Procedural Map Generation for 'Splatted': Enhancing Player Experience through Genetic Algorithms and AI Finite State Machines in a Snowball Throwing Game

Lukky Hariyanto1 dan Hendrawan Armanto1

1Departemen Informatika, Fakultas Sains dan Teknologi, Institut Sains dan Teknologi Terpadu Surabaya, Surabaya, Indonesia

**Corresponding author:** Lukky Hariyanto (e-mail: lukky.h20@mhs.istts.ac.id).

ABSTRACT The world of gaming, now a global entertainment phenomenon, has grown to become one of the largest entertainment industries, even surpassing books, films, and music. However, game development is a complex endeavor requiring diverse talents, including directors, project leaders, designers, and programmers. This study aims to alleviate this complexity by creating a game named "Splatted" with key features centered around procedural map generation using genetic algorithms and artificial intelligence employing state machines. The genetic algorithm used is inspired by Darwin's theory of evolution, where each entity possesses unique traits, and superior ones can thrive and pass on their characteristics to their descendants. The research explores two map generation methods: tile-based and template-based, each with its own advantages and drawbacks. The results of the experiments in this study suggest that the genetic algorithm performs effectively and leads to a relatively high level of player satisfaction.

KEYWORDS Game, Genetic Algorithm, Map, Procedural Map Generation.

ABSTRAK Game merupakan aspek hiburan yang sekarang ini merajalela di seluruh dunia dan telah menjadi salah satu industri hiburan terbesar melebihi buku, film, bahkan musik. Tetapi dunia game bukanlah dunia yang mudah, sejumlah talenta diperlukan dalam pembuatannya mulai dari direktur, kepala proyek, desainer, hingga programmer. Untuk membantu beban pekerjaan tersebut maka penelitian ini akan membuat sebuah game bernama Splatted dengan fitur utama adalah procedural map generation berbasiskan algoritma genetik dan kecerdasan buatan berbasiskan state machine. Algoritma genetik itu sendiri adalah sebuah algoritma yang didasarkan pada teori evolusi Darwin, dimana setiap makhluk hidup memiliki fitur sendiri-sendiri dimana makhluk hidup yang unggul dapat bertahan hidup dan meneruskan fiturnya ke keturunannya. Dalam penelitian ini terdapat dua jenis map generation yaitu tile based generation dan template based generation dimana masing-masing jenis memiliki kelebihan dan kelemahannya sendiri-sendiri. Berdasarkan hasil ujicoba yang telah dilakukan pada penelitian ini dapat kami simpulkan bahwa algoritma genetik bekerja dengan baik dan memperoleh tingkat kepuasan pemain cukup tinggi.

KATA KUNCI Algoritma Genetik, Game, Map, Procedural Map Generation.

1. PENDAHULUAN

Setiap manusia membutuhkan hiburan di dalam kehidupannya. Hiburan ini dapat beraneka ragam yang salah satunya berupa permainan. Tidak kalah dengan jenis hiburan lain seperti film atau buku, permainan merupakan jenis hiburan yang memiliki kompleksitas tinggi di dalam pembuatannya. Berbagai komponen seperti cerita, gameplay, balancing, hingga marketing dibutuhkan dalam mendesain sebuah permainan [1]–[3]. Semakin kompleks sebuah permainan maka akan semakin kompleks juga komponen-komponen pembentuknya. Salah satu komponen penting dalam pembuatan permainan adalah level design dan kecerdasan buatan. Walaupun kedua komponen ini bukanlah komponen paling penting dalam pembuatan permainan akan tetapi tanpa level design yang baik dan kecerdasan yang bermutu maka permainan akan terasa hambar.

Dalam pembuatannya, level design membutuhkan usaha yang tidak sedikit[[1]](#footnote-1), pembuat level harus mempertimbangkan berbagai hal mulai dari posisi pemain, posisi musuh, posisi item, pengaturan jalan yang baik sehingga game menarik dan balance, atau pertimbangan-pertimbangan lain. Misalnya pada permainan Live Service seperti Valorant, Apex Legends, Fortnite, atau Overwatch 2 memerlukan update map terus menerus sehingga pemain tidak bosan bermain dan beralih ke permainan lain. Karena kompleksitas ini, berbagai penelitian dilakukan untuk mempermudah desainer dalam melakukan level design yang pada akhirnya lebih dikenal dengan nama Procedural Map Generation [4]–[6]. Beberapa contoh algoritma dalam procedural map generation adalah Perlin Noise [7] yang digunakan pada permainan minecraft, Fractal Terrain Generation [8] pada permainan Terraria, atau algoritma Genetik [9]–[12] pada berbagai penelitian yang ada.

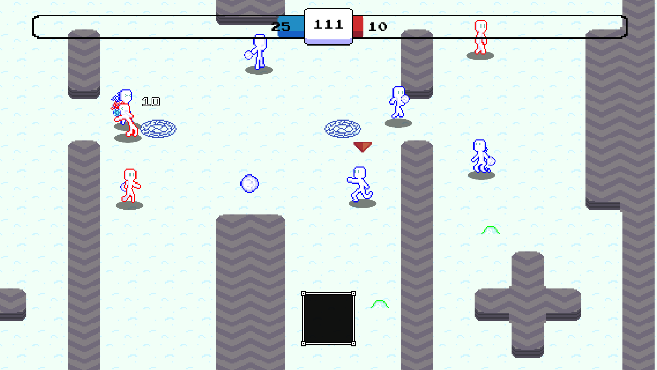
Penelitian ini berfokus pada pembuatan permainan lempar bola salju[[2]](#footnote-2) dimana pembuatan level design nya tidak akan dilakukan secara manual tetapi secara otomatis menggunakan algoritma genetik. Selain itu, dalam permainan ini juga akan dikembangkan kecerdasan buatan yang dapat mempengaruhi tingkah laku NPC sehingga permainan dapat berjalan dengan menarik. Diharapkan melalui penelitian ini, permainan sejenis dapat menerapkan cara kerja yang sama sehingga dapat permainan dapat menjadi lebih menarik.

1. PERMAINAN “SPLATTED”

Permainan ini merupakan permainan yang dikembangkan dalam penelitian ini dan digunakan sebagai studi kasus dalam ujicoba. Sub bab ini akan membahas detail terkait gameplay hingga kecerdasan buatan yang dikembangkan.

