BAB 2
LANDASAN TEORI

2.1 *Tower Defense*

Game Tower Defense adalah *game* bertipe *Real Time Strategy* yang berfokus pada pengaturan sumber daya yang dimiliki pemain dan penempatan menara pertahanan (*Tower*). Dalam bentuk yang paling sederhana, sebuah *game Tower Defense* terdiri dari pemain yang bertugas membeli dan mengorganisir *Tower* yang akan menembak berbagai jenis musuh yang menyerang pertahanan pemain.

2.1.1 Sejarah

Rampart merupakan prototype game *Tower Defense* pertama yang dipublish oleh *Atari game* sebuah perusahaan *game* pada tahun 1990.



Gambar 2. 1 Tampilan *Game* Rampart

Pada awal tahun 2000-an *Map* (Peta) pada *game* Rampart menjadi acuan pada *game-game* seperti StartCraft, Age of Empires II, dan Warcraft III. Pada tahun 2007 para *developer game indie* mulai menggunakan Adobe Flash untuk membuat *game Tower Defense* berbasis *web* dan *desktop*, diawali dengan dirilisnya Flash *Element Tower Defense* pada bulan Januari tahun 2007 yang dibuat oleh David Scott. Game Tower Defense semakin populer dan meraih penghargaan Independen Game Festival (IGF) yang merupakan pertemuan tahunan terbesar dari perindustrian *game indie*.

Keberhasilan ini mendorong para developer untuk mengembangkan *genre game* ini untuk versi konsol dan ponsel.

2.1.2 Elemen *Game Tower Defense*

Berikut ini elemen-elemen umum yang terdapat pada *game Tower Defense* :

1. *Terrain* atau *Map*

   Map (Peta) merupakan elemen penting pada *Game Tower Defense*, *Map* menentukan pemain dalam strategi penempatan *Tower*, dalam permainan *Tower Defense* klasik, *map* masih mengadopsi sistem *linear-path*, dimana sistem *map* tersebut membatasi kepada pemain dalam menempatkan *tower*.

2. *Tower*

   Tower pada *game-game Tower Defense* biasanya dirancang dengan fungsionalitas yang berbeda, sehingga pemain perlu strategi yang tepat agar penempatan *tower* lebih efektif ketika menghadang pergerakan musuh.

3. *Enemy* (Musuh)

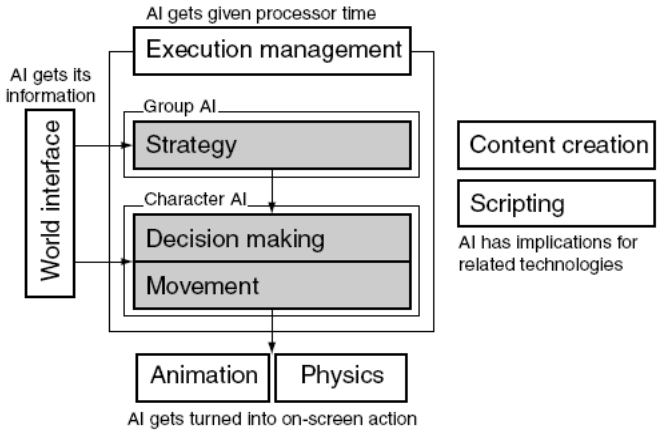
   Musuh pada *game Tower Defense* biasanya dirancang dengan *skill*, *hit point* dan *armor* yang berbeda-beda, sehingga perlu jenis *tower* yang berbeda pula untuk menghadang setiap tipe musuh yang berbeda.

2.2 AI (*Artificial Intelligence*)

AI *(Artificial Intelligence)* atau kecerdasan buatan adalah adalah bagian dari ilmu komputer yang mempelajari bagaimana membuat mesin (komputer) dapat melakukan pekerjaan seperti dan sebaik yang dilakukan oleh manusia bahkan bisa lebih baik daripada yang dilakukan manusia **[1]**. Cerdas (*Intelligent*) adalah memiliki pengetahuan, pengalaman, penalaran (bagaimana membuat keputusan dan mengambil tindakan).

Kecerdasan buatan (AI) dalam *game* komputer mencakup perilaku dan proses pengambilan keputusan dari lawan (juga dikenal sebagai karakter nonplayer atau NPC). Teknik *machine-learning* dapat memungkinkan NPC untuk meningkatkan

kemampuan mereka dengan belajar dari kesalahan dan keberhasilan, untuk secara otomatis beradaptasi dengan kekuatan dan kelemahan, atau untuk belajar dengan meniru taktik lawan (pemain). Arsitektur model AI dalam permainan , dapat dilihat pada Gambar 2.2 **[4]**.



