PENGGUNAAN ALGORITMA A* PADA GAME GODEF BERBASIS ANDROID

Alvin Oktavian Yondra[1], Tony Darmanto[2], Alfred Yulius A.P.[3]
[123]Teknik Informatika, Universitas Widya Dharma Pontianak
e-mail: [1]oktavianalvin999@gmail.com, [2]tony.darmanto@yahoo.com, [3]alfredyulius7032020@gmail.com

Abstract

*Tower Defense (TD) game is a popular strategy genre game because it can train the brain to prevent incoming enemies from destroying the user. The enemy that comes will go through a path that has been made by the user. To determine the path that the enemy will follow, an algorithm called the A * algorithm is needed. Algorithm A * is an algorithm to find the route to be followed. This algorithm is widely used in computer programs to generate a route to be followed. The advantage of using this algorithm is that it is easy to implement. Therefore, this study is to find out how to apply the A * algorithm to the TD game with the Android platform, which is designed so that the resulting value can be used to get the route the enemy will take.*

Keywords: *Tower Defense, A* Algorithm Android.*

Abstrak

Game Tower Defense (TD) adalah salah satu jenis *game* bergenre strategi yang digemari karena dapat melatih otak untuk berpikir bagaimana mencegah musuh yang datang untuk tidak menghancurkan markas *user*. Musuh yang datang itu akan melalui suatu jalur yang telah dibuat oleh *user*. Untuk menentukan jalur yang akan dilalui oleh musuh itu diperlukan suatu algoritma yang dinamakan algoritma A*. Algoritma A* merupakan salah satu algoritma untuk pencarian rute yang akan dilalui. Algoritma ini banyak digunakan dalam *program* komputer untuk menghasilkan rute yang akan dilalui. Keuntungan dari menggunakan algoritma ini adalah mudah diimplementasikan. Oleh karena itu, penelitian ini bermaksud mencari tahu bagaimana cara menerapkan algoritma A* pada game TD ber-*platform* Android yang dirancang sehingga nilai yang dihasilkan dapat digunakan untuk mendapatkan rute yang akan dilalui oleh musuh.

Kata Kunci : Tower Defense, Algoritma A*, Android.

1. PENDAHULUAN

Game telah menjadi satu hal yang ada di dalam keseharian masyarakat. Dahulu, *game* hanya dijadikan sarana hiburan semata namun sekarang *game* telah menjadi luas fungsinya, misalnya *game* dapat dijadikan sarana pembelajaran, lahan bisnis, dan dipertandingkan sebagai salah satu dari cabang olahraga oleh para profesional. Perkembangan *game platform* juga dapat dilihat secara langsung oleh masyarakat, pada mulanya *game* hanya dimainkan di komputer dan *console* tetapi sekarang sudah memasuki era *mobile game*.

Mobile game adalah sebuah game yang didesain dan dimainkan menggunakan *mobile devices,* seperti *smartphone, tablet PCs, portable media player,* dan *PDA.* Pada zaman sekarang ini, *mobile game* telah dibuat di berbagai macam *platform* seperti Symbian, Apple IOS, Android serta Windows Phone. Keuntungan tersendiri memainkan *mobile game* adalah portabilitas, yaitu *player* dapat bermain *game* dimana saja selama *player* mempunyai *mobile devices* yang mampu menjalankan *mobile games*.

Perkembangan *game* saat ini sangat pesat. *Game* yang awalnya hanya bersifat dua dimensi (2D) dan berupa tampilan yang sederhana saja. Kemudian berkembang hingga menjadi *game* tiga dimensi (3D) yang tampilannya sudah bagus. Hal ini membuat para *programmer* tertarik untuk membuat dan mengembangkan sebuah *game* yang semakin bagus untuk menjadi yang terbaik di antara *programmer* lainnya.

Dengan munculnya sistem operasi berbasis Android, *programmer* berlomba-lomba untuk membuat *game smartphone* yang menarik dan dapat menghibur *user.* Salah satu contoh *game* yang dapat dikatakan menarik dan dapat menghibur *user* adalah *game Tower Defense (*TD). *Game Tower Defense* (TD) adalah salah satu jenis *game* bergenre strategi yang digemari karena dapat melatih otak untuk berpikir bagaimana mencegah musuh yang datang untuk tidak menghancurkan markas *user*. Musuh yang datang itu akan melalui suatu jalur yang telah dibuat oleh *user*. Untuk menentukan jalur yang akan dilalui oleh musuh itu diperlukan suatu algoritma yang dinamakan algoritma A*.

Algoritma A* merupakan salah satu algoritma untuk pencarian rute yang akan dilalui. Algoritma ini banyak digunakan dalam *program* komputer untuk menghasilkan rute yang akan dilalui. Keuntungan dari menggunakan algoritma ini adalah mudah diimplementasikan.

Dengan demikian, penulis bermaksud untuk membuat sebuah *Game* 3D "GoDef" dalam *platform* Android dengan konsep yang menarik dan dapat menghibur. Dalam penerapannya penulis menggunakan algoritma A* untuk menghasilkan rute yang digunakan untuk mengetahui rute yang akan dilalui oleh musuh.

