Game Membangun Kerajaan dengan Procedural Generated Map Menggunakan *Perlin Noise*

Joshua Ginove Hertanto Wijaya[1], Justinus Andjarwirawan[2], Rudy Adipranata[3]
[1,2,3]Program Studi Teknik Informatika, Fakultas Teknologi Industri, Universitas Kristen Petra
Jln. Siwalankerto 121-131 Surabaya 60236
Telp. (031) – 2983455, Fax. (031) - 8417658
rxjoshua14@gmail.com, justin@petra.ac.id, rudya@petra.ac.id

ABSTRAK

Video game sudah banyak beredar pada zaman sekarang. Salah satu *genre* yang digemari adalah *city building survival*. Pada tipe game ini *map* berperan sangat penting karena pemain akan membangun bangunannya diatas *map*. Pada penelitian ini akan digunakan metode *perlin noise* untuk membuat *map* secara *procedural* sehingga pada setiap permainan *map* akan berbeda.

Program berupa *game* yang dibuat memiliki 2 metode untuk *generate map*. Metode pertama dengan *perlin noise* dan metode kedua dengan *random.* Kedua metode akan dibandingkan metode manakah yang dapat menghasilkan *map* yang lebih baik.

Uji coba dilakukan melalui kuisioner setelah responden mencoba permainan dengan kedua metode *map generation*. Hasil dari kuisioner menyimpulkan bahwa metode *perlin noise* dapat membuat *map* yang terlihat lebih natural dan *balance* daripada *random*.

Kata Kunci: *Perlin Noise, City Building Survival, Real-Time-Strategy, Video Game*

ABSTRACT

Today, video game has been widespread. One of the favorite genre is city building survival. In this kind of game the map has a big role because the players are going to build their buildings on the map. In this study perlin noise will be used to generate the map procedurally so in each game the map will be different.

The program is going to be a game that has 2 methods of map generation. The first method will use perlin noise and the second mehod will use random to generate the map. Both methods will be compared to determine which method will produce a natural looking and balanced map.

To test which method could generate a natural looking and balanced map, respondent will be asked to play the game in both methods then answer the question from the survey. The survey result concluded that perlin noise could generate a more better looking and balanced map than random.

Keywords: *Perlin Noise, City Building Survival, Real-Time-Strategy, Video Game*

1. PENDAHULUAN

Pada zaman modern ini, *video game* sudah berkembang dan menyebar luas. Salah satu jenis *video game* yang digemari adalah *Construction and management Simulation.* Pemain berperan merancang pembangunan untuk memenuhi kebutuhan seperti makanan, tempat tinggal, dan pertumbuhan ekonomi.

Salah satu unsur terpenting dalam jenis *game Construction and management Simulation* adalah map yang digunakan. Pemain harus bisa mengatur strateginya dalam mengembangkan kotanya untuk beradaptasi dengan *map* yang ada. *Map* bisa dibuat oleh *developer game* atau di-*generate* secara *random* atau istilahnya adalah *procedural content generation (*PCG*)*. Map yang dibuat oleh *developer game* terbatas jumlahnya, sehingga seiring waktu pemain akan kehabisan *map* baru untuk dimainkan. Oleh karena itu, *map* perlu di-*generate* secara *random* untuk membuat banyak variasi *map*.

Terdapat banyak cara untuk membuat PCG seperti murni secara *random*, *Midpoint displacement algorithm*, *The diamond-square algorithm*, dan *perlin noise*. Murni secara *random* hasilnya tidak akan terlihat natural. *Midpoint displacement algorithm* dan *The diamond-square algorithm* mempunyai hasil yang cukup baik. Tetapi *perlin noise* mempunyai kelebihan karena lebih fleksibel dan tidak bergantung pada *size maximum* seperti *Midpoint displacement algorithm* [7]. Sedangkan *The diamond-square algorithm* memiliki hasil yang tidak terlalu baik pada bagian *Edge* [1].