Gambar 2. 2 Arsitektur Model AI

Pada Gambar 2.2 arsitektur model AI didalam sebuah *game* dibagi menjadi tiga bagian :

1. *Movement* (Gerakan)

   Movement merupakan komponen paling dasar dari sebuah model AI, gerakan pada AI biasanya mengacu pada perintah algoritma yang diterapkan. Misalnya perintah menyerang ketika karakter musuh dalam keadaan tidak bersenjata dan lain sebagainya.

2. *Decision Making* (Pengambilan keputusan)

   Seperti halnya manusia AI dalam sebuah *game* juga harus mampu mengambil suatu keputusan, misalnya menyerang, berdiri, bersembunyi dan berpatroli.

3. *Strategy*

   Elemen ini biasanya digunakan untuk sekelompok karakter (tim), setiap karakter dalam tim dapat mengambil keputusan sendiri, tetapi secara keseluruhan pengambilan keputusan mereka dipengaruhi oleh strategi tim

2.3 Logika *Fuzzy* (*Fuzzy Logic*)

Logika *Fuzzy* adalah cabang dari sistem kecerdasan buatan (*artificial intelegent*) yang mengemulasi kemampuan manusia dalam berfikir ke dalam bentuk algoritma yang kemudian dijalankan oleh mesin. *Fuzzy* merupakan istilah yang dipakai oleh Lotfi A Zadeh pada bulan Juli 1964 untuk menyatakan kelompok atau himpunan yang dapat dibedakan dengan himpunan lain berdasarkan derajat keanggotaan dengan batasan yang tidak begitu jelas (samar), tidak seperti himpunan klasik yang membedakan keanggotaan himpunan menjadi dua, himpunan anggota atau bukan anggota.

2.3.1 Konsep Logika *Fuzzy*

Berikut ini adalah konsep dasar dari logika *Fuzzy*.

1. Himpunan *Fuzzy*

   Pada himpunan tegas setiap elemen dalam semestanya selalu ditentukan secara tegas apakah elemen itu merupakan anggota himpunan tersebut atau tidak. Tetapi dalam kenyataanya tidak semua himpunan terdefinisi secara tegas. Oleh karena itu perlu didefinisikan suatu himpunan *Fuzzy* yang bisa menyatakan kejadian tersebut. Himpunan *Fuzzy* memiliki dua atribut, yaitu :

   a. Linguistik

      Linguistik, yaitu penamaan suatu kelompok yang mewakili suatu keadaan atau kondisi tertentu dengan menggunakan bahasa alami, seperti: LAMBAT, SEDANG, CEPAT.

   b. Numeris

      Numeris, yaitu suatu nilai (angka) yang menunjukkan ukuran dari suatu variabel, seperti: 40, 50, 60, dan sebagainya.

   Hal – hal yang perlu diketahui dalam memahami sistem *Fuzzy*.

   a. Variabel *Fuzzy*

      Variabel *Fuzzy* merupakan variabel yang akan dibahas dalam suatu sistem *Fuzzy*, seperti: umur, berat badan, tinggi badan, dan sebagainya.

b. Himpunan *Fuzzy*

Himpunan *Fuzzy* merupakan suatu kelompok yang mewakili suatu keadaan tertentu dalam suatu variabel *Fuzzy*.

c. Semesta Pembicaraan

Semesta pembicaraan adalah keseluruhan nilai yang diperbolehkan untuk dioperasikan dalam suatu variabel *Fuzzy*. Nilai semesta pembicaraan dapat berupa bilangan positif maupun negatif. Sebagai contoh, semesta pembicaraan untuk variabel laju kendaraan adalah [0,160].

d. Domain

Domain himpunan *Fuzzy* adalah keseluruhan nilai yang diperbolehkan untuk dioperasikan dalam suatu himpunan *Fuzzy*. Nilai domain dapat berupa bilangan positif maupun negatif. Sebagai contoh, domain dari himpunan *Fuzzy* kecepatan adalah sebagai berikut:

NORMAL : [0, 40]

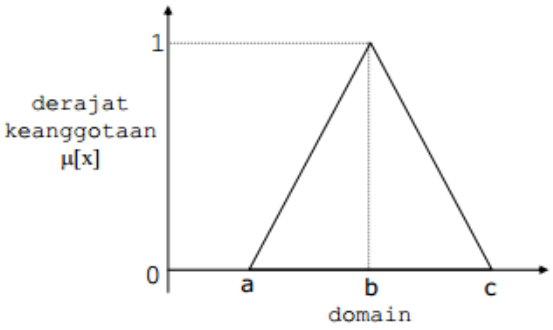
SANGAT TINGGI : [35, 120]

2. Fungsi Keanggotaan

Fungsi keanggotaan adalah suatu kurva yang menunjukkan pemetaan titik input data ke dalam nilai keanggotaannya yang memiliki interval antara 0 sampai 1. Ada beberapa fungsi yang bisa digunakan. diantaranya, yaitu:

a. Representasi Kurva Segitiga

Kurva Segitiga pada dasarnya merupakan gabungan antara dua garis (linear) seperti terlihat pada Gambar 2.3.



