BAB I

PENDAHULUAN

1.1 Latar Belakang

Dewasa ini perkembangan teknologi khususnya teknologi dalam permainan mengalami perkembangan yang cukup pesat. Pada Penelitian *Entertainment Software Association* 2011, di Asia Tenggara pemain *game* usia remaja (dibawah 18 tahun) adalah sebanyak 40%. Hal tersebut menunjukkan bahwa rata-rata remaja di usia sekolah menengah pernah memainkan *game*. Salah satu komponen *game* yang sekarang banyak dikembangkan yaitu teknologi *NPC* (*Non-Player Character*). *NPC* adalah jenis *otonomus agent* yang ditujukan sebagai agen yang mewakili toko pendukung dalam sebuah permainan (Arif, 2010). *Non-Player Character* memainkan peranan penting dalam banyak permainan, seperti menyajikan *story-line*, menjadi musuh ataupun teman dan memberikan informasi kepada pengguna yang berhubungan dengan permainan (Hermawan, et al., 2017). Dalam pembuatan *game* memerlukan desain *Artificial Intelligence* yang baik untuk mengontrol *NPC*.

Salah satu penerapan *Artificial Intelligence* pada *game* yaitu pada pergerakan NPC. Pergerakan bertujuan untuk membuat NPC bergerak layaknya pergerakan yang dipikirkan oleh manusia. Pemilihan pergerakan yang independen mampu membuat permainan menjadi lebih menarik. Namun, perilaku independen tanpa disertai dengan kecerdasan justru dapat mengurangi daya tarik permainan (Haida, et al., 2016). Salah satu contoh *game* yang menggunakan *Artificial Intelligence* adalah *game* yang bergenre RTS (*Real-Time Strategy)*. *Real Time Strategy* merupakan turunan dari sub-genre *game Strategy*. *Real Time Strategy* sendiri adalah *game* berjalan dalam waktu sebenarnya dan serentak antara semua pihak dan pemain harus memutuskan setiap langkah yang diambil saat itu juga bersamaan mungkin saat itu pihak lawan juga sedang mengeksekusi strateginya (Hasan, 2009).

Dalam game RTS (*Real Time Strategy*) pergerakan dan perilaku NPC diatur dengan sebuah algoritma. Algoritma yang digunakan yaitu Algoritma A* berbasis *Navigation Mesh.*

Algoritma A* merupakan algoritma *Best First Search* (BFS) yang menggabungkan *Uniform Cost Search* dan *Greedy Best First Search*. Algoritma A* digunakan dalam melakukan pencarian jalur yang optimal yang menghubungkan dua titik dari permainan yang ada (Hermawan & Bellaniar I, 2015). Dengan jalur yang dapat dicari dengan perhitungan algoritma A* maka NPC akan bergerak sesuai dengan perhitungan algoritma tersebut. Perhitungan dari pencarian rute tersebut dibantu dengan basis dari *Navigation Mesh* dalam pembentukan rute yang dapat dilalui.

Navigation Mesh sendiri merupakan sebuah metode yang digunakan untuk mengatur pergerakan *NPC*. *Navigation Mesh* merupakan suatu bentuk struktur data yang sangat dibutuhkan dalam *pathfinding* dalam *game* 3D (Yonathan, 2011). Sehingga *Navigation Mesh* digunakan untuk mengatur pergerakan dari titik awal hingga titik akhir agar pergerakan tersebut tidak terjadi tabrakan antar NPC dan juga NPC terhadap *obstacle*.

Game yang akan di buat adalah game *Fortress Fight*. *Game Fortress Fight* sendiri merupakan *game* bergenre *real time strategy* yang di mana *game* ini adalah permainan satu orang (*Single Player Game*). *Game* ini memiliki mekanisme permainan di mana pemain dapat menggerakkan *player* untuk menyerang ke benteng musuh dan di saat bersamaan musuh juga dapat menyerang *player* dan menghancurkan benteng *player*. Kondisi *game* akan berakhir dengan dua kemungkinan di mana jika benteng musuh hancur maka *player* akan menang sedangkan jika benteng *player* yang hancur maka musuh yang akan menang atau *game over*.

Pada penelitian sebelumnya di UKMC terdapat penelitian yang menerapkan Algoritma A* untuk mengatur pergerakan NPC. Tetapi terdapat kekurangan pada penelitian sebelumnya yaitu NPC sering tersangkut, sehingga saran dalam penelitian sebelumnya adalah menerapkan *navigation mesh*.

