Penggunaan Audio Spectrum dan Genetic Algorithm Untuk Obstacle Generation Game Platforming Berbasis Musik

Calvin Octaviano, Kristo Radion Purba, Henry Palit
Program Studi Teknik Informatika Fakultas Teknologi Industri Universitas Kristen Petra
Jl. Siwalankerto 121 – 131 Surabaya 60236
Telp. (031) – 2983455, Fax. (031) – 841765
E-mail: calocta77@gmail.com, kristo@petra.ac.id, hnpalit@petra.ac.id

ABSTRAK

Perkembangan jaman mempermudah akses ke hobi seperti mendengarkan musik dan bermain *game*, namun jarang ditemui kedua hobi tersebut dapat dilakukan secara bersamaan, sehingga dikembangkanlah sebuah program yang membantu menggabungkan kedua hobi tersebut menjadi satu dengan bantuan *game engine* Unity dan menggunakan file musik dengan ekstensi *.ogg* untuk menghasilkan sebuah level untuk game dengan genre *platforming*.Metode yang digunakan adalah *genetic algorithm* untuk membantu penyusunan dalam *level* sehingga diharapkan tingkat kesulitan yang didapatkan cocok dengan pilihan pengguna, dengan memperkecil kemungkinan terjadinya susunan *obstacle* yang sama untuk mengurangi kebosanan terhadap sebuah permainan / *level*.

Kata Kunci: *OGG, GA, genetic algorithm, unity, platforming, game, level generation*

ABSTRACT

Advancement in technology helps the development of hobbies such as listening to music and playing games, but in reality, both hobbies done simultaneously is rarely met, because of that, a program was developed to help in combining both hobbies into one activity with the help of Unity game engine while using widely known OGG audio file format to generate a level for a platforming game.Genetic algorithm was used as a method to deliver a process of developing level with hopes that the level structure matches the difficulty chosen by user, while also reducing the probability of a same level appearing twice, increasing game repetitiveness and minimalizing bore level.

Keywords: *OGG, GA, genetic algorithm, unity, platforming, game, level generation*

1. PENDAHULUAN

Dengan berkembangnya jaman dan teknologi, kehidupan manusia semakin sulit dipisahkan dengan teknologi.Berbagai macam teknologi mulai mempengaruhi kegiatan manusia sehari-hari seperti bekerja, belajar, hingga menyalurkan hobi.Dua dari banyak hobi yang dipengaruhi perkembangan teknologi adalah mendengarkan musik dan bermain game.

Berbagai macam teknologi mulai musik yang sebelumnya menggunakan CD dan kaset sekarang semakin praktis dengan digitalisasi. Sama seperti musik, game mengalami digitalisasi sehingga berbagai macam game sudah ada di alat-alat elektronik, seperti PC. Seringkali kedua hobi tersebut tidak bisa dilakukan bersamaan, karena game pada umumnya memiliki soundtrack yang diputar saat dimainkan, dan lagu didalam game meningkatkan imersi dari game tersebut, sehingga akan mengurangi imersi jika dimatikan [7].

Terdapat dua game sejenis yang didasari oleh lagu untuk gameplay nya, yaitu Beat Hazard yang bergenre Shoot 'Em Up, dan Audiosurf yang bergenre Racing. Kedua game tersebut menggunakan lagu untuk level generation, dimana perbedaannya adalah Beat Hazard menggunakan spectrum, namun tidak menghasilkan level yang benar-benar random dengan lagu yang sama, dan Audiosurf menggunakan beat detection.

Masalah yang dihadapi adalah kurangnya jumlah game yang bisa dimainkan bersamaan dengan adanya lagu yang diputar sambil tetap menjaga imersi dari game tersebut.Memberi pilihan dari lagu yang di build ke dalam game juga bukan solusi yang terbaik, karena setiap orang memiliki selera musik yang berbeda.Masalah lain yang berhubungan dengan lagu adalah kebosanan terhadap sebuah lagu jika terdengar terlalu sering, akan menimbulkan rasa bosan karena tidak ada perbedaan dalam setiap putaran.

Dari masalah tersebut, dibutuhkan sebuah game yang dapat menggunakan lagu yang sesuai dengan selera player, sambil tetap menjaga imersi dari game tersebut, dan memiliki replay value [7] yang tinggi agar tidak menyebabkan kebosanan terhadap sebuah lagu dan level.

2. TINJAUAN PUSTAKA

2.1 Audio Spectrum

Audio spectrum adalah representasi dari suara yang umumnya berupa sample, yang menunjukkan getaran pada setiap frekuensi.Spectrum biasanya ditampilkan dalam bentuk graph kekuatan atau tekanan yang mewakili frekuensi.Kekuatan atau tekanan biasanya diukur dalam desibel(dB), sedangkan unit standar internasional untuk frekuensi diukur dalam hertz [8]. Setelah melakukan identifikasi pada delapan file mp3 dari lagu dengan genre dan tempo berbeda menggunakan analisa spektrum dalam aplikasi Audacity, ditemukan bahwa rentang frekuensi dari kedelapan sampel tersebut berada pada 11 Hz hingga 15662 Hz.Judul-judul dari kedelapan lagu beserta rentang frekuensinya adalah sebagai berikut :

