NON-PLAYER CHARACTER IN FIRE FIGHTER GAMES USING GENETIC ALGORITHM

Muhammad Fikriansyah [1], Giri Wahyu Wiriasto [2], A. Sjamsjiar Rachman [3]
[1, 2, 3] Program Studi Teknik Elektro, Fakultas Teknik, Universitas Mataram
giriwahyuwiriasto@unram.ac.id

ABSTRAK

Game ini berjenis *role playing game (RPG)* dengan tema *Game Pemadam Kebakaran*. Dalam role skenario, terdapat dua karakter pemain, yakni human player dan komputer player. Karakter *human player* memiliki dua misi yaitu menyelamatkan hutan yang terbakar dengan cara memadamkan dan mengalahkan karakter komputer player menggunakan senjata air, sedangkan misi komputer player kebalikannya. Khusus untuk komputer player, karakternya bergerak secara mandiri. Karakter komputer player berupa *NPC (non-player character*) dimana pada permainan ini berwujud karakter *monster-api*. Perilaku mandiri dari NPC dikendalikan oleh algoritma yang disematkan padanya. Pada artikel ini dijelaskan implementasi metode Algoritma Genetika (GA) sebagai metode yang melatarbelakangi perubahan perilaku pada NPC bersifat acak dipengaruhi berdasarkan kondisi parameter disekitarnya. Parameter tersebut diinisialisasikan sebagai variabel dalam struktur algoritma genetiknya. Metode ini berjalan dengan memaksimalkan nilai parameter yang disimpan dalam *gen* pada suatu kromosom suatu *individu*. Pada satu individu kromosom yang dibangkitkan, terdapat empat gen seperti kemampuan menyerang (*Attack*), kecepatan berpindah (*Movement Speed*), kecepatan menembak musuh (*Attack speed*) dan parameter ketahanan (*Health point*). Perilaku NPC pada karakter *monster-api* tergambarkan dalam diagram *finite state machine* (FSM). Pada *FSM* tampak perilaku dalam bentuk *state*, *event*, dan *action* dimana kondisi dapat berubah sesuai interaksi yang terjadi dan direncanakan dalam *godot scene*. Hasil ujicoba pada penelitian ini menunjukkan bahwa pembangkitan populasi pada GA sampai dengan generasi ke-5 mendapatkan nilai parameter *fitness* 134,87. Untuk pembangkitan populasi hingga generasi ke-10, nilai *fitness* 128,64. Pada setiap pembangkitan generasi akan memperoleh nilai parameter *fitness* yang optimum ditunjukkan dengan perolehan nilai *gen* pada kromosom populasi mendapat nilai yang tertinggi.

Keyword : *Algoritma Genetika, NPC, Non-Player Character, Game Pemadam Kebakaran.*

1. PENDAHULUAN

G*ame*s merupakan permainan elektronik yang melibatkan interaksi antarmuka dengan pengguna untuk menghasilkan umpan balik secara visual pada perangkat komputer [1]. *Game* memiliki berbagai macam jenis yang dapat dimainkan dan salah satunya adalah *Role Playing Game* (RPG). *Game* RPG adalah sebuah permainan yang para pemainnya memainkan peran tokoh-tokoh khayalan dan berkolaborasi untuk merajut sebuah cerita bersama [2]. Pada *game* jenis RPG akan memasukkan aspek penceritaan yang kompleks serta karakter yang membuat seseorang merasa seperti menjadi tokoh yang diperankannya dalam *game* tersebut [3].

Selain tokoh yang diperankan oleh pemain, adapun tokoh yang diperankan oleh NPC. NPC merupakan objek dinamis yang tidak dikendalikan oleh user atau pengguna melainkan dapat memutuskan tindakannya sendiri dan beroperasi dalam ruang virtual [4]. Untuk NPC dapat memutuskan tindakannya sendiri diperlukan kecerdasan buatan atau AI. Kecerdasan buatan merupakan tentang bagaimana membuat komputer dapat berpikir layaknya manusia dan hewan lakukan [5]. Dengan menggunakan NPC yang mengimplementasikan kecerdasan buatan maka sebuah *game* akan menjadi seru karena permainan tidak lagi monoton, dan lebih menantang untuk dimainkan [6].

Salah satu algoritma yang dapat digunakan dalam pembuatan AI di dalam *game* adalah algoritma genetika (GA) [21]. GA merupakan salah satu metode heuristik yang merupakan cabang dari *evolutionary algorithm* [22], yaitu suatu teknik untuk memecahkan masalah-masalah optimasi yang rumit dengan menirukan proses evolusi biologis makhluk hidup [7].

Berdasarkan latar belakang tersebut, pada penelitian ini dirancang *game* RPG dengan tema *"Game Pemadam Kebakaran"*. *Role* skenario *Game* ini berlatar di suatu wilayah yang terdampak bencana kebakaran hutan yang bersumber dari objek bangunan tengah hutan karena kelalaian penghuni bangunan. Kebakaran yang terjadi di tengah hutan segera menjalar cepat ke wilayah hutan yang lain sehingga pancaran api diilustrasikan sebagai karakter *monster-api* yang muncul dan membakar lingkungan sekitar secara acak serta menguasai wilayah tersebut dengan api. Monster-api ini sebagai komputer player atau NPC. Untuk mencegah hal tersebut, seorang petugas pemadam kebakaran sebagai karakter human player bertugas memadamkan api dengan cara membasmi para monster-api dengan senjata tembakan-air sehingga sumber titik api berhasil dipadamkan.

