Rancang Bangun Pembangkit *World* Dinamis menggunakan Algoritma *Recursive Backtracking* pada *Game 2D Platformer* "Mine Meander"

Faishal Azka Jellyanto, Imam Kuswardayan, dan Nanik Suciati
Jurusan Teknik Informatika, Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember (ITS)
Jl. Arief Rahman Hakim, Surabaya 60111 Indonesia
e-mail: imam@its.ac.id, nanik@if.its.ac.id, faishal.azka12@mhs.if.its.ac.id

***Abstrak*—Salah satu *genre game* yang cukup banyak diminati adalah *genre 2D Platformer*. Dalam perkembangan *game 2D Platformer*, kebanyakan *game* memiliki jenis *map* yang bersifat statis. Statis dalam hal ini artinya lingkungan atau area permainan akan tetap sama setiap dimainkan kembali. Dengan jenis *map* yang statis ini, tentunya bisa membuat beberapa orang merasa cepat bosan. *Game* Mine Meander adalah *game 2D Platformer* dengan fitur *Dynamic World Generator* dimana pemain bisa menjumpai bentuk *world map* yang berbeda dan juga menjumpai platform atau rintangan yang letaknya selalu berubah setiap kali dimainkan kembali. Dalam pembuatan *layout map*, metode yang digunakan adalah Algoritma *Recursive Backtracking*. Kemudian platform dan tangga ditata supaya *world map* selalu mempunyai jalan keluar bagi pemain untuk dapat menyelesaikan level. Dalam proses pengujian, dilakukan pengecekan apakah bentuk *world map* berubah saat memainkan level tertentu sebanyak 2 kali. Kemudian hasilnya menunjukkan bahwa bentuk *world map* dapat berubah dan selalu ada jalan keluar untuk menyelesaikan level tersebut.**

***Kata Kunci*—2D Platformer Game, Algoritma *Recursive Bactracking, Dynamic World Generator*, Platform**

I. PENDAHULUAN

SAAT ini perkembangan dunia *game* semakin meningkat secara pesat, mulai dari *genre* dan berbagai aturan main (*gameplay*) yang baru. Salah satu genre *game* yang cukup banyak diminati adalah genre *2D Platformer*. Pada genre *game* ini, pemain harus menggerakkan karakternya menyusuri *map* yang tersusun dari daratan (platform), tangga, dan beberapa rintangan. Contoh *game platformer* yang banyak dikenal antara lain Terraria, Donkey Kong, Space Panic, dan Mario Bros. Namun kebanyakan *game platformer* saat ini cenderung memiliki bentuk *world map* yang statis dimana posisi dan letak platform akan selalu tetap sama pada satu level. Dengan bentuk *map* yang statis seperti ini tentunya akan membuat beberapa pemain cepat merasa bosan.

Untuk dapat mengatasi hal tersebut, diperlukan bentuk *world map* yang dinamis dimana *world map* dapat terus berubah ketika pemain mengulang kembali suatu level. Salah satu Algortima yang bisa digunakan untuk membuat *world map* dinamis adalah Algortima *Recursive Backtracking*. Algoritma *Recursive* Backtracing pada penelitian ini digunakan untuk menentukan *layout* utama dari *world map*. Dari *layout* utama yang sudah dibentuk secara *random*, barulah diletakkan platform, tangga, dan rintangan. Saat pemain mengulangi kembali level tersebut, program akan me-*generate* ulang *layout* utama.

Usulan dari penelitian ini adalah membuat suatu *game 2D Platformer* dengan fitur *world map* yang dinamis menggunakan Algoritma *Recursive Backtracking*. Dengan adanya fitur yang baru ini, diharapkan pemain akan menjadi lebih tertarik untuk bermain *game 2D Platformer* dan tidak cepat merasa bosan.