2. METODE PENELITIAN

2.1 Metode Penelitian

Metode penelitian yang digunakan penulis dalam menyusun skripsi ini yaitu: Rancangan Penelitian, Metode Pengumpulan Data, Teknik Analisis Aplikasi, dan Teknik Perancangan Aplikasi.

2.1.1 Rancangan Penelitian

Rancangan penelitian yang digunakan penulis dalam penulisan skripsi ini adalah penelitian deskriptif dan perancangan eksperimen. Desain penelitian ini akan memberikan gambaran mengenai bagaimana penerapan algoritma A* pada *platform* Android. Eksperimen dilakukan dengan mengimplementasikan aplikasi *game* ke dalam *smartphone* dengan *platform* Android.

2.1.2 Metode Pengumpulan Data

Metode pengumpulan data yang dilakukan adalah dengan cara menelusuri literatur dan artikel yang dapat digunakan sebagai bahan yang dapat digunakan untuk menyusun penelitian.

2.1.3 Teknik Analisis Aplikasi

Teknik analisis sistem yang digunakan oleh penulis dalam penelitian ini adalah *Unified Modeling Language* (UML) sebagai teknik analisis sistem yang digunakan untuk menggambarkan, dan mendokumentasikan perancangan aplikasi *game.*

2.1.4 Teknik Perancangan Aplikasi

Perancangan sistem menggunakan *game engine Unity 2020.2.3f1.,* bahasa pemrograman *Microsoft Visual Studio 2019* untuk melakukan *code program* dan menerapkan algoritma A*.

2.2 Landasan Teori

2.2.1 Game

Game adalah jenis kegiatan bermain, dilakukan dalam konteks pura-pura nyata, dimana peserta mencoba untuk mencapai setidaknya satu wewenang, tujuan yang tidak sepele dengan berindak sesuai dengan aturan[1]. *Game* berasal dari kata bahasa inggris yang memiliki arti dasar permainan. Permainan dalam hal ini merujuk pada pengertian kelincahan intelektual (*intellectual playability*). Game juga bisa diartikan sebagai arena keputusan dan aksi pemainnya. Ada target-target dan misi untuk dapat dicapai pemainnya[2].

2.2.2 Algoritma A*

Algoritma A* adalah algoritma *Best First Search* yang merupakan perpaduan antara *Uniform Cost Search* dan *Greedy-Best First Search. Uniform Cost Search* ini akan memilih jarak paling kecil dari simpul awal ke simpul berikutnya sampai ke simpul tujuan, sedangkan *Greedy-Best First Search* yang menggunakan fungsi heuristik akan memperkirakan biaya dari simpul awal ke simpul tujuan. Rumus yang digunakan pada algoritma A* adalah:

$$F(n) = G(n) + H(n)$$

Dimana:
F(n) = biaya yang dibutuhkan
G(n) = biaya yang ditempuh dari node asal
H(n) = nilai perkiraan dari node saat ini ke tujuan[3].

A*(dibaca "A star") adalah algoritma komputer yang digunakan secara luas dalam mencari jalur (*pathfinding)* dan grafik melintang (*graph traversal)*, proses *plotting* sebuah jalur melintang secara efisien antara titik-titik, disebut *node.* Pseudocode dari algoritma A* adalah:

Function A* (masalah) returns solusi
    OPEN <- S
    CLOSED <- array kosong
Loop sampai ditemukan atau sampai tidak ada
    If OPEN = kosong then
        Gagal
Else
    BestNode simpul yang ada di OPEN
    Dengan f minimal
    Pindahkan simpul terbaik tersebut dari
    OPEN ke CLOSED
    If BestNode goal then
        Sukses
Else
    Bangkitkan semua suksesor BestNode
    tapi jangan buat pointer
    Untuk setiap suksesor kerjakan:
    Hitung g(suksesor) g(BestNode) +

actual cost(dari BestNode ke suksesor)
{Periksa suksesor}
If suksesor ada di OPEN then (sudah pernah dibangkitkan tapi belum diproses)
OLD simpul di OPEN yang sama
dengan suksesor tersebut
Tambahkan OLD sebagai suksesor BestNode
Buat pointer dari OLD ke BestNode
Bandingkan nilai g(OLD) dengan g(suksesor)
If g(OLD) lebih baik then
Ubah parent OLD ke BestNode
Ubah nilai g dan f yang ada pada OLD
End
Else
If suksesor ada di CLOSED then (sudah pernah dibangkitkan dan sudah diproses)
OLD simpul di CLOSED yang sama dengan suksesor tersebut
Tambahkan OLD sebagai suksesor BestNode
Bandingkan nilai g(OLD) dengan g(suksesor)
If g(OLD) lebih baik then
Ubah parent OLD ke BestNode
Ubah nilai g dan f yang ada pada OLD
Propagasi untuk semua suksesor OLD dengan penelusuran DFS dengan aturan:
Loop sampai simpul suksesor tidak ada di OPEN atau simpul tidak punya suksesor
If suksesor ada di OPEN then
Propagasi diteruskan
Else
If nilai g via suksesor lebih baik then
Propagasi diteruskan
Else
Propagasi dihentikan
End
End
End

Else {suksesor tidak ada di OPEN maupun CLOSED}
Masukkan suksesor ke OPEN
Tambahkan suksesor tersebut sebagai suksesornya BestNode
Hitung f:g(suksesor) + h(suksesor)
End
End
End
End[5].

