PERBANDINGAN ALGORITMA A-STAR, DIJKSTRA, DAN BFS UNTUK PATHFINDING NPC PADA GAME

Disusun Oleh :

Muhammad Bintang Nugraha (21081010071)

PROGRAM STUDI INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS PEMBANGUNAN NASIONAL "VETERAN"
JAWA TIMUR
2024

BAB I
PENDAHULUAN

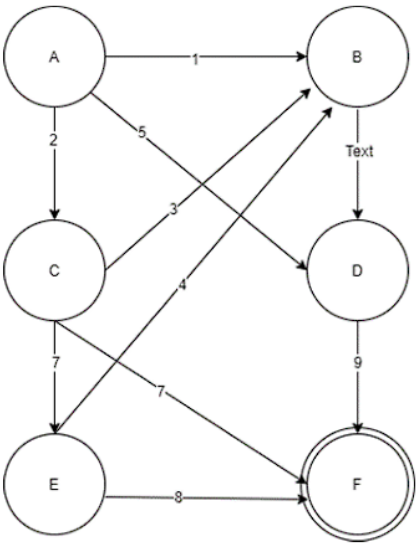
1.1. Latar Belakang

Video game adalah sebuah permainan elektronik yang menggabungkan teknologi, seni, dan hiburan. Permainan ini membutuhkan interaksi antara manusia dan antarmuka perangkat untuk dijalankan serta menghasilkan umpan balik visual (Bates, 2004). Setiap video game memiliki tujuan tertentu yang dapat dicapai oleh pemain, di mana tujuan tersebut bervariasi sesuai dengan jenis game yang dikembangkan. Secara umum, video game dirancang sebagai sarana hiburan yang menyenangkan dan tidak membosankan. Namun, dalam beberapa tahun terakhir, video game juga mulai dikembangkan untuk tujuan edukasi yang interaktif dan menarik. Jenis permainan yang dirancang khusus untuk mendidik sambil menghibur dikenal sebagai game edukasi.

Salah satu elemen penting dalam video game adalah Non-Player Character (NPC), yaitu karakter yang tidak dikendalikan oleh pemain, melainkan oleh komputer. NPC merupakan hasil pengembangan teknologi kecerdasan buatan (AI). Dengan AI, NPC dapat dirancang untuk berinteraksi dengan objek, pemain, maupun NPC lainnya. Dalam pergerakannya, NPC sering menggunakan metode pathfinding untuk berpindah dari satu posisi ke posisi lain. Beberapa algoritma yang digunakan untuk pathfinding antara lain Dijkstra, A Star (A*), dan Breadth-First Search (BFS).

Algoritma Dijkstra digunakan untuk menemukan jalur terpendek dari satu titik ke titik lainnya dalam sebuah graf berbobot. Metode ini bekerja dengan memilih jalur berdasarkan bobot terkecil dari satu node ke node lainnya, tetapi hanya mampu menentukan satu rute saja. Sementara itu, BFS bekerja dengan melakukan pencarian menyeluruh dari node awal ke node tujuan. Algoritma ini menjelajahi setiap node yang bertetangga dengan node awal, kemudian menyimpan node yang sudah dikunjungi agar tidak diperiksa ulang.

A Star (A*) merupakan algoritma pathfinding yang menggunakan fungsi heuristik untuk menentukan jalur terpendek. Algoritma ini menghitung total bobot dari node awal ke tujuan menggunakan heuristik, lalu memilih jalur terbaik berdasarkan perhitungan tersebut. Meski algoritma-algoritma ini efektif dalam menentukan jalur terpendek, proses pencarian yang panjang sering menjadi tantangan, terutama dalam graf yang kompleks.



Gambar 1.1 Graph Berbobot

Gambar 1.1 menjelaskan cara kerja algoritma Dijkstra dan BFS. Dalam pencarian jalur dari node awal A ke node tujuan F, algoritma Dijkstra akan memeriksa semua node pada jalur tersebut satu per satu. Sementara itu, BFS akan memeriksa setiap node yang dikunjungi dan menyimpannya dalam sebuah tabel, di mana node yang sudah tercatat dalam tabel tidak akan diperiksa kembali. Kedua metode ini memiliki kelemahan yang sama, yaitu memeriksa semua node tetangga pada jalur, yang menyebabkan proses pencarian memakan waktu lama. Berbeda dengan Dijkstra dan BFS, algoritma A* menggunakan fungsi heuristik untuk membantu proses pencarian jalur dari node awal ke node tujuan. Fungsi heuristik ini merupakan karakteristik dari daftar tertutup (closed set), yang mencatat area yang sudah dievaluasi, serta daftar terbuka (open set), yang mencatat area berdekatan yang belum dievaluasi.

Algoritma A* bekerja dengan menghitung jarak yang sudah ditempuh dari node awal ke posisi saat ini (current state) serta jarak yang diperkirakan menuju node tujuan. Penelitian oleh Peter Hart, Nils Nilsson, dan Bertram (1968) menunjukkan bahwa A* lebih efisien dibandingkan metode lain karena memprioritaskan node yang memiliki nilai terbaik berdasarkan perhitungan heuristik. Hal ini membuat A* berbeda dari Dijkstra, yang memeriksa semua jalur tanpa mempertimbangkan prioritas.

Berdasarkan latar belakang tersebut, penulis tertarik untuk membuat simulasi yang membandingkan performa algoritma A*, Dijkstra, dan BFS dalam melakukan pathfinding untuk NPC pada game. Simulasi ini diharapkan dapat memberikan pemahaman lebih dalam mengenai efektivitas dan efisiensi ketiga algoritma dalam menentukan jalur optimal dalam lingkungan dinamis pada game.

1.2. Rumusan Masalah

Berdasarkan latar belakang masalah diatas, maka identifikasi masalah dari penelitian ini adalah bagaimana performa algoritma A-Star, Dijkstra, dan BFS dibandingkan dalam konteks pathfinding NPC pada game dan apa saja kelebihan dan kekurangan masing-masing algoritma dalam lingkungan game yang dinamis dengan tingkat kompleksitas yang berbeda.

1.3. Tujuan Penelitian

Tujuan dari penelitian ini adalah untuk mengetahui performa algoritma A-Star, Dijkstra, dan BFS dalam hal pathfinding untuk NPC pada game dengan lingkungan yang dinamis..

1.4. Manfaat Penelitian

Manfaat yang dapat diambil dari penelitian ini adalah penulis dapat memberikan kontribusi untuk permasalahan path finding oleh NPC dalam lingkungan dinamis untuk game – game kedepannya, dengan mengetahui performa dari masing-masing algoritma A-Star, Dijkstra, dan BFS.

BAB II
TINJAUN PUSTAKA

2.1. Penelitian Terdahulu

Pada bagian penelitian terdahulu ini akan membahas beberapa referensi pustaka dan penelitian yang sebelumnya telah dilakukan sebagai pendukung dan menambah teori-teori dalam penelitian yang dilakukan. Berikut beberapa penelitian terdahulu yang digunakan penulis sebagai acuan dalam penelitian :

1. Berdasarkan penelitian yang berjudul “Penerapan Algoritma A Star (A*) Untuk Penentuan Jarak Terdekat Wisata Kuliner Di Kota Bandar Lampung” yang dilakukan oleh Sandy Purnama, Ayu megawaty, Yusra Fernando dari informatika, Universitas Teknokrat Indonesia, Bandar Lampung. Permasalahan dari penelitian ini adalah Penentuan rute terdekat sudah banyak diterapkan di berbagai macam aplikasi navigasi. Proses perhitungan rute terdekat adalah proses mencari biaya terkecil suatu rute dari node awal ke node tujuan dalam sebuah jaringan. Pada proses perhitungan rute terdekat terdapat dua macam proses yaitu proses pemberian label dan proses pemeriksaan node. Penentuan jalur terdekat dapat dilakukan menggunakan algoritma seperti Dijkstra, A Star (A*), Floyd Warshall dan lain-lain. Para peneliti mengambil algoritma A* untuk

menentukan rute terdekat. Algoritma A* merupakan salah satu algoritma pencarian rute yang optimal dan lengkap. Optimal berarti rute yang dihasilkan adalah rute yang paling baik dan lengkap berarti algoritma tersebut dapat mencapai tujuan yang diharapkan. Dalam penerapannya, algoritma A* menggunakan jarak sebagai proses kalkulasi nilai terbaik.

2. Berdasarkan penelitian yang berjudul "Penerapan A Star (A*) Menggunakan Graph Untuk Menghitung Jarak Terpendek" yang dilakukan oleh Ida Bagus Gede Wahyu Antara Dalem dari Program Studi Ilmu Komputer, Pasca Sarjana Universitas Pendidikan Ganesha, Singaraja Bali. Penelitian ini bertujuan untuk mempelajari cara kerja algoritma A* dalam mencari jarak tercepat, yang disimulasikan seperti kondisi ketika seorang mencari rute dalam keadaan jalanan macet. Simulasi ini memberikan gambaran yang lebih realistis terhadap perilaku algoritma A* dalam pencarian jarak rute terpendek. Berdasarkan hasil simulasi algoritma A* pada penelitian ini dapat disimpulkan bahwa penentuan rute terbaik dapat dilakukan dengan Algoritma A*. Peneliti merekomendasikan untuk pengembangan lebih lanjut disarankan menggunakan algoritma lain selain algoritma A* untuk menentukan jalur (rute) yang terbaik dan an juga dapat membandingkan algoritma lain tersebut apakah lebih baik dalam penentuan jalur tercepat.
3. Berdasarkan penelitian yang berjudul "Implementasi Algoritma Dijkstra dalam Penentuan Jalur Terpendek Di Yogyakarta Menggunakan GPS Dan Qt Geolocation" yang dilakukan oleh Blasius Neri Puspika, Antonius Rachmat C., dan Erick Kurniawan Dari Fakultas Teknik Informatika, Universitas Kristen Duta Wacana, Yogyakarta. Penelitian ini dilakukan karena dengan perkembangannya teknologi yang sekarang peta sudah tidak berbentuk gambar, tetapi dalam bentuk digital. Contohnya adalah Google Maps dan GPS (Global Position System). Meskipun dengan adanya peta digital atau GPS untuk mengetahui posisi pengguna, pengguna tidak dapat menentukan jalur terpendek untuk mencapai lokasi tujuan. Salah satu cara untuk menentukan jalur terpendek adalah untuk menginterpretasikan peta ke dalam suatu graph. Ada beberapa metode untuk menentukan jalur dalam sebuah graph, salah satunya adalah algoritma Dijkstra. Dari permasalahan ini para peneliti menggunakan teknologi GPS dan QT Library Geolocation dalam menentukan suatu lokasi pada perangkat bergerak telepon seluler dengan sistem operasi Symbian untuk menentukan jalur terpendek menuju ke lokasi yang ditentukan menggunakan Algoritma Dijkstra. Tujuan dari penelitian ini adalah untuk mencari jalur terpendek dari suatu lokasi ke lokasi tujuan pada peta. Para