2. TINJAUAN PUSTAKA

2.1. *Non Player Character* (NPC)

Non-Player Character atau biasa disingkat NPC merupakan karakter dengan perilaku mandiri yang biasa digunakan dalam komputer animasi dan media interaktif seperti pada *games* dan *virtual reality*. NPC dalam sebuah *game* biasanya akan mewakili tokoh dalam cerita atau permainannya yang memiliki kemampuan untuk mengimprovisasikan tindakannya sendiri [8].

2.2. Algoritma Genetika

Algoritma genetika (GA) adalah sebuah algoritma pencarian *class* stokastik yang didasari dari teori evolusi biologis [9]. Pendekatan yang diambil oleh algoritma ini adalah dengan menggabungkan secara acak berbagai pilihan solusi terbaik di dalam suatu kumpulan untuk mendapatkan generasi solusi terbaik berikutnya yaitu pada suatu kondisi yang memaksimalkan kecocokannya atau lazim disebut *fitness.* Generasi ini akan merepresentasikan perbaikan-perbaikan pada populasi awalnya. Dengan melakukan proses ini secara berulang, algoritma ini diharapkan dapat mensimulasikan proses evolusioner [10]. Secara umum, GA akan memiliki 5 tahapan utama sebagai berikut.

2.3. Pembangkitan Populasi Awal

Tahapan pertama dalam GA adalah pembangkitan populasi awal. Ada beberapa hal yang perlu diketahui sebelum bisa melakukan pembangkitan populasi yaitu [11], [12] :

a. Populasi merupakan kumpulan dari individu yang diproses dalam satu siklus GA.
b. Individu merupakan objek yang terbentuk dari sebuah kromosom atau lebih.
c. Kromosom adalah gabungan dari gen-gen yang akan membentuk arti tertentu.
d. Gen merupakan variabel yang menjadi dasar dalam pembentukan suatu kromosom.

2.4. Perhitungan *Fitness*

Fungsi *fitness* merupakan fungsi evaluasi yang akan menyeleksi kromosom untuk tetap bertahan atau tidak, dimana setiap generasi pada GA akan memilih kromosom yang akan mendekati nilai solusi. Fungsi ini memiliki dua jenis pemilihan kromosom yaitu kondisi maksimasi dan minimasi. Untuk maksimasi, fungsi tujuan akan dijadikan fungsi *fitness* yang menuntut nilainya untuk tetap semakin besar dengan tujuan agar kemungkinan terpilihnya suatu kromosom juga semakin besar. Untuk animasi, fungsi *fitness* akan dirumuskan sedemikian rupa untuk membuat fungsi tujuan yang semakin kecil akan mendapatkan nilai *fitness* yang optimum [13].

2.5. Seleksi

Roulette wheel (roda rolet) merupakan salah satu metode seleksi yang paling sering digunakan dalam GA. Prinsip dari metode ini adalah masing-masing individu dalam populasi akan dilakukan perhitungan probabilitas nilai *fitness* yang kemudian akan direpresentasikan sebagai suatu roda rolet dengan pembagian wilayahnya secara proporsional tergantung dari nilai probabilitas yang sudah dihitung sebelumnya. Proses pemilihan dilakukan dengan membangkitkan nilai secara acak. Jika nilai bilangan acak masuk dalam range dari salah satu nilai probabilitas *fitness* dari suatu individu ke-i, maka kromosom individu ke-i tersebut akan terpilih sebagai salah satu pasangan kromosom yang akan dilakukan *crossover* [14]. Untuk menghitung nilai probabilitas dari setiap individu dapat menggunakan persamaan berikut:

$$P_{individu} = \frac{f_{individu}}{f_{total}} \quad (1)$$

Dari persamaan 3, $P_{individu}$ merupakan probabilitas atau peluang individu tersebut akan terpilih, $f_{individu}$ merupakan nilai *fitness* dari individu tersebut dan f_{total} merupakan total nilai *fitness* dari seluruh individu dalam populasi [15]. Untuk menghitung probabilitas komulatif dari setiap individu akan menggunakan persamaan :

$$P_{k1} = P_1 \quad (2)$$
$$P_{k(individu)} = P_{k(individu-1)} + P_{individu} \quad (3)$$

Dari persamaan 2 dan 3, $P_{k(individu)}$ merupakan probabilitas komulatif dari individu saat ini, $P_{k(individu-1)}$ merupakan probabilitas komulatif dari individu sebelumnya dan $P_{individu}$ merupakan probabilitas *fitness* individu saat ini. Sehingga untuk nilai probabilitas komulatif untuk individu pertama P_{k1} akan sama dengan nilai probabilitas *fitness* untuk individu pertama P_1 [16].

2.6. *Crossover*

Crossover atau bisa disebut perkawinan silang memerlukan dua kromosom induk untuk dapat menukarkan sebagian informasi antara kromosom induk pertama dengan kromosom induk kedua. Adapun parameter yang cukup penting dalam proses *crossover* yaitu *crossover rate* yang merupakan nilai perbandingan dari jumlah kromosom yang diharapkan akan melakukan *crossover* terhadap jumlah kromosom dalam populasi. Jika *crossover rate* memiliki nilai yang relatif tinggi maka akan memungkinkan untuk mendapatkan alternatif solusi yang lebih bervariasi dan mengurangi kemungkinan untuk menghasilkan nilai optimum yang tidak dikehendaki. Tetapi, jika nilai *crossover rate* terlalu tinggi maka akan terjadi pemborosan waktu untuk melakukan perhitungan di daerah solusi yang tidak menjanjikan hasil yang optimal [17].