2.2.3 Android

Android merupakan sebuah sistem operasi perangkat *mobile* berbasis Linux yang mencakup sistem operasi, *middleware*, dan aplikasi. [5]. Android adalah sistem operasi berbasis Linux yang dimodifikasi untuk perangkat bergerak (mobile devices) yang terdiri dari sistem operasi, *middleware,* dan aplikasi-aplikasi utama[6].

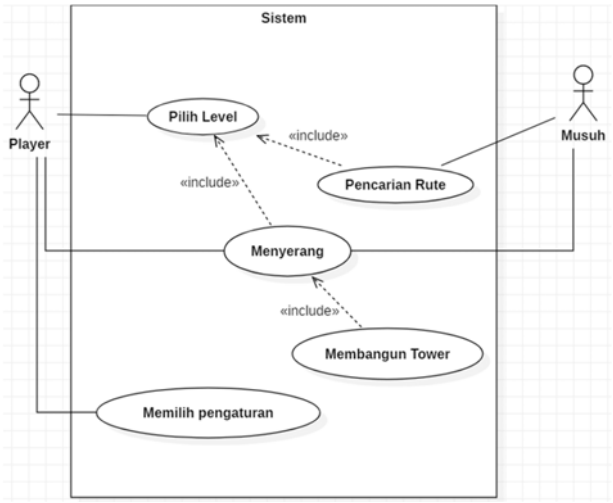
Tabel 1. Daftar Android

Versi	Tanggal rilis	Kode
1.1	9 Februari 2009	-
1.5	30 April 2009	Cupcake
1.6	15 September 2009	Donut
2.0/2.1	26 Oktober 2009	Éclair
2.2	20 Mei 2010	Frozen Yoghurt(Froyo)
2.3	6 Desember 2010	Gingerbread
3.0	22 Februari 2011	Honeycomb
4.0	19 Oktober 2011	Ice Cream Sandwich
4.1	27 Juni 2012	Jelly Bean
4.2	29 Oktober 2012	Jelly Bean
4.3	24 Juli 2013	Jelly Bean
4.4	3 September 2013	KitKat

3. HASIL DAN PEMBAHASAN

3.1 Perancangan Unified Modeling Language (UML)

3.1.1 Diagram Use Case Aplikasi Game



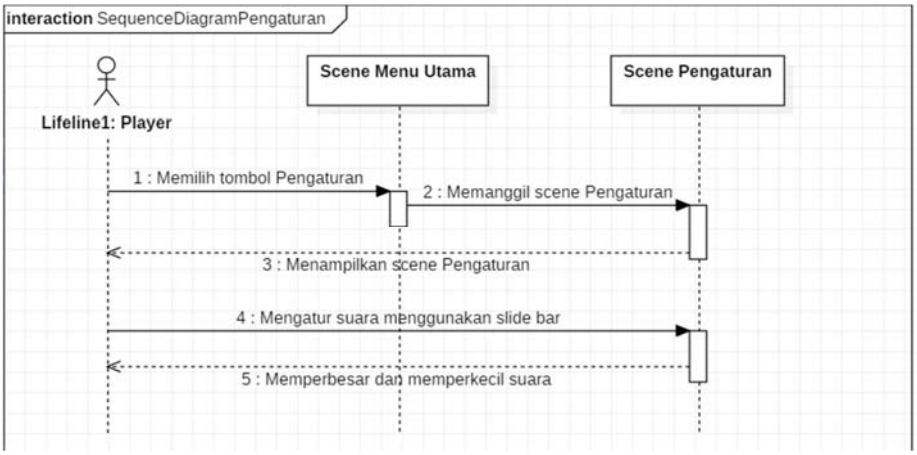
Gambar 1. Use Case Aplikasi Game

Pada diagram *Use Case Aplikasi Game* akan dijelaskan proses-proses yang berjalan di dalam aplikasi *game*. *Actor* yang digunakan adalah *player* dan musuh. Berdasarkan *use case* yang ada pada Gambar 1 diketahui aplikasi *game* ini memiliki tiga tombol utama yaitu Mulai, Pengaturan, dan Keluar. Untuk memainkan *game, player* dapat menekan tombol Mulai. Setelah tombol Mulai dipilih, *player* harus memilih level berapa yang ingin dimainkan terlebih dahulu. Setelah level selesai dipilih, *player* dapat menyerang musuh dengan membangun *tower*. Selain itu, pencarian rute yang dilakukan musuh akan menggunakan algoritma A* oleh sistem. Jika *player* ingin memperbesar ataupun memperkecil suara *background* permainan, dapat menekan tombol Pengaturan dan sistem akan mengarahkan *player* menuju *scene* Pengaturan yang berisi *slide bar* untuk pengaturan suara. Tombol Keluar merupakan salah satu tombol pada menu utama yang dapat dipilih untuk keluar dari permainan.

