BAB IV

IMPLEMENTASI DAN PENGUJIAN

4.1 Implementasi

Tahap implementasi merupakan tahap penciptaan perangkat lunak dan juga tahap kelanjutan dari kegiatan perancangan aplikasi, tahap ini merupakan tahap dimana aplikasi siap untuk dioperasikan. Aplikasi ini diimplementasikan dengan nama Neural Network FPS *game* dengan *file extension*-nya *.exe*.

4.1.1 Implementasi Perangkat Keras

Perangkat keras yang dapat digunakan untuk menjalankan aplikasi *game* ini harus memiliki spesifikasi yang cukup tinggi. Sebagai ilustrasi sistem minimum agar aplikasi ini dapat dijalankan adalah sebagai berikut :

1. Processor Dual Core 1,80 Ghz
2. RAM 1 GB
3. VGA 512 MB
4. Harddisk 1 GB
5. *Mouse* dan *Keyboard*

4.1.2 Implementasi Perangkat Lunak

Adapun spesifikasi perangkat lunak yang dibutuhkan untuk menjalankan aplikasi game ini adalah :

1. Sistem operasi minimal Windows Xp
2. *Graphic card*
3. DirectX 9 keatas

4.1.3 Implementasi Instalasi Aplikasi Game

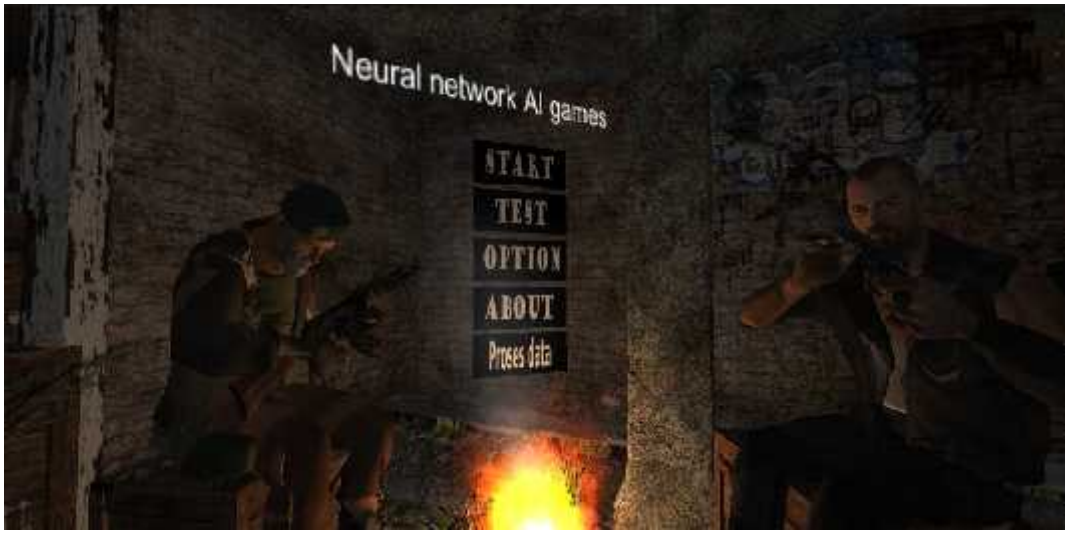
Untuk dapat memainkan game Implementasi Algoritma Neural Network Dalam AI BOT Game FPS , pemain terlebih harus mempunyai *file* Neural Network FPS.*exe* . Untuk mulai memainkan *game*, pemain hanya perlu menjalankan *file* Neural Network FPS.*exe* yang telah ada dalam perangkat keras.

4.1.4 Implementasi Antarmuka

Antarmuka dapat membuat pengguna merasa nyaman dan bertahan lama berada dalam aplikasi atau sebaliknya. Berikut ini merupakan antarmuka yang ada pada aplikasi *game* Implementasi Algoritma Neural Network Dalam AI BOT Game FPS.

