IMPLEMENTASI ALGORITMA GENETIKA UNTUK NILAI KEKUATAN *NPC* PADA *GAME* *"FUN ENGLISH"*

SKRIPSI

Oleh:
GINANJAR GALANG MAHARDHIKA
NIM. 12650044

JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2017

IMPLEMENTASI ALGORITMA GENETIKA UNTUK NILAI KEKUATAN *NPC* PADA *GAME* *"FUN ENGLISH"*

SKRIPSI

Oleh:
GINANJAR GALANG MAHARDHIKA
NIM. 12650044

JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2017

IMPLEMENTASI ALGORITMA GENETIKA UNTUK NILAI KEKUATAN *NPC* PADA *GAME* *"FUN ENGLISH"*

SKRIPSI

Diajukan Kepada:
Fakultas Sains dan Teknologi
Universitas Islam Negeri (UIN)
Maulana Malik Ibrahim Malang
Untuk Memenuhi Salah Satu Persyaratan Dalam
Memperoleh Gelar Sarjana Komputer (S.Kom)

Oleh :

Ginanjar Galang Mahardhika
NIM. 12650044

JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK BRAHIM MALANG
2017

LEMBAR PERSETUJUAN

IMPLEMENTASI ALGORITMA GENETIKA UNTUK UNTUK NILAI KEKUATAN *NPC* PADA *GAME* *"FUN ENGLISH"*

SKRIPSI

Oleh:
Ginanjar Galang Mahardhika
NIM 12650044

Telah Diperiksa dan Disetujui untuk Diuji :
Tanggal : 10 Maret 2017

Pembimbing I	Pembimbing II
Hani Nurhayati, M.T	Fresy Nugroho, M.T
NIP. 19780625 200801 2 006	NIP. 19710722 201 101 1 001

Mengetahui,
Ketua Jurusan Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Cahyo Crysdian
NIP. 19740424 200901 1 008

HALAMAN PENGESAHAN

IMPLEMENTASI ALGORITMA GENETIKA UNTUK UNTUK NILAI KEKUATAN *NPC* PADA *GAME* *"FUN ENGLISH"*

SKRIPSI

Oleh :

Ginanjar Galang Mahardhika

NIM 12650044

Telah Dipertahankan di Depan Dewan Penguji Skripsi dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan Untuk Memperoleh Gelar Sarjana Komputer (S.Kom)

Tanggal 10 Maret 2017

Susunan Dewan Penguji			Tanda Tangan
1. Penguji Utama	:	Yunifa Miftachul Arif, M.T NIP. 19830616 201101 1 004	( )
2. Ketua	:	Fachrul Kurniawan, M.MT NIP. 19771020 200901 1 001	( )
3. Sekretaris	:	Hani Nurhayati, M.T NIP. 19780625 200801 2 006	( )
4. Anggota	:	Fresy Nugroho, M.T NIP. 19710722 201101 1 001	( )

Mengetahui dan Mengesahkan
Ketua Jurusan Teknik Informatika

Dr. Cahyo Crysdian
NIP. 19740424 200901 1 008

iv

HALAMAN PERNYATAAN

ORISINALITAS PENELITIAN

Saya yang bertanda tangan di bawah ini:

Nama	:	Ginanjar Galang Mahardhika
NIM	:	12650044
Fakultas / Jurusan	:	Sains dan Teknologi / Teknik Informatika
Judul Penelitian	:	Implementasi Algoritma Genetika Untuk Nilai Kekuatan *NPC* Pada Game *"FUN ENGLISH"*

Menyatakan dengan sebenar-benarnya bahwa hasil penelitian saya ini tidak terdapat unsur-unsur penjiplakan karya penelitian atau karya ilmiah yang pernah dilakukan atau dibuat oleh orang lain, kecuali yang secara tertulis dikutip dalam naskah ini dan disebutkan dalam sumber kutipan dan daftar pustaka. Apabila ternyata hasil penelitian ini terbukti terdapat unsur-unsur jiplakan, maka saya bersedia untuk mempertanggung jawabkan, serta diproses sesuai peraturan yang berlaku.

Malang, 14 Maret 2017

Yang Membuat Pernyataan,

Ginanjar Galang Mahardhika

12650044

HALAMAN MOTO

Hidup terlalu singkat untuk menjadi orang lain

HALAMAN PERSEMBAHAN

Bismillah, dengan rasa syukur Alhamdulillah ku persembahkan
karya tulis ini untuk :

Ayahanda dan ibunda Tercinta

Ir. Setyo Tri Harnanto** dan **Suprihatin

*Yang selalu memberi semangat, nasehat dan do'a. Untuk adik **M daffa Syahputra**, dan semua keluarga besarku yang juga selalu memberi nasehat dan semangat saat pengerjaan, untuk semua dosen yang ikhlas memberi ilmunya selama masa studi, dan juga tak lupa teman mahasiswa/i teknik Informatika UIN Maliki yang selalu saling memberi semangat, saling mengingatkan ketika lalai, dan bersama-samaberjuang dalam menyelesaikan skripsi.*

Jazakumullah ahsanal jaza'

KATA PENGANTAR

Assalamu'alaikum Wr. Wb.

Segala puji bagi Allah SWT tuhan semesta alam, karena atas segala rahmat dan karunia-Nya sehingga penulis mampu menyelesaikan skripsi dengan judul "Implementasi Algoritma Gentika Untuk Nilai Kekuatan NPC Pada *Game "Fun English"* " dengan baik dan lancar. Shalawat serta salam selalu tercurah kepada tauladan terbaik Nabi Agung Muhammad SAW yang telah membimbing umatnya dari zaman kebodohan menuju Islam yang *rahmatan lil alamiin*.

Dalam penyelesaian skripsi ini, banyak pihak yang telah memberikan bantuan baik secara moril, nasihat dan semangat maupun materiil. Atas segala bantuan yang telah diberikan, penulis ingin menyampaikan doa dan ucapan terimakasih yang sedalam-dalamnya kepada :

1. Prof. DR. H. Mudjia Raharjo, M.Si, selaku rektor UIN Maulana Malik Ibrahim Malang beserta seluruh staf. Bakti Bapak dan Ibu sekalian terhadap UIN Maliki Malang turut membesarkan dan mencerdaskan penulis.
2. Dr. Hj. Bayyinatul M., drh., M.Si, selaku Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang beserta seluruh staf. Bapak dan ibu sekalian sangat berjasa memupuk dan menumbuhkan semangat untuk maju kepada penulis.
3. Bapak Dr. Cahyo Crysdian, selaku Ketua Jurusan Teknik Informatika Universitas Islam Negeri Maulana Malik Ibrahim Malang, yang sudah memberi banyak pengetahuan, inspirasi dan pengalaman yang berharga.
4. Ibu Hani Nurhayati, M.T selaku dosen pembimbing I yang telah meluangkan waktu untuk membimbing, memotivasi, mengarahkan dan memberi masukan kepada penulis dalam pengerjaan skripsi ini hingga akhir.

5. Bapak Fresy Nugroho, M.T, selaku dosen pembimbing II yang juga senantiasa memberi masukan dan nasihat serta petunjuk dalam penyusunan skripsi ini.
6. Ayah, Ibu, dan Adik serta keluarga besar tercinta yang selalu memberi dukungan yang tak terhingga serta doa yang senantiasa mengiringi setiap langkah penulis.
7. Segenap Dosen Teknik Informatika yang telah memberikan bimbingan keilmuan kepada penulis selama masa studi.
8. Teman – teman seperjuangan Teknik Informatika 2012
9. Para peneliti yang telah mengembangkan *Game* dengan *Engine Unity3d* yang menjadi acuan penulis dalam pembuatan skripsi ini. Serta semua pihak yang telah membantu yang tidak bisa disebutkan satu satu. Terimakasih banyak.

Berbagai kekurangan dan kesalahan mungkin pembaca temukan dalam penulisan skripsi ini, untuk itu penulis menerima segala kritik dan saran yang membangun dari pembaca sekalian. Semoga apa yang menjadi kekurangan bisa disempurnakan oleh peneliti selanjutnya dan semoga karya ini senantiasa dapat memberi manfaat. Amin. *Wassalamualaikum Wr. Wb.*

Malang, 14 Maret 2017

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

ABSTRAK

Mahardhika, Ginanjar Galang. 2016. ***IMPLEMENTASI ALGORITMA GENETIKA UNTUK NILAI KEKUATAN NPC PADA GAME "FUN ENGLISH"*** Skripsi. Jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.

Pembimbing : (I) Hani Nurhayati, MT, (II) Fresy Nugroho, M.T

Kata Kunci: *NPC, Permainan, Algoritma Genetika*

Bahasa inggris merupakan bahasa internasional dimana kita dituntut untuk menggunakannya ketika berkomunikasi dengan negara lain. Pembelajaran bahasa inggris tentu sangat dibutuhkan mengingat peringkat indonesia dalam kecakapan ber bahasa inggris tidak begitu baik dibandingkan negara tetangga. Pada masa sekarang ini model-model pembelajaran telah berkembang pesat, salah satunya adalah model pembelajaran di dalam *game.*

Game merupakan salah satu sarana untuk pembelajaran dimana dengan menggunakan media *game* akan menjadikan pembelajaran lebih menarik, dikarenakan terdapat konten-konten yang unik pada setiap *game* untuk mendukung iklim belajar.

Penelitian ini menjelaskan bagaimana merancang metode Algoritma Genetika dalam *NPC* pada suatu *game* dan juga menjelaskan bagaimana membangun *survivale game* yang bernuansa edukasi. *Fun english* adalah permainan edukasi berjenis survival berbasis *desktop* yang dibangun dengan memanfaatkan Unity

3D. Dimana pemain dituntut untuk bertahan dari serangan-serangan musuh dan menjawab soal ketika mengambil *item heal*.

Implementasi kecerdasan buatan pada penelitian ini diterapkan pada *NPC* dengan memanfaatkan Algoritma Genetika. Algoritma Genetika digunakan sebagai pembangkit nilai kekuatan *NPC*. Penelitian ini difokuskan pada *platform desktop*.

ABSTRAK

Mahardhika, Ginanjar Galang. 2016. ***IMPLEMENTASI ALGORITMA GENETIKA UNTUK NILAI KEKUATAN NPC PADA GAME "FUN ENGLISH"*** Thesis. Informatics Engineering Department of Science and Technology Faculty Islamic State University Maulana Malik Ibrahim Malang.

Supervisor: (I) Hani Nurhayati, MT, (II) Fresy Nugroho, M.T

Keywords: *NPC, Game, Genetic Algorithm*

People often talk about English as a global language or lingua franca. With more than 350 million people around the world speaking English as a first language and more than 430 million speaking it as a second language, there are English speakers in most countries around the world. According to English Proficiency Index for Companies (EF EPI-c), Indonesia skill in english is ranked 32 of 72 countries under Malaysia, the Philippines, and Vietnam.

In order to improve English language skills we must provide an interesting way to learn English. Game is one of the entertainment media are widely popular. Unique content is one of the attractions to be able to support the learning. This paper describes how to design the evolution of NPCs power in a game and also explains how to build educational survival game. Fun English is educational adventure game based desktop that is built by using Unity3D. Player have to survive through enemy wave to gain more score.

Implementation of artificial intelligence in this study applied to the NPC by utilizing the Genetic Algorithm(GA). GA is used as a method of generating NPC power. This study focused on desktop platforms.

ملخص

ماهارديكا، غالانغ غينانجار. تنفيذ خوارزميات وراثية 2016. قيمة سلطة الشخصيات في لعبة أطروحة "متعة الإنجليزية". وكان قسم الهندسة المعلوماتية للعلوم والتكنولوجيا كلية لإسلامية الدولة جامعة مولانا مالك إبراهيم .المؤسفة

.المشرف: (ط) هاني نورهاياتي، طن متري، نوغروهو فريسي (ثانيا)، م. ت

اللغة الإنجليزية هي اللغة العالمية التي نحن مطالبون استخدامها عند التواصل مع الدول الأخرى. تعلم الإنجليزية هو مطلوب بالتأكيد من ذلك بكثير نظرا اندونيسيا في المرتبة الكفاءة الهواء في اللغة الإنجليزية ليست جيدة جدا بالمقارنة مع الدول المجاورة. في نماذج التعلم الوقت الحاضر قد تنمو بسرعة، واحدة منها هي نموذج للتعلم في اللعبة.

اللعبة هي واحدة يعني لتعلم أن استخدام وسائل الاعلام لعبة وجعل التعلم أكثر إثارة للاهتمام، لأن هناك المحتويات التي هي فريدة من نوعها في كل مباراة لدعم مناخ التعلم.

من أجل تحسين مهارات اللغة الإنجليزية يجب أن نوفر وسيلة ممتعة لتعلم اللغة الإنجليزية. اللعبة هي واحدة من وسائل الإعلام والترفيه تحظى بشعبية على نطاق واسع. محتوى فريد هو واحد من عوامل الجذب لتكون قادرة على دعم التعلم. وتصف هذه الورقة كيفية تصميم تطور قوة الشخصيات في اللعبة وأيضا يوضح كيفية بناء لعبة البقاء على قيد الحياة التعليمية. متعة الإنجليزية هو سطح المكتب على أساس التعليمية لعبة المغامرة التي تم إنشاؤها باستخدام Unity3D. لاعب يجب أن البقاء على قيد الحياة من خلال موجة العدو لكسب المزيد من النقاط.

BAB I

PENDAHULUAN

1.1 Latar Belakang

Bahasa Inggris merupakan alat untuk berkomunikasi secara lisan dan tulisan. Pengertian berkomunikasi dimaksudkan adalah memahami dan mengungkapkan informasi, pikiran, perasaan serta mengembangkan ilmu pengetahuan, teknologi, dan budaya dengan menggunakan bahasa Inggris. Kemampuan berkomunikasi dalam pengertian yang utuh adalah kemampuan berwacana dalam bahasa Inggris pada tingkat literasi tertentu (Depdiknas, 2003:13).

Dalam kurikulum 2013 dikatakan bahwa pelajaran bahasa inggris dirancang untuk menyongsong model pembelajaran abad ke-21. Di dalamnya terdapat pergeseran pembelajaran dari siswa diberitahu menjadi siswa mencari tahu dari berbagai sumber belajar melampaui batas guru dan satuan pendidikan. Peran bahasa Inggris dalam model pembelajaran seperti itu menjadi sangat sentral mengingat lebih banyak sumber belajar yang menggunakan bahasa Inggris dibandingkan bahasa lainnya.

Dalam proses belajar mengajar bahasa inggris *grammar* memiliki peran yang sangat penting untuk mendukung keempat ketrampilan berbahasa. Diasumsikan bahwa pembelajaran dan pengajaran yang banyak berfokus pada makna atau komunikasi tidaklah cukup untuk mencapai kompetensi kebahasaan. Menurut Nunan (1991) pembelajar bahasa dan guru bahasa adalah mengajarkan *grammar* dan mengoreksi kesalahan pembelajar. Bahkan menurut LittleWood

(1983) bahwa pengajaran yang menggunakan pendekatan komunikatif , *Grammar* dianggap sebagai *language usage* yang merupakan fondasi untuk berkomunikasi.

Menurut lembaga yang mengindeks peringkat bahasa inggris di seluruh dunia *EF English Proficiency Index for Companies (EF EPI-c),* Indonesia berada di peringkat 32 dari 72 negara di bawah Malaysia, Filiphina, dan Vietnam. Sehingga peningkatan kemampuan berbahasa Inggris pada tingkat sekolah maupun masyarakat umum perlu ditingkatkan karena bahasa Inggris adalah bahasa global.

Permainan atau *game* adalah suatu struktur kegiatan, yang biasanya dilakukan untuk kesenangan dan kadang-kadang digunakan sebagai sarana pendidikan. Permainan berbeda dengan pekerjaan, yang biasanya dilakukan untuk mendapatkan suatu upah tertentu, atau dengan seni, yang lebih peduli dengan ekspresi ide. Namun ada kalanya perbedaan itu menjadi tidak jelas, banyak permainan dinilai sebagai pekerjaan (seperti pemain olahraga profesional) atau sebagai seni karena melibatkan *layout* artistik. Komponen kunci dari permainan adalah tujuan, aturan, tantangan, dan interaksi.