peneliti membuat sebuah rancangan sistem pencari rute terdekat dalam bentuk sebuah aplikasi. Berdasarkan hasil penelitian ini perhitungan jalur terpendek sistem dengan jarak aslinya memiliki tingkat keakuratan 98,69520% yang berarti jarak hasil perhitungan sistem hampir mendekati jarak aslinya dan penggunaan algoritma Dijkstra dalam menentukan jalur terpendek dari lokasi pengguna menuju lokasi yang ditentukan mampu menghasilkan solusi jalur terpendek dengan tepat

4. Berdasarkan penelitian yang berjudul "Penerapan Algoritma A Star (A*) Dalam Penyelesaian Rute Terpendek Pada Pendistribusian Barang" yang dilakukan oleh Rosita Ayu Nugraeni dan Mulyono Rochmad dari jurusan matematika, Universitas Negeri Semarang. Fokus dari penelitian ini adalah mengkaji sebuah permasalahan pencarian solusi untuk masalah penentuan rute terpendek pendistribusian barang. Tujuan penelitian ini adalah untuk mengetahui dasar-dasar algoritma A*, untuk meneliti penentuan rute terpendek pendistribusian barang dengan algoritma A*, dan untuk meneliti penentuan rute terpendek pendistribusian barang diaplikasikan dengan berbantuan software. Pengambilan data dilakukan dengan cara mendokumentasikan data di kantor CV Mitra Adi Busana Semarang, selanjutnya dilakukan pencarian jarak dengan bantuan Google Maps. Analisa data dilakukan dengan menggunakan algoritma A* yang kemudian diaplikasikan dengan software Visual Basic .Net. Dari analisa yang dilakukan dengan cara manual maupun berbantuan software, diperoleh rute terpendek pendistribusian barang yaitu, CV Mitra Adi Busana - Hotel Gumaya - Hotel Merbabu - Hotel Novotel - Hotel Ibis - Hotel Grand Saraswati - Hotel Royal Phoenix - Hotel Neo Candi - Hotel Plaza dengan panjang rute terpendek 19.595 meter. Dari hasil analisa dengan algoritma A* dan bantuan software diperoleh rute terpendek yang sama dalam pendistribusian barang untuk mencapai seluruh lokasi pendistribusian barang dengan rute terpendek yang minimal. Dalam percobaan ini peneliti membuat sebuah graph menggambarkan lokasi – lokasi hotel. Berdasarkan hasil analisis yang dilakukan, diperoleh rute terpendek pendistribusian barang yaitu, CV Mitra Adi Busana - Hotel Gumaya - Hotel Merbabu - Hotel Novotel - Hotel Ibis - Hotel Grand Saraswati - Hotel Royal Phoenix - Hotel Neo Candi - Hotel Plaza dengan panjang rute terpendek 19595 meter. Hasil ini sesuai dengan perhitungan bantuan software Visual Basic.Net. Berdasarkan penelitian di atas, dapat disimpulkan bahwa algoritma A* dapat digunakan untuk menyelesaikan masalah pencarian rute terpendek pendistribusian barang CV Mitra Adi Busana Semarang.

5. Berdasarkan penelitian yang berjudul “Penerapan Algoritma A Star (A*) Pada Game Edukasi “The Maze Island” Berbasis Android” yang dilakukan oleh Agus Pamungkas, Eka Puji Widiyanto, dan Renni Angreani, dari Jurusan Teknik Informatika, STMIK GI MDP, Palembang. Peneliti melakukan percobaan ini karena game edukasi memiliki kelebihan dibandingkan dengan metode pembelajaran konvensional karena cara pembelajarannya disajikan dengan visualisasi bergerak yang menarik. Namun, game edukasi jarang menerapkan algoritma dalam penyelesaiannya. Alasan inilah yang membuat penulis ingin mencoba untuk menerapkan algoritma A Star (A*) pada game edukasi The Maze Island berbasis android. Game edukasi ini termasuk dalam game labirin dimana pemain diharuskan untuk mencari jalan keluar dengan rute terpendek. Algoritma A* memberikan solusi terbaik untuk memecahkan masalah ini. Tujuan utama dari game ini adalah untuk menerapkan algoritma A* dalam memberikan hasil pencarian untuk menemukan pintu keluar dengan rute terpendek. Game ini juga diharapkan dapat menjadi salah satu media pembelajaran untuk menambah wawasan ilmu pengetahuan dan matematika. Dan game ini dikembangkan pada platform android yang memungkinkan pengguna untuk bermain di mana pun dengan menggunakan smartphone. Berdasarkan hasil uji coba penelitian ini adalah Algoritma A* lebih baik daripada algoritma lainnya karena algoritma ini memiliki nilai heuristic sebagai nilai pembanding sehingga dapat mencari solusi yang terbaik dari permasalahan yang ada jika solusi itu memang ada.

2.2. Pengertian game

Permainan atau dalam bahasa inggris biasa disebut dengan game merupakan sebuah permainan elektronik yang berasal dari penggabungan 3 unsur yaitu teknologi, seni, dan hiburan. Game memerlukan interaksi antara manusia dengan antarmuka perangkat agar dapat dijalankan dan menghasilkan umpan balik visual. Game dapat berbentuk 2 dimensi (2D) maupun 3 dimensi (3D) dimana game memiliki sebuah tujuan yang dapat dicapai atau dimenangkan oleh pengguna, tujuan dari tiap game berbeda-beda sesuai dengan jenis yang dikembangkannya.

Pada umumnya, game memiliki tujuan sebagai sarana hiburan yang menyenangkan dan tidak membuat bosan ketika digunakan oleh pengguna, namun beberapa tahun belakang game mulai dikembangkan dengan tujuan memberikan nilai edukasi kepada pengguna dengan cara yang menyenangkan dan tidak membosankan. Terdapat beberapa pengertian game atau permainan yang dijelaskan menurut para ahli yaitu, menurut Clark C. Abt game adalah suatu

kegiatan atau aktivitas yang melibatkan keputusan pemain atau pengguna yang berusaha untuk mencapai tujuan dengan dibatasi oleh konteks tertentu, Contohnya seperti dibatasi oleh suatu peraturan.

Jika menurut Kattie dan Eric game adalah suatu sistem dimana pemain atau pengguna terlibat dalam konflik buatan yang didalamnya ditentukan oleh peraturan dan akan menghasilkan hasil yang terukur. Sedangkan menurut Greg Costikyan game merupakan suatu bentuk karya seni dimana pengguna yang disebut dengan pemain membuat sebuah keputusan untuk mengelola sumber daya yang dimilikinya melalui objek di dalam game demi tercapainya tujuan.

Beberapa para ahli dari Indonesia juga memberikan pengertian game atau permainan seperti menurut Dawang Muchtar yang menjelaskan game adalah sesuatu hal yang dapat dimainkan oleh pemain dengan aturan-aturan tertentu sehingga terdapat pemenang dan seorang yang kalah, game biasanya berada dalam konteks hiburan atau tidak serius dengan tujuan refreshing. Sedangkan menurut Agustinus Nilwan yang menjelaskan pada buku yang ia tulis dengan judul Pemrograman Animasi dan Game Profesional terbitan Elex Media Komputindo, menjelaskan bahwa game merupakan permainan komputer yang dibuat dengan teknik dan metode animasi. Jika ingin mendalami penggunaan dari animasi haruslah memahami pembuatan game terlebih dahulu. Atau jika ingin membuat sebuah game, maka haruslah memahami teknik dan metode animasi terlebih dahulu, sebab kedua hal tersebut saling berkaitan.

2.3. Sejarah dan Perkembangan Game

Pada akhir tahun 1940-an game pertama kali ditemukan, game ditemukan oleh Edward U. Condon dimana ia membuat dan mendesain komputer yang dapat memainkan permainan tradisional Nim. Pada tahun 1952, seorang profesor dari Universitas Cambridge yang bernama A. S. Gouglas membuat dan mengembangkan game yang lebih ramah saat digunakan oleh pengguna, game tersebut berjudul OXO atau saat ini biasa disebut dengan tic tac toe, game ini awalnya dibuat untuk mendemonstrasikan tesisnya tentang interaksi manusia dan komputer. Pada saat itu game ini dimainkan pada komputer besar yang menggunakan CRT display (Rio Caesar, 2015).

Game terus berkembang seiring dengan berjalannya waktu, pada tahun 1958 sebuah game yang berjudul Tennis For Two dibuat dan didesain oleh William Higin Botham yang dapat dimainkan menggunakan komputer analog yang terhubung dengan layar oscilloscope. Game Tennis For Two dibuat khusus untuk hari pengunjung tahunan yang ada di Brookhaven

National Laboratory, New York. Tidak membutuhkan waktu yang lama, game Tennis For Two menarik banyak sekali minat hingga mendapatkan dukungan oleh pengunjung yang saat itu hadir. Kesuksesan yang didapatkan oleh game Tennis For Two ini menginspirasi sekelompok mahasiswa Massachusetts Institute of Technology (MIT) yang berhasil membuat dan mendesain game berjudul Spacewar pada tahun 1961. Spacewar dibuat dalam komputer mainframe DEC PDP-1. Namun karena pada tahun itu komputer masih sangat sulit untuk diakses oleh publik, game Spacewar pun tidak dirilis secara umum kepada publik.