Pada penelitian ini penulis menerapkan Algoritma A* berbasis *Navigation Mesh* pada *game Fortress Fight*. Penulis menggunakan Algoritma A* berbasis *Navigation Mesh* karena dirasa sebagai algoritma yang cukup mangkus dan dipakai dalam berbagai *game*, dalam pergerakan *navigation mesh* menggunakan perhitungan *node* sehingga cocok untuk di gunakan dalam *navigation mesh* dan unity dan juga pada penelitian sebelumnya juga terdapat saran untuk menambahkan *navigation mesh* agar pergerakan NPC tidak sering tersangkut.

1.2 Rumusan Masalah

Berdasarkan pendahuluan di atas maka dapat dirumuskan masalah adalah Bagaimana mengatur pergerakan NPC dalam sebuah *game* agar NPC dapat bergerak mencari *player*?

1.3 Batasan Masalah

Batasan masalah dalam penelitian ini adalah sebagai berikut:

1. *Game* dibuat hanya untuk menerapkan Algoritma A* berbasis *Navigation Mesh* pada *game Fortress Fight* untuk mengatur pergerakan NPC.
2. NPC diuji coba dengan cara NPC dijalankan untuk mencari dan menyerang *player* serta benteng *player* pada lokasi permainan.
3. *Game* merupakan permainan *single player* di mana pemain hanya dapat menggerakkan *player*.
4. *Game* dibuat menggunakan *Unity 3D*.

1.4 Tujuan dan Manfaat Penelitian

1.4.1 Tujuan Penelitian

Tujuan penelitian ini adalah menerapkan Algoritma A* berbasis *Navigation Mesh* pada *game Fortress Fight* untuk mengatur pergerakan NPC.

1.4.2 Manfaat Penelitian

Manfaat dari penelitian ini adalah membuat sensasi permainan dapat lebih realistis dan seolah-olah seperti manusia yang dapat berpikir dan juga membuat

pergerakan NPC dalam *game* lebih teratur saat melewati rintangan dan mencari *player*.

1.5 Metodologi Penelitian

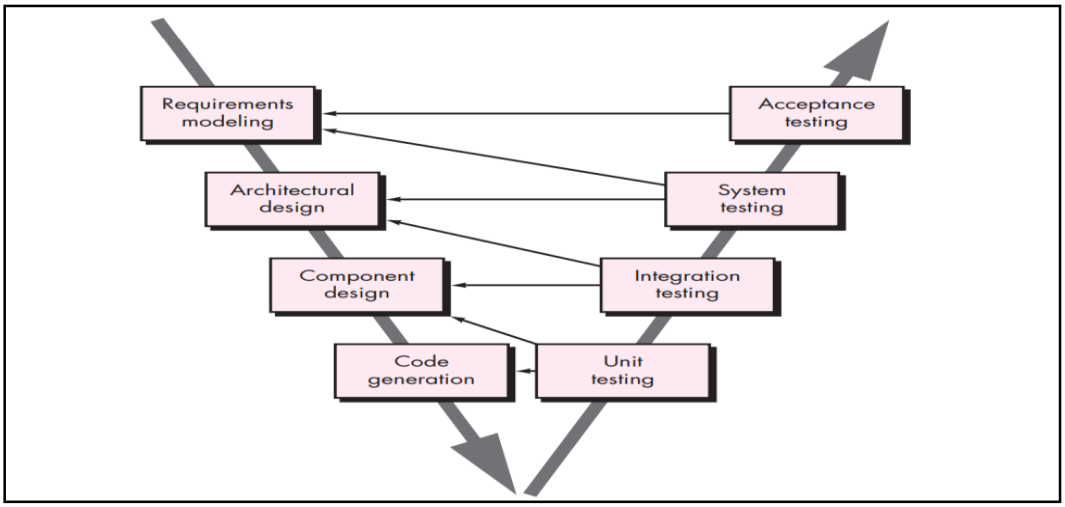
Metodologi penelitian dalam membangun aplikasi kompresi audio ini akan dijabarkan sebagai berikut.

