# BAB II LANDASAN TEORI

Sebelum melakukan desain perancangan dari gim yang akan dibuat, ada baiknya jika dilakukan pembahasan terhadap struktur dari sebuah gim mulai dari pengertian sampai dengan pembuatannya.

## 2.1 Kajian terdahulu

Terdapat beberapa penelitian sebelumnya yang meneliti tentang kasus serupa mengenai penggunaan algoritma A\* sebagai algoritma *pathfinding* pada gim. Kajian penelitian terdahulu tersebut diuraikan sebagai berikut.

### **2.1.1** **Algoritma Pathfinding A\* pada Game RPG Tanaman Higienis** (Pramono, 2015)

Pada penelitian ini algoritma A\* digunakan pada pergerakan aktor pada gim bergenre simulasi pertanian 3D. Penggunaan algoritma *pathfind* ditujukan agar aktor tidak mengalami *stuck* saat digerakkan oleh pemain. Jadi pada penelitian ini dapat disimpulkan bahwa *pathfind* diterapkan pada subjek yang bersifat dinamis namun objek yang menjadi tujuan *pathfind* bersifat statis. Karena objek *pathfind* tidak berubah posisi maka perhitungan algoritma A\* dalam *pathfinding* hanya perlu dilakukan sekali saja.

### 2.1.2 Penerapan Algoritma A Star (A\*) pada Game Petualangan Labirin Berbasis Android (Ahmad, 2017)

Pada penelitian ini, gim yang dibuat memiliki genre *adventure* dan *puzzle.* Sedangkan engine yang digunakan untuk membuat gim adalah Android Studio. Pada gim ini, pemain akan menggerakkan aktor yang berupa kelinci menuju makanan kelinci melewati suatu labirin. *Pathfinding* pada gim ini dibuat dengan menggunakan algoritma A\* untuk membantu pemain jika pemain mengalami kesusahan menemukan jalur menuju makanan kelinci. Jadi pemain dapat menggunakan tombol bantuan untuk mengaktifkan *pathfinding* algoritma A\* (A Star) untuk menuju lokasi tempat makanan kelinci berada. *Pathfinding* hanya akan diaktifkan ketika pemain menekan tombol bantuan dan *pathfinding* akan dijalankan sekali saja.

### 2.1.3 Penerapan Algoritma A\* Pathfinding Dalam Pencarian Solusi Game Peanut Labirin Dengan Macromedia Flash (Wibowo, 2012)

Pada penelitian ini dibuat sebuah gim berjudul Peanut Labirin dengan genre *puzzle* 2D*.* Pada gim ini pemain akan menggerakkan aktor berupa tupai menuju kacang yang merupakan tujuan gim agar gim dapat dimenangkan. Pemain harus dapat menggerakkan aktor melewati labirin dengan waktu terbatas. Jika waktu habis maka permainan akan berakhir. Pada gim ini algoritma A\* diterapkan sebagai *pathfind* untuk membantu pemain jika pemain mengalami kesulitan menemukan posisi kacang. *Pathfind* dapat diaktifkan oleh pemain dengan menekan tombol bantuan. Setelah tombol ditekan maka aktor akan bergerak secara otomatis menuju posisi kacang. Dalam hal ini, kacang yang merupakan tujuan *pathfinding* bersifat statis sehingga perhitungan algoritma A\* dalam *pathfind* lebih sederhana.

### 2.1.4 Perbedaan Penelitian ini dengan kajian penelitian sebelumnya

Penerapan algoritma A\* sebagai *pathfinding* pada penelitian sebelumnya selalu diterapkan pada pergerakan aktor dan bukannya pada pergerakan NPC seperti musuh. Jadi pada penelitian sebelumnya, *pathfind* selalu diterapkan dengan objek *pathfinding* yang bersifat statis. Karena objek *pathfinding* tidak mengalami perubahan posisi maka proses perhitungan *pathfind* hanya perlu dilakukan sekali saja untuk menemukan jalan yang paling efektif menuju posisi objek. Perbedaan penelitian ini dengan penelitian sebelumnya terletak pada objek *pathfind* yang bersifat dinamis. Pada penelitian ini *pathfind* akan diterapkan pada pergerakan musuh. Sedangkan yang menjadi tujuan atau objek dari *pathfind* adalah aktor yang akan digerakkan oleh pemain sehingga posisi aktor akan selalu mengalami perubahan. Karena objek yang menjadi tujuan dari *pathfind* selalu mengalami perubahan maka perhitungan algoritma pada *pathfind* harus dilakukan berulang – ulang untuk dapat mengimbangi pergerakan aktor yang digerakkan pemain. Pada penelitian sebelumnya, perhitungan *pathfind* baru akan dihentikan ketika pengguna *pathfind* dapat menemukan dengan tepat posisi dari objek *pathfind.* Namun pada penelitian ini, *pathfind* dapat dimanipulasi sehingga musuh dapat berhenti beberapa petak dari posisi aktor untuk melakukan serangan jarak jauh.

## 2.2 Pengertian gim

Gim adalah suatu kegiatan yang dilakukan dengan aturan yang sudah ditentukan sebelumnya sehingga akan menciptakan suatu kondisi tertentu di mana ada kondisi menang dan kondisi kalah, biasanya kegiatan ini menitik beratkan pada unsur hiburan dan refreshing.

