**BAB III**

**ANALISIS DAN PERANCANGAN**

Pengembangan *bot* pada permainan ini menggunakan metode *waterfall*, yang memiliki tahap analisis, perancangan, dan pengkodean. Pada bab ini akan dijelaskan secara rinci tahap analisis dan perancangan.

**3.1 Analisis**

Pada subbab analisis akan dijelaskan 2 (dua) tahapan analisis, yaitu analisis proses dan analisis kebutuhan. Berikut penjelasan masing-masing analisis.

**3.1.1 Analisis Proses**

Analisis proses menjelaskan tentang proses pembelajaran *bot* menggunakan *Reinforcement Learning* (RL). Pada penelitian ini, dipilih menggunakan pendekatan algoritma SARSA, proses pembelajaran akan dijelaskan secara bertahap dimulai dengan merekam cara gerak *bot* dan perhitungan reward untuk pergerakan, mengolah RL, membaca data gerakan pada RL *memory* dan merekam cara gerak *bot*, merekam cara *bot* menembak dan perhitungan reward untuk menembak serta membaca rekaman cara *bot* menembak.

**3.1.1.1 Merekam Cara Gerak *Bot* dan Perhitungan *Reward* untuk Pergerakan**

*Bot* akan diberikan instruksi untuk bergerak dengan berjalan ke kiri (L) , kanan (R), melompat (J) dan berhenti (S). Semua koordinat posisi pergerakan dari *bot* (*State*) dan langkah-langkah yang diambil oleh *bot* (*Action*) untuk mencapai koordinat tertentu (*State'*) akan direkam ke RL *memory*. Diagram rekam dan penyimpanan data dapat dilihat pada Gambar 3.1.

****

**Gambar 3.1 Diagram rekam dan penyimpanan data serta Diagram Membaca Data Gerakan pada RL *Memory* dan Merekam Cara Gerak *Bot***

Agar dapat memahami proses kerja rekam dan penyimpanan data, diberikan contoh kasus berikut. Pada tahap pertama, jalankan *bot* untuk mengeksplorasi lingkungan dalam batas waktu tertentu, setelah itu semua titik koordinat yang dicapai oleh *bot* akan disimpan dalam tabel, setelah itu akan dilakukan perhitungan reward untuk mengetahui nilai yang didapat oleh *bot* untuk setiap pergerakan yang dilakukan. Misalkan diambil pergerakan *bot* sebesar 30 frame pada detik pertama, hasilnya dapat dilihat pada Tabel 3.1.