Uniform Crossover melakukan penukaran gen-gen dari satu kromosom dengan kromosom lain melalui tiap index berdasarkan probabilitas. Setiap gen memiliki probabilitas seperti koin. Jika yang muncul kepala, maka gen akan ditukar dan jika muncul buntut, maka posisi gen tetap. *Uniform crossover* dapat menghasilkan rekombinasi kromosom yang lebih baik [18].

2.7. Mutasi

Mutasi merupakan salah satu proses dalam GA untuk menghasilkan variasi kromosom dengan melakukan perubahan acak pada kromosom di mana akan dilakukan pada satu atau lebih bagian dari kromosom tergantung dari *mutation rate*. *Mutation rate* merupakan probabilitas atau kemungkinan jumlah bit atau gen dalam populasi yang diharapkan akan mengalami mutasi. Jika nilai *mutation rate* terlalu kecil, banyak bit atau gen-gen yang berguna mungkin tidak akan muncul dalam populasi. Tetapi, jika nilai *mutation rate* terlalu tinggi maka keturunan yang dihasilkan akan kehilangan sifat-sifat yang mungkin saja merupakan sifat yang unggul dari orang tuanya dan GA akan kehilangan kemampuan untuk belajar dari pencarian-pencarian sebelumnya [17].

Mutasi dengan metode *Random resetting* hanya dapat dilakukan untuk individu dengan nilai *integer*/*float*. Proses yang dilakukan adalah merandom satu indeks gen pada kromosom, kemudian nilai gen tersebut diubah dengan angka yang dirandom ulang [19].

2.8. *Finite State Machine* (FSM)

Finite State Machine (FSM) merupakan teknik perancangan AI dengan memodelkan perilaku (*behavior*) dari sistem atau objek yang kompleks dengan kondisi yang telah didefinisikan dalam setnya [12]. Proses dari FSM terdiri atas tiga hal yaitu state (keadaan), event (kejadian), dan action (aksi). Alur proses dari FSM yaitu ketika sistem sedang berada di state yang sedang aktif, sistem akan berpindah ke state lain jika mendapatkan input atau event tertentu dan saat transisi perpindahan state ini akan disertai dengan action atau aksi yang akan dilakukan oleh sistem, baik berupa aksi yang sederhana maupun aksi yang kompleks [20].

3. METODE PENELITIAN

3.1. Perancangan Algoritma Genetika

Tahapan GA pada *game pemadam kebakaran* akan terdiri dari 5 tahapan yaitu pembangkitan populasi, perhitungan *fitness*, seleksi, *crossover* dan mutasi

Pada tahapan pertama akan dibangkitkannya populasi baru berdasarkan desain kromosom yang dapat dilihat pada tabel 1 berikut.

Tabel 1. Desain kromosom

***Health Point* (HP)**	***Attack* (ATK)**	***Attack Speed* (ASPD)**	***Movement Speed* (SPD)**
50 – 100	10 - 15	1 - 2	0 - 25

Pada tabel 1 dapat dilihat bahwa untuk kromosom pada suatu individu akan memiliki 4 gen yang akan mempengaruhi perilaku dari NPC saat di wilayah permainan. Empat gen tersebut adalah gen *Health Point* (HP) yang akan mempengaruhi parameter ketahanan NPC dari serangan pemain, gen *Attack* (ATK) yang akan mempengaruhi besarnya kekuatan serangan milik NPC, gen *Attack Speed* (ASPD) yang akan mempengaruhi kecepatan serangan NPC untuk setiap detiknya dan gen *Movement Speed* (SPD) yang akan mempengaruhi cepatnya pergerakan NPC di wilayah permainan.

Pada tahapan ini akan dibangkitkan 4 individu awal sebagai satu populasi dengan mengacak nilai gen dari setiap individu berdasarkan batas nilai yang telah ditentukan pada tabel 1 agar setiap individu akan memiliki kromosom dengan nilai gen yang berbeda satu sama lain. Hal ini bertujuan untuk membuat populasi dapat memiliki karakteristik yang bervariasi. Kemudian populasi ini akan dimunculkan di wilayah permainan dalam bentuk NPC monster api yang menjadi lawan dari *player*nya.

Tabel 2. Data hasil permainan

Individu ke-i			
Total *Damage*	*Life time*	*Attack time*	Total *Passed Building*

Ketika semua NPC pada wilayah permainan telah hancur dikalahkan oleh pemain, sistem akan mencatat beberapa data untuk setiap individu sesuai pada tabel 3 yaitu Total *Damage* atau total kerusakan yang dihasilkan oleh individu, *Lifetime* atau lama waktu masa hidup individu, *Attack time* atau total waktu yang diperlukan individu untuk menyerang dan Total *Passed Building* atau jumlah bangunan yang dilewati oleh individu. Selanjutnya data tersebut akan digunakan pada tahapan perhitungan *fitness* dengan menggunakan persamaan 4 berikut.

$$Fitness(i) = \frac{TotalDMG_i + HP_i - AttackTime_i}{AVG\ TimetoBuilding_i} \quad (4)$$

Dimana $Fitness(i)$ merupakan fungsi *fitness* untuk individu ke-i, $TotalDMG_i$ merupakan data total kerusakan (*Damage*) yang dihasilkan oleh individu ke-i selama masa hidupnya di wilayah permainan dimana data ini akan tercatat dan bertambah setiap NPC melakukan serangan, HP_i merupakan data jumlah gen HP atau daya tahan dari individu ke-i sejak awal muncul di wilayah permainan, $AttackTime_i$ merupakan data jumlah waktu yang dihabiskan oleh individu ke-i yang akan