Istilah gimpertama kali ditemukan pada tahun 1944 oleh 2 orang ahli matematika. Teori mengenai gim dikemukakan oleh John von Neumann and Oskar Morgenstern yang berisi:

“Permainan terdiri atas sekumpulan peraturan yang membangun situasi bersaing dari dua sampai beberapa orang atau kelompok dengan memilih strategi yang dibangun untuk memaksimalkan kemenangan sendiri ataupun untuk meminimalkan kemenangan lawan. Peraturan-peraturan menentukan kemungkinan tindakan untuk setiap pemain, sejumlah keterangan diterima setiap pemain sebagai kemajuan bermain, dan sejumlah kemenangan atau kekalahan dalam berbagai situasi “ (Neumann & Morgenstern, 1944).

Sedangkan menurut Clark C. Abt, “Gim adalah kegiatan yang melibatkan keputusan pemain, berupaya mencapai tujuan dengan dibatasi oleh konteks

tertentu(misalnya, dibatasi oleh peraturan)” (Abt, 1970). . Namun menurut Roger E. Pederson, “gim dapat dikatakan sebuah gim jika memiliki 4 syarat yaitu tidak linier, mempunyai tujuan / goal, dapat dimenangkan, dan memiliki bagian awal, pertengahan dan akhir” (Pederson, 2003).Jadi dapat disimpulkan bahwa gimadalah suatu kegiatan yang dilakukan dengan tujuan untuk mengalahkan lawan dengan mengikuti peraturan – peraturan tertentu.

## 2.3 Sejarah gim

Dunia gimternyata memiliki sejarah yang cukup panjang dan komplex. Perkembangan gim di dunia dibagi menjadi generasi ke generasi dimulai dari gimyang berbasis *bit* atau *text* hingga saat ini di mana gimdibuat begitu nyata sampai tidak dapat dibedakan mana yang virtual dan mana yang nyata. Perkembangan gimberdasarkan generasinya dimulai dari :

### 2.3.1 Generasi pertama

Pada tahun 1952, Alexander Shafto Douglas menciptakan sebuah permainan XOX (*tic-tac-toe)* dalam versi grafik pada komputer*.* Permainan ini ia kembangkan ketika hendak mendemonstrasikan tesisnya tentang interaksi antara manusia dan komputer pada saat ia mengejar gelar di Universitas Cambridge. Dari sinilah titik awal perkembangan gim mulai bermunculan.

### 2.3.2 Generasi ke-2

Pada tahun 1976, Atari mendesain sebuah video gimyang bernama Pong. Namun ditahun yang sama, pendiri atari Busnell menjual saham Atari pada Warner communication meskipun ia masih bekerja pada Atari. Atari merupakan perusahaan yang menciptakan gim *console* pertama di dunia yang diberi nama sesuai dengan nama perusahaannya yaitu gim *console Atari.*  Pada generasi inilah istilah gim *console* mulai dikenal dunia.

### 2.3.3 Generasi ke-3

Tahun 1983, gim *console* yang dikenal dengan nama NES dan SG-1000 mulai dipasarkan di dunia. NES diciptakan oleh Japan Family Computer “FAMICOM” yang sekarang lebih dikenal dengan sebutan Nintendo Entertainment System (NES). Sedangkan perusahaan yang menjadi pesaingnya adalah Sega Enterprise Tokyo (SEGA) dengan gim *console* buatannya yaitu SG-1000.Kedua perusahaan inilah yang memulai perang gim *console* pertama di dunia.

### 2.3.4 Generasi ke-4

Tahun 1988, adalah tahun di mana gimdengan grafik 16 bit mulai dikembangkan. Pada generasi ini juga produsen – produsen gim *console* semakin marak bermunculan seperti SNK dengan *console* NeoGeo-nya, NEC dengan *console* TurboGrafx-16 dan Phillips CD-i. Meskipun tidak bertahan lama, munculnya perusahaan – perusahaan kecil yang memproduksi gim *console* menandakan bahwa gimbukanlah lagi hal yang tidak lazim di mata dunia atau setidaknya sudah umum di Jepang. Generasi ini menunjukkan bahwa pada generasi – generasi berikutnya akan muncul gim *console* – gim *console* baru yang diproduksi oleh perusahaan baru.

### 2.3.5 Generasi ke-5

Pada tahun 1990 – 1994 mulai diperkenalkannya gim *console* yang menggunakan *compact disc (CD).* Kemunculan media baru sebagai gim *console* semakin memanaskan persaingan antara Sega dengan Nintendo ditambah lagi dengan munculnya gim *console* baru bernama *Playstation* buatan Sony. Berbeda dengan *console* buatan Sega dan Nintendo yang masih menggunakan *cartridge*, *Playstation* memanfaatkan media baru *CD* dengan baik sehingga Sony dapat menekan biaya produksi video gimdengan efisien karena *CD* lebih mudah digunakan dan lebih murah daripada *cartridge*. Kelebihan penggunaan *CD* daripada *cartridge* mengakibatkan banyaknya developer gimyang membuat gimuntuk *Sony* daripada *sega* atau *nintendo.*

### 2.3.6 Generasi ke-6

Pada tahun 1998, Sega menghentikan produksi gim *console* setelah kegagalan Sega dalam merilis *console* terbarunya yaitu Sega *Dreamcast*. Hal ini juga menandakan kemenangan Sony dalam perang gim *console* dengan Nintendo dan Sega.

### 2.3.7 Generasi ke-7

Pada tahun 2005, Perusahaan OS terkenal di dunia bernama *Microsoft* mencoba membuka pasar pada bidang gim *console* dengan gim *console* mereka bernama *Xbox 360.* Kemunculan *Xbox 360* dengan fitur *Xbox Live* juga merupakan awal dari kemunculan berbagai macam *online* gimpada *home console.* Kepopuleran *Xbox* juga menandakan perang gim *console* kedua antara Sony dengan *Playstation* melawan *Microsoft* dengan *Xbox*.