Gambar 2. 3 Representasi Kurva Segitiga

Dengan fungsi keanggotaan sebagai berikut :

$$\mu[x] = \begin{cases} 0; x \leq a \; atau \; x \geq c \\ \dfrac{(x-a)}{(b-a)}; \; a \leq x \leq b \\ \dfrac{(b-x)}{(c-b)} \; ; b \leq x \leq c \end{cases} \quad (2.1)$$

Keterangan:

a = nilai domain terkecil yang mempunyai derajat keanggotaan nol

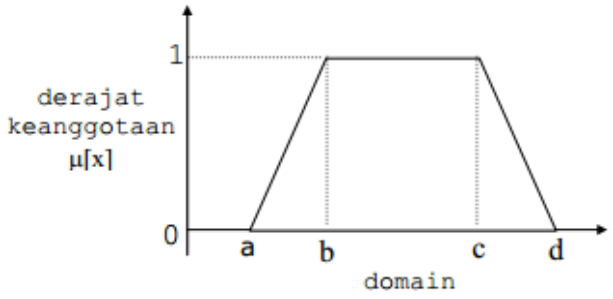
b = nilai domain terkecil yang mempunyai derajat keanggotaan satu

c = nilai domain terbesar yang mempunyai derajat keanggotaan nol

x = nilai input yang akan di ubah ke dalam bilangan *Fuzzy*

b. Representasi Kurva Trapesium

Kurva Trapesium pada dasarnya seperti bentuk segitiga, hanya saja ada beberapa titik yang memiliki nilai keanggotaan 1 untuk lebih jelasnya bisa dilihat pada Gambar 2.4.



Gambar 2. 4 Representasi Kurva Trapesium

Fungsi keanggotaan kurva trapesium :

$$\mu[x] = \begin{cases} 0; & x \leq a \; atau \; x \geq b \\ \dfrac{x-a}{x-b}; & a \leq x \leq b \\ 1 & b \leq x \leq c \\ \dfrac{d-x}{d-c}; & x \geq d \end{cases} \qquad (2.2)$$

Keterangan:

a = nilai domain terkecil yang mempunyai derajat keanggotaan nol

b = nilai domain terkecil yang mempunyai derajat keanggotaan satu

c = nilai domain terbesar yang mempunyai derajat keanggotaan satu

d = nilai domain terbesar yang mempunyai derajat keanggotaan nol

x = nilai input yang akan di ubah ke dalam bilangan *Fuzzy*

3. Operator Dasar *Fuzzy*

Seperti halnya himpunan konvensional, ada beberapa operasi yang didefinisikan secara khusus untuk mengkombinasi dan memodifikasi himpunan *Fuzzy*. Nilai keanggotaan sebagai hasil dari operasi dua himpunan sering dikenal dengan nama *fire strength* atau α–predikat. Ada tiga operator dasar yang diciptakan oleh Zadeh, yaitu:

a. Operator AND

Operator ini berhubungan dengan operasi interseksi pada himpunan. α–predikat sebagai hasil operasi dengan operator AND diperoleh dengan mengambil nilai keanggotaan terkecil antar elemen pada himpunan-himpunan yang bersangkutan.

$$\mu_{A \cap B} = min\,(\mu_A[x], \mu_B[y]) \qquad (2.3)$$

b. Operator NOT

Operasi komplemen pada himpunan *Fuzzy* adalah sebagai hasil operasi dengan operator NOT diperoleh dengan mengurangkan nilai keanggotaan elemen pada himpunan yang bersangkutan.

$$\mu_{A'} = 1 - \mu_A[x] \qquad (2.4)$$

c. Operator OR

Operator ini berhubungan dengan operasi union pada himpunan. α–predikat sebagai hasil operasi dengan operator OR diperoleh dengan mengambil nilai keanggotaan terbesar antar elemen pada himpunanhimpunan yang bersangkutan.

$$\mu_{A\cup B} = \max(\mu_A[x], \mu_B[y]) \quad (2.5)$$

4. Fungsi Implikasi

Tiap – tiap aturan (proposisi) pada basis pengetahuan *Fuzzy* akan berhubungan dengan suatu relasi *Fuzzy*. Bentuk umum dari aturan yang digunakan dalam fungsi implikasi adalah:

```
IF x is A THEN y is B
```