II. TINJAUAN PUSTAKA

A. 2D Platformer Game

2D Platformer Game adalah salah satu *genre video game* yang memiliki *gameplay* meliputi perjalanan antar platform [1]. Platform dalam konteks ini adalah objek berupa daratan yang berfungsi sebagai tempat berpijak karakter dalam permainan [7]. Elemen-elemen lain yang seringkali dimasukkan pada *genre* ini antara lain berlari, melompat, menaiki atau menuruni tangga, dan menghindari rintangan. Contoh dari *game 2D platformer* yang terkenal adalah: Donkey Kong, Space Panic, dan Mario Bros.

B. LibGDX

LibGDX adalah suatu *framework* untuk pengembangan *game* yang sebagian besar ditulis dalam bahasa pemrograman Java. LibGDX bersifat gratis dan *cross*-platform dimana *game* yang dihasilkan dapat dijalankan di beberapa platform seperti Windows, Linux, Mac OS X, Android, iOS, dan HTML [4].

Proses pengembangan dari LibGDX secara umum menggunakan bahasa pemrograman Java dan pengguna disarankan untuk menggunakan IDE Eclipse, meskipun pada penerapannya dimungkinkan untuk menggunakan IDE lain seperti Netbeans atau Android Studio. Hampir sama seperti Unity, meskipun LibGDX tersedia untuk berbagai macam platform namun kelebihan LibGDX ini adalah proses *development* dan *debugging* dapat dilakukan hanya dengan menggunakan komputer *desktop* tanpa memerlukan perangkat

lain. Setelah *debugging* siap maka pengguna dapat melakukan proses kompilasi dan uji coba sesuai dengan platform yang diinginkan. Hal ini sangat memudahkan karena proses kompilasi dan proses berjalannya program menjadi lebih cepat.

Untuk menangani masalah *math* dan *physics*, LibGDX menyediakan ekstensi Box2D. Pada LibGDX juga tersedia ekstensi lain seperti *Controllers*, AI, *Tools* (2D/3D *editor*), dan lain-lain.

Beberapa fitur lainnya adalah LibGDX mampu menangani berbagai jenis masukan seperti *keyboard*, *mouse*, *touch screen*, bahkan lengkap dengan *gesture detector*. Untuk masalah penyimpanan progres *game*, LibGDX sudah menyediakan semacam kelas yang berfungsi sebagai *file manager*. *Output file* yang disediakan dapat berupa *file* json maupun xml.

C. Android Studio

Android Studio merupakan sebuah *Integrated Development Environment* (IDE) khusus untuk membangun aplikasi yang berjalan pada platform android. Android studio ini berbasis pada IntelliJ IDEA [2], sebuah IDE untuk bahasa pemrograman Java. Bahasa pemrograman utama yang digunakan adalah Java, sedangkan untuk membuat tampilan atau *layout*, digunakan bahasa XML. Android studio juga terintegrasi dengan Android *Software Development Kit* (SDK) untuk *deploy* ke perangkat Android. Sebagai pengembangan dari Eclipse, Android Studio mempunyai banyak fitur-fitur baru dibandingkan dengan Eclipse. Berbeda dengan Eclipse yang menggunakan Ant, Android Studio menggunakan Gradle sebagai lingkungan *build*-nya.

D. Tiled

Tiled adalah suatu *map* editor yang bertujuan khusus dalam pembuatan *map* berbasis *tile* [5]. Aplikasi ini menyediakan *tool* gratis untuk membuat *layout map* dalam *game*. *Tiled* mampu mengolah objek-objek seperti *background map*, *sprites*, *collision area*, hingga posisi *spawn* musuh. Semua data ini kemudian disimpan ke dalam format .tmx.

Selain berfungsi sebagai *map* editor dalam pengolahan *tiled-based map*, Tiled juga bisa digunakan sebagai level editor. Dengan Tiled, pengguna dapat menentukan ukuran-ukuran dari tiap *tile* pada gambar sehingga pengguna dapat membuat *map* tanpa dibatasi oleh ukuran tetap dari gambar.

E. Algoritma Recursive Backtracking

Algoritma *Backtracking* pertama kali diperkenalkan oleh D.H. Lehmer pada tahun 1950. Algoritma ini cukup praktis untuk digunakan dalam beberapa penyelesaian masalah dan juga untuk memberikan kecerdasan buatan dalam *game*. Beberapa *game* seperti sudoku, *maze* (labirin), atau catur juga bisa diimplementasikan dengan Algoritma *Backtracking* [3].