- Ink Spots – If I Didn't Care (11 Hz hingga 10947 Hz)
- Charlene – I've Never Been To Me (11 Hz hingga 15643 Hz)

- Julie Andrews – The Hills Are Alive (43 Hz hingga 15275 Hz)
- Feint – Phospor (54 Hz hingga 15659 Hz)
- Puppet – Scribble (56 Hz hingga 15660 Hz)
- Heroes of the Storm OST – Dragonshire (11 Hz hingga 15609 Hz)
- Two Steps From Hell – Skyworld (11 Hz hingga 13022 Hz)
- Maria Callas – O Mio Babbino Cano (62 Hz hingga 15662 Hz)

Dari hasil identifikasi diatas, ditentukan bahwa frekuensi akan dibagi menjadi 4 band, sesuai jumlah jenis obstacle yang disediakan, dimana pembagiannya adalah :

- frekuensi <4000 Hz menghasilkan obstacle slide
- frekuensi 4000 – 7999 Hz menghasilkan obstacle lompat
- frekuensi 8000 – 12000 Hz menghasilkan obstacle attack
- frekuensi >12000 Hz menghasilkan obstacle reverse(melawan arah lari)

2.2 Genetic Algorithm

Genetic algorithm adalah metode untuk memecahkan masalah yang didasarkan pada proses seleksi alam yang meniru evolusi biologis dimana algoritma ini memodifikasi population dari solusi.Setiap langkah genetic algorithm menentukan individu dari population secara acak dan menggunakannya sebagai parent untuk menghasilkan children untuk generasi berikutnya.Setiap perulangan, populasi berkembang menjadi solusi yang optimal [1].

Dalam pengerjaan ini, genetic algorithm digunakan untuk membuat susunan obstacle dalam sebuah level, dimana parameter yang digunakan adalah amplitudo dari frekuensi sesuai empat pembagian band yang dibuat, tingkat kesulitan individu dan kombinasi dari obstacle, dan tingkat kesulitan yang dipilih oleh player sebagai fitness function.

Keempat band frekuensi yang digunakan dalam genetic algorithm akan menentukan obstacle yang akan digunakan dalam level, sesuai waktu pada audio yang diputar. Jika lebih dari satu band frekuensi sedang diputar, maka semua band frekuensi akan dihitung perbandingan total amplitudonya, lalu akan diambil satu jenis obstacle secara acak dengan kemungkinan yang proporsional dengan perbandingannya.

3. ANALISA DAN DESAIN SISTEM

3.1 Desain Sistem

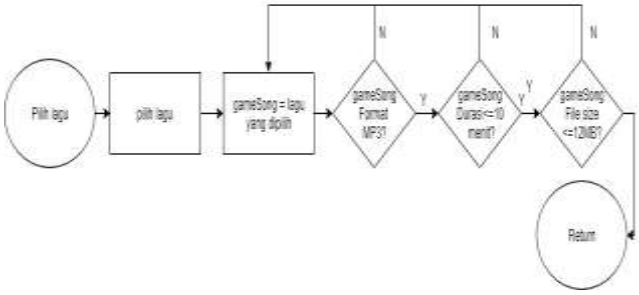
Desain sistem dilakukan dengan tujuan untuk menentukan arah pembuatan dan mengurangi kemungkinan terjadinya masalah untuk aliran proses pada program, dimana desain sistem meliputi desain proses dan desain tampilan.

3.1.1. Desain Flowchart Program

Desain flowchart digunakan untuk menunjukkan garis besar arah pembuatan program, sehingga langkah-langkah pengerjaan menjadi jelas sebelum memulai pengerjaan program.

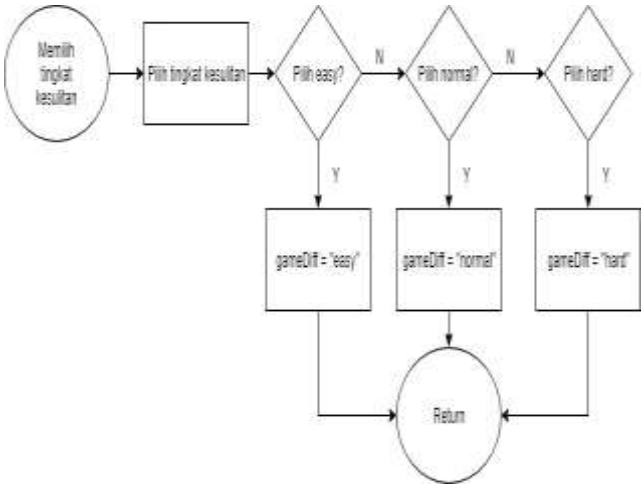
Gambar 1. Flowchart utama program

Gambar 1 menunjukkan proses utama ketika user memilih untuk memulai permainan pada menu utama dalam game.Langkah pertama adalah memilih lagu yang akan digunakan sebagai BGM dan parameter untuk generate susunan obstacle yang akan dimainkan. Langkah berikutnya adalah memilih tingkat kesulitan yang akan menentukan tingkat kesulitan dari susunan obstacle dengan menggunakannya sebagai parameter GA. Proses berikutnya adalah men-generate level yang akan dimainkan user, dan user bisa mulai memainkan game.



