IMPLEMENTASI ALGORITMA PATHFINDING DAN DECISION TREE DALAM PEMBUATAN VIDEO GAME BERGENRE THIRD PERSON SHOOTER

Bambang Sugianto[1)], **Gunawan Pria Utama**[2)]
[1]Teknik Informatika, Fakultas Teknologi Informasi, Universitas Budi Luhur
[1,2]Jl. Raya Ciledug, Petukangan Utara, Kebayoran Lama, Jakarta Selatan 12260
E-mail : bambangsugianto68@gmail.com[1)], gunawan.priautama@budiluhur.ac.id[2)]

Abstrak

Permainan Video (Video Game) adalah permainan yang memanfaatkan teknologi berupa alat elektronik yang mampu menampilkan suatu gambar sebagai media antarmuka dengan pengguna dan pengguna dapat mengendalikan Obyek yang ada dalam gambar tersebut dengan alat pengendali (Input Device). Seiring perkembangan zaman, Permainan Video memiliki media keluaran lain (Output) seperti suara dan juga getaran. Universitas Budi Luhur memiliki komunitas yang dinamakan Kotak Kreatif, yaitu sebuah komunitas developer video game di Universitas Budi Luhur. Komunitas Kotak Kreatif sudah menciptakan beberapa permainan untuk platform Android dan PC. Namun, pada komunitas ini belum pernah menerapkan teknik khusus Artificial Intelligence (AI) dalam judul permainan yang telah diciptakan oleh komunitas Kotak Kreatif. Oleh karena itu tujuan utama dari Penelitian ini adalah untuk menciptakan konten AI dalam permainan bergenre Third Person Shooter (TPS) sebagai lawan main, karena genre TPS tersebut bergantung pada lawan main sebagai aspek utama. Untuk itu dibutuhkan Metode Pathfinding sebagai penentu arah jalan dan Decision Tree sebagai penentu tindakan (Behaviour) lawan main. Metode Pathfinding menggunakan Algoritma A, cara kerjanya dengan mengambil jarak antara setiap node dan tujuan. Metode Decision Tree adalah sebuah pohon hirarki yang diimplementasikan dengan menggunakan Nested if-else, setiap kondisi akan diperiksa dan keputusan akan diambil bedasarkan kondisi.*

Kata kunci: Kecerdasan Tiruan, *Artificial Intelligence, Pathfinding, A* Algorithm, Decision Tree*, *AI Behaviour*

1. PENDAHULUAN

Industri Hiburan tak luput dari ranah Teknologi Informasi. Dalam dunia Teknologi Informasi memiliki banyak produk hasil dari penerapan cabang ilmu yang diterapkan menjadi hiburan, yaitu Permainan Video (*Video Game*). Untuk menciptakan suatu Permainan Video, dibutuhkan informasi pengetahuan baik di bidang Komputer, khususnya Teknologi Informasi, Matematika dan bahkan di bidang Seni dan Kreativitas. Permainan Video juga disebut sebagai hasil Seni Elektronik karena pada umumnya pembuatan karya suatu permainan memiliki aspek estetika, contoh kecilnya musik dan gambar.

Dalam pembuatan Permainan Video dibutuhkan ilmu yang terkait dengan Pemograman, Matematika, Seni digital seperti pembuatan gambar, efek suara dan musik. Pemograman dibutuhkan untuk merancang Permainain Video secara teknis bagaimana program permainan akan berjalan. Dengan *Software* bernama *Unity*, Peneliti akan merancang Permainan Video dengan menerapkan ilmu dalam bidang Teknik Informatika dan beberapa Fisika dasar sebagai *rule* atau aturan dunia permainan seperti menciptakan hukum alam.

Kategori permainan yang akan dibuat adalah permainan Aksi tembak menembak dengan sudut pandang orang ketiga (*Third Person Shooter*). Arti dari sudut pandang orang ketiga adalah perspektif gambar yang akan ditampilkan adalah sudut pandang ketiga / orang lain dari Tokoh yang dimainkan oleh Pemain. Dikarenakan kategori atau *genre* ini membutuhkan simulasi dunia nyata dan hukum fisika sebagai *rule* nya, misalkan obyek peluru akan bergerak ketika pelatuk dipicu dan memiliki kecepatan tertentu hingga menabrak obyek lain dan membuat kerusakan. Jenis permainan ini memiliki penggemar yang cukup banyak mengingat kebanyakan orang bermain *Game* menyukai permainan yang membutuhkan ketangkasan bergerak dan sesuatu yang tidak bisa mereka lakukan di dunia nyata.