Pada proses pembentukan *Maze*, Algoritma ini awalnya memilih jalan secara *random* dari ruang-ruang yang tersedia. Apabila pada langkah tertentu proses pemilihan jalan tidak bisa dilanjutkan dan masih terdapat ruang yang belum dikunjungi, maka akan mundur (*backtrack*) ke langkah sebelumnya dan akan memilih jalan lain yang belum pernah dikunjungi [6]. Proses akan berhenti apabila telah mengunjungi setiap ruang.

III. ANALISIS DAN PERANCANGAN

A. Analisis Permasalahan

Kemajuan teknologi informasi menjadikan perkembangan dunia *game* menjadi semakin bermacam-macam jenisnya baik dalam hal *genre*, *gameplay*, serta kualitas grafis yang diimplementasikan. Di lingkungan pasar *game*, terdapat banyak sekali *game* yang memiliki genre *2D Platformer*. *Game 2D Platformer* ini tentunya memiliki fitur dan keunikan masing-masing.

Game ini dibangun dengan tujuan memberikan inovasi dan hiburan kepada pemain *game* khususnya pecinta *game 2D Platformer*. *Game* ini juga diharapkan dapat meningkatkan ketertarikan seseorang pada *game* genre *2D Platformer*.

B. Deskripsi Umum Aplikasi

Aplikasi yang akan dibuat dalam penelitian ini adalah sebuah permainan berjenis *2D Platformer*. Permainan ini hanya bisa dimainkan oleh satu orang. Fokus utama dari permainan ini terletak pada bentuk *world map* yang dinamis. Pada setiap level pemain akan menjumpai bentuk *world map* yang berubah-ubah. *World map* yang dinamis ini dibentuk dengan menggunakan Algoritma *Recursive Backtracking*. Setiap level dari permainan ini juga memiliki tingkat kesulitan yang berbeda-beda. Semakin tinggi level yang dipilih maka tingkat kesulitan yang dihadapi pemain untuk menyelesaikan level tersebut menjadi semakin sulit. Pemain dinyatakan menang apabila mampu menemukan portal atau pintu keluar.

C. Perancangan Aturan Main

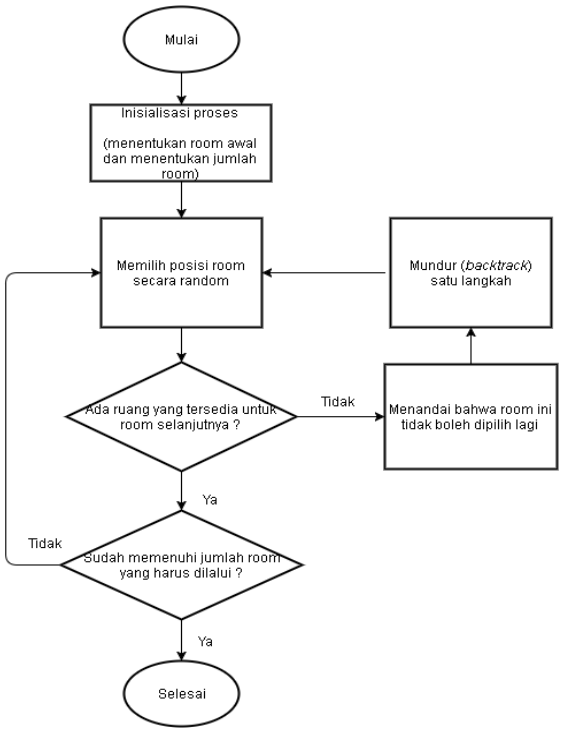
Aturan main secara umum dari permainan ini adalah pemain bisa menggerakkan karakternya secara horizontal, melompat, dan bisa menaiki atau menuruni tangga. Selama permainan berlangsung, pemain akan menemukan kristal-kristal di dalam *map*. Kristal-kristal tersebut berfungsi untuk menambah perolehan skor dari pemain.