Pada tahun 1967 game mulai banyak dinikmati oleh masyarakat umum, hal ini dikarenakan Ralph Baer bersama dengan Bob Tremblay menciptakan sebuah konsol game untuk pertama kali. Konsol game tersebut dijual oleh Baer ke perusahaan Magnavox, dan kemudian konsol game tersebut secara resmi dijual kepada publik oleh perusahaan Magnavox. Konsol game tersebut Bernama Odyssey. Pada tahun 1972 Nolan Bushnell bersama dengan Ted Dabney memutuskan untuk membuat perusahaan game bernama Atari setelah melihat demonstrasi konsol Odyssey milik Magnavox. Atari berhasil menciptakan konsol rumahan buatannya sendiri yang dapat menggunakan joystick dan cartridge, konsol tersebut diberi nama Atari 2600. Atari 2600 memiliki kelebihan yang lebih menarik pembeli yaitu memiliki game multi-warna. Atari juga berhasil membuat dan mendesain game yang saat itu sangat digemari dan diminati oleh banyak masyarakat, game tersebut bernama Pong.

Sejak saat itu, game terus-menerus berkembang seiring dengan berkembangnya zaman dan hal ini membuat game semakin banyak diminati oleh masyarakat. Perkembangan game juga diikuti oleh berkembangnya desain grafis yang terdapat di dalam game tersebut. Sebagai contoh game dahulu hanya dapat dimainkan menggunakan komputer ataupun konsol dengan desain grafis yang monoton, kurang berwarna, dan sederhana, namun sekarang game juga dapat dimainkan menggunakan perangkat mobile yang memiliki desain grafis secara 3 dimensi (3D) dengan desain realistis atau menyerupai kehidupan nyata yang mampu menggambarkan manusia, binatang, tumbuhan, bangunan, hingga mampu menggambarkan lingkungan sekitar secara detail. Game pada perangkat mobile banyak digemari karena memiliki sifat yang fleksibel atau dapat dimainkan kapan dan dimanapun ketika pengguna menginginkannya.

Game pada perangkat mobile pertama kali diciptakan oleh seorang desainer game asal Rusia yang bernama Alexey Pajitnov pada tahun 1994 pada perangkat mobile Hagenuk MT-2000 sebuah ponsel buatan Denmark. Game tersebut diberi nama Tetris, sebuah game yang menjadi asal mula game pada perangkat mobile, hingga sampai saat ini muncul banyak sekali game perangkat mobile yang terus berkembang, baik dari segi hiburan, dari segi desain grafis maupun dari segi edukasi.

2.4. Jenis-Jenis Game

Game sendiri memiliki jenis-jenis yang dapat dibedakan berdasarkan beberapa kategori yaitu berdasarkan platform atau perangkat yang digunakan untuk memainkan game dan juga berdasarkan genre.

2.4.1 Jenis Game Berdasarkan Platform

Platform atau sering disebut dengan perangkat yang dapat digunakan untuk memainkan game merupakan sebuah kombinasi antara perangkat keras dan perangkat lunak. Berdasarkan platform, game dibedakan menjadi beberapa jenis yaitu diantaranya (Nur Abdilah, 2023)

1. Arcade Game

Arcade Game adalah sebuah game yang dimainkan menggunakan mesin elektronik yang dilengkapi dengan tombol-tombol kontrolernya. Mesin game arcade pada umumnya di desain hanya untuk jenis game tertentu saja, karena pada setiap mesin memiliki kontroler yang berbeda beda. Sebagai contoh untuk game balapan maka mesin game arcade akan dilengkapi dengan kontroler sensor pijakan dan stir mobil. Di Indonesia arcade game biasa disebut dengan ding-dong.

2. Console Game

Console Game merupakan sebuah game yang dapat dimainkan menggunakan console tertentu. Console game dilengkapi dengan kontroler yang lebih dikenal dengan sebutan joystick. Contoh console game yang cukup terkenal di kalangan masyarakat yaitu PlayStation yang di produksi oleh perusahaan bernama Sony dan juga Nintendo wii yang di produksi oleh perusahaan Nintendo.

3. Handheld Game

Handheld Game merupakan sebuah game yang dapat dimainkan menggunakan console khusus. Sama halnya dengan console game, namun yang menjadi pembeda adalah console dari handheld game dapat dibawa dan digunakan dimana saja atau biasa disebut dengan portable console. Handheld game cukup digemari oleh masyarakat, contoh dari handheld game yang banyak digemari ialah PlayStation Portable atau dikenal dengan sebutan PSP yang di produksi oleh perusahaan Sony.

4. PC Game

PC Game atau singkatan dari Personal Computer Game merupakan sebuah game yang dapat dimainkan menggunakan perangkat komputer pribadi dengan sistem operasi Windows, Linux, ataupun MacOS. PC game tidak dilengkapi dengan kontroler khusus sehingga pemain dapat memainkan PC game menggunakan keyboard dan mouse.

5. Mobile Game

Mobile Game merupakan sebuah game yang dapat dimainkan menggunakan Mobile Phone atau disebut juga ponsel. Mobile game banyak digemari oleh masyarakat karena memiliki sifat yang fleksibel atau dapat digunakan kapan dan dimana saja, sama seperti Handheld game namun yang menjadi pembeda adalah daya atau baterai dari handheld game lebih boros dan tidak tahan lama.

2.4.2 Jenis Game Berdasarkan Genre

Genre atau dalam bahasa Indonesia memiliki arti aliran merupakan suatu cara yang digunakan untuk membedakan game berdasarkan jalan cerita dan game flow di dalam sebuah game. Berdasarkan genre, game dibedakan menjadi beberapa jenis yaitu diantaranya (Deny Prasetia Hermawan, 2017)

1. Action

Action menghadirkan unsur aksi atau pertarungan di sepanjang game dimainkan. Genre action ini menjadi salah satu genre yang paling terkenal sejak kemunculannya. Dalam game action pemain harus dapat mengkombinasikan refleks, akurasi, dan waktu (timing) yang ia miliki untuk menyelesaikan sebuah tantangan pada game action.

2. Adventure

Adventure merupakan sebuah genre game yang menyajikan sebuah petualangan dalam dunia game kepada pemain. Dalam genre adventure pemain dapat mengeksplorasi dunia game secara bebas dan tidak ada batasan. Ciri khas dari genre adventure ini adalah pemain dapat menjelajah dan membuka tempat baru di dunia game, mencari dan mengoleksi barang-barang yang berguna, dan menyelesaikan misi untuk mendapatkan hadiah.

3. Role Playing

Role playing merupakan sebuah genre game yang mirip dengan genre adventure, sebab pemain juga dapat mengeksplorasi atau menjelajah di dalam dunia game. Namun yang menjadi pembeda genre role playing ialah pemain dapat mengembangkan karakter yang ada di dalam game sesuai dengan keinginannya. Bahkan beberapa game dengan genre ini, dapat membuat pemain menentukan hasil akhir dari game. Game dengan genre role playing lebih dikenal dengan sebutan RPG.

4. Strategy

Strategy merupakan sebuah genre game yang menyajikan unsur strategi di dalamnya. Pemain diharuskan untuk berpikir dan merencanakan tindakan yang akan dilakukan. Pemain juga diharuskan untuk mengelola sumber daya yang telah disediakan secara baik. Proses perencanaan dan pelaksanaan strategi menjadi salah satu kunci yang dapat memenangkan game. Game dengan genre strategy ini membutuhkan waktu yang cukup lama untuk dimainkan karena membutuhkan waktu dalam merancang dan melaksanakan sebuah strategi.

5. Platformer

Platformer adalah sebuah genre game yang menyajikan dunia game dalam bentuk level. Pada genre platformer selalu memiliki garis start dimana pemain memulai game dan garis finish untuk menyelesaikan game. Genre platformer mengharuskan pemain untuk menggerakan karakter baik itu berjalan maupun melompat dari satu titik ke titik yang lain dengan menghindari rintangan-rintangan yang ada. Kebanyakan dari game dengan genre platformer memiliki jenis 2 Dimensi (2D). Salah satu contoh game dengan genre platformer yang terkenal yaitu game Mario Bros.

6. Racing

Racing merupakan sebuah genre game yang memberikan pengalaman menjadi pengemudi kendaraan di sebuah ajang balapan kepada pemain. Game dengan genre ini mengharuskan pemain menjadi yang tercepat dalam balapan agar memenangkan permainan. Genre racing cukup diminati oleh masyarakat karena memiliki permainan yang mudah dan menantang.

7. Shooter

Shooter adalah sebuah genre game yang menyajikan unsur aksi atau laga dalam bentuk tembak-menembak. Genre shooter memberikan pemain pengalaman untuk mengalahkan semua musuh dengan cara menembak dengan senjata tertentu. Game dengan genre shooter memiliki 2 kategori yaitu first person shooter atau biasa dikenal dengan FPS merupakan game shooter yang menyajikan permainan dalam pandangan orang kesatu dan juga third person shooter atau biasa dikenal dengan sebutan TPS merupakan game shooter yang menyajikan permainan dalam pandangan orang ketiga.

8. Board Game

Board game adalah sebuah genre game yang memberikan pengalaman bagi pemain untuk memainkan permainan papan dalam bentuk digital. Game dengan genre board game banyak digemari oleh masyarakat karena memiliki permainan yang tidak rumit dan dapat dimainkan bersama-sama dengan pemain lain. Contoh game dengan genre board game yang cukup dikenal oleh masyarakat yaitu catur dan monopoli.