1. GAMEPLAY

Splatted adalah sebuah permainan dimana 2 kelompok yang masing-masing terdiri atas 5 orang bermain perang salju. Setiap lemparan pemain mengenai pemain lawan maka pemain akan memperoleh poin dan tujuan utama permainan adalah mendapatkan poin tertinggi dan menjadi pemenang dari perang salju tersebut. Kondisi menang dari permainan ini adalah tim memenuhi target poin yang diminta atau memiliki poin yang lebih banyak dibandingkan tim lawan saat waktu permainan berakhir. Terdapat beberapa aksi yang dapat dilakukan pada permainan ini antara lain: mengambil bola dari tanah, melempar bola dengan tujuan mengenai pemain lawan, menangkap bola yang dilempar pemain lawan, dan melakukan aksi "Fakeout" untuk menipu lawan yang hendak menangkap bola lemparan pemain. Gambar 1 merupakan contoh cuplikan permainan Splatted.



1. Cuplikan layar game Splatted
2. BOLA SPESIAL

Selain bola salju yang umum, agar permainan menjadi lebih menarik maka dalam level disediakan beberapa spawner khusus. Masing-masing spawner memiliki bola spesial yang dapat diambil dan digunakan untuk menyerang lawan. Tabel I merupakan tabel bola spesial yang ada dan fungsi dari bola tersebut.

TABEL I

Bola Spesial Dalam Splatted

|  |  |  |
| --- | --- | --- |
| **Nama** | **Tampilan** | **Fungsi** |
| Ice Piercer |  | Menembus pemain yang dilewati |
| Snow-A-Rang |  | Bola kembali ke pelempar |
| Explod-o-Ball |  | Bola meledak dan mengenai seluruh pemain di area ledakan |
| Freezing Winter |  | Bola memperlambat pemain yang terkena lemparan |
| Stone Auger |  | Menembus pemain dan dinding kemudian pecah menjadi 3 bola salju normal |

1. KECERDASAN BUATAN (NPC)

Pada permainan splatted, tingkah laku NPC diatur oleh sebuah finite state machine (FSM). Dengan menggunakan FSM, NPC akan memiliki beberapa state dimana masing-masing state akan memberikan perlaku yang berbeda. Pergeseran state dalam FSM akan dipengaruhi oleh kondisi NPC secara real time baik yang dipengaruhi oleh NPC lain, pemain, atau lingkungan sekitar. Berikut adalah state yang dapat dimiliki oleh sebuah NPC:

1. RaNDOM WALKING

Apabila NPC sedang tidak membaca bola salju maka NPC akan berjalan menuju lokasi acak yang dipilih dan berusaha mencari bola pada tanah sepanjang perjalanan. Sedangkan apabila NPC sudah memiliki bola salju maka NPC akan mencari lawan. Di saat NPC telah mencapai lokasi yang dituju akan tetapi belum menemukan bola salju atau lawan maka akan dilakukan pemilihan lokasi baru secara random dan berjalan kembali.

1. AMBIL BOLA

Apabila NPC dalam state Random Walking dan NPC melihat sebuah bola salju atau bola spesial maka NPC akan berpindah ke state Ambil Bola. Pada state ini, NPC akan berjalan menuju bola yang telah dilihat dan mengambilnya. Setelah memperoleh bola tersebut atau apabila bola yang sedang dituju diambil pemain lain maka state NPC kembali ke Random Walking.

1. AIM & THROW

Apabila NPC dalam state Random Walking memegang bola ditangan dan melihat tim lawan maka NPC akan memasuki state Aim & Throw. Dalam state ini NPC akan berhenti bergerak dan membidik lawan. Setelah memastikan bidikannya tepat maka NPC akan melempar bola ke arah yang diprediksi merupakan posisi lawan berikutnya. Setelah melempar bola, maka NPC akan kembali ke state Random Walking.

1. FOLLOW TARGET

Apabila target yang sedang dibidik dalam State Aim & Throw hilang dari pandangan NPC, maka NPC akan memasuki State Follow Target. Pada state ini, NPC akan mengejar lawan sehingga lawan dapat terlihat untuk dibidik kembali. Di dalam pengejaran tersebut apabila NPC tidak menemukan lawan dalam kurun waktu tertentu maka NPC akan menyerah dan kembali ke state Random Walking.

1. CATCH BALL

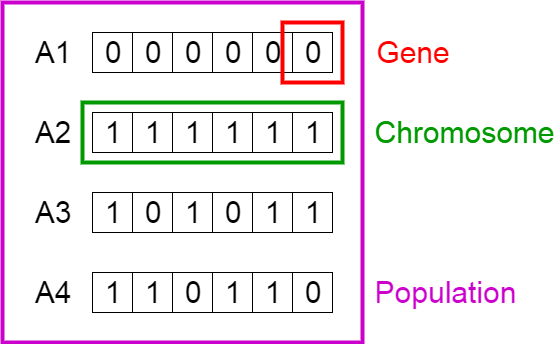
Khusus state Catch Ball, state ini dapat diraih dari state apapun disaat NPC melihat sebuah bola dilemparkan ke NPC tersebut. State ini bertujuan agar NPC dapat mencoba menangkap bola yang sedang dilemparkan ke dirinya.

1. ALGORITMA GENETIK

Algoritma genetik merupakan salah satu algoritma evolutionary yang banyak digunakan oleh berbagai jenis penelitian. Beberapa penelitian tersebut diataranya bertopikan focus crawler [13], optimasi untuk algoritma lain [14], [15], hingga penelitian dalam dunia permainan itu sendiri [12]. Pada penelitian ini sendiri, algoritma genetik merupakan algoritma utama dalam menghasilkan sebuah level design. Oleh sebab itu, sebelum dilakukan pembahasan detail terkait penelitian ini maka sub bab ini akan membahas secara singkat tahapan yang dimiliki oleh algoritma genetik beserta terminologi yang ada.