Sekarang ini perkembangan gim *console* semakin maju dengan dibuatnya teknologi yang mendukung seperti *Virtual Reality (VR), Augment Reality (AR), Motion Sensor, Sound Sensor,* dan lain sebagainya.

## 2.4 *Genre* Gim

*Genre* gim adalah klasifikasi dari beragam jenis video gimyang dibedakan berdasarkan *gameplay* yang ada pada gimtersebut*. Genre* gimmemudahkan pemain untuk menemukan jenis gimdengan *gameplay* yang mereka suka. *Genre* gimyang ada sekarang ini sangatlah beragam mulai dari *first person shooting (FPS), Third person shooting (TPS), Action, Adventure, (role playing* gim*) RPG, visual novel, fighting, hack ’n slash, racing, Real time strategi (RTS),* dan lain sebagainya. Untuk penelitian ini, *genre* yang akan dibahas lebih detail adalah *genre action, adventure,* dan *RPG.*

### 2.4.1 *Genre* *Action*

Gimdengan *genre action* membutuhkan kecepatan, kelincahan dan ketangkasan pemain dalam memainkan gimdengan *genre* ini. Gimdengan *genre action* lebih menonjolkan ketangkasan seorang pemain mengambil keputusan, kelincahan pemain dalam bermain dan kecepatan pemain dalam mengamati situasi daripada memikirkan strategi yang tepat. Dalam *action* gim*,* pemain biasanya menggerakan seorang karakter sebagai tokoh utama / protagonis dari gimtersebut. Pemain diminta untuk menyelesaikan sebuah misi tertentu dengan mengalahkan lawan, menyelamatkan seseorang, mengumpulkan sebuah objek tertentu, pergi ke suatu tempat, melewati rintangan sampai dengan mengalahkan tokoh antagonis sebagai bos akhir yang harus dikalahkan. Gimyang dibuat dengan *genre* ini biasanya memiliki tingkat kesulitan yang cukup tinggi dan membutuhkan waktu yang relatif singkat untuk mencapai akhir gim*.*  *Sub-genre* dari *genre* ini adalah *shooting (First Person Shooting / Third Person Shooting), Hack ‘n Slash, Beat ‘em Up,* dan lain sebagainya. Contoh dari gimdengan *genre* ini adalah *Legend of Zelda, Final Fantasy XV, Kingdom Hearts.*

### 2.4.2 *Genre* *Adventure*

Gimdengan *genre adventure* merupakan gimyang fokus pada naratif alur cerita pada gim*. Gameplay* yang ditawarkan oleh gimini lebih fokus pada alur cerita yang harus diambil pemain daripada bertarung dengan musuh. Salah satu fitur yang paling digemari para pemain gim pada *genre* ini adalah *dialogue tree* dan *butterfly effect. Dialogue tree* merupakan fitur di mana pemain dapat memilih beberapa dialogketika berinteraksi dengan *NPC.* Dialog yang dipilih akan mempengaruhi alur interaksi antara pemain dengan *NPC.* Sedangkan *Butterfly effect* adalah fitur di mana tindakan atau pilihan yang dilakukan pemain akan mempengaruhi alur cerita utama pada gim*.* Jadi jika *dialogue tree* hanya sebatas mempengaruhi dialog dengan *NPC, butterfly effect* akan mempengaruhi seluruh alur cerita, vital, dan bersifat permanen . Contoh dari gimdengan *genre* ini adalah *The Witcher 3, Elder scroll : Skyrim, undertale.*

### 2.4.3 *Genre* RPG

Gim dengan *genre RPG* adalah gimdi mana pemain akan memainkan seorang tokoh fiksi dengan latar belakang dan alur cerita yang sudah dibuat sebelumnya. Tokoh fiksi dalam gim *RPG* sudah memiliki latar belakang entah tokoh tersebut seorang pahlawan, raja, ataupun tentara sehingga pemain hanya perlu mengikuti kisah dari tokoh fiksi tersebut. Ciri khas dari gim *RPG* adalah perkembangan karakter. Gim *RPG* memiliki alur cerita yang berkaitan langsung dengan perkembangan karakter pada gimtersebut. Perkembangan yang dimaksud bukan hanya perkembangan kepribadian dan watak saja namun juga perkembangan kemampuan dan kekuatan tokoh. Perkembangan karakter seperti watak dan kepribadian tokoh sebagian besar dipengaruhi oleh jalan cerita dari gimsedangkan perkembangan kekuatan (*level)* dan kemampuan (*skill)* dipengaruhi oleh misi (*quest)* yang diselesaikan pemain. Misi yang diberikan dapat berupa mengalahkan lawan, mengumpulkan barang tertentu, menjelajahi suatu tempat, ataupun menyelamatkan seseorang. *Gameplay* dalam *genre* ini dirancang agar pemain menghabiskan banyak waktu pada gimini. Berkebalikan dengan *genre action, genre RPG* memiliki tingkat kesulitan yang rendah dan membutuhkan waktu yang lama untuk mencapai bagian akhir gim*.* Contoh gimdengan *genre* ini adalah *final fantasy, dragon quest, persona.*