9. Puzzle

Puzzle merupakan sebuah genre game yang memiliki misteri atau teka-teki dimana pemain diharuskan untuk memecahkan misteri atau teka teki tersebut untuk memenangkan permainan. Genre puzzle mampu mengembangkan dan mengasah kemampuan pemecahan masalah pemain dengan menyajikan teka-teki yang rumit.

10. Sports

Sports merupakan sebuah genre game yang menyajikan permainan dengan tema olahraga. Game dengan genre sports biasanya dibuat mirip dengan olahraga yang ada di dunia baik dari cara memainkannya maupun aturan-aturan yang ada, sehingga pemain dapat merasakan keseruan yang sama ketika memainkan game dengan genre sports. Banyak sekali olahraga yang telah dibuatkan game, contohnya yaitu sepak bola dan basket yang cukup digemari oleh masyarakat.

11. Simulation

Simulation adalah sebuah genre game yang dirancang untuk menggambarkan atau menirukan aktivitas di dunia nyata secara mirip di dalam dunia game. Game dengan genre simulation memiliki berbagai konsep dan jalan permainan, jalan permainan dari genre simulation akan disesuaikan dengan aktifitas atau gambaran apa

yang diambil di dalam dunia game. Contohnya dalam game simulation yang membangun dan mengembangkan suatu negara maka pemain akan diharuskan untuk mengembangkan suatu negara sampai memenuhi target untuk memenangkan game. Dalam genre simulation keputusan pemain akan mempengaruhi jalan permainan tersebut, sehingga dibutuhkan kemampuan untuk berpikir dalam mencapai target yang telah ditentukan. Genre simulation juga biasa disebut dengan simulasi.

2.5. Game Engine

Game engine merupakan sebuah software atau perangkat lunak yang digunakan untuk pembuatan dan pengembangan suatu game. Game engine memiliki beberapa fungsi yang dapat mempermudah dan mempercepat proses pembuatan game, beberapa fungsi tersebut yaitu rendering, scripting, audio setup, physic engine, networking, dan masih banyak lagi fungsi dari game engine yang dapat menunjang dalam pembuatan game (Grivin, 2015).

Game engine pada umumnya memiliki berbagai jenis dan ditujukan untuk berbagai kemampuan pemrograman mulai dari game engine yang mudah digunakan oleh pemula hingga game engine yang digunakan oleh developer professional. Game engine harus dipilih sesuai dengan skala game yang sedang dikembangkan, karena pada dasarnya setiap game engine memiliki kelebihan dan kekurangannya masing-masing. Contoh seperti game engine Unity dan Unreal Engine yang cukup terkenal di dunia game developer.

Game engine Unity memberikan berbagai fitur dan fungsi yang dapat memudahkan seorang pemula dalam membuat dan mengembangkan game. Unity juga menyediakan banyak sekali asset atau desain yang dapat digunakan dalam pembuatan dan pengembangan game secara gratis. Sedangkan game engine Unreal Engine dirasa kurang sesuai digunakan oleh seorang pemula. Meskipun Unreal Engine juga memberikan berbagai fitur dan fungsi yang memudahkan pengguna namun, cara penggunaan Unreal Engine dianggap terlalu sulit untuk pemula. Tetapi ketika digunakan oleh developer professional maka, Unreal Engine merupakan game engine yang sangat memudahkan dalam pembuatan dan pengembangan game.

2.6. Unity

Unity merupakan salah satu game engine yang sangat terkenal dan banyak digunakan oleh developer dalam membuat dan mengembangkan game. Unity dipenuhi perpaduan dengan aplikasi professional, meskipun penuh perpaduan dengan aplikasi professional unity masih sangat ramah untuk digunakan oleh pemula dalam pembuatan game. Hal ini disebabkan karena unity di desain untuk mengembangkan game dengan berbagai platform atau multi platform.

Unity dapat digunakan dalam pembuatan dan pengembangan game 2 dimensi (2D) maupun 3 dimensi (3D). Unity memiliki editor dengan user interface yang sederhana dan mudah untuk dipahami oleh pemula. Grafis pada unity juga dibuat dengan grafis tingkat tinggi untuk OpenGL dan directX.

Unity dapat menghasilkan game untuk Windows, Mac, Android, PlayStation, dan beberapa platform game lainnya. Dalam hal importing file atau memasukan berkas ke dalam editor, unity mendukung semua format file baik dalam format gambar atau model 3 dimensi, format video, format teks, format audio, maupun format animasi. Unity dapat beroperasi pada sistem operasi windows dan Mac OS, unity juga cocok digunakan dengan versi 64-bit.

Selain digunakan untuk membuat dan mengembangkan game, unity juga dapat digunakan untuk membuat realtime animasi 3 dimensi (3D) dan visualisasi arsitektur. Unity dilengkapi dengan berbagai fitur yang dapat mempermudah developer dalam membuat dan mengembangkan game. Selain dilengkapi dengan fitur yang mudah digunakan, unity juga dilengkapi dengan sistem scripting atau penulisan kode yang cukup sederhana dan mudah untuk dipelajari. Oleh karena itu unity game engine merupakan salah satu game engine yang sangat cocok untuk digunakan oleh pemula maupun professional.

2.6.1. Sejarah Unity dan Perkembangannya

Unity adalah sebuah game engine yang berguna dalam membuat media interaktif, khususnya dalam pembuatan game. Sebelum menjadi terkenal dan banyak digunakan oleh developer, dalam masa perkembangannya unity mengalami beberapa tahap untuk menjadi game engine yang dikenal dan digunakan oleh banyak orang di dunia game.

Unity Technologies didirikan pada tanggal 2 Agustus 2004 tepatnya di Copenhagen, Denmark oleh David Helgason (CEO) bersama dengan Nicholas Francis (CCO) dan juga Joachim Ante (CTO). Mereka bertiga memiliki mimpi yang sama yaitu untuk menciptakan sebuah mesin pembuat game yang dapat dikembangkan di rumah dan dapat digunakan oleh semua orang dengan harga yang terjangkau, dimana pada saat itu mesin pembuat game sangatlah mahal dan tidak banyak dikenal (Andi Taru, 2020).

Pada bulan Maret tahun 2005, unity diluncurkan oleh ketiga pendirinya sebagai pra-rilis dengan game yang berjudul GooBall sebuah game yang didesain khusus untuk Apple Macintosh. 3 bulan setelah David Helgason dan teman-teman melakukan pra-rilis, unity diluncurkan atau diumumkan secara resmi sebagai aplikasi atau software yang bersifat komersial tepatnya pada bulan Juni tahun 2005. Tidak lama bagi unity

untuk menjadi nominasi dalam Apple Design Award dalam kategori “Best OS X Graphics” yaitu pada tahun 2006, satu tahun setelah unity diluncurkan secara resmi kepada publik.

Sejak unity dirilis secara resmi kepada publik yaitu pada versi 1.0.1, banyak sekali update atau pembaharuan fitur dan sistem untuk meningkatkan dan menunjang performa penggunaan unity. Seperti halnya yang terdapat pada update versi 1.1.1 dimana developer sudah bisa membuat sebuah game yang dimainkan menggunakan windows dan web browser. Pada update 2.0.1 unity juga mendapatkan pembaharuan dalam fitur penambahan bayangan dan dalam pembuatan sistem terrain. Hal ini bertujuan agar unity lebih mudah digunakan dalam membuat dan mengembangkan game, versi 2.0.1 ini dirilis pada tahun 2008.

Pada September tahun 2010, unity merilis versi 3.0.1 yang memperbarui kualitas grafik dari editor sehingga terlihat lebih realistis. Pada tahun 2015 juga unity merilis versi 5.0.1 yang menambahkan fitur baru yaitu pencahayaan dan efek video pada editor. Hingga pada saat ini unity tercatat terus memperbarui fitur dan pelayanannya dimana pada saat ini unity memiliki versi 2022 yang sangat memudahkan developer dalam membuat dan mengembangkan game. Unity akan terus berkembang dan menyesuaikan fitur-fiturnya seiring dengan berkembangnya zaman.

2.6.1. Fitur Dalam Unity

Unity memiliki berbagai fitur yang dapat digunakan oleh developer dalam membuat dan mengembangkan game. Fitur-fitur yang dimiliki oleh unity adalah sebagai berikut.

1. Rendering

Rendering merupakan sebuah proses penyempurnaan suatu bentuk model 2 dimensi ataupun 3 dimensi di dalam game agar menjadi lebih detail dan halus. Unity mampu memproses asset dengan format desain 3ds Max, Maya, Blender, Adobe Photoshop, ZBrush, Cinema 4D, dan modo. Aset-aset tersebut dapat ditambahkan ke dalam game project melalui unity dan dapat diatur melalui graphical user interface atau biasa disebut dengan GUI.

Unity dapat menggunakan beberapa graphics engine yaitu diantaranya Direct3D (Windows dan Xbox 360), OpenGL (Mac, Windows, Linux, dan PlayStation 3), OpenGL ES (Android dan iOS), dan Proprietary APIs (Wii). Unity juga memiliki

kemampuan untuk bump mapping, parallax mapping, full screen post-processing effect, reflection mapping, screen space ambient occlusion (SSAO), dynamic shadow, dan render-totexture. Dalam mengelola dan mengolah shader, unity menggunakan ShaderLab untuk mengetahui video card yang terpasang pada perangkat.

2. Scripting

Scripting merupakan sebuah proses yang dilakukan oleh programmer untuk menuliskan kode atau mekanisme game yang dibuat di dalam game engine. Proses scripting pada unity dapat dilakukan menggunakan Mono 2.6, sebuah implementasi Open Source dari .NET Framework. Programmer dapat menggunakan beberapa bahasa pemrograman yang telah disediakan oleh unity yaitu diantara lain C# dan UnityScript (bahasa terkustomisasi yang terinspirasi dari syntax ECMAScript, dalam bentuk JavaScript). Pada awalnya unity menyediakan MonoDevelop yang terkustomisasi untuk digunakan para programmer melakukan debug scripts. Namun pada saat ini, proses debug scripts pada unity juga dapat dilakukan menggunakan Visual Studio yang dibuat oleh Microsoft.