Pada *Genre Action*, Setiap tokoh yang tidak dimainkan (*Non Player Character*/*NPC*) dapat melakukan sesuatu agar memiliki kesan hidup. Dalam hal ini diperlukan konsep Kecerdasan Buatan (*Artificial Intelligence*). Setiap Permainan Video memiliki mekanisme Kecerdasan Buatan yang berbeda bergantung *genre* apa yang dipilih pengembang dalam membuat sebuah Permainan Video. NPC melakukan interaksi berdasarkan keputusan baik dengan perintah maupun melakukan keputusan sendiri.

Deskripsi naratif di atas dapat disimpulkan tujuan penelitian ini memiliki beberapa poin antara lain:

1. Bagaimana cara membuat Permainan Video yang kompatibel dengan berbagai mesin komputer yang memiliki kecepatan proses yang bervariasi?
2. Bagaimana penerapan mekanisme permainan dengan *genre Third Person Shooter*?

3. Bagaimana menerapkan *Artificial Intelligence* pada Tokoh lawan main?

2. METODE PENELITIAN

2.1. Studi Literatur

1) Judul : Implementasi Metode Fonetis Pada Game Edukasi Menyusun Huruf Alphabet
   Penulis : Hamdan Mukhlis
   Terbit : 2015
   Penerbit : Pelita Informatika Budi Darma
   ISSN : 2301-9425
   Deskripsi : Pola pikir anak yang suka berma-in digabungkan dengan edukasi melahirkan sebuah *game* edukasi dengan menerapkan metode fotosintesis digunakan dalam membantu dalam pembuatan soal.
2) Judul : A * -based Pathfinding in Modern Computer Games A * -based Pathfinding in Modern Computer Games
   Penulis : Xiao Cui, Hao Shi
   Terbit : 2010
   Penerbit : ResearchGate
   Deskripsi : *Pathfinding* umumnya mengacu pada menemukan rute ter-pendek antara dua titik akhir. Dengan semakin pentingnya industri *game*, *pathfinding* telah menjadi sebuah masalah populer dan frustasi dalam industri *game*. Masalah yang paling umum dari merintis jalan di *video game* adalah bagaimana menghindari rintangan secara cerdik dan mencari jalan yang paling efisien di berbagai medan.
3) Judul : Decision Tree Berbasis Algoritma Untuk Pengambilan Keputusan.
   Penulis : Zulfian Azmi, Muhammad Dahria
   Terbit : 2013
   Penerbit : Saintikom
   ISSN : 1978-6603
   Deskripsi : Kesalahan dalam pengambilan keputusan bisa berdampak kepada kerugian. Misal dalam perusahaan, keputusan yang diambil oleh pimpinan perusahaan merupakan hasil pemikiran yang harus dilaksanan oleh bawahannya atau mereka yang harus dilaksanakan oleh bawahannya atau mereka yang bersangkutan dengan organisasi yang dia pimpin.

2.2. Artificial Intelligence

Kecerdasan buatan (*Artificial Intelligence*/AI) adalah kecerdasan yang di miliki oleh suatu entisitas buatan. Kecerdasan diciptakan dan dimasukan kedalam suatu mesin (Komputer) agar dapat melakukan pekerjaan seperti yang dapat dilakukan manusia. Beberapa macam bidang yang menggunakan kecerdasan buatan antara lain adalah system pakar dan permainan komputer (game) [1].

2.3. Algoritma A* untuk *Pathfinding*

Permainan Video pada umumnya mere-presentasikan obyek yang dapat dikendalikan atau bergerak sendiri. Mereka digambarkan dapat bergerak dari satu posisi menuju posisi yang berbeda. Tentunya obyek tersebut haruslah mengenali lintasan serta halangan. *Pathfinding* adalah sebuah metode Kecerdasan buatan yang memungkinkan suatu obyek mengenali lintasan dan salah satu algoritma *pathfinding* adalah Algoritma A* (A star).

A* adalah algoritma pencarian generik yang dapat digunakan untuk menemukan solusi untuk banyak masalah, mencari jalan hanya menjadi salah satu dari fungsi Algoritma tersebut. Untuk mencari jalan, algoritma A* berulang kali memeriksa lokasi yang belum dijelajahi yang paling menjanjikan yang pernah dilihatnya. Ketika suatu lokasi dieksplorasi, algoritma itu berakhir jika posisi yang sekarang adalah posisi tujuan; jika tidak, algoritma membuat catatan semua "tetangga" lokasi itu untuk lebih jauh di eksplorasi. A * mungkin merupakan algoritma pencarian jalur paling populer di game AI (Kecerdasan Buatan) [2].

