[[1]](#footnote-1)

Procedural Map Generation untuk game “Splatted”

Lukky Hariyanto, *Departemen Informatika Institut Sains dan Teknologi Terpadu Surabaya*, Hendrawan Armanto, S.Kom., M.Kom., *Departemen Informatika Institut Sains dan Teknologi Terpadu Surabaya*,

*Abstrak*—Game merupakan aspek hiburan yang sekarang ini merajalela di seluruh dunia, dimana video game telah menjadi salah satu industri hiburan terbesar melebihi buku, film dan musik. Tetapi itu disebabkan oleh jumlah talenta yang diperlukan untuk membuat sebuah game, diisi oleh beberapa desainer / direktur yang memimpin pekerjaan yang diperlukan seperti direktur seni, direktur lagu, direktur efek suara, deasainer mekanik game dan masih banyak lagi, dengan salah satu aspek yang tidak kalah penting tapi tidak terlalu terkenal adalah desain level. Dalam tugas akhir ini, akan dibuat sebuah game yang mencoba memakai Level Generation. Game ini adalah Splatted, sebuah game 5v5 dimana setiap orang akan bertarung dengan mengambil salju di tanah, membuat nya menjadi sebuah bola salju, lalu melempar bola ke tim musuh sekaligus menangkap atau menghindari bola – bola yang datang ke tim nya. Di dalam game ini juga terdapat beberapa bola spesial yang bisa membuat permainan lebih menarik dengan menambah jumlah bola yang bisa dilempar oleh setiap orang. Lalu untuk level generation yang dipakai akan menggunakan algoritma genetik, sebuah algoritma yang didasarkan pada teori evolusi Darwin, dimana hewan – hewan yang memiliki fitur tertentu akan memiliki keunggulan dalam bertahan hidup dan beranak. Menggunakan itu akan dibuat 2 jenis pembuatan level, yaitu Tile Generation yang membuat level berdasarkan isi dari setial tile di level atau Template Generation dimana setiap tile 5x5 akan direpresentasikan menggunakan sebuah template yang telah dibuat menggunakan tangan. Dengan ini akan diberi 3 opsi ukuran level kepada player, yaitu kecil (20 x 15 tile), sedang (30 x 20) tile dan besar (40 x 25 tile).Dari game yang telah dibuat ini disimpulkan bila Procedural Map Generation bisa dipakai di kasus – kasus tertentu dalam sebuah game dan bekerja dengan baik, tapi salah satu saran yang penting untuk diingat setiap pemakai dari Procedural Map Generation adalah untuk memasukkan unsur – unsur buatan tangan kedalam level – level yang dibuat karena bila tidak maka level akan bersifat membosankan, dan dikarenakan memasukkan unsur – unsur manusiawi akan membuat level jauh lebih unik dan menarik untuk dimainkan.

*Kata Kunci*—Algoritma Genetik, Game, Procedural Map Generation.

# Pendahuluan

D

esain level telah menjadi salah satu komponen utama dari membuat sebuah game, dimana meskipun komponen ini tidak terlalu tampak ketimbang peran – peran lain seperti direktur seni atau desain musuh, desain level merupakan sebuah peran yang tidak kalah pentingnya dalam membuat game. Tetapi seperti peran – peran lain, mendesain level memakan waktu dan tenaga yang signifikan. Sebagai contoh, game Live Service seperti Valorant, Apex Legends, Fortnite, Overwatch 2 dan banyak lagi memerlukan desainer map untuk selalu bekerja membuat map baru agar para pemain tidak bosan bermain di map yang sama terus. Alternatif lain dari membuat level secara manual adalah untuk membiarkan game tersebut membuat level sendiri dengan proses yang dinamakan Procedural Map Generation[1]-[4].

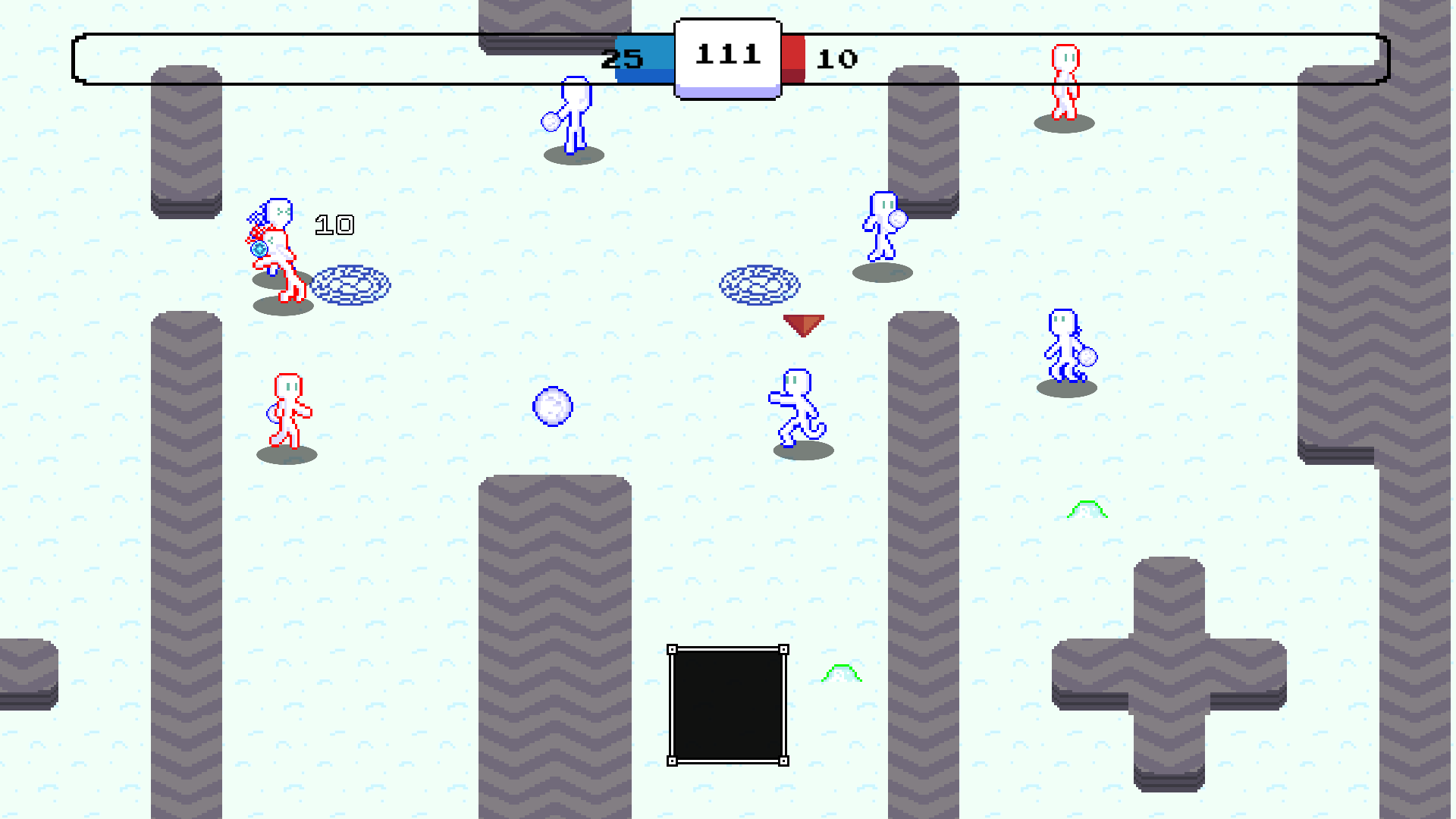
Procedural Map Generation adalah sebuah proses dimana sebuah game menggunakan sebuah algoritma untuk membuat level. Contoh populer dari game – game yang menggunakan Procedural Map Generation adalah Minecraft, No Man’s Sky, Terraria dan Deep Rock Galactic. Seluruh game tersebut membuat level mereka sendiri dengan algoritma seperti Perlin Noise untuk Minecraft atau Fractal Terrain Generation untuk Terraria. Untuk tugas akhir ini, kita akan mencoba menggunakan algoritma genetik[5]-[7] untuk membuat level dalam Game Splatted.