3. Asset Tracking

Asset Tracking berguna untuk memberi banyak pilihan asset dan script bagi developer game atau pengguna unity, asset tracking dapat dilakukan menggunakan Server Unity Asset. Server tersebut menggunakan PostgreSQL sebagai backend, sistem audio dibuat menggunakan FMOD library, video playback menggunakan Theora codec, built-in lightmapping dan global illumination menggunakan Beast, multiplayer networking menggunakan RakNet, dan NavMesh atau navigasi mesh pathfinding Built-in.

4. Physics

Unity memiliki built-in yang mendukung untuk PhysX physics engine yang telah tersedia sejak unity versi 3.0 dari Nvidia. Built-in yang dimiliki unity memiliki kemampuan untuk simulasi real time cloth pada arbitrary, collision layer, thick raycast, skinned meshes, dan juga gravity. Hal ini dapat mendukung para developer yang ingin menambahkan unsur physics ke dalam proyek mereka.

5. Platform

Platform merupakan sebuah perangkat atau device yang digunakan untuk menggunakan aplikasi atau memainkan game yang dibuat menggunakan unity. Saat ini unity telah mendukung pengembangan ke berbagai platform, beberapa platform yang telah disediakan unity yaitu Android, Windows, iOS, Web Browser, Linux, PlayStation, Xbox, dan Nintendo.

2.7. C#

C# atau biasa dibaca C sharp merupakan bahasa pemrograman yang cukup terkenal karena kesederhanaannya. C# memiliki konsep pemrograman berorientasi objek yang memungkinkan developer untuk membuat dan mengembangkan aplikasi ataupun game yang aman dan kuat berjalan di .NET Framework. Beberapa contoh konsep pemrograman berorientasi objek yang dimiliki oleh C# yaitu class dan inheritance.

C# pertama kali muncul pada tahun 1990 yang diciptakan oleh Microsoft. Microsoft merekrut Andres Helsberg yang merupakan karyawan Borland penemu bahasa Turbo Pascal dan Borland Delphi untuk membantu proyek Microsoft dalam pembuatan bahasa pemrograman baru yaitu bahasa C#. Pada tahun 2000, C# dikenalkan dan diluncurkan kepada publik dan disebut sebagai bahasa pemrograman utama dalam pengembangan platform Microsoft .NET Framework. Hingga saat ini C# banyak sekali digunakan oleh developer untuk membuat aplikasi dan game.

2.8. Non Playable Character (NPC)

Pada penelitian terdahulu (Chong-Han Kim, 2006) dikatakan bahwa NPC adalah obyek dinamis yang tidak berada di bawah kendali pemain mereka dan dapat memutuskan perilaku sendiri dan beroperasi di ruang virtual dalam game, NPC akan berperilaku dengan cara mengulangi tiga tahap yakni sense, think dan act.

NPC disebut juga autonomous character yakni sebuah karakter dalam game dimana karakter tersebut mempunyai kemampuan untuk menentukan pergerakan secara otomatis dan tanpa dikendalikan oleh pemain. Untuk membuat permainan yang lebih menarik perlu membangun berbagai strategi. Oleh karena itu NPC harus memiliki berbagai pola perilaku dalam suatu lingkungan dalam game (JinHyuk Hong, 2005).

Menurut Craig W. Reynolds, Non Player Character (NPC) atau yang disebut juga dengan autonomous agent adalah karakter atau entitas yang berada di dalam komputer dan media interaktif seperti game dan virtual reality dimana NPC memiliki kemampuan untuk melakukan gerakan atau interaksi secara otomatis dan tidak dapat dikontrol oleh pemain. NPC

dikendalikan secara otomatis oleh komputer, NPC dapat dibagi menjadi teman atau musuh. NPC merupakan komponen yang sangat penting dalam game, karena dengan adanya NPC dapat menciptakan sebuah karakter cerdas dan nyata, jika tidak ada NPC yang berperilaku cerdas, maka akan membuat game terlihat tidak menarik dan membosankan.

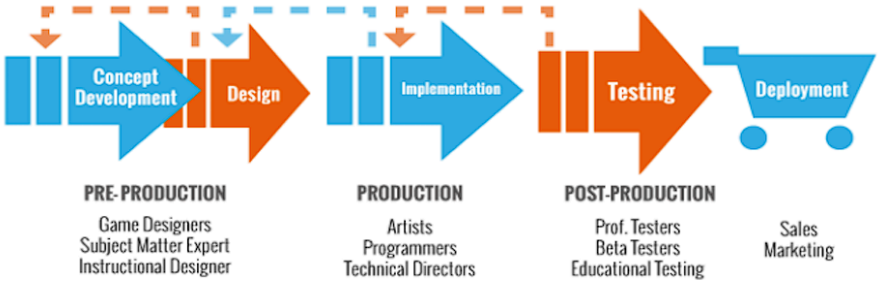
NPC yang diharapkan dapat berperilaku cerdas layaknya manusia. Oleh karena itu NPC dapat mendeteksi lingkungan, berpikir, dan memilih aksi lalu bertindak sebagai respon atas perubahan pada lingkungannya. Untuk dapat memperoleh perilaku cerdas dari NPC digunakan kecerdasan buatan atau Artificial Intelligence (AI). Penggunaan AI pada NPC dilakukan dengan pemberian algoritma khusus sesuai dengan perilaku cerdas yang diharapkan.

Secara garis besar maka NPC dapat diartikan sebagai sebuah karakter atau entitas yang sepenuhnya dikendalikan oleh komputer dan tidak dapat dimainkan oleh pemain. Pengendalian NPC umumnya menggunakan bidang ilmu kecerdasan buatan atau yang disebut dengan Artificial Intelligence (AI) dengan diberikan fungsi – fungsi seperti dapat bergerak, berinteraksi dengan object / pemain, serta melakukan path finding.

2.9. Agile Development

Ada banyak sekali metode dalam perancangan dan pengembangan perangkat lunak, salah satunya yang cocok digunakan dalam perancangan dan pengembangan perangkat lunak berjenis game yaitu metode agile development. Metode agile development merupakan sebuah metode pengembangan aplikasi game yang berbasis pada iterative dan incremental model. Agile development dinilai lebih efektif dan efisien daripada metode pengembangan model tradisional yang kurang efektif.

Agile development lebih mementingkan kecepatan dalam pengembangan aplikasi game yang dibuat, karena dalam pengembangan aplikasi game menggunakan metode agile development akan dilakukan pemecahan dalam pengerjaan fitur-fitur sesuai dengan skala prioritas masing-masing fitur. Metode agile development memiliki 5 tahapan yaitu diantaranya Concept Development, Design, Implementation, Testing, dan Deployment.



Gambar 2.2 Tahapan pengembangan metode agile development

2.9.1. Concept Development

Concept development atau biasa disebut dengan pengembangan konsep merupakan tahapan yang pertama dalam pembuatan dan pengembangan game menggunakan metode agile development. Pada tahap pengembangan konsep ini akan dilakukan pengumpulan ide-ide, pembuatan konsep, dan pengumpulan data-data yang dibutuhkan untuk pembuatan dan pengembangan aplikasi game edukasi. Tahapan ini bertujuan untuk menentukan tujuan dibuatnya game dan pengguna yang akan memainkan game. Karakteristik dan kemampuan pengguna harus dipertimbangkan pada tahap ini, sebab kedua hal tersebut sangat berpengaruh terhadap desain game. Hasil dari tahap pengembangan konsep ini berupa dokumen yang biasa disebut Game Design Document (GDD).

2.9.2. Design

Design atau perancangan adalah tahapan yang dilakukan untuk menentukan model tampilan dan bentuk dari game yang akan dibuat dan dikembangkan. Pada tahapan ini juga dilakukan pembuatan rancangan dari aplikasi game yang akan dikembangkan. Rancangan ini meliputi rancangan untuk asset atau bahan yang digunakan dalam pembuatan game, user interface (UI), game mechanic atau sistem utama dari permainan, dan juga game flow atau alur dari game.

2.9.3. Implementation

Implementation atau implementasi merupakan tahapan yang dilakukan setelah selesai melakukan perancangan. Pada tahapan ini akan dilakukan implementasi dari rancangan desain yang telah dibuat sebelumnya. Desain akan diterapkan pada sistem aplikasi game dimana aplikasi game akan dikembangkan menggunakan perangkat lunak Unity game engine dan menggunakan bahasa pemrograman C#.

Pada tahap implementasi ini juga akan dilakukan beberapa proses yaitu proses pengimplementasian desain yang telah dibuat sebelumnya ke dalam kode dan fitur-fitur, setelah proses tersebut maka akan dilakukan proses pengecekan ulang kode dan fitur-fitur yang telah dibuat sebelumnya apakah telah sesuai dengan konsep dan tujuan yang dirancang sebelumnya, jika belum sesuai maka proses pengulangan akan terus dilakukan hingga implementasi sesuai dengan konsep dan ide.

