Rancang Bangun *Game* Labirin Menggunakan Algoritma *A Star* Berbasis *Mobile*

Nur Budi Nugraha
Program Studi Informatika, Sekolah Tinggi Teknologi Dumai
Jl. Utama Karya Bukit Batrem II
Email: nurbudinugroho87@gmail.com

ABSTRAK

Game adalah sebuah permainan dengan tujuan bersenang-senang dan mengisi waktu luang. Dalam perkembangannya *game* yang dulu dimainkan di halaman rumah bersama teman-teman sekitar lingkungan, dan kini semakin berkurang dengan berkembangnya teknologi dan permainan-permainan tradisional sudah berganti dengan permainan-permainan modern. Salah satu permainan klasik yang hingga saat ini masih sangat digemari adalah labirin. Labirin biasanya berawal dari sebuah jalur yang merupakan jalur buntu, dan hanya satu jalur yang merupakan jalan keluar, namun untuk membuat menarik, labirin dapat dibuat berujung banyak atau memiliki banyak jalan keluar sehingga misi penyelesaian masalah bertambah yang semula hanya satu jalan keluar menjadi banyak, terlebih lagi bila ada beberapa musuh yang menghalangi jalan keluar. Perancangan *game* labirin dengan menggunakan algoritma *A Star* ini dilakukan melalui tahap-tahap berikut: 1) tahap pengumpulan data, 2) tahap analisa dan perancangan, 3) implementasi dan evaluasi *game*. Hasil dari penelitian ini adalah *game* labirin terdiri dari 3 *stage*, dimana pada masing-masing *stage* mempunyai tingkat kesulitan yang berbeda-beda sehingga dapat mengasah kecerdasan pemain dalam menyelesaikan misi setiap *stage*nya.

Kata kunci*: Game, Stage*, Labirin

ABSTRACT

Game is a game with the aim of having fun and free time. In its development, games that were previously played on the home page with friends around the environment, and are now diminishing with the development of technology and games - traditional games have changed with modern games. One classic game that is still very popular today is the labyrinth. The labyrinth usually starts from a lane which is a dead end, and only one lane is a way out, but to make it interesting, the labyrinth can be made to a lot or has a lot of way out so that the problem solving mission increases only one exit becomes a lot, especially again if there are some enemies that block the exit. The design of the labyrinth game using the A Star algorithm is carried out through the following stages: 1) the data collection phase, 2) the analysis and design stage, 3) the implementation and evaluation of the game. The results of this study are maze games

consisting of 3 stages, where each stage has different levels of difficulty so that it can hone the player's intelligence in completing each team's mission.

***Keywords**: Game, Stage, Maze*

Pendahuluan

Dunia bermain sudah tidak lagi harus diluar ruangan. Manusia menciptakan permainan bukan hanya sebagai penghilang rasa bosan, tetapi banyak hal bisa dijadikan alasan untuk mengawali sebuah permaian, termasuk untuk belajar. Belajar dengan permainan merupakan metode yang cukup bagus digunakan karena tidak membuat pelajar menjadi cepat jenuh, sehingga semangat belajar selalu ada (wicaksono, 2016). Perkembangan dunia teknologi memungkinkan penempatan permainan didalam sebuah perangkat android. Karena telepon genggam android bukan hanya berguna sebagai alat komunikasi untuk melakukan panggilan telepon atau pengiriman pesan singkat saja, akan tetapi juga sebagai sarana yang dapat digunakan untuk menghilangkan kejenuhan dengan menanamkan aplikasi permainan di dalamnya (Rahadian, 2016).

Game adalah sebuah permainan dengan tujuan bersenang-senang dan mengisi waktu luang (sudarmilah 2016), Dalam perkembangannya *game* yang dulu dimainkan di halaman rumah bersama teman-teman sekitar lingkungan, dan kini semakin berkurang dengan berkembangnya teknologi dan permainan-pemainan tradisional sudah berganti dengan permainan-permainan modern. Permainan video adalah permainan yang mengunakan interaksi dengan antar muka pengguna melalui gambar yang dihasilkan oleh piranti video, pada umumnya permainan ini mempunyai tujuan atau gol (Putra, 2012). Salah satu permainan klasik yang hingga saat ini masih sangat digemari adalah labirin.