1. TERMINOLOGI ALGORITMA GENETIK

Dalam algoritma genetik, seluruh konsep yang diterapkan merupakan inspirasi dari kromosom DNA kita. Chromosome/Individu merupakan representasi dari calon solusi yang sedang dicari, dimana setiap chromosome akan memiliki komponen-komponen kecil yang diberi nama Gene. Dalam pencariannya, individu/chromosome bukanlah individu tunggal melainkan bergerak secara berkelompok. Sekelompok individu ini disebut dengan istilah populasi. Terminologi terakhir dalam algoritma genetik adalah generasi, yang merupakan lambang lama hidup sebuah individu. Individu yang dapat bertahan hidup maka akan hidup pada setiap generasi, sedangkan yang tidak maka akan punah dan digantikan individu lain yang lebih kuat.



1. Terminologi yang digunakan pada algoritma genetik
2. MENENTUKAN REPRESENTASI SOLUSI

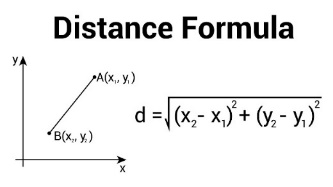
Tahap pertama dalam algoritma genetik adalah penentuan representasi calon solusi permasalahan yang sedang dicari. Representasi ini akan merubah solusi dari permasalahan yang bentuknya abstrak menjadi tipe data[[3]](#footnote-3) yang dapat dikenali algoritma genetik. Misalnya pada permasalahan sederhana seperti mencari nilai x yang menghasilkan y tertinggi pada sebuah persamaan fungsi kuadrat akan direpresentasikan dalam bentuk array 1 dimensi dengan panjang 8 gen, dimana masing-masing gen merupakan nilai bit (0 atau 1) dari calon solusi.

1. PEMBUATAN POPULASI PERTAMA

Populasi pertama akan berperan sebagai kakek buyut dari seluruh individu yang akan datang. Di tahap ini, algoritma genetik akan membuat sebuah populasi yang merupakan kumpulan beberapa indivdu awal secara random. Nilai dari setiap individu tersebut wajib berbeda satu dengan yang lain karena semain banyak variasinya akan mempermudah algoritma genetik dalam mencari solusi di generasi-generasi berikutnya.

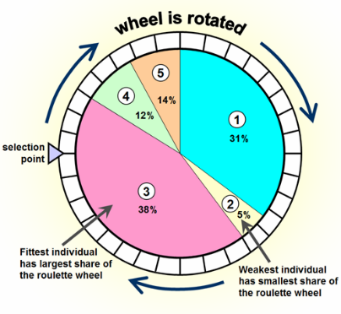
1. PERHITUNGAN FITNESS

Untuk mengukur kebaikan dari setiap individu yang ada (baik populasi awal ataupun populasi pada generasi mendatang) maka diperlukan sebuah nilai fitnes. Nilai ini didasarkan pada constraint dari permasalahan yang ada. Apabila sebuah permasalahan memiliki constraint terkait jarak[[4]](#footnote-4) maka semakin kecil nilai fitnes semakin baik juga individu tersebut akan tetapi berbeda ketika permasalahan memiliki constraint gaji pegawai, maka ada kemungkinan semakin tinggi nilai fitnes malah semakin baik.



1. Rumus jarak antara 2 titik [16]
2. SELEKSI & CROSSOVER

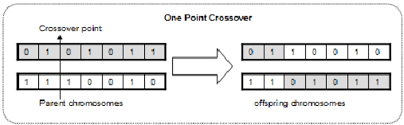
Agar generasi berikutnya dapat menjadi lebih baik dibandingkan generasi saat ini maka algoritma genetik menerapkan konsep reproduksi dimana tahap pertama dari konsep ini adalah penentuan parent (Seleksi). Penentuan parent selalu didasarkan pada nilai fitnes masing-masing individu, dimana semakin baik sebuah individu maka semakin tinggi juga peluang dia terpilih sebagai parent. Beberapa algoritma penentuan parent yang telah dikembangkan antara lain Roulette Wheel Selection [17], Tournament Selection [18], Ranking Selection, atau algoritma lain [19].



1. Ilustrasi Roulette Wheel Selection

Algoritma penentuan parent yang paling popular saat ini adalah Roulette Wheel. Gambar 4 merupakan ilustrasi algoritma roulette wheel yang digunakan. Dimana semakin besar nilai fitnes dari sebuah individu maka semakin besar juga bagian yang diperoleh dalam Roulette Wheel, sehingga kemunculan individu tersebut sebagai calon parent akan semakin besar. Misalnya pada gambar 4, individu 3 merupakan individu dengan nilai fitnes tertinggi maka peluang terpilihnya juga paling besar yaitu 38%.

Setelah dilakukan penentuan parent, tahap berikutnya adalah crossover [20]–[23] dimana pada tahap ini gene dari kedua parent akan ditukar-tukar untuk membentuk kromosom baru. Sama halnya seperti seleksi, crossover juga memiliki beberapa metode seperti N-Point, Cycle Crossover, 3 Parent Crossover. Tetapi untuk permasalahan yang sederhana, One Point Crossover dapat bekerja dengan baik.

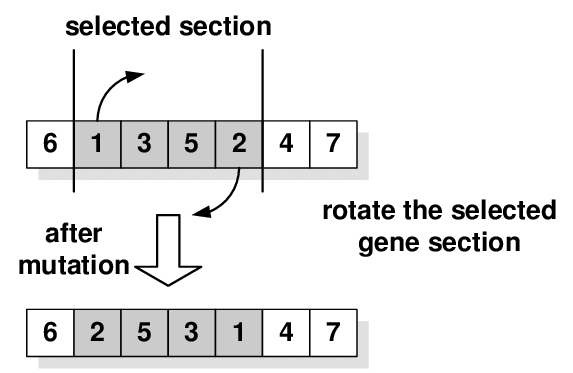


1. Contoh One Point Crossover

Dalam One Point Crossover dilakukan pemilihan 1 angka batas secara random dan dilakukan penukaran gene di kiri atau kanan pembatas untuk menghasilkan 2 individu baru (dapat dilihat contohnya pada gambar 5). Setelah proses crossover berakhir maka dilakukan proses seleksi dan crossover terus menerus hingga jumlah anak yang dibutuhkan tercapai.