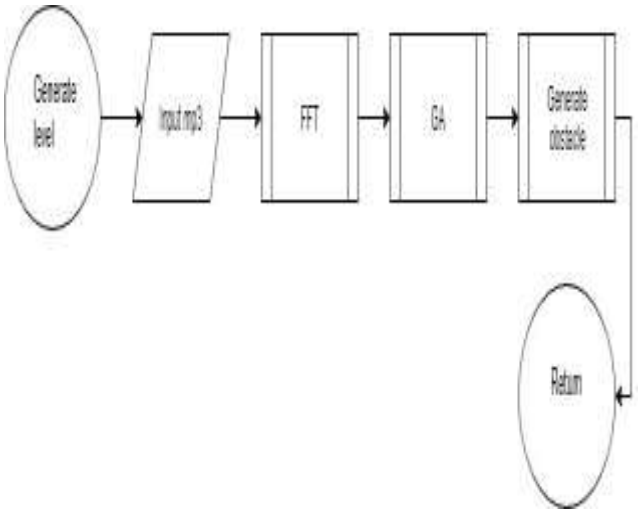
Gambar 2. Flowchart proses memilih lagu

Gambar 2 menunjukkan proses untuk memilih lagu yang akan digunakan sebagai BGM dan parameter untuk men-generate susunan obstacle. Pertama user akan diminta memilih sebuah file melalui sebuah explorer yang diintegrasikan ke dalam game.Langkah berikutnya adalah melakukan pengecekan pada file yang dipilih user.Jika file yang dipilih adalah file dalam format OGG, maka pengecekan berikutnya dilakukan untuk durasi dari lagu, jika durasinya tidak lebih besar dari 10 menit, maka pengecekan berikutnya dilakukan untuk ukuran dari file, jika ukuran file tidak lebih besar dari 12 MB, maka file lagu memenuhi syarat dan dapat digunakan untuk melakukan penyusunan obstacle dalam level.



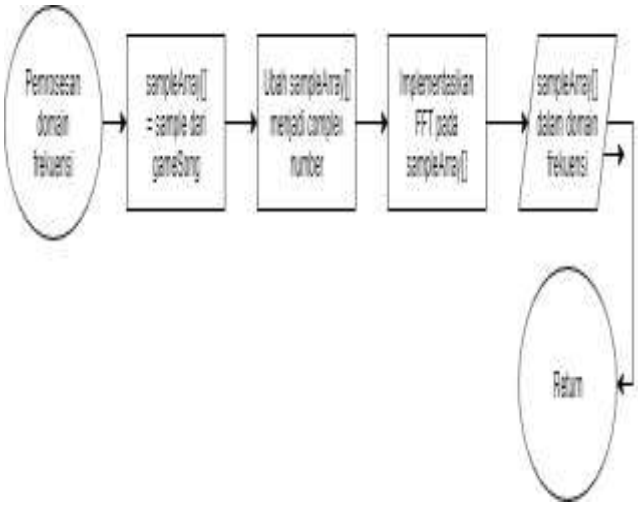
Gambar 3. Flowchart proses memilih tingkat kesulitan

Gambar 3 menunjukkan proses pemilihan tingkat kesulitan yang dilakukan oleh user. Pertama user akan diminta menentukan tingkat kesulitan yang diinginkan pada sebuah menu.Berikutnya game akan melakukan pengecekan untuk tingkat kesulitan yang dipilih oleh player.Jika easy maka game akan melakukan penyusunan obstacle dalam level sehingga level akan menjadi relatif mudah dimainkan dibandingkan dengan normal dan hard. Tingkat kesulitan normal dan hard juga berlaku sama dengan easy, sesuai nama yang diwakilinya masing-masing.



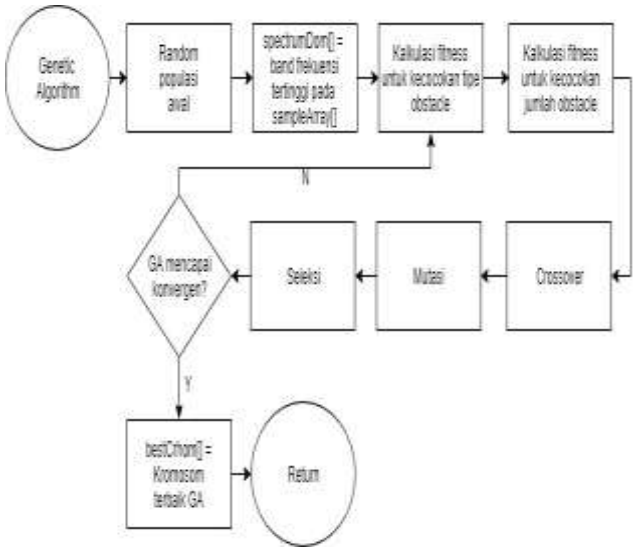
Gambar 4. Flowchart proses generasi level

Gambar 4 menunjukkan proses yang dilewati untuk menghasilkan susunan level.Input yang dibutuhkan adalah file OGG yang dipilih user pada bagian menu dan telah melalui proses pengecekan format file, durasi dari lagu, dan ukuran dari file yang dipilih, lalu proses yang dilakukan berikutnya adalah melakukan Fast Fourier Transform(FFT) yang menghasilkan data spectrum sebagai parameter Genetic Algorithm(GA). Proses berikutnya dalah mengeksekusi GA dengan parameter-parameter yang ada, lalu berikutnya dilakukan generasi susunan obstacle ke dalam level sesuai dengan output dari GA yang dilakukan.