Labirin merupakan salah satu permasalahan yang cukup terkenal dalam sejarah kehidupan manusia, pada zaman dahulu, banyak kerajaan yang menggunakan model labirin untuk strategi pertahanan istana, menyembunyikan tempat rahasia, jalur pelarian dan lain-lain (Pribadi, 2012). Dan labirin pada masa kini lebih sering dinikmati sebagai sebuah permainan pemecahan masalah. Labirin biasanya berawal dari sebuah jalur yang merupakan jalur buntu, dan hanya satu jalur yang merupakan jalan keluar, namun untuk membuat menarik, labirin dapat dibuat berujung banyak atau memiliki banyak jalan keluar sehingga misi penyelesaian masalah bertambah yang semula hanya satu jalan keluar menjadi banyak, terlebih lagi bila ada beberapa musuh yang menghalangi jalan keluar.

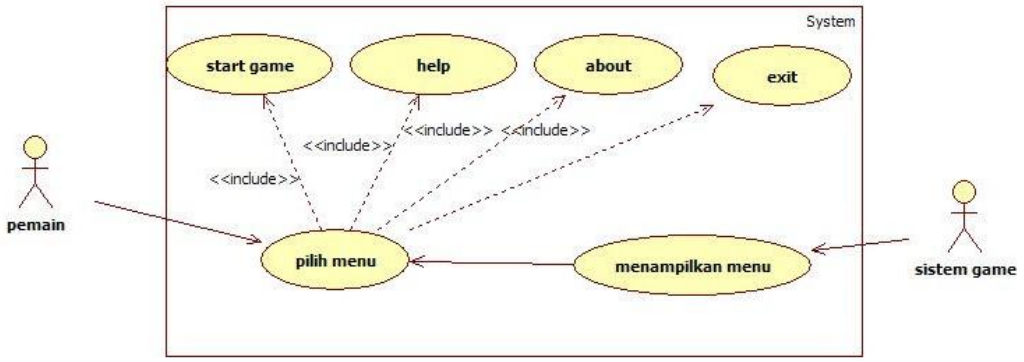
Metode Penelitian

Perancangan *game* labirin dengan menggunakan algoritma A Star ini dilakukan melalui tahap-tahap berikut: 1) tahap pengumpulan data, 2) tahap analisa dan perancangan, 3) implementasi dan evaluasi *game*.

1. Tahap Pengumpulan data
   Pada tahap ini dilakukan pengumpulan data yang akan digunakan sebagai sumber dalam pembuatan *game* labirin. Pengumpulan data ini meliputi studi pustaka, mencari buku acuan atau jurnal yang berhubungan dengan *game* labirin.
2. Tahap analisa dan perancangan
   Tahap kedua merupakan tahap analisa dan perancangan. Dimana pada tahap ini perancangan dibuat melalui bantuan UML yang meliputi *usecase* diagram, *class* diagram, *statechart* diagram serta perancangan *interface* yang akan digunakan dalam *game* ini. *Game* ini terdiri dari 3 *stage*, dimana masing masing stage memiliki tingkat kesulitan yang berbeda-beda.
3. Implementasi dan evaluasi *game*
   Tahap akhir dilakukan implementasi dan evaluasi apakah *game* yang telah dibuat dapat berjalan dengan baik. Hasil perancangan UML tersebut diimplementasikan kedalam program *android* studio dengan menggunakan bahasa java. Setelah selesai diimplementasi, *game* labirin diuji dan evaluasi untuk dilakukan pengembangan atau perbaikan apabila dalam *game* masih ditemukan kesalahan (*error*).