Perlin noise yang sering digunakan untuk membuat *map*. Karena *perlin noise* merupakan *gradient noise* sehingga perbedaan value dari 1 titik ke titik lainnya lebih terlihat. Penelitian yang dilakukan oleh Olsson dan Frank pada 2017 berhasil menata sebuah kota menggunakan *perlin noise* yang terlihat natural [4]. Oleh karena itu *perlin noise yang akan digunakan.*

2. LANDASAN TEORI

2.1 *Procedural Content Generation*

PCG adalah pengaplikasian komputer untuk membuat konten dari game dengan *input* yang terbatas dari *user*. Dari *input* yang terbatas komputer mengembangkan dan membuat konten yang besar. Konten yang dimaksud dapat berupa *map*, *level*, peraturan game, tekstur, cerita, *items*, *quests*, musik, senjata, karakter, dan lain-lain [9].

2.2 *Perlin Noise*

Perlin Noise adalah sebuah *gradient noise* yang dibuat oleh Ken Perlin pada 1983 dan diimprovisasi pada 2002. *Perlin noise menggunakan* kotak pembantu dan sebuah *vector* pada setiap suduh kotak tersebut [8, 10].

2.3 *Isometric Projection*

Isometric Projection adalah teknik untuk mempresentasikan objek 3D pada gambar 2D. Untuk membuat *isometric projection* maka *projection plane* harus dengan 3 axis projeksi pada derajat yang sama, yaitu 120^{o} [3].

2.4 Perkembangan *Map* pada *Games*

Perkembangan map untuk game yang tidak digenerate secara prosedural bergantung pada map awal yang didesain oleh developer game dan community made map yang disebar luaskan melalui platform seperti steam workshop [2].

Game yang menngunakan sering menggunakan PCG untuk generate map adalah game bertipe sandbox. Game melakukan generate map secara berbagai metode random yang salah satunya adalah perlin noise, untuk dimainkan oleh pemain. Contoh game yang mapnya dibuat dengan PCG adalah Minecraft dan Spore [5].

2.5 *City Building Games*

Game pada genre ini mempunyai gameplay di mana pemain dapat mengelolah kotanya dengan cara berinteraksi dengan map yang ada. Interaksi dilakukan dengan cara membangun bangunan di atas map. Pemain membangun bangunan yang dapat menghasilkan uang untuk dapat membangun kotanya lebih lanjut. Contoh game pada genre ini seperti sim city, cities skylines, frostpunk, dan tropic [4].

3. DESAIN SISTEM

3.1 Desain *Game*

Game yang akan dibuat bernama "Wonder". *Game* ini adalah *game survival-city-building* di mana pemain dapat membuat bangunan-bangunan untuk membentuk kerajaannya. *Map* pada *game* akan di-*generate* secara random menggunakna *perlin noise* diawal permainan.

Alur cerita pada game adalah pemain mempunyai kerajaan baru di tanah baru yang tidak berpenghuni. Tetapi dalam perkembangannya kerajaan tersebut diserang oleh musuh berupa binatang-binatang alien. musuh tersebut muncul dari portal misterius. Tujuan pemain adalah membangun kerajaan yang makmur dan membangun *wonder* untuk menutup portal.

3.2 *Map Generation*

Untuk melakukan *generate map* pada game dilakukan dengan 2 cara yaitu, *perlin noise* dan *random* sebagai pembanding. Pada metode perlin noise dilakukan *generate noise* sebanyak 3 kali. Yang pertama untuk menentukan wilayah mana yang adalah *swamp* dan *water*. Yang kedua adalah untuk menentuka *terrain* apa yang akan muncul pada 1 koordinat di *map*. Yang ketiga adalah untuk memetakan perseberan dari *natural resources* pada map.

Pada metode *random* dilakukan *random terrain* apa yang akan di munculkan kemudian titik dan radiusnya. Pada titik tersebut dan sekitarnya tergantugn dari besar radius akan di-*set* semua menjadi *terrain* yang telah di-*random* di awal. Hal yang sama dilakukan pada *natural resources*.

