MAX, Max8 )
231         MAX = max(MAX, Max9 )
232         MAX = max(MAX, Max10)
233         MAX = max(MAX, Max11)
234         MAX = max(MAX, Max12)
235         MAX = max(MAX, Max13)
236         MAX = max(MAX, Max14)
237         MAX = max(MAX, Max15)
238         MAX = max(MAX, Max16)
239         MAX = max(MAX, Max17)
240         MAX = max(MAX, Max18)
241         MAX = max(MAX, Max19)
242         MAX = max(MAX, Max20)
243         MAX = max(MAX, Max21)
244         MAX = max(MAX, Max22)
245         MAX = max(MAX, Max23)
```

Potongan program diatas merupakan potongan dari tahap aturan dasar dan penalaran. Terdapat 49 *rule* yang telah di buat seperti yang ditunjukkan pada

baris ke 1-146. Sedangkan pada baris 148-171 merupakan tahapan dari perbandingan nilai minimum dari setiap *rule* yang telah dibuat contoh nya pada baris 148. Pada baris ini dibuat sebuah fariabel dengan nama `Min1 = min(p1,p2)` maksud dari variabel tersebut adalah pada `Min1` akan dilakukan perbandingan nilai minimum dari rule `p1` dan `p2`. Setelah perbandingan antar *rule* selesai selanjutnya akan dilakukan lagi perbandingan antar variabel `MIN` yang telah di buat contoh nya seperti pada baris 173-196 perbandingan ini akan terus dilakukan hingga mendapat nilai paling minimum dari semua *rule* yang di bandingkan. Setelah variabel `MIN` selesai selanjutnya akan dilanjutkan pada variabel `MAX` seperti ditunjukan pada baris 198-246 proses yang dilakukan sama seperti `MIN` mamun pada baris ini yang dicari adalah nilai maksimum. Berikut adalah penjelasan perbaris dari potongan kode program di atas.

a. Baris 1-146 : Digunakan untuk inisialisasi aturan
b. Baris 148-171 : Digunakan untuk mencari nilai minimal
c. Baris 173-196 : Digunakan untuk perbandingan nilai minimal antar aturan
d. Baris 198-221 : Digunakan untuk mencari nilai maximum
e. Baris 223-246 : Digunakan untuk perbandingan nilai minimal antar aturan

```
1         # [-1660 -1330 -1000 -670]
2         MinSpeed = -0.8
3         MaxSpeed = -0.5
4         a1 = (MIN * (MaxSpeed - MinSpeed)) + (MaxSpeed )
5
6         # [-1000 -670 -340]
7         MinSpeed = MaxSpeed
8         MaxSpeed = -0.25
9         a2 = (MAX * (MaxSpeed - MinSpeed)) + (MaxSpeed )
10        # [-670 -340 0]
11
12        MinSpeed = MaxSpeed
13        MaxSpeed = -0.15
14        a3= (MAX * (MaxSpeed - MinSpeed)) + (MaxSpeed )
15        # [-340 0 340]
16
17        MinSpeed = MaxSpeed
18        MaxSpeed = 0.15
19        a4 = (MAX * (MaxSpeed - MinSpeed)) + (MaxSpeed )
20        # [0 340 670]
21
22        MinSpeed = MaxSpeed
23        MaxSpeed = 0.25
24        a5 = (MAX * (MaxSpeed - MinSpeed)) + (MaxSpeed )
25        # [340 670 1000]
26
27        MinSpeed = MaxSpeed
28        MaxSpeed = 0.5
29        a6 = (MAX * (MaxSpeed - MinSpeed)) + (MaxSpeed )
30        # [670 1000 1330 1660]
31
32        MinSpeed = MaxSpeed
33        MaxSpeed = 0.8
34        a7 =  (MIN * (MaxSpeed - MinSpeed)) + (MaxSpeed )
35
36    #pemetaan area output
37        m1 = round((MIN / 2) * (pow(a1, 2)), 2)
```

```
38
39          m2 = round((((1 / (MaxSpeed - MinSpeed)) / 2) * (pow(a2, 3))) -
40      ((MinSpeed / (MaxSpeed - MinSpeed)) / 2) * pow(a2, 2)) - (((1 /
41      (MaxSpeed - MinSpeed)) / 3) * (pow(a1, 3))) - (((MinSpeed / (MaxSpeed
42      - MinSpeed)) / 2 * (pow(a1, 2))))
43
44          m3 = round((((1 / (MaxSpeed - MinSpeed)) / 2) * (pow(a3, 3))) -
45      ((MinSpeed / (MaxSpeed - MinSpeed)) / 2) * pow(a3, 2)) - (((1 /
46      (MaxSpeed - MinSpeed)) / 3) * (pow(a2, 3))) - (((MinSpeed / (MaxSpeed
47      - MinSpeed)) / 2 * (pow(a2, 2))))
48
49          m4 = round((((1 / (MaxSpeed - MinSpeed)) / 2) * (pow(a4, 3))) -
50      ((MinSpeed / (MaxSpeed - MinSpeed)) / 2) * pow(a4, 2)) - (((1 /
51      (MaxSpeed - MinSpeed)) / 3) * (pow(a3, 3))) - (((MinSpeed / (MaxSpeed
52      - MinSpeed)) / 2 * (pow(a3, 2))))
53
54          m5 = round((((1 / (MaxSpeed - MinSpeed)) / 2) * (pow(a5, 3))) -
55      ((MinSpeed / (MaxSpeed - MinSpeed)) / 2) * pow(a5, 2)) - (((1 /
56      (MaxSpeed - MinSpeed)) / 3) * (pow(a4, 3))) - (((MinSpeed / (MaxSpeed
57      - MinSpeed)) / 2 * (pow(a4, 2))))
58
59          m6 = round((((1 / (MaxSpeed - MinSpeed)) / 2) * (pow(a6, 3))) -
60      ((MinSpeed / (MaxSpeed - MinSpeed)) / 2) * pow(a6, 2)) - (((1 /
61      (MaxSpeed - MinSpeed)) / 3) * (pow(a5, 3))) - (((MinSpeed / (MaxSpeed
62      - MinSpeed)) / 2 * (pow(a5, 2))))
63
64      m7 = round(((MAX * pow(MaxSpeed, 2)) / 2) - (MAX * (MaxSpeed -
65      MinSpeed) / 2) * (MaxSpeed - MinSpeed), 2)
66
67          AA1 = a1 * MIN
68          AA2 = (MIN + MAX) * (a2 - a1) / 2
69          AA3 = (MIN + MAX) * (a3 - a2) / 2
70          AA4 = (MIN + MAX) * (a4 - a3) / 2
71          AA5 = (MIN + MAX) * (a5 - a4) / 2
72          AA6 = (MIN + MAX) * (a6 - a5) / 2
73          AA7 = (MaxSpeed - (MaxSpeed - MinSpeed)) * MAX
74          z =  (m1 + m2 + m3 +m4 +m5 +m6 + m7) / (AA1 + AA2 + AA3 + AA4 +
75      AA5 + AA6 + AA7) /5
76          return z
77
```

Potongan program diatas adalah tahapan dari penentuan area *output* dari *fuzzyfikasi.* Selain penentuan area *output* pada potongan baris program diatas juga merupakan tahapan dari *defuzzyfikasi.* Berikut merupakan penjelasan perbaris dari potongan program di atas.

a. Baris 1-34 : Digunakan untuk mencari kriteria *output*

b. Baris 36-65 : Digunakan untuk mencari batas max-min area *output*

c. Baris 67-78 : Digunakan untuk proses *defuzzyfikasi*