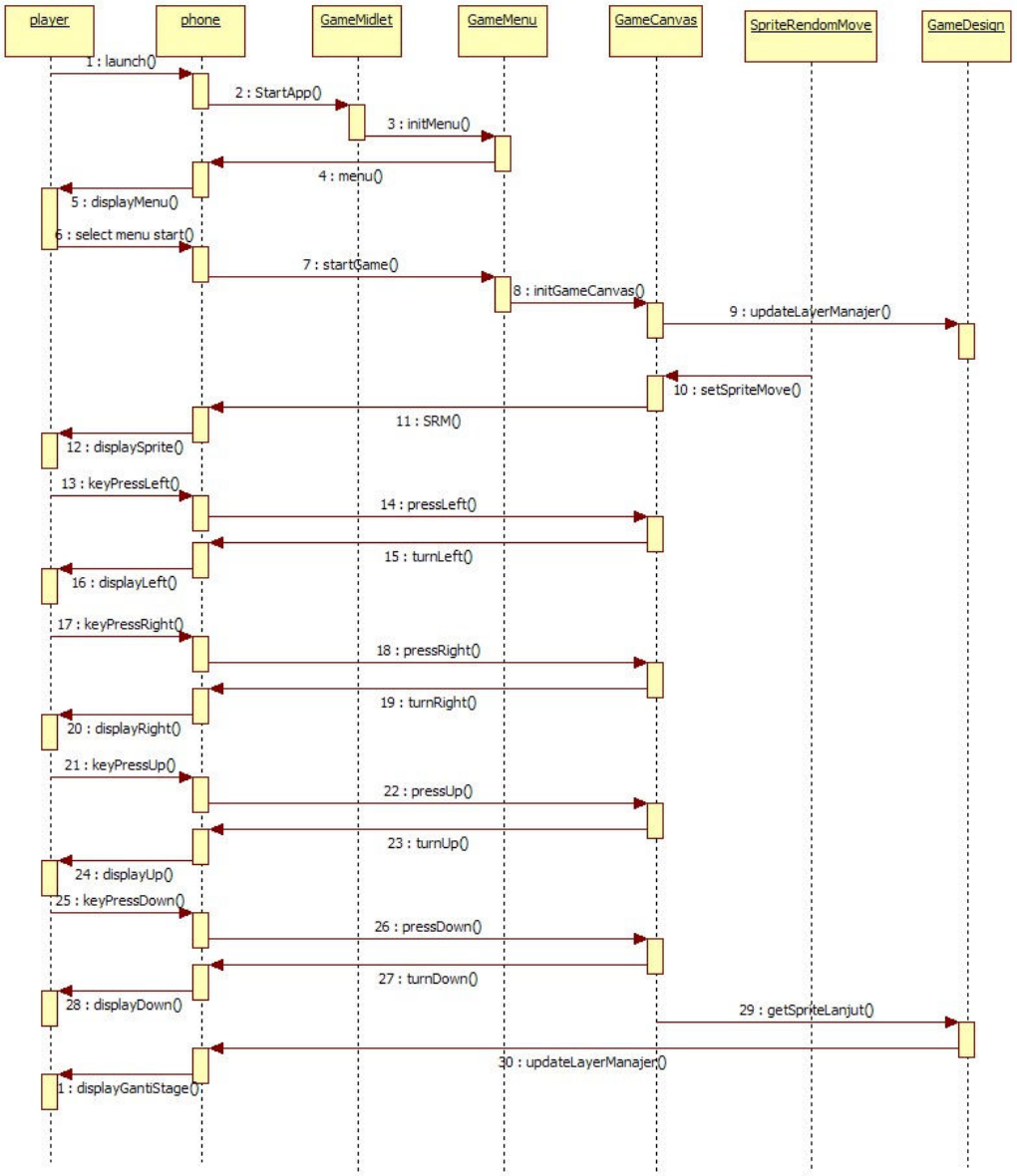
Hasil dan Pembahasan

Rancangan *game* labirin menggunakan *usecase* digaram. *Usecase* digaram berguna untuk mendeskripsikan fungsi fungsi dari sebuah sistem dari sudut pandang pengguna (aktor) dengan sistemnya sendiri. Dimana yang ditekankan dalam *usecase* diagram ini adalah apa yang dilakukan, bukan bagaiamana melakukannya.