1. MUTASI

Mutasi bukanlah tahap wajib yang harus dilakukan dalam algoritma genetik akan tetapi adanya mutasi dapat membantu menambahkan variasi kemungkinan bahkan dapat melepaskan diri dari local minima. Apabila digunakan mutasi dilakukan setelah crossover, dimana setiap anak hasil crossover memiliki peluang untuk bermutasi. Beberapa metode mutasi yang telah dikembangkan adalah Inversion Mutation, Polynomial Mutation [24], Gaussian Mutation, atau algoritma mutasi lainnya [25]. Untuk beberapa kasus sederhana, inversion mutation sudah cukup baik untuk membantu proses pembentukan individu baru.



1. Ilustrasi Inversion Mutation

Dalam Inversion Mutation, dipilih 2 gene secara acak, dimana kedua gene tersebut berfungsi sebagai pembatas. Semua gene diantara kedua pembatas akan dibalik urutannya sehingga menghasilkan individu yang berbeda sama sekali dari induknya (contoh dapat dilihat di gambar 6).

1. ELITISME

Setelah parent menghasilkan individu-individu baru maka proses berikutnya adalah elistisme [26], [27]. Pada tahap ini, algoritma genetik akan mengambil individu terbaik untuk diproses ke generasi berikutnya. Pengambilan individu terbaik dapat dilakukan dengan beberapa metode. Ada penelitian yang mengambil berdasarkan nilai fitnes akan tetapi ada juga yang hanya mengambil 3 parent dengan fitnes tertinggi, sisa populasi diisi dengan anak yang dihasilkan.

1. KONDISI BERHENTI

Hal terakhir yang penting dalam algoritma genetik adalah kondisi berhenti sehingga algoritma tidak berjalan terus tanpa batas. Terdapat beberapa solusi untuk kondisi berhenti, misalnya berhenti setelah sejumlah generasi, berhenti setelah individu di dalamnya sudah konvergen, atau berhenti setelah batas waktu tertentu. Hal ini disesuaikan dengan permasalahan dan lama waktu yang diijinkan untuk proses terjadi.

IMPLEMENTASI ALGORITMA GENETIK KE PERMAINAN SPLATTED

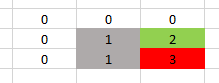
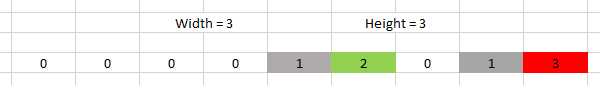
Dalam sub bab ini akan dibahas detail terkait penggunaan algoritma genetik pada permainan splatted mulai dari representrasi, nilai fitnes, hingga metode yang digunakan dalam setiap operasi algoritma genetik.

1. REPRESENTASI

Pada penelitian ini terdapat 2 model level generation, tile-based generation dan template-based generation. Masing-masing model memiliki proses kerjanya dan cara representasinya sendiri-sendiri.

**Tile-Based Generation**

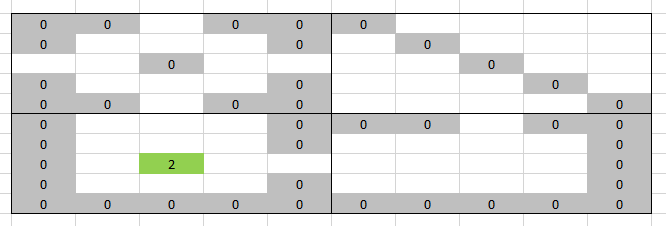
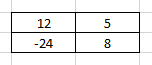
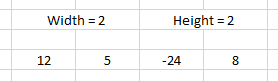
Pada model ini, representasi individu yang digunakan adalah array 1 dimensi dengan panjang luas level yang akan dibuat (Misalnya level berukuran 10x10 maka panjang individu adalah 100). Sedangkan masing-masing gene pada individu tersebut akan bernilai 0 hingga 3, dimana angka 0 melambangkan jalan, 1 melambangkan batu/hambatan, 2 melambangkan posisi bola spesial, dan 3 melambangkan posisi pemain. Gambar 7 merupakan contoh representasi tile-based apabila dirubah menjadi level yang dapat dipahami oleh pemain, dimana ukuran level adalah 3x3.



1. Contoh Representasi Tile Based menjadi sebuah map

**Template-Based Generation**

Sama hal nya dengan tile-based generation, template-based generation direpresentasikan dalam array 1 dimensi akan tetapi tidak seperti tile-based, nilai gene pada model ini bukanlah berisikan angka 0-3 yang melambangkan objek pada level melainkan bernilai -n hingga n, dimana n adalah jumlah template yang disiapkan. Nilai negatif disini berarti pada level di template ber nomor id x tersebut akan terdapat 1 bola spesial pada bagian tengah. Selain nilai dari gene, perbedaan kedua dengan representasi tile-based adalah panjang individu. Template-based memiliki panjang individu yang lebih pendek karena 1 template berukuran 5x5 tile. Misalnya level memiliki ukuran 10x10 maka panjang individu yang digunakan adalah 100/(5x5) atau 4.



1. Contoh Representasi Template Based menjadi sebuah map

Penelitian ini memiliki 3 jenis template dimana masing-masing jenis memiliki beberapa variasi. Ketiga jenis template tersebut antara lain:

1. Oneway Template

Oneway template berarti variasi template yang disediakan apabila diputar baik 90o, 180o, atau 270o akan tetapi menghasilkan level yang sama. Misalnya pada gambar 8, level yang dilambangkan dengan kode 12. Jenis template ini hanya disediakan 3 variasi saja.

1. Twoway Template

Twoway template berarti variasi template yang disediakan apabila diputar 90o atau 270o akan memiliki hasil level yang berbeda akan tetapi apabila diputar 180o maka akan menghasilkan level yang sama. Misalnya pada gambar 8, level yang dilambangkan dengan kode 5. Jenis template ini disediakan 8 variasi.

1. Fourway Template

Fourway template berarti variasi template yang disediakan apabila diputar 90o, 180o, atau 270o akan menghasilkan level yang berbeda. Misalnya pada gambar 8, level yang dilambangkan dengan kode 8 atau 24. Jenis template ini disediakan 5 variasi.

Total template yang disediakan pada penelitian ini adalah 39 template yang diperoleh berdasarkan 3 oneway template + (8 x 2[[5]](#footnote-5) twoway template) + (5 x 4[[6]](#footnote-6) fourway template). Gambar 8 merupakan contoh representasi template-based apabila dirubah menjadi level yang dapat dipahami oleh pemain, dimana ukuran level adalah 10x10.

