BAB 3
Metodologi Penelitian

3.1 Pendekatan Penelitian

Penelitian ini menggunakan pendekatan kuantitatif eksperimental yang bertujuan untuk menguji hubungan sebab-akibat antara variabel dengan cara memanipulasi satu atau lebih variabel independen dan mengamati pengaruhnya terhadap variabel dependen, sambil mengontrol variabel lain yang relevan, dengan fokus pada perancangan dan pengujian sistem kecerdasan buatan berbasis ANFIS dalam permainan RTS. Tujuan dari pendekatan ini adalah untuk mengukur efektivitas dan efisiensi sistem ANFIS dalam membantu pengambilan keputusan AI saat bermain melawan pemain dalam game.

Melalui eksperimen berbasis *gameplay* nyata, data yang dikumpulkan untuk mengukur akurasi pengambilan keputusan, kecepatan respons sistem serta adaptivitas strategi terhadap kondisi permainan yang berubah-ubah.

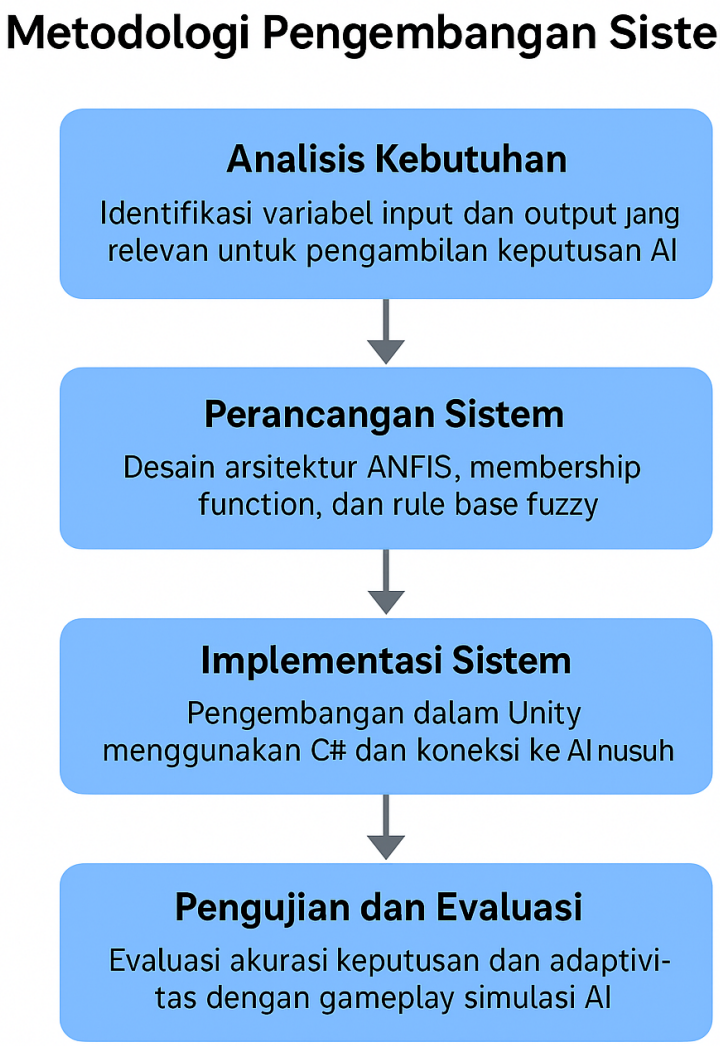
3.2 Metodologi Pengembangan Sistem

Pengembangan sistem dalam penelitian ini menggunakan metode *waterfall* karena proses pembuatan dilakukan secara bertahap dan terstruktur. Setiap tahapan harus diselesaikan terlebih dahulu sebelum melanjutkan ke tahapan berikutnya. Berikut ini adalah tahapan pengembangan yang dilakukan:

1. **Analisis Kebutuhan:** Menentukan kebutuhan sistem dalam konteks game seperti variabel masukan yang dibutuhkan AI untuk mengambil keputusan. *Input* ANFIS yang diidentifikasi meliputi: *Enemy Range, Energy Amount, Health Status, Ally Position* dan *Attack Priority*.
2. **Perancangan Sistem:** Mendesain struktur logika *fuzzy*, *rule base* dan MF untuk tiap variabel input. Termasuk dalam tahap ini adalah perancangan alur kerja AI, integrasi dengan kartu pasukan musuh, serta pengaturan skenario pertarungan dalam game.
3. **Implementasi Sistem:** Mengimplementasikan sistem ANFIS dalam lingkungan Unity, menggunakan bahasa pemrograman C# untuk AI dalam game dan Python untuk perhitungan awal dan pelatihan model (jika

diperlukan). Sistem dibuat agar AI musuh dapat merespons kondisi permainan secara adaptif.

4. **Pengujian dan Evaluasi:** Sistem diuji dengan menjalankan game dalam berbagai skenario permainan. Evaluasi dilakukan dengan mengukur akurasi keputusan AI, waktu respons dalam pengambilan tindakan, serta efektivitas strategi AI melawan pemain.



Gambar 3.1. Diagram alir sistem ANFIS dalam game RTS

3.3 Platform Game dan Integrasi dengan ANFIS

3.3.1 Platform Game

Game ini dikembangkan menggunakan **Unity Engine versi 2021**, salah satu platform pengembangan game yang populer berkat fleksibilitasnya, dukungan *multiplatform*, dan kemampuan untuk mengintegrasikan kecerdasan buatan. Game ini termasuk dalam *genre* RTS yang terinspirasi dari *Clash Royale*, di mana pemain dapat menurunkan pasukan, membangun *tower*, dan menyerang pertahanan musuh secara *real-time*.

Permainan ini dijalankan pada:

- **PC (Windows)** sebagai target utama pengujian.
- Potensi pengembangan ke **Android** karena dukungan *cross-platform* dari Unity.

3.3.2 Struktur Game

Game dibagi ke dalam beberapa fase utama:

- ***Main Menu***: Pemain memilih untuk memulai permainan atau keluar.
- ***Building Phase***: Pemain dapat menempatkan menara untuk bertahan.
- ***Attack Phase***: Pasukan AI bergerak dan menyerang berdasarkan strategi dari sistem ANFIS.
- ***Victory/Lose Condition***: Evaluasi apakah *tower* utama hancur.

3.3.3 Integrasi dengan ANFIS Engine

Kecerdasan buatan dalam game ini mengimplementasikan pendekatan **ANFIS** untuk menentukan keputusan pasukan AI, khususnya dalam menentukan:

- Jarak serang optimal,
- Strategi menyerang atau bertahan,
- Prioritas serangan berdasarkan konteks permainan.

Sistem ANFIS mempertimbangkan lima variabel utama:

1. ***Enemy Range***
2. ***Energy (Gold)***
3. ***Health***
4. ***Ally Position***
5. ***Attakc Priority***

A Cara Integrasi ANFIS dengan Unity

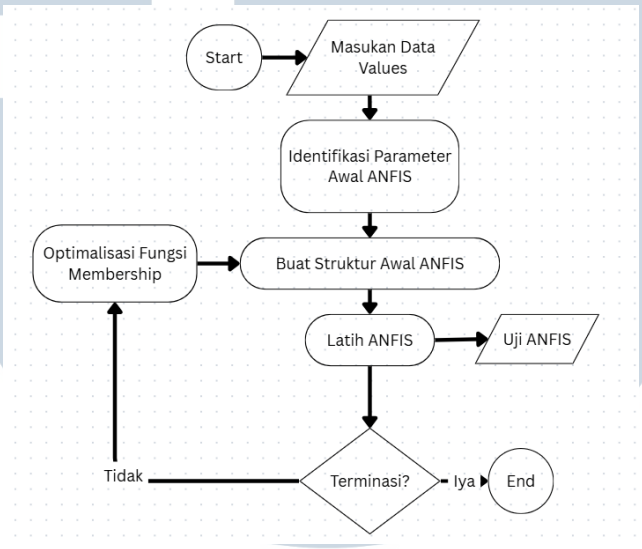
1. **Implementasi Fungsi ANFIS di Unity (C#)**
   Fungsi-fungsi ANFIS seperti kombinasi bobot dan konsekuen linier diimplementasikan langsung dalam skrip Unity. Karena tidak dilakukan pelatihan model, bobot ditetapkan secara manual dan diuji melalui eksperimen.
2. **Modul AI**
   Skrip seperti `ANFISAITrainedTroopManager.cs` membaca status pasukan dan mengolahnya menggunakan fungsi ANFIS untuk menentukan aksi unit secara *real-time*.
3. **Logging dan Evaluasi**
   Keputusan AI dicatat menggunakan modul `AIDecisionLogger.cs` ke dalam file log, memudahkan proses evaluasi performa AI.
4. **Optimasi dan Perbandingan**
   Sistem dapat dikembangkan untuk mengintegrasikan algoritma optimasi seperti *Genetic Algorithm* agar bobot dan parameter ANFIS lebih adaptif.