Game Komputer merupakan program komputer yang memungkinkan pemain dapat berinteraksi lewat kontrol untuk memenuhi tujuan tertentu. *Game* komputer banyak dimainkan oleh berbagai kalangan dari anak-anak sampai orang tua. Selain memberikan kesenangan kepada pemain, *game* juga dapat menjadi sarana untuk melatih kecerdasan, kemampuan menyelesaikan masalah, kemampuan menganalisa dan dapat memberikan informasi atau wawasan yang

ada dalam *game* tersebut. Telah dikemukakan bahwa pengembangan teknologi informasi tidak hanya mempengaruhi bentuk bermain, tetapi juga membawa *game* sebagai sebuah konsep lebih jelas ke dalam kehidupan anak-anak dimasa yang akan datang (Sheridan S, Pramling-Samuelsson I.2003). *Game* edukasi lebih unggul dalam beberapa aspek jika dibandingkan dengan metode pembelajaran konvensional. Salah satu keunggulan yang signifikan adalah adanya animasi yang dapat meningkatkan data ingat sehingga anak dapat menyimpan materi pembelajaran dalam waktu yang lebih lama dibandingkan dengan metode pembelajaran konvensional. (Clark, 2006).

Holland (1975) menunjukan bahwa algoritma genetika dapat memainkan peran sebagai sistem adaptif melalui reproduksi buatan dan evolusi. Algoritma genetika diimplementasikan sebagai simulasi komputer dimana sebuah populasi dari representasi abstrak (disebut kromosom, *genotipe*, atau *genom*) kandidat solusi disebut (individu, makhluk, atau *fenotipe*). Biasanya, solusi direpresentasikan dalam biner sebagai string yang terdisi dari 0 dan 1 namun, penggunaan metode *encoding* lain juga mungkin dipakai. Berbeda dengan teknik pencarian konvensional, algoritma genetika bermula dari himpunan solusi yang di hasilkan secara acak. Kromoson-kromosom berevolusi dalam suatu proses iterasi yang berkelanjutan yang disebut generasi. Pada setiap generasi, kromosom dievaluasi berdasarkan suatu fungsi evaluasi. Setelah beberapa generasi maka algoritma genetika akan konvergen pada kromosom terbaik, yang di harapkan merupakan solusi optimal.

Sesuai dengan permasalahan yang dikemukakan di atas, peneliti akan membangun aplikasi *game* pembelajaran bahasa inggris dengan menggunakan

algoritma genetika pada NPC untuk mengoptimalkan kepuasan pemain, diharapkan dengan *game* ini dapat membantu dalam belajar mengenali, memahami, dan mengerti bahasa inggris dengan baik dan benar.

1.2 Identifikasi Masalah

1. Bagaimana menerapkan Algoritma genetika untuk menetapkan model nilai-nilai kekuatan NPC pada *game "Fun English"*?.
2. Seberapa optimal peningkatan *fitness* yang terjadi setiap generasi dari NPC pada *game* "*Fun English*"?.

1.3 Batasan Masalah

Batasan masalah pada penelitian ini sebagai berikut:

1. Aplikasi ini di bangun sebagai aplikasi berbasis desktop.
2. Soal Yang diujikan merupakan soal *"Structure"* yang diambil dari buku"*Practice Exercise for the TOEFL 6th edition*" .
3. *Game* bergenre *survival* yang bersifat *single player*.

1.4 Tujuan Penelitian

1. Mengimplementasikan algoritma genetika untuk desain NPC pada *game* "*Fun English* ".
2. Mengukur tingkat keoptimalan algoritma genetika di lihat dari *fitness* yang muncul tiap-tiap generasi.

1.5 Manfaat Penelitian

Penelitian ini di tujukan untuk menerapkan *dynamic difficulty adjustment* menggunakan algoritma genetika dimana akan ada evolusi kekuatan NPC secara

bertingkat, yang diharapkan akan meningkatkan *interestingness* dari pemain yang nantinya juga akan meningkatkan kepuasan pemain dalam memainkan sebuah *game*. Sehingga *game* yang dibangun tidak akan membosankan ketika *game* itu terlalu mudah dan tidak akan membuat *player* frustasi ketika *game* tersebut terlalu sulit.

BAB II

TINJAUAN PUSTAKA

2.1 Penelitian Terkait

Omid E.David, et al, 2014, penelitian yang mendemonstrasikan *game* catur dengan menggunakan algoritma genetika. Penelitian tersebut mengembangkan algortima genetika dengan fungsi 1). Evaluasi pemain tingkat *grandmaster* 2).Mekanisme pencarian pada *game* catur, nilai parameter yang dipakai diinisialisasi secara acak. Fungsi evaluasi dipelajari dari *knowledge* database pemain tingkat *grandmaster*. Hasil penelitian menunjukan bahwa program berevolusi melebihi juara komputer catur dunia.

Nirvana S. Antonio, et al, 2013 penelitian tentang fungsi evaluasi untuk langkah terbaik pada *game* Domino menggunakan algoritma genetika (GA). Fungsi evaluasi terdiri dari penggabungan strategi *game* didefinisikan sebagai penekanan ke lawan, dan gerak adaptif ke lawan. Hasil yang diperoleh selama simulasi menunjukan bahwa tim (terdiri dari dua pemain) menggunakan fungsi evaluasi dengan koefisien dioptimalkan oleh (GA) menang lebih dari 69,18 % dari total pertandingan.

Dini Nur Setyorini, et al, 2012 dalam penelitian mereka menyimpulkan hasil perhitungan nilai *fitness* untuk waktu terbaik atau penyelesaian terbaik kurang akurat. Proses algoritma genetika sangatlah berpengaruh pada hasil *random* iterasi. Apabila dilakukan percobaan ulang, hasil yang diperoleh akan lebih baik atau lebih buruk. Berdasarkan percobaan dan perbandingan hasil optimasi GA dengan perhitungan manual yang nyata dilapangan, sistem ini

terbukti dapat menyelesaikan masalah sehingga didapatkan waktu terbaik dan optimal.

Moshe Sipper, et al, 2008, dalam penelitiannya merumuskan pemrograman genetika bertindak sebagai alat untuk mengembangkan strategi dan desain mesin untuk penyusun strategi. Hasilnya menyatakan pemrograman genetika pada permainan telah berevolusi dengan kecerdasan buatan, menghasilkan strategi yang kompetitif menyaingi logika manusia, bahkan seringkali menang terhadap manusia.

Hans K.G. Fernlund et al, 2006 dalam penelitiannya membahas perilaku rekan tim pada *game* simulasi militer dengan algoritma GA. Peneliti memberikan pendekatan baru yang mempekerjakan pemrograman genetika dalam hubungannya dengan penalaran berbasis konteks perkembangan agen taktis berdasarkan observasi otomatis manusia melakukan misi di simulator. Kesimpulan dari hasil penelitian tersebut adalam program genetika mampu menciptakan agen baru dengan mempelajari performa, taktik dan strategi manusia pada simulsi tersebut

Francisco Azuaje, 2003, dalam penelitiannya mengangkat tema pendekatan komputasi evolusioner pada strategi *game* dan kerjasama makhluk *virtual*. Penelitian ini menggunakan algoritma genetika dimana akan dicari satu per satu individu-individu yang memiliki nilai *fitness* kompetitif. Satu individu mengimplementasikan aturan *game* didasarkan dari seperangkat kemampuan kognitif. Individu juga menentukan pola strategi *game* yang menarik.

Utomo Sarjono Putro, et al, 2000, penelitian tentang pembelajaran adaptif untuk mendukung kelompok pembuat keputusan dengan seperangkat strategi dan preferensi yang menghadapi perilaku yang tidak menentu. Menggambarkan situasi keputusan sebagai situasi *hypergame*, dimana setiap pengmabil keputusan secara explicit diasumsikan memiliki kesalahan persepsi tentang set strategi alam. GA dipakai untuk mengoptimalkan persepsi terdapat 3 prosedur belajar yang berbeda satu sama lain sehubungan seperti evaluasi *fitness*, *cross-over* yang dimodifikasi, dan pilihan tindakan strategi.

Dalam penelitian Sayed Fachrurrazi, 2012, menyebutkan algoritma genetika untuk penyelesaian masalah pencarian jarak terpendek di katakan bahwa dapat mengatasi masalah. Optimasi ditekankan pada pencarian jalur terpendek, dimana kana dicari beberapa alternatif solusi penyelesaian yang lebih efektif dan efisien. Algoritma genetika terbukti dapat memberi solusi jalur yang lebih baik.

Dalam penelitian Johan Saputra, 2008, et al, menggunakan algoritma genetika mendapat rata-rata tingkat keberhasilaan untuk keseluruhan pengujian adalah 70,80%. Tingkat nilai *fitness* cost dipengaruhi oleh jumlah data-data yang diproses dalam perangkat lunak. Metode cukup baik untuk digunakan dalam objek penelitiannya terlebih lagi prosentase keberhasilan sangat tinggi.

Fathelalem F. Ali, et al, 2000, penelitiannya menjelaskan tentang strategi NPC pad *game* Rock-Paper-Scissor. NPC bisa membuat keputusan, dan aturan diadopsi dari *game* dalam beberapa putaran terakhir, setelah itu diamati dan kemudian pengetahuan sementara dibuat. Proses penyusunan strategi seperti pengkodean ke *string* genetika dan algoritma genetika (GA) bekerja pada populasi

string tersebut. *String* yang memiliki nilai *fitness* diproduksi di generasi berikutnya. Kesimpulan penelitian ini adalah algoritma genetika dapat mengubah *game* NPC sama baiknya dengan *game* manusia.

2.2 Algoritma genetika

2.2.1 Pengertian

Algoritma genetika adalah algoritma pencarian heuristik yang didasarkan atas mekanisme evolusi biologis. Keberagaman pada evolusi biologis adalah variasi dari kromosom antar individu organisme. Variasi kromosom ini akan mempengaruhi laju reproduksi dan tingkat kemampuan organisme untuk tetap hidup. Algoritma genetika pertama kali dikembangkan oleh John Holland dari universitas Michigan (1975). John Holland mengatakan bahwa setiap masalah yang berbentuk adaptasi (alasmi maupun buatan) dapat diformulasikan dalam terminologi genetika. Algoritma adalah simulasi dari proses evolusi Darwin dan operasi genetika atas kromosom (Kusumadewi, 2003: 279).

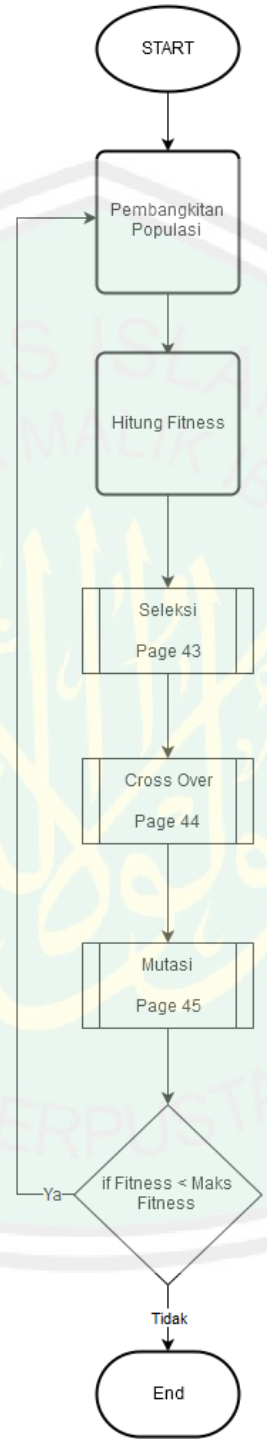
2.2.2 Struktur Umum Algortitma Genetika

Sebuah solusi yang dibangkitkan dalam algoritma genetika disebut sebagai ***chromosome***, sedangkan kumpulan chromosome-chromosome tersebut disebut sebagai ***populasi.*** Sebuah chromosome dibentuk dari komponen-komponen penyusun yang disebut sebagai ***gen*** dan nilainya dapat berupa bilangan numerik, biner, simbol ataupun karakter tergantung dari permasalahan yang ingin disebelsaikan. *Chromosome-chromosome* tersebut akan berevolusi secara berkelanjutan yang disebut ***generasi.*** Dalam tiap generasi *chromosome-chromosome* tersebut dievaluasi tingkat keberhasilan nilai solusinya terhadap

masalah yang ingin disebelsaikan (***fungsi_objektif***) menggunakan ukuran yang disebut ***fitness***. Untuk memilih *chromosome* yang tetap dipertahankan untuk generasi selanjutnya dilakukan proses yang disebut ***seleksi.*** Proses seleksi *chromosome* menggunakan konsep aturan evolusi Darwin yang telah disebutkan sebelumnya yaitu *chromosome* yang mempunyai nilai *fitness* tinggi akan memiliki peluang lebih besar untuk terpilih lagi pada generasi selanjutnya.

Chromosome-chromosome baru yang disebut dengan ***offspring***, dibentuk dengan cara melakukan perkawinan antar *chromosome-chromosome* dalam satu generasi yang disebut sebagai proses ***cross-over.*** Jumlah *chromosome* dalam populasi yang mengalami ***cross-over*** ditentukan oleh parameter yang disebut dengan ***cross-over rate***. Mekanisme perubahan susunan untuk penyusun makhluk hidup akibat adanya faktor alam yang disebut dengan ***mutasi*** direpresentasikan sebagai proses berubahnya satu atau lebih nilai gen dalam *chromosome* dengan suatu nilai acak. Jumlah gen dalam populasi yang mengalami mutasi ditentukan oleh parameter ***mutation-rate.*** Setelah beberapa generasi akan dihasilkan *chromosome-chromosome* yang nilai gen-gennya konvergen ke suatu nilai tertentu yang merupakan solusi terbaik yang dihasilkan oleh algoritma genetika terhadap permasalahan yang ingin diselesaikan.

Berikut flowchart secara umum algoritma Genetika:



Gambar 2.1 Flowchart Algoritma Genetika

2.2.3 Komponen-Komponen Utama Algortitma Genetika

Terdapat 6 komponen-komponen dalam Algoritma Genetika menurut Kusumadewi, 2003: 280-283 yaitu :

a. Teknik Pengkodean

Teknik pengkodean disini meliputi pengkodean gen dari kromosom. Gen merupakan bagian dari kromosom. Satu gen biasanya akan mewakili satu variabel. Gen dapat direpresentasikan dalam bentuk:string bit, pohon, array bilangan real, daftar aturan, elemen permutasi, elemen program, atau representasi lainnya yang dapat diimplementasikan untuk operator genetika. Demikian juga kromosom dapat direpresentasikan dengan menggunakan:

String bit	:10011:01101,11101,dst
Bilangan real	:65.65,-67.98,562.88,dst
Elemen permutasi	:E2,E10,E5,dst
Daftar aturan	:R1,R2,R3.dst
Elemen program	:pemrograman genetika

Berikut adalah beberapa jenis pengkodean yang umum digunakan

a. Pengkodean Biner
b. Pengkodean Permutasi
c. Pengkodean Nilai
d. Pengkodean Pohon

b. Membangkitkan Populasi Awal

Membangkitkan populasi awal adalah proses membangkitkan sejumlah individu secara acak atau melalui prosedur tertentu. Ukuran untuk populasi tergantung pada masalah yang akan diselesaikan dan jenis operator genetika yang akan diimplementasikan. Setelah ukuran populasi ditentukan, kemudian dilakukan pembangkitan populasi awal. Syarat yang harus dipenuhi untuk menunjukan suatu solusi harus benar-benar diperhatikan dalam pembangkitan setiap individu. Teknik dalam pembangkitan populasi awal ini ada beberapa cara, diantaranya sebagai berikut:

1. *Random* Generator

Inti dari cara ini adalah melibatkan pembangkitan bilangan *random* untuk nilai setiap gen sesuai dengan representasi kromosom yang digunakan. Jika merupakan representasi biner, salahsatu contoh penggunaan *random generator* adalah penggunaan rumus berikut untuk pembangkitan populasi awal:

$$IPOP = round\{random(N_{IPOP}, N_{bits})\}$$

Dimana IPOP adalah gen yang nantinya berisi pembulatan dari bilangan *random* yang dibangkitkan sebanyak N_{IPOP}(Jumlah populasi) X N_{bits} (Jumlah Gen dalam tiap kromosom).