1. OPERATOR ALGORITMA GENETIK

Seperti yang telah dibahas pada sub section III, algoritma genetik memiliki beberapa operator dengan beberapa algoritma pilihan. Penelitian ini telah mencoba algoritma-algoritma tersebut dan mencari algoritma yang paling cocok dan tepat dalam menyelesaikan pembuatan level yang menjadi fokus utama penelitian ini. Beberapa operator beserta pilihannya antara lain:

1. Seleksi Parent

Metode seleksi yang digunakan oleh penelitian ini adalah Roulette Wheel. Metode ini cocok digunakan pada penelitian ini mengingat semakin baik nilai fitnes maka semakin besar juga peluang terpilihnya sebuah individu atau calon level sebagai parent. Dimana level yang kurang baik umumnya kurang dapat dimainkan sehingga apabila terpilih akan menyebabkan generasi berikutnya juga kurang dapat dimainkan.

1. Crossover

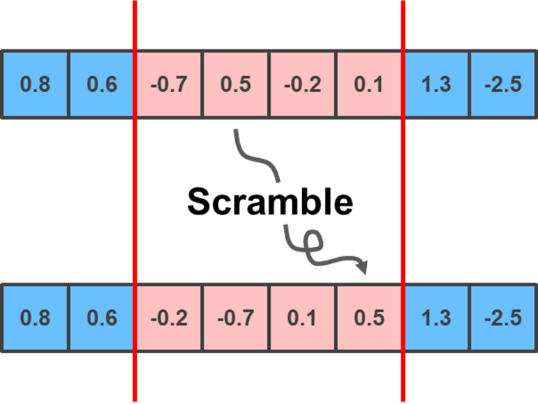


1. Visualisasi Uniform Crossover

Penelitian ini menggunakan algoritma uniform crossover sebagai operator. Dimana algoritma ini, akan memberikan peluang 50% untuk setiap gene dapat saling bertukar pada 2 parent terpilih. Gambar 9 merupakan contoh visualisasi uniform crossover. Algoritma ini cocok diterapkan karena pada penelitian ini 1 gene mewakili 1 tile/template. Sehingga level yang dihasilkan pada generasi berikutnya akan memiliki variasi yang baik.

1. Mutasi

Dengan alasan yang sama seperti pemilihan algoritma crossover, algoritma partial shuffle mutation atau scramble mutation dipilih dan digunakan pada penelitian ini. Algoritma ini memiliki lebih banyak peluang perubahan gene pada generasi berikutnya mengingat seluruh gene diantara 2 pembatas[[7]](#footnote-7) akan diacak urutannya. Gambar 10 merupakan contoh visualisasi scramble mutation.



1. Visualisasi Scramble Mutation
2. Elitisme

Metode elitisme yang digunakan pada penelitian ini adalah menggabungkan seluruh offspring (anak) dan sebagian parent yang memiliki nilai fitnes terbaik pada generasi sebelumnya. Misalnya minimal populasi yang harus dimiliki berjumlah 100 sedangkan offspring yang terbentuk berjumlah 80 maka 20 sisanya diambil dari parent yang memiliki nilai fitnes terbaik.

1. KONDISI BERHENTI

Kondisi berhenti dari algoritma genetik pada penelitian ini adalah dengan cara konvergensi. Apabila seluruh individu dalam 100 generasi terakhir sudah tidak memiliki perkembangan yang signifikan atau sudah konvergen maka iterasi algoritma genetik dihentikan. Hal ini dilakukan mengingat pembentukan level dilakukan saat loading awal permainan dan membutuhkan kecepatan yang baik agar pemain tidak menunggu terlalu lama. Walaupun tidak dipungkiri terkadang, pembentukan level membutuhkan waktu yang cukup lama (sekitar 20 detik) akan tetapi hal ini masih dalam toleransi waktu tunggu pemain.

*FUNGSI FITNES*

Fungsi fitnes merupakan fungsi yang penting untuk menentukan kebaikan sebuah individu pada algoritma genetik. Pada penelitian ini akan digunakan 5 fungsi fitnes untuk tile-based generation dan 4[[8]](#footnote-8) fungsi fitnes untuk template-based generation.

1. FITNESS JUMLAH BATU

Fungsi fitnes ini digunakan untuk mengatur kemunculan batu di dalam level. Tujuan utamanya agar sebuah level tidak hanya berisi batu saja atau bahkan tidak ada batu sama sekali. Beberapa parameter yang ditentukan terlebih dahulu adalah MinR (jumlah minimal batu dalam sebuah level) dan MaxR (jumlah maksimal batu dalam sebuah level).

(1)

Dapat dilihat pada rumus 1, bahwa nilai m akan dijadikan negatif ketika jumlah batu kurang (lebih kecil dari MinR) atau terlalu banyak (lebih besar dari MaxR). Sedangkan apabila berada dalam range, nilai m akan dijadikan 0 yang artinya tidak ada kendala pada jumlah batu dalam level tersebut.

(2)

Agar nilai fitnes tidak bernilai negatif maka dilakukan normalisasi pada nilai m. Rumus 2 adalah rumus untuk mencari pembagi agar normalisasi berjalan dengan baik. Terdapat 2 cara mencari nilai bagi untuk normalisasi yaitu mengurangkan area dengan jumlah maksimal batu atau mengambil secara langsung jumlah minimal batu disaat hasil pengurangan area dan jumlah maksimal batu lebih kecil dari jumlah minimal batu.

(3)

Rumus 3 adalah rumus normalisasi yang digunakan pada fungsi fitnes ini. Melalui fungsi ini dapat dipastikan bahwa nilai F tidak akan pernah menjadi negatif. Fungsi fitnes ini hanya digunakan pada tile-based generation mengingat pada template-based jumlah batu ditentukan oleh komposisi template yang digunakan sehingga apabila fungsi ini diterapkan akan mengganggu variasi template yang ada.

1. FITNESS UKURAN KELOMPOK BATU

Fungsi fitnes ini digunakan untuk menghitung ukuran dari kelompok batu yang ada di sebuah level. Tujuan utama dari fungsi fitnes ini adalah memastikan tidak ada kelompok batu yang terlalu besar ataupun terlalu kecil. Sama halnya dengan fungsi fitnes sebelumnya, rumus 1 dan rumus 2 tetap digunakan dalam melakukan perhitungan fungsi fitnes ini. Akan tetapi berbeda dengan fungsi fitnes sebelumnya yang dihitung untuk 1 level, pada fungsi fitnes ini kedua rumus dihitung untuk setiap kelompok batu yang ditemukan.