3.1.2 Diagram Sekuensial (Sequence Diagram)

Diagram sekuensial adalah diagram yang digunakan untuk menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Diagram sekuensial untuk perancangan *game* ini terdiri atas tujuh diagram sebagai berikut:

3.1.2.1 Sequence Diagram Memilih Pengaturan



Gambar 2. Sequence Diagram Pengaturan

Diagram sekuensial yang ada pada Gambar 2, dapat dijelaskan sebagai berikut:

a. Ketika *player* memilih tombol Pengaturan pada menu utama, sistem akan memanggil *scene* Pengaturan dan menampilkannya pada *player*.
b. Setelah *scene* Pengaturan muncul, *player* dapat mengatur suara *background* dari permainan menggunakan *slide bar* yang ada dalam *scene* Pengaturan tersebut.

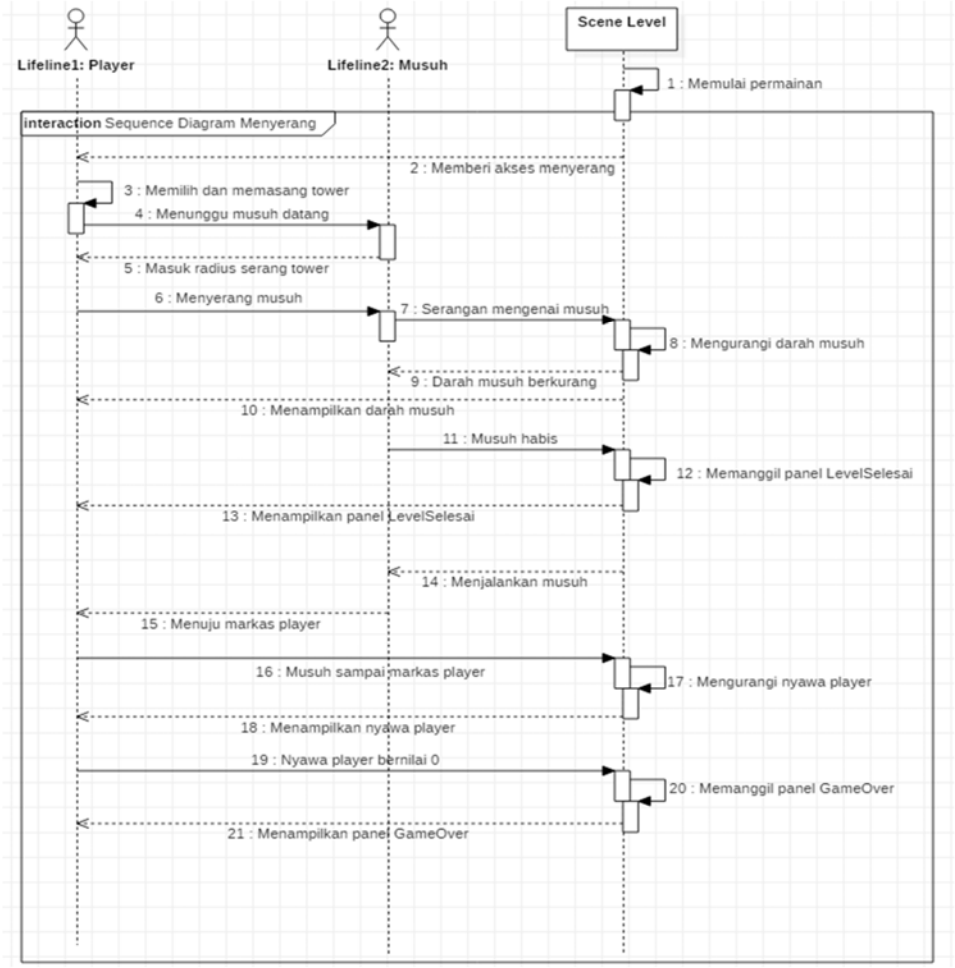
c. Jika *slide bar* digeser ke kiri maka suara permainan akan dikecilkan, sedangkan jika digeser ke kanan maka suara permainan akan dibesarkan.

3.1.2.2 Sequence Diagram Menyerang

Berdasarkan diagram sekuensial yang ada pada Gambar 3, setelah permainan dimulai, terdapat duah buah *sequence* yang dilakukan disaat yang bersamaan dan dapat dituliskan sebagai berikut:

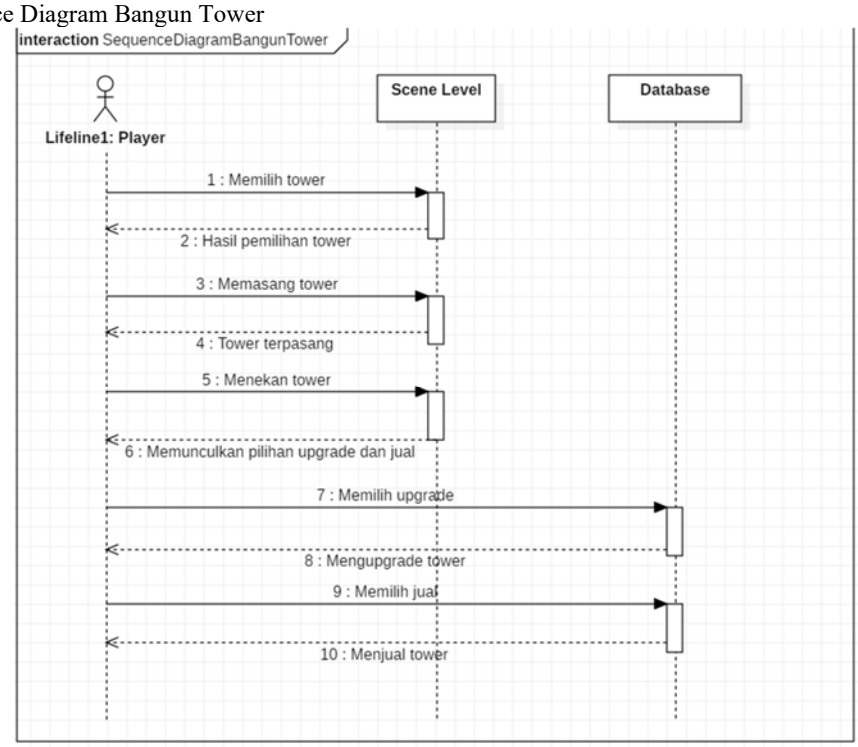
a. Diagram Sekuensial Menyerang oleh Player
   1) Permainan dimulai oleh sistem dan sistem akan memberikan akses penyerangan agar *player* dapat menyerang musuh.
   2) *Player* memiliki misi untuk mengalahkan musuh untuk memenangkan permainan, oleh karena itu *player* harus memilih dan memasang *tower* untuk menyerang musuh.
   3) Setelah musuh memasuki radius tembakan dari *tower* dan serangan mengenai musuh, maka sistem akan mengurangi darah musuh dan menampilkan darah yang dimiliki musuh kepada *player*.
   4) Apabila semua musuh sudah habis, sistem akan secara otomatis memunculkan panel LevelSelseai dan menampilkannya pada *player*.

b. Diagram Sekuensial Musuh Datang
   1) Setelah permainan dimulai, sistem kemudian akan mengaktifkan musuh agar musuh dapat muncul dari *spawner*.
   2) Musuh yang datang akan menelusuri jalan menuju markas *player*.
   3) Apabila musuh sampai ke markas *player*, maka sistem akan mengurangi nyawa dari *player* dan menampilkan nyawa yang telah berkurang tersebut kepada *player*.
   4) Ketika nyawa yang dimiliki oleh *player* bernilai 0, maka sistem akan secara otomatis memunculkan panel GameOver dan menampilkannya pada *player*.



Gambar 3. Sequence Diagram Menyerang

3.1.2.3 Sequence Diagram Bangun Tower



Gambar 4. Sequence Diagram Bangun Tower

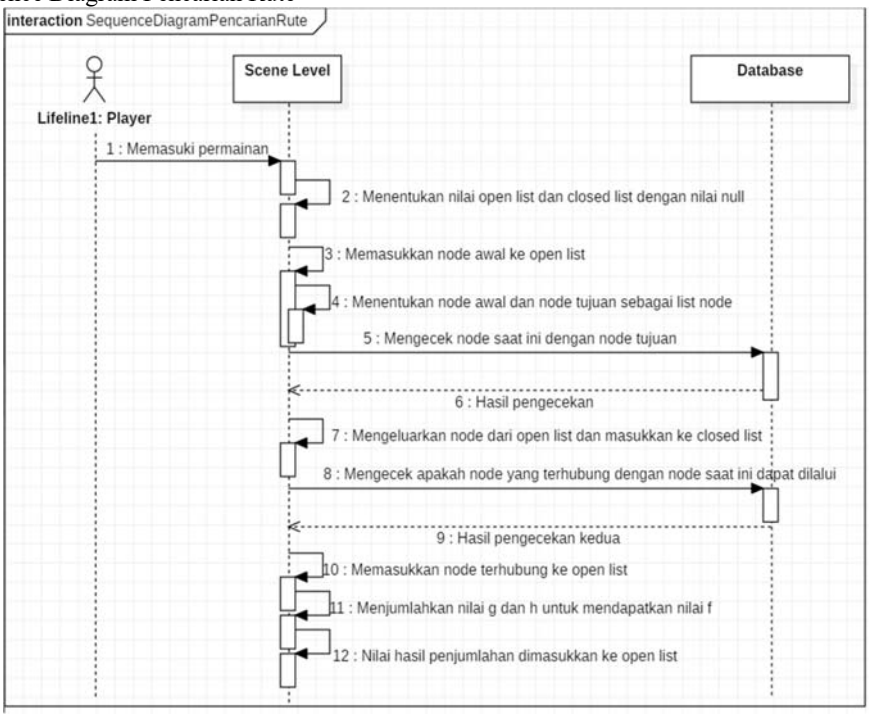
Diagram sekuensial yang ada pada Gambar 4, dapat dijelaskan sebagai berikut:

a. *Player* akan memilih *tower* dan memasang *tower* terlebih dahulu, kemudian sistem akan menampilkannya pada *player*.

b. Apabila *player* menekan *tower* tersebut, sistem akan menampilkan pilihan *upgrade* dan jual kepada *player*.

c. Ketika *player* memilih pilihan *upgrade*, maka sistem akan meng-*upgrade tower* tersebut. Sedangkan ketika *player* memilih pilihan jual, maka sistem akan menjual *tower* tersebut.