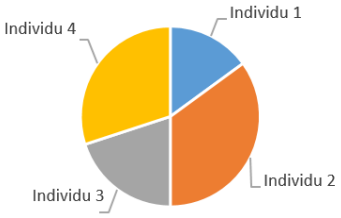
tercatat saat NPC dalam keadaan menyerang selama masa hidupnya di wilayah permainan dan $AVG\ TimetoBuilding_i$ merupakan waktu rata-rata yang diperlukan oleh individu ke-i untuk berjalan dari gedung ke gedung selama masa hidupnya di wilayah permainan dimana waktu rata-rata ini akan dihitung saat NPC berhasil dikalahkan. Perhitungan $AVG\ TimetoBuilding_i$ akan menggunakan persamaan 5 berikut.

$$AVG\ TimetoBuilding_i = \frac{(Lifetime_i - Attacktime_i)}{Total\ Passed\ Building_i} \quad (5)$$

Dengan $Lifetime_i$ merupakan data total waktu hidup individu ke-i dari awal muncul di wilayah permainan hingga hancur, $Attacktime_i$ merupakan total waktu individu ke-i saat berada dalam status menyerang dan $Total\ Passed\ Building_i$ merupakan jumlah bangunan yang dilewati oleh individu ke-i selama masa hidupnya.

Selanjutnya pada tahapan seleksi akan dilakukan pemilihan individu yang akan dijadikan sebagai individu indukan berdasarkan hasil perhitungan *fitness* pada tahapan sebelumnya. Nilai *fitness* untuk setiap individu akan dihitung nilai probabilitasnya menggunakan persamaan 1 dan akan dihitung nilai kumulatifnya menggunakan persamaan 2 dan 3. Dari nilai komulatif ini akan digunakan pada proses pemilihan dengan menggunakan metode *roulette wheel* dimana nilai kumulatif dari suatu individu akan menentukan besarnya probabilitas terpilihnya individu tersebut sebagai individu indukan. Contoh roda rolet dari hasil nilai probabilitas komulatif fitness dapat dilihat pada gambar berikut.

Individu	P_k
1	1,5
2	3,5
3	2
4	3



Gambar 1. Contoh metode *roulette wheel*

Berdasarkan contoh roda rulet pada gambar 1, didapatkan bahwa individu 2 pada roda rolet memiliki wilayah lebih besar dibandingkan dengan individu lainnya. Hal ini disebabkan karena nilai probabilitas individu 2 merupakan nilai probabilitas tertinggi yang menandakan bahwa individu 2 akan memiliki peluang terpilih lebih besar dibandingkan dengan individu lainnya. Sebaliknya, individu 1 pada roda rolet memiliki wilayah lebih kecil dibandingkan dengan individu lainnya. Hal ini disebabkan karena nilai probabilitas individu 1 merupakan nilai probabilitas terendah yang menandakan bahwa individu 1 akan memiliki peluang terpilih lebih kecil dibandingkan dengan individu lainnya.

Saat nilai pada rulet sudah ada maka akan dilakukan 4 kali pemilihan yang mewakilkan 4 individu yang akan dijadikan sebagai indukan. Pemilihan akan dilakukan dengan cara membangkitkan bilangan acak untuk menentukan posisi rulet yang terpilih berdasarkan nilai probabilitasnya.

Pasangan individu indukan yang telah terpilih akan dilakukan proses *crossover* atau persilangan untuk setiap gennya menggunakan metode *uniform crossover* dimana setiap gen akan memiliki nilai probabilitas yang akan dibandingkan dengan nilai *crossover rate*. Nilai *crossover rate* yang digunakan adalah 0,7. Jadi, ketika nilai probabilitas suatu gen lebih rendah dari nilai *crossover rate* maka gen tersebut akan melakukan proses persilangan antara induk yang satu dengan induk pasangannya. Sebaliknya, ketika nilai probabilitas suatu gen lebih tinggi dari nilai *crossover rate* maka gen tersebut tidak akan melakukan proses persilangan. Setelah tahapan ini selesai maka akan didapatkan individu-individu baru yang mewarisi gen-gen milik indukannya.

Tabel 4. Contoh metode *uniform* crossover

Pasangan induk	Kromosom			
	HP	ATK	ASPD	SPD
Induk 1	81	14	1	5
Induk 2	88	12	1	10
Probabilitas	0,467	0,994	0,655	0,552
Anakan 1	*88*	14	*1*	*10*
Anakan 2	*81*	12	*1*	*5*

Berdasarkan contoh crossover pada tabel 5, proses persilangan terjadi pada gen HP, ASPD dan SPD sedangkan gen ATK tidak terjadi proses persilangan. Hal ini disebabkan karena nilai probabilitas yang dibangkitkan pada gen HP, ASPD dan SPD bernilai lebih kecil dibandingkan *crossover rate* yang sudah ditentukan. Sedangkan untuk nilai probabilitas yang dibangkitkan pada gen ATK bernilai lebih besar dibandingkan *crossover rate* yang sudah ditentukan.

Tahapan terakhir adalah mutasi. Pada tahapan ini, beberapa nilai gen pada individu akan diubah menggunakan metode *random resetting*. Metode ini akan memilih satu gen atau lebih yang kemudian nilai dari gen tersebut akan direset nilainya berdasarkan nilai range yang telah ditentukan pada tahap pembangkitan populasi yang dapat dilihat pada tabel 1.