Proposisi yang mengikuti `IF` disebut sebagai anteseden, sedangkan proposisi yang mengikuti `THEN` disebut sebagai konsekuen. Secara umum, ada dua fungsi implikasi yang dapat digunakan, yaitu

a. Min

Pengambilan keputusan dengan fungsi min, yaitu dengan cara mencari nilai minimum berdasarkan aturan ke-i dan dapat dinyatakan dengan:

$$\alpha_i = \mu_{Ai}(x) \cap \mu_{Bi}(x) = \min\{\mu_{Ai}(x), \mu_{Bi}(x)\} \quad (2.6)$$

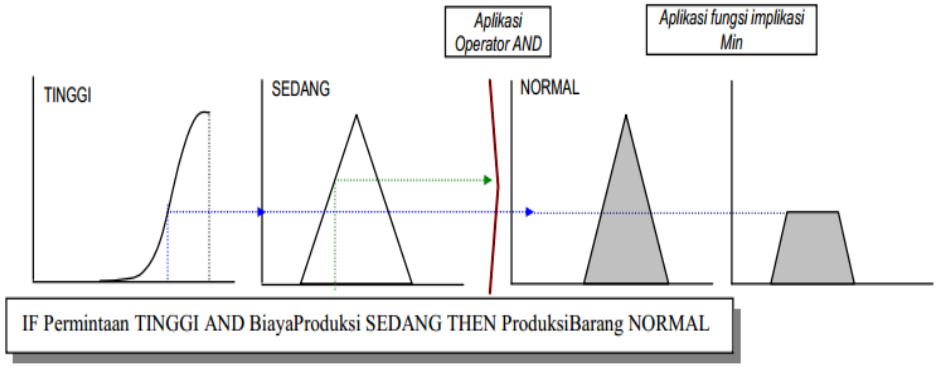
Keterangan:

α_i = nilai minimum dari himpunan *Fuzzy* A dan B pada aturan ke-i

$\mu_{Ai}(x)$ = derajat keanggotaan x dari himpunan *Fuzzy* A pada aturan ke-i

$\mu_{Bi}(x)$ = derajat keanggotaan x dari himpunan *Fuzzy* B pada aturan ke-i

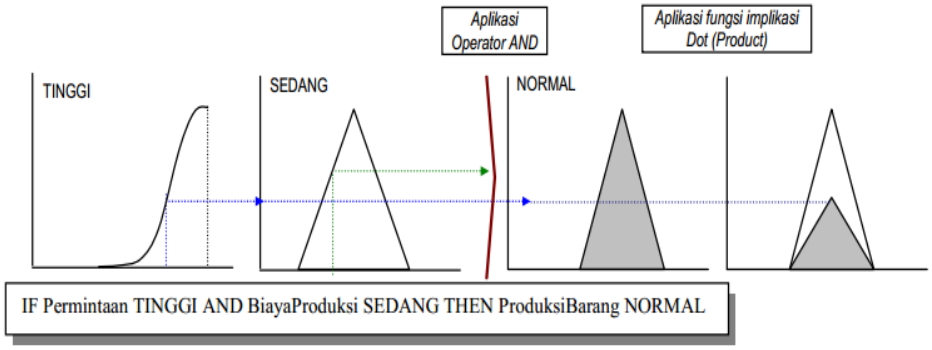
Contoh penggunaan fungsi min untuk kasus produksi barang seperti terlihat pada Gambar 2.5.



Gambar 2. 5 Fungsi Implikasi : MIN

b. Dot

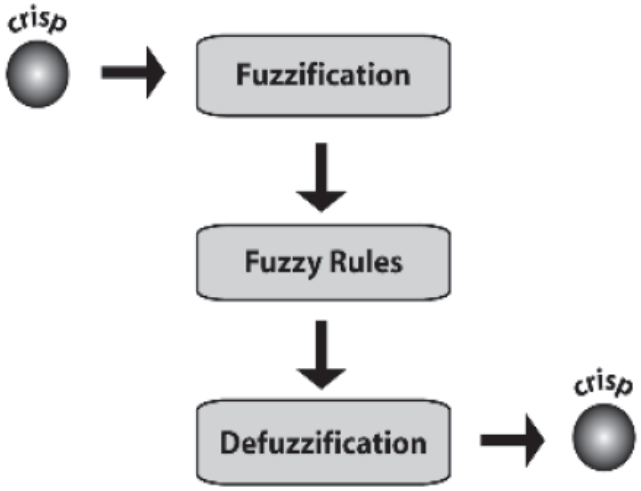
Fungsi ini akan menskala output himpunan *Fuzzy*. Gambar 2.6 menunjukkan salah satu contoh penggunaan fungsi dot.