1) Tambahkan simpul awal ke daftar terbuka.
2) Ulangi langkah-langkah berikut:
   a) Cari simpul yang memiliki f terendah pada daftar terbuka. Rujuk ke simpul ini sebagai simpul saat ini.
   b) Alihkan ke daftar tertutup.
   c) Untuk setiap simpul yang dapat dijangkau dari simpul saat ini i. Jika ada di daftar tertutup, abaikan saja.
      i) Jika tidak ada dalam daftar terbuka, tambahkan ke daftar terbuka. Jadikan simpul saat ini sebagai induk dari simpul ini. Catat nilai f, g, dan h dari simpul ini.
      ii) Jika sudah ada di daftar terbuka, periksa untuk melihat apakah ini jalan yang lebih baik. Jika demikian, ubah induknya ke simpul saat ini, dan hitung ulang nilai f dan g.
   d) Berhenti ketika:
      i) Tambahkan node target ke daftar tertutup.
      ii) Gagal menemukan simpul target, dan daftar terbuka kosong.
3) Menelusuri mundur dari simpul target ke simpul awal. Itu adalah jalurnya.

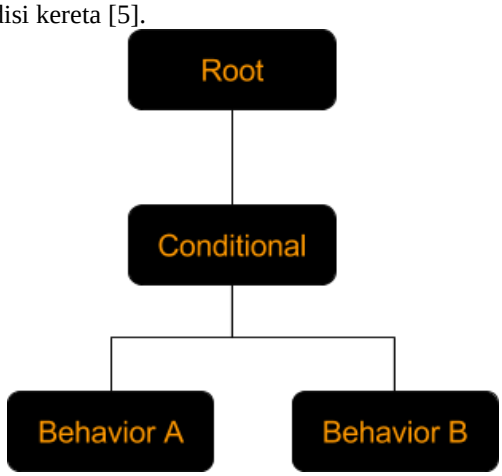
Gambar 1: *Pseudocode A* Algorithm* [3].

2.4. Decision Tree

Decision Tree merupakan penerapan dari teknik pengelompokan yang merupakan proses penentuan suatu fungsi tujuan yang memetakan tiap himpunan *behaviour* ke satu kelompok dari kelas yang didefenisikan sebelumnya. *Decision tree* dapat menemukan hubungan tersembunyi antara sejumlah nilai masukan dengan sebuah nilai keluaran.

Dan *Decision tree* dapat menggabungkan antara pencarian data dan pemodelan data. Dengan kemampuannya untuk mengendalikan proses pengambilan keputusan yang kompleks menjadi lebih sederhana. Dan pengambilan keputusan merupakan suatu proses pemikiran dalam rangka pemecahan suatu masalah untuk memperoleh hasil akhir untuk dilaksanakan [4].

Idenya adalah bahwa perilaku dijalankan satu demi satu atau secara berurutan. Keputusan perilaku mana yang akan dilaksanakan dievaluasi pada tingkat *tick-rate* tertentu, seperti sekali per detik. Pada setiap centang seluruh pohon dievaluasi, dan jika perilaku lain dipilih dari yang terakhir dipanggil, status eksekusi dapat diubah, jika tidak maka perilaku akan terus dijalankan. Seperti terjebak dalam suatu negara dan mengalami perubahan kondisi kereta [5].



Gambar 2: Contoh bagan *Decision Tree* [5].

3. HASIL DAN PEMBAHASAN

3.1. Solusi Pemecahan Masalah

Berdasarkan masalah yang diuraikan sebelumnya, dibuatlah sebuah Permainan dengan AI untuk menerapkan konsep dalam *genre Third Person Shooter*. AI diharapkan mampu bergerak untuk berpindah tempat dan melakukan serangkaian aksi bedasarkan kondisi setiap tokoh (NPC) itu sendiri.

3.2. Perancangan Mekanisme Permainan

Karakter akan bergerak dalam bidang 3 dimensi, dimana pemain memiliki kebebasan mengendalikan tokoh untuk bergerak maju, mundur, menyamping dan melompat. Karakter pemain dilengkapi persenjataan yang menjadi alat utama untuk menyelesaikan permainan, peralatan tambahan dapat ditemukan ketika memasuki pertengahan permainan seperti obat-obatan untuk menyembuhkan karakter. Permainan akan selesai dalam kondisi kalah jika Karakter pemain kehilangan banyak *Health Points*, yaitu sebuah nilai yang menandakan kesehatan karakter. Untuk menyelesaikan permainan, karakter harus mengalahkan musuh, yaitu karakter lain yang bersifat agresif menyerang siapapun.