Gambar 1. Usecase diagram *game* labirin

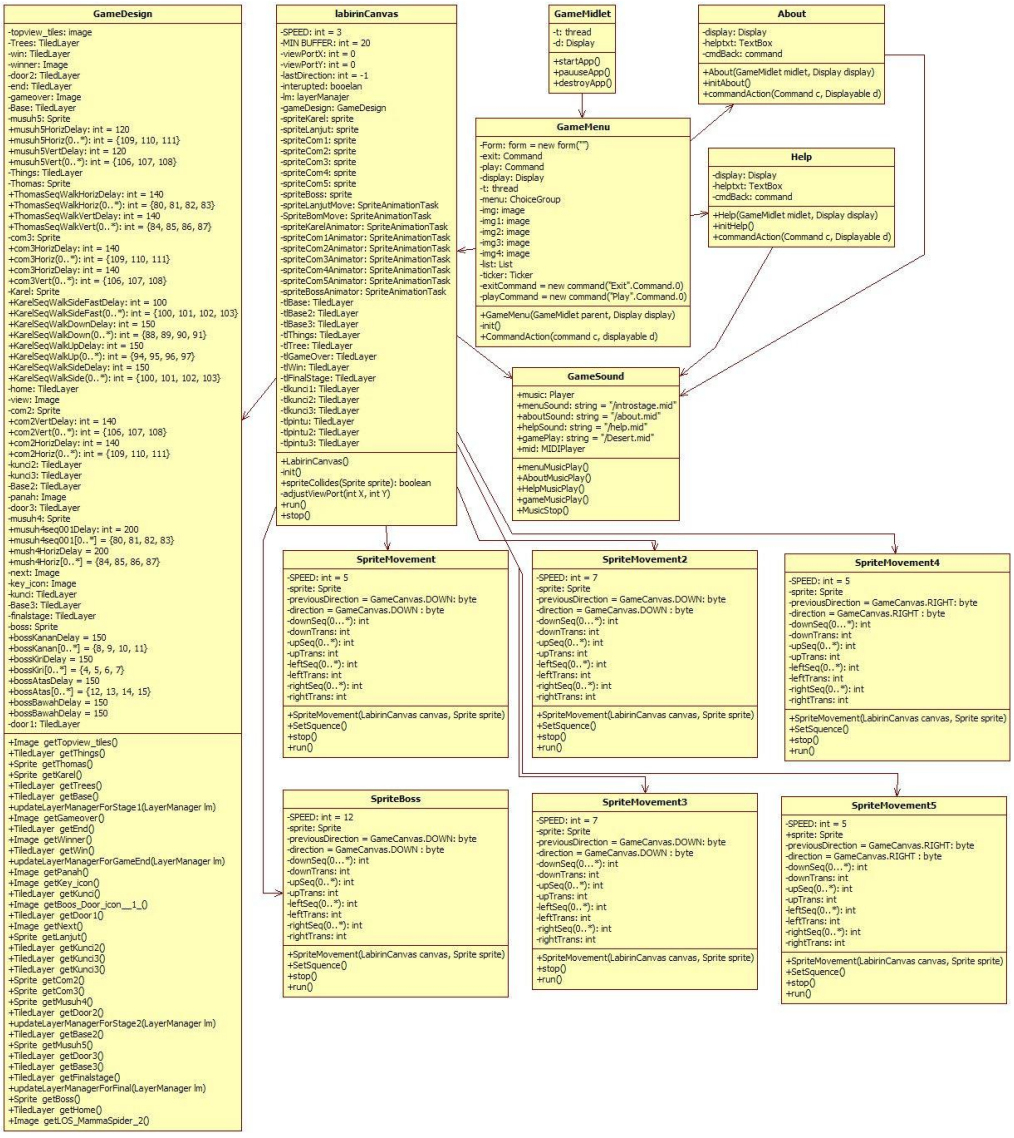
Skenario pada *Usecase* Diagram diatas menjelaskan tentang bagaimana pemain memulai permainan labirin ini. Diawali dari pemain memilih menu yang ditampilkan oleh sistem *game*. Menu awal yang ditampilkan oleh sistem *game* adalah Start, About, Help, dan Exit. Apabila pemain memeilih menu Help maka sistem *game* akan menampilkan penjelasan mengenai spesifikasi *game* labirin ini. Apabila pemain memilih menu About maka sistem *game* akan menampilkan data author *game*. Apabila pemain ingin keluar dari *game* ini maka dapat memilih menu *Exit*. Dan untuk dapat memainkan *game* maka pemain harus memilih menu *Start*. Pada saat pemain memilih menu Start maka sistem akan menampilkan permainan *Stage*1, pada pemainan ini pemain ditugaskan untuk mencari kunci agar dapat membuka pintu, pemain harus membuka kunci secara berurutan agar dapat melanjutkan ke Stage berikutnya. Pemain harus waspada terhadap musuh laba-laba yang berkeliaran, apabila pemain terkena musuh maka permainan akan berakhir.