Tabel 6. Contoh metode mutasi *random resetting*

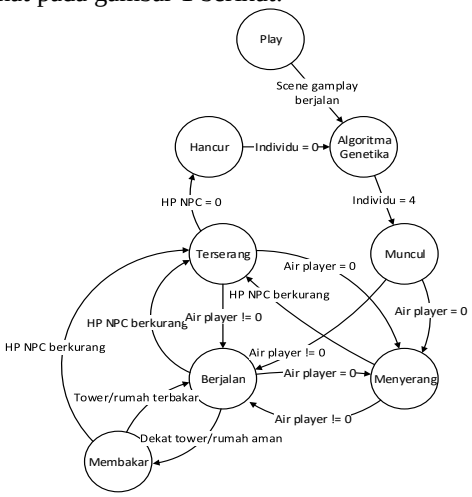
Individu	Kromosom			
	HP	ATK	ASPD	SPD
Sebelum	88	14	1	10
Gen terpilih	HP, ATK			
Sesudah	*93*	*13*	1	10

Berdasarkan contoh mutasi pada tabel 7, nilai pada gen terpilih yaitu HP dan ATK akan diubah secara acak berdasarkan range pada tabel 1. Gen HP mendapat nilai 93 dari range 50 – 100, dan gen ATK mendapatkan nilai 13 dari range 10 – 15. Sedangkan untuk gen yang tidak terpilih tidak akan terjadi perubahan nilai.

Ketika semua tahapan GA telah dilakukan akan didapatkan populasi dengan variasi nilai kromosom yang baru hasil evolusi dari populasi yang lama. Untuk satu siklus tahapan GA akan dihitung sebagai satu generasi. Jika generasi yang dihitung belum mencapai generasi maksimal yang ditentukan maka tahapan GA akan diulang kembali dari perhitungan *fitness* dengan menggunakan populasi baru dari hasil GA pada generasi sebelumnya. Sebaliknya, algoritma ini akan berhenti mengulang prosesnya jika generasi yang dihitung telah mencapai generasi maksimal yang telah ditentukan dan akan menggunakan gen terbaik hasil GA sebagai gen tetap untuk setiap individu pada proses pemunculan NPC selanjutnya.

3.2. Perancangan *Finite State Machine*

Perilaku NPC musuh di dalam *game* akan dirancang menggunakan metode FSM yang dapat dilihat pada gambar 1 berikut.



Gambar 2. Diagram FSM NPC

Pada gambar 2 terdapat diagram FSM yang menggambarkan perilaku dari NPC ketika game dimainkan. Dapat dilihat bahwa diagram FSM memiliki beberapa state yang akan berubah berdasarkan input atau event yang diberikan dan menghasilkan aksi tertentu. Adapun beberapa aksi yang dapat dilakukan oleh NPC berdasarkan state pada diagram FSM di gambar 1 sebagai berikut:

- *state* algoritma genetika = melakukan pembangkitan populasi baru dengan nilai gen yang bervariasi baik pada pembangkitan awal maupun hasil perhitungan GA yang nantinya akan dimunculkan sebagai NPC baru.
- *state* muncul = *generate* individu baru saat semua NPC hancur dan dimunculkan di wilayah permainan kembali
- *state* berjalan = NPC dapat berjalan dengan arah yang acak
- *state* menyerang = NPC dapat menyerang (tembak bola api) saat bertemu *player* saat air habis
- *state* membakar = NPC dapat membakar bangunan yang aman
- *state* terserang = HP NPC akan berkurang saat terkena serangan
- *state* hancur = NPC akan hancur saat HP habis

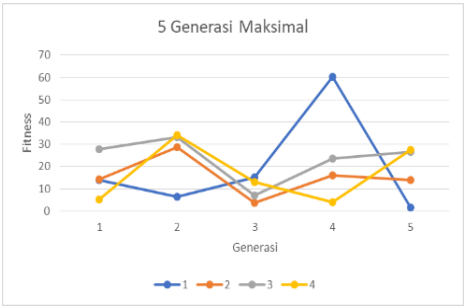
4. HASIL DAN PEMBAHASAN

Seperti pada penelitian [21] disebutkan bahwa dapat dilihat bahwa algoritma genetika dapat diterapkan untuk mengoptimalkan objek parameter GA pada game RPG yang digunakan oleh human player dan komputer player (NPC), berdasarkan statistik masing-masingnya. Pada penelitian lain disebutkan bahwa mempertimbangkan sifat stokastik dari algoritma evolusi, generasi populasi diulang 100 kali [22].

Pada game ini juga telah diperoleh hasil pengujian GA untuk melihat perubahan karakteristik gen dan perilaku NPC pada saat game dimainkan dengan pembangkitan generasi random didapati hasil uji petik nilai parameter fitness terbaik pada generasi ke-5 hingga generasi ke-10. Berikut ini hasil pengujian lebih lengkap

4.1. Hasil Pengujian Algoritma Genetika

Pada pengujian GA ini akan dilakukan dengan menggunakan 2 kondisi yaitu pengujian dengan batas 5 generasi maksimal dan pengujian dengan batas 10 generasi maksimal.



Gambar 3. Grafik fitness 5 generasi maksimal

Berdasarkan gambar 3 dapat dilihat bahwa pada pengujian dengan 5 generasi maksimal didapatkan nilai *fitness* yang berubah-ubah pada setiap generasinya. Hal ini dipengaruhi oleh karakteristik dan lama waktu hidup dari individu pada setiap generasinya. Semakin bagus karakteristik dan lama waktu hidup yang dimiliki maka akan memiliki lebih banyak kesempatan untuk melakukan serangan yang bernilai besar sehingga akan berpeluang menghasilkan *fitness* yang lebih

tinggi. Sebaliknya, semakin kurang bagus karakteristik dan sedikit waktu hidup yang dimiliki maka akan memiliki lebih sedikit kesempatan untuk melakukan serangan yang bernilai besar sehingga akan sangat sulit untuk menghasilkan *fitness* yang lebih tinggi.