### 2.4.4 *Genre* Hybrid

*Genre Hybrid* adalah sebuah gimyang menggabungkan beberapa *genre* menjadi satu. Gimdengan *genre* *hybrid* mampu menciptakan sebuah alur cerita dan *gameplay* yang unik jika *genre* tersebut dipadukan dengan benar. Sebagai contoh dan *legend of zelda* merupakan gimdengan *genre RPG, action,* dan *adventure.* Sedangkan gimseperti *half life* merupakan gabungan dari *genre FPS, adventure,* dan *RPG.*



##### Gambar 2.1 *Legend Of Zelda* Yang Bergenre *Action Adventure RPG*

## 2.5 Elemen pada Gim

Menurut Monica Wachowicz,

“ada 11 elemen gimyang perlu diperhatikan sebagai dasar dalam membuat gimyang baik. Ke-11 elemen tersebut adalah:

* + - 1. **Format**, Mendefinisikan bentuk dari gim. Sebuah gim terdiri dari beberapa *level*, dan setiap *level* tersebut memiliki fungsinya masing-masing. Gabungan dari setiap *level* yang ada pada gimitulah yang disebut *format. Format* dari gimakan mendefinisikan *genre* dari gimtersebut.
      2. ***Rules***, Di dalam sebuah gim, harus terdapat perjanjian atau peraturan yang tidak dapat diubah atau dihapus oleh pemain. Oleh karena itu, dalam memainkan suatu gim, pemain harus patuh dan bermain sesuai aturan yang berlaku. *Rules* dibuat untuk membatasi kebebasan pemain di dalam gim. Salah satu dari fungsi *rules* adalah untuk menciptakan keseimbangan akan tingkat kesulitan di dalam gim. Biasanya ada hukuman tertentu bagi pemain yang berani melanggar *rules* yang ada di dalam gimseperti tidak dapat melanjutkan ke *level* berikutnya atau tidak dapat menyimpan data. Meskipun demikian tetap saja ada beberapa pemain yang berani berbuat curang dengan menggunakan *cheat code, bot, piracy* dan lain sebagainya.
      3. **Policy**, atau kebijakan dapat didefinisikan sebagai aturan yang bisa diubah oleh pemain. Dengan adanya elemen ini, maka pemain akan dapat menggunakan dan mengembangkan strategi dalam bermain gim sesuai kemampuan dirinya. Salah satu contoh dari *policy* adalah gim *difficulty* yang biasanya dibagi menjadi 3 yaitu mudah, sedang, dan sulit. Pemain bebas mengganti gim *difficulty* sesuai dengan kemampuan pemain.
      4. ***Scenario***, Merupakan alur cerita yang digunakan sebagai jalan atau panduan dalam bermain gim. *Scenario* yang dibuat tidak harus linear namun dapat melibatkan *multiple scenario. Multiple scenario* biasanya melibatkan *dialogue tree* dan *butterfly effect* sehingga pemain dapat memilih salah satu alur cerita dari beberapa alur yang ada. *Multiple scenario* digunakan untuk memberikan pemain sebuah sensasi bahwa pemain itu sendiri terlibat langsung pada alur cerita.
      5. ***Event*s**, adalah sebuah peristiwa yang berupa tantangan bagi pemain sekaligus menambah keceriaan dalam bermain gim. Contoh *event* dalam gim diantaranya adalah berupa konflik, misi, pertarungan dan kompetisi. Berbeda dengan *scenario* di mana pemain harus mengikuti alur cerita yang ada untuk menuju akhir permainan*, events* sebagian besar dibuat agar pemain tidak jenuh atau kewalahan dengan *scenario* yang ada. Dengan demikian *events* tidaklah wajib diselesaikan oleh pemain.
      6. ***Roles***, Sebuah gambaran dari fungsi dan peran yang dapat dibagi antar pemain di dalam bermain gim. *Roles* dibuat untuk menciptakan keseimbangan di dalam gim*.* Gimdengan desain yang baik akan memiliki pembagian *roles* yang merata dan sederajat. Pembagian *roles* yang tidak seimbang akan membuat *spamming* pada *roles* tertentu dan membuat kesan bahwa *roles* yang lainnya tidak berguna.
      7. ***Decision*s**, *Decision*s merupakan suatu keputusan yang harus diambil oleh si pemain di dalam bermain gim. *Decision* berhubungan langsung dengan *scenario* dan *events* melalui *dialogue tree* dan *butterfly effect.* Elemen ini dibuat agar pemain tidak hanya mengikuti atau membaca alur cerita tetapi juga berperan dan mengambil keputusan di dalam cerita tersebut.
      8. ***Level*s**, Sebuah gim perlu memiliki *level* tingkat kesulitan agar gim tersebut lebih menarik dan menantang, serta dapat digunakan oleh banyak pemain. Setiap *level* akan dibuat bertingkat sehingga semakin tinggi tingkatnya akan semakin sulit *level* tersebut. *Level* 1 atau tutorial *level* ditujukan untuk mengajari pemain tentang mekanisme dan fungsi – fungsi yang ada di dalam gim*.* Namun *level* terakhir akan dibuat jauh lebih sulit untuk menguji kemampuan pemain dalam memainkan gim*.*
      9. ***Score model***, merupakan elemen yang digunakan untuk menghitung, mencatat, dan menampilkan hasil dari gim yang dimainkan. *Score model* ada untuk menilai kemampuan pemain dalam memainkan gimtersebut. Elemen ini digunakan untuk membuat pemain memainkan kembali gimtersebut dengan tujuan untuk memperoleh *score model* yang lebih tinggi.
      10. ***Indicators***, *Indicators* memberikan pemain suatu petunjuk (*hints*) terhadap raihan atau pencapaian yang telah mereka lakukan. *Indicator* dibuat agar pemain bermain dengan baik untuk mencapai indikator – indikator yang ada. Elemen ini penting adanya untuk menjaga agar pemain bisa selalu termotivasi dan fokus dalam bermain.
      11. ***Symbols***, Bentuk visual dari simbolisasi elemen, aktivitas, dan keputusan. *Symbols* erat kaitannya dengan *user interface (UI)* atau *user experience (UX).* Pemilihan simbol yang tepat akan membantu pemain dalam memahami gim yang dimainkan. Keberadaan elemen ini sangat penting karena sebelum pemain bermain gim, pemain pertama kali akan melihat *UI* atau *UX* yang ada pada gim*.* Kesalahan pada pemilihan, penggunaan, ataupun penataan simbol dapat menurunkan keinginan pemain dalam bermain bahkan sebelum pemain tersebut memulai permainan.” (Wachowicz, 2003)