2. Pendekatan Tertentu (Memasukan Nilai Tertentu ke Dalam Gen)

Cara ini adalah dengan memasukan nilai tertentu ke dalam gen dari populasi awal yang dibentuk.

3. Permutasi Gen

Salah satu cara permutasi gen dalam pembangkitan populasi awal adalah penggunaan permutasi josephus dalam permasalahan kombinatorial seperti TSP, misalkan ada kota dari 1 sampai 9, permutasi dari lintasan dapat dilakukan dengan menentukan titik awal dan selan. Misalnya titik awal adalah 6 dan selang adalah 5, maka lintasan berangkat dari kota 6, selang dari 6 kota adalah kota 2 (dengan asumsi kota 1 sampai kota 9 membentuk *circular list*). Kota 2 dihapus dari list. Selang 5 kemudian adalah kota 7. Proses ini duilang hingga ada satu lintasan dalam *list*. Hasil dari permutasi ini adalah 2-7-3-8-4-9-5-1-6

c. Prosedur Inisialisasi

Ukuran populasi tergantung pada masalah yang akan dipecahkan dan jenis operator genetika yang akan diimplementasikan. Setelah ukuran populasi ditentukan, kemudian harus dilakukan inisialisasi terhadap kromosom yang terdapat pada populasi tersebut. Inisialisasi kromosom dilakukan secara acak, namun demikian harus tetap memperhatikan domain solusi dan kendala permasalahan yang ada.

d. Fungsi Evaluasi

Ada 2 hal yang harus dilakukan dalam melakukan evaluasi kromosom, yaitu: evaluasi dengan fungsi objektif (fungsi tujuan) dan konversi fungsi objektif dengan nilai yang tidak negatif. Apabila ternyata fungsi objektif memiliki nilai negatif, maka perlu ditambahkan satu konstanta C agar nilai *fitness* yang terbentuk menjadi tidak negatif.

e. Seleksi

Seleksi bertujuan memberikan kesempatan reproduksi yang lebih besar bagi anggota populasi yang paling fit (Anita dan Muhammad, 2006:193). Seleksi akan menentukan individu-individu mana saja yang akan dipilih untuk dilakukan rekombinasi dan bagaimana offspring terbentuk dari individu-individu terpilih tersebut. Langkah pertama yang dilakukan dalam seleksi ini adalah pencarian nilai *fitness*. Masing-masing individu dalam wadah seleksi akan menerima probabilitas reproduksi yang tergantung pada nilai objektifnya sendiri terhadap nilai objektif dari semua individu dalam wadah seleksi tersebut. Nilai *fitness* inilah yang nantinya akan digunakan pada tahap-tahap seleksi berikutnya(kusumadewi dan Hari, 2005: 235).

- ***Rank-based Fitness***

  Populasi diurutkan menurut nilai objektifnya. Nilai *fitness* dari tiap-tiap individu hanya tergantung pada posisi individu tersebut dalam urutan, dan tidak dipengaruhi oleh nilai objektifnya.

- **Seleksi Roda Roulette**

  Istilah lain dari roda metode seleksi ini adalah *stocastic sampling with replacement*. Individu dipetakan dalam suatu segmen garis secara berurutan sedemikian hingga tiap-tiap segmen individu memiliki ukuran yang sama dengan ukuran *fitness*nya. Sebuah bilangan *random* dibangkitkan dan individu yang memiliki segmen dalam kawasan bilangan *random* tersebut akan terseleksi. Proses ini berulang hingga didapatkan sejumlah individu yang diharapkan.

- ***Stocastic Universal Sampling***

  Stocastic Universal Sampling memiliki nilai bias nol dan penyebaran yang minimum. Individu-individu dipetakan dalam suatu segmen garis secara berurut sedemikian hingga tiap-tiap segmen individu memiliki ukuran yang sama dengan ukuran *fitness*nya seperti halnya pada seleksi roda roulette.

- **Seleksi Lokal**

  Setiap individu yang berada di dalam konstrain tertentu disebut dengan nama lingkungan lokal. Interaksi antar individu hanya dilakukan di dalam wilayah tersebut. Lingkungan tersebut ditetapkan sebagai struktur dimana populasi tersebut terdistribusi. Lingkungan tersebut juga dapat dipandang sebagai kelompok pasangan-pasangan yang potensial.

- **Seleksi dengan pemotongan**

  Seleksi dengan pemotongan merupakan seleksi buatan yang digunakan oleh populasi yang jumlahnya sangat besar. Individu-individu diurutkan berdasarkan nilai *fitness*nya. Hanya individu terbaik saja yang akan diseleksi sebagai induk.

- **Seleksi dengan Turnamen**

  Seleksi dengan Turnamen adalah seleksi dengan menetapkan suatu nilai tour untuk individu-individu yang dipilih secara *random* dari suatu populasi. Individu-individu yang terbaik dalam kelompok ini akan

diseleksi sebagai induk. Parameter yang digunakan adalah ukuran tour yang bernilai antara 2 sampai N (jumlah individu dalam populasi).

f. Operator Genetika

Ada 2 operator Genetika yaitu: perkawinan silang (*crossover*) dan mutasi

1. *Crossover*

Crossover (perkawinan silang) adalah operator genetik yang utama. Operator bekerja dengan mengambil dua individu dan memotong *string* kromosom mereka pada posisi yang terpilih secara acak, untuk memproduksi dua segmen *head* dan dua segmen *tail* (Son, 2007:185), *Crossover* bertujuan menambah keanekaragaman string dalam satu produksi dengan penyilangan antar string yang diperoleh dari reproduksi sebelumnya (Anita dan Muhammad,2006:196).

- ***Crossover* satu titik**

  Crossover satu titik dan banyak titik biasanya digunakan untuk representasi kromosom dalam biner. Pada *crossover* satu titik, posisi *crossover* k (k=1,2...,N-1) dengan N=panjang kromosom diseleksi secara *random*. Variabel-variabel ditukar antar kromosom pada titik tersebut untuk menghasilkan anak.

- ***Crossover* Aritmatika**

  Crossover aritmatika digunakan untuk representasi berupa bilangan float (pecahan). *Crossover* ini dilakukan dengan menentukan nilai r sebagai bilangan *random* lebih dari 0 dan kurang dari 1. Selain itu juga ditentukan posisi dari gen yang dilakukan *crossover* menggunakan bilangan *random*.

- ***Crossover* untuk Representasi Kromosom Permutasi**

- *Partial-Mapped Crossover (PMX)*

Prosedur PMX

Langkah 1:Pilih posisi untuk menentukan *substring* secara acak

1	2	3	4	5	6	7	8	9

Induk 1:

5	4	6	9	2	1	7	8	3

Induk 2:

Langkah 2: tukar dua *substring* antar induk untuk menghasilkan *protochild*.

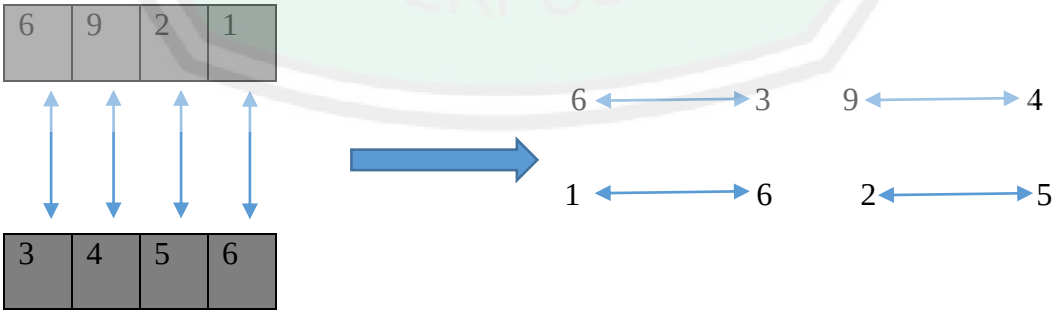
1	2	6	9	2	1	7	8	9

Proto-child 1:

5	4	3	4	5	6	7	8	3

Proto-child 2:

Langkah 3: tentukan hubungan pemetaan diantara dua daerah pemetaan.



Langkah 4: tentukan kromosom keturunan mengacu pada hubungan pemetaan.

3	5	3	4	5	6	7	8	4

Keturunan 1:

2	9	6	9	2	1	7	8	1

Keturunan 2:

- *Order Crossover (OX)*

Prosedur OX:

Langkah 1: pilih *substring* dari sebuah induk secara acak

1	2	3	4	5	6	7	8	9

Induk 1:

5	4	6	9	2	1	7	8	3

Induk 2:

Langkah 2: bangkitkan sebuah *Proto-child* dengan mengosongkan tempat Substring induk 2 pada induk 1

		3	4	5	6	7	8	

Proto-child 1:

		6	9	2	1	7	8	

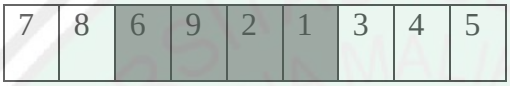
Proto-child 2:

Langkah 3: SHR *allele* dari *substring* antara 2 induk

Proto-child 1:

Proto-child 2:

Langkah 4: tukar substring antara 2 induk



Keturunan 1:

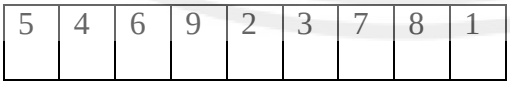
Keturunan 2:

- *Cycle Crossover (CX)*

Prosedur CX

Langkah 1: tentukan pola *cycle*

Induk 1:



Induk 2:

Pola cycle 1 → 5 → 2 → 4 → 9 → 1

Langkah 2:*copy* nilai dalam *cycle* pada *Proto-child*

1	2		4	5				9

Proto-child:

Langkah 3: tentukan nilai yang di ingat dari induk yang lain

5	4	6	9	2	3	7	8	1

Induk 2:

Nilai yang

6	3	7	8

tidak ada

Langkah 4: mengisi nilai yang diingat dalam keturunan

1	2	3	4	5	6	7	8	9

Keturunan 1:

Dengan cara yang sama memperoleh keturunan 2

5	4	3	9	2	6	7	8	1

Keturunan 2:

2. Mutasi

Mutasi merupakan proses mengubah nilai dari satu atau beberapa gen dalam suatu kromosom (Anita dan Muhammad,2006: 197). Mutasi ini berperan untuk menggantikan gen yang hilang dari populasi akibat proses seleksi yang memungkinkan munculnya kembali gen yang tidak muncul pada inisialisasi populasi (Kusumadewi, 2003: 296). Mutasi diterapkan dengan probabilitas sangat kecil. Peluang mutasi (Pm) mengendalikan banyaknya gen baru yang akan dimuncukan untuk dievaluasi, Jika peluang mutasi terlalu kecil, banyak gen yang

mungkin tidak pernah dievaluasi. Tetapi bila peluang mutasi terlalu besar, maka akan terlalu banyak gangguan acak, sehingga anak akan kehilangan kemiripan dari induknya.

- **Mutasi Pengkodean Biner**

  Mutasi pengkodean biner merupakan operasi yang sangat sederhana. Proses mtuasi pengkodean biner dilakukan dengan cara mengitervensi nilai bit pada kromosom yang terpilih secar acak (atau menggunakan skema tertentu) dengan diubah nilainya menjadi nilai lawannya (0 ke 1 atau 1 ke 0). Sebagai contoh, dapat dilihat pada tabel berikut ini :

Tabel 2.1 Mutasi Pengkodean Biner

Keadaan Kromosom	Proses Mutasi
Kromosom sebelum mutasi	1000111110110110
Kromosom sesudah mutasi	1001110110110

- **Mutasi Pengkodean Nilai**

  Mutasi pengkodean nilai adalah proses yang terjadi pada saat pengkodean nilai. Proses mutasi dalam pengkodean nilai dapat dilakukan dengan cara memilih sembarang posisi gen pada kromosom, dan nilai yang ada kemudian ditambahkan atau dikurangkan dengan suatu nilai kecil tertentu yang diambil secara acak. Sebagai contoh, dapat dilihat pada tabel berikut ini, yaitu nilai riil ditambahkan dan dikurangkan dengan nilai 0 dan 1.

Keadaan Kromosom	Proses Mutasi

Kromosom sebelum mutasi	1,45 2,67 1,87 2,56
Kromosom sesudah mutasi	1,55 2,67 1,77 2,56

Tabel 2.2 Mutasi Pengkodean Nilai

- **Mutasi Pengkodean Permutasi**

  Proses mutasi pengkodean permutasi tidak sama halnya dengan proses mutasi yang dilakukan pada pengkodean biner dengan mengubah langsung bit-bit pada kromosom. Salah satu cara yang dapat dilakukan adalah dengan memilih dua posisi (*locus*) dari kromosom dan kemudian nilainya saling dipertukarkan. Orang tua yang berada di bawah titik *crossover* dipertukarkan untuk menghasilkan anak baru. Contoh mutasi pada pengkodean permutasi, dapat dilihat pada tabel di bawah ini:

Tabel 2.3 Mutasi Pengkodean Permutasi

Keadaan Kromosom	Proses Mutasi
Kromosom sebelum mutasi	1 2 3 4 5 6 7 8 9
Kromosom sesudah mutasi	1 2 7 4 6 5 8 3 9

g. Penentuan Parameter

Yang disebut dengan parameter di sini adalah parameter kontrol algoritma genetika, yaitu ukuran populasi (popsize), peluang *crossover* (Pc), dan peluang mutasi (Pm). Nilai parameter ini ditentukan juga berdasarkan permasalahan yang

akan dipecahkan. Ada beberapa rekomendasi yang bisa digunakan, antara lain (Kusumadewi, 2003: 283):

- Untuk permasalahan yang memiliki kawasan solusi cukup besar, De jong merekomendasikan untuk nilai parameter kontrol :

  (Popsize;Pc;Pm) = (30;0.6;0.001)

- Nila rata-rata *fitness* setiap generasi digunakan sebagai indikator, maka Grefensette merekomendasikan

  (Popsize;Pc;Pm) = (30;0.95;0.01)

- Bila *fitness* dari individu terbaik dipantau pada setiap generasi maka usulannya adalah :

  (Popsize;Pc;Pm) = (80;0.45;0.01)

- Ukuran populasi sebaiknya tidak lebih kecil dari 30, untuk sembarang jenis permasalahan

3. *Game*

Menurut kamus bahasa indonesia *game* memiliki arti yaitu permainan. Permainan adalah kegiatan yang di dalamnya terdapat beberapa peraturan. Dalam permainan terdapat sebuah konflik buatan yang dibuat untuk pemain sehingga dapat berinteraksi dengan sistem dan konflik tersebut. Sebuah *game* memiliki sebuah tujuan untuk menyelesaikan masalah/konflik yang dibuat. *Game* adalah

lingkungan pelatihan yang baik bagi dunia nyata dalam organisasi yang menuntut pemecahan masalah secara kolaborasi.

Permainan terdiri atas sekumpulan peraturan yang membangun situasi bersaing dari dua sampai beberapa orang atau kelompok dengan memilih strategi yang dibangun untuk memaksimalkan kemenangan sendiri ataupun untuk meminimalkan kemenangan lawan. Peraturan-peraturan mennetukan kemungkinan tindakan untuk setiap pemain, sejumlah setiap pemain sebagai kemajuan bermain, dan sejumlah kemenangan atau kekalahan dalam berbagai situasi, (Neuman,2007).

Pengertian *game* menurut beberapa ahli:

1. Menurut Clark C.Abt *Game* adalah kegiatan yang melibatkan keputusan pemain, berupaya mencapai tujuan dengan dibatasi oleh konteks tertentu, misal oleh aturan.
2. Menurut Greg Costikyan, *Game* adalah sebentuk karya seni dimana peserta, yang disebut *player* membuat keputusan untuk mengelola sumberdaya yang dimilikinya melalui benda di dalam *game* dalam mencapai tujuan.

Game ditinjau dari *platform* yang digunakan:

1. *Arcade game*s

Arcade games atau di indonesia sering disebut ding-dong, biasanya berada di daerah/tempat khusus dan memiliki *box* atau mesin yang memang khusus di desain untuk jenis *video games* tertentu dan tidak jarang bahkan memiliki fitur yang dapat membuat pemainnya lebih merasa “masuk” dan “menikmati”, seperti pistol, kursi khusus, sensor injakan dan stir mobil.