Gambar 5. Flowchart pemrosesan domain frekuensi

Gambar 5 menunjukkan langkah-langkah pemrosesan domain frekuensi, dimulai dengan mengambil sample audio dari lagu yang dipilih oleh player, dengan total sample keseluruhan adalah (sample rate*durasi lagu). Langkah berikutnya adalah mengkonversi sample yang didapat menjadi format complex number karena proses FFT yang diimplementasi membutuhkan input berupa complex number. Proses berikutnya adalah melakukan implementasi FFT sesuai dengan implementasi yang ditulis pada bab 2, dimana output dari FFT ini adalah spectrum yang akan digunakan pada proses GA sebagai salah satu parameter dalam perhitungannya.



Gambar 6. Flowchart proses genetic algorithm

Flowchart pada gambar 6 menunjukkan proses yang dilewati dalam eksekusi Genetic Algorithm(GA). Yang dilakukan pertama adalah random populasi awal sebagai parent pertama.Populasi dalam GA yang digunakan dalam game ini adalah berupa array yang masing-masing index mewakili posisi obstacle terhadap waktu(X ms, X*2 ms, X*3 ms, dst, dimana X adalah tingkat akurasi dari posisi obstacle dalam milisecond) dimana isi dari array mewakili salah satu dari keempat bentuk obstacle yang akan digenerate ke dalam level, atau dapat mewakili "kosong" dimana pada timestamp tersebut tidak ada obstacle yang akan diletakkan.

Proses berikutnya adalah menghitung fitness dari masing-masing populasi dengan mempertimbangkan kecocokan obstacle yang dihasilkan, dan tingkat kesulitan yang dipilih player.kecocokan obstacle pada timestamp tertentu dihitung dengan membandingkan bentuk obstacle yang ada dengan band frekuensi yang dominan pada saat tersebut, jika cocok maka fitnessnya akan semakin besar, selain itu fitnessnya akan semakin kecil.Kecocokan terhadap tingkat kesulitan dihitung dengan membandingkan jumlah obstacle per 4 detik, dimana jumlah obstacle yang diharapkan adalah 3, 5, 7 pada tingkat kesulitan easy, normal, dan hard sesuai urutan.Semakin mendekati jumlah obstacle yang diharapkan maka fitnessnya akan semakin besar.

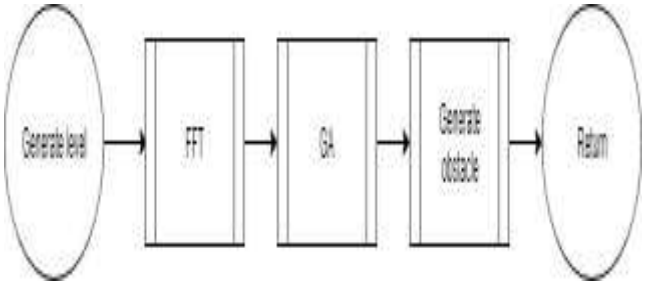
Langkah berikutnya adalah melakukan two-point crossover pada populasi yang ada dengan menggunakan konsep yang mirip dengan multi-point crossover namun sisi yang ditukar antara masing-masing populasi ditentukan titik potongnya dengan melakukan random angka.

Proses selanjutnya adalah melakukan mutasi pada populasi yang ada. Mutasi yang diterapkan adalah konsep random resetting dengan menentukan gen yang akan direset secara random.Jumlah gen yang akan direset akan mengalami *diminishing return* untuk mengurangi kemungkinan terjadinya mutasi yang terlalu besar, namun juga agar tidak terlalu sedikit.

Proses seleksi pada populasi yang ada akan dilakukan berdasarkan konsep fitness-based selection karena prioritas dari susunan obstacle yang baik lebih besar daripada variasi susunan obstacle.Seleksi yang dilakukan akan menghilangkan kromosom dengan fitness paling rendah sehingga diharapkan kromosom baru yang digenerate akan lebih baik.

Proses berikutnya adalah proses pengecekan untuk mengetahui jika fitness tertinggi dalam GA mencapai peresntase fitness yang ditentukan. Jika fitness telah mencapai persentase fitness yang ditentukan, maka GA akan menganggapnya konvergen dan mengembalikan kromosom dengan nilai fitness tertinggi. Jika fitness belum mencapai konvergen, maka GA akan terus berjalan hingga mencapai konvergen.

Hasil dari GA adalah kromosom dengan fitness terbaik dari seluruh proses yang akan di output kan menuju proses men-generate susunan obstacle sesuai dengan gen-gen dalam kromosom yang dipilih.



Gambar 7. Flowchart proses plotting obstacle

Gambar 7 menunjukkan proses-proses yang akan dilakukan setelah generasi GA selesai.Pertama dibutuhkan sebuah variabel untuk menghitung gen keberapa yang harus diletakkan di dalam level, dimana variabel ini akan diberi nama X dan diatur memiliki nilai awal=0.