(4)

Setelah nilai m dan M dari seluruh kelompok batu selesai dihitung maka rumus 6 akan digunakan untuk menentukan nilai fitnes ukuran kelompok batu. Sama halnya dengan fitnes jumlah batu, fitnes ukuran kelompok batu juga hanya digunakan pada tile-based generation dan tidak digunakan pada template-based.

1. FITNESS LUAS AREA YANG BISA DILEWATI

Fungsi fitnes ini digunakan untuk menghitung seberapa besar area yang dapat dilalui oleh pemain. Semakin banyak area yang terhubung satu dengan yang lain maka semakin baik level tersebut. Tujuan utama dari fungsi ini adalah untuk memastikan hanya ada sedikit area dalam level yang tidak dapat diakses oleh pemain karena permainan splatted tidak memiliki kemampuan untuk menembus atapun menghancurkan rintangan.

(5)

Rumus 5 merupakan fungsi fitnes yang digunakan untuk melihat kebaikan sebuah level berdasarkan areanya. Dimana nilai a terbesar adalah luas area terbesar yang terhubung satu dengan yang lain sedangkan nilai a total adalah jumlah objek pada level yang bukan batu.

1. FITNESS BOLA SPESIAL YANG BISA DIRAIH

Fungsi fitnes ini bertujuan memastikan bahwa setiap bola spesial yang terdapat pada level dapat dicapai oleh pemain. Akan tetapi bukan untuk seluruh pemain melainkan hanya untuk pemain terdekat dari bola saja. Sehingga kemunculan bola spesial di level paling tidak dapat diambil minimal oleh pemain terdekat.

(6)

Rumus 6 adalah rumus nilai fitnes yang digunakan untuk memastikan seluruh bola spesial dapat dijangkau oleh pemain terdekat. Dimana P total adalah jumlah bola spesial dan P akses adalah jumlah bola spesial yang dapat diakses oleh pemain. Sebuah bola spesial dikatakan dapat diakses apabila dari pemain terdekat ke bola spesial, dilakukan pencarian path menggunakan algoritma A\* dapat mengembalikan sebuah path tanpa hambatan.

1. FITNESS RASIO BOLA SPESIAL

Fungsi fitnes rasio bola sepsial digunakan untuk memastikan adanya bola spesial dalam sebuah level. Beberapa parameter yang ditentukan terlebih dahulu adalah MinP (jumlah minimal bola spesial dalam sebuah level) dan MaxP (jumlah maksimal bola spesial dalam sebuah level).

(7)

Dapat dilihat pada rumus 7, bahwa nilai m akan dijadikan negatif ketika jumlah bola spesial kurang (lebih kecil dari MinP) atau terlalu banyak (lebih besar dari MaxP). Sedangkan apabila berada dalam range, nilai m akan dijadikan 0 yang artinya tidak ada kendala pada jumlah bola spesial dalam level tersebut.

(8)

Agar nilai fitnes tidak bernilai negatif maka dilakukan normalisasi pada nilai m. Rumus 8 adalah rumus untuk mencari pembagi agar normalisasi berjalan dengan baik. Terdapat 2 cara mencari nilai bagi untuk normalisasi yaitu mengurangkan area dengan jumlah maksimal bola spesial atau mengambil secara langsung jumlah minimal bola spesial disaat hasil pengurangan area dan jumlah maksimal bola spesial lebih kecil dari jumlah minimal batu.

(9)

Rumus 9 adalah rumus normalisasi yang digunakan pada fungsi fitnes ini. Melalui fungsi ini dapat dipastikan bahwa nilai F tidak akan pernah menjadi negatif.

1. FITNESS VARIASI TEMPLATE

Fungsi fitnes terakhir yang digunakan pada penelitian ini adalah variasi template. Tujuan dari fitnes ini adalah menghindari beberapa kali kemunculan template yang sama dalam sebuah level. Sehingga diharapkan level yang terbentuk memiliki variasi template yang tinggi dan bukan 1 atau 2 template diulang berkali-kali.

(10)

Dapat dilihat pada rumus 10 bahwa nilai x akan dijadikan negatif ketika jumlah kemunculan sebuah template melebihi batas toleransi yang diberikan. Sedangkan jika masih di dalam toleransi maka nilai x akan dijadikan 0. Perhitungan rumus 10 ini dilakukan untuk setiap template yang ditemukan.

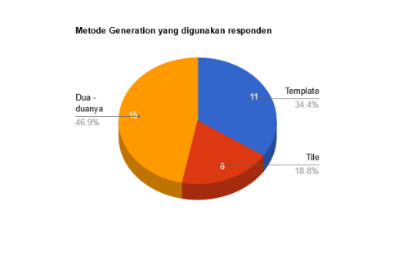
(11)