Algoritma genetik adalah sebuah algoritma yang didasarkan pada teori evolusi Darwin, dimana spesies yang berevolusi untuk beradaptasi mengikuti lingkungannya akan mendapat keuntungan dalam bertahan hidup dan melahirkan anak – anak. Dengan teori tersebut, algoritma genetik adalah sebuah algoritma dimana ada banyak solusi terhadap masalah optimisasi yang dihadapi, dimana solusi – solusi terbaik akan membuat solusi – solusi baru didasarkan dari solusi – solusi terbaik sebelumnya. Dan proses ini akan diulang terus hingga solusi tidak bisa berkembang lagi atau algoritma telah berjalan untuk waktu yang cukup lama.

Dengan itu, tujuan dari Tugas Akhir ini adalah untuk mencoba membuat sebuah game dengan Procedural Map Generation yang menggunakan algoritma genetik, untuk melihat apabila algoritma genetik dapat digunakan untuk membuat level – level game dan melihat apabila algoritma genetik merupakan sebuah alternatif yang sama bagusnya ketimbang algoritma – algoritma yang lain.

# Game “Splatted”

Sebelum memasuki algoritma genetik, perlu diketahui apa itu Splatted dan peran algoritma genetik dalam membuat game ini. Splatted adalah sebuah game 5 vs 5 dimana 2 kelompok bermain perang salju untuk mendapat poin tertinggi dari kedua kelompok tersebut. Player bisa melempar bola ke tim lawan untuk mendapatkan skor, dan player juga bisa menangkap atau menghindari bola yang dilempar oleh tim lawan untuk mencegah tim lawan untuk mendapat skor. Dan siapapun yang mengisi bar skor duluan atau memiliki bar skor yang lebih tinggi saat waktu habis memenangkan permainannya. Tidak hanya itu, player memiliki aksi khusus yang hanya bisa dilakukan player, yaitu fakeout, dimana bila ada bot yang melihat player melakukan fakeout akan mencoba menangkap bola yang tidak dilempar.



Gambar. 1. Cuplikan layar game Splatted

Dalam game ini juga terdapat bola spesial yang bila diambil memiliki perilaku khusus dibandingkan bola biasa. Masing – masing dengan efek – efek khusus tersendiri untuk membuat game lebih menarik. Berikut adalah bola – bola beserta efek – efeknya :

Tabel I

Bola – bola spesial dalam Splatted

|  |  |  |
| --- | --- | --- |
| Nama | Tampilan | Kegunaan |
| Ice Piercer |  | Bisa Menembus Player – player yang dilewati |
| Snow-A-Rang |  | Bola bisa kembali ke pelempar |
| Explod-o-Ball |  | Bola bisa meledak, mengenai seluruh player di area ledakan |
| Freezing Winter |  | Bola Bisa memperlambat player yang kena |
| Stone Auger |  | Bisa menembus player dan dinding lalu pecah menjadi 3 |

Terakhir game ini memiliki sebuah tutorial untuk mengajari player cara bermain game ini, dikarenakan game dengan cara bermain ini tidak umum. Tutorial mengandung beberapa aksi dasar seperti bergerak, dan juga mengajari aksi khusus game ini seperti mengambil bola, menangkap bola, melempar bola dan fakeout, tidak lupa juga membiarkan player mencoba bola spesial Splatted di akhir tutorial.

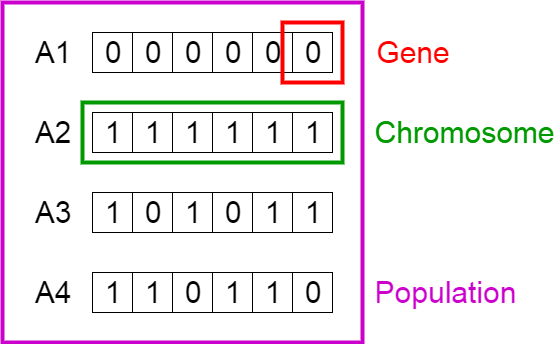
Peran algoritma genetik dalam membuat game ini adalah untuk membentuk level – level sesuai dengan ukuran dari level yang diminta oleh player. Terdapat 4 komponen yang bisa mengisi sebuah tile di dalam level Splatted, yaitu tile kosong, batu, bola spesial dan player. Dengan 4 komponen tersebut, dibuat 2 jenis Map Generation, yaitu Tile Generation dimana setiap tile memiliki isi tersendiri dan Template Generation dimana setiap 5X5 tile diisi menggunakan template yang telah dibuat oleh tangan.

# Algoritma Genetik

Dengan game telah dijelaskan, sekarang memasuki algoritma genetik[8]-[11] , dimana terdapat beberapa tahap yang perlu dilakukan untuk menemukan solusi terbaik untuk optimisasi, dalam kasus ini, isi dari level yang akan dibuat oleh game Splatted. Tetapi sebelum memasuki tahap – tahap dari membuat algoritma genetik perlu diketahui beberapa terminologi dalam algoritma genetik:

## Terminologi Algoritma Genetik

Dalam algoritma genetik, seluruh konsep – konsep yang diterapkan mengambil inspirasi dari kromosom DNA kita, dimana sebuah representasi dari sebuah solusi dinamakan sebuah kromosom, lalu di dalam kromosom tersebut terdapat variabel – variabel yang membentuk solusi dinamakan sebuah Gene. Dan terakhir, sebuah kelompok yang berisi banyak kromosom dinamakan sebuah populasi dimana sebuah populasi ke – n juga bisa dipanggil generasi n.