## 2.6 Pathfinding

*Pathfinding* dalam Kamus Besar Bahasa Indonesia berawal dari kata *path* yang berarti “jalan kecil / sempit” dengan kata *finding* yang berarti “menemukan”. Jadi secara garis besar, *pathfinding* adalah metode untuk menemukan jalan. *Pathfinding* dalam videogimmerupakan sebuah metode untuk bagaimana cara suatu subjek agar dapat bergerak melewati jalan optimal dengan menghindari rintangan yang ada untuk mencapai tujuan dari subjek tersebut. “*Pathfinding* adalah proses pemindahan posisi karakter gimdari lokasi awal ke lokasi tujuan yang diinginkan” (Bourg & Seemann, 2004). *Pathfinding* dapat diimplementasikan dalam *video* gimdengan menggunakan algoritma *pathfinding.*

*“*Sebuah upaya untuk menemukan jalur terpendek merupakan hasil optimalisasi dari *pathfinding*. Setiap jalur yang ditemukan pada *pathfinding* akan dicari jalur terpendeknya. Pencarian jalur terpendeknya akan tentukan dengan penjumlahan besaran vektor yang dilalui untuk mencapaitujuan dari dilakukannya *pathfinding*. Akumulasi biayayang memiliki nilai minimum merupakan jalur terpendek dari semua jalur yang ada.”. (Ulva, 2014)

*Pathfinding* sekarang ini sudah menjadi unsur pokok yang harus ada pada sebuah videogim*.* Semakin rumit gimyang dibuat maka akan semakin rumit pula algoritma *pathfinding* yang harus digunakan. *Pathfinding* dapat diterapkan pada pergerakan *NPC /* unit musuh sehingga musuh dapat bergerak secara efektif menuju posisi pemain dengan menemukan jalur terpendek dan melakukan serangan.

*“Pathfinding* sering menjadi salah satu *bug /* masalah yang paling sering ditemui dari pembuatan kecerdasan buatan yang ada pada gim. *Pathfinding* yang buruk dapat membuat karakter yang ada pada gimmenjadi bodoh.” (Bourg & Seemann, 2004). Jika unit musuh memiliki algoritma *pathfinding* yang burukmaka unit musuh akan sering kali bergerak tidak efektif seperti menabrak penghalang, berputar – putar di satu tempat, mengambil jalan memutar untuk menuju posisi pemain dan lain sebagainya. Penanganan masalah *pathfinding* secara efektif dapat membuat gimlebih menyenangkan dan memberikan pengalaman bermain yang mendalam bagi pemain



##### Gambar 2.2 Contoh Musuh Yang Mengalami *Stuck* Di Gim

## 2.7 Algoritma A\*

Salah satu algoritma yang dapat digunakan sebagai algoritma *pathfinding* adalah algoritma *A star* (A\*). Algoritma A\* adalah salah satu algoritma *pathfinding* yang memperhitungkan jalur yang dapat dilewati ditambah dengan jalur perkiraan yang dapat dilalui untuk mendapatkan solusi yang terbaik. “Algoritma A\* adalah algoritma komputer yang digunakan secara luas dalam *graph* traversal dan penemuan jalur serta proses perencanaan jalur yang bisa dilewati secara efisien di sekitar titik-titik yang disebut *node”* (Reddy, 2013). Algoritma ini merupakan perkembangan dari algoritma *dijkstra.* Bedanya adalah algoritma A\* memperhitungkan nilai heuristik antara titik awal ke titik target. Meskipun algoritma A\* lebih kompleks dibandingkan algoritma *dijkstra,* algoritma ini lebih cepat untuk menemukan jalur terpendek untuk kasus yang lebih rumit.

“Ciri khas dari algoritma A\* adalah pengembangan dari “daftar tertutup” untuk merekam area yang akan dievaluasi, kemudian melakukan perhitungan jarak yang dikunjungi dari titik mulai dengan jarak diperkirakan ke titik tujuan” (Reddy, 2013).

A\* menggunakan jalan dengan biaya paling kecil dari titik ke titik yang membuatnya sebagai algoritma pencarian nilai pertama yang terbaik atau *best first search*. Dengan rumus :

𝑓(𝑥) = 𝑔(𝑥) + ℎ(𝑥)

di mana:

* 𝑔(𝑥) adalah jarak total dari posisi awal ke posisi sekarang.
* ℎ(𝑥) adalah fungsi heuristik yang digunakan untuk memperkirakan jarak dari posisi sekarang ke posisi tujuan. Perlu diingat bahwa fungsi ini merupakan perkiraan dan bukan perhitungan sehingga hasil yang didapat bukan nilai asli melainkan mendekati nilai asli atau nilai estimasi. “Semakin besar keakuratan heuristik, semakin cepat dan pendek posisi tujuan ditemukan dan dengan tingkat akurasi yang lebih baik. Fungi 𝑓(𝑥) = 𝑔(𝑥) + ℎ(𝑥) ini adalah perkiraan saat ini dari jarak terdekat ke tujuan” (Lubis, 2016).