**Tabel 3.1 Tabel GerakRL**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Nama | Class | Pos\_x | Pos\_y | Movement | Previous Reward | Reward | Total Reward |
| My Bot 1 | 6 | 3348 | 564 | L | 0 | 0.59 | 2391.46 |
| My Bot 1 | 6 | 3347.41 | 564.6 | L | 0.59 | 1.11 | 2390.87 |
| My Bot 1 | 6 | 3346.3 | 565.8 | L | 1.11 | 1.55 | 2389.76 |
| My Bot 1 | 6 | 3344.75 | 567.6 | L | 1.55 | 1.94 | 2388.21 |
| My Bot 1 | 6 | 3342.81 | 570 | L | 1.94 | 2.28 | 2386.27 |
| My Bot 1 | 6 | 3340.53 | 573 | L | 2.28 | 2.57 | 2383.99 |
| My Bot 1 | 6 | 3337.96 | 576.6 | L | 2.57 | 2.83 | 2381.42 |
| My Bot 1 | 6 | 3335.13 | 580.8 | L | 2.83 | 3.07 | 2378.59 |
| My Bot 1 | 6 | 3332.05 | 582.49 | L | 3.07 | 3.24 | 2375.52 |
| My Bot 1 | 6 | 3328.81 | 582.49 | L | 3.24 | 3.41 | 2372.28 |
| My Bot 1 | 6 | 3325.4 | 582.49 | L | 3.41 | 3.56 | 2368.87 |
| My Bot 1 | 6 | 3321.84 | 582.49 | L | 3.56 | 3.69 | 2365.31 |
| My Bot 1 | 6 | 3318.15 | 582.49 | L | 3.69 | 3.8 | 2361.62 |
| My Bot 1 | 6 | 3314.36 | 582.49 | L | 3.8 | 3.89 | 2357.82 |
| My Bot 1 | 6 | 3310.46 | 582.49 | L | 3.89 | 3.98 | 2353.93 |
| My Bot 1 | 6 | 3306.49 | 582.49 | L | 3.98 | 4.05 | 2349.95 |
| My Bot 1 | 6 | 3302.44 | 582.49 | L | 4.05 | 4.11 | 2345.9 |
| My Bot 1 | 6 | 3298.33 | 582.49 | L | 4.11 | 4.17 | 2341.79 |
| My Bot 1 | 6 | 3294.16 | 582.49 | L | 4.17 | 4.21 | 2337.62 |
| My Bot 1 | 6 | 3289.94 | 582.49 | L | 4.21 | 4.26 | 2333.41 |
| My Bot 1 | 6 | 3285.69 | 582.49 | L | 4.26 | 4.29 | 2329.15 |
| My Bot 1 | 6 | 3281.39 | 582.49 | L | 4.29 | 4.32 | 2324.86 |
| My Bot 1 | 6 | 3277.07 | 582.49 | L | 4.32 | 4.35 | 2320.54 |
| My Bot 1 | 6 | 3272.72 | 582.49 | L | 4.35 | 4.37 | 2316.19 |
| My Bot 1 | 6 | 3268.34 | 582.49 | L | 4.37 | 4.4 | 2311.82 |
| My Bot 1 | 6 | 3263.95 | 582.49 | L | 4.4 | 4.41 | 2307.42 |
| My Bot 1 | 6 | 3259.54 | 582.49 | L | 4.41 | 4.43 | 2303.01 |
| My Bot 1 | 6 | 3255.11 | 582.49 | L | 4.43 | 4.44 | 2298.58 |
| My Bot 1 | 6 | 3250.66 | 582.49 | L | 4.44 | 4.45 | 2294.14 |
| My Bot 1 | 6 | 3246.21 | 582.49 | L | 4.45 | 4.46 | 2289.69 |

Keterangan Isi Tabel :

-Nama : menunjukkan nama *bot* dimana terdiri dari 3 bot.

-*Class* : Jenis karakter yang dipakai oleh *bot*. Ada 9 (sembilan) jenis karakter yaitu : *sniper, spy, engineer, medic, demoman, heavy, soldier, pyro,* dan *scout.* Pada permainan ini digunakan 3 (tiga) karakter saja yaitu : *soldier, heavy,* dan *scout*.

-Pos\_x : koordinat posisi bot berdasarkan titik horizontal dengan satuan pixel.

-Pos\_y : koordinat posisi bot berdasarkan titik vertikal dengan satuan pixel.

-*Movement* : Langkah yang dipakai oleh bot. Terdapat *Left* (L), *Right* (R), *Jump* (J), dan *Stop* (S).

-*Previous Reward* : Hasil / Poin yang akan diterima oleh *bot* berdasarkan aksi yang dilakukan sebelumnya.

-*Reward* : Hasil / Poin yang akan diterima oleh *bot* berdasarkan aksi yang dilakukan.

Reward­i­ = pos\_x­i - pos\_xi-1

-*Total Reward* : Total Reward akhir yang akan diterima oleh bot berdasarkan langkah yang telah dilakukan.