3.1.2.4 Sequence Diagram Pencarian Rute



Gambar 5. Sequence Diagram Pencarian Rute

Diagram sekuensial yang ada pada Gambar 5, dapat dijelaskan sebagai berikut:

a. Pertama-tama, saat *player* sudah memasuki permainan, sistem akan menentukan nilai dari Open List dan Closed List menjadi *null* terlebih dahulu.
b. Setelah itu, node awal akan dimasukkan ke dalam Open List dan menentukan node awal dan node tujuan sebagai List Node.
c. Kemudian sistem akan mengecek node saat ini atau node awal apakah sama dengan node tujuan. Hasil pengecekan tersebut berfungsi untuk menentukan apakah musuh sudah mencapai tujuan atau belum.
d. Jika sudah maka musuh sudah tidak bergerak, sedangkan jika belum maka dilanjutkan proses pengeluaran node dari Open List dan dimasukkan ke dalam Closed List.
e. Tahap selanjutnya, sistem akan kembali mengecek apakah node yang terhubung dengan node saat ini dapat dilalui atau tidak. Hasil pengecekan kedua ini digunakan untuk mencari rute terbaik yang dapat dilalui
f. Setelah didapatkan hasil pengecekan tersebut, node yang terhubung itu akan dimasukkan ke dalam Open List dan melakukan penjumlahan nilai g dan h untuk mendapatkan nilai f terkecil.
g. Nilai terkecil tersebut akan dimasukkan ke Open List dan ulangi proses pengecekan pertama kembali sampai mendapatkan jalur yang terbaik.

3.2 Implementasi Algoritma Pada Permainan

3.2.1 Gambaran Algoritma Pada Permainan

Dalam perancangan permainan 3D GoDef ini penulis menggunakan algoritma A* sebagai dasar dalam membuat permainan ini untuk membuat permainan ini menjadi menarik dan dapat dimainkan pengguna. Algoritma A* ini merupakan salah satu algoritma yang digunakan untuk pencarian rute atau jalur. Penulis menggunakan Algoritma A* ini karena algoritma ini mudah diimplementasikan. Algoritma A* digunakan dalam penelitian ini sebagai cara untuk menentukan rute atau jalur yang dilalui musuh. Algoritma ini memiliki rumus sebagai berikut:

$F(n) = G(n) + H(n)$

Dimana :

F(n) = biaya yang dibutuhkan

G(n) = biaya yang ditempuh dari node asal

H(n) = nilai perkiraan dari node saat ini ke tujuan

Penulis menggunakan algoritma A* ini berdasarkan syarat-syarat di atas sehingga menghasilkan nilai-nilai node yang terdekat menuju target. Berikut merupakan contoh dari algoritma A* untuk menghasilkan nilai-nilai node tersebut:

Nilai n.x dan n.y = 0, nilai goal.x = 0 dan goal.y = 1 yang diambil dari jarak simpul pada *game*

Posisi simpul awal = n.x : 0, n.y : 0 dan Posisi simpul tujuan = goal.x : 19, goal.y : 25

Nilai g(0,1) = 1

Penyelesaian :

$h(n) = \sqrt{(n.x - goal.x)^2 + (n.y - goal.y)^2}$

$h(0,1) = \sqrt{(0 - 19)^2 + (1 - 25)^2}$

$h(0,1) = \sqrt{(-19)^2 + (-24)^2}$

$h(0,1) = \sqrt{(-19)^2 + (-24)^2}$

$h(0,1) = \sqrt{361 + 576} = \sqrt{937}$

h(0,1) = 30,61

f(n) = g(n) + h(n)

f(0,1) = 1 + 30,61 = 31,61

Setelah didapatkan biaya yang dibutuhkan pada node awal tersebut, akan dilanjutkan perhitungan pencarian biaya pada node-node selanjutnya sampai mendapatkan rute terpendek yang dapat dilalui oleh musuh untuk sampai ke markas dari *player*.

3.2.2 Hasil Implementasi Algoritma A* Pada Game 3D GoDef Berbasis Android

Penerapan algoritma A* pada *game* ini terdapat pada *scene* Level. Saat permainan dimulai, sistem akan menjalankan proses pencarian rute dengan algoritma A* yang diterapkan pada musuh agar musuh dapat berjalan menuju markas *player*. Dalam *game* 3D GoDef ini, *programmer* menggambarkan rute yang didapat dari algoritma A* berupa lintasan yang menggunakan tekstur tanah agar terlihat seperti jalanan asli.