Gambar 2. 6 Fungsi Implikasi : Dot

2.3.2 Sistem Berbasis Aturan *Fuzzy*

Sistem kontrol logika *Fuzzy* **[5]** diimplementasikan dalam tiga tahapan. Untuk lebih jelasnya bisa dilihat pada Gambar 2.7.



Gambar 2. 7 Sistem Kontrol Logika *Fuzzy*

Berdasarkan Gambar 2.7. sistem kontrol logika *Fuzzy* memiliki tiga proses, yaitu sebagai berikut :

1. *Fuzzification*

   Fuzzification merupakan proses konversi input-input (masukan) yang bersifat tegas (*crips*) kedalam bentuk (*Fuzzy*) variable linguistik menggunakan fungsi keanggotaan.

2. *Fuzzy Rules*

   Fuzzy Rules merupakan proses pengkonversian *input- Fuzzy* menggunakan aturan-aturan "IF-THEN" menjadi *output-Fuzzy*.

3. *Defuzzification*

   Defuzzification merupakan proses konversi *Output-Fuzzy* dari sistem inferensi ke dalam bentuk tegas *(crips*) menggunakan fungsi keanggotaan serupa (sebelumnya) menjadi sebuah nilai.

2.3.3 Sistem Inferensi *Fuzzy*

Untuk mengolah input dan output *Fuzzy* pada penelitian ini menggunakan sistem inferensi Mamdani. Dimana pada setiap aturan yang berbentuk implikasi (sebab-akibat) anteseden yang berbentuk konjungsi (AND) mempunyai nilai

keanggotaan berbentuk minimum (MIN), sedangkan konsekuen gabungannya berbentuk maksimum (MAX). Selain Mamdani, adapula beberapa sistem inferensi *Fuzzy* diantaranya adalah :

1. *Fuzzy* Sugeno

   Diperkenalkan oleh Michio Sugeno dimana output dari sistem inferensi *Fuzzy* ini menerima input berupa nilai *Fuzzy* dan memetakan output dalam bentuk konstanta atau persamaan linear.

2. *Fuzzy* Tsukamoto

   Pada Metode Tsukamoto, setiap konsekuen pada aturan yang berbentuk "IF-THEN" harus direpresentasikan dengan suatu himpunan *Fuzzy* dengan fungsi keanggotaan yang monoton. Sebagai hasilnya, output hasil inferensi dari tiap-tiap aturan diberikan secara tegas (*crisp*) berdasarkan $\alpha -$ predikat ($fire\ strength$). Hasil akhirnya diperoleh dengan menggunakan rata-rata terbobot.

Sedangkan *Fuzzy* Mamdani sendiri, baik input maupun outputnya merupakan nilai *Fuzzy*. Metode ini juga cocok digunakan apabila input diterima dari manusia bukan mesin serta lebih diterima oleh banyak pihak. Untuk lebih jelasnya perbandingan antara metode *Fuzzy* Sugeno, Mamdani dan Tsukamoto bisa dilihat pada Tabel 2.1.

Tabel 2. 1 Perbandingan Metode Tsukamoto, Sugeno dan Mamdani

Metode	**Input**	**Output**	**De*fuzzi*fikasi**	**Penggunaan**
Tsukamoto	Himpunan *Fuzzy*	Himpunan *Fuzzy*	*Weighted Average*	- *Humanis* - *Controll*
Sugeno	Himpunan *Fuzzy*	- Konstanta - Linear (Orde satu)	*Weighted Average*	*Controll*
Mamdani	Himpunan *Fuzzy*	Himpunan *Fuzzy*	- CoG (Centroid) - Lom - Som - Mom - Bisector	*Humanis*

Apabila Tabel 2.1 menjadi acuan. Maka pemilihan sistem inferensi Mamdani apabila dibandingkan dengan metode inferensi *Fuzzy* yang lain dapat dilihat sebagai berikut :

1. Jika menggunakan sistem inferensi *Fuzzy* Tsukamoto, maka himpunan *Fuzzy* yang dilibatkan hanya terbatas pada fungsi keanggotaan yang monoton saja.
2. Jika menggunakan sistem inferensi *Fuzzy* sugeno akan sulit untuk menentukan target tembak pada NPC *Tower* karena sulit untuk memodelkannya dalam bentuk konstanta atau persamaan linear.
3. Dengan menggunakan sistem *Fuzzy* Mamdani akan lebih mudah untuk membandingkan hasil yang didapat karena input dan output yang didapat merupakan nilai *Fuzzy*.