1. Antarmuka Menu Utama

   Saat pengguna membuka aplikasi pertama kali, maka antarmuka yang ditampilkan oleh sistem adalah antarmuka menu utama. Berikut ini merupakan tampilan dari antarmuka menu utama :



Gambar 4.1 Antarmuka Menu Utama

Gambar 4.1 di atas merupakan menu utama pada *game* Implementasi Algoritma Neural Network Dalam AI BOT Game FPS. Terdapat 6 (tiga) sub menu di dalamnya, yaitu menu Start yang berfungsi untuk memulai permainan, menu Test yang berfungsi untuk memulai testing stage, menu option untuk melihat informasi kontrol dan pengaturan sistem, menu about untuk melihat informasi tentang *game* , menu proses data untuk memulai proses pengolahan data training dan menu Keluar yang berfungsi untuk keluar dari aplikasi.

2. Antarmuka Menu Mainkan

Saat pengguna memilih menu start, maka antarmuka yang ditampilkan oleh sistem adalah; Pertama, antarmuka *Loading schene* berupa Tampilan loading dengan logo Unity . Selanjutnya akan masuk ke antarmuka permainan. Berikut ini merupakan tampilan dari antarmuka menu mainkan.



Gambar 4.2 Antarmuka *Loading scene*

Gambar IV.2 merupakan antarmuka *Loading scene* Permainan utama yang terdapat pada *game* Implementasi Algoritma Neural Network Dalam AI BOT Game FPS.



Gambar 4.3 Antarmuka Option

Gambar 4.3 merupakan antarmuka opsi yang terdapat pada *game* Implementasi Algoritma Neural Network Dalam AI BOT Game FPS.



Gambar 4.4 Antarmuka *Testing range*

Gambar 4.4 merupakan antarmuka permainan yang terdapat pada *game* Implementasi Algoritma Neural Network Dalam AI BOT Game FPS. Gambar tersebut memperlihatkan posisi pemain ketika masuk ke dalam *Testing range*.



Gambar 4.5 Antarmuka Permainan Utama

Gambar IV.5 menunjukan posisi pemain ketika akan menghadapi AI BOT dalam menu permainan utama.

4.2 Pengujian

Pada umunya ada 2 (dua) macam teknik pengujian yaitu *white box testing* dan *black box testing*. Pengujian yang menggunakan teknik *white box testing* lebih menitikberatkan pada struktur internal (*source code*) program, dimana setiap baris kode yang akan di*compile* atau dicek satu persatu untuk mengetahui apakah masih terjadi *error*. Pengujian pada *black box testing* tidak terlalu memperhatikan struktur internal (*source code*) dari program, tetapi lebih kepada menemukan kesalahan program. Pada *game* Implementasi Algoritma Neural Network Dalam AI BOT Game FPS digunakan pengujian *black box testing*. Cara pengujian dilakukan melalui dua tahap, yaitu pengujian *alpha* (fungsional) dan pengujian *beta*.

4.2.1 Pengujian Blackbox

Pengujian dilakukan terhadap aplikasi untuk memastikan bahwa aplikasi dapat berjalan dengan benar sesuai dengan kebutuhan dan tujuan yang diharapkan. Pengujian *alpha* berfokus pada persyaratan fungsional perangkat lunak.

4.2.1.1 Skenario Pengujian Aplikasi

Skenario pengujian menjelaskan pengujian terhadap sistem yang ada pada aplikasi *game* Implementasi Algoritma Neural Network Dalam AI BOT Game FPS. Skenario pengujian yang akan dilakukan pada aplikasi ini selengkapnya dapat dilihat pada table 4.1 :

Tabel 4.1 Skenario Pengujian Aplikasi

No	Komponen yang di uji	Skenario Pengujian	Jenis Pengujian
1.	Menu Start	Klik menu start	*Black Box*
2.	Menu Testing	Klik menu Testing	*Black Box*
3.	Menu Option	Klik menu Option	*Black Box*
4.	Menu About	Klik Menu About	*Black Box*
5.	Menu Proses Data	Klik Menu Proses Data	*Black Box*
6.	Menu Exit	Klik menu Exit	*Black Box*
7.	Berjalan	Tekan tombol W *keyboard*	*Black box*
		Tekan tombol A *keyboard*	*Black box*
		Tekan tombol D *keyboard*	*Black box*
		Tekan tombol S *keyboard*	*Black box*
8.	Meloncat	Menekan spasi pada *keyboard*	*Black box*
9.	Mengambil *Health Pack*	Mengarahkan karakter hingga menyentuh *icon Health Pack*	*Black box*
10.	Mengambil Peluru	Mengarahkan karakter hingga menyentuh *icon* peluru	*Black box*