3.3 Tampilan Game GoDef

Dalam *game* 3D GoDef ini terdiri atas beberapa panel dan *scene* yang dapat diuraikan sebagai berikut:

3.3.1 Tampilan Scene Menu Utama

Tampilan *scene* Menu Utama dapat dilihat pada Gambar 6. Dalam menu utama ini terdiri atas tiga tombol yang dapat dipilih oleh *player*. Tombol-tombol tersebut adalah tombol Mulai, tombol Pengaturan, dan tombol Keluar. Ketika *player* ingin memainkan permainan dapat memilih tombol Mulai. Setelah tombol Mulai dipilih, *player* akan diarahkan menuju *scene* Pilih Level sebelum memasuki permainan dan memilih level yang ingin dimainkan terlebih dahulu. Jika merasa terganggu dengan suara *default* permainan yang keras, *player* dapat memilih tombol Pengaturan yang ada pada *scene* Menu Utama untuk mengatur suara permainan. *Scene* Pengaturan akan muncul ketika tombol Pengaturan dipilih. Untuk keluar dari aplikasi, *player* dapat memilih tombol Keluar.

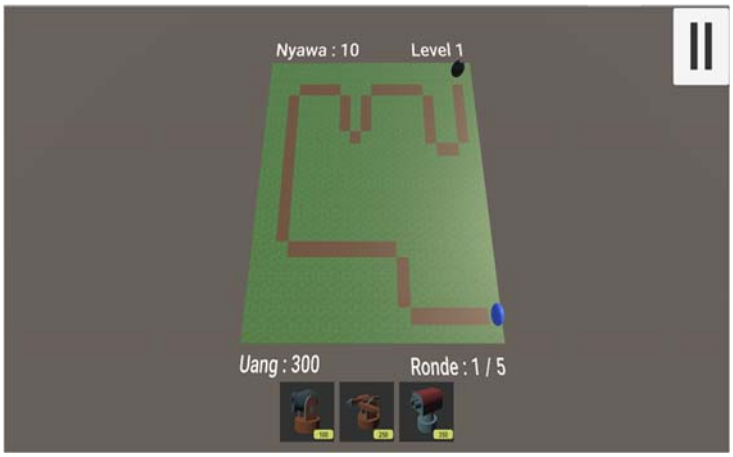
Gambar 6. Tampilan Scene Menu Utama

3.3.2 Tampilan Scene Pilih Level

Tampilan *scene* Pilih Level dapat dilihat pada Gambar 7. Dalam *scene* ini terdiri atas sepuluh tombol level yang dapat dipilih oleh *player* dan satu tombol kembali. Setiap tombol itu akan mengarahkan *player* ke *scene* Level sesuai dengan pilihan *player.* Tombol Kembali berfungsi untuk kembali ke *scene* Menu Utama.

Gambar 7. Tampilan Scene Pilih Level

3.3.3 Tampilan Scene Level

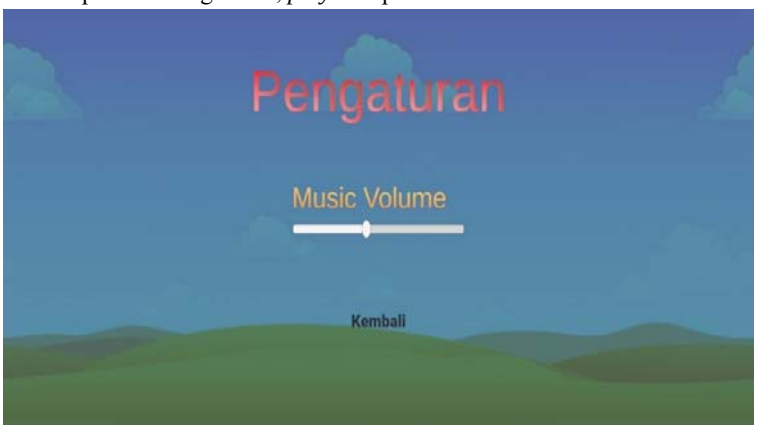


Gambar 8. Tampilan Scene Level

Tampilan *scene* Level ditunjukkan pada Gambar 8. Pada *scene* Level ini berisikan suatu *Map* yang merupakan tempat dimana *player* dapat membangun tower dan meletakkannya di dalam *Map* itu untuk mengalahkan musuh yang datang. Musuh yang datang sudah ditentukan oleh waktu dan akan terdapat beberapa ronde yang harus diselesaikan. Terdapat beberapa pilihan *tower* yang dapat digunakan oleh *player* untuk mengalahkan semua musuh yang datang.

3.3.4 Tampilan Scene Pengaturan

Tampilan *scene* Pengaturan dapat dilihat pada Gambar 9. Pada *scene* ini, *player* dapat mengatur suara permainan dengan menggeser *slider* Music Volume yang ada pada *scene* tersebut. Suara dari permainan akan semakin keras jika *slider* digeser ke kanan dan sebaliknya jika digeser ke kiri akan membuat suara permainan semakin kecil. Untuk menutup *scene* Pengaturan, *player* dapat memilih tombol Kembali.



Gambar 9. Tampilan Scene

3.3.5 Tampilan Panel LevelSelesai

Tampilan panel LevelSelesai dapat dilihat pada Gambar 10. Panel ini ditampilkan ketika *player* mengalami kemenangan. Kondisi *player* mengalami kemenangan yaitu pada saat semua ronde sudah diselesaikan dan musuh tidak muncul lagi.