Gambar. 2. Terminologi yang digunakan pada algoritma genetik

## Menentukan representasi dari solusi

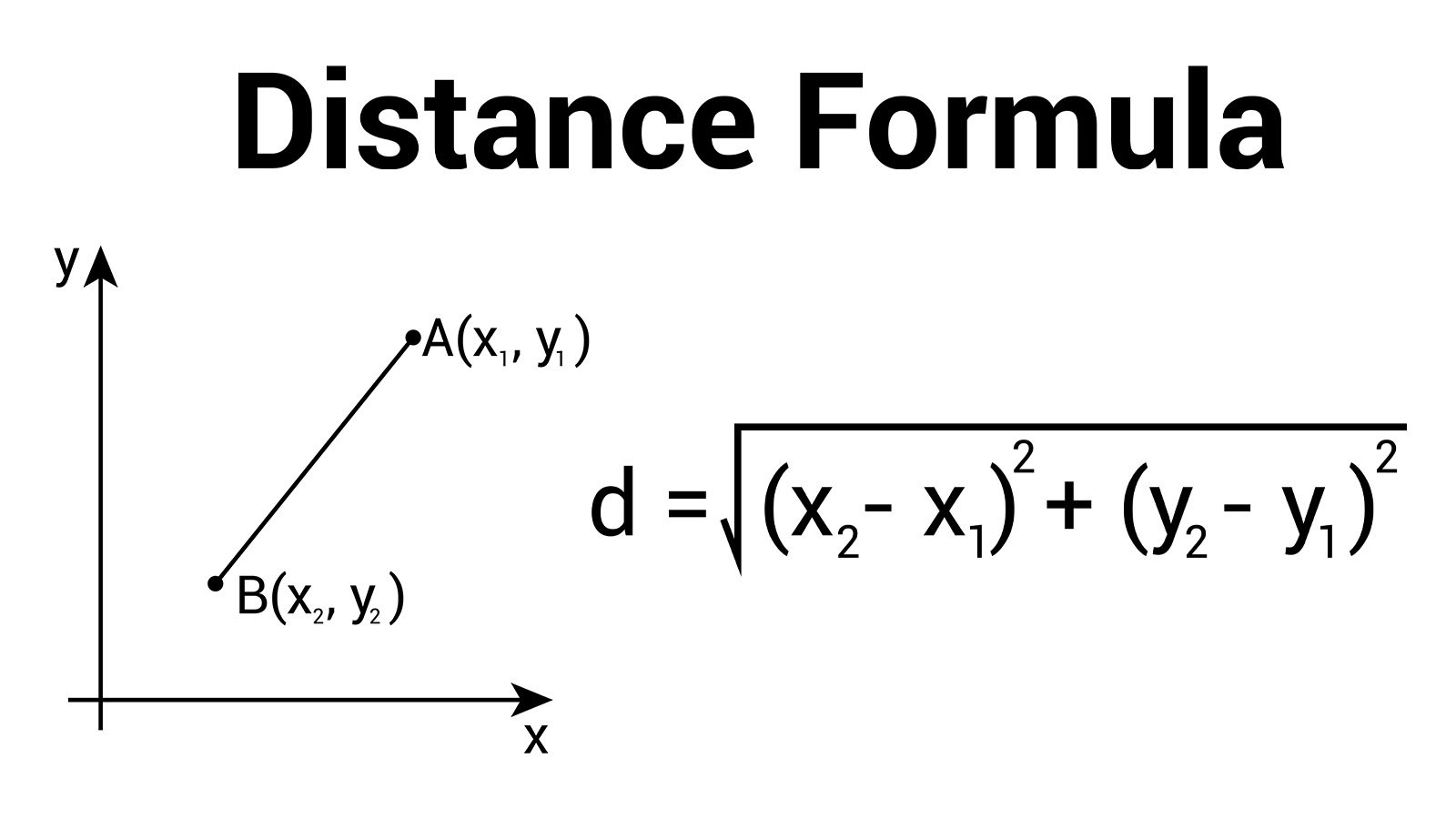
Tahap paling pertama dalam melakukan algoritma Genetik adalah untuk menentukan representasi dari solusi yang akan kita pakai untuk permasalah kita. Solusi dari masalah akan diubah bentuknya menjadi sebuah tipe data agar bisa dibaca oleh algoritma kita. Salah satu contoh permasalahan yang sederhana adalah mencari garis terpanjang yang bisa dibuat dalam sebuah persegi. Dalam kasus ini diperlukan 2 titik, dengan masing – masing titik memerlukan nilai x dan y. Sehingga dikarenakan ada 2 titik, solusi dari permasalahan ini akan direpresentasikan dengan sebuah kromosom berisi 4 gene, 2 gene untuk x dan y titik pertama dan 2 lagi untuk x dan y titik kedua.

## Pembuatan Populasi Pertama

Untuk memulai proses algoritma genetik, kita memerlukan populasi pertama untuk berperan sebagai kakek – kakek buyut dari seluruh kromosom yang akan mendatang. Di tahap ini, algoritma genetik akan membuat sebuah populasi berisi kromosom – kromosom dengan jumlah yang telah ditentukan, masing – masing dengan Gene yang benar – benar acak di dalam batasan yang telah ditetapkan. Setiap Gene akan memiliki nilai yang berbeda diantara satu sama lain dengan tujuan untuk memperbesar variasi dari anak – anak kromosom yang bisa lahir pada generasi selanjutnya.

## Perhitungan fitness

Dengan populasi pertama dibuat, tahap selanjutnya dalam proses algoritma genetik adalah untuk menghitung Fitness dari seluruh kromosom yang tersedia. Fitness adalah sebuah nilai yang menandakan apabila sebuah kromosom memiliki solusi yang bagus dengan permasalahan yang diberikan, dimana semakin tinggi fitness yang didapat menandakan semakin bagus kualitas kromosom dalam menangani masalah yang diberikan. Setiap kromosom akan dihitung fitnessnya menggunakan rumus – rumus yang ditetapkan sesuai kebutuhan yang diperlukan, untuk contohnya kita gunakan kembali contoh garis terpanjang dalam sebuah persegi diatas. Seperti yang kita tahu garis terpanjang dalam sebuah persegi adalah garis diagonal dimana titik – titik diletakkan di ujung persegi yang berlawanan. Tetapi algoritma tidak mengetahui ini dan inilah peran dari sebuah fitness.

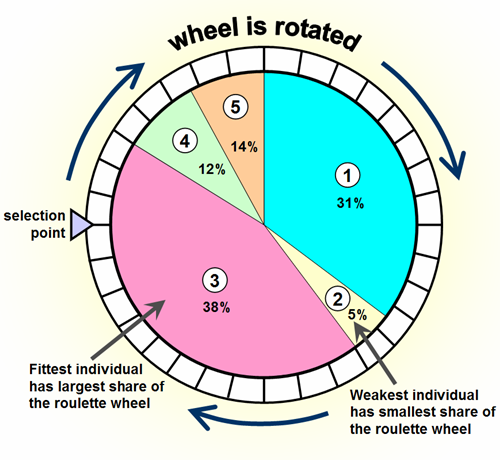


Gambar. 3. Rumus jarak antara 2 titik

Di gambar 3 terdapat rumus yang dapat digunakan untuk mencari jarak antara 2 titik. Variabel d diatas dapat digunakan sebagai nilai fitness kita, dimana kita mencari panjang garis terpanjang dalam sebuah persegi, dan bila garis semakin panjang, maka fitness akan semakin membesar, menandakan bahwa solusi ini adalah solusi yang diinginkan

## Seleksi & Crossover

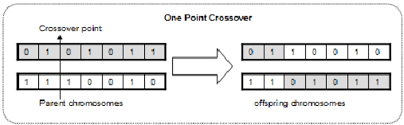
Setelah selesai menghitung nilai Fitness dari seluruh populasi, tahap selanjutnya adalah untuk melakukan tahap seleksi, dimana dalam tahap ini digunakan algoritma yang menentukan kromosom – kromosom yang akan dipilih untuk tahap Crossover berdasarkan fitness yang dimiliki kromosom – kromosom tersebut. Terdapat banyak algoritma yang bisa dipakai, seperti Tournament Selection, Ranking Selection, dan masih banyak lagi [12]–[15], tetapi seluruh algoritma memiliki tujuan utama untuk memprioritaskan kromosom – kromosom dengan fitness tertinggi, dengan contoh yang paling populer adalah Roulette Wheel Selection.