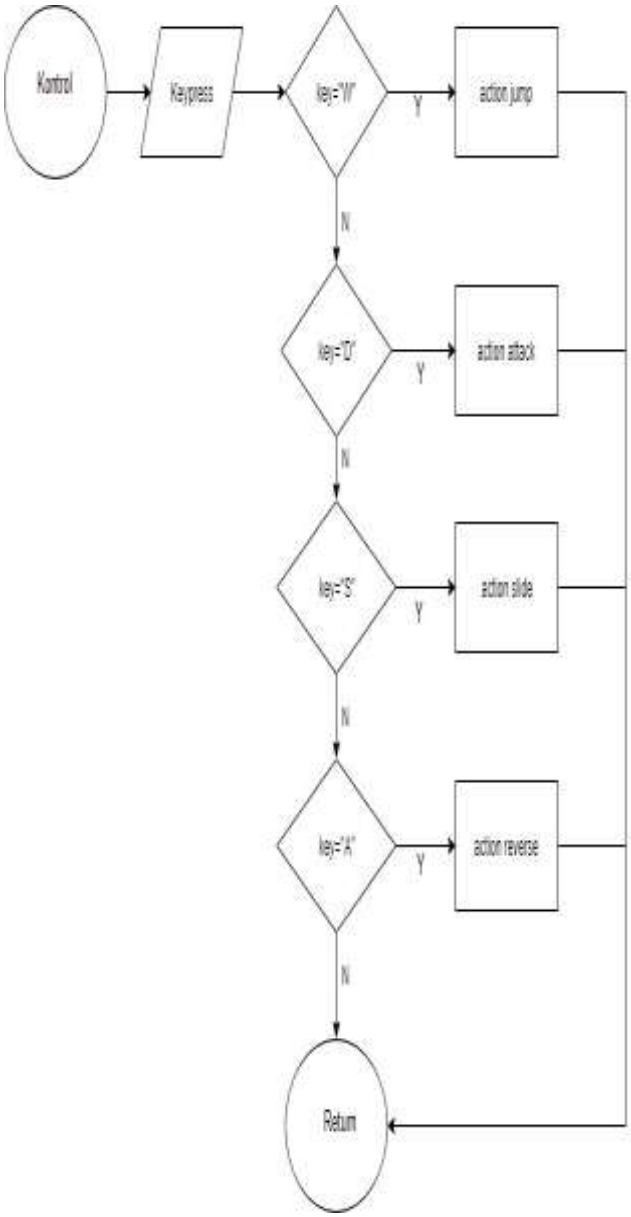
Berikutnya adalah mendapatkan input berupa kromosom terbaik dari proses GA yang telah dikerjakan sebelumnya, dimana kromosom terbaik didapatkan dengan melihat kromosom mana yang memiliki fitness paling besar dalam sebuah populasi.

Langkah berikutnya adalah membuat sebuah variabel dimana isinya adalah jumlah gen dalam kromosom GA yang didapatkan dalam proses sebelumnya.Variabel ini digunakan untuk membatasi jumlah perulangan dalam melakukan plotting individual gen, dan diberi nama Y dengan nilai awal adalah jumlah gen dalam kromosom GA.

Proses selanjutnya adalah melakukan plotting obstacle ke dalam level sesuai dengan gen dalam kromosom.Gen dalam kromosom memiliki nilai yang berisi jenis obstacle yang akan diletakkan, jika kosong maka tidak ada obstacle yang akan diletakkan di dalam level.Untuk setiap satu gen dalam kromosom, posisi obstacle akan bertambah 500ms terhadap waktu, menyebabkan posisi obstacle berada di 500ms*X sehingga mengeliminasi kemungkinan terjadinya overlap antar obstacle.

Langkah berikutnya yaitu melakukan increment pada X agar tidak terjadi plotting obstacle yang sama dengan gen sebelumnya.X akan mengalami increment setiap kali plotting telah selesai dilakukan.

Berikutnya adalah pengecekan untuk membatasi plotting yang dilakukan agar sesuai dengan jumlah gen dalam kromosom.Jika X masih lebih kecil daripada Y-1 maka belum semua gen mengalami plotting, sehingga proses diulang dari plotting hingga increment terhadap X.Jika X lebih besar daripada Y-1 maka semua gen sudah di plot dan proses plotting selesai.



Gambar 8. Flowchart proses kontrol player

Gambar 8 menunjukkan proses pengecekan input yang dilakukan user saat bermain, meliputi berbagai kontrol untuk navigasi karakter. Action yang digunakan ada 4 sesuai dengan jenis obstacle yang dihadapi, untuk obstacle jump maka membutuhkan action jump, dan seterusnya.

Pertama adalah menunggu input dari user, jika user telah menekan tombol di keyboard, maka input akan dicatat dan akan digunakan untuk pengecekan-pengecekan berikutnya.

Pengecekan pertama adalah jika user menekan tombol ‘W’ di keyboard. Yang terjadi adalah karakter akan melakukan action jump yang akan menghindarkan karakter dari obstacle yang menghalangi ketika karakter berada di tanah level.