2. PC *Game*s

PC Games adalah permainan yang dimainkan menggunakan *Personal Computer*.

3. *Console Game*s

Console Games adalah permainan yang dimainkan menggunakan *console* tertentu, seperti Playstation,Xbox 360, dan Nintendo Wii.

4. *Handheld Game*s

Handheld Games adalah *game* yang dapat dimainkan menggunakan *console* khusus yang dapat dibawa kemana-mana, contoh Nintendo DS dan Sony PSP.

1. Mobile *Game*s

   Yaitu *game* yang dapat dimainkan khusus untuk *mobile phone* atau PDA.

   Berikut adalah *game* yang ditinjau dari segi genre permainannya:

2. *Fighting* (pertarungan)

   Game fighting adalah *game* yang memerlukan kecepatan refleks dan koordinasi mata-tangan, tetapi inti dari *game* ini adalah penguasaan jurus (hafal caranya dan lancar mengeksekusinya), pengenalan karakter dan *timing* sangatlah penting, *combo*-pun menjadi esensial untuk mengalahkan lawan secepat mungkin. Dan berbeda seperti *game* aksi pada umumnya yang hanya melawan *Artificial Intellegence* atau istilah umumnya melawa komputer saja, pemain jenis *fighting game* ini baru teruji kemampuan sesungguhnya dengan melawan pemain lainnya. Contoh

dari *game* genre ini adalah Street Fighter, Tekken, Mortal Kombat, Soul Calibur dan King of Fighter.

5. Petualangan

Video games murni petualangan lebih menekankan pada jalan cerita dan kemampuan berpikir pemain dan menganalisa tempat secara visual, memecahkan teka-teki maupun menyimpulkan rangkaian peristiwa dan percakapan karakter hingga penggunaan benda-benda pada tempat yang tepat. Termasuk di dalamnya:

a. Petualangan dengan teks atau sistem tunjuk dan klik, contoh: Kings Quest, Space Quest, Heroes Quest, Monkey Island, Sam and Max.
b. Novel atau film interaktif, seperti *game* "*dating*" yang banyak beredar di jepang, Dragons Lair dan Night trap.

6. *Role Playing*

Video games jenis ini sesuai dengan terjemahannya, bermain peran, memiliki penekanan pada tokoh/peran perwakilan pemain di dalam permainan, yang biasanya adalah tokoh utamanya, dimana seiring kita memainkannya, karakter tersebut dapat berubah dan berkembang ke arah yang di inginkan pemain (biasanya semakin hebat, semakin kuat, semakin berpengaruh, dll) dalam berbagai parameter yang biasanya ditentukan dengan naiknya level, baik dari status kepintaran, kecepatan dan kekuatan karakter, senjata yang semakin sakti, ataupun jumlah teman maupuan makhluk peliharaan. Secara kebudayaan, pengembang *game* jepang biasanya membuat *Role Playing Game* ke arah cerita linear yang diarahkan seolah karakter kita adalah tokoh dalam cerita itu, seperti Final Fantasy, Dragon Quest dan Xenogears. Sedangkan pengembang *game* RPG eropa,

cenderung membuat karakter kita bebas memilih jalan cerita sendiri secara *non-linear*, seperti Ultima, Never Winter Nights, Baldurs gate, Elder Scroll, dan Fallout.

1. *Puzzle*

Video games jenis ini sesuai namanya adalah *game* yang mempunyai inti permainan memecahkan teka-teki, baik itu menyusun balok, menyamakan warna bola, memecahkan perhitungan matematika, melewati labirin, sampai mendorong kotak ke tempat yang seharusnya, itu semua termasuk dalam *game* bergenre *puzzle*. Sering pula unsur permainan jenis ini terdapat dalam permainan ber jenis petualangan maupun edukasi. Contoh dari permainan *puzzle* antara lain Tetris, Minesweeper, Bejeweled, Sokoban dan Bomberman.

2. Olahraga

Sudah sangat jelas bahwa genre permainan ini mempunyai konsep olahraga di dalam konsol maupun di *PC*. Gaya permainan diadopsi dari olahraga yang ada di dunia nyata, walaupun ada juga yang menambahkan unsur fiksi seperti pada *game* NBA JAM. Contoh dari *game* bergenre ini adalah Winning Eleven, seri NBA, seri FIFA.

3. Aksi – Petualangan

Memasuki gua bawah tanah, melompati bebatuan di antara lahar, bergelayutan dari pohon ke pohon lain, bergulat dengan ular sambil mencari kunci untuk membuka pintu kuil legendaris, atau sekedar mencari telpon umum untuk mendapatkan misi berikutnya, itulah beberapa dari banyak hal yang karakter permain harus lakukan dan lalui dalam *video games* jenis ini. Contoh dari *game*

ber genre aksi-petualangan antara lain Tomb Rider, Grand Theft Auto dan Prince of Persia.

4. Simulasi, Konstruksi dan Manajemen

Video games jenis ini sering kali menggambarkan dunia di dalamnya sedekat mungkin dengan dunia nyata dan memperhatikan dengan detail berbagai faktor yang terkait. Dari mencari jodoh dan pekerjaan, membangun rumah, gedung hingga kota, mengatur pajak dan dana kota hingga keputusan menambah atau memecat karyawan. Dunia kehidupan rumah tangga sampai bisnis membangun konglomerasi, dari jualan limun di pinggir jalan hingga membangun laboratorium kloning. *Video games* jenis ini juga membuat pemain berpikir untuk mendirikan, membangun dan mengatasi masalah dengan menggunakan dana yang terbatas, contohnya pada *game* Sim City, The Sims.

5. Strategi

Kebalikan dari *video games* jenis aksi yang berjalan dengan cepat dan perlu reflek yang baik, *video games* jenis strategi memerlukan keahlian berpikir dan menentukan setiap gerakan dengan hati-hati dan terencana, seperti pada permainan catur. *Video games* strategi biasanya memberikan pemain atas kendali tidak hanya satu orang tetapi minimal sekelompok orang dengan berbagai jenis tipe kemampuan, sampai kendaraan, bahkan hingga pembangunan berbagai bangunan, pabrik dan pusat pelatihan tempur, tergantung dari tema ceritanya. Pemain *game* strategi melihat dari sudut pandang lebih meluas dan lebih kedepan dengan waktu permainan yang biasanya lebih lama dan santai dibandingkan *game* action. Unsur-unsur permainannya biasanya berkisar sekitar, prioritas

pembangunan, peletakan pasukan, mencari dan memanfaatkan sumberdaya (uang, besi, kayu, minyak, dll), hingga ke pembelian dan memperkuat pasukan atau teknologi. *Game* jenis ini terbagi atas:

a. *Real time strategy*, *game* berjalan dalam waktu sebenarnya dan serentak antara semua pihak dan pemain harus memutuskan setiap langkah yang diambil saat itu juga berbarengan mungkin saat itu pihak lawan juga sedang mengeksekusi streateginya, Contoh: Starcraft, Warcraft, dan Comand and Conquer.

b. *Turn based strategy, game* yang berjalan secara bergiliran, saat kita mengambil keputusan dan menggerakan pasukan, saat itu pihak lawan menunggu, begitu pula sebaliknya, layaknya catur. Contoh Front Mission, Super robot wars, Final Fantasy tactics, Heroes of might and magic, Master of orion.

7. Simulasi Kendaraan

Video games jenis ini memberikan pengalaman atau interaktifitas sedekat mungkin dengan kendaraan yang aslinya, meskipun terkadang kendaraan tersebut masih eksperimen atau bahkan fiktif, tapi ada penekanan khusus pada detil dan pengalaman realistik menggunakan kendaraan tersebut. *Game* bergenre ini terbagi dalam beberapa jenis diantaranya:

a. Perang, Video *game*s simulasi kendaraan yang sempat tenar di tahun 90-an ini mengajak pemain untuk menaiki kendaraan dan berperang dengan kendaraan lainnya, dan kebanyakan diantaranya memiliki judul sama dengan nama kendaraannya. Contoh: Apache 64, Comandhe, Abrahams, YF-23, F-16.

b. Racing, Video games jenis ini memiliki objektif kemenangan berupa siapa pertamakali sampai ke garis finish. Pemain dapat memilih kendaraan, mendadani, *upgrade* mesin bahkan mengecat. Contoh: Top Gear, Test Drive, Need for Speed.

c. Luar Angkasa, Walau masih dapat dikategorikan sebagai simulai kendaraan perang, tetapi segala unsur fiksi ilmiah dan banyaknya judul yang beredar membuat subgenre ini pantas dikategorikan di luar simulasi kendaraan perang. Jenis ini memungkinkan pemain untuk menjelajah luar angkasa, berperang dengan alien, mendarat di planet atau sekedar ingin merasakan bagaimana menjadi kapten di film-film fiksi. Contoh : Wing Commander, Freelancer, Star Wars X-Wing.

d. *Mecha,* Pendapat bahwa hampir tidak ada orang yang terekspos oleh film robot jepang saat kecilnya tidak memimpikan ingin mengendalikan robot, memang sulit dibantah. Dipopulerkan oleh serial Mechwarrior oleh Activision, subgenre Simulasi Mecha ini memungkinkan pemainnya untuk mengendalikan robot dan menggunakannya untuk menghancurkan gedung, helikopter dan tentu saja robot lainnya. Contoh: Mechwarrior, Gundam Last War Chronicles, dan Armored Core.

8. Aksi – Shooting, *game* bergenre aksi-*shooting* sangat memerlukan kecepatan refleks, koordinasi mata-tangan, juga *timing*, inti dari *game* jenis ini adalah tembak menembak. *Game* aksi-*shooting* ini memiliki subgenre sebagai berikut:

a. *First person shooting (FPS)* seperti Counter Strike dan Call of Duty.
b. Drive n' shoot, menggunakan unsur kendaraan namun tetap dengan tujuan utama menembak dan menghancurkan lawan, contoh*: Spy Hunter*, *Rock and Roll Racing.*
c. Shoot em' up, seperti *raiden, 1942*, dan *gradius*

d. *Beat 'em up* seperti *Double Dragon* dan *Final Fight,* lalu *hack and slash* seperti *Shinobi* dan *Legend of Kage.*

e. *Light gun shooting, yang menggunakan alat yang umumnya berbentuk seperti senjata, seperti Virtual Cop dan Time Crisis.*

Menurut Teresa Dillon elemen-elemen dasar sebuah *game* adalah:

1. *Game Rule*

   Game rule merupakan aturan perintah, cara menjalankan, fungsi objek dan karakter di dunia *game*. Dunia *game* bisa berupa pulau, dunia khayal, dan tempat-tempat lain yang sejenis yang dipakai sebagai *setting* tempat dalam *game.*

2. Plot

   Plot berisi informasi tentnag hal-hal yang akan dilakukan oleh *player* dalam *game* atau perintah tentang hal yang harus dicapai dalam *game*.

3. Tema

   Dalam *game*, tema *game* lebih cenderung kepada genre *game*, yaitu berisi informasi mengenai jenis *game*

4. Karakter

   Pemain sebagai karakter utama maupun karakter yang lain yang memiliki sifat dan ciri tertentu.

5. Objek

   Hal yang digunakan pemain untuk memecahkan masalah, adakalanya pemain harus punya keahlian dan pengetahuan untuk bisa memainkannya.

6. *Text*, Grafik, dan *Sound*

   Game merupakan kombinasi dari media teks, grafik dan suara, meskipun ada beberapa *game* tidak menggunakan ketiganya.

7. Animasi

   Animasi ini selalu melekat pada dunia *game*, khususnya untuk gerakan karakter, properti dan objek.

8. *User Interface*

   Merupakan fitur-fitur yang mengkomunikasikan *user* dengan *game*.

Orang-orang telah bermain *game* di komputer selama komputer telah ada, dan beragam permainan telah ditemukan. *National Writting Project* (2011) memberikan penjelasan dalam bukunya Jane McGonigal "*Reality is Broken*", mengakatan bahwa permainan paling tidak memiliki empat atribut yaitu:

1. Tujuan (*goal*): Permainan jelas mendefinisikan tujuan bagi para pemain untuk mencapainya. Tujuan yang menantang itu penting namun tujuan tersebut harus dapat dicapai.
2. Aturan (*rule*): Permainan memiliki aturan yang harus diikuti oleh pemain. Aturan sering membuat tujuan yang dicapai menjadi lebih sulit, sehingga mengharuskan pemain menjadi lebih kreatif dalam memainkan *game*.
3. Arus Balik (*feedback*): Sebuah permainan harus memberitahu apa yang harus dilakukan seorang pemain.
4. Partisipasi Sukarela (*Voluntary Participation*): Aspek permainan meyiratkan penerimaan dari pemain dari tujuan, aturan, dan sistem *feedback.*

4. *Non-Playable Character*

Menurut Febrian bahri Adi, et al, (2010) *Non-Playable Character* (NPC) atau disebut juga agen adalah suatu entitas dalam *game* yang tidak dikendalikan secara langsung oleh pemain. NPC dikemndalikan secara langsung oleh komputer. NPC bisa berupa teman, musuh atau netral. NPC diinginkan dapat berperilaku cerdas layaknya manusia. Dia bisa mengindera lingkungan, berpikir, memilih aksi lalu bertindak sebagai respon atas perubahan pada linkungannya. Untuk dapat memperoleh perilaku cerdas dari NPC digunakan kecerdasan buatan atau artificial intelligence (AI). Pengunaan Ai pada NPC dilakukan dengan pemberian algoritma khusus sesuai dengan perilaku cerdas yang diharapkan. Pada adegan (*scene*) pertempuran dari *game computer*, prajurit pemain dan prajurit musuh merupakan contoh dari NPC. Perilaku seorang prajurit dalam medan pertempuran bervariasi mulai dari mengikuti pimpinan, menghindari halangan, berlari, berjalan, menjauhi musuh, bertarung, membantu teman dan lainnya.

BAB III

DESAIN DAN PERANCANGAN

3.1. Keterangan Umum *Game*

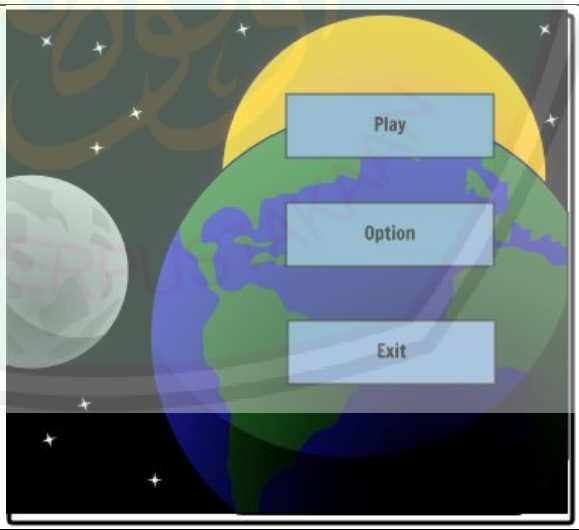
Game ini adalah *game* yang bergenre *Action-Survival Game* dan dimainkan secara *single player*. Pada *game* terdapat karakter sebagai pemain utama yang akan dijalankan oleh pengguna, karakter musuh yang merupakan karakter lawan akan dijalankan secara otomatis oleh komputer dengan algoritma genetika sebagai proses evolusi kekuatan NPC. Objek penelitian dalam permainan ini adalah algoritma genetika yang digunakan NPC untuk meningkatkan kekuatan NPC setiap berganti generasi.