Gambar 4. Grafik perbandingan individu awal dengan individu hasil GA pada 5 generasi maksimal

Untuk mencapai generasi maksimal diperlukan waktu bermain sekitar 30 menit. Didapatkan individu dengan nilai *fitness* tertinggi sebesar 60,34 pada generasi ke-4 yang mana individu ini merupakan individu terbaik yang dipilih sebagai hasil dari GA. Jika individu hasil GA dibandingkan dengan individu terbaik pada generasi awalnya akan didapatkan hasil seperti pada grafik gambar 3.

Pada gambar 4 dapat dilihat bahwa karakteristik dari individu awal cenderung lebih baik dibandingkan karakteristik dari individu hasil GA, di mana nilai HP dan SPD milik individu awal memiliki nilai yang lebih tinggi dibandingkan dengan milik individu hasil GA. Hal ini menandakan bahwa NPC hasil dari GA dengan 5 generasi maksimal memiliki karakteristik yang lebih buruk dibandingkan dengan NPC pada generasi awal. Yang membuat *fitness* individu hasil GA lebih buruk dibandingkan dengan individu awal adalah individu awal memiliki waktu rata-rata pergerakan lebih tinggi dibandingkan individu hasil GA untuk bergerak dari satu gedung ke gedung lainnya di mana hal ini dipengaruhi oleh lamanya waktu hidup NPC selama permainan dan gerakan NPC yang random.

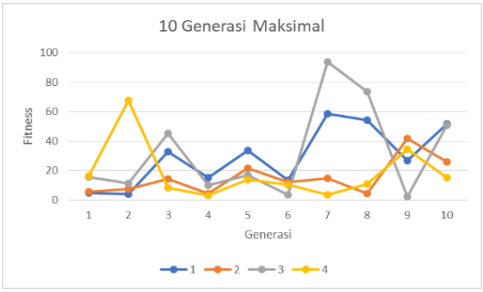
Pengujian ini dilakukan sebanyak 3 kali dan didapatkan 3 individu terbaik untuk masing-masing percobaan sesuai tabel 3 berikut.

Tabel 8. Hasil 3 kali pengujian GA untuk 5 generasi maksimal

Percobaan	Generasi terbaik	Kromosom				*Fitness*
		Health Point (HP)	*Attack* (ATK)	*Attack Speed* (ASPD)	*Movement Speed* (SPD)	
1	4	72	13	1	5	60,34
2	2	88	11	2	10	73,66
3	2	51	14	1	25	134,87

Mengacu data pada tabel 9 dapat dilihat setelah dilakukan 3 kali percobaan GA untuk mendapatkan individu terbaik dalam 5 generasi maksimal, setiap percobaan mendapatkan nilai *fitness* yang cukup tinggi. Untuk karakteristik individu yang dihasilkan dari pengujian ini masih belum merata untuk setiap gennya. Dapat dilihat bahwa ada satu sampai dua gen yang mendapatkan nilai yang mendekati nilai minimal range dari gennya sehingga mempengaruhi setidaknya dua perilaku NPC yang akan dirasa cukup lemah.

Pada percobaan ke-3 didapatkan individu terbaik dengan nilai *fitness* yang paling tinggi di antara dua percobaan lainnya yaitu 134,87. Hal ini disebabkan karena terdapat dua gen yang memiliki nilai yang hampir dan bahkan mencapai batas maksimal dari range-nya. Perilaku yang didapatkan pada individu ini adalah kelincahan pergerakan yang lebih cepat dan kekuatan serangan yang lebih tinggi dimana hal ini dapat dilihat dari nilai gen SPD dan ATK individu yang mendekati dan bahkan mencapai nilai range maksimal dari masing-masing gen. Sedang untuk gen HP dan ASPD mendapat nilai yang mendekati range minimal dari masing-masing gen sehingga perilaku yang didapatkan oleh individu ini adalah pertahanan NPC yang lebih tipis sehingga membuatnya mudah untuk dikalahkan dan durasi menyerang yang dibutuhkan oleh NPC terasa cukup lama untuk mendapat serangan yang besar.



Gambar 5. Grafik fitness 10 generasi maksimal

Berdasarkan gambar 5 dapat dilihat bahwa pada pengujian dengan 10 generasi maksimal didapatkan nilai *fitness* yang berubah-ubah pada setiap generasinya, sama seperti pada pengujian 5 generasi maksimal. Hal ini dipengaruhi oleh karakteristik dan lama waktu hidup dari masing-masing individu pada setiap generasinya.

Untuk mencapai generasi maksimal diperlukan waktu bermain sekitar 50 - 60 menit. Didapatkan individu dengan nilai *fitness* tertinggi sebesar 93,89 pada generasi ke-7 yang mana individu ini merupakan individu terbaik yang dipilih sebagai hasil dari GA. Jika individu hasil GA ini dibandingkan individu terbaik pada generasi awalnya akan didapatkan hasil seperti pada grafik gambar 5 berikut.

Gambar 6. Grafik perbandingan individu awal dengan individu hasil GA pada 10 generasi maksimal

Dari gambar 6 dapat dilihat bahwa karakteristik dari individu hasil GA cenderung lebih baik dibandingkan karakteristik dari individu awal, dimana nilai *HP*, *ASPD* dan SPD milik individu hasil GA memiliki nilai yang lebih tinggi dibandingkan dengan milik individu awal. Hal ini menandakan bahwa NPC hasil dari GA dengan 10 generasi maksimal memiliki memiliki karakteristik yang lebih baik dibandingkan dengan NPC pada generasi awal.