3.3.4 Manfaat Integrasi ANFIS di Unity

- **Efisiensi *Real-Time***: Perhitungan ANFIS secara langsung memberikan respon cepat dalam gameplay.
- **Keputusan Adaptif**: AI mempertimbangkan berbagai variabel sekaligus dalam pengambilan keputusan.

- **Mudah Dikembangkan**: Sistem dapat diperluas dengan menambahkan variabel, aturan *fuzzy* baru atau metode AI lainnya seperti *Reinforcement Learning*.

3.4 Desain Sistem ANFIS



Gambar 3.2. Flowchart ANFIS

Gambar 3.2 menunjukkan alur tahapan pengembangan sistem ANFIS yang digunakan dalam penelitian ini. Proses ini terdiri dari beberapa tahap berurutan, mulai dari input data pelatihan hingga validasi hasil pelatihan ANFIS. Berikut adalah penjelasan setiap tahap:

1. **Mulai:** Tahap awal yang menandai dimulainya proses pengembangan sistem ANFIS.
2. **Masukkan Data Latih:** Sistem menerima data pelatihan berupa kombinasi nilai dari lima variabel input utama, yaitu *Enemy Range*, *Energy*, *Health*, *Ally Position*, dan *Priority*.
3. **Identifikasi Parameter Awal ANFIS:** Parameter awal ANFIS diidentifikasi, termasuk jumlah himpunan *fuzzy*, jenis fungsi *membership* dan parameter awal dari *rule base*.

4. **Bangun Struktur Awal ANFIS:** Struktur awal jaringan ANFIS dibentuk berdasarkan parameter yang telah ditentukan. Ini mencakup layer-layer ANFIS dan hubungan antar input dan output.
5. **Latih ANFIS:** Proses pelatihan dilakukan menggunakan metode *hybrid learning* untuk menyesuaikan parameter fungsi *membership* dan fungsi konsekuen, dengan tujuan meminimalkan *error*.
6. **Uji ANFIS:** Setelah pelatihan, model diuji menggunakan data uji untuk menilai akurasi, efektivitas dan kemampuan adaptasi sistem dalam situasi di simulasi permainan.
7. **Terminasi?:** Jika hasil pengujian belum memenuhi kriteria, proses dilanjutkan ke tahap optimasi. Jika sudah memenuhi, maka proses berakhir.
8. **Optimalkan Parameter *Membership*:** Sistem akan mengoptimasi parameter *fuzzy* untuk memperbaiki performa inferensi ANFIS.
9. **Selesai:** Proses dihentikan setelah model ANFIS mencapai performa optimal.

3.5 Rumus ANFIS yang diimplementasikan

Berikut adalah bentuk sederhana dari rumus ANFIS yang diimplementasikan pada Unity:

$$f = 0.3x_1 + 0.1x_2 + 0.3x_3 + 0.2x_4 + 0.1x_5 \tag{3.1}$$

- x_1: *Enemy Range*
- x_2: *Energy*
- x_3: *Health*
- x_4: *Ally Position*
- x_5: *Attack Priority*

$$y = 8f + 2 \tag{3.2}$$

- y: *Action Decision*

Nilai y ini merepresentasikan batas jarak untuk AI mengambil keputusan menyerang. Jika musuh berada dalam jarak $\leq y$, maka AI akan menyerang. Jika tidak, pasukan akan mendekat terlebih dahulu.