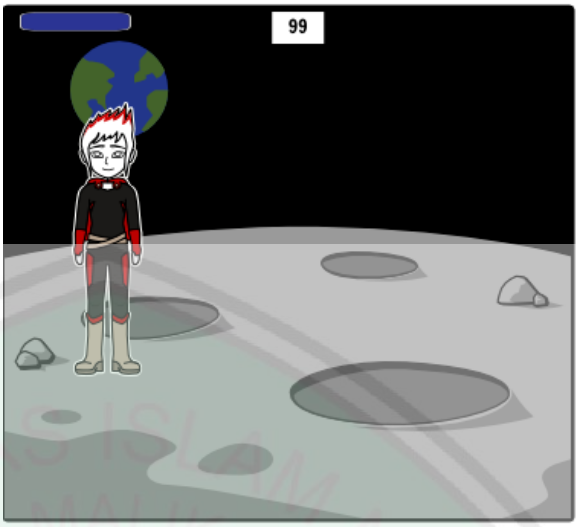
3.2 *Storyline*

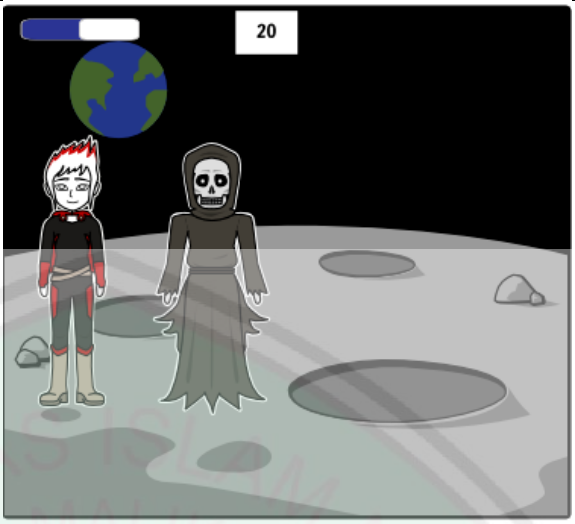
Game Fun English merupakan permainan edukasi bergenre *action-survival* yang berbasis *dekstop* sebagai media untuk pembelajaran bahasa inggris khusus untuk *structure. Game* ini tidak memiliki objektif untuk memenangkan permainan, namun ditekankan pada berapa lama pemain mampu bertahan di dalam *game*. Tantangan di dalam *game* ini ada dua yaitu dari soal yang diberikan pada waktu mengambil *item heal* dan dari NPC musuh. Di dalam *Game Fun English* ini ada dua kondisi dimana *player* akan mengalami kekalahan. Kondisi pertama adalah ketika *player* terbunuh oleh musuh, dan ketika *player* kehabisan waktu. Darah akan berkurang ketika di serang oleh musuh dan *player* dapat mengisi darah dengan mengambil *item heal*. Setelah *item* diambil maka akan keluar soal bahasa inggris dan jika *player* berhasil menjawab maka darah akan disembuhkan dalam jumlah tertentu, jika tidak maka *item heal* hilang dan darah

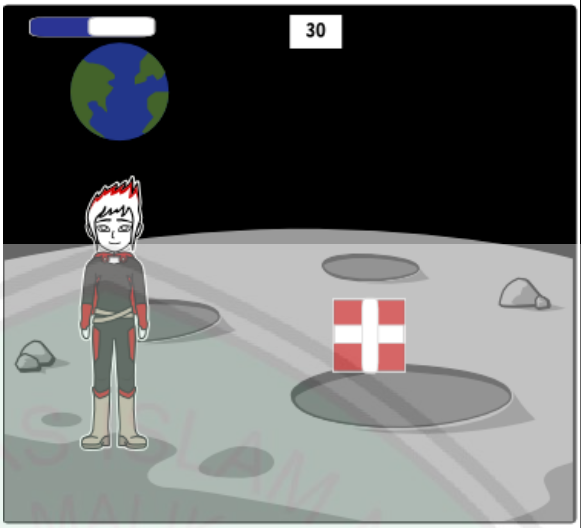
tidak akan disembuhkan. *Item heal* ini sendiri akan muncul di *map game*, tempat munculnya akan di acak oleh system dengan kurun waktu kemunculan setelah di ambil mempunyai rentang yang sama. Setelah itu pemain juga berhadapan dengan kesulitan yang lain yaitu waktu. Waktu akan di hitung mundur mulai dari awal permainan. Waktu akan bertambah hanya ketika *player* membunuh NPC musuh. Sehingga *player* tidak hanya berlari-lari menghindari musuh dan tidak mengambil *item*. Dengan demikian kemungkinan jumlah *item heal* digunakan lebih banyak dan penulis berharap akan lebih banyak soal yang akan dihadapi oleh *player* sehingga *player* menjadi lebih tertantang sekaligus teredukasi dalam hal bahasa inggris.

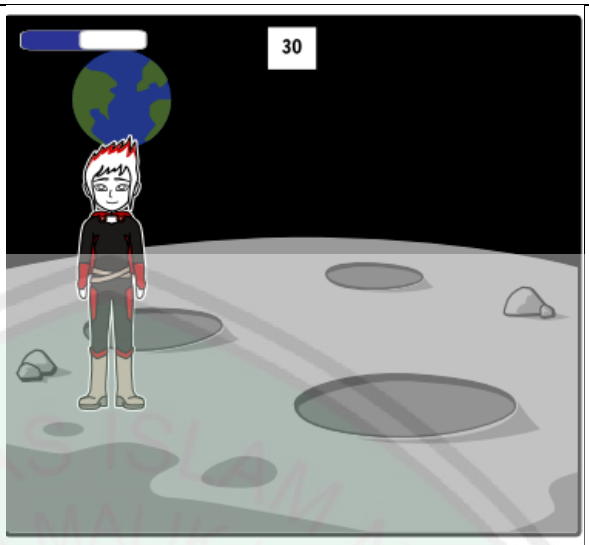
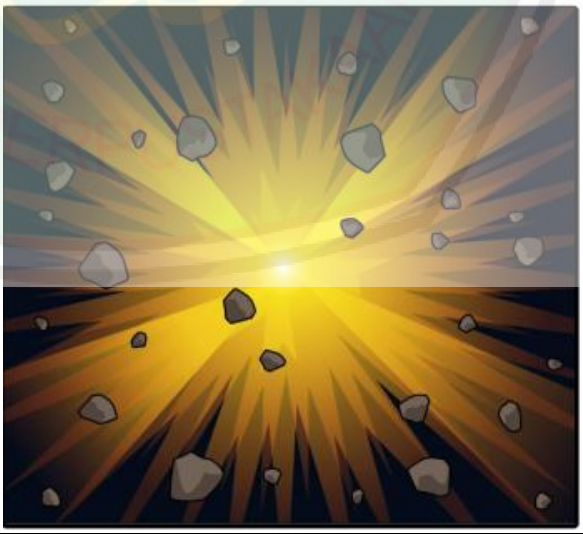
3.3 *Storyboard*

No	*Scene*	Subjek dan Object Terlibat	Gambar	Keterangan
1	*Main Menu*	Pemain		Setelah klik *icon Game* maka pemain akan dihadapkan dengan menu utama

2	Layar Awal	Pemain		*Player* masuk ke dalam *game*
3	Musuh terlihat	Pemain Karakter Musuh		Musuh terlihat dan *player* mendekati musuh untuk menyerang
4	Jarak serang	Pemain Karakter Musuh		NPC musuh masuk kedalam jarak serang dan *player* mulai menyerang.

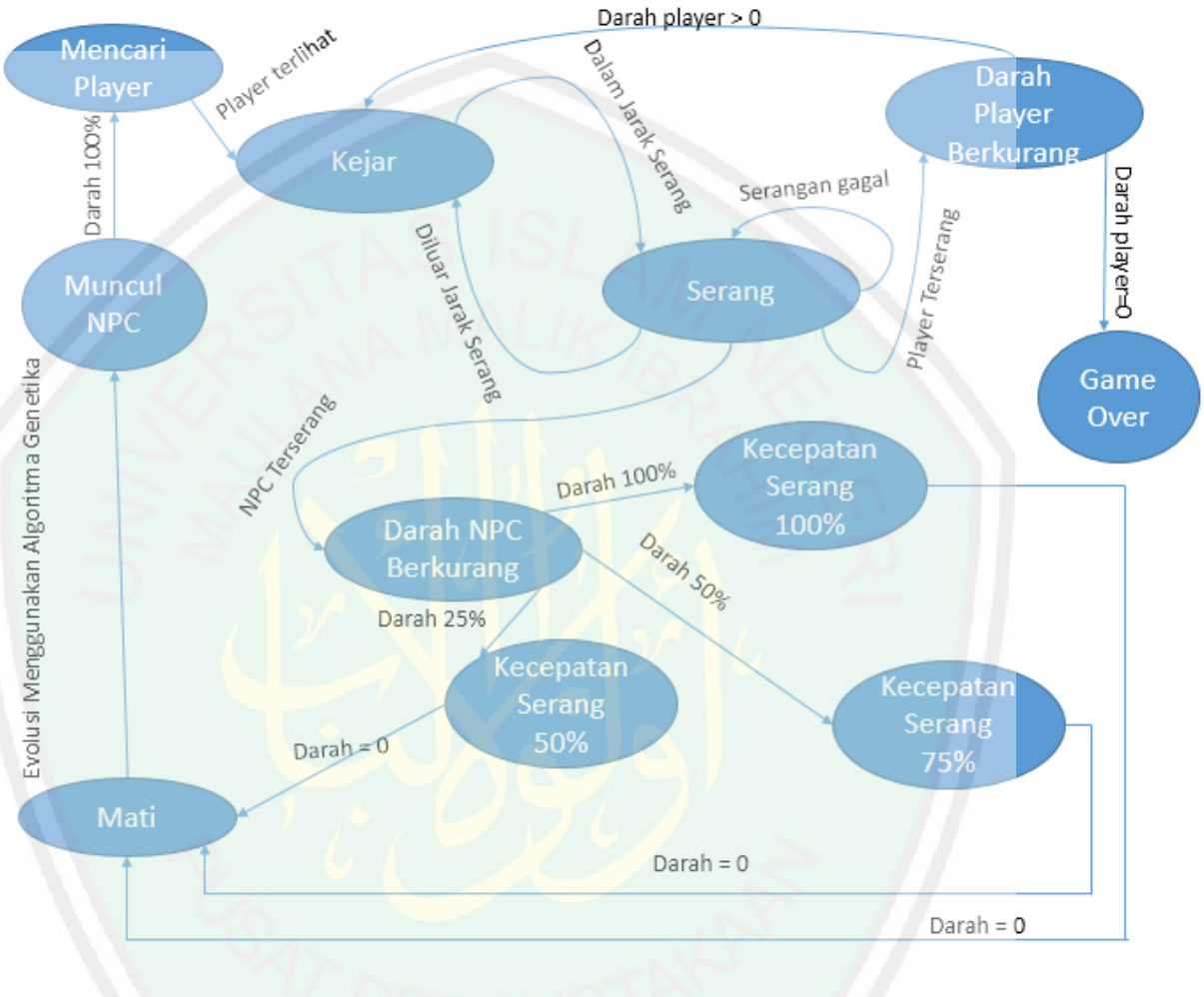
5	Musuh Terbunuh	Pemain Karakter Musuh		Ketika pemain berhasil membunuh musuh maka waktu akan bertambah
6	*Player* terserang	Pemain Karakter Musuh		Darah *player* akan berkurang ketika terkena serangan oleh musuh, sejumlah dengan *attack damage* yang dipunyai musuh
7	*Player* menghindar	Pemain Karakter Musuh		*Player* menghindari musuh untuk menghindari kematian dengan mencari *item heal*

8	*Item Heal*	Pemain Karakter *Item Heal*		*Item heal* berada di posisi yang *random* di dalam *map* yang harus ditemukan oleh *player*
9	Soal	Pemain Soal		Setelah *item heal* diambil maka akan muncul soal
10	Berhasil Menjawab	Pemain Soal		Pemain akan mendapatkan bonus darah ketika benar menjawab soal

11	Tidak Berhasil Menjawab	Pemain Soal		Pemain tidak akan mendapat bonus darah karena salah menjawab
12	Waktu Habis	Pemain Karakter		Ketika waktu habis maka *game* akan selesai, dan *player* kalah
13	*Game Over*	Pemain		Terdapat 2 kondisi untuk *game over* yaitu darah pemain habis atau waktu permainan habis

3.2 Perancangan Sistem

3.2.1. *FSM*

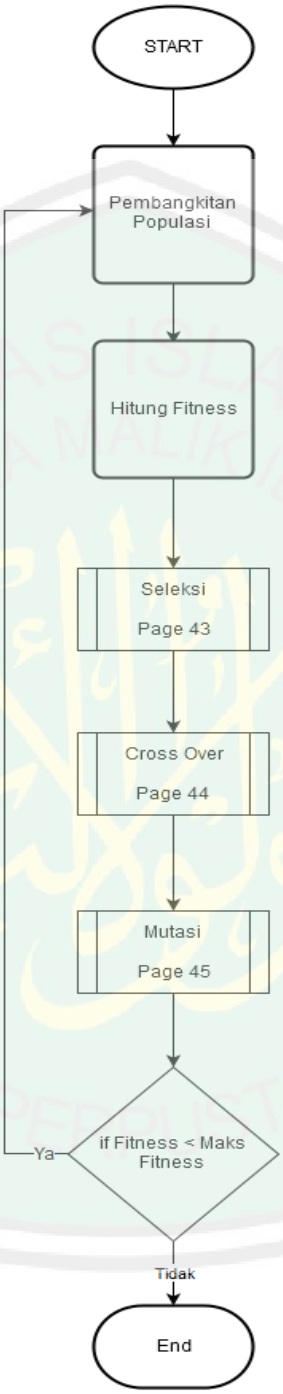


Gambar 3.1. *FSM*

Dari *FSM* di halaman sebelumnya dapat dilihat alur dari NPC mulai dari awal muncul sampai NPC berhasil di bunuh atau membunuh karakter. Pertama kali setelah NPC muncul adalah mencari *player*, ketika *player* terlihat maka NPC mulai mengejar. Saat *player* sudah di dalam jarak serang maka NPC akan mulai menyerang dan jika *player* berlari dan keluar dari jarak serang maka NPC akan kembali mengejar. Ketika serangan dari NPC mengenai *player* maka darah *player* akan berkurang dan jika darah *player* habis maka *game* berakhir, sebaliknya jika serangan gagal maka NPC akan mencoba menyerang kembali.

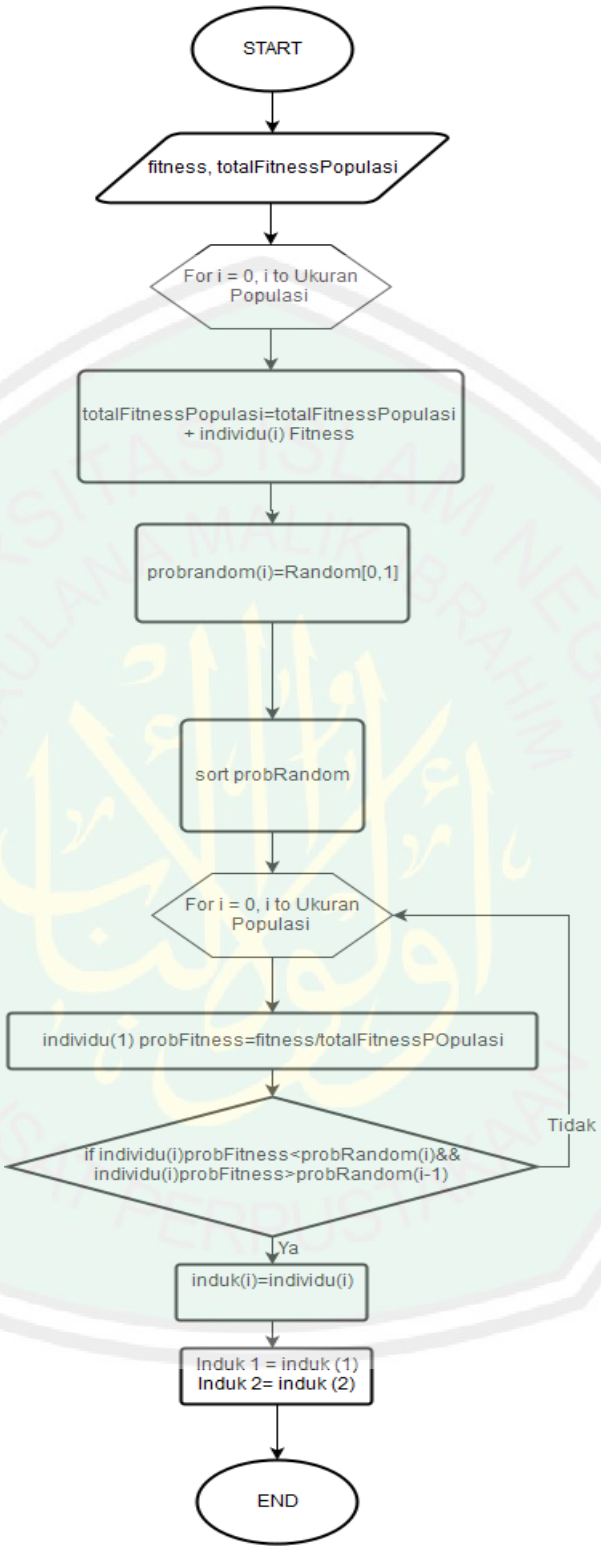
Jika NPC terserang maka darah NPC akan berkurang. Ketika darah NPC 100%, NPC akan memiliki kecepatan serang 100% dari variabel yang sudah ditentukan dari algoritma genetika. Ketika darah NPC 50% maka kecepatan serang menurun menjadi 75% dan jika sisa darah NPC 25% maka kecepatan serangnya hanya 50% dari maksimal kecepatan serang. Saat darah NPC habis maka akan mati dan muncul NPC dari generasi berikutnya.