Setelah mengetahui metode inferensi atau penalaran *Fuzzy* yang akan digunakan (Mamdani), maka proses selanjutnya adalah mendapatkan nilai output. Untuk mendapatkan nilai output pada metode Mamdani diperlukan empat tahapan yaitu:

1. Pembentukan Himpunan *Fuzzy*

   Pada metode Mamdani, baik variabel input maupun variabel output dibagi menjadi satu atau lebih himpunan *Fuzzy*.
2. Aplikasi Fungsi Implikasi

   Pada Metode Mamdani, fungsi implikasi yang digunakan adalah Min. yang berarti tingkat keanggotaan yang didapat sebagai konsekuen dari proses ini adalah nilai minimum dari himpunan *Fuzzy*.
3. Komposisi Aturan

   Apabila sistem terdiri dari beberapa aturan, maka inferensi diperoleh dari gabungan antar aturan. Ada tiga metode yang digunakan dalam melakukan inferensi sistem *Fuzzy*, yaitu: max, additive dan probabilistik OR (probor).

Pada penelitian ini menggunakan metode max. Dimana solusi himpunan *Fuzzy* diperoleh dengan cara mengambil nilai maksimum aturan, kemudian menggunakannya untuk memodifikasi daerah *Fuzzy*, dan mengaplikasikannya ke output dengan menggunakan operator OR (*union*). Jika semua proposisi telah dievaluasi, maka output akan berisi suatu himpunan *Fuzzy* yang merefleksikan konstribusi dari tiap-tiap proposisi. Secara umum dapat dituliskan:

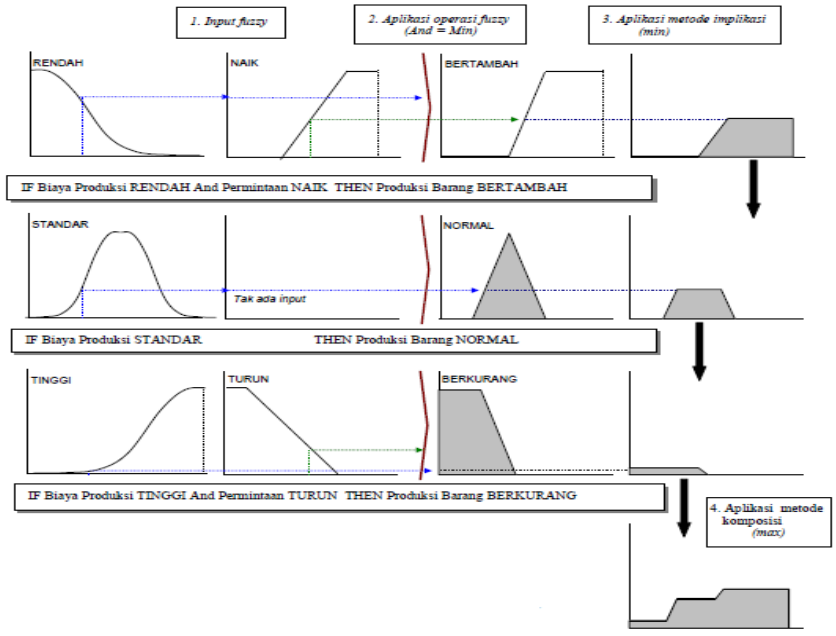
$$U_{sf}[x_i] = \max(\mu_{sf}[x_i], \mu_{kf}[x_i]) \qquad (2.7)$$

Keterangan:

$\mu_{sf}[x_i]$ = nilai keanggotaan solusi *Fuzzy* sampai aturan ke-i;

$\mu_{kf}[x_i]$ = nilai keanggotaan konsekuen *Fuzzy* aturan ke-i.

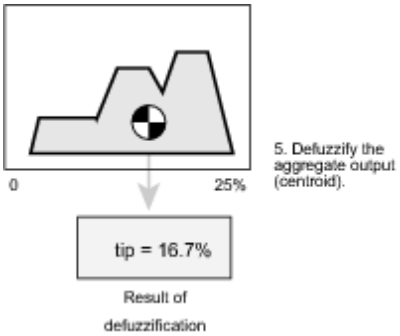
Proses inferensi dengan menggunakan metode Max dalam melakukan komposisi aturan seperti terlihat pada Gambar 2.8. Apabila digunakan fungsi implikasi Min, maka metode komposisi ini sering disebut dengan nama Max-Min atau Min-Max atau Mamdani.



Gambar 2. 8 Komposisi Aturan Fuzzy: Metode Max

4. Penegasan (*defuzzi*fikasi)

Input dari proses *defuzzi*fikasi adalah suatu himpunan *Fuzzy* yang diperoleh dari suatu komposisi aturan – aturan *Fuzzy*, sedangkan output yang dihasilkan merupakan suatu bilangan pada himpunan *Fuzzy* tersebut. Dalam peneltian ini metode *deFuzzy*fikasi yang digunakan menggunakan metode Centroid. Pada metode Centroid solusi crisp diperoleh dengan cara mengambil titik pusat daerah *Fuzzy*. Bisa dilihat pada Gambar 2.9.