Gambar. 4. Ilustrasi Roulette Wheel Selection

Dalam Roulette Wheel Selecrtion, setiap kromosom memiliki kesempatan untuk dipilih, tetapi semakin besar fitness sebuah kromosom dibandingkan dengan kromosom – kromosom yang lain, maka semakin besar bagian yang diperoleh dalam Roulette Wheel, sehingga semakin besar juga kesempatan Roulette Wheel memilih kromosom dengan Fitness yang besar. Seperti di gambar 4 diatas, kromosom 3 yang merupakan kromosom dengan Fitness tertinggi memiliki kesempatan 38% untuk dipilih, diikuti dengan kromosom 3, tetapi kromosom 2,4 dan 5 memiliki kesempatan yang kecil dikarenakan Fitness mereka yang kecil, membuat mereka kesulitan dipilih.

Menggunakan metode seleksi yang dipilih, diambil 2 kromosom yang akan menjadi orang tua dari sepasang kromosom baru. Tahap selanjutnya setelah memilih 2 kromosom menggunakan seleksi adalah tahap Crossover [16]-[19], dimana Gene – Gene di dalam kedua orang tua kromosom ditukar – menukar untuk membuat sepasang anak kromosom baru. Ini berfungsi untuk membuka posibilitas untuk mencari solusi atau kromosom yang lebih bagus daripada seluruh kromosom yang ada di generasi sekaran ini. Sama seperti Seleksi, terdapat banyak metode untuk melakukan Crossover, seperti N-Point, Cycle Crossover, ada juga Crossover yang menggunakan 3 parent ketimbang 2, tetapi untuk Crossover yang sederhana, One Point Crossover dapat melakukan tugas ini dengan baik.



Gambar. 5. Contoh One Point Crossover

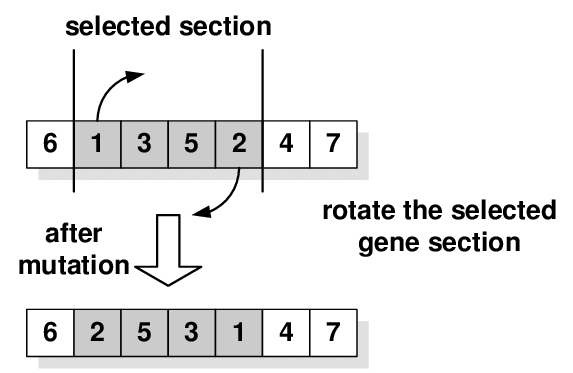
Dalam One Point Crossover, dipilih secara acak sebuah pembatas diantara 2 Gene. Tahap selanjutnya dari One Point Crossover adalah untuk menukar entah sisi kanan atau sisi kiri dari pembatas tersebut membuat 2 anak kromosom baru dengan isi Gene yang tercampur seperti pada gambar 5 diatas.

Setelah tahap Crossover dilakukan, maka akan dilakukan lagi tahap Seleksi dan Crossover. Kedua tahap tersebut akan diulangi terus hingga jumlah kromosom yang telah dibuat dapat memenuhi sebuah populasi.

## Mutation

Tahap opsional yang sangat membantu setelah membuat anak - anak kromosom dari tahap Crossover diatas adalah untuk melakukan Mutation untuk menambah variasi dari setiap kromosom yang dilahirkan. Mutation atau mutasi dalam algoritma genetik adalah sebuah tahap dimana Gene – Gene dalam sebuah kromosom dapat diubah – ubah, entah isinya yang diubah atau urutan Gene-nya. Metode – metode Mutation tidak sedikit juga, terdapat banyak Mutation seperti

Polynomial Mutation, Gaussian Mutation, dan banyak lagi Mutation sejauh yang bisa dipikirkan [20] – [21]. Untuk Mutation yang sederhana, bisa dilihat Inversion Mutation :



Gambar. 6. Visualisasi Inversion Mutation

Dalam Inversion Mutation, diletakkan 2 pembatas diantara 2 Gene secara acak, lalu untuk seluruh Gene yang berada diantara 2 pembatas tersebut, balik urutan dari Gene yang terpengaruh, seperti Gene yang memiliki urutan 1 3 5 2 dibalik menjadi 2 5 3 1 setelah Mutation dilaksanakan.

## Elitism

Setelah dibentuk generasi terbaru, maka terdapat pilihan untuk melakukan Elitism, dimana Elitism mengambil beberapa dari kromosom yang memiliki fitness tertinggi dari generasi sebelumnya, dan meletakkan mereka ke dalam generasi terbaru, dimana Elitism dapat berfungsi sebagai jaring keamanan yang menjamin solusi terbaik tidak akan hilang dan bisa dipakai bila algoritma tidak menemukan solusi yang baik.

## Kondisi berhenti

Dan terakhir yang perlu ditentukan untuk algoritma genetik adalah kondisi berhenti dari algoritma genetik yang dibuat, atau algoritma akan berjalan terus untuk mencari jawaban terbaik tanpa berhenti. Terdapat beberapa solusi untuk kasus ini, seperti berhenti setelah n generasi dimana fitness terbaik dari tiap generasi tidak berubah banyak, atau mungkin algoritma akan berhenti setelah n generasi sehingga waktu berjalan algoritma akan kurang lebih sama, dimana kondisi berbeda dapat diuntungkan dalam kondisi – kondisi tertentu.

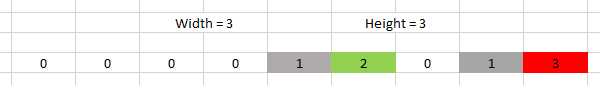
# Implementasi Algoritma Genetik untuk Splatted

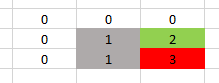
Dengan Metodologi Algoritma Genetik dijelaskan, sekarang akan dijelaskan Implementasi dari algoritma genetik untuk game Splatted ini, seperti metode representasi, Fitness – Fitness yang digunakan, dan metode – metode yang digunakan untuk operasi – operasi dalam algoritma genetik.

## Representasi

Dalam game Splatted, dikarenakan terdapat 2 jenis generasi level, maka terdapat 2 representasi yang berbeda untuk algoritma genetik kita.