Gambar 2. Sequence diagram *game* labirin

Sequence Diagram pada Gambar 2 menggambarkan urutan kerja ketika memainkan *game*. Semakin kebawah maka waktu yang dibutuhkan maka semakin lama. Ketika pemain menjalankan *game* maka class yang dipanggil adalah class *GameMidlet*, yang kemudian class *GameMidlet* memanggil *Class Game* Menu yang

akan ditampilkan pada layar Ponsel. Saat pemain memilih menu "Start *Game*" maka akan memanggil *class GameCanvas*. *Class GameCanvas* akan menanggil *class GameDesign* yang menangani semua *image* yang berkaitan dengan *game*. Didalam *class GameCanvas* menangani *event* yang berkaitan dengan pengaturan arah/gerak dari *player*, jika *player* menekan tombol atas (up) maka *class GameCanvas* akan memanggil *syntax* ((*keyState & UP_PRESSED*) !=0) dan akan menampilkan gerak sprite pindah keatas. Jika player menekan tombol bawah (*down*) maka *class GameCanvas* akan memanggil *syntax* ((*keyState & DOWN_PRESSED*) != 0) dan akan menampilkan gerak *sprite* pindah kebawah. Jika *player* menekan tombol kanan (*right*) maka *class GameCanvas* akan memanggil *syntax* ((*keyState & RIGHT_PRESSED*) != 0) dan akan menampilkan gerak *sprite* pindah kekanan. Jika *player* menekan tombol kiri (*left*) maka class GameCanvas akan memanggil *syntax* ((*keyState & LEFT_PRESSED*) != 0) dan akan menampilkan gerak *sprite* pindah kekiri. Jika *player* telah menelesaikan satu *Stage* maka ia dapat lanjut ke *Stage* berikutnya.



Gambar 3. *Class* diagram *game* labirin

Class Diagram pada Gambar 3 menggambarkan hubungan yang terjadi antar *class* yang terkait dengan *Game* Labirin yang dibuat. *Class GameMidlet* merupakan *class* utama didalam sistem *game* labirin ini, karena class GameMidlet ini merupakan *class* yang pertama kali dipanggil pada saat menjalankan aplikasi. Pada saat aplikasi dijalankan pada perangkat *handphone* maka *class GameMidlet* akan

memanggil *method startApp*( ), pada *method startApp*( ) dilakukan pemanggilan *class Game*Menu untuk menampilkan menu *game* ke layar *handphone*, selanjutnya *class Game*Menu akan melakukan pemanggilan *class-class* yang diperlukan dalam proses berjalannya aplikasi. *Class Game*Menu akan melakukan pemanggilan terhadap *class-class* berikut:

1. *Class Labirin Canvas*

   Class ini dijalankan pada saat pemain memilih menu "*Start*". *Class* ini digunakan untuk menjalankan semua *method* yang digunakan untuk *game* labirin ini. *Class* Labirin *canvas* ini juga memanggil *class* lain yaitu:

   a. *Class GameDesignClass*

      GameDesign merupakan *class* yang digunakan untuk men*design* karakter, musuh, serta *stage* didalam *game* labirin ini.

   b. *Class SpriteRandomMovement*

      Class SpriteRandomMovement ini digunakan untuk menggerakkan karakter musuh.

   c. *Class SpriteRandomMovement2*

      Class SpriteRandomMovement2 ini digunakan untuk menggerakkan karakter musuh2.

   d. *Class SpriteRandomMovement3*

      Class SpriteRandomMovement3 ini digunakan untuk menggerakkan karakter musuh3.

   e. *Class SpriteRandomMovement4*

      Class SpriteRandomMovement4 ini digunakan untuk menggerakkan karakter musuh4

   f. *Class SpriteRandomMovement5*

      Class SpriteRandomMovement5 ini digunakan untuk menggerakkan karakter musuh5.

   g. *Class SpriteBoss*

      Class SpriteBoss ini digunakan untuk menggerakkan karakter Boss.

   h. *Class GameSound*

      Class ini digunakan untuk menampilkan *background* musik pada saat *game* dimainkan

2. *Class About*

   Class ini digunakan untuk menampilkan informasi mengenai data pembuat *game*. *Class* ini juga memanggil 1 *class* yaitu: *Class GameSoundClass* ini digunakan untuk menampilkan *background* musik pada saat pemain memilih menu *About*.

3. *Class Help*

Class ini berisi spesifikasi dari *game* labirin ini. Apabila pemain memilih menu *Help* maka sistem akan menampilkan isi dari *class Help*. *Class* ini juga memanggil 1 *class* yaitu *Class GameSound* yaitu *Class* ini digunakan untuk menampilkan *background* musik pada saat pemain memilih menu *Help*.

User Interface
Pada saat aplikasi dijalankan maka *class* yang pertama kali dipanggil adalah *class GameMidlet*. *Class GameMidler* akan memanggil *method*:

```
public void startApp() {
d = Display.getDisplay(this);
menu = new GameMenu(midlet,d);
d.setCurrent(gameCanvas);
}
```

Method tersebut merupakan method untuk menampilkan *Main* Menu.

Gambar 4. Tampilan menu *game*

Pada saat pemain memilih menu "Play *Game*" maka *class Game*Menu akan memanggil *method*:

```
gameCanvas = new LabirinCanvas();
t = new Thread(gameCanvas);
t.start();
display = Display.getDisplay(parent);
display.setCurrent(gameCanvas);
gameCanvas.addCommand(ExitCommand);
gameCanvas.setCommandListener(this);
```

Gambar 5. Tampilan *stage* 1

Untuk menampilkan objek-objek pada *Stage*1 maka pada *class LabirinCanvas* akan menginisialisasi objek-objek tersebut melalui *method*:

```
this.tlkunci = this.gameDesign.getKunci();
this.tlpintu = this.gameDesign.getDoor1();
this.tlkunci2 = this.gameDesign.getKunci2();
this.tlpintu2 = this.gameDesign.getDoor2();
this.tlkunci3 = this.gameDesign.getKunci3();
this.tlpintu3 = this.gameDesign.getDoor3();
this.tlTrees = this.gameDesign.getTrees();
this.tlBase = this.gameDesign.getBase();
this.lm = new LayerManager();
gameDesign.updateLayerManagerForStage1(lm);
```

Apabila *sprite* membentur objek/dinding maka *class LabirinCanvas* akan memanggil *method*:

```
Sprite.collidesWith(
this.tlTrees, true)
|| sprite.collidesWith(this.tlpintu2, true)
|| sprite.collidesWith(this.tlpintu3, true)
|| sprite.collidesWith(this.tlThings, true)
|| sprite.collidesWith(this.tlFinalStage, true)
|| sprite.getX() < 0 || sprite.getY() < 0
|| sprite.getX() > (this.tlBase.getWidth() - sprite.getWidth())
|| sprite.getY() > (this.tlBase.getHeight() - sprite.getHeight()
);
```