2.9.4. Testing

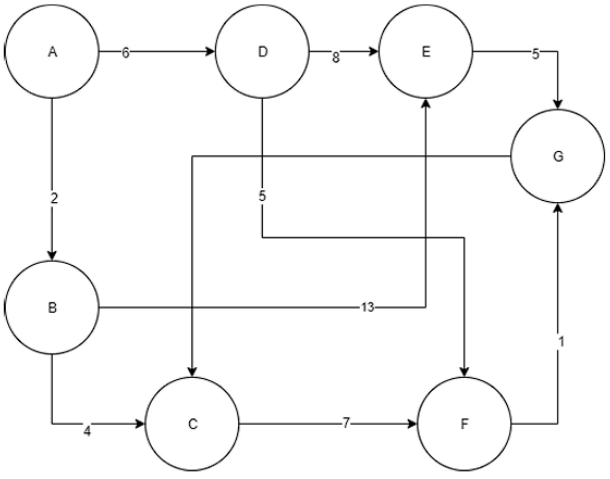
Pada tahapan testing, aplikasi game edukasi yang telah menyelesaikan tahap konsep, desain, dan implementasi akan dilakukan uji coba. Tahapan testing ini dilakukan untuk memastikan apakah aplikasi game yang dibuat dan dikembangkan sesuai dengan target dan konsep yang sebelumnya telah

2.9.4. Deployment

Deployment atau biasa disebut dengan perilisan merupakan tahapan terakhir pada metode agile development. Pada tahapan perilisan ini akan dilakukan perilisan atau peluncuran aplikasi game edukasi yang telah dibuat kepada public menggunakan berbagai media dan layanan. Pada tahap ini juga akan dilakukan proses pemeliharaan aplikasi game secara berkala agar selalu dapat menyesuaikan kebutuhan dari pengguna.

2.10. Path Finding

Path Finding merupakan salah satu materi yang sangat penting di dalam AI. Path Finding biasanya digunakan untuk menyelesaikan masalah pada sebuah graph. Dalam matematika, graph merupakan himpunan titik - titik atau biasa disebut dengan node yang terhubung oleh edge (penghubung antar node). Edge yang menghubungkan setiap node merupakan suatu vektor yang memiliki arah dan besaran tertentu. Untuk dapat menemukan jalan dari node awal menuju node tujuan, dilakukan penelusuran terhadap graph tersebut. Penelusuran biasanya dilakukan dengan mengikuti arah edge yang menghubungkan antar node.



Gambar 2.2 Graph Path Finding

Dapat dilihat dari Gambar 2.2, suatu graph dengan node Awal A dan Node Tujuan G terhubung dengan node - node lain dengan edge - edge yang memiliki besaran yang berbeda.

Jika ditelusuri, terdapat banyak kombinasi rute yang dapat dilalui untuk menuju node tujuan. Bisa dikatakan dari graph tersebut, setiap node akan memberikan solusi arah menuju node tujuan. Untuk melaksanakan path finding dapat dilakukan beberapa metode pathfinding seperti algoritma Breadth First Search, Dijkstra, dan A Star

2.11. Algoritma A*

Algoritma A* (A star) merupakan sebuah algoritma yang dikategorikan kedalam metode pencarian yang memiliki informasi (Informed Search Method). Algoritma ini menghasilkan hasil yang sangat baik ketika digunakan dalam proses path finding. Algoritma ini mencari jarak rute terpendek dari sebuah jalan atau path, dimulai suatu point awal (starting point) sampai ke objek tujuan. Algoritma A* dapat memberikan hasil yang maksimal dalam pencarian rute terpendek. Metode cara kerja dari algoritma ini menggunakan dasar best first search (Dalem, 2018).

Cara Kerja A* dapat dijelaskan dengan ilustrasi dari Gambar 2.2, dari graph dengan node awal A dan node tujuan *G* terhubung dengan node - node lain oleh edge yang memiliki besaran yang berbeda. Jika ditelusuri, terdapat banyak kombinasi rute yang dapat dilalui untuk menuju node tujuan. Bisa dikatakan dari graph tersebut, bahwa setiap node akan memberikan beberapa solusi arah untuk mencapai node tujuan. A* mengevaluasi node dengan menggabungkan g(n), yaitu cost untuk mencapai node tujuan, dan h(n), yaitu cost yang diperlukan dari node untuk mencapai tujuan, dalam notasi matematika dituliskan sebagai berikut:

$f(x) = g(x) + h(x)$, dimana:

$f(n)$ = Biaya evaluasi

$g(n)$ = Biaya yang sudah dikeluarkan dari keadaan awal sampai keadaan n

$h(n)$ = Estimasi biaya untuk sampai pada suatu tujuan mulai dari n

Dengan menggunakan algoritma A* dapat menentukan rute teroptimal dengan waktu yang cepat karena, dengan bantuan fungsi heuristic dapat memilih node yang terbaik untuk sampai tujuan, akan tetapi akan membutuhkan proses yang besar karena algoritma ini akan memasukkan semua rute kedalam sebuah tabel lalu menggunakan fungsi heuristic akan menentukan rute terpendek.

2.12. Algoritma Dijkstra

Algoritma Dijkstra merupakan suatu algoritma disebut juga sebagai single source shortest path yang digunakan dalam menentukan jalur terpendek dari node sumber menuju node tujuan

berdasarkan bobot tetangga. Bobot tetangga dapat berupa jarak, waktu, biaya, ataupun bobot yang lainnya. Algoritma Dijkstra bekerja dengan cara mengunjungi semua semua node - node yang terdapat pada graph dengan dimulai pada node sumber ke node dengan bobot paling kecil. Secara berulang algoritma ini akan memilih node - node terdekat dan menghitung total bobot semua sisi yang dilewati untuk mencapai node tujuan. Secara singkat proses algoritma Dijkstra dapat dijelaskan sebagai berikut:

1) Inisialisasi verteks (simpul).
2) Inisialisasi jarak antar verteks.
3) Tentukan verteks awal (s) dan verteks tujuan (t).
4) Beri label permanen= 0 ke verteks awal (s) dan label sementara= infinity ke verteks lainnya.
5) Untuk setiap verteks V, yang belum mendapat label permanen, mendapatkan label sementara
6) Cari harga minimum diantara semua verteks yang masih berlabel sementara.
7) Jadikan verteks minimum yang berlabel sementara menjadi verteks dengan label permanen, jika lebih dari satu verteks dipilih sembarang.
8) Ulangi langkah 5 sampai 7 hingga verteks tujuan mendapat label permanen.
9) Simpan hasil perhitungan.
10) Tampilkan hasil pencarian.

Algoritma ini akan mencari node tujuan dengan menelusuri semua node- node yang dapat dikunjungi, yang nantinya akan menghasilkan rute teroptimal, akan tetapi membuat proses pencarian menjadi panjang, karena proses mengunjungi semua node - node untuk mencapai tujuan akan membutuhkan performa komputer yang tinggi.

2.13. Algoritma Breadth First Search (BFS)

Breadth first Search atau disingkat dengan BFS adalah metode pencarian yang bertujuan untuk memperluas dan memeriksa semua node dari sebuah graph atau kombinasi dari urutan dengan menggunakan semua solusi secara sistematis. dengan kata lain, BFS mencari ke seluruh graph atau urutan secara mendalam tanpa mempertimbangkan tujuannya (goal) sampai tujuan itu tercapai.

Algoritma ini memerlukan sebuah tabel antrian untuk menyimpan node yang telah dikunjungi. Node - node ini diperlukan sebagai acuan untuk mengunjungi node - node yang bertetanggaan dengannya. Tiap node yang telah dikunjungi masuk ke dalam antrian hanya satu kali saja. Algoritma ini juga membutuhkan table Boolean untuk menyimpan node yang telah

dikunjungi sehingga tidak ada node yang dikunjungi lebih dari satu kali.

Dalam algoritma BFS, node yang telah dikunjungi disimpan dalam suatu antrian. Antrian ini digunakan untuk mengacu kepada node - node yang bertetangga dengan yang akan dikunjungi kemudian sesuai urutan pengantrian. Untuk memperjelas cara kerja algoritma BFS beserta antrian yang digunakannya, berikut langkah-langkah algoritma BFS:

1) Masukkan node sumber ke dalam antrian
2) Ambil node dari awal antrian, lalu cek apakah node merupakan solusi
3) Jika node merupakan solusi, pencarian selesai dan hasil dikembalikan.
4) Jika node bukan solusi, masukkan seluruh node yang bertetangga dengan node tersebut (node tetangga) ke dalam antrian.
5) Jika antrian kosong dan setiap node sudah dicek, pencarian selesai dan mengembalikan hasil solusi tidak ditemukan.
6) Ulangi pencarian dari langkah kedua.

Dengan cara kerja BFS, algoritma ini dapat menemukan rute teroptimal tanpa menemukan adanya jalan buntu, karena proses pengulangan yang dilakukan oleh algoritma ini, menjamin ditemukannya solusi dan solusi tersebut adalah yang terbaik.

2.14. Success Completion Rate (SCR)

Success Completion Rate (SCR) adalah metrik yang digunakan untuk mengevaluasi tingkat keberhasilan suatu proses atau sistem dalam menyelesaikan tugas yang diberikan. Metrik ini sering digunakan dalam berbagai bidang, seperti kecerdasan buatan, sistem navigasi, dan pengujian perangkat lunak, untuk mengukur seberapa sering sistem mampu mencapai tujuan yang diinginkan. SCR dinyatakan sebagai persentase dari jumlah percobaan sukses dibandingkan dengan total jumlah percobaan yang dilakukan. SCR memberikan gambaran mengenai keandalan sistem atau algoritma dalam menyelesaikan tugas tertentu.

2.15. Average Completion Time (ACT)

Average Completion Time (ACT) adalah metrik yang digunakan untuk mengukur rata-rata waktu yang diperlukan oleh suatu sistem atau proses untuk menyelesaikan tugas tertentu. ACT memberikan indikasi efisiensi temporal suatu sistem dalam menyelesaikan pekerjaannya. ACT dihitung sebagai rata-rata dari waktu penyelesaian semua percobaan yang dilakukan. Metrik ini sering digunakan untuk mengevaluasi kecepatan suatu algoritma atau sistem dalam berbagai bidang, seperti transportasi, simulasi, dan pengujian perangkat lunak.

BAB III
DESAIN DAN IMPLEMENTASI SISTEM

Metodologi penelitian merupakan sebuah pedoman dalam pelaksanaan penelitian sehingga hasil yang dicapai tidak berbeda dari tujuan yang telah ditentukan sebelumnya. Bab ini menjelaskan metode penelitian yang digunakan untuk menganalisis dan membandingkan kinerja algoritma A-Star, Dijkstra, dan BFS dalam pathfinding NPC pada game. Penelitian ini dilakukan dengan pendekatan eksperimental, menggunakan peta dinamis dan rintangan yang dapat berubah. Evaluasi kinerja dilakukan berdasarkan Success Completion Rate (SCR) dan Average Completion Time (ACT).