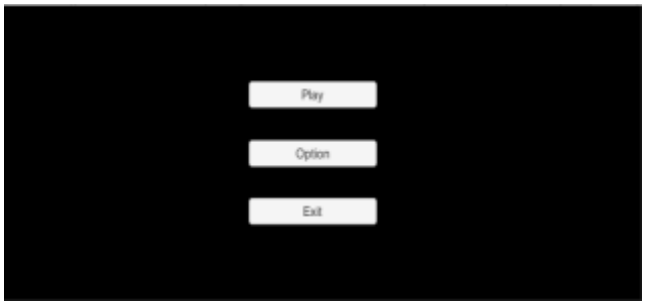
Pengecekan kedua adalah ketika user menekan tombol ‘D’ di keyboard.Action yang terjadi adalah attack dimana karakter akan menghancurkan obstacle dengan tipe khusus yang bisa di attack ketika menyentuhnya, menghilangkan obstacle sehingga karakter bisa melanjutkan level.

Pengecekan ketiga adalah saat tombol yang ditekan user adalah ‘S’. Action tersebut akan menyebabkan karakter yang dimainkan melakukan slide, sehingga tinggi dari karakter berkurang dan karakter bisa melewati obstacle yang memiliki celah yang cukup dibawahnya. Action ini cocok untuk obstacle berjenis slide dimana karakter akan melewatinya dan bergerak dengan kecepatan stabil.

Pengecekan terakhir adalah untuk tombol keyboard ‘A’ dimana action yang dilakukan oleh karakter adalah reverse. Reverse menyebabkan karakter berlari mundur menjauhi akhir level, untuk menghindari obstacle berjenis reverse dimana karakter tidak bisa maju meskipun melakukan action jump, slide, maupun attack. Obstacle reverse adalah berupa tanjakan dengan sebuah tembok melayang berbentuk J di depannya, sehingga karakter akan menabrak tembok jika tidak mundur untuk turun terlebih dahulu.

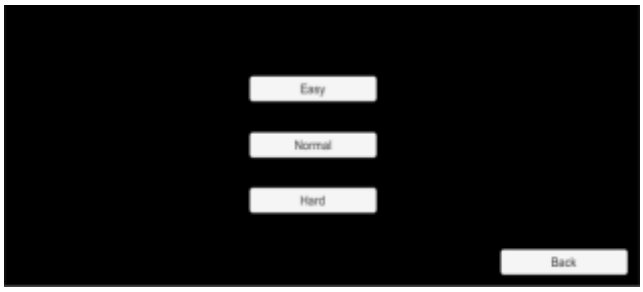
3.2 Desain tampilan

Desain tampilan akan menjadi acuan mengenai tampilan kasar yang akan dihadapi oleh pengguna, dengan memperhatikan komponen-komponen yang perlu dibuat di dalam program seperti tombol, *slider*, dan teks.



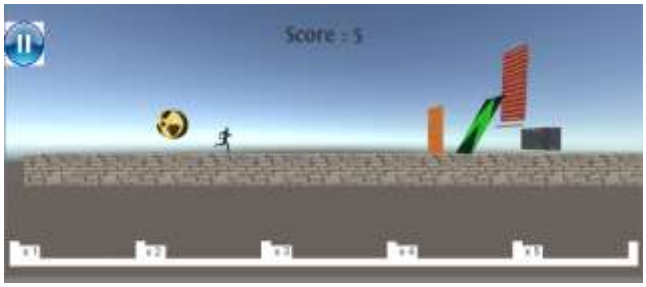
Gambar 9. Tampilan main menu

Gambar 9 menunjukkan tampilan pada menu awal ketika game pertama diluncurkan.Terdapat judul dari game di bagian atas UI berupa teks.Terdapat tiga tombol pada tampilan ini, tombol pertama bertulisakan play game yang jika diklik akan menampilkan browser dari windows explorer, ketika user telah selesai memilih lagu, maka akan ditampilkan menu berikutnya yaitu menu untuk memilih tingkat kesulitan.



Gambar 10. Tampilan pemilihan difficulty

Gambar 10 menunjukkan tampilan ketika user memilih play game pada menu sebelumnya, menu ini menunjukkan pilihan difficulty yang bisa ditentukan user untuk permainan, jika memilih salah satu difficulty yang disediakan maka user akan dibawa ke dalam game untuk bermain, jika user memilih untuk Back maka user akan dibawa ke menu awal kembali.



Gambar 11. Tampilan dalam game

Gambar 11 menunjukkan tampilan interface ketika user sedang bermain, dimana Actor adalah karakter yang digerakkan oleh user, dengan sebuah persegi di sebelah kanan Actor mewakili salah satu obstacle yang akan digenerate ke level.

4. IMPLEMENTASI SISTEM

4.1 Spesifikasi Pengujian

Pengujian dilakukan dengan menggunakan *personal computer* dengan spesifikasi sebagai berikut :

Prosesor :	Intel(R) Core(TM) i5-4570 CPU @3.20Ghz(4 CPUs), ~3.2Ghz
RAM :	8192MB RAM
Kartu grafis :	AMD Radeon R7 200 Series 2048MB VRAM
OS :	Windows 10 Pro 64-bit
Resolusi :	1366x768
DirectX :	DirectX 12

Program dibuat dengan menggunakan Unity versi 5.61f.