“Algoritma A\* juga menerapkan 2 senarai *Open List* dan *Closed List*. Terdapat 3 (tiga) kategori bagi setiap pewaris yang ada, yaitu sudah berada pada di *Open*, sudah berada di *Closed*, dan tidak berada di *Open* maupun *Closed*.” (Suyanto, 2014)

Jika pewaris sudah pernah berada di *Open*, maka dilakukan pengecekan apakah perlu pengubahan *parent* atau tidak tergantung pada nilai 𝑔(x) melalui *parent* lama atau *parent* baru. Jika melalui *parent* baru memberikan nilai 𝑔(x) yang lebih kecil, maka dilakukan pengubahan *parent*.

Jika pewaris sudah pernah berada di *Closed*, maka dilakukan pengecekan apakah perlu pengubahan *parent* atau tidak, jika ya, maka dilakukan perbaharuan nilai 𝑔(x) dan 𝑓(x) pada pewaris tersebut serta pada semua keturunannya yang sudah pernah berada di *Open*.

Jika pewaris tidak berada di *Open* maupun *Closed*, maka suksesor tersebut dimasukkan ke dalam *Open*. Tambahkan pewaris tersebut sebagai pewarisyang memiliki nilai terkecil atau jalur terpendek. Berikut adalah contoh perhitungan pada algoritma A\* :

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| A |  |  | 13 |  |  | 11 |  |  |  |
| B |  |  | 11 |  | 9 |  |  |  |  |
| C | 11 | 9 | 9 | 9 | 9 | 11 | 13 | 15 |  |
| D |  |  |  |  | 9 | 11 | 13 |  |  |
| E |  |  |  |  |  | 11 | 13 |  |  |
| F |  |  |  |  |  |  | 13 |  |  |
| G |  |  | 13 | 13 | 13 | 13 | 13 |  |  |

1. Kotak merah : Mulai 3. Kotak abu : Halangan

2. Kotak biru : Tujuan 4. Kotak putih : Jalan

Pemilihan langkah dimulai dari kotak /*node* merah (A5) dengan tujuan akhir *node* biru (G2). Langkah dihitung dari *node* putih (jalan) yang dapat dilalui dan menghindari *node* abu (halangan). Perhitungan langkah dilakukan dengan menghitung nilai ℎ(𝑥) dan 𝑔(𝑥) pada tiap node. Nilai 𝑔(𝑥) merupakan biaya yang dibutuhkan dari *current node* untuk mencapai node berikutnya, sedangkan berikutnya ℎ(𝑥) merupakan biaya yang dibutuhkan dari *current node* ke *goal node* tanpa melihat *node* halangan. Dari *node* awal (A5), ada 2 jalan yang dapat dilalui yaitu A6 dan B5. Untuk menentukan *node* mana yang harus diambil, maka akan dihitung nilai *f(x)* terhadap *node* A6 dan B5. Jika jarak tiap *node* adalah 1 maka :

f(A6) = g(A6) + h(A6)

f(A6) = 1 + 10

f(A6) = 11

f(B5) = g(B5) + h(B5)

f(B5) = 1 + 8

f(B5) = 9

Kedua hasil perhitungan akan ditaruh pada *open list.* Dari hasil perhitungan di atas, dapat disimpulkan bahwa *node* B5 memiliki biaya yang lebih kecil daripada A6 sehingga *node* B5 lah yang diambil. *Node* A6 tetap berada pada *open list* sementara *node* permulaan (A5) dan *node* B5 akan ditaruh pada *closed list.* Dari *node* B5 hanya ada 1 jalan yaitu *node* C5 maka *node C5* secara otomatis akan dipindahkan ke *closed list. Node* C5 inilah yang disebut sebagai pewaris dari *node* B5. Perhitungan ini akan terus dilakukan sampai mencapai *node* tujuan (G2) atau terjadi *deadlock*.

Jika terjadi *deadlock* seperti pada *node* C1 di mana tidak ada jalan lagi, maka penelusuran akan kembali ke *node* yang terakhir (*node* D5) didaftarkan ke dalam *Open List* . Proses ini akan terus diulang hingga mencapai *node* tujuan. Dari seluruh hasil perhitungan maka diperoleh rute terpendek yaitu : A5 – B5 – C5 – D5 – D6 – E6 – E7 – F7 – G7 – G6 – G5 – G4 – G3 – G2.