Pertama adalah Tile Generation. Level kita akan berbentuk sebuah array 2D, tetapi untuk algoritma genetik disarankan untuk memakai array 1 dimensi, sehingga untuk representasi Tile Generation, digunakan array 1 dimensi sepanjang luas dari level yang akan dibuat, dimana setiap Gene akan merepresentasikan sebuah tile yang ada di dalam level dengan isi Gene 0 hingga 3, merepresentasikan objek yang diletakkan di tile tersebut, dan setelah game mendapat array hasil algoritma genetik, maka array 1 dimensi tersebut dapat diubah menjadi array 2D, yang bisa diubah menjadi level yang siap dimainkan.



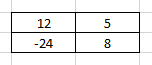
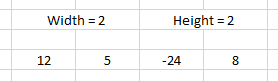


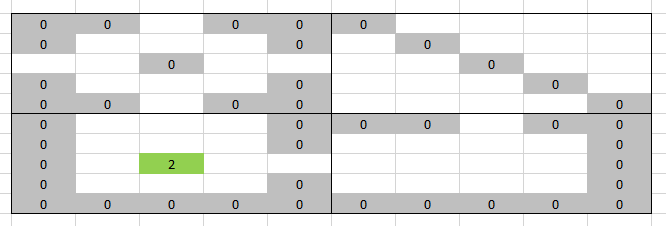
Gambar. 7. Contoh Representasi Tile Based menjadi sebuah level

Dengan contoh gambar 7 diatas, Representasi untuk Splatted ada di bagian atas gambar 7. Menggunakan variabel Width dan Height yang ditentukan oleh player, array 1 dimensi yang berisi angka angka berwarna (abu abu = batu, hijau = bola spesial dan 3 = player) diatas dapat dibentuk menjadi sebuah array 2D yang siap diolah menjadi level.

Lalu selanjutnya, mirip dengan Tile Generation diatas Template Generation direpresentasikan oleh sebuah kromosom 1 dimensi, tetapi tidak seperti Tile Generation Template Generation memiliki 2 perbedaan besar dalam representasi, yang pertama adalah setiap Gene di dalam Template Generation tidak diisi dengan angka 0 – 3 yang merepresentasikan obyek yang bisa diletakkan, tetapi -n hingga n dimana n adalah jumlah template yang siap dipakai. Setiap Gene akan diisi dengan id dari sebuah template, dimana bila id tersebut bernilai negatif maka template tersebut akan memiliki bola spesial di tengah template tersebut.

Perbedaan kedua yang berhubungan dengan perbedaan pertama diatas adalah panjang dari kromosom Template Generation tidak sepanjang luas dari level, tetapi luas dari level dibagi dengan 25. Ini dikarenakan setiap Template yang akan dipakai memiliki ukuran 5 x 5, membuat setiap Gene merepresentasikan sebuah tile berukuran 5 x 5 di dalam level. Untungnya level telah diatur sehingga pilihan – pilihan yang disediakan berkelipatan 5 untuk menghindari error – error yang mungkin terjadi.





Gambar. 8. Contoh Representasi Template Based menjadi sebuah level

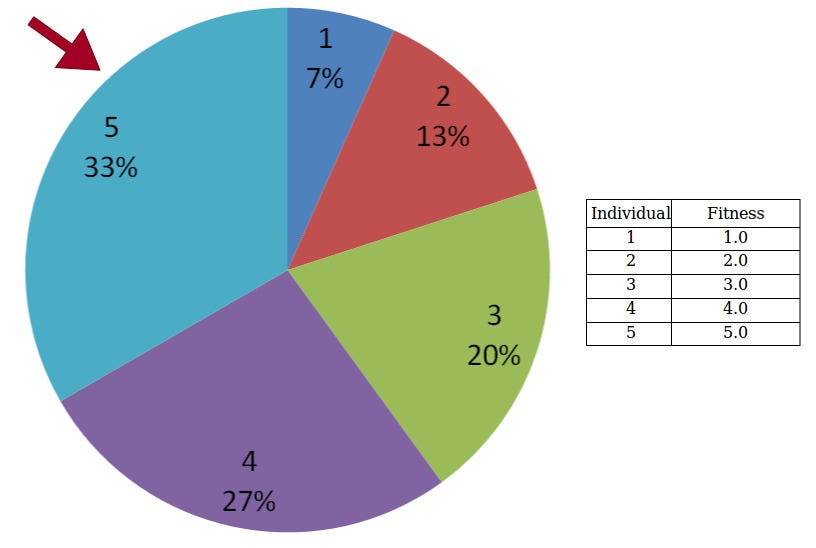
Dalam gambar 8, terdapat sebuah array 1 dimensi yang berisi 4 id dari template yang disiapkan. Dilakukan perubahan menjadi 2D, lalu setelah itu template akan dibuka, memperlihatkan sebuah pola yang diibuat oleh tangan.Di bawah kiri ada bola spesial dikarenakan bila template bersifat negatif maka bila di level tersebut akan diletakkan bola spesial di tengah – tengah template, sebagai pengganti bola spesial yang tidak mumcul dari Template Management.

## Fitness – Fitness

Dalam Splatted, terdapat 6 fitness yang akan bekerja sama untuk menentukan kualitas dari sebuah level. Seluruh fitness – fitness ini akan dijelaskan di bab selanjutnya, dimana kegunaan fitness beserta dengan rumus yang dipakai fitness tersebut akan dijelaskan.

## Metode Seleksi

Metode seleksi yang dipakai oleh Splatted adalah Roulette Wheel Selection, dimana setiap kromosom bisa dipilih sejelek apapun Fitness dari kromosom tersebut, tetapi kromosom yang memiliki fitness lebih tinggi akan memiliki kemungkinan lebih besar untuk dipilih dibandingkan dengan kromosom yang memiliki fitness rendah.



Gambar. 9. Visualisasi Roulette Wheel Selection

Dengan contoh gambar 7 diatas, terdapat 5 kromosom dengan fitness dicantumkan di kanan gambar, dengan individual 1 memiliki Fitness 1, individual 2 memiliki fitness 2 dst. Bisa dilihat bila kromosom 5 dengan fitness 5 memiliki kemungkinan dipilih sebanyak 33 %, sementara kromosom 1 dengan fitness 1 hanya memiliki 7% untuk dipilih. Metode ini dipilih dikarenakan metode ini menghadiahkan kromosom yang memiliki Fitness bagus tetapi masih menyimpan variasi potensial dari kromosom – kromosom dengan Fitness rendah.