4.2 Pengujian kecocokan dengan band frekuensi dominan

Pengujian kecocokan dengan band frekuensi dominan dilakukan untuk mengetahui tingkat kecocokan susunan *obstacle* yang dihasilkan *Genetic Algorithm* dengan band frekuensi yang dominan setiap 4 detik, dan hubungannya dengan peningkatan *fitness* untuk jenis *obstacle* yang cocok dan yang tidak cocok.Perhitungan dilakukan dengan cara menghitung jumlah gen keseluruhan, jumlah gen yang cocok, dan jumlah gen nol, dimana gen nol adalah gen yang menunjukkan bahwa di titik tersebut tidak ada *obstacle* yang akan diletakkan di dalam area permainan.Langkah berikutnya adalah membandingkan jumlah gen yang cocok dengan jumlah gen bukan nol (didapatkan dengan mengurangkan jumlah gen keseluruhan dengan jumlah gen nol) untuk mendapatkan persentase perbandingannya. Pengujian dilakukan menggunakan dua lagu yang berbeda dengan masing-masing menggunakan nilai penambahan fitness 600-450, 600-150, dan 600-300.Penjelasan mengenai kriteria dalam pengujian adalah sebagai berikut:

- Fitness cocok : Perubahan pada fitness jika gen cocok dengan band frekuensi dominan
- Fitness tidak cocok : Perubahan pada fitness jika gen tidak cocok dengan band frekuensi dominan
- Total gen per kromosom : Panjang dari sebuah kromosom yang menentukan jumlah gen/*obstacle*

- Fitness kromosom tertinggi : Fitness tertinggi sebuah kromosom dari hasil pengujian *genetic algorithm* selama 10 menit
- Total gen cocok : Jumlah gen yang cocok dengan band frekuensi dominan untuk setiap segmen
- Total gen bukan nol : Jumlah gen yang bukan bernilai nol (mewakili tidak ada *obstacle*) untuk setiap kromosom
- Kecocokan : Persentase dari Total gen cocok dibandingkan dengan Total gen bukan nol

Tabel 1. Pengujian kecocokan lagu pertama

Kriteria	Pengujian 1	Pengujian 2	Pengujian 3
Fitness cocok	+600	+600	+600
Fitness tidak cocok	+450	+150	+300
Total gen per kromosom	440	440	440
Fitness kromosom tertinggi	301800	266700	301800
Total gen cocok	282	350	306
Total gen bukan nol	423	384	392
Kecocokan	**66.66%**	**91.1%**	**78.06%**

Tabel 2. Pengujian kecocokan lagu kedua

Kriteria	Pengujian 1	Pengujian 2	Pengujian 3
Fitness cocok	+600	+600	+600
Fitness tidak cocok	+450	+150	+300
Total gen per kromosom	598	598	598
Fitness kromosom tertinggi	438750	360000	416520
Total gen cocok	450	375	450
Total gen bukan nol	520	407	517
Kecocokan	**86%**	**92%**	**87%**

Dari pengujian-pengujian pada tabel 1 dan 2 , dapat disimpulkan untuk masalah kecocokan jenis *obstacle* dengan band frekuensi yang dominan lebih baik menggunakan peningkatan *fitness* 600-150 dengan persentase kecocokan 91.1% untuk lagu Dramatic dan 92% untuk lagu at2.

4.3 Pengujian kecocokan tingkat kesulitan yang ditentukan

Pengujian kecocokan dengan tingkat kesulitan dilakukan untuk mengetahui kecocokan tingkat kesulitan yang dipilih dengan kromosom hasil *genetic algorithm* dimana akan dilakukan perbandingan jumlah *obstacle*.Untuk pengujian, masing-masing tingkat kesulitan memiliki nilai yang dikaitkan dengan tingkat kesulitan itu sendiri, dimana untuk *easy* adalah 3, *normal* adalah 5, dan *hard* adalah 7.Pengujian dilakukan dengan menggunakan lagu Gavin G. – Dramatic.

Total *obstacle* yang diharapkan didapatkan dengan mengalikan nilai tingkat kesulitan dengan jumlah segmen 4 detik dalam permainan(durasi dibagi 4). Total *obstacle* yang dihasilkan oleh *genetic algorithm* akan dibandingkan dengan *obstacle* yang diharapkan untuk mendapatkan persentase kecocokan.Rata-rata *obstacle* per 4 detik dihitung untuk menunjukkan penyimpangan nilai yang diharapkan dengan nilai yang terjadi, dimana rata-rata *obstacle* per 4 detik yang terjadi dihitung dengan membagi total *obstacle* dalam kromosom dengan jumlah segmen 4 detik dalam permainan.Penjelasan mengenai kriteria dalam pengujian adalah sebagai berikut :