Di dalam *map* juga terdapat beberapa rintangan dimana apabila karakter dari pemain bersentuhan dengan rintangan tersebut, maka *Heart* karakter akan berkurang. Rintangan yang ada di dalam *world map* antara lain *spike* (duri), *dangerous flower*, *spider*, dan *zombie*. *Spike* adalah jenis rintangan yang diam atau tidak bergerak. Apabila karakter melompat dan jatuh di atasnya, maka *Heart* karakter akan berkurang. *Dangerous flower* adalah jenis rintangan yang juga diam dan tidak bergerak. Namun apabila karakter bersentuhan baik dari samping atau atas/bawah, maka *Heart* karakter akan berkurang. *Spider* adalah jenis rintangan yang bisa bergerak apabila karakter mendekat ke tempat *spider* berada. *Heart* karakter juga akan berkurang apabila bersentuhan dengannya. *Zombie* juga merupakan jenis rintangan yang bisa bergerak secara horizontal dan bisa menaiki atau menuruni tangga. Apabila bersentuhan dengan *zombie*, *Heart* karakter akan berkurang.

Untuk dapat memenangkan level dalam permainan ini, pemain harus bisa menemukan dan masuk ke dalam portal atau pintu keluar yang berada pada *world map*. Apabila karakter kehabisan *life* (nyawa) sebelum bisa masuk ke dalam pintu keluar, maka permainan akan berakhir (*Game over*).

D. Perancangan World Map

World map merupakan area permainan dimana pemain menggerakkan karakternya. Pada permainan ini tiap satu level permainan memiliki satu *world map* yang dinamis atau bisa berubah-ubah. Pembuatan *world map* ini akan menggunakan Algoritma *Recursive Backtracking*. Diagram alur dari Algoritma bisa dilihat pada Gambar 1.

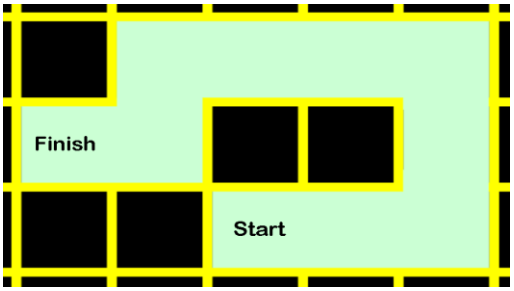


Gambar 1 Diagram alur Algoritma *Recursive Backtracking*

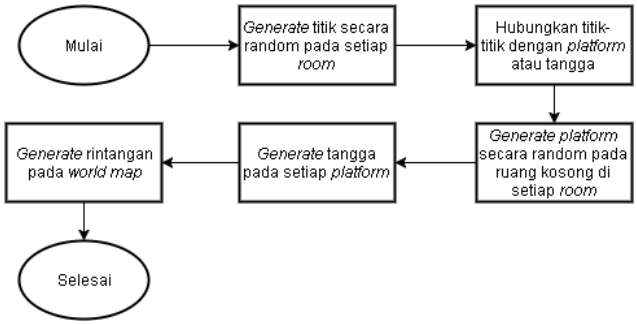
Langkah awal yang dilakukan adalah melakukan inisialisasi *room* pertama. *Room* adalah suatu ruang kecil yang berfungsi sebagai tempat menampung objek seperti karakter, platform, rintangan, dan objek-objek lain. Kemudian ditentukan berapa banyak *room* yang harus dibuat. Setelah *room* pertama selesai dibuat, Algoritma akan memilih secara *random room* mana yang akan dipilih selanjutnya. Apabila menemui jalan buntu dan belum mencapai jumlah minimal *room* yang harus ditempuh, maka *room* saat ini ditandai supaya pada langkah selanjutnya tidak akan dipilih lagi. Kemudian dilakukan *backtrack* satu langkah sehingga akan kembali ke posisi *room* yang sebelumnya. Algoritma ini akan berhenti apabila telah memenuhi jumlah *room* yang harus dibuat atau dilalui. Contoh ilustrasi dari proses pembentukan *layout* utama dari *world map* bisa dilihat pada Gambar 2.