Gambar 10. Tampilan Panel LevelSelesai

3.3.6 Tampilan Panel GameOver

Tampilan panel GameOver dapat dilihat pada Gambar 11. Panel ini hanya ditampilkan ketika *player* mengalami kekalahan. Kondisi *player* mengalami kekalahan yaitu pada saat nyawa yang dimiliki oleh *player* sudah bernilai nol.

Gambar 11. Tampilan Panel GameOver

4. KESIMPULAN

Berdasaran hasil implementasi pada perancangan *game* 3D GoDef yang memiliki basis Android dan pembahasan sebelumnya, maka penulis dapat menyimpulkan bahwa:

a. Aplikasi *game* 3D GoDef yang dirancang adalah sebuah permainan yang ditujukan untuk pengguna *smartphone* Android.
b. Pencarian rute dengan algoritma A* memiliki peran dalam penentuan rute terdekat yang dapat dilalui oleh musuh namun penerapan algoritma A* belum terlihat jelas pada aplikasi yang dirancang.
c. *Player* dapat memanfaatkan komponen-komponen yang ada pada *interface scene* Level seperti tombol *Upgrade* jika menekan *tower* yang telah dibangun untuk memperkuat *tower* dan tombol Jual untuk menjual *tower* itu jika sudah merasa *tower* itu tidak berguna namun *tower* hanya dapat dilakukan sekali *upgrade*.
d. Kondisi kemenangan *player* bergantung pada ronde yang diselesaikan dan kondisi kekalahan bergantung pada jumlah nyawa yang dimiliki oleh *player*.

5. SARAN

Penulis menyadari bahwa aplikasi *game* 3D GoDef yang memiliki basis Android ini masih belum sempurna. Penulis memiliki harapan agar pembaca atau *programmer* lain dapat mengembangkan dan memperbaiki kekurangan-kekurangan yang ada pada aplikasi *game* ini. Oleh karena itu, agar penelitian dapat memberikan manfaat bagi pembaca, maka penulis akan memberikan beberapa saran, yaitu:

a. Aplikasi *game* 3D GoDef yang dirancang masih belum memiliki tampilan yang menarik, terutama pada objek-objek yang digunakan dalam permainan. Disarankan bagi pembaca yang ingin mengembangkan permainan ini untuk memperbaharui objek-objek permainan dengan model-model yang lebih realistis, misalnya seperti pada arena permainan, yang saat ini masih terlalu polos dapat dibuat dengan model yang lebih realistis. Markas *player* dan tempat musuh muncul yang sekarang hanya berbentuk bulat juga dapat dibuat lebih bagus.
b. Aplikasi *game* GoDef yang dirancang belum sepenuhnya berbentuk 3D (tiga dimensi), disarankan untuk merancang aplikasi *game* ini menjadi sepenuhnya 3D.
c. Menambahkan jenis *tower,* fitur baru seperti *skill* yang dapat dilakukan oleh *tower* dan juga jenis musuh yang lebih beragam agar permainan dapat menjadi lebih menarik dan tidak membosankan.
d. Menambahkan jumlah level permainan dan tingkatan musuh yang lebih kuat agar permainan tidak cepat berakhir.
e. Menambahkan bentuk *map* yang berbeda agar tampilan permainan tidak membosankan.
f. Menambahkan animasi yang lebih bagus pada tiap aksi yang terjadi pada permainan akan membuat permainan lebih interaktif.
g. Menambahkan tampilan level yang dimainkan dan jumlah ronde yang ada di level tersebut.

UCAPAN TERIMAKASIH

Penulis mengucapkan banyak terima kasih kepada ketua Universitas Widya Dharma Pontianak, dosen pembimbing, keluarga, sahabat, dan semua pihak yang tidak dapat penulis tulis satu per satu yang telah memberikan bimbingan, arahan, saran, dan kritik dalam proses menyelesaikan penelitian ini.

DAFTAR PUSTAKA

[1] Adams, E. (2014). *Fundamentals of Game Design Third Edition.* United States of America: Penerbit New Riders.
[2] Asmiatun, S., dan Astrid Novita Putri. (2017). *Belajar Membuat Game 2D dan 3D Menggunakan Unity.* Yogyakarta: Penerbit Deepublish.
[3] Dalem, I. B. (April 2018). Jurnal Resistor. *Penerapan Algoritma A*(STAR) Menggunakan Graph Untuk Menghitung Jarak Terpendek*, Vol 1, no. 1,hal 43.
[4] Hermanto, D., dan Sepri Dermawan. (Juni 2018). Jurnal Nasional Teknik Elektro. *Penerapan Algoritma A* Sebagai Pencari Rute Terpendek pada Robot Hexapod*, Vol 7, no.2, hal 1.
[5] Juhara, Z. P. (2016). *Panduan Lengkap Pemrograman Android.* Yogyakarta: Penerbit CV Andi Offset.
[6] Supardi, Y. (2017). *Koleksi Program Tugas Akhir dan Skripsi dengan Android.* Jakarta: Penerbit PT Elex Media Komputindo.