- Total *obstacle* yang diharapkan : Jumlah *obstacle* yang diharapkan sesuai dengan tingkat kesulitan dimana easy adalah 3, normal adalah 5, dan hard adalah 7, dikalikan dengan jumlah segmen yang didapatkan dengan membagi durasi lagu dengan 4.
- Total *obstacle* yang ada : Jumlah *obstacle* yang ada di dalam kromosom dengan *fitness* terbaik yang diwakili dengan nilai 1 sampai 4 dalam gen
- Rata-rata *obstacle* / 4 detik(harapan) : Jumlah *obstacle* yang diharapkan per 4 detik, didapatkan dengan membagi total *obstacle* yang diharapkan dngan jumlah segmen
- Rata-rata *obstacle* / 4 detik(realita) : Jumlah *obstacle* yang ada dalam kromosom per 4 detik, didapatkan dengan membagi total *obstacle* dalam kromosom dengan jumlah segmen

Tabel 3. Tabel pengujian kecocokan tingkat kesulitan

Kriteria	Easy	Normal	Hard
Total obstacle yang diharapkan	165 *obstacle*	275	385
Total obstacle yang ada	286 *obstacle*	215	322
Rata-rata *obstacle* / 4 detik(harapan)	3	5	7
Rata-rata *obstacle* / 4 detik(realita)	5.2	3.9	5.86
Kecocokan	**57.7%**	**78.18%**	**83.63%**

4.4 Pengujian jumlah iterasi

Pengujian jumlah iterasi dilakukan untuk mencari batas perhentian genetic algorithm yang paling cocok dengan algoritma yang diterapkan. Hasil dari pengujian ini akan digunakan untuk

mendapatkan persentase fitness yang harus dicapai sebelum genetic algorithm berhenti melakukan iterasi, dimana perhitungan dilakukan dengan mendapatkan nilai saat *fitness* tertinggi dengan iterasi saat *fitness* digunakan untuk didapatkan rata-ratanya, dimana rata-rata dari dua lagu ini akan dirata-rata kembali untuk dijadikan nilai yang akan diimplementasi. Pengujian dilakukan dengan durasi 10. Tabel 4 menunjukkan hasil pengujian dengan menggunakan lagu pertama, yaitu Gavin G. – Dramatic. Tabel 5 menunjukkan hasil pengujian lagu kedua, yaitu at2.

Tabel 4. Pengujian iterasi dengan lagu pertama

Pengujian	1	2	3	4	5
Fitness uji maks	266700	259350	300300	280500	279150
Rata-rata	277200				

Tabel 5. Pengujian iterasi dengan lagu kedua

Pengujian	1	2	3	4	5
Fitness uji maks	360000	360000	393750	393750	360000
Rata-rata	373500				

Tabel 6. Pengujian akhir rata-rata GA

Nilai uji	277200	373500
Fitness max lagu	313500	427500
Persentase	88.42%	87.3%
Rata-rata Persentase	87.86%	

Berdasarkan hasil perhitungan tabel 6, rata-rata pengujian menunjukkan hasil 87.86% nilai fitness dari fitness maksimal dalam lagu. Dari hasil tersebut, ditentukan nilai 87.86% sebagai batas berhentinya proses GA ketika fitness tertinggi dalam populasi mencapai persentase yang telah ditentukan dari fitness maksimal yang bisa didapatkan.

5. KESIMPULAN

Dari hasil perancangan dan pembuatan implementasi penggunaan audio spectrum dan genetic algorithm untuk obstacle generation game platforming berbasis musik, dapat diambil kesimpulan antara lain :

- Rancangan *genetic algorithm* dapat berjalan dengan mengikuti parameter-parameter yang diberikan dan sesuai dengan rancangan awal.
- Program dapat melakukan input berupa *audio* dalam format OGG.
- Program dapat menghasilkan susunan *obstacle* sesuai kromosom dari *genetic algorithm* dengan rata-rata kecocokan 91.55% dari hasil pengujian yang dilakukan.
- Fitness yang harus dicapai untuk menyatakan bahwa genetic algorithm telah mencapai konvergen adalah ketika fitness tertinggi populasi mencapai 87.86% maksimal fitness yang bisa didapatkan.

6. DAFTAR PUSTAKA

[1] Anonymous. Genetic Algorithm. Diambil kembali dari mathworks.com:ttps://www.mathworks.com/discovery/genetic-algorithm.html

[2] Connor, D. 2008 Sample Rate and Bitrate: The Guts of Digital Audio. Diambil kembali dari thestereobus.com: https://thestereobus.com/2008/01/12/sample-rate-and-bitrate-the-guts-of-digital-audio/

[3] Naletto, A. 2011. Audio Analysis: Isolating Frequency Values.Diambil kembali dari answers.unity.com: https://answers.unity.com/questions/175173/audio-analysis-pass-filter.

[4] Politis & Dionysios. 2016. Digital Tools for Computer Music Production and Distribution.

[5] Robertson, H. 2010. Digital Audio Sampling. Diambil kembali dari videomaker.com : https://www.videomaker.com /article/c4/14524-digital-audio-sampling

[6] Walker, L. 2017. Audio Spectrum Explained. Diambil kembali dari teachmeaudio.com: https://www. teachmeaudio.com /mixing/techniques/audio-spectrum/

[7] Wolf, M. 2012. Encyclopedia of Video Games: The Culture, Technology, and Art of Gaming.

[8] Wolfe, J. What is a Sound Spectrum? .Diambil kembali dari unsw.edu.au: http://newt.phys.unsw.edu.au /jw/sound.spectrum.html