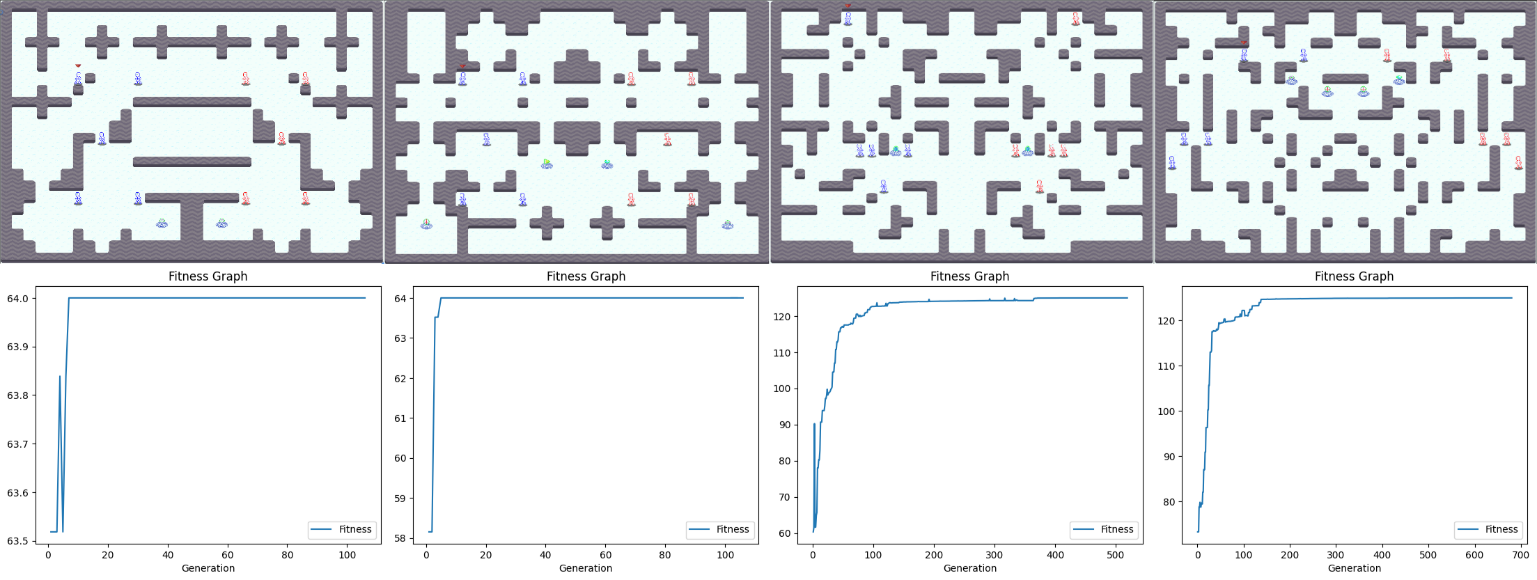
Setelah seluruh template pada sebuah level dihitung dengan rumus 10. Rumus 11 digunakan untuk menghitung nilai fitnes dengan melakukan kuadrat rata-rata nilai x. Apabila nilai yang dihasilkan negatif maka nilai fitnes dijadikan 0 agar tidak mengganggu nilai dari fungsi fitnes lainnya. Fungsi fitnes ini khusus digunakan untuk template-based generation.

UJI COBA

Penelitian ini menggunakan 2 teknik ujicoba. Pertama adalah dengan menggunakan user acceptance atau kuisioner. Sedangkan teknik kedua adalah analisa level yang dihasilkan oleh pakar permainan.

1. USER ACCEPTANCE



1. Pemilihan Level Generation oleh Responden
2.  Level hasil Map Generation dan Fitnessnya

Kuisioner diisi oleh 32 orang dengan profile responden adalah pemain game dengan rentang umur 18-22 tahun, memiliki jam bermain game minimal 2-3 jam per hari, dan pernah memainkan permainan MOBA atau permainan real time strategy 2D seperti bomberman. Gambar 11 merupakan persentase responden dalam memilih mode level generation. 46% memilih mencoba kedua mode dan sisanya hanya mencoba salah satu mode saja.

A graph with different colored bars

Description automatically generated

1. Skor yang diberi Responden

Sedangkan gambar 13, menunjukan rating responden baik yang telah mencoba kedua mode ataupun yang mencoba salah satu saja. Hasil kuisioner menunjukan bahwa hanya 1 responden yang merasa level yang dihasilkan kurang sedangkan 31 orang lainnya menyatakan bahwa level yang dihasilkan memiliki rating minimal 3. Melalu gambar 13 ini, dapat disimpulkan bahwa level yang dihasilkan memenuhi harapan pemain.

1. ANALISA LEVEL

Pada ujicoba analisa level oleh pakar, dilakukan generate beberapa level baik tile-based generation atau template-based generation. Pakar dengan profile pemain game MOBA yang memiliki rating permainan cukup tinggi dan waktu bermain per harinya 5-6 jam, diminta untuk menganalisa beberapa level yang ada. Gambar 12 (dari kiri ke kanan) adalah gambar level terbaik dengan ukuran 20 x 30 tile untuk 2 contoh template-based generation dan 2 contoh tile-based generation. Dari 4 contoh level terbaik tersebut dapat dilihat bahwa seluruh level memiliki area terbuka yang dapat digunakan pemain untuk berstrategi, tidak ada tempat yang tertutup dan tidak dapat diakses, bola spesial tersebut dengan baik dan tidak terlalu banyak, serta secara estetika terlihat cukup baik dan menarik untuk dimainkan.

Sedangkan dari segi analisa level berdasarkan nilai fitnesnya. Kedua level yang dihasilkan oleh template-based generation memiliki kecepatan yang tinggi dalam memperoleh maksimal fitnes atau menghasilkan level yang baik. Sedangkan dua level yang dihasilkan oleh tile-based generation walaupun dapat menghasilkan nilai fitnes yang baik akan tetapi membutuhkan waktu lebih lama karena pada tile-based generation perubahan yang terjadi dilakukan perlahan-lahan atau per tile.

KESIMPULAN

Pada penelitian menggunakan permainan splatted, dapat disimpulkan bahwa algoritma genetik bekerja dengan baik dalam menghasilkan level yang baik, dapat dimainkan, dan sesuai dengan harapan pemain. Akan tetapi kelemahan utama dari level generation menggunakan algoritma genetik adalah besarnya ketergantungan hasil yang diperoleh dengan fungsi fitnes yang digunakan. Level yang dihasilkan akan menjadi baik ketika fungsi fitnes yang digunakan tepat dan sesuai. Hal ini membutuhkan waktu yang tidak sedikit dalam mencoba dan meningkatkan berbagai varian fungsi fitnes.

UCAPAN TERIMA KASIH

Terima kasih kami ucapkan kepada Institut Sains dan Teknologi Terpadu Surabaya yang sudah mendukung dan memfasiltiasi penelitian ini.

COPYRIGHT

[Creative Commons License](https://creativecommons.org/licenses/by-nc-sa/4.0/)This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

DAFTAR PUSTAKA

[1] J. Schell, *The Art of Game Design: A Book of Lenses, Third Edition*. CRC Press, 2019.

[2] E. Adams, *Fundamentals of Game Design*. Pearson Education, 2010.

[3] M. Moore, *Basics of Game Design*. CRC Press, 2016.

[4] N. A. Barriga, *A Short Introduction to Procedural Content Generation Algorithms for Videogames*. 2018.