No	Komponen yang di uji	Skenario Pengujian	Jenis Pengujian
11.	Algortima A* (A *star*)	Pencarian rute atau jalan terbaik untuk NPC	*Black Box*
12.	Algoritma *Neural Network Multilevel Feed Forward*	Perilaku NPC	*Black Box*

4.2.1.2 Kasus dan Hasil Pengujian (*Black Box Testing*)

Berikut ini merupakan kasus untuk menguji perangkat lunak yang sudah dibangun dengan menggunakan metode *black box testing* berdasarkan skenario pengujian aplikasi yang terdapat pada tabel 4.1 yang sudah dijabarkan. Hasil pengujian yang akan dilakukan pada aplikasi ini selengkapnya dapat dilihat pada tabel 4.2

Tabel 4.2 Skenario Pengujian Black Box

No	Kasus yang akan di uji	Skenario dan Hasil Uji		
		Skenario Uji	Hasil yang Diharapkan	Kesimpulan
1.	Menu Start	Klik Menu Start	Sistem menampilkan halaman *Loading*	[ ] Berhasil [ ] Tidak Berhasil
2.	Menu Option	Klik Menu Option	Sistem menampilkan kontrol dan *setting* yang digunakan dalam *game*	[ ] Berhasil [ ] Tidak Berhasil
		Klik Back	Sistem menampilkan menu utama	[ ] Berhasil [ ] Tidak Berhasil

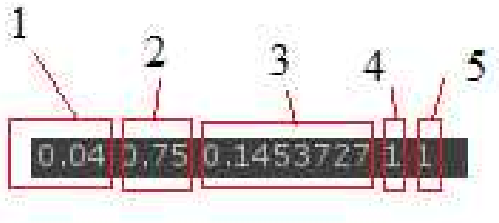
No	Kasus yang akan di uji	Skenario dan Hasil Uji		
		Skenario Uji	**Hasil yang Diharapkan**	**Kesimpulan**
4.	Menu Testing	Klik Menu testing	Sistem menampilkan Testing Scene	[✓] Berhasil [ ] Tidak Berhasil
5.	Menu About	Klik menu About	Sistem menampilkan Scene informasi tentang program	[✓] Berhasil [ ] Tidak Berhasil
6.	Menu Keluar	Klik Menu Keluar	Menutup atau Keluar dari aplikasi	[✓] Berhasil [ ] Tidak Berhasil
4.	Berjalan	Tekan tombol W *keyboard*	Karakter pemain bergerak kedepan	[✓] Berhasil [ ] Tidak Berhasil
		Tekan tombol A *keyboard*	Karakter pemain bergerak kekiri	[✓] Berhasil [ ] Tidak Berhasil
		Tekan tombol S *keyboard*	Karakter pemain bergerak kebelakang	[✓] Berhasil [ ] Tidak Berhasil
		Tekan tombol D *keyboard*	Karakter pemain bergerak kekanan	[✓] Berhasil [ ] Tidak Berhasil
5.	Meloncat	Menekan tombol spasi pada *keyboard*	Karaker akan meloncat	[✓] Berhasil [ ] Tidak Berhasil

No	Kasus yang di uji	Skenario dan Hasil Uji		
		Skenario Uji	**Hasil yang Diharapkan**	**Kesimpulan**
6.	Mengambil *Health Pack*	Mengarahkan karakter hingga menyentuh *icon Health Pack* menggunakan *mouse* dan *keyboard*	*Health Bar* pada permainan akan bertambah jika darah pemain memang tidak penuh dan *icon* darah menghilang	[√] Berhasil [ ] Tidak Berhasil
7.	Mengambil Peluru	Mengarahkan karakter hingga menyentuh *icon* peluru menggunakan *mouse* dan *keyboard*	*Bar* peluru pada permainan akan bertambah jika peluru pemain tidak penuh dan *icon* peluru menghilang	[✓] Berhasil [ ] Tidak Berhasil
8.	Menembak	Mengarahkan *pointer* / *cursor* senjata menggunakan *keyboard* dan *mouse* lalu menekan tombol kiri *mouse* untuk mengeluarkan peluru / *mouse*	Senjata yang dipegang karakter pada *game* akan mengeluarkan peluru dan suara	[√] Berhasil [ ] Tidak Berhasil
9.	Mengganti Senjata	Menekan tombol 1 pada *keyboard* untuk menggunakan senjata laras Pendek dan menekan tombol 2 pada *keyboard* untuk menggunakan senjata laras Panjang	Senjata laras panjang akan digunakan apablia pemain menekan tombol 2 dan Senjata laras pendek akan digunakan apablia pemain menekan tombol 1	[√] Berhasil [ ] Tidak Berhasil
10.	Memberhentikan permainan	Menekan tombol P pada *keyboard*	*Game* akan mengeluarkan *screen pause*	[√] Berhasil [ ] Tidak Berhasil