**3.1.1.2 Mengolah RL**

RL *Memory* dinormalisasi agar pembacaan data akan lebih efisien. Pada penormalisasi gerakan RL *Memory* dapat dilakukan dengan cara membatasi nilai ambang dari *state*(S/posisi),dan *action* yang bebeda(A) dengan total *reward* masing masing yang *maximum (*Rt) serta membuang semua *state* dan *action* yang dilakukan dengan *reward* <= 0. Dalam SQL ( **CREATE VIEW RLGerakbot AS SELECT RL2.nama, printf('%d', RL2.pos\_x) as pos\_x, RL2.movement, RL2.totalreward FROM (SELECT round(pos\_x) as pos\_x, movement, max(totalreward) as totalreward FROM GerakRL where reward>0 group by nama, round(pos\_x), movement) RL1 JOIN (SELECT round(pos\_x) as pos\_x, movement, nama, totalreward FROM GerakRL where reward>0 group by round(pos\_x), movement, nama, totalreward) RL2 ON RL1.pos\_x = RL2.pos\_x where RL1.totalreward =RL2.totalreward and RL1.movement=RL2.movement**).

**3.1.1.3 Membaca Data Gerakan pada RL *Memory* dan Merekam Cara Gerak *Bot***

Data Gerakan pada RL *Memory* yang sudah dinormalisasi akan dibaca untuk pergerakan *bot*. Semua koordinat posisi pergerakan dari *bot* dan langkah-langkah yang diambil oleh *bot* untuk mencapai koordinat tertentu akan kembali direkam lagi ke RL *Memory*. Perekaman data tersebut berfungsi sebagai bahan pertimbangan untuk pembelajaran berikutnya dan akan kembali diolah untuk mendapatkan peningkatan dalam pergerakan. Diagram dapat dilihat pada Gambar 3.1.

**3.1.1.4 Merekam Cara Menembak *Bot***

*Bot* akan diberikan instruksi untuk menembak dengan sudut tembak, jenis tembakan dan musuh yang ditargetkan. Pergerakan *bot* akan membaca Data Gerakan pada RL *Memory* dan hasil pergerakan yang terjadi tidak akan direkam. Perekaman data hanya akan terjadi apabila *bot* sudah mendekati musuh. Koordinat posisi pergerakan dari *bot* (*State*) dan aksi yang diambil oleh *bot* (*Action*) untuk menjatuhkan musuh akan direkam ke RL *Memory*. Diagram rekam dan penyimpanan data dapat dilihat pada Gambar 3.3.



**Gambar 3.3 Diagram Merekam Cara Menembak *Bot***

**3.1.2 Analisis Kebutuhan**

Analisis kebutuhan merupakan tahapan penting dalam pengembangan sistem untuk mengetahui secara detail sistem yang ingin dikembangkan. Kebutuhan sistem akan ditentukan oleh pemilik dan pengguna sistem. Analisis kebutuhan dibagi menjadi dua, yaitu analisis kebutuhan fungsional yang merupakan spesifikasi inti mengenai hal-hal yang bisa dilakukan oleh sistem dan kebutuhan non-fungsional yang merupakan komponen pendukung pada sistem.

**3.1.2.1 Kebutuhan Fungsional**

1. **Kebutuhan Fungsional untuk Pembelajaran**

Adapun beberapa persyaratan fungsional yang harus dipenuhi oleh perangkat lunak adalah sebagai berikut:

1. Perangkat Lunak Game Maker Ver 8.0 untuk menjalankan permainan.
2. Perangkat Lunak SQL Lite untuk mengolah data pembelajaran.
3. Mengunakan Script GML untuk melakukan *coding*.
4. Mengunakan *source code* Gang Garrison 2.
5. Text Files untuk menulis dan membaca data.
6. Pengunaan Version Control Git untuk melakukan pengaturan versi.
7. Instruksi dasar pergerakan bot sebagai pengetahuan awal dari *bot*.
8. Nilai ambang untuk mengoptimalkan data yang telah ditulis.
9. **Kebutuhan Fungsional untuk Pengujian**

Adapun beberapa persyaratan fungsional yang harus dipenuhi untuk melakukan pengujian adalah sebagai berikut:

1. Data pembelajaran yang disimpan sudah diolah.
2. Pengujian dilakukan dengan cara membaca data yang sudah diolah.
3. Proses pengujian dilakukan dengan memulai permainan antara *bot* RL dengan *bot* dari pengembang lain.
4. Proses pengujian dimulai dengan pembacaan data dari database, kemudian bot akan memilih rangkaian gerakan yang memiliki reward yang optimal. Hasil yang diperoleh dari satu set permainan menentukan apakah *bot* sudah mengetahui cara untuk memperoleh kemenangan.

**3.1.2.1 Kebutuhan Non-Fungsional**

Untuk merumuskan persyaratan non-fungsional dari sistem, maka harus dilakukan analisis terhadap kinerja, informasi, ekonomi, keamanan aplikasi, efisiensi, dan pelayanan customer. Panduan ini dikenal dengan analisis PIECES (*performance*, *information*, *economic*, *control*, *efficiency*, dan *services*). Berikut penjabaran dari analisis yang digunakan :

1. **Kebutuhan Non-Fungsional untuk Pembelajaran**
2. *Performance*

Penulisan data dilakukan pada RAM terlebih dahulu untuk mengurangi beban pada *disk* (memori utama).

1. *Information*

Data yang telah ditulis disimpan dalam Text Files.

1. *Efficiency*

Data yang sudah ditulis diolah terlebih dahulu sebelum dibaca.

1. **Kebutuhan Non-Fungsional untuk Pengujian**
2. *Performance*

Pembacaan data dilakukan pada RAM terlebih dahulu untuk mengurangi beban pada *disk* (memori utama).

1. *Information*

Data dari *bot* dapat ditampilkan ketika bot melakukan pergerakan.

1. *Efficiency*

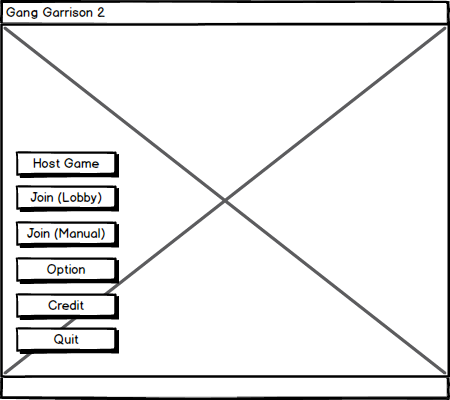
Data yang dibaca sudah diolah terlebih dahulu untuk mendapatkan hasil yang optimal.

**3.2 Pemodelan Sistem dan Perancangan**

Pada tahap ini, dilakukan pemodelan untuk tampilan yang sudah dibangun.

**a. Pemodelan Halaman Utama**

Tampilan ini merupakan tampilan pembuka dari aplikasi dan merupakan tampilan utama yang berfungsi untuk menghubungkan ke tampilan berikutnya. Pemodelan tampilan utama dapat dilihat pada gambar 3.4.

****

Gambar 3.4 Mockup tampilan awal.

Pada tampilan awal terdapat beberapa menu yaitu :

1. Host Game

Membuat server dengan settingan yang diatur oleh pemain sendiri.

1. Join (Lobby)

Mencari server yang dibuat oleh orang lain secara otomatis ( tidak digunakan )

1. Join (Manual)

Mencari server yang dibuat oleh orang lain dengan cara mencari nama server yang dibuat oleh orang lain ( tidak digunakan )

1. Option

Pengaturan dari game. ( tidak digunakan )

1. Credit

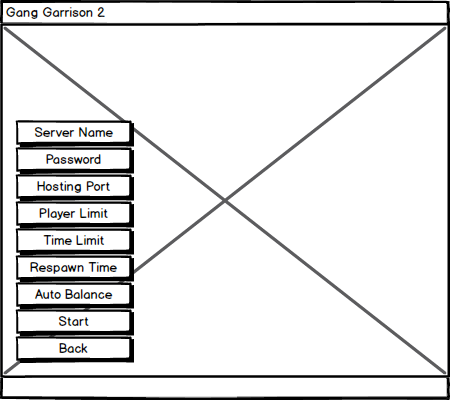
Keterangan mengenai pembuat game serta pengembangnya ( tidak digunakan )

1. Quit

Keluar dari aplikasi permainan.