1.5.1 Jenis Penelitian

Berdasarkan klasifikasi penelitian berdasarkan tujuan dan manfaatnya, penelitian ini merupakan jenis penelitian terapan. Penelitian ini dilakukan untuk menerapkan Algoritma A* berbasis *Navigation Mesh* untuk mengatur pergerakan NPC pada game *Real Time Strategy* berjudul *Fortress Fight*

1.5.2 Metode pengembangan sistem

Metode pengembangan sistem yang digunakan dalam penelitian ini adalah metode *V-Model*. Model ini merupakan perluasan dari model *waterfall*. Disebut sebagai perluasan karena tahap-tahapnya mirip dengan yang terdapat dalam model *waterfall*. Jika dalam model *waterfall* proses dijalankan secara linear, maka dalam *V-Model* proses dilakukan bercabang. Dalam *V-Model* ini digambarkan hubungan antara tahap pengembangan *software* dengan tahap pengujiannya (Pressman, 2010). Adapun ilustrasinya sebagai berikut :



Gambar 1.1 : Model *V-Model*

(Sumber: Pressman, 2010)

Berikut penjelasan berdasarkan Gambar 1.1 :

a. Requirements Modeling

Requirements modeling merupakan proses pengumpulan informasi tentang kebutuhan-kebutuhan pengguna terhadap perangkat lunak yang akan dikembangkan. Informasi ini nantinya akan digunakan sebagai acuan untuk mengetahui fitur apa saja yang akan ada pada *game*.

b. Architectural Design

Architectural design merupakan proses pembuatan desain arsitektur dari game yang akan di buat. Bagian dari *architectural design* adalah membuat rancangan sistem dan juga disain arsitektur sistem.

c. Component Design

Component design merupakan proses menciptakan desain untuk perangkat *game* dan desain *game* yang akan diimplementasikan. Pada tahap ini peneliti membuat *design game* dari *game Real Time Strategy.*

d. Code Generation

Code generation merupakan proses membuat kode. *Coding* atau pengkodean merupakan penerjemahan desain *game* dalam bahasa yang bisa dikenali oleh komputer. *Programmer* akan menerjemahkan konsep-konsep yang digunakan dalam *game*. Tahapan inilah yang merupakan tahapan secara nyata dalam mengerjakan suatu software (*game*), artinya penggunaan komputer akan dimaksimalkan dalam tahapan ini.

e. Unit Testing

Unit testing merupakan proses pengujian perangkat lunak dilakukan untuk memastikan bahwa perangkat lunak yang di kembangkan sudah berjalan dengan semestinya. Unit testing menggunakan teknik *White Box* dalam pengujiannya.

f. Integration Testing

Integration testing merupakan proses pengujian hasil dari pengabungan unit-unit/komponen yang berinteraksi di dalam software. *Integration testing* menggunakan teknik *Black Box* dalam pengujiannya.

g. System Testing

System testing merupakan proses pengujian yang dilakukan terhadap keseluruhan sistem (secara lengkap) dan sistem yang telah terintegrasi untuk mengevaluasi apakah sistem yang dibuat telah sesuai dengan kebutuhan pengguna.

h. Acceptance Testing

Acceptance testing merupakan proses pengujian oleh pengguna yang dimaksudkan untuk menghasilkan dokumen yang dijadikan bukti bahwa software yang telah dikembangkan telah dapat diterima oleh pengguna, apabila hasil pengujian (*testing*) sudah bisa dianggap memenuhi kebutuhan dari pengguna.

1.6 Sistematika Penulisan

Penulisan laporan dilakukan secara sistematis menggunakan lima bab. Gambaran umum mengenai isi laporan penelitian secara keseluruhan adalah sebagai berikut.

BAB I PENDAHULUAN

Pada bab ini akan dijelaskan mengenai latar belakang permasalahan, perumusan masalah, batasan masalah, tujuan dan manfaat penelitian, metodologi penelitian, serta sistematika penulisan.

BAB II LANDASAN TEORI

Pada bab ini akan dijelaskan mengenai landasan teori dan perbandingan literatur yang dijadikan landasan dalam mengimplementasikan Algoritma *Arithmetic Coding* pada kompresi audio.

BAB III ANALISIS DAN PERANCANGAN

Pada bab ini berisi analisis kebutuhan sistem pada aplikasi yang akan dibangun, desain UML dan *flowchart*, dan desain antarmuka sistem.

BAB VI IMPLEMENTASI DAN PENGUJIAN

Pada bab ini berisi proses implementasi aplikasi, tampilan aplikasi yang dibangun, dan hasil pengujian aplikasi. Pengujian sistem

menggunakan *black box testing, white box testing,* dan pengujian deskriptif.

BAB V PENUTUP

Bab ini berisi kesimpulan secara umum dan saran-saran yang dapat digunakan dalam pengembangan lebih lanjut di masa yang akan datang.