[5] V. Kraner, I. Fister jr, and L. Brezočnik, “Procedural Content Generation of Custom Tower Defense Game Using Genetic Algorithms,” 2021, pp. 493–503.

[6] S. Putra and W. Istiono, “Implementation Simple Additive Weighting in Procedural Content Generation Strategy Game,” *vol*, vol. 4, pp. 9–18, 2022.

[7] E. Frank and N. Olsson, “Procedural city generation using Perlin noise.” 2017.

[8] N. Sainio, “TERRAIN GENERATION ALGORITHMS,” 2023.

[9] A. Lambora, K. Gupta, and K. Chopra, “Genetic Algorithm- A Literature Review,” in *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*, 2019, pp. 380–384, doi: 10.1109/COMITCon.2019.8862255.

[10] L. Haldurai, T. Madhubala, and R. Rajalakshmi, “A study on genetic algorithm and its applications,” *Int. J. Comput. Sci. Eng*, vol. 4, no. 10, pp. 139–143, 2016.

[11] S. Katoch, S. S. Chauhan, and V. Kumar, “A review on genetic algorithm: past, present, and future,” *Multimed. Tools Appl.*, vol. 80, no. 5, pp. 8091–8126, 2021, doi: 10.1007/s11042-020-10139-6.

[12] H. Armanto, H. A. Rosyid, M. Muladi, and G. Gunawan, “Evolutionary Algorithm in Game – A Systematic Review,” *Kinet. Game Technol. Inf. Syst. Comput. Network, Comput. Electron. Control*, May 2023, doi: 10.22219/kinetik.v8i2.1714.

[13] W. Alfonsus, A. Hendrawan, and T. J. Gunawan, “Focused Web Crawler Using Genetic Algorithms and Symbiotic Organism Search,” 革新的コンピューティング・情報・制御に関する速報, vol. 15, no. 12, p. 1345, 2021.

[14] H. Armanto, K. Setiabudi, and C. Pickerling, “Komparasi Algoritma WOA, MFO dan Genetic pada Optimasi Evolutionary Neural Network dalam Menyelesaikan Permainan 2048,” *J. Inov. Teknol. dan Edukasi Tek.*, vol. 1, no. 9, pp. 676–684, 2021.

[15] H. Armanto, R. D. Putra, and C. Pickerling, “MVPA and GA Comparison for State Space Optimization at Classic Tetris Game Agent Problem,” *Inf. J. Ilm. Bid. Teknol. Inf. dan Komun.*, vol. 7, no. 1, pp. 73–80, 2022.

[16] unacademy, “Analytical Geometry-Two Dimensions-Distance Formula.” https://unacademy.com/content/nda/study-material/mathematics/analytical-geometry-two-dimensions-distance-formula/ (accessed Nov. 02, 2023).

[17] J. Y. Setiawan, D. E. Herwindiati, and T. Sutrisno, “Algoritma Genetika Dengan Roulette Wheel Selection dan Arithmetic Crossover Untuk Pengelompokan,” *J. Ilmu Komput. dan Sist. Inf.*, vol. 7, no. 1, pp. 58–64, 2019.

[18] S. Prayudani, A. Hizriadi, E. B. Nababan, and S. Suwilo, “Analysis effect of tournament selection on genetic algorithm performance in traveling salesman problem (TSP),” in *Journal of Physics: Conference Series*, 2020, vol. 1566, no. 1, p. 12131.

[19] S. L. Yadav and A. Sohal, “Comparative study of different selection techniques in genetic algorithm,” *Int. J. Eng. Sci. Math.*, vol. 6, no. 3, pp. 174–180, 2017.

[20] J. L. Pachuau, A. Roy, and A. Kumar Saha, “An overview of crossover techniques in genetic algorithm,” *Model. Simul. Optim. Proc. CoMSO 2020*, pp. 581–598, 2021.

[21] P. Kora and P. Yadlapalli, “Crossover operators in genetic algorithms: A review,” *Int. J. Comput. Appl.*, vol. 162, no. 10, 2017.

[22] A. Malik, “A study of genetic algorithm and crossover techniques,” *Int. J. Comput. Sci. Mob. Comput.*, vol. 8, no. 3, pp. 335–344, 2019.

[23] L. Manzoni, L. Mariot, and E. Tuba, “Balanced crossover operators in genetic algorithms,” *Swarm Evol. Comput.*, vol. 54, p. 100646, 2020.

[24] B. H. Abed-alguni, “Island-based cuckoo search with highly disruptive polynomial mutation,” *Int. J. Artif. Intell.*, vol. 17, no. 1, pp. 57–82, 2019.

[25] A. Hassanat, K. Almohammadi, E. Alkafaween, E. Abunawas, A. Hammouri, and V. B. S. Prasath, “Choosing mutation and crossover ratios for genetic algorithms—a review with a new dynamic approach,” *Information*, vol. 10, no. 12, p. 390, 2019.

[26] G. Guariso and M. Sangiorgio, “Improving the performance of multiobjective genetic algorithms: An elitism-based approach,” *Information*, vol. 11, no. 12, p. 587, 2020.

[27] H. Du, Z. Wang, W. E. I. Zhan, and J. Guo, “Elitism and distance strategy for selection of evolutionary algorithms,” *IEEE Access*, vol. 6, pp. 44531–44541, 2018.

1. Baik dalam segi waktu, aset, hingga tenaga desainer yang membuatnya [↑](#footnote-ref-1)
2. Yang diberi nama Splatted [↑](#footnote-ref-2)
3. Umumnya Array 1 dimensi akan tetapi bergantung pada permasalahan bisa dirubah menjadi array multi dimensi [↑](#footnote-ref-3)
4. Gambar 3 merupakan contoh rumus jarak yang dapat digunakan untuk permasalahan ini [↑](#footnote-ref-4)
5. 1 variasi akan menghasilkan 2 level yang berbeda ketika diputar [↑](#footnote-ref-5)
6. 1 variasi akan menghasilkan 4 level yang berbeda ketika diputar [↑](#footnote-ref-6)
7. Dimana penempatannya juga dilakukan secara acak [↑](#footnote-ref-7)
8. 3 fungsi fitnes diantaranya sama dengan fungsi fitnes yang digunakan oleh tile-based generation [↑](#footnote-ref-8)