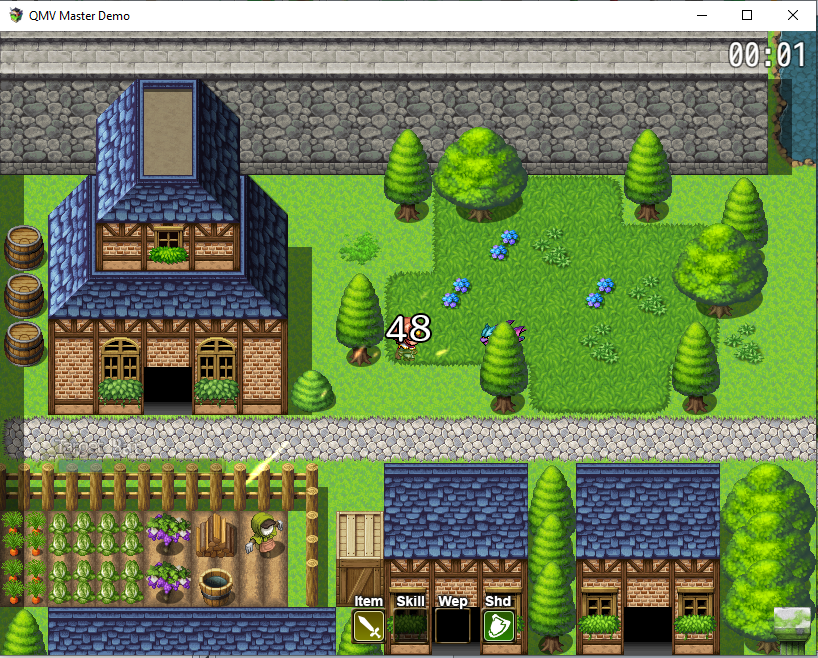
## 2.8 Game Engine RPG Maker MV

*RPG Maker MV* adalah sebuah gim *engine* buatan jepang yang memang khusus dibuat untuk menciptakan gimdengan *genre RPG* berbasis *turn base battle system.* Engine ini dibuat dengan menggunakan bahasa pemrograman *javascript* sehingga untuk membuat *gameplay* dengan *genre action* dan menerapkan algoritma untuk *pathfind* diperlukan pemahaman terhadap *javascript.* Setelah program tersebut dibuat, engine ini dapat memanggil fungsi dari program tersebut menggunakan *script command.* Meskipun demikian, engine ini mudah digunakan dan tidak membebani CPU karena engine ini diperuntukan untuk membuat gimdengan grafik 2D.

*RPG Maker MV* sekarang ini sudah mendukung *multiplatform distribution* seperti windows/EXE, Mac Os X/APP, Android/APK, IOS/IPA dan HTML5 untuk web browser gim*.* Pada engine ini sendiri sudah mendukung penggunaan *mouse* dan *touch screen* sehingga memudahkan pengembangan gimpada platform android dan web browser. Berbeda dengan engine lainnya yang mengharuskan pengembang untuk membuat dan menghubungkan gimbuatannya dengan *database, RPG Maker MV* sudah dilengkapi dengan *database* sehingga pengembang tidak perlu lagi membuat *database.*

Seperti yang sudah dituliskan di atas, engine ini lebih fokus pada *RPG* dengan *gameplay turn base battle system.* Namun salah satu kelebihan engine ini dibandingkan dengan engine lainnya adalah kemudahan dalam *level design.* Engine ini memiliki fitur tersendiri untuk membuat *level design* termasuk materialyang sudah disediakan dan dapat digunakan untuk mendesain *level* tersebut. Kelebihan lainnya dari engine ini adalah spesifikasi untuk mengembangkan gim *RPG.* Fitur yang ada pada engine ini sudah mampu untuk menciptakan gim *RPG* sederhana tanpa memerlukan *coding.* Fitur tersebut adalah *parameter character, party system, item drop, equipment,* dan lain sebagainya. Untuk menambahkan fitur – fitur yang tidak ada pada *engine,* pengembang dapat menggunakan *plugin.*

*Plugin* adalah sebuah program tambahan yang digunakan untuk menambah fungsionalitas program utama. Pengembang hanya memerlukan *coding* untuk membuat fitur yang tidak ada pada engine menggunakan ­*plugin* seperti penggunaan algoritma untuk *pathfind* atau *gameplay* dengan *genre action.* Banyak pengembang yang sudah membuat *plugin* untuk *engine* ini dan beberapa diantaranya bahkan ada yang gratis. Namun perlu diingat bahwa penggunaan *plugin* dalam RPG Maker MV sangat sensitif. Penggunaan *plugin* dibatasi dengan *plugin* buatan satu orang saja terutama jika *plugin* tersebut sangatlah kompleks. Jadi jika pengembang menggunakan 2 buah *plugin* buatan orang yang berbeda maka fungsi dari *plugin* tersebut akan berbenturan dan akan membuat gim mengalami *error.* Selain itu penggunaan *plugin* harus urut sesuai dengan ketentuan yang diberikan pembuat *plugin.*



##### **Gambar 2.3** Contoh Gim Yang Akan Dibuat



##### **Gambar 2.4 Gim Engine RPG Maker MV**

## *2.9 Flowchart*

*Flowchart* atau bagan alir adalah penjelasan yang lebih detail mengenai bagaimana jalannya setiap prosedur atau langkah suatu sistem dijalankan secara berurutan. Bagan alir dibuat untuk menggambarkan urutan dan tahapan dari penyelesaian masalah secara rapi, ringkas namun jelas dan terurai. Berikut adalah simbol – simbol yang digunakan dalam bagan alir.