## Metode Crossover

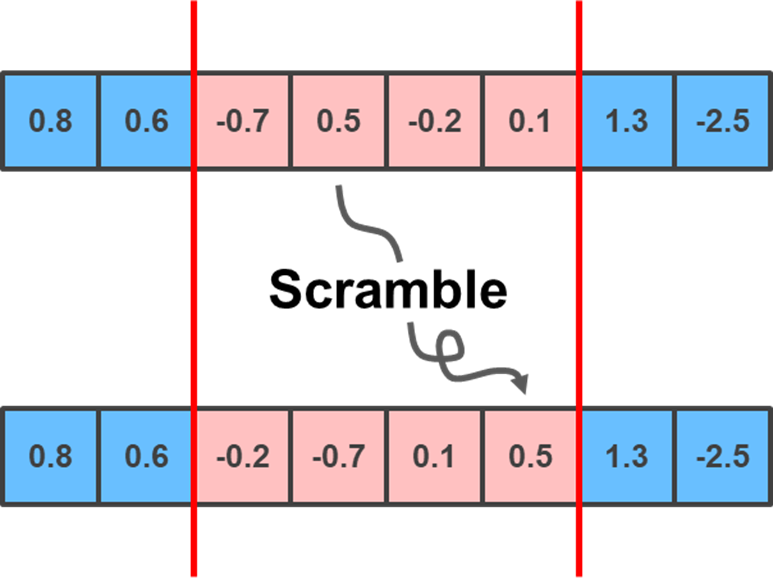
Untuk Splatted, algoritma Crossover yang dipakai adalah Uniform Crossover, dimana dimulai dari Gene pertama di paling kiri hingga Gene terakhir di paling kanan, setiap Gene memiliki kesempatan 50% untuk ditukar, membuat Uniform Crossover sebagai Crossover yang bisa mencampur aduk isi dari kromosom orang tua secara individual dimana kebanyakan Crossover lain memindahkan Gene secara kelompok, yang tidak cocok untuk game ini.



Gambar. 10. Visualisasi Uniform Crossover

## Metode Mutation

Algoritma mutasi yang digunakan oleh Splatted adalah Partial Shuffle Mutation atau juga bisa disebut Scramble Mutation, dimana diletakkan 2 pembatas diantara kedua Gene dalam kromosom kita, lalu seluruh Gene yang terdapat diantara kedua kromosom tersebut akan diacak posisinya. Mutation ini berguna untuk mengocok urutan dari bayak Gene secara sekaligus.



Gambar. 11. Visualisasi Scramble Mutation

## Kondisi Berhenti

Kondisi Berhenti dari Splatted adalah bila algoritma genetik tidak menemukan perkembangan yang signifikan setelah 100 generasi, maka algoritma akan dihentikan. Ini berguna agar algoritma berusaha mencari solusi terbaik yang bisa didapat, tetapi akan berhenti bila tidak membuahkan hasil untuk sementara. Ini bekerja di game Splatted dikarenakan kita perlu mencari level dengan kualitas bagus, tetapi tidak bisa mengorbankan terlalu banyak waktu dikarenakan ada player yang menunggu algoritma genetik tersebut.

# Fitness – Fitness Splatted

Seperti yang telah ditetapkan di Bab IVB, fitness – fitness akan dipisah ke bab baru dikarenakan jumlah isinya yang banyak dibandingkan dengan aspek – aspek lain dari sebuah algoritma genetik. Berikut adalah fitness – fitness yang akan dijelaskan :

## Fitness Jumlah Batu

Fitness Jumlah Batu digunakan untuk mengatur jumlah batu yang tersedia di dalam sebuah level agar level tidak koson atau memiliki terlalu banyak batu. Di sini terdapat beberapa variabel yang diperlukan oleh Fitness, yaitu MinR, yaitu jumlah minimal batu yang diperbolehkan di dalam level dan MaxR yang berisi jumlah maksimal batu yang diperbolehkan di dalam level.

(1)

Di rumus 1 diatas dicari m, sebuah nilai minus yang nanti akan digunakan untuk mengurangi nilai fitness yang berawal dari 1, dimana semakin jauh jumlah batu dari batas yang ditentukan maka semakin besar nilai minus. Sesuai dengan rumus 1, terdapat 3 posibilitas isi dari m. Pertama adalah batu minimal dikurang dengan batu di level, kedua adalah batu di level dikurangi batu maksimal bila batu di level melebihi batas batu maksimal yang ditentukan, dan terakhir 0 bila jumlah batu berada diantara kedua batas yang ditentukan.

(2)

Di rumus 2 diatas dicari nilai minus maksimal, yang nanti akan digunakan untuk menormalisasi nilai minus agar bila nilai minus digunakan untuk mengurangi Fitness, nilai Fitness tidak akan berada dibawah 0. Terdapat 2 posibilitas dari nilai minus maksimal, diantara luas level dikurangi jumlah batu maksimal atau jumlah batu minimal, entah yang mana yang lebih besar.

(3)

Tahap terakhir sesuai dengan rumus 3 adalah untuk mengurangi 1 dengan nilai minus yang dibagi nilai minus maksimal (untuk normalisasi). Dengan itu fitness sudah bisa didapatkan. Tetapi untuk membuat algoritma genetik lebih sensitif terhadap kesalahan maka nilai fitness akan dikuadratkan. Lalu W yang merupakan Weight akan dikalikan ke dalam nilai fitness agar bisa mengatur besar pengaruh fitness ini terhadap fitness lain. Rumus ini hanya digunakan pada Tile Generation dikarenakan bila ini diaplikasikan pada Template Generation maka akan sangat membatasi variasi dalam Template Generation.

## Fitness Ukuran Kelompok Batu

Selanjutnya adalah Fitness ukuran kelompok batu, dimana Fitness sebelumnya menghitung jumlah batu yang ada di dalam level, Fitness ini menghitung ukuran dari kelompok – kelompok batu yang ada di dalam level, sehingga batu – batu yang berkumpul bersama di dalam level tidak akan membentuk kelompok batu yang terlalu besar maupun kecil.

(4)

(5)

(6)

Fitness ini memiliki variabel – variabel yang sama dengan Fitness jumlah batu, seperti MaxR dan MinT, dan iikarenakan dengan konsepnya yang mirip dengan fitness jumlah batu, maka rumus yang dipakai untuk kedua fitness memiliki banyak kemiripan, kecuali pada rumus 6, dimana rumus 6 menghitung rata – rata dari seluruh fitness kelompok batu dibandingkan dengan rumus 3 yang hanya menghitung jumlah batu.

Rumus 4 dan 5 akan diulangi untuk setiap kelompok batu yang ditemui, dimana ditentukan nilai minus dan nilai minus maksimal dari kelompok batu tersebut. Setelah itu, dilakukan sebuah fungsi yang seperti rumus 3, lalu simpan nilai fitness itu untuk sementara. Setelah seluruh kelompok batu dihitung, maka dijalankan rumus 6, dimana seluruh fitness yang didapatkan dijumlah dan di rata – rata. Setelah selesai maka nilai Fitness dikuadratkan dan dikali W dan nilai Fitness siap dipakai. Fitness ini juga hanya dipakai pada Tile Generation karena Template Generation memiliki kelompok – kelompok yang cukup besar tapi hampir terjamin aman dan tidak akan mengganggu permainan.

## Fitness Luas Area yang bisa dilewati

Fitness ini mengecek seluruh tanah kosong yang ada di dalam level, lalu mengecek berapa segmen – segmen di dalam level yang tidak bisa mengakses satu sama lain dan mengurangi Fitness sesuai dengan segmen – segman yang tidak bisa diakses. Fitness ini digunakan untuk mencegah dibuatnya area – area yang tidak bisa diakses oleh siapapun, dikarenakan bila dalam sebuah level hanya sedikit area yang bisa diakses oleh player, maka level akan sulit digunakan untuk bermain perang salju, dimana terdapat banyak bola yang dilempar untuk memenangkan permainan.