3.1. Desain Penelitian

Pada desain penelitian ini memiliki beberapa tahapan yang dilakukan untuk menyelesaikan penelitian ini. Berikut tahapan-tahapan yang digunakan.

3.1.1. Studi Literatur

Studi literatur dilakukan dengan cara mengumpulkan data dari sumber yang sudah ada dari jurnal, artikel, video tutorial, media internet, buku digital, teks dan yang terkait mengenai penelitian sejenis untuk menjadi dasar dari penelitian yang dilakukan ini. Beberapa kata kunci yang peneliti gunakan adalah sebagai berikut :

1) Path Finding.
2) A Star (A*).
3) Dijkstra.
4) Breadth First Search (BFS).
5) Heuristic.
6) Cost.
7) Open Set.
8) Closed Set

3.1.2. Analisa Kebutuhan Kebutuhan

Analisa Kebutuhan Kebutuhan yang akan dikembangkan oleh penulis tentu memiliki kebutuhan – kebutuhan agar penelitian dapat terselesaikan dengan baik. Kebutuhan – kebutuhan ini akan dibagi menjadi 2 bagian yaitu kebutuhan Hardware dan Software.

1) Kebutuhan Hardware

a) Personal Computer (PC)
   Personal Computer atau PC diperlukan untuk melakukan pengembangan gim yang tentunya diperlukan dengan spek yang cukup untuk menggunakan aplikasi – aplikasi yang diperlukan selama proses pengembangan gim.

2) Kebutuhan Software
   a) Unity Game Engine
      Unity Game Engine atau biasa disebut dengan unity merupakan software pengembangan gim yang cukup populer dan untuk mengembangkan gim ini penulis menggunakan aplikasi Unity karena penggunaannya yang relatif mudah dan fitur yang ditawarkan cukup lengkap.
   b) Visual Studio
      Visual Studio merupakan aplikasi scripting untuk menuliskan kode – kode yang diperlukan untuk pengembangan gim, dimana aplikasi ini dapat terintegrasi langsung dengan aplikasi Unity yang digunakan penulis untuk mengembangkan gim.

3.2. Metode Pengembangan

Metode pengembangan dalam penelitian ini menggunakan Agile Development, yaitu pendekatan iteratif yang memungkinkan penyesuaian dan perbaikan pada setiap tahap pengembangan berdasarkan umpan balik yang diperoleh. Agile Development dipilih karena fleksibilitas nya, terutama dalam proyek dengan banyak variabel seperti pengujian algoritma pathfinding dalam lingkungan game yang dinamis. Metode ini terdiri dari beberapa tahapan berikut:

3.2.1. Concept Development

Pada tahap ini, dilakukan pengembangan konsep awal game yang menjadi landasan seluruh proses penelitian. Langkah-langkah yang dilakukan meliputi:

1) Identifikasi Tujuan Game

Membuat game yang memfasilitasi pengujian algoritma pathfinding NPC, yaitu A-Star, Dijkstra, dan BFS. Serta menentukan kriteria evaluasi algoritma dengan waktu penyelesaian rute (Average Completion Time - ACT) dan tingkat keberhasilan mencapai tujuan (Success Completion Rate - SCR).

2) Penentuan Lingkungan Pengujian

Peta dengan variasi kompleksitas dari peta sederhana hingga peta dinamis dengan rintangan yang dapat berubah. Penggunaan rintangan yang muncul tiba-tiba dan jalur yang terblokir secara acak.

3) Perancangan Mekanika Game

NPC harus bergerak menuju tujuan dengan algoritma pathfinding tertentu. Pemain tidak mengontrol NPC tetapi dapat melihat bagaimana algoritma bekerja di latar belakang dan bia mencoba untuk berkompetisi dengan NPC tersebut. Serta tambahan elemen visual untuk jejak rute yang dilalui NPC untuk setiap algoritma.

3.2.2. Desain

Desain mencakup pembuatan blueprint yang lebih rinci untuk implementasi game. Aktivitas yang dilakukan pada tahap ini meliputi:

1) Perancangan Alur Permainan (Game Flow)

NPC mulai dari titik awal dan diberi misi mencapai titik tujuan. Serta setiap iterasi, rintangan yang berbeda diterapkan dan performanya dicatat.

2) Perancangan Navigasi dan Rintangan

Menggambarkan bagaimana NPC memilih jalur berdasarkan algoritma pathfinding. Menentukan area yang dapat dilalui dan area terlarang dalam peta.

3) Desain Antarmuka

Panel untuk pemain melihat algoritma mana yang sedang berjalan. Grafik dan animasi yang menampilkan perbandingan waktu tempuh dan jalur yang dilalui.

3.2.3. Implementasi

Implementasi melibatkan penerjemahan desain ke dalam kode program. Langkah-langkah utama pada tahap ini meliputi:

1) Pemrograman Algoritma Pathfinding

Mengimplementasikan algoritma A-Star, Dijkstra, dan BFS dalam Unity menggunakan bahasa pemrograman C#. Menambahkan fitur untuk prioritas jalur, heuristik untuk A-Star, dan perhitungan bobot untuk Dijkstra.

2) Integrasi Algoritma dengan Peta Dinamis:

Menyesuaikan algoritma agar dapat beradaptasi terhadap perubahan rintangan secara real-time. Menguji algoritma dalam berbagai skenario seperti jalur yang terblokir atau munculnya rintangan baru.

3) Pembuatan Visualisasi:

Menampilkan jalur yang diambil oleh algoritma dalam bentuk animasi. Menyediakan grafik atau tabel yang memperlihatkan hasil pengujian setiap algoritma.

3.2.4. Testing

Tahap ini mencakup pengujian terhadap performa algoritma dan elemen-elemen lain dalam game. Jenis pengujian yang dilakukan meliputi:

1) Pengujian Black Box

Memastikan algoritma menghasilkan jalur yang valid dan NPC dapat mencapai tujuan dan menguji antarmuka untuk memastikan data hasil pengujian ditampilkan dengan benar.

2) Pengujian Performansi:

Menggunakan metrik Success Completion Rate (SCR) dan Average Completion Time (ACT) untuk membandingkan algoritma. Simulasi dilakukan dengan jumlah iterasi yang cukup untuk mendapatkan data yang konsisten.

3.2.4. Deployment

Setelah pengujian selesai, game dirilis untuk diuji oleh pengguna. Pada tahapan ini juga akan disesuaikan dengan tahap deployment pada metode agile development. Deployment akan dilakukan pada platform website itch.io agar dapat diunduh dan dimainkan oleh anak-anak untuk belajar mengenai cara menghadapi bencana alam dengan menarik.

3.3. Penerapan Algoritma

Pada tahap ini, algoritma pathfinding A-Star, Dijkstra, dan BFS diterapkan dalam Unity 3D untuk digunakan oleh NPC dalam menemukan jalur terbaik menuju tujuan. Setiap algoritma memiliki karakteristik unik yang diimplementasikan menggunakan bahasa pemrograman C#. Selain itu, peta yang digunakan dirancang dengan berbagai elemen dinamis,

seperti rintangan yang muncul secara acak, untuk menguji kemampuan adaptasi algoritma terhadap perubahan lingkungan.

3.3.1. Penerapan Algoritma A-Star

Algoritma A-Star merupakan salah satu algoritma pencarian jalur yang paling banyak digunakan karena kemampuannya dalam menemukan jalur optimal dengan efisiensi tinggi. A-Star menggabungkan kelebihan Dijkstra dan Greedy Best-First Search dengan menggunakan fungsi heuristik.

1) **Tahapan Penerapan di Unity 3D**
    a) **Representasi Peta**
    Peta diwakili oleh grid 2D, di mana setiap node memiliki status (dapat dilalui atau tidak).
    b) **Pengelolaan Node**
    Node disimpan dalam struktur data seperti list atau priority queue.
    c) **Evaluasi Node**
    Node dengan nilai f(n) terendah dalam Open Set dipilih untuk dievaluasi.
    d) **Integrasi dengan NPC**
    NPC menggunakan jalur yang dihitung oleh A-Star untuk bergerak ke tujuan.

3.3.2. Penerapan Algoritma Dijkstra

Algoritma Dijkstra adalah metode pencarian jalur yang dijamin memberikan jalur terpendek. Namun, algoritma ini tidak menggunakan fungsi heuristik, sehingga cenderung memproses lebih banyak node dibandingkan A-Star.

1) **Tahapan Penerapan di Unity 3D**
    a) **Pengelolaan Node**
    Menggunakan struktur data seperti priority queue untuk memilih node dengan bobot terendah.
    b) **Evaluasi Jalur**
    Setiap node tetangga diperiksa dan bobotnya diperbarui jika ditemukan jalur yang lebih pendek.
    c) **Integrasi dengan NPC**
    NPC bergerak melalui jalur yang dihitung oleh algoritma.

3.3.3. Penerapan Algoritma BFS

Algoritma BFS menggunakan pendekatan traversal level-order, di mana semua node pada satu tingkat dievaluasi sebelum bergerak ke tingkat berikutnya.

1) **Tahapan Penerapan di Unity 3D**
    a) **Antrian Node**
    Node disimpan dalam struktur data queue untuk memprosesnya secara berurutan.
    b) **Evaluasi Node**
    Setiap node tetangga yang belum dievaluasi ditambahkan ke queue.
    c) **Keunggulan**
    BFS sederhana dan cocok untuk peta kecil atau tanpa bobot, tetapi tidak optimal untuk peta besar dengan banyak rintangan.

3.3.4. Integrasi Algoritma dengan Unity 3D

1) **Tahapan Penerapan di Unity 3D**

Lingkungan dibuat dalam Unity 3D menggunakan komponen terrain untuk peta dan obstacle untuk rintangan.