Bobot pada fungsi linier ditentukan berdasarkan pengetahuan domain dan eksperimen dalam game, dengan pertimbangan sebagai berikut:

- ***Enemy Range*** (x_1): Bobot 0.3 — variabel utama yang menentukan apakah AI perlu mendekat atau sudah dalam jangkauan serang.
- ***Energy*** (x_2): Bobot 0.1 — energi dibutuhkan untuk menurunkan pasukan, tapi tidak dominan dalam keputusan penyerangan.
- ***Health*** (x_3): Bobot 0.3 — semakin tinggi HP, AI akan lebih berani menyerang karena kemungkinan bertahan hidup lebih besar.
- ***Ally Position*** (x_4): Bobot 0.2 — posisi pasukan terhadap musuh juga penting, meskipun bersifat kontekstual.
- ***Attack Priority*** (x_5): Bobot 0.1 — prioritas bertindak sebagai penyesuaian minor dalam inferensi.

Pendekatan ini tidak menggunakan proses pelatihan seperti pada ANFIS klasik, sehingga seluruh bobot bersifat statis. Keputusan diambil berdasarkan kombinasi nilai-nilai *input* yang telah dinormalisasi ke dalam rentang [0, 1], lalu dihitung menggunakan fungsi linier. Pendekatan ini memberikan efisiensi tinggi dalam gameplay *real-time* karena tidak memerlukan pembelajaran berat.

Misalkan diberikan nilai *input*:

$$x_1 = 0.5, \quad x_2 = 0.6, \quad x_3 = 0.8, \quad x_4 = 0.3, \quad x_5 = 0.2$$

Maka skor *decision* AI adalah:

$$f = 0.3(0.5) + 0.1(0.6) + 0.3(0.8) + 0.2(0.3) + 0.1(0.2) = 0.15 + 0.06 + 0.24 + 0.06 + 0.02 = 0.53$$

Sehingga jarak keputusan akhir:

$$y = 8(0.53) + 2 = 6.24$$

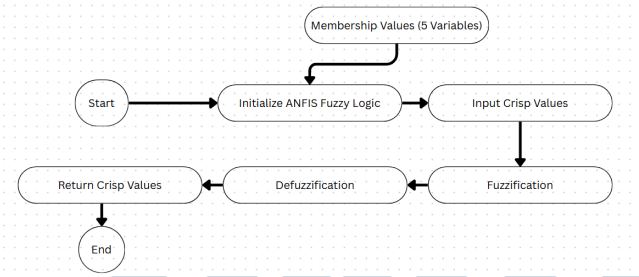
Artinya, jika musuh berada dalam jarak ≤ 6.24 unit, AI akan segera menyerang. Jika tidak, AI akan bergerak mendekat terlebih dahulu.

Formula ini digunakan oleh AI untuk menghitung skor keinginan menyerang berdasarkan 5 variabel input, yang kemudian dikonversi menjadi jarak serang (y) — misalnya: apakah pasukan harus mendekat dulu, atau sudah cukup dekat untuk mulai menyerang

Dalam implementasi skripsi ini, bobotnya statis artinya ditentukan di awal dan tidak diubah secara otomatis melalui pelatihan (*training*). Pendekatan ini dipilih karena:

- Tidak menggunakan *backpropagation* atau *hybrid learning* dari ANFIS asli.
- Fokus pada efisiensi di Unity secara *real-time*.
- Bobot ditentukan secara praktis berdasarkan logika domain game dan hasil eksperimen manual.

3.6 Alur ANFIS



Gambar 3.3. Alur ANFIS

Proses dimulai dari tahap inisialisasi seluruh komponen fuzzy untuk setiap variabel input yang dibutuhkan seperti *Enemy Range*, *Energy Amount*, *Health Status*, *Ally Position* dan *Attack Priority*. Setelah inisialisasi, sistem menerima data input berupa *crisp values* contohnya jarak antara titik *pawn* dan pasukan lawan, serta jumlah energi yang tersedia. Nilai-nilai crisp ini kemudian melalui proses *fuzzification* sehingga dapat direpresentasikan sebagai derajat *membership* dalam berbagai kategori.