3.2.3. Flowchart Algoritma Genetika



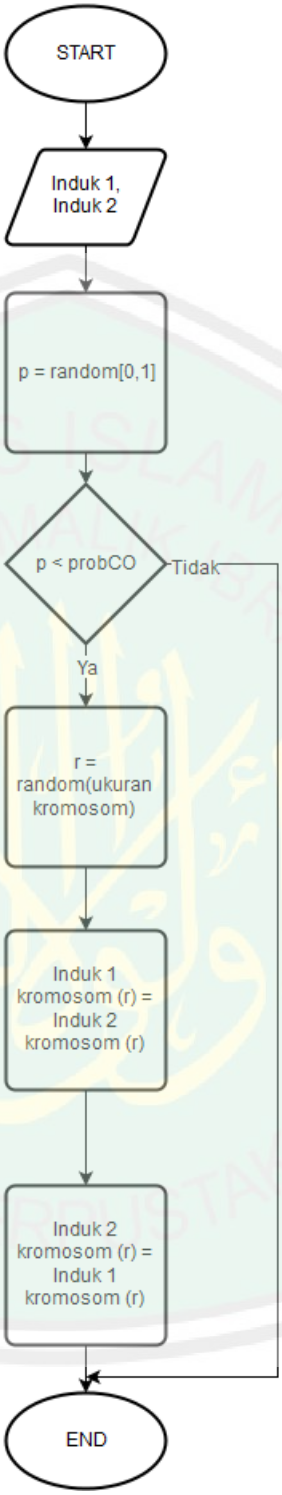
Gambar 3.2 Flowchart Algoritma Genetika

Dari *flowchart* di atas setelah *start*, dilakukan pembangkitan populasi dimana tiap individu memiliki parameter *health, attack damage, attack speed,* dan *movement speed*. Setelah itu menghitung nilai *fitness* dari tiap individu dimana fungsi *fitness*nya di hitung dari perbandingan kromosom yang dimiliki dengan kromosom maksimal yang ingin dicapai. Jika *fitness* sudah didapatkan maka dimulai proses evolusi menggunakan seleksi lalu *cross-over* dan mutasi. Proses terakhir yaitu mengambil *fittest* dari generasi yang baru saja dibuat dan dibandingkan dengan *fittest* yang ingin dicapai. Jika *fittest* belum sesuai dengan tujuan maka akan dilakukan perulangan dengan membuat generasi berikutnya, dan jika *fittest* dari generasi tersebut sesuai dengan tujuan maka program berakhir.



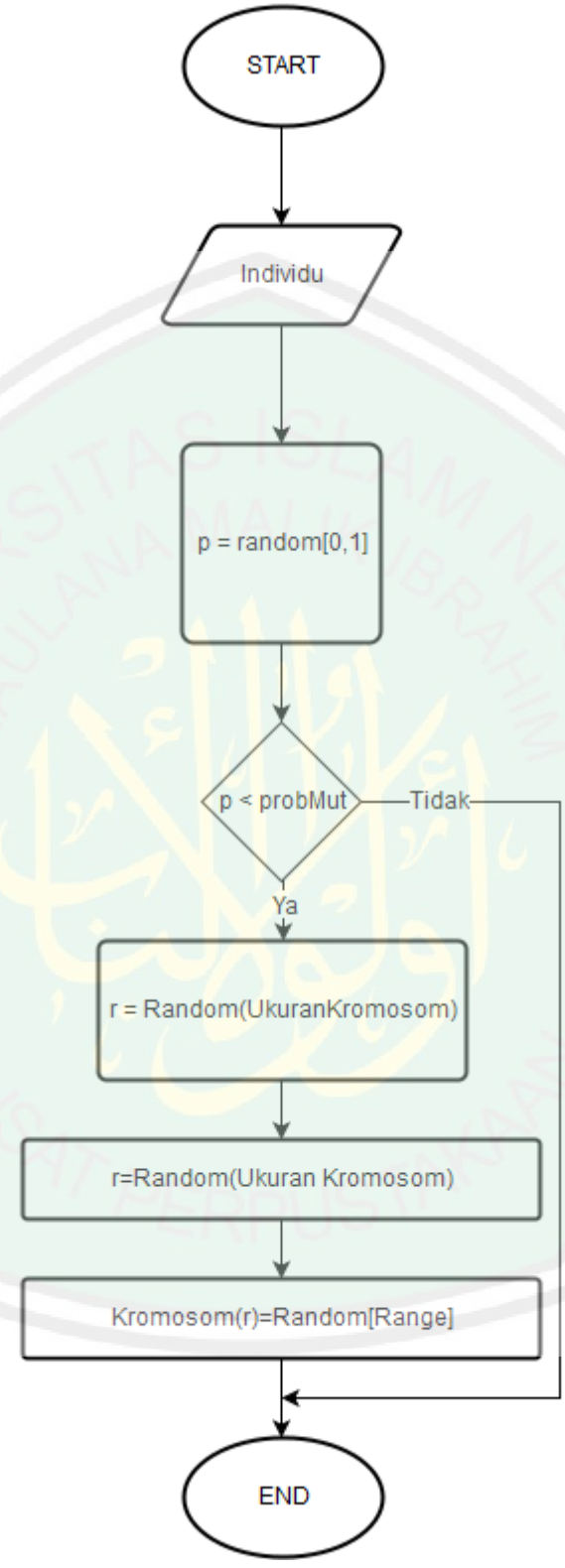
Gambar 3.3 Flowchart Seleksi

Seleksi merupakan tahap ke 2 dari proses evolusi algoritma genetika. Proses ini bertujuan untuk menyaring individu-individu yang ingin di *cross-over* maupun mutasi. Proses seleksi dimulai dari membuat ruang bagi masing-masing individu sesuai dengan *fitness*nya semakin besar *fitness* maka semakin besar peluang individu tersebut terpilih. Setelah itu *random* probabilitas seleksi ketika nilai dari individu dengan probabilitas *random* lebih kecil dan lebih besar dari individu sebelumnya maka individu tersebut dipilih sebagai induk. Begitu seterusnya induk akan dipakai untuk memnbuat keturunan berikutnya dengan *cross-over*.



Gambar 3.4 Flowchart CrossOver

Proses *cross-over* dimulai setelah program seleksi mendapatkan induk-induk untuk di *cross-over*. Pertama-tama setelah mendapatkan induk maka dibuat nilai secara *random*, nilai ini untuk menentukan kromosom mana yang akan ditukar. Ketika nilai tersebut lebih kecil daripada probabilitas *cross-over* maka *random* kembali sebuah nilai sebanyak jumlah kromosom untuk menentukan kromosom mana yang akan ditukar. Setelah kromosom tertukar mana terbentuklah individu baru dari hasil pertukaran kromosom, dan program *cross-over* berakhir



Gambar 3.5 Flowchart Mutasi

Mutasi adalah tahap evolusi terakhir dari algoritma genetika. Mutasi dimulai dari mengambil salah satu individu dan dibangkitkan nilai *random* ketika nilai *random* tersebut lebih besar daripada probabilitas mutasi maka program berakhir dan individu tidak di mutasi, namun jika nilai *random* lebih kecil dari probabilitas seleksi maka akan kembali membangkitkan nilai *random* untuk memnentukan kromosom mana yang akan di mutasi setelah terpilih kromosom yang dimutasi maka *random* nilai lagi untuk salah satu gen yang terpilih tadi. Setelah gen berganti dengan hasil mutasi maka terbentuklah individu baru, dna program mutasi berakhir

3.2.4. Perancangan Algoritma Genetika

Dalam pengembangan aplikasi, algoritma genetika akan diterapkan pada perilaku NPC. Berikut adalam langkah-langkah implementasi algoritma genetika.

Langkah pertama yang dilakukan adalah pembentukan individu. Pembentukan individu dimulai dengan deklarasi kromosom yang dibutuhkan. Nilai dari kromosom adalah angka acak dalam rentang tertentu.

Tabel 3.1 Kromosom

Kromosom
Attack Damage
Attack Speed
Health Point
Movement Speed

Setelah ditentukan kromosom maka langkah selanjutnya adalah mengatur gen-gen dari tiap kromosom secara *random.* Peneliti tidak mengatur setiap individu menggunakan satu set konstanta yang telah ditentukan agar populasi menjadi beragam sebisa mungkin. Sebuah populasi yang tidak beragam akan tidak efektif dalam menentukan solusi terbaik.

Generasi pertama:

Tabel 3.2 Generasi Pertama

Individu	Kromosom			
	Health	*Attack Damage*	*Attack Speed*	*Move Speed*
A	R (75-150)	R (5-10)	R (1-3)	R (4-8)
B	R (75-150)	R (5-10)	R (1-3)	R (4-8)
C	R (75-150)	R (5-10)	R (1-3)	R (4-8)
D	R (75-150)	R (5-10)	R (1-3)	R (4-8)

Keterangan Tabel 3.2:

R =*Random Integer*

Berikut adalah contoh individu yang dihasilkan pada generasi pertama:

Misal terdapat 4 individu dalam populasi saat ini dengan kromosom yang digambarkan pada tabel dan ingin mengetahui populasi baru yang optimal.

Tabel 3.3 Individu Pertama

Kromosom	
Health Point	139
Attack Damage	5
Attack Speed	2
Movement Speed	8

Tabel 3.4 Individu Kedua

Kromosom	
Health Point	87
Attack Damage	7
Attack Speed	2
Movement Speed	5

Tabel 3.5 Individu Ketiga

Kromosom	
Health Point	105
Attack Damage	6
Attack Speed	3
Movement Speed	7

Tabel 3.6 Individu Keempat

Kromosom	
Health Point	120
Attack Damage	8
Attack Speed	3
Movement Speed	6

Langkah berikutnya menghitung nilai *fitness* dari masing-masing individu.

Individu 1: $fitness = \frac{152}{171}$ =0.900

Individu 2: $fitness = \frac{101}{171}$ =0.590

Individu 3: $fitness = \frac{121}{171}$ =0.707

Individu 4: $fitness = \frac{137}{171}$ =0.801

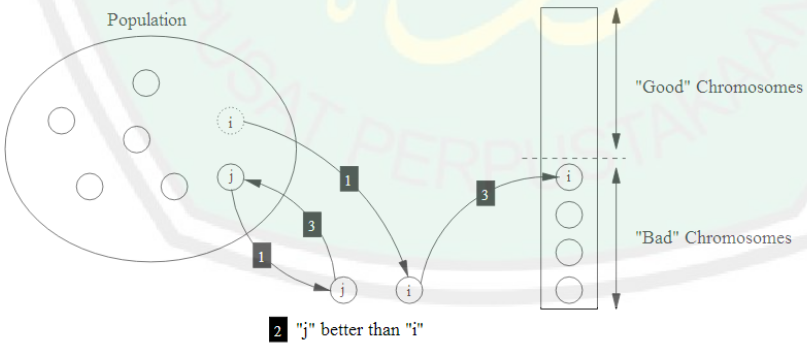
Dari perhitungan di atas bisa ditarik ke dalam tabel sebagai berikut:

Tabel 3.7 Perhitungan *Fitness*

Individu Ke-	1	2	3	4
Fitness	0.900	0.590	0.707	0.801

Setelah itu dilakukan proses seleksi untuk menentukan individu mana yang cocok untuk dijadikan induk untuk generasi selanjutnya. Pada proses seleksi individu di dalam populasi yang mempunyai *fitness* tertinggi berkemungkinan besar untuk menjadi inputan proses *cross-over*.

Berikut adalah simulasi dalam proses seleksi dengan menggunakan metode *Tournament Selection*:



Gambar 3.20 *Tournament Selection* (Sumber : www.otlet-institute.org)

Tabel 3.6 Probabilitas Seleksi

Individu ke-	1	2	3	4
Fitness	0.900	0.590	0.707	0.801
Lolos seleksi	Tidak	Tidak	Ya	Ya

Cross-over (perkawinan Silang) mengkombinasikan dua individu untuk memperoleh individu-individu baru yang diharapkan mempunyai *fitness* lebih baik. Prosesnya menyalin isi kromosom individu 1 ke dalam kromosom *offspring* 1 dan kromosom individu 2 ke kromosom *offspring* 2, selanjutnya menurunkan isi dari kedua kromosom tersebut. Banyaknya pasangan induk yang mengalami *cross-over* ditentukan dengan nilai probabilitas *cross-over*. Pada proses *cross-over* dipilih inputan yaitu dua induk dari hasil seleksi. Kromosom yang ditukar sesuai dengan syarat probabilitas rate *cross-over*.

Langkah-langkah penyelesaian *cross-over:*

Identifikasi induk yang dijadikan inputan *cross-over*

Tabel 3.9 Induk Untuk *Crossover*

Individu	Kromosom			
1	HP=105	ATKDMG=6	ATKSPEED=3	MOVESPEED=7
2	HP=120	ATKDMG=8	ATKSPEED=3	MOVESPEED=6

Bangkitkan bilangan acak [0,1] untuk menentukan kromoosom yang akan digunakan sebagai pertukaran nilai.

Misal bilangan acak sebagai berikut:

Kromosom 1: 0.32

Kromosom 2: 0.43

Kromosom 3: 0.86

Kromosom 4: 0.92

Membandingkan bilangan acak dengan probabilitas *cross-over rate*. Misal *cross-over rate* ditentukan dengan nilai 0.5 maka bilangan acak yang kurang dari sama dengan 0.5 akan ditukar.

Kromosom 1: 0.32 (Ya)

Kromosom 2: 0.43 (Ya)

Kromosom 3: 0.86 (Tidak)

Kromosom 4: 0.92 (Tidak)

Melakukan pertukaran sesuai syarat:

Induk

Tabel 3.10 Induk Untuk *Crossover*

Individu	Kromosom			
1	HP=105	ATKDMG=6	ATKSPEED=3	MOVESPEED=7
2	HP=120	ATKDMG=8	ATKSPEED=3	MOVESPEED=6

Offspring

Tabel 3.11 *Offspring*

Individu	Kromosom			
Offspring1	HP=120	ATKDMG=8	ATKSPEED=3	MOVESPEED=7
Offspring2	HP=105	ATKDMG=6	ATKSPEED=3	MOVESPEED=6

Setelah itu akan dilakukan proses mutasi

Membangkitkan bilangan secara acak [0,1] untuk menentukan kromosom yang akan dimutasi.

Misal bilangan acak sebagai berikut:

Tabel 3.12 Bilangan *Random* Untuk Mutasi

Individu	Bilangan Acak			
1	0.83	0.12	0.22	0.12
2	0.15	0.87	0.99	0.1
3	0.97	0.01	0.42	0.66
4	0.88	0.2	0.1	0.71

Membandingkan bilangan acak dengan *mutation rate*

Misal *mutation rate* ditentukan nilainya 0.6 maka bilangan yang kurang dari sama dengan 0.6 akan dimutasi.