(7)

Di dalam Fitness ini hanya terdapat 1 rumus, yaitu area terbesar dibagi dengan area total sesuai dengan rumus 7. Game akan menyimpan ukuran – ukuran dari segmen – segmen dalam level, mengambil segmen terbesar dalam map dan membaginya dengan total ukuran dari semua segmen di dalam map. Dengan ini apabila sebuah level tidak memiliki sebuah segmen yang tidak dapat diakses, maka level tersebut akan mendapatkan nilai Fitness 1 dari Fitness luas area ini.

## Fitness Bola Spesial yang bisa diraih

Fitness ini mengecek setiap bola spesial yang ada di dalam level, dan mengecek bola – bola spesial tersebut terhadap player terdekat. Apabila player terdekat tersebut bisa mengakses bola spesial, maka Fitness akan bertambah. Tetapi bila Player tidak bisa mengakses bola spesial yang ada di dekatnya, maka Fitness ini akan berkurang. Fitness ini berguna untuk mencegah adanya bola spesial yang tidak bisa diraih oleh player, dimana bila player terdekat tidak bisa mengakses bola spesial, maka player lain akan dilewati untuk menghemat waktu dalam algoritma genetik.

(8)

Dalam fitness ini, diambil seluruh posisi dari bola spesial dan player yang ada di dalam level, lalu setiap bola spesial akan melihat player yang terdekat diantara seluruh player yang dicatat. Setelah menemukan player terdekat, dilakukan algoritma A\* dari player menuju bola. Bila A\* berhasil membuat sebuah jalan, maka pakses yang menandakan jumlah bola spesial yang bisa diakses akan bertambah sebanyak 1. Tahap ini akan diulangi terus hingga seluruh bola telah diperiksa, lalu pakses akan dibagi dengan ptotal yang merupakan jumlah bola spesial dalam sebuah level. Dengan itu nilai Fitness telah didapatkan.

## Fitness Rasio Bola Spesial

Fitness berikutnya adalah Fitness Rasio Bola Spesial, dimana fitness ini digunakan untuk mengatur jumlah bola spesial yang ada di dalam level, sehingga bola spesial pasti hadir dalam sebuah level dan tidak memenuhi level.

(9)

(10)

(11)

Fitness ini bekerja dengan menghitung jumlah bola spesial yang tersedia, dan memasukkannya ke dalam rumus 9 hingga 11, yang sama seperti rumus 1 hingga 3 dikarenakan konsep nilai minusnya yang sama, dengan satu – satunya perbedaan adalah rumus ini menghitung jumlah bola spesial dibandingkan rumus 1 hingga 3 yang menghitung jumlah batu.

## Fitness Variasi Template

Terakhir adalah Fitness variasi template. Fitness ini melihat seluruh template yang ada di dalam level, dan mengurangi fitness apabila terdapat template sama yang muncul beberapa kali. Fitness ini berfungsi untuk menambah variasi dalam Template Generation agar tidak terjadi sebuah level yang mengandung 1 / 2 template yang sama diulang berkali – kali.

(12)

Untuk Fitness ini disediakan sebuah variabel x yang akan membantu untuk rumus 13 dibawah, dimana x adalah nilai negatif yang diberikan Fitness apabila ada sebuah template yang terlalu sering tampil dalam level, dimana bila frekuensi tampil melebihi toleransi (Tl), maka akan diberi fitness nilai frekuensi – toleransi yang akan bernilai negatif.

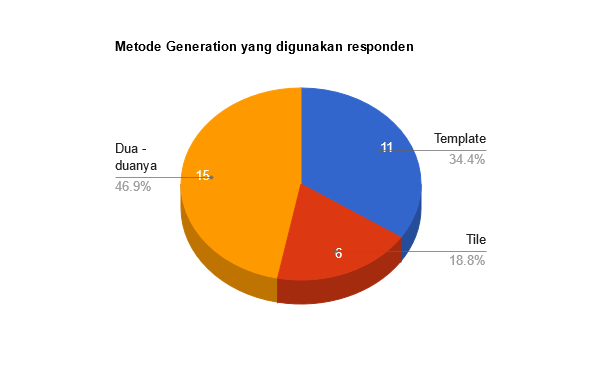
(13)

Setelah diperiksa seluruh template yang ada di dalam level, maka lakukan rata – rata dari seluruh fitness yang disimpan lalu dirata – rata dengan jumlah template yang unik di dalam level seperti rumus 6. Bila hasil dari rata – rata tersebut diatas 0 maka kuadratkan lalu kali W. Tetapi bila hasil rata – rata dibawah 0, maka nilai Fitness yang diberikan adalah 0, dikarenakan bila Fitness diberi nilai negatif bisa menyebabkan error yang tidak diinginkan.

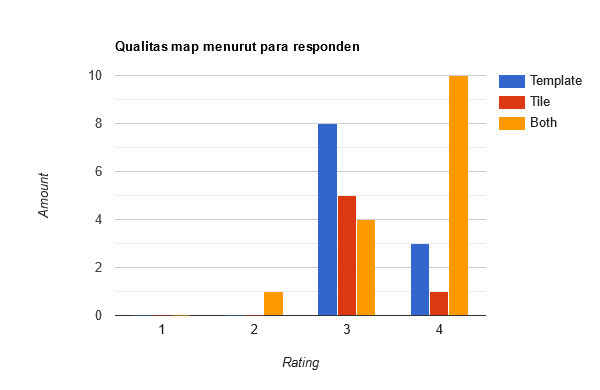
# Uji Coba

Untuk melihat kualitas dari level – level yang telah dibuat oleh game “Splatted”, akan diadakan 2 metode uji coba. Yang pertama adalah untuk membuat kuisioner yang berisi pendapat – pendapat dari para responden yang memainkan game ini, lalu kedua adalah untuk mengambil 2 level dari masing – masing metode Generation, dan mencoba untuk menganalisa tiap level yang ada tersebut.

## Tanggapan kuisioner



Gambar. 12. Metode Generation yang dipakai oleh para responden



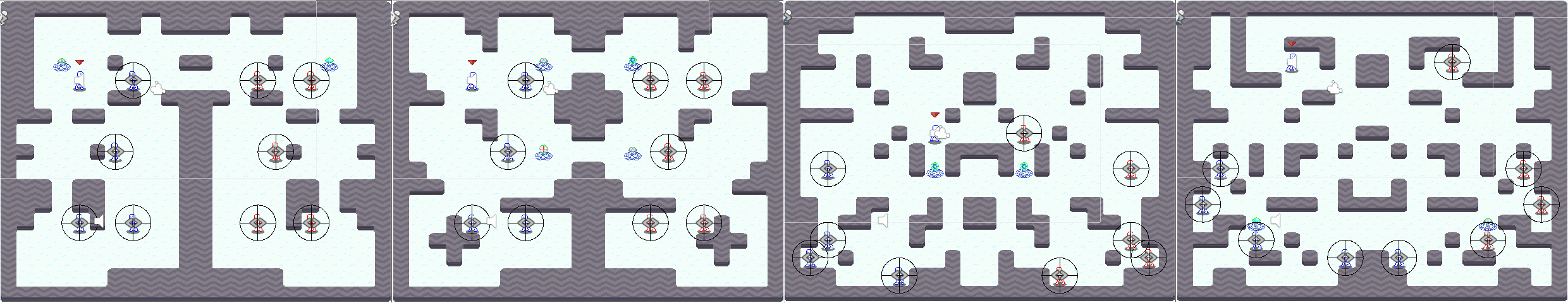
Gambar. 13. Skor yang diberi Responden

Dua gambar diatas merupakan responden – responden mahasiswa iSTTS yang telah memainkan game Splatted dan memberikan pendapat mereka mengenai kualitas dari level – level yang telah dibuat. Terdapat 3 kelompok responden yang direpresentasikan menggunakan warna, yaitu responden yang mencoba Template Generation direpresentasikan warna biru, responden yang mencoba Tile Generation menggunakan warna merah dan terakhir adalah responden yang mencoba kedua – dua Generation ditampilkan dengan warna kuning.