###### Tabel 2.1 Simbol – simbol dalam bagan alir

|  |  |  |  |
| --- | --- | --- | --- |
| No | Simbol | Nama | Keterangan |
| 1 |  | Simbol arah aliran | Digunakan untuk menghubungkan antara simbol dengan simbol lain |
| 2 |  | Simbol terminal | Digunakan sebagai simbol untuk memulai (*start*) atau mengakhiri (*end*) jalannya suatu program |
| 3 |  | Simbol proses | Simbol yang menggambarkan proses atau tugas yang dilakukan oleh komputer |
| 4 |  | Simbol  keputusan | Simbol untuk memilih proses berdasarkan kondisi yang ada seperti ya atau tidak, berhasil atau gagal |
| 5 |  | Simbol keluar masuk | Menunjukkan proses keluar masuk yang terjadi dalam sistem tanpa bergantung pada jenis peralatannya |

## 2.10 *Unified Modeling Language (UML)*

*Unified Modeling Language* adalah suatu teknik atau metode untuk menggambarkan kinerja suatu sistem berorientasi objek menggunakan diagram berbentuk garis dan simbol. UML digunakan sebagai alat bantu dalam menjelasakan alur jalannya dari sistem yang dibuat. UML sendiri sudah dijadikan sebagai bahasa standar dalam pemodelan, visualisasi, cetak biru termasuk penulisan dokumentasi sistem aplikasi.

### 2.10.1 Use Case Diagram

*Use Case Diagram* merupakan diagram yang digunakan untuk menggambarkan interaksi antara aktor (*user)* dengan sistem. Bagan ini menggambarkan fungsi sistem yang dapat dilihat dari luar (eksternal*)* oleh aktor (sistem yang digunakan secara langsung oleh *user*) dan bukan bagaimana jalannya program dari dalam sistem (internal). *Use case* dibuat dengan tujuan untuk menggambarkan interaksi antara pengguna sistem dan sistem iut sendiri dalam bentuk bagan. Berikut adalah simbol – simbol dalam *use case diagram.*

###### Tabel 2.2 Simbol – simbol dalam *use case diagram*

|  |  |  |  |
| --- | --- | --- | --- |
| No | Simbol | Nama | Keterangan |
| 1 | Use Case | *Use case* | Menunjukkan fungsionalitas yang ada pada sistem sebagai unit yang saling berinteraksi antara sistem dengan aktor. Simbol ini biasanya diberi nama menggunakan kata kerja |
| 2 | Actor | Aktor | Menunjukkan pengguna, orang, sistem yang lain, atau alat yang berinteraksi dengan sistem utama. Simbol dinyatakan menggunakan kata benda |
| 3 |  | Asosiasi | Digunakan untuk menghubungkan simbol *use case* dan aktor yang saling berinteraksi dalam sistem |
| 4 | simbol ekstend | Extend | Menunjukkan jika suatu *use case* merupakan fungsionalitas tambahan dari *use case* lainnya bergantung dari kondisi yang terpenuhi |
| 5 | simbol include | Include | Menunjukkan suatu *use case* tambahan baru dapat dijalankan jika *use case* sebelumnya dijalankan terlebih dahulu sebagai syarat aktivasi *use case* tambahan tersebut |
| 6 |  | Generalisasi | Menunjukkan hubungan antara 2 buah *use case* dimana fungsi yang satu bersifat lebih umum dan yang lain bersifat khusus |

### 2.10.2 Activity Diagram

Diagram aktivitasadalah diagram yang menggambarkan aktivitas atau kinerja yang ada pada sistem. Diagram ini juga menggambarkan alur kontrol antara pengguna dengan sistem. Tujuan dari diagram aktivitas adalah untuk menggambarkan urutan aktivitas dalam sebuah proses antara pengguna dengan sistem pada gim agar lebih mudah dipahami. Berikut adalah simbol – simbol pada *activity diagram.*

###### Tabel 2.3 Simbol – simbol dalam *activity diagram*

|  |  |  |  |
| --- | --- | --- | --- |
| No | Simbol | Nama | Keterangan |
| 1 |  | *Initial Node* | Menunjukan titik awal dimulainya suatu proses pada sistem |
| 2 |  | *Final Node* | Menunjukkan titik akhir dari suatu proses pada sistem |
| 3 |  | Aktivitas | Menunjukkan aktivitas atau proses yang dilakukan oleh sistem sehingga menggunakan kata kerja |
| 4 |  | Percabangan | Menunjukkan pilihan keputusan pada percabangan jika terdapat lebih dari satu aktivitas |
| 5 |  | Garis penghubung | Digunakan untuk menghubungkan simbol yang satu dengan yang lainnya |

### 2.10.3 Sequence Diagram

Sequence diagram adalah diagram yang digunakan untuk menggambarkan kolaborasi dinamis antara suatu objek dengan objek lainnya secara runtut berdasarkan aliran waktu. Diagram ini digunakan untuk menunjukkan aliran pesan yang dikirim antara objek beserta dengan bentuk interaksinya. Berikut adalah simbol – simbol yang terdapat pada *sequence diagram.*

###### Tabel 2.4 Simbol – simbol pada *sequence diagram*

|  |  |  |  |
| --- | --- | --- | --- |
| No | Simbol | Nama | Keterangan |
| 1 | Actor | Aktor | Merepresentasikan entitas yang berada di luar sistem namun tetap saling berinteraksi dengan sistem itu sendiri. Entitas ini biasanya adalah pengguna sistem |
| 2 |  | Aktivasi | Menunjukkan suatu objek mulai aktif dan berinteraksi pada jalannya sistem. Setiap simbol masukan yang terhubung dengan aktivasi menunjukkan alur prosedur yang dijalankan oleh sistem |
| 3 |  | Masukan | Menggambarkan suatu objek mengirimkan pesan kepada objek lain yang ditunjuk oleh anak panah |
| 4 |  | Keluaran | Menunjukkan bahwa suatu objek mengirimkan kembalian kepada objek yang ditunjuk oleh anak panah sebagai respon |
| 5 | Objek : kelas | Objek | Menunjukkan objek yang aktiv dan berinteraksi dengan mengirim atau menerima pesan secara berurutan dari objek lainnya |