Pengujian ini dilakukan sebanyak 3 kali kali dan didapatkan 3 individu terbaik untuk masing-masing percobaan sesuai tabel 5 berikut.

Tabel 10. Hasil 3 kali pengujian GA untuk 10 generasi maksimal

Percobaan	Generasi terbaik	Kromosom				*Fitness*
		Health Point (HP)	*Attack* (ATK)	*Attack Speed* (ASPD)	*Movement Speed* (SPD)	
1	7	86	12	2	20	93,89
2	10	79	11	2	10	80,06
3	3	51	13	2	25	128,64

Berdasarkan data pada tabel 11 dapat dilihat setelah dilakukan 3 kali percobaan GA untuk mendapatkan individu terbaik dalam 10 generasi maksimal, setiap percobaan mendapatkan nilai *fitness* yang cukup tinggi. Untuk karakteristik individu yang dihasilkan dari pengujian ini masih belum merata sepenuhnya untuk setiap gennya. Namun hasil perataan gennya sedikit lebih baik dari pada saat hanya menggunakan 5 generasi maksimal. Dapat dilihat bahwa setidaknya hanya ada satu gen yang mendapatkan nilai yang sangat mendekati nilai minimal range dari gennya sehingga masih ada setidaknya hanya satu perilaku NPC yang akan dirasa paling lemah.

Pada percobaan ke-3 didapatkan individu terbaik dengan nilai *fitness* yang paling tinggi di antara dua percobaan lainnya yaitu 128,64. Hal ini disebabkan karena terdapat dua gen yang memiliki nilai yang mencapai batas maksimal dari range-nya. Karakteristik yang didapatkan pada individu ini adalah kelincahan pergerakan yang lebih cepat dan kecepatan serangan yang sangat tinggi dengan nilai serangan yang tidak terlalu lemah. Hal ini dapat dilihat pada nilai gen SPD dan ASPD yang mencapai nilai range maksimal dari masing-masing gen dan nilai gen ATK yang berada ditengah range dari gennya. Sedang untuk gen HP mendapat nilai yang sangat mendekati range minimal dari gennya sehingga perilaku yang didapatkan oleh individu ini adalah pertahanan NPC yang lebih tipis sehingga membuatnya mudah untuk dikalahkan.

4.2. Tampilan Antarmuka *Game*

Pada gambar 7 merupakan tampilan dari scene menu awal dimana scene ini akan muncul pertama kali saat *game* dibuka. Pada scene ini terdapat logo dari *game Fire Force Tower* dan diikuti dengan 2 tombol di bawahnya yaitu tombol "Play" untuk memulai permainan dan "Quit" untuk keluar dari *game*.

Gambar 7. Tampilan scene menu awal

Gambar 8. Tampilan scene gameplay

Untuk gambar 8 merupakan tampilan dari scene *game*play atau tampilan saat pemain akan mulai memainkan *game*nya. Pada scene ini terdapat beberapa fitur yaitu :

a. Pada bagian kiri atas layar terdapat HP dan *Water* bar untuk indikator nilai ketahanan pemain dan banyaknya air yang ada ditas pemain. Kemudian dibawahnya terdapat panel skor untuk memperlihatkan skor yang diperoleh oleh pemain selama bermain dan terdapat beberapa Icon yang menunjukkan situasi pada lingkungan permainan.

b. Pada bagian kanan atas layar terdapat tombol pause yang digunakan untuk menghentikan permainan untuk sementara.
c. Pada bagian kiri bawah layar terdapat 4 tombol navigasi untuk menggerakan karakter utama sesuai arahnya.
d. Pada bagian kanan bawah layar terdapat 2 tombol aksi. Tombol pertama yang paling kanan adalah tombol serangan (*Splash*) yang digunakan untuk *player* menembakkan peluru airnya. Kemudian tombol di sebelah kirinya digunakan untuk mengisi ulang tas air *player* (*Refill*) jika tas airnya kosong.

5. KESIMPULAN

Setelah melakukan beberapa kali percobaan untuk pengujian GA pada *game* ini, untuk pengujian yang menggunakan 5 generasi maksimal mendapatkan nilai *fitness* 134,87 dengan hasil 2 gen yang mendapatkan nilai minimal, sedangkan untuk pengujian yang menggunakan 10 generasi maksimal mendapatkan hasil *fitness* 128,64 dengan hasil hanya 1 gen yang mendapatkan nilai minimal. Ketika dibandingkan dengan generasi awal pada setiap percobaannya maka akan didapatkan bahwa generasi hasil GA cenderung akan mendapatkan karakteristik lebih baik dari individu generasi awal. Dari penelitian ini didapatkan bahwa implementasi GA pada karakteristik atau perilaku NPC bisa dilakukan dan mampu berjalan dengan lancar serta pada generasi maksimal yang lebih banyak mampu menghasilkan NPC dengan perilaku yang lebih baik dan memiliki karakteristik yang cukup merata namun dengan waktu perhitungan yang lebih lama.

DAFTAR PUSTAKA

[1] N. Iman, "Pembangunan Aplikasi Game Hybrid Shooter Side-Scrolling Destroyer Garuda Berbasis Dekstop," Universitas Komputer Indonesia, 2013.

[2] W. Pratama, "Game Adventure Misteri Kotak Pandora," *J. Telemat.*, vol. 7, no. 2, pp. 13–31, 2016.

[3] B. S. Ginting and F. Ramadhan, "Perancangan Game Become a King Berbasis," *Manaj. Inform. Komputerisasi Akunt.*, vol. 2, no. 1, pp. 12–21, 2018.