Total responden saat menulis tugas akhir / karya ilmiah ini adalah 32 responden, dengan 11 mencoba template generation, 8 mencoba tile generation dan 15 sisanya mencoba kedua – dua generation berdasarkan gambar 3. Lalu dari seluruh responden tersebut, nilai nilai yang diberi dapat dilihar di Gambar 4 dimana rata rata nilai berada diantara 3 dan 4, dengan 1 responden tidak puas dengan level yang dibuat dan memberi nilai 2.

Dari gambar 4 diatas, diambil konklusi apabila menurut para responden, level – level yang dibuat oleh algoritma genetik bersifat memuaskan secara keseluruhan meskipun mungkin masih ada beberapa aspek yang dapat dikembangkan dari generasi level game “Splatted’.

## Analisa Level



Gambar. 14. Level - level yang dibuat oleh algoritma genetik

Di gambar 5 diatas terdapat 4 level berukuran kecil (15 X 20 Tile) yang dibuat oleh algoritma genetik, dimana terdapat 2 level di kiri yang menggunakan Template Generation dan 2 lagi di kanan menggunakan Tile Generation. Dilihat dari sini, keempat level memiliki beberapa tempat terbuka yang bisa digunakan para player untuk bermain, dan juga tidak ada tempat yang tidak bisa diakses berkat fitness yang dibuat. Lalu selanjutnya, seluruh level memiliki bola spesial tersebar dalam setiap level dengan jumlah yang tidak terlalu banyak. Untuk konklusi dari pengamatan level yang dibuat, level – level bisa dilihat tidak menghalang player dalam bermain dan terlihat tidak jelek secara estetika.

# Kesimpulan

Algoritma Genetik dapat digunakan untuk membuat sebuah Procedural Map Generation yang bagus, mungkin setara atau lebih bagus dengan metode – metode Procedural Map Generation lainnya selama Fitnesss yang diterapkan pada algoritma genetik didesain dengan baik.

# Ucapan Terima Kasih

Terima kasih diucapkan kepada keluarga saya yang mendorong saya untuk berkembang terus, dosen – dosen iSTTS yang bersabar dalam mengajari saya, staff iSTTS yang selalu siap membantu, teman – teman seperjuangan saya yang juga membantu saya secara langsung maupun tidak langsung, dan terakhir pak Hendrawan karena telah membimbing saya dari awal hingga akhir pembuatan tugas akhir ini.

Daftar Pustaka

1. Barriga, Nicholas A. *Short Introduction to Procedural Content Generation Algorithms for Videogames.* Chile : University of Talca. 2018.
2. Linden, Roland van der et. All. *Procedural generation of dungeons.* IEEE. 2013
3. Hendrikx, Mark et. All. *Procedural Content Generation for Games: A Survey.* Multimedia Comput. Commun. Appl. 2011
4. Togelius, Julian et. All. *Search-Based Procedural Content Generation:*

*A Taxonomy and Survey.* IEEE. 2011

1. L. Raffe, William et. All. *An Integrated Approach to Personalized Procedural Map Generation using Evolutionary Algorithms* IEEE. 2014
2. Togelius, Julian et. All. *Towards multiobjective procedural map generation.* PCGames '10: Proceedings of the 2010 Workshop on Procedural Content Generation in Games. 2010
3. Pereira, Leonardo et. All. *A multi-population genetic algorithm for procedural generation of levels for platform games*. Sao Carles : Conference Paper. 2014.
4. Whitley, Darrell. *A Genetic Algorithm Tutorial*. Technical Report CS-93-103. 1993.
5. Katoch, Sourabh. *A review on genetic algorithm: past, present, and future*. Multimed Tools Appl. 2020.
6. Haldurai, L. *A Study on Genetic Algorithm and its Applications*. Kongunadu Arts and Science College. 2016.
7. Schmitt, Lothar M.. *Fundamental Study Theory of genetic algorithms*. The University of Aizu. 2000.
8. Khalid Jebari, Mohammed Madiafi.. *Selection Methods for Genetic Algorithms*. Mohammed V–Agdal University. 2013.
9. Shukla, Anupriya et. All. *Comparative Review of Selection Techniques in Genetic Algorithm. Amity University*.
10. Goldberg, David E. and Deb, Kalyanmoy. *A Comparative Analysis of Selection Schemes Used in Genetic Algorithms*. University of Illinois at Urbana-Champaign. 1991.
11. Razali, Noraini Mohd and Geraghty, John. *Genetic Algorithm Performance with Different Selection Strategies in Solving TSP*. The Proceedings of the World Congress on Engineering. 2011.
12. Umbarkar, A.J. and Sheth, P.D. *Crossover Operators In Genetic Algorithm : A Review*. Walchand College of Engineering.
13. Kora, Padmavathi and Yadlapalli, Priyanka. *Crossover Operators in Genetic Algorithms: A Review*. Institute of Engineering & Technology, Hyderabad. 2017.
14. De Jong, Kenneth A and Spears, William M. *An Analysis of the Interacting Roles of Population Size and Crossover in Genetic Algorithms*. International Conference on Parallel Problem Solving from Nature, 1990.
15. Syswerda, Gilbert. *Uniform Crossover in Genetic Algorithm*. BBN Laboratories. 1989.
16. Deb, Kalyanmoy and Deb, Debayan. *Analysing mutation schemes for real-parameter genetic algorithms.* International Journal of Artificial Intelligence and Soft Computing. 2014.
17. Katayama, K et. All. *The Efficiency of Hybrid Mutation Genetic Algorithm for the Travelling Salesman Problem*. Mathematical And

Computer Modelling. 2000.

**Lukky Hariyanto** lahir di Surabaya, Jawa Timur pada tahun 2001. Dia memulai studi S1 di program studi Informatika STTS pada tahun 2019. Minatnya pembelajarannya adalah bidang game development dan Computer Science.

1. Lukky Hariyanto, Departemen Informatika, Institut Sains dan Teknologi Terpadu Surabaya, Surabaya, Jawa Timur, Indonesia (e-mail: lukky.h20@mhs.istts.ac.id)

   Hendrawan Armanto, Departemen Informatika, Institut Sains dan Teknologi Terpadu Surabaya, Surabaya, Jawa Timur, Indonesia (e-mail: hendrawan@stts.edu) [↑](#footnote-ref-1)