Apabila *sprite* Karel berhasil mencapai tujuan (*sprite*Lanjut) maka akan memanggil *method*:

```
this.lm = new LayerManager();
```

gameDesign.updateLayerManagerForStage2(lm);
this.tlBase2 = this.gameDesign.getBase2();

Ketika memainkan *game* labirin maka *state* awal dimulai dari pemanggilan method *StartApp*( ) dari *class GameMidlet* selanjutnya proses inisialisasi menu yang dijalankan pada *class Game*Menu. Apabila dipilih menu *Exit*, maka aplikasi akan ditutup. Kemudian apabila dipilih menu *Start*, *class Game*Menu akan melakukan *action GameThread Stage*1 untuk menampilkan permainan *Stage*1.

Program utama akan melakukan inisialisasi terhadap objek musik untuk menampilkan *background* suara pada permainan, dan inisialisai objek *GameDesign Stage*1 untuk menampilkan gambar-gambar yang diperlukan selama permainan berjalan. Ketika terjadi proses penekanan tombol arah atas pada *handphone*, maka *class* utama (*class LabirinCanvas*) akan melakukan action untuk menginisialisasi perubahan *frame* dari *sprite Karel* untuk bergerak/berjalan keatas/naik. Kemudian apabila terjadi proses penekanan tombol arah bawah pada *handphone*, maka *class* utama (*class LabirinCanvas*) akan melakukan *action* untuk menginisialisasi perubahan *frame* dari *sprite Karel* untuk bergerak/berjalan turun.

Apabila terjadi proses penekanan tombol arah kanan pada *handphone*, maka *class* utama (*class LabirinCanvas*) akan melakukan *action* untuk menginisialisasi perubahan *frame* dari *sprite Karel* untuk bergerak/berjalan kekanan. Apabila terjadi proses penekanan tombol arah kiri pada *handphone*, maka *class* utama (*class LabirinCanvas*) akan melakukan *action* untuk menginisialisasi perubahan *frame* dari sprite Karel untuk bergerak/berjalan kekiri. Apabila sprite Karel menabrak objek maka *class* utama akan memanggil *method spriteCollides* yang akan menginisialisasi jika terjadi benturan, jika *sprite Karel* membentur objek pohon atau batu maka *sprite* akan berhenti, jika *sprite Karel* membentur *sprite* musuh maka akan ditampilkan GameOver, dan apabila *sprite Karel* berhasil mencapai tujuan, maka *class* utama akan menampilkan *Stage* selanjutnya.

Gambar 6. Tampilan *stage* 2

Gambar 7. Tampilan *stage* 3

Apabila *sprite Karel* mengenai musuh maka *class LabirinCanvas* akan memanggil *method*:

this.lm = new LayerManager();
gameDesign.updateLayerManagerForGameEnd(lm);
this.tlGameOver = gameDesign.getEnd();
music.MusicStop();

Kemudian akan menampilkan "Game Over" pada layar *handphone.*

Gambar 8. Tampilan layar kalah

Jika pemain berhasil menyelesaikan *stage* terakhir maka *class GameCanvas* akan memanggil *method*:

this.lm = new LayerManager();
gameDesign.updateLayerManagerForGameWin(lm);
this.tlWin= gameDesign.getWin();
music.MusicStop();

Gambar 9. Tampilan layar menang

Pengujian

Di bawah ini merupakan pengujian aplikasi algoritma A* (*A Star*) untuk menguji perangkat lunak yang sudah dibangun sudah berjalan seperti yang diharapkan atau belum. Pengujian dilakukan berdasarkan algoritma A* (*A Star*) yang ada pada tombol bantuan *stage* 1.