Setelah selesai melakukan *generate map* dilakukan *generate bridge* awal yang adalah bangunan *bridge* tetapi bukan dibuat oleh pemain, melainkan sudah di-*generate* pada saat awal ketika pembuatan map.

3.3 Desain *Map*

Map pada game terdiri dari *tile* sebesar 100x100. Tiap *tile* terdiri dari 1 *terrain* dan 1 objek diatasnya. Objek tersebut dapat berupa bangunan atau *natural resources.*

3.3.1 Terrain & Resources

Terrain yang terdapat dalam *game* adalah *plain land, high land,water, swamp, dan mountain*. Sedangkan *natural resources* yang terdapat dalam *game* adalah *tree, mountain, gold deposit*, dan *iron deposit*.

3.3.2 Buildings

Terdapat 22 bangunan yang dapat dibuat oleh pemain. Bangunan tujuan dari pemain yaitu *wonder*, pemain akan menang jika berhasil membuat bangunan ini. Bangunan penambah populasi *wooden house, stone house, brick house* serta *builder hut* untuk menambah jumlah *builder*. *Supply depot* untuk menambah area pembangunan. Bangunan ekonomi yang dapat menghasilkan *resources farmmill, farm, pasture, fishing hut, logging center, stone mine, gold mine,* dan *iron mine.* Bangunan *upgrade* untuk bangunan ekonomi *sawmill, stone cutter,* dan *smeltery*. Bangunan pertahanan untuk menyerang musuh *watchtower, guntoewr, cannontower,* dan *fort*. *Bridge* untuk menyebrangi *water*.

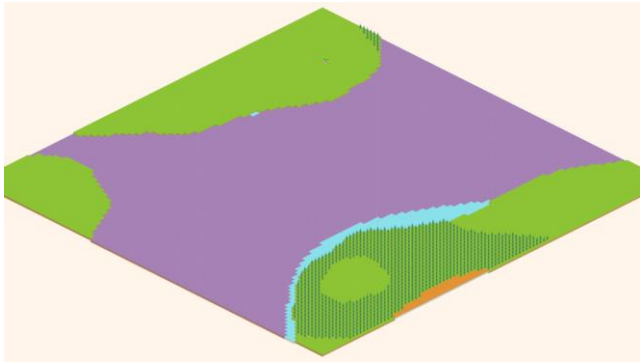
4. PENGUJIAN SISTEM

4.1 *Map Generation*

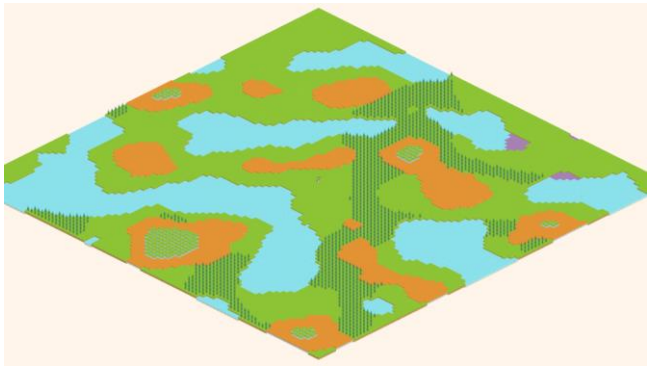
4.1.1 Perlin Noise

Hasil dari *map* yang di-*generate* dengan metode *perlin noise* pada ukuran map 100 blok x 100 blok dengan parameter berbeda seperti pada Gambar 1, Gambar 2, Gambar 3.

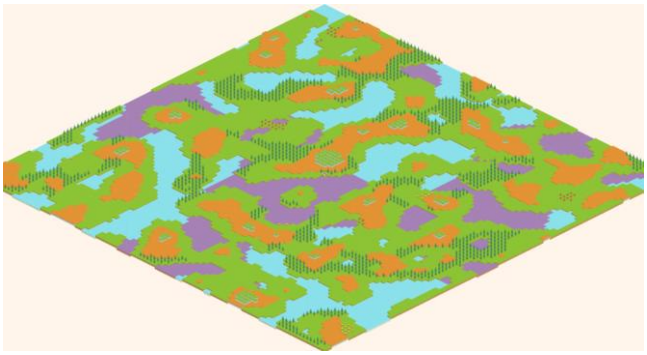
Kombinasi *perlin noise* yang digunakan dalam pengujian adalah menggunakan *octave perlin* dengan *octave* 16 dan *persistance* 0,25. *Size perlin noise* untuk *terrain* adalah 10 sedangkan untuk *resources* adalah 1,5.