Tabel 3.13 Hasil Mutasi *Rate*

Individu	Bilangan Acak			
1	0.83	0.12 (Ya)	0.22 (Ya)	0.85
2	0.03 (Ya)	0.87	0.99	0.1 (Ya)
3	0.97	0.01 (Ya)	0.78	0.66
4	0.88	0.2 (Ya)	0.1 (Ya)	0.71

Kromosom yang terpilih untuk dimutasi akan kembali di acak:

Tabel 3.14 *Random* Kromosom Termutasi

Individu	Kromosom			
	Health	*Attack Damage*	*Attack Speed*	*Move Speed*
1	R (75-150)	R (5-10)	R (1-3)	R (4-8)
2	R (75-150)	R (5-10)	R (1-3)	R (4-8)
3	R (75-150)	R (5-10)	R (1-3)	R (4-8)
4	R (75-150)	R (5-10)	R (1-3)	R (4-8)

Sehingga Populasi baru yang terbentuk dari mutasi dapat digambarkan sebagai berikut :

Tabel 3.15 Generasi Baru

Individu	Kromosom			
	Health	*Attack Damage*	*Attack Speed*	*Move Speed*
1	120	6	2	7
2	140	8	2	6
3	105	5	3	7
4	105	9	3	7

Tabel di atas adalah hasil dari evolusi yaitu terbentuknya populasi baru. Proses evolusi akan terus-menerus dilakukan sampai mendapatkan solusi optimal atau nilai *fitness* sama dengan nilai maks *fitness*. Misal maks *fitness* adalah 10, maka perulangan akan berakhir ketika nilai *fitness* mencapai 10. Sebagai contoh pada tabel berikut:

Tabel 3.16 Perhentian Perulangan

Generasi	*Fittest*
1	0.505
2	0.717
3	0.717
4	0.811
5	0.901
6	0.950
7	1

Dari tabel di atas dapat dilihat bahwasanya didapatkan solusi optimal pada generasi ke-7, maka perulangan dihentikan pada saat itu.

BAB IV

HASIL DAN PEMBAHASAN

Langkah selanjutnya setelah perancangan sebuah sistem adalah mengimplementasikan sistem yang telah dibuat. Langkah terakhir adalah pengujian rancangan sistem secara keseluruhan, langkah ini dilakukan karena dibutuhkan untuk memastikan hasil rancangan sistem sesuai dengan kebutuhan sistem.

4.1 Implementasi sistem

Dalam bab ini membahas mengenai implementasi metode terhadap aplikasi yang sudah dibuat dan juga pengujian metode yang diterapkan. Serta melakukan uji coba pada aplikasi yang telah dibangun, apakah telah sesuai dengan perancangan dan hasil yang diharapkan. Sebelum diimplementasikan, terlebih dahulu dipaparkan spesifikasi sistem perangkat keras *(hardware)* dan perangkat lunak *(software).* Untuk pembuatan dan melakukan uji coba aplikasi ini diperlukan perangkat keras dan perangkat lunak.

4.1.1 Kebutuhan Perangkat Keras

Perangkat keras yang digunakan dalam pembuatan *game* dan pengujian metode, sebagai berikut:

Tabel 4.1 Kebutuhan Perangkat Keras

No	Perangkat Keras	Spesifikasi
1	Processor	AMD A8-5550M 2.1 GHz
2	Ram	4 GB
3	VGA	AMD Radeon HD Grapics
4	HDD	600 GB
5	Monitor	14’
6	Speaker	On
7	Mouse & Keyboard	On

4.1.2 Kebutuhan Perangkat Lunak ba

Tabel 4.2 Kebutuhan Perangkat Lunak

No	Perangkat Lunak	Spesifikasi
1	Sistem Operasi	Windows 7 64 Bit
2	*Game Engine*	Unity 3D 5.2.2f1
3	Desain 3D	Blender 2.76
4	*Script Writer*	Mono Develop

4.2 Implementasi metode

4.2.1 Algoritma Gentika

Hasil implementasi algoritma genetika adalah daftar generasi yang diperoleh dari iterasi metode *GA* dari awal sampai menemukan solusi terbaik.

Langkah pertama yang dilakukan adalah membangkitkan individu, yang di tunjukan pada gambar di bawah ini.

```
public void GenerateIndividual(){


 health= random.nextInt(150 - 75) + 75;
 attack = random.nextInt(10 - 5) + 5;
 attspeed = random.nextInt(3 - 1) + 1;
 move = random.nextInt(8 - 4) + 4;
 Kromosom[0]=health;
 Kromosom[1]=attack;
 Kromosom[2]=attspeed;
 Kromosom[3]=move;
}
```

Gambar 4.1 Pembangkitan Individu

Setelah individu sudah didapat maka dilakukan perulangan untuk mendapatkan populasi.

```
public Population(int populationSize, boolean initialise) {
        individuals = new Individual[populationSize];

        if (initialise) {

            for (int i = 0; i < size(); i++) {
                Individual newIndividual = new Individual();
                newIndividual.GenerateIndividual();
                saveIndividual(i, newIndividual);

            }
        }
    }
```

Gambar 4.2 Pembangkitan Populasi

Penghitungan *fitness* dilakukan untuk evaluasi terhadap setiap individu di populasi.

```
static double getFitness(Individual individu){
double max=0;


      fitness=(individu.getGene(0)+individu.getGene(1)+individu.getGene(2)+individu.getGene(3));
     fitness=fitness/171;

 return fitness;
 }
```

Gambar 4.3 Perhitungan *Fitness*

Setelah populasi didapatkan maka dimulai proses evolusi dari tiap individu melalui seleksi, *crossover* dan mutasi berikut *source code*nya.

```
    private static Individual tournamentSelection(Population pop) {

        Population tournament = new Population(tournamentSize, false);

        for (int i = 0; i < tournamentSize; i++) {
            int randomId = (int) (Math.random() * pop.size());
            tournament.saveIndividual(i, pop.getIndividual(randomId));

           // System.out.println("Hasil Individual "+pop.getIndividual(randomId));
        }

        Individual fittest = tournament.getFittest();
        //System.out.println(fittest);
        return fittest;
    }

}
```

Gambar 4.4 Seleksi

```
    private static Individual crossover(Individual indiv1, Individual indiv2) {
        Individual newSol = new Individual();

        for (int i = 0; i < indiv1.size(); i++) {
            // Crossover
            if (Math.random() <= uniformRate) {
                newSol.setGene(i, indiv1.getGene(i));
            } else {
                newSol.setGene(i, indiv2.getGene(i));
            }
        }
        return newSol;
    }
```

Gambar 4.5 *Cross-Over*

```
private static void mutate(Individual indiv) {

    for (int i = 0; i < indiv.size(); i++) {
        if (Math.random() <= mutationRate) {

            if(i==0){
                //System.out.println(i);
                int random = (int )(Math.random() * ((150 - 75) + 1)) + 75;
            //System.out.println(random);
            indiv.setGene(i, random);
            }if(i==1){
                //System.out.println(i);
                int random = (int )(Math.random() * ((10 - 5) + 1)) + 5;
           //  System.out.println(random);
            indiv.setGene(i, random);
            }if(i==2){
                //System.out.println(i);
                int random = (int )(Math.random() * ((3 - 1) + 1)) + 1;
            //System.out.println(random);
            indiv.setGene(i, random);
            }if(i==3){
                //System.out.println(i);
                int random = (int )(Math.random() * ((8 - 4) + 1)) + 4;
            //System.out.println(random);
            indiv.setGene(i, random);
            }


        }
    }
}
```

Gambar 4.6 Mutasi

Tabel 4.3 Populasi Algoritma Genetika

Populasi	Individu	Kromosom				*Fitness*	*Fittest*
		Health	*Attack*	*Attack Speed*	*Move Speed*		
1	1	87	7	2	5	0.590	0.935
	2	*82*	*5*	*2*	*4*	0.543	
	3	*107*	*5*	*1*	*5*	0.690	
	4	*147*	*7*	*2*	*4*	0.935	
2	1	147	7	2	6	0.947	0.947

	2	147	7	2	4	0.935	
	3	147	7	1	4	0.929	
	4	143	6	2	6	0.918	
3	1	147	9	2	6	0.959	0.959
	2	147	7	2	4	0.935	
	3	147	7	1	4	0.929	
	4	147	7	2	6	0.947	
4	1	147	10	3	7	0.976	0.976
	2	144	5	2	4	0.906	
	3	147	7	2	4	0.935	
	4	147	7	2	6	0.947	
5	1	149	9	3	7	0.987	0.987
	2	147	9	2	7	0.964	
	3	147	10	3	7	0.976	
	4	147	7	2	4	0.935	
6	1	150	10	3	8	1.0	1.0
	2	149	9	3	7	0.987	
	3	147	10	3	7	0.976	
	4	149	10	3	8	0.994	

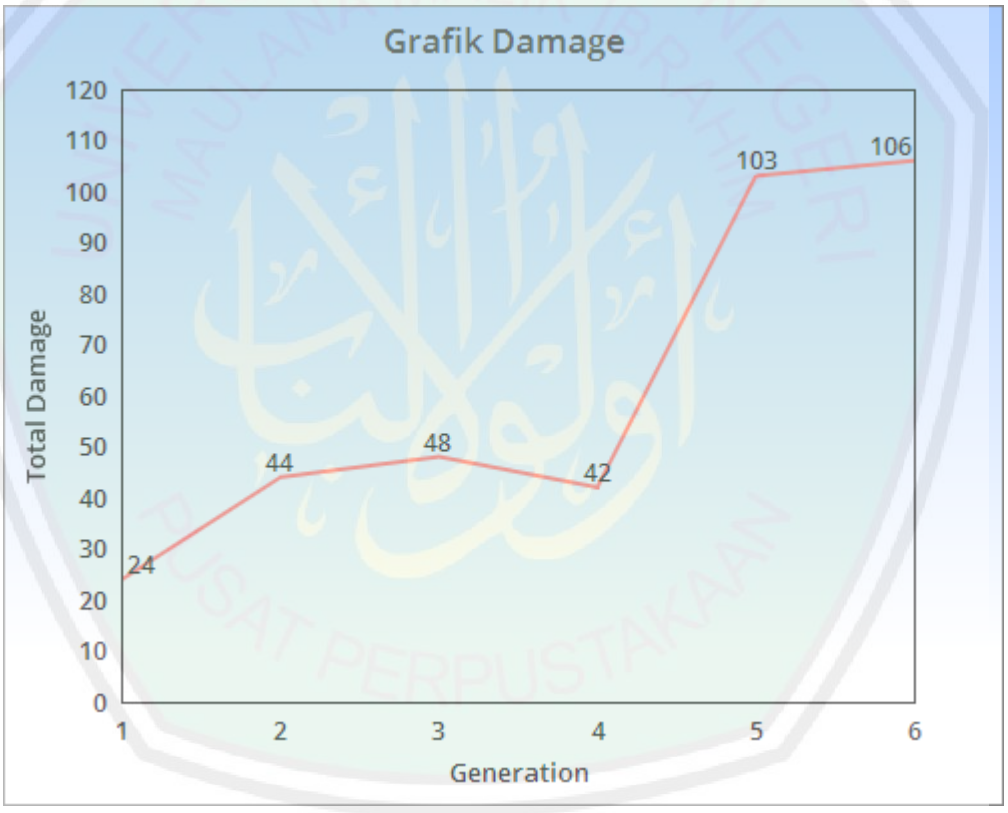
Dari populasi di atas setelah diujicobakan di dalam *game* ditemukan data sebagai berikut:

Tabel 4.4 Hasil Uji Coba

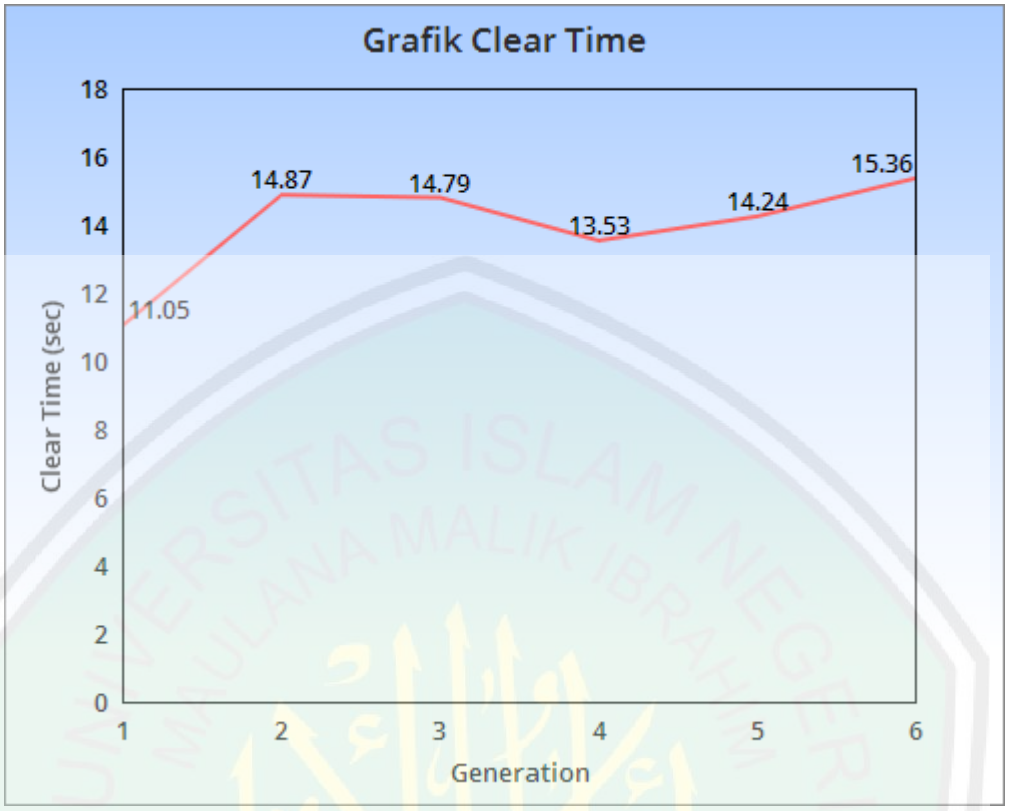
Populasi	*Damage Done*	*Clear Time*	*Item Used*
1	24	11.05 detik	0
2	44	14.87 detik	0

3	48	14.79 detik	0
4	42	13.53 detik	0
5	103	14.24 detik	1
6	106	15.36 detik	1

Untuk memudahkan melihat data hasil uji coba di atas maka dapat di tarik kedalam grafik seperti di bawah ini:



Gambar 4.7 Grafik *Damage*



Gambar 4.8 Grafik *Clear Time*

Dari kedua grafik di atas dapat dilihat bahwa grafik *damage* menunjukan kenaikan yang cepat kecuali dari generasi ke 3 dan 4 yang mengalami penurunan *damage*. Sedangkan pada grafik *clear time* dari generasi 1 ke generasi 2 mengalami kenaikan signifikan waktu sebesar 3.82 detik, sedangkan generasi ke 2 sampai ke 5 *clear time* tidak mengalami perubahan yang besar dan pada generasi terakhir *clear time* menunjukan waktu yang paling lama.

4.2.2 Presentase Kecepatan Algoritma Genetika

Prosentase kecepatan dari hasil iterasi algoritma genetika dapat dihitung berdasarkan rumus berikut (Nirvana S.Antonio et al,2013):

$$Prosentase\ Kecepatan = \frac{\sum Populasi - \sum hasilkurangsempurna}{\sum Populasi}\ x\ 100\%$$

Keterangan:

$\sum Populasi$ = Jumlah total populasi selama program di jalankan

$\sum hasilkurangsempurna$ = Jumlah total dimana *fittest* mengalami penurunan atau *fittest* tetap dari generasi sebelumnya

Dari 20 hasil iterasi populasi, prosentase kecepatan populasi mengalami peningkatan nilai *fittest* setiap generasinya adalah:

$$Prosentase\ Kecepatan = \frac{20 - 5}{5}\ x\ 100\% = 80\%$$

Hasil uji coba di atas menunjukan bahwa NPC musuh semakin baik dalam karakteristik dan perilaku, hal itu akan menambah tantangan untuk pemain dalam menghadapi NPC musuh.