No	Kasus yang akan di uji	Skenario dan Hasil Uji		
		Skenario Uji	**Hasil yang Diharapkan**	**Kesimpulan**
11.	Algortima A* (A *star*)	Pencarian *walkable Node* dalam *map* permainan.	*Walkable node* yang optimal ditemukan dan bisa dilalui.	[✓] Berhasil [ ] Tidak Berhasil

Untuk pengujian algoritma neural network akan dilakukan secara black box, berikut adalah prosesnya:

1. Proses pengecekan nilai input yang ditandai dengan nilai 1 (nilai sesuai format) dan 0 (tidak sesuai Format) pada bagian akhir dari setiap baris dalam data input



Gambar 4.6 Keterangan variabel nilai input untuk proses learning

Penjelasan untuk nilai nilai yang digunakan sebagai inputan adalah sebagai berikut :

1. Nilai Total tembakan

   total tembakan keseluruhan yang dilakukan, nilai maksimal 100 kali tembakan, pengklasifikasian kemampuan *player* berdasarkan nilai total tembakan, untuk kemampuan *expert* berada pada range 0,03 hingga 0,35, untuk kemampuan *medium* berada pada range 0,36 hingga 0,75, dan untuk kemempuan *easy* berada pada range 0,76 hingga 1.

2. Nilai akurasi tembakan

   perbandingan antara tembakan yang mengenai target dan tembakan yang meleset, pengklasifikasian kemampuan *player* berdasarkan nilai akurasi tembakan, untuk kemampuan expert berada pada range 1 hingga 0,66, untuk kemampuan *medium* berada pada range 0,65 hingga 0,34, dan untuk kemampuan easy berada pada range 0,33 hingga 0,1.

3. Nilai Waktu rata-rata

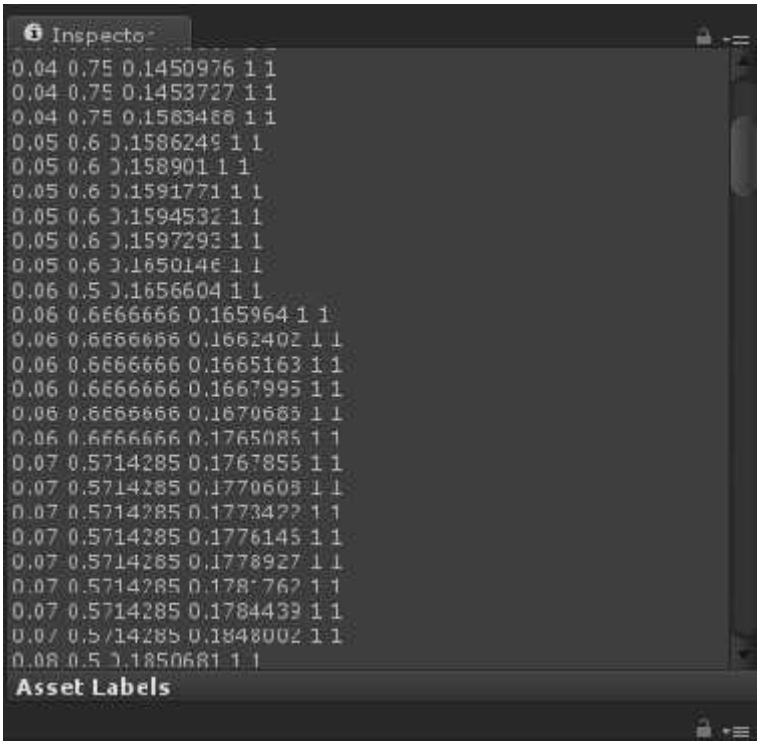
   Waktu keelurukan dalam menyelesaikan stage testing dibagi dengan jumlah stage, yaitu 5 stage, pengklasifikasian kemampuan *player* berdasarkan nilai waktu rata-rata, untuk kemampuan *expert* berada pada range 0,14 hingga 0,35, untuk kemampuan *medium* berada pada range 0,36 hingga 0,66 dan untuk kemampuan *easy* antara 0,67 hingga 1.