Selanjutnya, sistem melakukan inferensi *fuzzy* berdasarkan aturan-aturan yang telah ditetapkan, menghasilkan output berupa nilai *fuzzy* yang kemudian dikonversi kembali menjadi *crisp value* melalui proses defuzzification. Hasil

akhir dari proses defuzzification ini berupa nilai *desirability*, yaitu skor yang digunakan untuk menentukan titik atau aksi terbaik yang diambil AI. Keputusan yang dihasilkan kemudian dijalankan dalam permainan seperti memilih lokasi spawn pasukan musuh yang paling optimal berdasarkan perhitungan *desirability*. Proses ini berjalan secara terus menerus, sehingga keputusan AI selalu responsif terhadap kondisi permainan terbaru.

3.7 Desain Game dan Sistem ANFIS

Game yang dikembangkan merupakan RTS bergaya mirip *Clash Royale*, di mana dua pemain—yaitu pemain utama dan AI musuh—memiliki dek kartu pasukan yang dapat dipanggil ke medan pertempuran secara bergantian. Setiap pasukan memiliki atribut dan strategi tertentu, seperti serangan jarak jauh, kecepatan, dan daya tahan.

Sistem ANFIS diterapkan pada AI musuh untuk menentukan keputusan strategis secara otomatis berdasarkan kondisi permainan yang berlangsung. Keputusan ini mencakup pemilihan jenis kartu pasukan serta waktu terbaik untuk melakukan pemanggilan (*spawn*). ANFIS dipilih karena kemampuannya dalam menggabungkan logika *fuzzy* dan pembelajaran adaptif dari jaringan saraf sehingga AI dapat menyesuaikan strategi berdasarkan konteks medan tempur.

- **Input ANFIS:**
  - ***Enemy Range*** — Jarak antara unit musuh dengan base atau unit sekutu.
  - ***Energy Amount*** — Jumlah energi yang tersedia untuk memanggil pasukan.
  - ***Health*** — Status kesehatan dari pasukan AI yang sedang aktif.
  - ***Ally Position*** — Posisi pasukan sekutu yang berada di sekitar area musuh.
  - ***Attack Priority*** — Tingkat urgensi untuk melakukan serangan berdasarkan jenis musuh yang muncul.
- ***Output* ANFIS:**
  - ***Action Decision*:** Keputusan untuk melakukan *Attack* yang kemudian digunakan untuk memilih kartu dan waktu pemanggilan.

3.8 Proses Pelatihan dan Pengujian ANFIS

Proses pelatihan dilakukan dengan data simulasi yang dihasilkan dari skenario simulasi yang divariasikan. Dataset dibagi menjadi dua bagian:

- **Data Latih:** 70% dari total data digunakan untuk melatih sistem ANFIS.
- **Data Uji:** 30% sisanya digunakan untuk menguji generalisasi sistem.

3.9 Evaluasi Kinerja AI

AI yang menggunakan ANFIS dibandingkan dengan AI berbasis aturan biasa. Aspek yang diuji antara lain:

- Ketepatan keputusan dalam berbagai situasi permainan
- Waktu respons dalam skenario *real-time*
- Persentase kemenangan

3.9.1 Perangkat Lunak

- **Game Engine:** Unity 2021 (untuk pengembangan dan simulasi game *real-time strategy*)
- **Bahasa Pemrograman:** C# (untuk scripting dalam Unity)
- **AI *Modeling*:** Sistem ANFIS ditanamkan langsung ke dalam game dan dilatih menggunakan data dari simulasi permainan Player vs AI. AI belajar dari pengalaman bermain dengan memperbarui parameter berdasarkan kondisi simulasi *versus* secara *real-time*.
- **Library Tambahan:** *Fuzzy Logic Toolbox*
- **Lingkungan Pengembangan:** *Visual Studio* 2022 dan Unity *Editor*.

3.9.2 Perangkat Keras

- **Laptop:** ASUS TUF Gaming F15
- **Prosesor:** Intel Core i5

- **RAM:** 16 GB DDR4
- **Penyimpanan:** SSD 512 GB
- **Kartu Grafis:** NVIDIA GeForce GTX 1650 4GB
- **Sistem Operasi:** Windows 11 Home 64-bit