4.3 Implementasi Aplikasi *Game*

Pada bagian ini akan dipaparkan tentang implementasi dari rancangan *game "Fun English"* ke dalam aplikasi berbasis *desktop* dan penjelasan tiap-tiap tahap pada *game* nya. Berikut adalah hasil implementasinya:

4.3.1 Antarmuka *Main Menu Game*

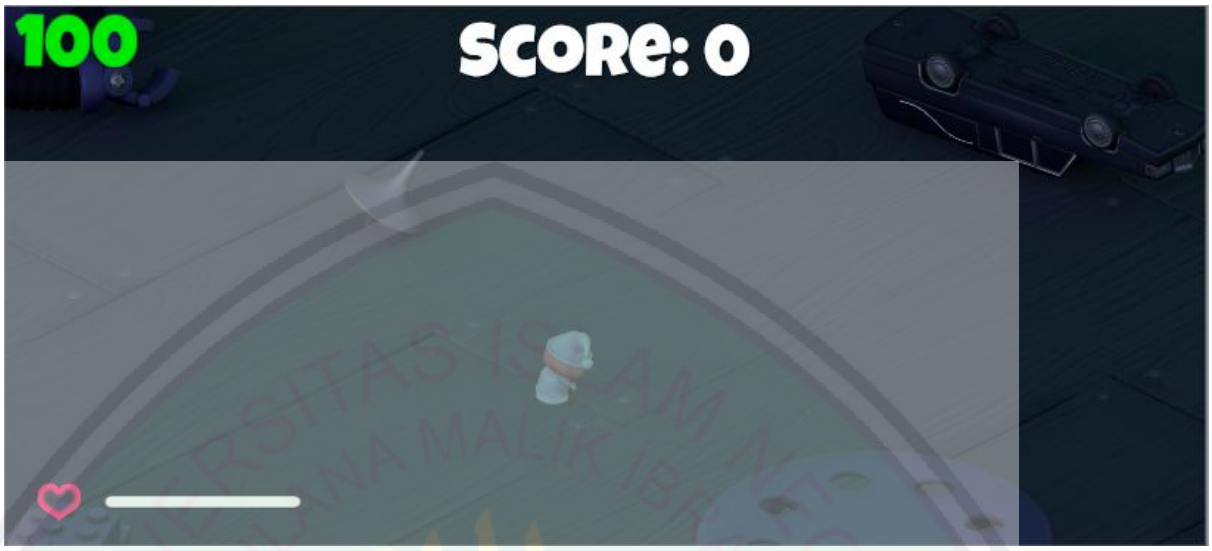
Gambar 4.9 *Main Menu*

Pada scene ini terdapat tampilan menu utama. Terdapat 4 menu pilihan yaitu *Play,Option* dan *Exit.* Berikut adalah tampilan dari *Main Menu:*

Fungsi dari tiap-tiap tombol di *main menu* adalah sebagai berikut:

1. *Play* : berfungsi untuk memulai *game.*
2. *Option* : berfungsi untuk melakukan *setting* di dalam *game* untuk menentukan apakah suara diaktifkan atau tidak .
3. *Exit* : berfungsi untuk keluar dari aplikasi *game*

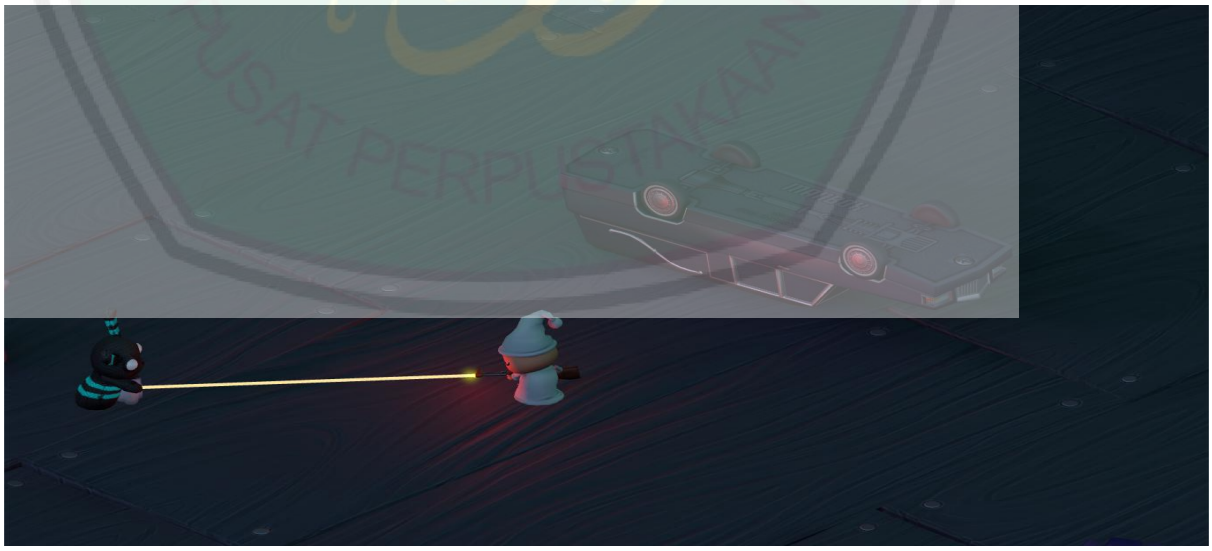
4.3.2 Antarmuka Arena *Game*



Gambar 4.10 Arena

Setelah pemain menekan tombol *play* maka akan muncul arena dan pemain yang akan di kontrol oleh *player*.

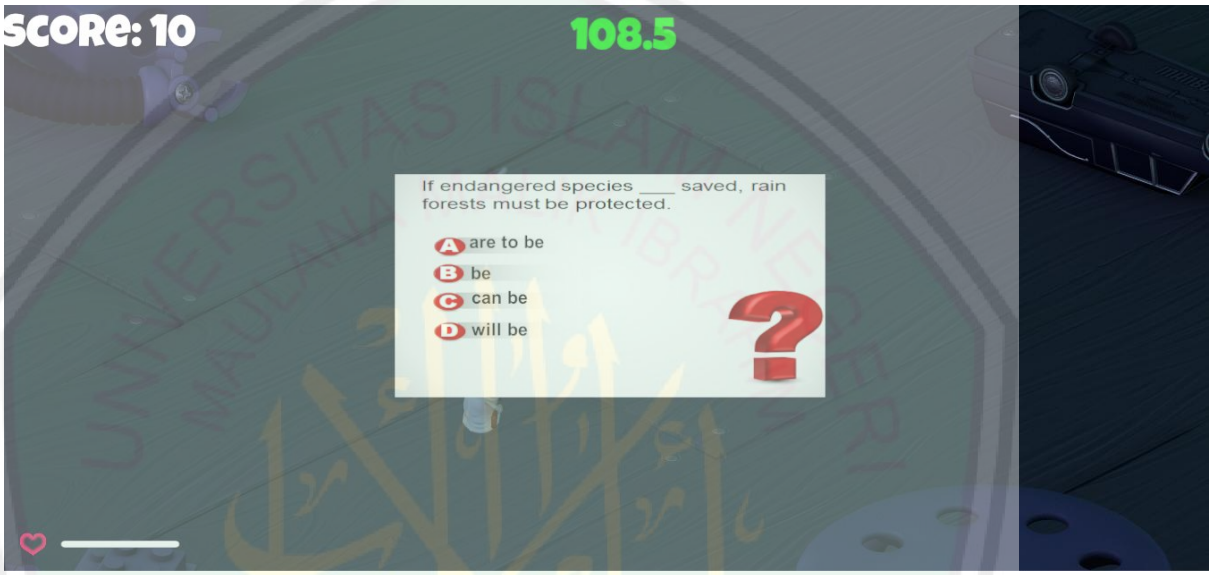
4.3.3 Antarmuka *Gameplay*



Gambar 4.11 *Gameplay*

Di sini *player* diharuskan membunuh musuh untuk bertahan diri selama mungkin di dalam *game*. *Player* dapat menembak musuh untuk mendapatkan waktu agar tidak terjadi *game over* saat waktu habis.

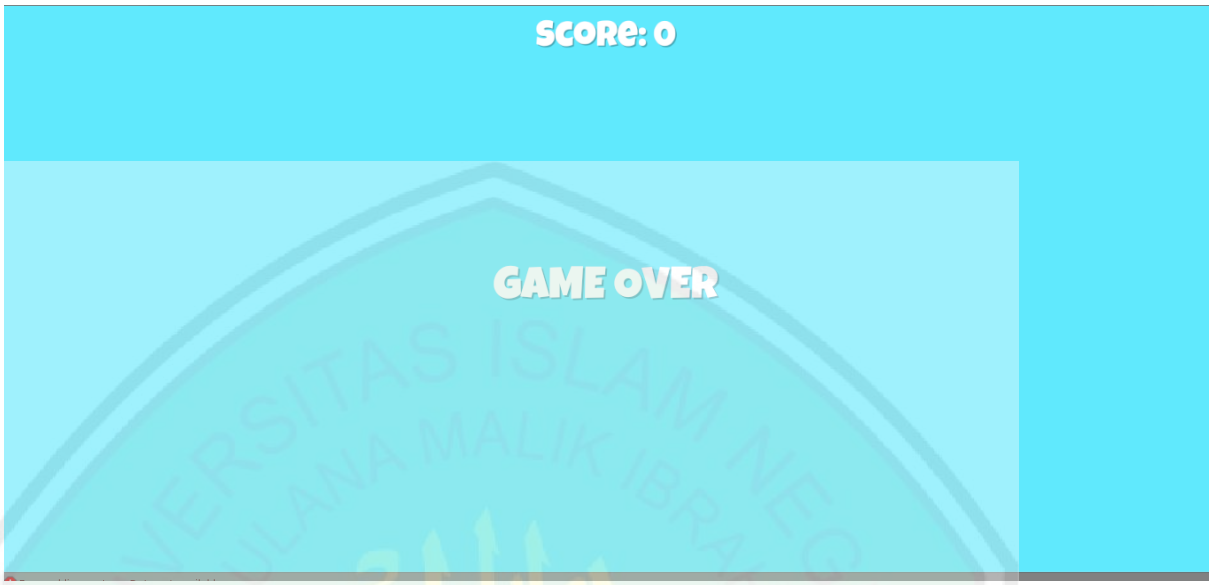
4.3.4 Antarmuka Soal



Gambar 4.12 Soal

Untuk menyembuhkan sebagian darah, *player* dapat mengambil *item*. Setelah mengambil *item* soal akan muncul, jika *player* menjawab dengan benar maka darah akan ditambah jika salah maka darah *player* akan tetap seperti saat sebelum mengambil *item*.

4.3.5 Antarmuka *GameOver*



Gambar 4.13 *Game Over*

Saat darah *player* habis atau waktu di dalam *game* nol maka *game* akan selesai dan menunjukan score yang di dapatkan oleh *player*.

4.4 Integrasi *Game Fun English* dengan Islam

Bahasa adalah salah satu faktor terpenting dalam menjalin hubungan komunikasi dengan orang lain. Tanpa bahasa, kita akan kesulitan berkomunikasi dengan manusia lainya. Bahasa juga merupakan salah satu kekuasaan Allah yang Maha Besar. Oleh karena itu, Allah menciptakan berbagai macam bahasa di dunia ini sebagai wujud kekuasaan yang dimiliki-Nya, Agar kita senantiasa bersyukur dan selalu mencari kebaikan di jalan yang benar. Sebagian firman Allah swt dalam Surah Ar-Rumm ayat 22 yang berbunyi:

وَمِنْ ءَايَٰتِهِۦ خَلْقُ ٱلسَّمَٰوَٰتِ وَٱلْأَرْضِ وَٱخْتِلَٰفُ أَلْسِنَتِكُمْ وَأَلْوَٰنِكُمْ ۚ إِنَّ فِى ذَٰلِكَ لَءَايَٰتٍ لِّلْعَٰلِمِينَ ﴿٢٢﴾

Artinya:

"Dan di antara tanda-tanda kekuasaanya-Nya ialah menciptakan langit dan bumi dan berlain-lainan bahasamu dan warna kulitmu. Sesungguhnya pada yang demikian itu benar-benar terdapat tanda-tanda bagi orang-orang yang mengetahui."

Berdasarkan tafsir Jalalain karya Imam Jalaludin Al-Mahalli dan Imam Jalaludin As-Suyuti, maksud dari ayat tersebut yaitu dengan bahasa yang berlainan ada yang berbahasa Arab dan ada yang berbahasa Ajam serta berbagai bahasa lainnya, diantara kalian ada yang berkulit putih, ada yang hitam dan lain sebagainya. Padahal kalian berasal dari seorang lelaki dan perempuan, yaitu Nabi Adam dan Siti Hawa. Yang demikian itu benar-benar terdapat tanda yang menunjukan kekuasaan Allah SWT. Bagi orang-orang yang berakal dan berilmu (Tafsir Jalalain,2008 :454).

Rasullullah SAW pun memerintahkan sahabat Zaid bin tsabit untuk mempelajari berbagai bahasa asing agar dalam perjalanan dakwahnya dapat terhindar dari tipu muslihat serta dapat memperluas penyebaran dakwah islam ke berbagai wilayah. Hal ini merupakan bukti bahwa mempelajari bahasa asing itu penting.

BAB V

PENUTUP

5.1 Kesimpulan

Berdasarkan hasil dari implementasi dan pengujian yang dilakukan peneliti, maka didapatkan kesimpulan sebagai berikut:

1. Algoritma genetika dapat digunakan untuk model nilai-nilai kekuatan dari *NPC* pada *game "Fun English".*
2. Tingkat keoptimalan yang diperoleh dari tiap generasi pada uji coba pada NPC memiliki persentase keberhasilan sebesar 80%. Hal ini dapat dilihat pada bab iv pada tabel 4.3.

5.2 Saran

Peneliti yakin dengan penuh kesadaran bahwa dalam pembuatan permainan ini masih banyak kekurangan yang nantinya sangat perlu untuk dilakukan pengembangan demi sumbangsih terhadap ilmu pengetahuan, diantaranya:

1. Membuat *design* karakter dan *environment* terlihat lebih menarik
2. Menambah perilaku dari *NPC* sehingga *NPC* mempunyai karakteristik atau kekuatan khusus yang unik sehingga permainan menjadi lebih menantang.
3. Pada algoritma genetika untuk meningkatkan tingkat kesuksesan dapat mencari nilai *mutation rate, crossover rate* dan cara seleksi yang lebih baik sehingga hasil yang didapat lebih cepat dan tepat.

DAFTAR PUSTAKA

Al-Qur'an dan Terjemahannya, 2003, Semarang, Toha Putera.

Omid E, David et al. *Genetic Algorithm for Evolving Computer Chess Program*, 2014

Dini Nur Setyorini. "*Sistem Penjadwalan Proyek Jaringan Pipa Air Bersih Menggunakan Algoritma Genetika*", 2012

Moshe Sipper. "*Evolving Game-Playing Strategies with Genetic Programming*", 2008

Hans K. G. Fernlund, et al. "*Learning Tactical Human Behavior Through Observation of Human Performance",* 2006

Francisco Azuaje. "*A Computational Evolutionary Approach to Evolving Strategy and Cooperation*", 2003

Utomo Sarjono Putro, Kijima, K. And Takahashi, S. " *Adaptive Learning of Hypergame Situation by Using Genetic Algorithm",* 2000

Johan Saputra. "*Perancangan dan Pembuatan Aplikasi untuk Mengoptimasi Penyusunan Iklan Gambar dengan metode Algoritma Genetik", 2008*

Fathelalem F, Ali, et al. "*Playing the Rock-Paper-Scissors game with a Genetic Algorithm", 2000*

Holland, J. H. *Adaptation in Natural and Artificial System,* University Michigan Press, 1975

Kusumadewi, Sri. 2003. *Artificial Intelligence (Teknik dan Aplikasinya).* Yogyakarta: Graha Ilmu.

Kusumadewi, Sri an Hari Purnomo. 2005. *Penyelesaian Masalah Optimasi dengan Teknik-Teknik Heuristik.* Yogyakarta: Graha Ilmu.

Kuswadi, Son. 2007. *Kendali Cerdas, Teori dan Aplikasi Praktisnya.* Yogyakarta: Andi.

Nirvana S. Antonio et al. *"Optimization of an Evaluation Function of the Four-Sided Dominos Game Using a Genetic Algorithm"*, 2013

http://opticalengineering.spiedigitallibrary.org diakses pada 15 Mei 2016

http://storyboardthat.com/ diakses pada 10 Mei 2016

www.altafsir.com/Al-Jalalayn.asp diakses pada 20 Mei 2016