2) **NPC Controller**

Skrip C# digunakan untuk mengontrol pergerakan NPC berdasarkan jalur yang dihitung algoritma.

3) **Adaptasi Peta Dinamis**

Ketika rintangan baru muncul, algoritma memicu proses perhitungan ulang jalur secara real-time.

3.3. Pengujian

Pengujian dilakukan untuk menguji kinerja berbagai algoritma pencarian jalur dalam berbagai skenario, baik pada peta statis maupun dinamis. Pengujian ini dilakukan untuk mengetahui keunggulan dan kelemahan masing-masing algoritma dalam berbagai kondisi. Proses pengujian terdiri dari beberapa tahapan yang dimulai dengan persiapan lingkungan hingga pengumpulan data hasil evaluasi performa algoritma.

3.3.1. Lingkungan Pengujian

Pengujian dilakukan dengan menggunakan dua jenis peta yang berbeda, masing-masing memberikan tantangan yang berbeda untuk algoritma yang diuji.

1. Peta Statis

Pada peta statis, elemen-elemen seperti jalur dan rintangan tetap selama seluruh simulasi. Rintangan pada peta ini berada pada posisi yang tetap dan tidak berubah sepanjang simulasi, sehingga algoritma hanya perlu mencari jalur dari titik awal menuju titik tujuan berdasarkan informasi statis yang diberikan. Keuntungan dari peta statis adalah dapat memberikan gambaran yang lebih jelas mengenai efisiensi dan ketepatan algoritma dalam menemukan jalur optimal di lingkungan yang tidak berubah.

2. Peta Dinamis

Sebaliknya, pada peta dinamis, rintangan dapat berpindah posisi secara acak selama simulasi berlangsung. Hal ini memaksa algoritma untuk terus menyesuaikan jalurnya secara real-time saat menghadapi perubahan rintangan yang tak terduga. Peta dinamis ini mensimulasikan kondisi yang lebih mirip dengan permainan video atau aplikasi dunia nyata, di mana lingkungan dapat berubah secara dinamis. Pengujian pada peta dinamis bertujuan untuk menguji kemampuan algoritma dalam menanggapi perubahan dan menyesuaikan jalur dengan cepat dan efisien.

3.3.2. Skenario Pengujian

Untuk menguji berbagai algoritma pencarian jalur, NPC (Non-Player Character) diberi tugas untuk mencapai tujuan tertentu di dalam peta menggunakan algoritma yang berbeda-beda. Beberapa skenario pengujian dilakukan sebagai berikut:

1) Jalur yang sudah diketahui

   Pada skenario ini, NPC mencoba mencapai tujuan di peta statis dengan algoritma yang digunakan sudah mengetahui semua informasi tentang peta, termasuk posisi awal, tujuan, dan semua rintangan yang ada. Algoritma diuji untuk menentukan seberapa cepat dan efektif jalur yang ditemukan dalam kondisi peta yang tidak berubah.

2) Jalur dengan rintangan dinamis

   Dalam skenario ini, NPC diberikan tugas untuk mencapai tujuan di peta dinamis yang memiliki rintangan yang dapat bergerak atau berubah posisi selama simulasi.

Tujuan dari pengujian ini adalah untuk mengukur kemampuan algoritma dalam beradaptasi dengan perubahan rintangan yang terjadi, serta seberapa cepat algoritma dapat menemukan jalur alternatif yang optimal.

3) Rintangan tambahan

Selain rintangan statis atau dinamis, dalam beberapa kasus, pengujian memperkenalkan rintangan yang lebih kompleks, seperti blokade yang memerlukan pencarian jalur alternatif atau perubahan bentuk peta. Ini menguji fleksibilitas algoritma dalam menangani situasi yang lebih rumit, misalnya saat jalur yang sudah ditemukan menjadi terhalang atau tidak lagi tersedia.

3.3.3. Pengumpulan Data

Selama pengujian, sejumlah data dikumpulkan untuk mengevaluasi performa setiap algoritma yang diuji. Data yang dikumpulkan meliputi beberapa metrik kunci yaitu Success Completion Rate (SCR) dan Average Completion Time (ACT), yang memberikan gambaran tentang efektivitas dan efisiensi algoritma dalam berbagai kondisi.

3.4. Evaluasi

Evaluasi merupakan tahap penting dalam penelitian ini, di mana analisis mendalam terhadap hasil pengujian digunakan untuk menentukan algoritma pencarian jalur yang paling efektif berdasarkan kriteria tertentu. Selama pengujian, berbagai metrik dan indikator kinerja diperoleh untuk memberikan gambaran yang lebih jelas mengenai keunggulan dan kelemahan masing-masing algoritma. Evaluasi ini meliputi analisis terhadap performa algoritma dan visualisasi hasil untuk membantu memahami karakteristik masing-masing algoritma.

3.3.3. Performa Algoritma

Performa algoritma merupakan salah satu faktor utama dalam mengevaluasi efektivitas setiap metode pencarian jalur. Untuk mengukur performa algoritma, dua metrik utama yang dikumpulkan selama pengujian adalah Success Completion Rate (SCR) dan Average Completion Time (ACT). Kedua metrik ini memberikan wawasan mengenai seberapa efisien dan handalnya algoritma dalam menyelesaikan tugas pencarian jalur.

1) Kecepatan Penyelesaian (ACT)

Kecepatan penyelesaian, yang diukur melalui Average Completion Time (ACT), menunjukkan seberapa cepat sebuah algoritma dapat menemukan jalur dari titik awal menuju tujuan. Semakin rendah nilai ACT, semakin efisien algoritma tersebut dalam hal waktu komputasi. Algoritma dengan ACT terendah dianggap lebih efisien, karena meminimalkan waktu yang dibutuhkan untuk menjalankan pencarian jalur.

2) Keberhasilan Penyelesaian (SCR)

Success Completion Rate (SCR) mengukur seberapa sering algoritma berhasil menemukan jalur dari titik awal menuju tujuan. Algoritma dengan SCR tertinggi dianggap lebih handal, karena menunjukkan kemampuannya dalam menyelesaikan pencarian jalur meskipun ada rintangan atau perubahan pada peta. SCR yang tinggi berarti algoritma mampu menangani berbagai tantangan yang ada, baik itu peta statis maupun dinamis.

Evaluasi performa algoritma dalam hal kecepatan penyelesaian dan tingkat keberhasilan penyelesaian akan memberikan gambaran yang lebih jelas mengenai kelebihan masing-masing algoritma dan bagaimana mereka dapat diimplementasikan dalam konteks yang berbeda.

3.3.3. Visualisasi Hasil

Visualisasi hasil pengujian memberikan pemahaman yang lebih intuitif mengenai perbedaan pendekatan antara masing-masing algoritma dalam memecahkan masalah pencarian jalur. Beberapa metode visualisasi yang digunakan meliputi grafik perbandingan dan visualisasi jalur yang dilalui NPC.

Grafik perbandingan antara Average Completion Time (ACT) dan Success Completion Rate (SCR) untuk setiap algoritma dapat membantu dalam mengidentifikasi kelebihan dan kekurangan dari masing-masing algoritma. Dengan memvisualisasikan kedua metrik ini, kita dapat melihat trade-off antara kecepatan dan keberhasilan algoritma.

DAFTAR PUSTAKA

Bates, B. 2004. Game Design 2nd Edition. Cengange Learning PTR. Boston M.A.

Rosita Ayu Nugraeni, M. R. (2015). PENERAPAN ALGORTIMA A* DALAM PENYELESAIAN RUTE TERPENDEK PENDISTRIBUSIAN BARANG. Jurusan Matematika, FMIPA, Universitas Negeri Semarang, Indonesia, Page 7 - 12.

Dalem, I. B. (2018). Penerapan A Star (A*) Menggunakan Graph Untuk Menghitung Jarak Terpendek. Program Studi Ilmu Komputer, Pasca Sarjana Universitas Pendidikan Ganesha, Singaraja - Bali, Page 41 - 47.

Blasius Neri Puspika, A. R. (2012). Implementasi Algoritma Dijkstra Daram penentuan Jalur Terpendek Di Yogyakarta Menggunakan GPS Dan Qt Geolocation. Informatika: Jurnal Teknologi Komputer dan Informatika, Page 141 - 149.

Rosita Ayu Nugraeni, M. R. (2015). PENERAPAN ALGORTIMA A* DALAM PENYELESAIAN RUTE TERPENDEK PENDISTRIBUSIAN BARANG. Jurusan Matematika, FMIPA, Universitas Negeri Semarang, Indonesia, Page 7 - 12.

Agus Pamungkas, E. P. (2014). Penerapan Algoritma A* (A Star) Pada Game Edukasi The Maze Island Berbasis Android. Jurusan Teknik Informatika, STMIK GI MDP, Palembang.

Caesar, R. (2015). “Kajian Pustaka Perkembangan Genre Game Dari Masa Ke Masa”. Journal of Animation and Games Studies 1, 2:113-134.

Nur Abdilah. (2023). Ada 4 Jenis dan 11 Genre Game, Yang Mana Favorit Kamu ?, <URL: https://www.pricebook.co.id/article/review/3593/ada-4-jenisdan-11-genre-game-yang-mana-favorit-kamu/>.

Hermawan, D., P., Herumurti, D., dan Kuswardayan, I. (2017). “Efektifitas Penggunaan Game Edukasi Berjenis Puzzle, RPG, dan Puzzle RPG Sebagai Sarana Belajar Matematika”. Jurnal Ilmiah Teknologi Informasi 15, 2:195-205.

Chong-Han Kim, S.-M. J.-T.-G. (2006). Verification of FSM using Attributes Definition of NPCs. Dong Shin University Digital Contents Cooperation Research Center, Department of Digital Contents, Dong Shin Unversity, Department of Computer Science, Chonnam National University, Page 168 - 174.