[4] C.-H. Kim, S.-M. Jeong, G.-T. Hur, and B.-G. Kim, "Verification of FSM using Attributes Definition of NPCs Models," *IJCSNS Int. J. Comput. Sci. Netw. Secur.*, vol. 6, no. 7, pp. 168–174, 2006.

[5] I. Millington and J. Funge, *Artificial Intelligence for Games*. San Francisco: CRC Press, 2018. doi: 10.1201/9781315375229.

[6] D. Nugroho, Siswanto, "Pembuatan Npc Dalam Simulator Game 'Sang Pedjoeang' Dengan Implementasi Artificial Intelligence Mdp (Markov Decision Process)," Universitas Muhammadiyah Malang, 2013. [Online]. Available: https://www.ptonline.com/articles/how-to-get-better-mfi-results

[7] W. F. Mahmudy, "Penerapan algoritma genetika pada optimasi model penugasan," *Natural*, vol. 10, no. 3, pp. 197–207, 2006, [Online]. Available: https://www.researchgate.net/publication/280698033_Penerapan_algoritma_genetika_pada_optimasi_model_penugasan

[8] Y. M. Arif, A. Wicaksono, and F. Kurniawan, "Pergantian Senjata NPC pada Game FPS Menggunakan Fuzzy Sugeno," *Semin. Nas. Compet. Advant. 2012*, vol. 1, no. 2, 2012, [Online]. Available: https://media.neliti.com/media/publications/175887-ID-pergantian-senjata-npc-pada-game-fps-men.pdf

[9] M. Negnevitsky, *Artificial Intelligence*, 3rd ed. Pearson Education : Harlow., 2011, 2011.

[10] V. Andriyawan and A. Qoiriah, "PERANCANGAN DAN PEMBUATAN WORD GAME SCRAMBLE DENGAN ALGORITMA GENETIKA," *J. Manaj. Inform.*, vol. 02, no. 2, pp. 29–36, 2013.

[11] D. Whitley, "A genetic algorithm tutorial," *Stat. Comput.*, vol. 4, no. 2, Jun. 1994, doi: 10.1007/BF00175354.

[12] I. M. S. Putra, "Penerapan Algoritma Genetika Dan Implementasi Dalam MATLAB," vol. 53, no. 9, pp. 1689–1699, 2018.

[13] N. Widyastuti and A. Hamzah, "PENGGUNAAN ALGORITMA GENETIKA DALAM PENINGKATAN KINERJA FUZZY CLUSTERING UNTUK (Application of Genetic Algorithm to Enhance the Performance of Clustering," *Berk. MIPA*, vol. 17, no. 2, pp. 1–14, 2007.

[14] H. Kenedy Tupan, R. N. Hasanah, and W. Wijono, "Optimasi Penempatan Load Break Switch (LBS) pada Penyulang Karpan 2 Ambon menggunakan Metode Algoritma Genetika," *Electr. Electron. Commun. Control. Informatics Syst.*, vol. 11, no. 2, pp. 1–8, 2017, doi: 10.21776/ub.eeccis.2017.011.01.1.

[15] J. Pranata and K. R. Purba, "Aplikasi Game Strategi Menggunakan Metode Algoritma Genetika untuk Pembuatan Pasukan," *J. Infra*, vol. 4, no. 1, pp. 199–204, 2016.

[16] P. Y. Utami, C. Suhery, and Ilhamsyah, "Aplikasi Pencarian Rute Terpendek Menggunakan Algoritma Genetika (Studi Kasus: Pencarian Rute Terpendek Untuk Pemadam Kebakaran Di Wilayah Kota Pontianak)," *J. Coding Sist. Komput. Univ. Tanjungpura*, vol. 02, no. 1, pp. 19–25, 2014.

[17] A. Hannawati, "Pencarian Rute Optimum Menggunakan Algoritma Genetika," *J. Tek. Elektro*, vol. 2, no. 2, pp. 78–83, 2004, doi: 10.9744/jte.2.2.

[18] J. Suryaputra, C. Lubis, and T. Sutrisno, "Pemilihan Crossover pada Algoritma Genetika Untuk Program Aplikasi Pengenalan Karakter Tulisan Tangan," *J. Ilmu Komput. dan Sist. Inf.*, vol. 6, no. 1, pp. 69–72, 2018, [Online]. Available: https://journal.untar.ac.id/index.php/jiksi

[19] B. N. Satrijo *et al.*, "Library Algoritma Genetik dan Whale Optimization berbasis GPU Programming," vol. 04, no. 01, pp. 32–44, 2022, doi: 10.52985/insyst.v4i1.208.

[20] R. Bimantika, "Pengembangan Game the Galaxy Menggunakan Metode Fsm (Finite State Machine)," *J. Mhs. Tek. Inform.*, vol. 1, no. 1, pp. 180–187, 2017.

[21] Purba, K.R. (2015). Optimization of Auto Equip Function in Role-Playing Game Based on Standard Deviation of Character's Stats Using Genetic Algorithm. In: Intan, R., Chi, CH., Palit, H., Santoso, L. (eds) Intelligence in the Era of Big Data. ICSIIT 2015. Communications in Computer and Information Science, vol 516. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-662-46742-8_6

[22] H. Chen, Y. Mori and I. Matsuba, "Solving the Balance Problem of On-Line Role-Playing Games Using Evolutionary Algorithms," *Journal of Software Engineering and Applications*, Vol. 5 No. 8, 2012, pp. 574-582. doi: 10.4236/jsea.2012.58066.