Gambar 2. 9 Deffuzifikasi : Metode Centroid

Secara umum dirumuskan :

$$Z_0 = \frac{\int_a^b Z.\mu(z)dz}{\int_a^b \mu(z)dz} \qquad (2.8)$$

Keterangan :

Z = nilai domain ke-i,
$\mu_{(z)}$ = derajat keanggotaan titik tersebut,
Z_0 = nilai hasil penegasan (*defuzzi*fikasi).

Meskipun Ada beberapa metode defuzzifikasi sistem inferensi Mamdani seperti metode Bisektor, metode *Mean of Maximum* (Mom), metode *Largest of Maximum* (Lom) dan metode *Smallest of Maximum* (Som). Namun metode *Centroid* dianggap paling mudah dari segi komputasi serta nilai deffuzzifikasi yang dihasilkan akan bergerak secara halus sehingga perubahan dari suatu himpunan *Fuzzy* juga akan berjalan secara halus atau lebih mendekati nilai akhir yang sebenarnya.

2.4 *Unified Modeling Language* (UML)

Unified Modeling Language (UML) adalah bahasa pemodelan untuk sistem atau perangkat lunak yang berparadigma berorientasi objek. Pemodelan (*modeling*) sesungguhnya digunakan untuk penyederhanaan permasalahan-permasalahan yang kompleks sedemikian rupa sehingga lebih mudah dipelajari dan dipahami.

UML tidak hanya merupakan sebuah bahasa pemograman *visual* saja, namun juga dapat secara langsung dihubungkan ke berbagai bahasa pemograman, seperti *JAVA*, *C++*, *Visual Basic*, atau bahkan dihubungkan secara langsung ke dalam sebuah *object-oriented database*. Begitu juga mengenai pendokumentasian dapat dilakukan seperti; *requirements*, *arsitektur*, *design*, *source code*, *project plan*, *tests*, dan *prototypes*.

UML sendiri terdiri atas pengelompokkan *Diagram-Diagram* sistem menurut aspek atau sudut pandang tertentu. *Diagram* adalah yang menggambarkan permasalahan maupun solusi dari permasalahan suatu model. UML bukan hanya sekedar *Diagram*, tetapi juga menceritakan konteksnya. Berikut adalah contoh beberapa UML yang digunakan untuk memodelkan aplikasi berorientasi objek :

Tabel 2. 2 *Diagram* UML

Diagram	**Deskripsi**
Use Case Diagram	Menggambarkan interaksi antara sistem dan pengguna.
Activity Diagram	Menggambarkan alur aktivitas sekuensial dari tiap *use case*, proses bisnis dan logika sistem.
Sequence Diagram	Menggambarkan secara grafis bagaimana objek-objek berinteraksi satu sama lain
Class Diagram	Menggambarkan struktur objek dari sistem, yang memuat objek-objek yang terdapat dalam sistem beserta hubungan antar objek.

2.5 *Tools* Yang Digunakan

Berikut ini adalah beberapa *tools* yang digunakan pada penelitian ini.