Perhitungan algoritma A* menggunakan *euclidean heuristic* yang digunakan untuk mencari jalur terpendek antara dua buah simpul horizontal dan vertikal. Nilai n.x, n.y = 0 dan goal.x, goal.y = 1 yang diambil dari jarak simpul pada *game*. Nilai *node* (*grid*) yaitu 20x26 *pixel* dimana 1 grid = 1 *pixel* ke arah horizontal atau 1 *pixel* ke arah vertical. Nilai h(n) *heuristic* di dapat dari jarak perkiraan dari simpul awal ke simpul tujuan. Nilai f(n) didapat dari penjumlahan dari g(n) dan h(n).

$$Posisi\ simpul\ awal = n.x: 0, n.y: 0$$
$$Posisi\ simpul\ tujuan = Goal.x: 19, Goal.y: 25$$
$$g(0,1) = 1$$
$$h(n) = \sqrt{(n.x - goal.x)^2 + (n.y - goal.y)^2}$$
$$h(0,1) = \sqrt{(0 - 19)^2 + (1 - 25)^2}$$
$$h(0,1) = \sqrt{(-19)^2 + (-24)^2}$$
$$h(0,1) = \sqrt{361 + 576} = \sqrt{937}$$
$$h(0,1) = 30{,}61$$
$$f(n) = g(n) + h(n)$$
$$f(0,1) = 1 + 30{,}61 = 31{,}61$$

Untuk mencari jalur terpendek dari *stage* 1 membutuhkan 156 langkah dengan simpul yang diperiksa yaitu 186 simpul, Dari hasil pengujian algoritma A*(*A star*) yang dilakukan, dapat disimpulkan bahwa bantuan yang ada pada *game* petualangan labirin sudah berjalan menggunakan *euclidean heuristic*.

Simpulan

Setelah melakukan proses perancangan, pembangunan, dan pengujian aplikasi *game* labirin ini, diperoleh kesimpulan bahwa dengan adanya *game* labirin ini dapat menjadi hiburan dan mengasah kecerdasan dari pemainnya. Karena dalam *game* ini terdapat 3 stage dengan tingkat kesulitan yang berbeda sehingga dapat mengasah strategi dan pola pikir dari pemain ini. Selain itu *game* ini besar kecilnya ukuran dari aplikasi *game* yang dibuat sangat dipengaruhi oleh ukuran file gambar, file suara atau musik, dan banyaknya source code yang digunakan. *Game* yang berjalan lacar pada emulator belum tentu dapat berjalan dengan baik pada

perangkat aslinya. Hal ini disebabkan keterbatasan yang dimiliki oleh handphone yang bersangkutan, baik dari segi memori dan spesifikasi sistem java dalam handphone tersebut.

Daftar Pustaka

Wicaksono, R. M., & R Sandhika Galih Amalga, D. S., 2016. Pembuatan *Game* Petualangan Menggunaka Construct2 (Doctoral dissertation, Fakultas Teknik Unpas).

Rahadian, M., Agustri, S., & Suhandi, N. 2016. Pembangunan *Game* Ayo Tarik Berbasis Android. Jurnal Informatika Global.

Sudarmilah, E., & Wibowo, P.A., 2016. Aplikasi Augmented Reality *Game* Edukasi untuk Pengenalan Organ Tubuh Manusia Khazanah Informatika: Jurnal Ilmu Komputer dan Informatika, 2(1), 20-25.

Putra, R. D., Aswin, M., & Djuriatno, W. 2012. Pencarian Rute Terdekat Pada Labirin Menggunakan Merode A*. Jurnal EECCIS, 6(2).

Pribadi, B. 2013. Pembangunan *Game* Edukasi Smart Labyrinth Berbasis Dekstop (Doctoral dissertation, Universitas Komputer Indonesia).

Jogiyanto HM, 2005. Analisis dan Disain Sistem Informasi, Yogyakarta: Andi Offset.

Lewis Moronta, 2003. *Game* Development with Action Script, Course Technology PTR.

M.Suyanto, 2007. Analisis & Desain Aplikasi Multimedia Untuk Pemasaran, Yogyakarta: Andi Offset.

Wahana Komputer, 2008. Menguasai Adobe Photoshop CS3, Yogyakarta: Andi Offset.