Setelah *layout* utama *world map* selesai dibuat, langkah selanjtunya adalah mengisi tiap *room* dengan objek-objek. Diagram alur dari proses ini bisa dilihat pada Gambar 3.

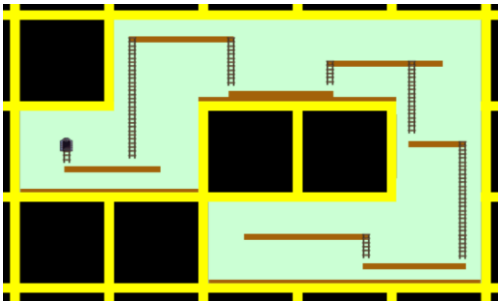
Dari diagram alur pada Gambar 3, proses pertama yang dilakukan ada me-*generate* titik-titik yang posisinya diatur secara *random* pada setiap *room*. Titik-titik ini berfungsi sebagai acuan jalan penghubung antar *room* dimana jalan ini bisa berupa tangga atau platform. Platform akan di-*generate* sepanjang selisih jarak horizontal dari *room* awal ke *room* selanjutnya. Sedangkan tangga akan di-*generate* sepanjang selisih jarak vertikal antara *room* awal dengan *room* selanjutnya. Khusus untuk titik pada *room* terakhir dari *world map*, akan diletakkan sebuah portal atau pintu keluar. Portal atau pintu keluar ini berfungsi sebagai syarat untuk dapat menyelesaikan level dimana pemain harus menggerakkan karakternya ke portal tersebut. Setelah langkah ini selesai, maka bisa dipastikan bahwa level atau *world map* ini memungkinkan untuk diselesaikan oleh pemain. Ilustrasi dari hasil proses ini bisa dilihat pada gambar 4.



Gambar 2 Contoh ilustrasi pembentukan *layout* utama *world map*

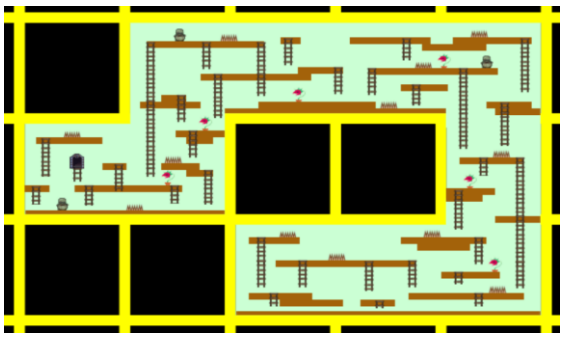


Gambar 3 Diagram alur proses pembentukan objek-objek di dalam *world map*



Gambar 4 Contoh lustrasi *world map* yang sudah diberi platform dan tangga

Langkah selanjutnya adalah mengisi bagian *room* yang kosong dengan platform dan tangga. Proses pembentukan platform dan tangga ini dilakukan secara *random*. Lalu dalam pembentukan platform ini harus dipastikan bahwa platform yang dibentuk tidak akan menutup jalan penghubung antar *room*. Setelah selesai membentuk platform dan tangga, langkah berikutnya adalah mengisi *world map* dengan rintangan. Sebelum meletakkan satu rintangan, pertama ditentukan dulu secara *random* platform mana yang akan dipilih sebagai tempat peletakkan rintangan. Setelah salah satu platform dipilih, kemudian rintangan diletakkan pada platform tersebut. Langkah ini diulangi kembali sampai mencapai jumlah rintangan yang telah ditentukan. Hasil dari proses ini diilustrasikan pada Gambar 5.



Gambar 5 Contoh ilustrasi *world map* yang telah diisi dengan platform, tangga, dan rintangan.

E. Perancangan Skenario dan Tingkat Kesulitan

Pada permainan ini pemain harus menggerakkan karakternya menyusuri *map* dan menemukan portal atau pintu keluar untuk dapat menyelesaikan satu level. Selama permainan berlangsung, karakter mempunyai 5 *Heart* dan 3 nyawa. Apabila karakter bersentuhan dengan salah satu rintangan, maka Heart akan berkurang 1. Lalu apabila karakter kehabisan *Heart*, maka nyawa akan berkurang 1 dan karakter akan kembali ke titik awal (*start point*). Setelah nyawa berkurang 1, *Heart* dari karakter akan kembali menjadi 3. Namun apabila karakter kehabisan nyawa, maka permainan akan berakhir (*Game over*).