4. Nilai Health Point

   Nilai health point yang dimiliki player, dibagi dengan 100 yang merupakan nilai maksimal health point player, pengklasifikasian kemampuan *player* berdasarkan nilai health point, untuk kemampuan *expert* berada pada range 1 hingga 0,66, untuk kemampuan *medium* berada pada range 0,65 hingga 0,45 dan untuk kemampuan *easy* pada rage 0,44 hingga 0,01.

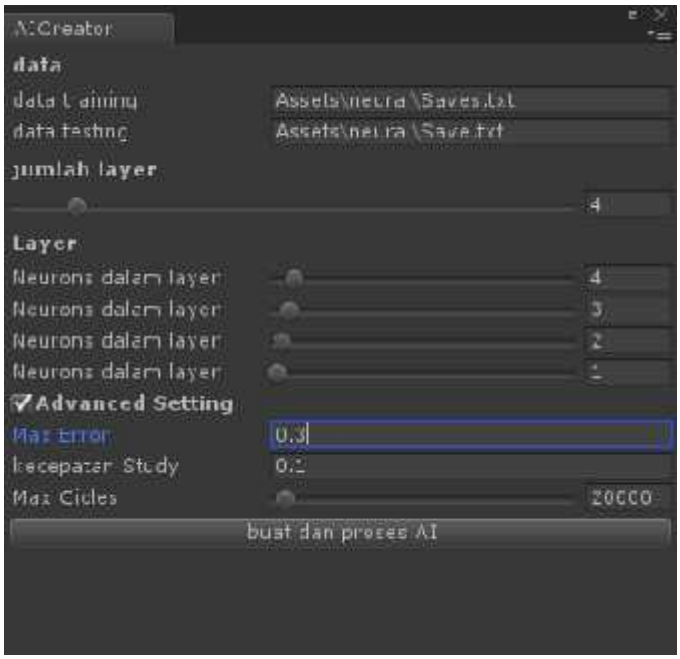
5. Nilai Checker

   Merupakan bentuk validasi nilai input, apabila nilai input sesuai dan berada di range 0 sampai dengan 1 maka checker akan memberikan nilai 1, dan diluar range yang telah ditentukan maka checker akan memberikan nilai 0.



Gambar 4.7 Data input untuk proses learning

6. Proses learning dilakukan sesuai dengan parameter parameter masukan yang telah ditentukan



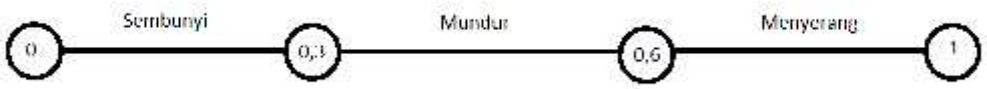
Gambar 4.8 Tampilan proses learning secara manual

7. Setelah proses learning berakhir, maka akan muncul nilai yang akan dijadikan penentu keputusan (ditandai dengan kotak merah)