Gambar 1. *Perlin noise* dengan *size* 1



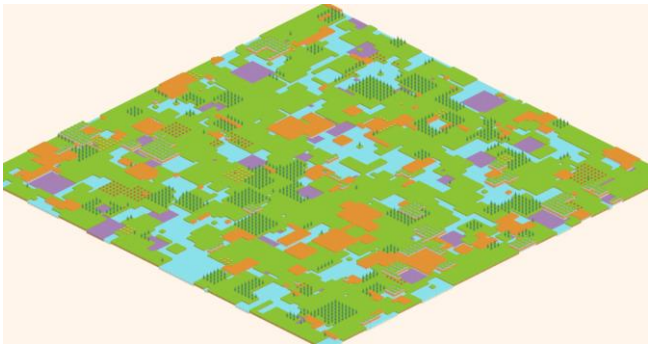
Gambar 2. *Perlin noise* dengan *size* 5



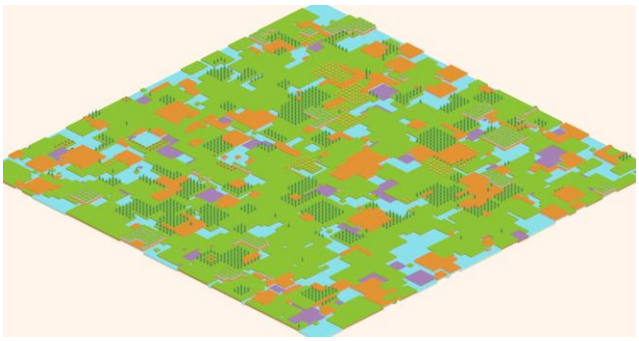
Gambar 3. *Perlin noise* dengan *size* 10

4.1.2 Random

Hasil dari *map* yang di-*generate* dengan metode *random* seperti pada Gambar 4, Gambar 5.



Gambar 4. Contoh *random map 1*



Gambar 5. Contoh *random map* 2

4.2 Implementasi Desain *Interface*

Pada Gambar 6, adalah tampilan *ingame menu* dari *game wonder*. Pemain dapat menggerakkan kamera dengan arah panah atau WASD pada keyboard, tombol spasi berguna untuk memindahkan kamera menuju *castle* sehingga pemain tidak kehilangan arah dimana *castle* berada. Pemain memiliki *resources* pada *panel* bagian atas. Dapat melakukan klik pada *terrain* di area permainan bagian tengah untuk memilih dimanakah akan membangun bangunan. Serta, dapat memilih bangunan apa yang akan dibangun pada *panel* bagian bawah yang bisa di-*scroll*. Pilihan paling terakhir dari *panel* bagian bawah adalah untuk melakukan *demolish building*, pemain dapat menghancurkan bangunan yang telah dibangun dengan tombol tersebut.

Gambar 6. Tampilan ketika *game* dimulai

Seperti pada Gambar 6, dalam game pemain dapat membangun berbagai macam bangunan. Dalam contoh terdapat *castle* dan *wooden house* pada bagian kanan atas. Beberapa bangunan ekonomi seperti *farm*, *logging center*, dan *fishing hut* yang dapat menghasilkan *resources* seperti *food* dan *wood*.

Gambar 7. Tampilan ketika membeli *troop*

4.3 Pengujian hasil *map generation* dengan kuisioner

Setelah program dibuat, beberapa responden diminta untuk memainkan *game wonder*. Responden diminta untuk mencoba permainan dengan metode *map generation perin noise* kemduian memainkan lagi dengan metode *map generation random*. Setelah memainkan game responden diminta untuk menjawab kuisioner. Terdapat 14 responden yang menjawab kuisioner.