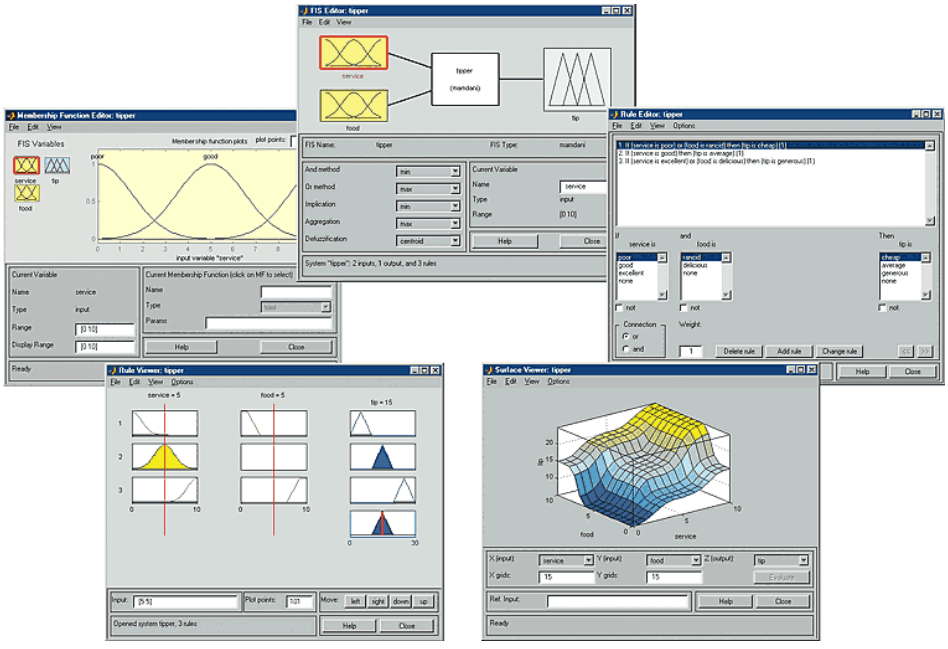
2.5.1 *Action Script*

ActionScript merupakan bahasa pemrograman berorientasi objek (OOP) yang spesifikasi bahasanya bebasis ECMA *Script* dan ditunjukan untuk berjalan di atas

sebuah *runtime environment* (serperti *Flash Player*, *Adobe AIR* atau *Flash Lite*. *Actionscript*) merupakan sebuah program yang mampu berinteraksi dengan bahsa pemrograman lain seperti Javascript, PHP, MySQL, karena pemrograman *actionscript* mempunyai sifat *case sensitive* .

2.5.2 *Fuzzy Logic Toolbox*

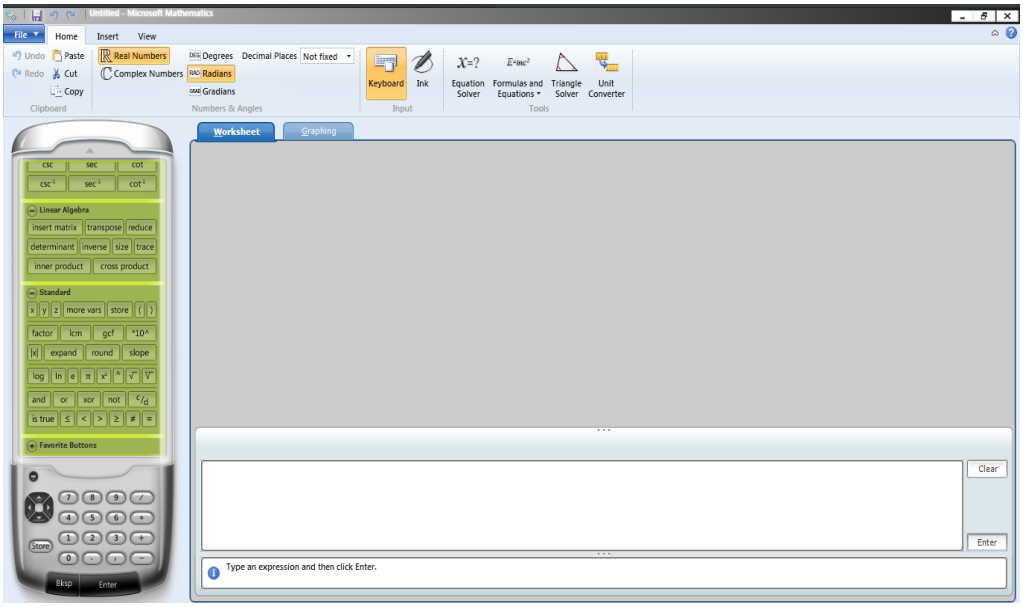
Fuzzy Logic Toolbox adalah fasilitas yang tersedia dalam program matlab untuk membangun suatu sistem logika *Fuzzy*. *Fuzzy Logic Toolbox* memberikan fasilitas *Graphical User Interface* (GUI) untuk mempermudah dalam membangun suatu sistem *Fuzzy*. pada saat membangun sistem *Fuzzy* dengan *Fuzzy Logic Toolbox*, maka pengguna harus melakukan editing yang meliputi : jumlah masukan atau keluaran, klasifikasi masukan atau keluaran, jenis MF (membership Function) masukan atau keluaran, *rule*, metode defuzzyfikasi. Dengan memanfaatkan tiga fasilitas GUI seperti, Fis Editor, MF Editor dan *Rule* Editor pengguna dapat membaca dan memodifikasi data tentang sistem *Fuzzy* yang dibangun. Untuk lebih jelasnya bisa dilihat pada Gambar 2.10.



Gambar 2. 10 *Fuzzy Logic Toolbox* Matlab

2.5.3 Microsoft *Mathematics*

Microsoft *Mathematics* adalah perangkat lunak sejenis kalkulator namun memiliki fitur yang lebih lengkap dan memiliki kemampuan untuk menjabarkan secara detail langkah demi langkah penyelesaian suatu persoalan, tidak hanya matematika namun untuk ilmu fisika dan kimia. Untuk lebih jelasnya dapat dilihat pada Gambar 2.11.



Gambar 2. 11 Microsoft *Mathematics*