**b. Pemodelan Tampilan Host Game**

Pemodelan ini adalah kelanjutan dari pemodelan tampilan utama. Dimana pemodelan ini adalah tampilan untuk mengatur *server* untuk melakukan permainan. Pemodelan dapat dilihat pada gambar 3.5.



Gambar 3.5 Mockup menu Host Game.

Pada tampilan awal terdapat beberapa menu yaitu

1. Server Name

Memberi nama pada server ( tidak digunakan )

1. Password

Memberi password pada server ( tidak digunakan )

1. Hosting Port

Port pada server game ( tidak digunakan )

1. Player Limit

Mengatur jumlah maksimum pemain dalam satu permainan ( tidak digunakan )

1. Time Limit

Mengatur lamanya waktu permainan dalam satu permainan.

1. Respawn Time

Mengatur waktu untuk bangkit kembali setelah terbunuh. ( tidak digunakan )

1. Auto Balance

Mengatur agar jumlah tim diseimbangkan secara otomatis. (tidak digunakan)

1. Start

Memulai permainan.

1. Back

Kembali ke tampilan sebelumnya.



Gambar 3.6 Mockup saat Start Game.

Keterangan mockup :

1. 1 : Memilih tim secara acak ( tidak digunakan )
2. 2 : Memilih sebagai penonton atau pengamat dalam permainan.
3. 3 : Memilih bermain sebagai tim merah ( tidak digunakan )
4. 4 : Memilih bermain sebagai tim biru ( tidak digunakan )
5. Keterangan : Menampilkan keterangan pada menu yang di pilih *pointer*.
6. Peta Game : Menampilkan peta permainan.

Perancangan sistem menggunakan activity diagram.



Gambar 3.7 Activity Diagram.

1. Run Game

Memulai permainan, bot akan diberikan instruksi untuk bergerak dengan berjalan ke kiri (L) , kanan (R), melompat (J) dan berhenti (S).

1. Learn Move

Semua koordinat posisi pergerakan dari bot (State) dan langkah-langkah yang diambil oleh *bot* (Action) untuk mencapai koordinat tertentu (State') akan direkam. Kemudian dilakukan pengecekan apakah *bot* mempelajari gerakan baru, jika ada maka akan dilakukan perhitungan *reward* (*Calculate Reward* ) dan data pembelajaran akan disimpan pada RL *Memory*.

1. Optimized Data & Run Game

RL *Memory* dinormalisasi agar pembacaan data menjadi lebih efisien. Penormalisasian gerakan pada RL *Memory* dapat dilakukan dengan cara membatasi nilai ambang dari *state* (S/posisi), dan *action* yang bebeda (A) dengan total reward masing masing yang *maximum* (Rt) serta membuang semua *state* dan *action* yang dilakukan dengan *reward* <= 0. Kemudian dilakukan permainan baru untuk belajar menembak.

1. Learn Shooting

*Bot* akan diberikan instruksi untuk menembak dengan sudut tembakan, jenis tembakan dan musuh yang ditargetkan. Pergerakan *bot* akan membaca Data Gerakan pada RL *Memory* dan hasil pergerakan yang terjadi tidak akan direkam. Perekaman data hanya akan terjadi apabila *bot* sudah mendekati musuh. Koordinat posisi pergerakan dari *bot* (*State*) dan aksi yang diambil oleh *bot* (*Action*) untuk menjatuhkan musuh akan direkam ke RL *Memory*.