Permainan ini mempunyai 15 level dimana tiap level akan memiliki tingkat kesulitan yang berbeda. Faktor-faktor penentu tingkat kesulitan antara lain adalah jumlah *room* yang harus dilewati, jumlah *Dangerous Flower*, jumlah *Spider*, dan jumlah *Zombie*. Jumlah minimum & maksimum serta nilai bobot dari faktor-faktor ini bisa dilihat pada Tabel 1. Nilai dari masing-masing faktor penentu tingkat kesulitan tersebut kemudian dimasukkan ke tiap level. Hasil dari penentuan faktor tingkat kesulitan pada semua level dapat dilihat pada Tabel 2.

Tabel 1
Faktor penentu tingkat kesulitan

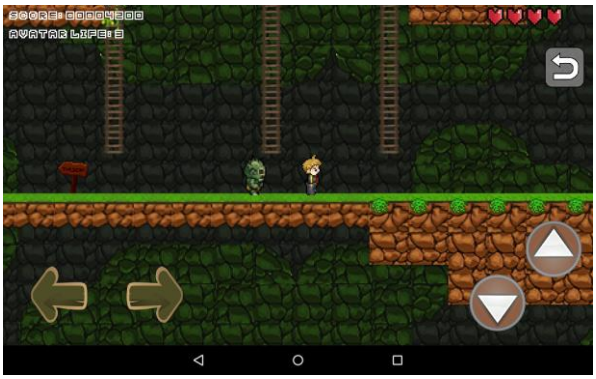
Faktor Penentu	Kode Faktor	Nilai Minimum	Nilai Maksimum	Nilai Bobot
Jumlah *Room*	JR	4	12	0.4
Jumlah *Dangerous Flower*	JF	8	24	0.1
Jumlah *Spider*	JS	0	24	0.2
Jumlah *Zombie*	JZ	0	12	0.3

Tabel 2
Tingkat kesulitan pada tiap level

Level	JR	JF	JS	JZ	Total Bobot
1	4	8	0	0	0
2	4	4	4	0	0.008333333
3	5	6	4	0	0.070833333
4	5	4	6	0	0.075
5	6	6	4	2	0.170833333
6	7	6	4	4	0.270833333
7	7	7	7	4	0.302083333
8	8	16	8	4	0.416666667
9	8	16	8	6	0.466666667
10	9	18	9	6	0.5375
11	9	18	9	9	0.6125
12	10	20	10	10	0.708333333
13	11	22	11	11	0.804166667
14	11	22	22	11	0.895833333
15	12	24	24	12	1

F. Implementasi Permainan

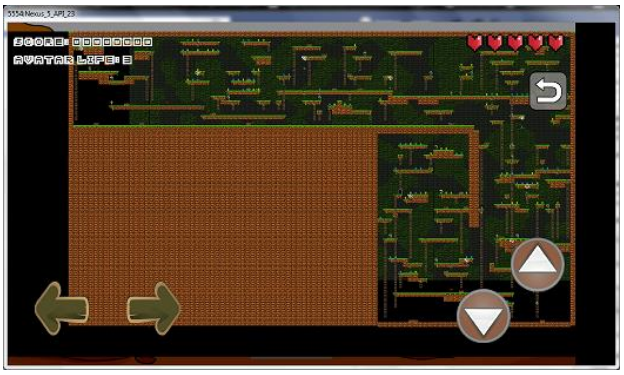
Game Mine Meander dibangun dengan bahasa Java dengan menggunakan *framework* LibGDX. IDE yang digunakan dalam proses pembuatan *game* ini adalah Android Studio yang telah dilengkapi dengan Android SDK agar permainan dapat berjalan pada perangkat Android. Proses pengaturan *asset* gambar dan *world map* dilakukan pada *Map Editor Tiled*. Antarmuka dari area permainan akan menampilkan karakter, tombol kontrol, papan skor, dan informasi mengenai status karakter. Antarmuka tersebut dapat dilihat pada Gambar 6.