Tujuan dari kuisioner adalah untuk melakukan pengujian terhadap aspek variatif, *natural*, dan *balance* dari map yang dihasilkan. Juga dilakukan perbandingan antara *map* yang dihasilkan dengan *perlin noise* dan *random* yang mana hasilnya lebih baik menurut kriteria tersebut.

Hasil dari jawaban kuisioner adalah 85.7% responden sudah pernah memainkan *game* dengan *genre city building survival* dan semua responden setuju bahwa variasi map sangatlah penting. Sebanyak 85.7% responden juga menjawab bahwa *map* yang dihasilkan dengan metode *perlin noise* sudah terlihat natural. Sedangkan untuk sisi *gameplay balance* seperti kecukupan *resources* dan tempat untuk membangun bangunan 78.6% responden menjawab bahwa *map* yang dihasilkan dengan metode *perlin noise* sudah *balance*.

Untuk perbandingan dengan metode *random*, 85.7% responden menjawab bahwa *map* yang dihasilkan oleh *perlin noise* terlihat lebih natural daripada *map* yang dihasilkan oleh *random*. 78.6%

responden menjawab bahwa *map* dari metode *perlin noise* lebih *balance* untuk sisi *gameplay* daripada metode *random*.

Kesimpulan dari pengujian kuisioner adalah map yang dihasilkan dari *perlin noise* sudah bisa terlihat natural dan *balance* untuk game *wonder*. *Perlin noise* juga mampu menghasilkan map yang lebih natural dan *balance* dari metode *random*.

5. KESIMPULAN

Dari penelitian yang dilakukan dapat diambil kesimpulan sebagai berikut:

- *Perlin noise* dapat menghasilkan *map* yang terlihat lebih natural dan *balance* dari pada *random*.
- Perlu dilakukan pengolahan lebih lanjut untuk menghasilkan *resources* yang lebih *balance* terutama untuk *gold deposit* dan *iron deposit*.
- Hasil dari *noise* yang dihasilkan oleh *perlin noise* dari skala 0 – 1 lebih banyak menempati posisi di daerah 0.5.

6. DAFTAR REFERENSI

[1] Archer, T. 2011. *Procedurally Generating Terrain* (Unpublished undergraduate thesis). Morningside College, Sioux City, USA.

[2] Barros, G. A., & Togelius, J. 2015. *Balanced civilization map generation based on Open Data*. *2015 IEEE Congress on Evolutionary Computation* (CEC).

[3] Carlbom, I., & Paciorek, J. 1978. *Planar Geometric projections and Viewing Transformations.* ACM Computing Surveys. ACM. Vol.10 No.4

[4] Korppoo, K. 2015. *Designing Game Analytics For A City-Builder Game.* (Unpublished master thesis). University of Tampere, Finland.

[5] Lara-Cabrera, R., Cotta, C., & Fernández-Leiva, A.J. 2013. A Procedural Balanced Map Generator with Self-adaptive Complexity for the Real-Time Strategy Game Planet Wars. *Applications of Evolutionary Computation 16th European Conference (pp 274-283).* Vienna, Austria.

[6] Olsson, N., & Frank, E. 2017. *Procedural City Generation using Perlin Noise* (Unpublished undergraduate thesis). Blekinge Institue of Technology, Sweden.

[7] Olsen, J. 2004. *Realtime procedural terrain generation – realtime synthesis of eroded fractal terrain for use in computer games.* University of Southern Denmark.

[8] Perlin, K. 2002. Improved Noise reference implementation. Retrieved November 22, 2017, from http://mrl.nyu.edu/~perlin/noise/

[9] Shaker N., Togelius, J., & Nelson, M.J. 2016. *Procedural Content Generation in Games: A Textbook and an Overview of Current Research.* Springer.

[10] Zucker, M. 2001. The Perlin noise math FAQ. Retrieved November 22, 2017, from https://mzucker.github.io/html/perlin-noise-math-faq.html#algorithm