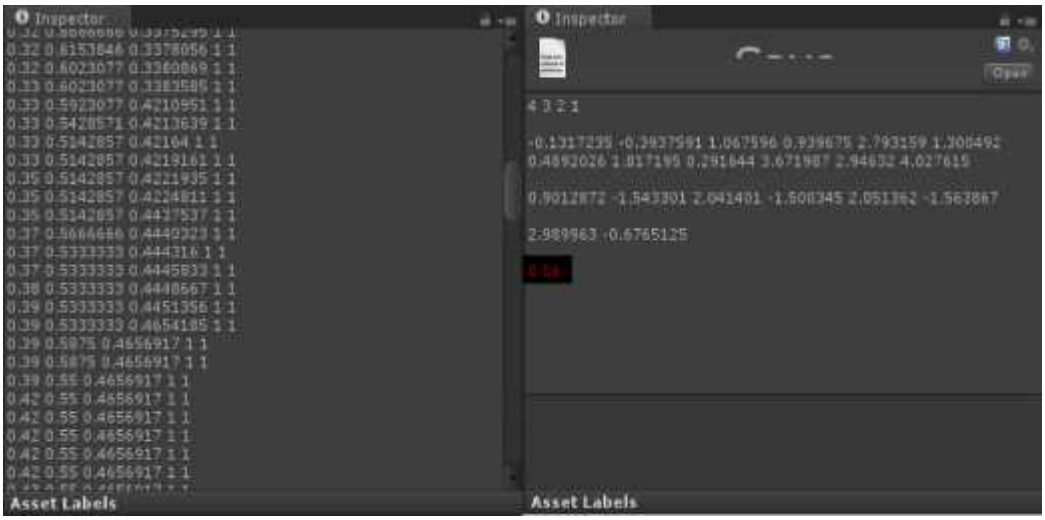
Gambar 4.9 Log hasil proses learning

Maka berdasarkan hasil yang didapatkan, keputusan yang akan diambil oleh AI BOT adalah menyerang, karena hasil yang didapatkan bernilai 1



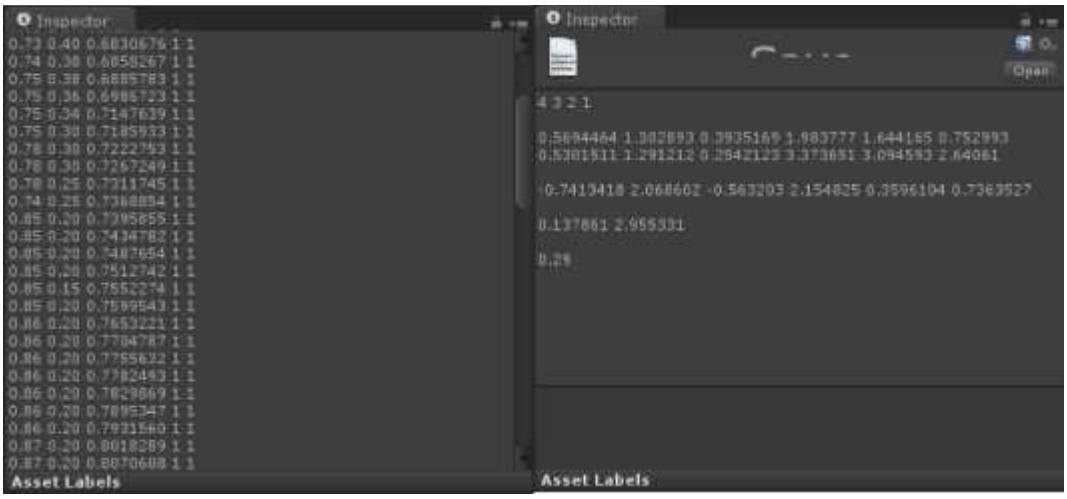
Gambar 4.10 Aturan dalam penentuan keputusan

Pengujian dilakukan pula menggunakan data learning berdasarkan kemampuan player yang beragam, berikut adalah hasil percobaan data training berdasarkan kemampuan *player medium* dan *easy:*



(a) (b)

Gambar 4.11 (a) Data training dan (b) hasil training *player* dengan kemampuan *medium.*



(a) (b)

Gambar 4.12 (a) Data training dan (b) hasil training *player* dengan kemampuan *easy.*

4.2.1.3 Kesimpulan Pengujian Blackbox

Dari pengujian Blackbox yang dilakukan, seluruh skenario berhasil dieksekusi sesuai dengan hasil yang diharapan, adapun untuk proses dan hasil pengujian algoritma A* secara Blackbox memberikan hasil Walkable Node yang sesuai untuk setiap independent path yang diproses tanpa menimbulkan terjadinya infinite loop dalam proses pencarianya, sedangkan untuk pengujian algoritma *Neural Network Multilevel Feed Forward* sendiri memberikan hasil yang sesuai dengan hasil yang diharapkan, dimulai dari proses pembacaan data training hingga proses looping dilakukan.

Selama looping untuk proses learning dilakukan, data learning akan mendapatkan tambahan inputan baru yang diperoleh selama berlangsungnya permainan, hal ini tentu saja akan berpengaruh pada keputusan yang akan di ambil, semakin banyak data baru yang ditambahkan, maka akan semakin sesuai pergerakan dan respon BOT dengan kemampuan yang player miliki.