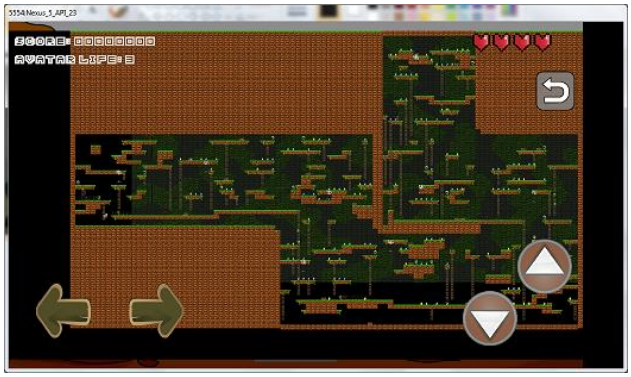
Gambar 6 Antarmuka area permainan

IV. PENGUJIAN

Setelah permainan selesai dibangun, langkah berikutnya adalah melakukan pengujian. Pengujian dilakukan untuk memastikan apakah Algoritma *Recursive Backtracking* mampu membentuk *world map* yang dinamis. Pengujian ini dilakukan sebanyak 2 kali. Pada pengujian pertama, akan dipilih salah satu level pada permainan. Kemudian pada pengujian kedua, level tersebut dipilih lagi dan kemudian dilihat apakah bentuk *world map* saat pengujian kedua berbeda dibandingkan dengan pengujian pertama. Gambar 7 menampilkan bentuk *world map* pada pengujian pertama, sedangkan Gambar 8 menampilkan bentuk *world map* pada pengujian kedua.



Gambar 7 Bentuk *world map* pada pengujian pertama



Gambar 8 Bentuk *world map* pada pengujian kedua

V. KESIMPULAN/RINGKASAN

Dengan menggunakan *framework* LibGDX, kita bisa membuat *game* 2D *Platformer* dan mengimplementasi fitur dynamic world generator di dalamnya. Kemudian dari hasil pengujian, didapat kesimpulan bahwa penggunaan Algoritma *Recursive Backtracking* mampu menghasilkan *world map* yang dinamis. Selain itu bisa dipastikan bahwa setiap level dari permainan ini memungkinkan untuk diselesaikan.

UCAPAN TERIMA KASIH

Penulis F.A.J mengucapkan terima kasih kepada Jurusan Teknik Informatika Institut Teknologi Sepuluh Nopember, Direktorat Pendidikan Tinggi, Departemen Pendidikan dan Kebudayaan Republik Indonesia yang telah memfasilitasi penelitian ini.

DAFTAR PUSTAKA

[1] "What is a Platform Video *Game*?," [Online]. Available: http://compaction*game*s.about.com/od/*game*index/a/*platformer*_def.htm. [Diakses 29 Juni 2016].

[2] "Android Studio," Android Studio, [Online]. Available: https://developer.android.com/studio/index.html. [Diakses 15 Maret 2016].

[3] J. Edmonds, "*Recursive Backtracking*," in *How to Think About Algorithms*, Cambridge, Cambridge University Press, 2008, pp. 251-266.

[4] libGDX, "libGDX," [Online]. Available: https://libgdx.badlogic*game*s.com/. [Diakses 15 Maret 2016].

[5] T. Lindeijer, "Tiled *Map* Editor," [Online]. Available: http://www.*map*editor.org/. [Diakses 20 Maret 2016].

[6] H. M. Pandey, "*Recursive Backtracking*," in *Design and Analysis Algorithm*, New Delhi, University Science Press, 2008, pp. 185-187.

[7] G. Smith, M. Cha and J. Whitehead, "A Framework for Analysis of 2D *Platformer* Levels," in *Sandbox '08: Proceedings of the 2008 ACM SIGGRAPH symposium on video games*, New York, 2008.