Setiap Karakter memiliki *State*, yaitu keadaan karakter yang mempengaruhi aksi yang dapat dilakukan. Berikut daftar State yang ada:

1) Normal: Keadaan dimana karakter berdiri di lantai dan tidak terbatasi pergerakannya.

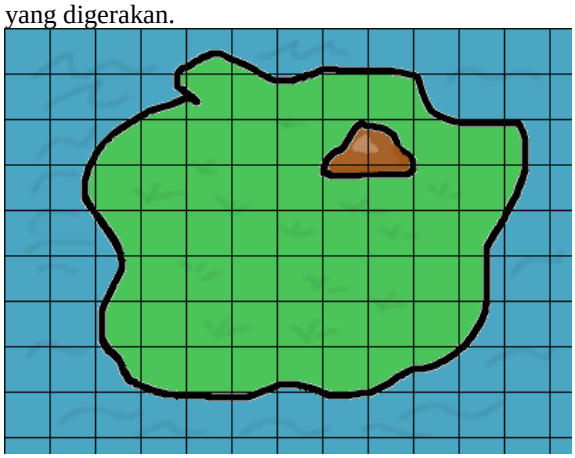
2) OnAir: Keadaan diamana karakter melambung di udara, karakter tidak dapat bergerak atas kehendak karakter itu sendiri (kendali pemain/AI). Pada keadaan ini karakter bergerak bergantung pada hukum gerak fisika (momentum, dorongan & gravitasi)

3) Death: Keadaan dimana *Health Points* karakter mencapai nol, karakter tidak akan aktif lagi di waktu kedepannya sampai permainan selesai, jika karakter pemain mencapai State ini maka permainan berakhir.

3.3. Implementasi *Pathfinding*

Pathfinding berfungsi sebagai penentu jalur dengan jarak sependek mungkin dari posisi awal sampai posisi tujuan. Dengan mengandalkan algoritma A* sebagai penentu jalur, namun algoritma A* harus mengenali wilayah untuk mengidentifikasi daerah mana yang dapat dilewati dan mana yang tidak dapat dilewati karena memiliki penghalang (Obstacle). Untuk itu diperlukan pengidentifikasian wilayah dengan *Node*.

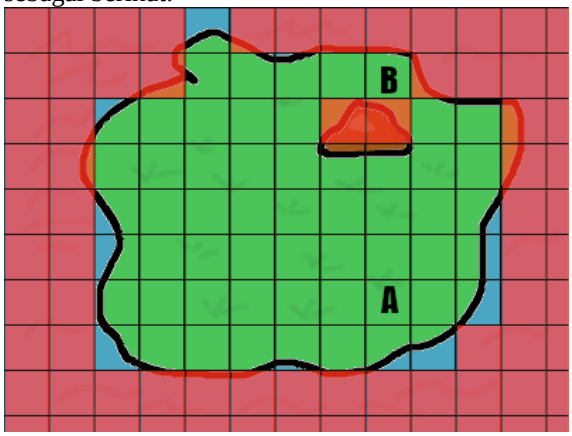
Node adalah pecahan wilayah menjadi banyak bagian yang lebih kecil sehingga dapat diklasifikasi sebagai daerah yang dapat atau tidaknya dilalui obyek, *Node* juga berperan sebagai penentu posisi jalur lintasan karena itu *Node* harus dibagi menjadi banyak bagian dan biasanya seukuran dengan obyek yang digerakan.



Gambar 3: ilustrasi wilayah dengan kotak *grid* sebagai *node*.

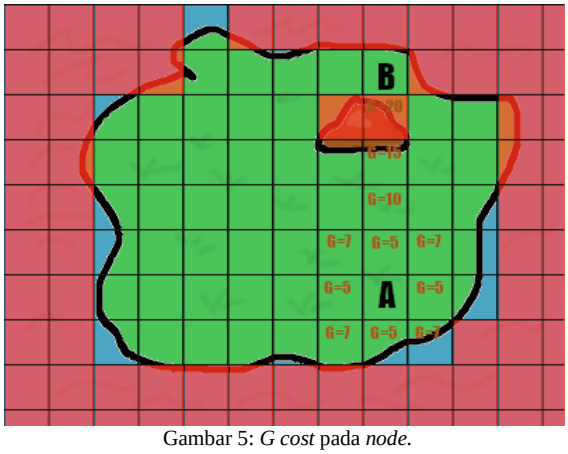
Setiap *node* akan diberi tanda merah jika wilayah itu dianggap tidak dapat dilewati (*unwalkable path*), contoh kasus AI ingin bergerak

dari titik A ke titik B, maka digambarkan ilustrasi sebagai berikut:



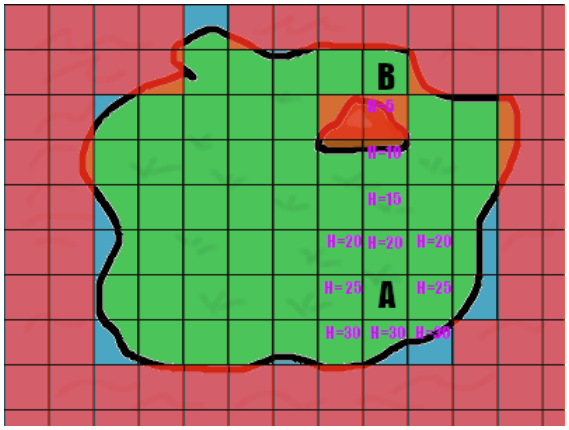
Gambar 4: ilustrasi wilayah dengan *node* yang sudah ditandai.

Setiap langkah diperbolehkan selama tidak berpindah ke *Node* berwarna merah karena sudah ditandai sebagai *Unwalkable Path*. Setiap langkah yang dapat dituju diberikan nilai G dimana G adalah *cost* atau biaya dalam setiap langkah. Nilai G bedasarkan jarak antara *Node* langkah dengan *Node* posisi obyek atau titik awal. Dalam kasus ini diasumsikan diberi nilai 5 untuk setiap langkah vertikal maupun horizontal, dan untuk diagonal diambil dari hasil pitagoras dari nilai horizontal dan vertikal, dimana pada diagonal awal: $\sqrt{(5^2 + 5^2)} = 7$. Hasil data nilai G ini selanjutnya kita gambarkan sebagai berikut:



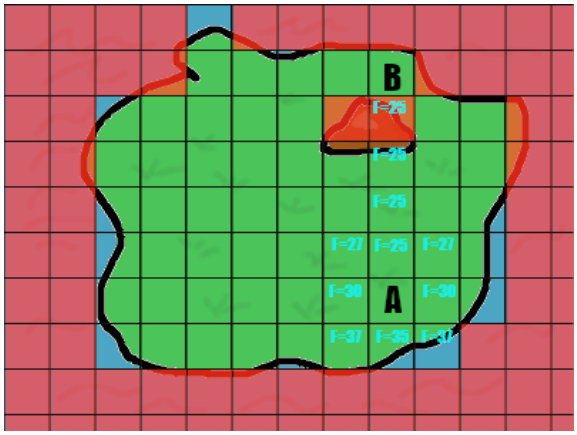
Gambar 5: *G cost* pada *node.*

Biaya Estimasi adalah jumlah jarak dari *Node* menuju posisi tujuan dan disimbolkan dengan "H" (heuristic), berikut beberapa *Node* yang sudah terhitung nilai H:



Gambar 6: *H cost* pada *node.*

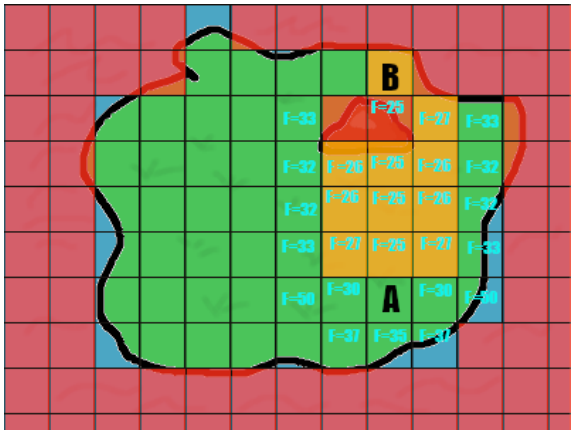
Nilai "G" dan "H" pada setiap *Node* akan dijumlahkan dan Hasil dari penjumlahan nilai "G" dan nilai "H" disimbolkan dengan "F", berikut ilustrasi *Node* yang sudah diberi nilai Akhir:



Gambar 7: *Final cost* pada *node.*

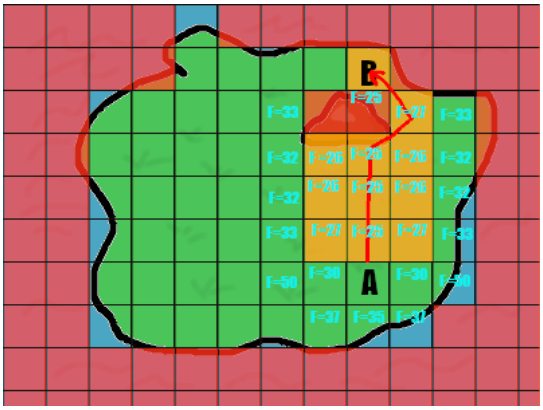
Posisi A merupakan posisi obyek yang akan bergerak ke posisi B, yang merupakan posisi tujuan. *Node* sudah memiliki *Final cost* yang merupakan hasil dari perhitungan algoritma A*. Kemudian Algoritma A* akan menandai *Node* yang memiliki *Final cost* yang paling sedikit, penandaan diasumsikan dengan *Closed Node*, dan *Final cost* dari *Node* yang akan diperiksa adalah *Node* yang bertetangga dengan *Closed Node*. Pada inisiasi proses *Node* yang diperiksa hanya *Node* yang bertetangga, *Node* yang terpilih menjadi *Closed Node* akan diberi warna jingga.

Setiap *Node* yang bertetangga dengan *Closed Node* namun belum memiliki nilai "F" akan dihitung dengan aturan Algoritma A*, setelah itu proses pencarian *Node* dengan nilai "F" terkecil akan diulangi, *Node* yang telah memiliki nilai "F" disebut *Open Node*.



Gambar 7: Hasil keluaran *looping* pencarian *closed node.*

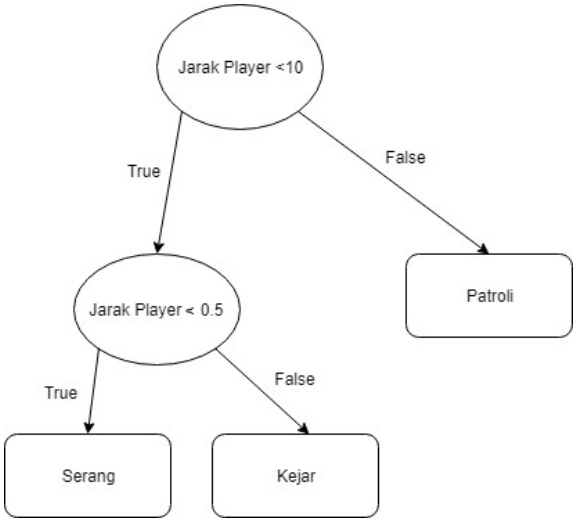
Seluruh *Closed Node* akan diseleksi lagi untuk menentukan *Path Node* atau jalur, dengan penentuannya adalah nilai "F", "H" dan "G". Jika tetangga *Path Node* memiliki nilai F terkecil lebih dari satu, nilai "H" akan dibandingkan diantara keduanya, dan jika nilai "H" terkecil lebih dari satu, maka nilai "G" yang menjadi acuannya, dan jika nilai "G" nilai "G" yang terkecil lebih daru satu maka *Path Node* akan dicabangkan, dan dari jalur yang tercabang akan dihitung total *Final cost* dan dibandingkan dan dipilih nilai yang terkecil.



Gambar 8: Penentuan *Path node.*

3.4. Implementasi *Decision Tree*

Karakter harus dapat bergerak bedasarkan keputusan, karena itu metode *Decision Tree* digunakan untuk menerapkan aksi dari setiap karakter NPC. Pada pembahasan ini *Tree* dirancang secara manual dan ditanamkan ke program (*Hardcoded*), hal ini dilakukan agar karakter NPC dapat memahami mekanisme permainan secara langsung karena kebutuhan Permainan yang harus siap dijalankan tanpa menunggu proses *learning* seperti kebanyakan penerapan metode *Decision Tree* pada Sistem Informasi. Secara ringkas penerapan metode *Decision Tree* untuk pembuatran Permainan Video hanya sebatas struktur dasarnya saja.



Gambar 9: Struktur *Tree* pada AI musuh.

Pada Permainana Video, *Decision Tree* tidak hanya digunakan untuk menentukan pergerakan, namun bisa juga digunakan untuk tugas yang lebih kompleks seperti menentukan kondisi kapan karakter menyerang. Konsep *Decision Tree* pada Permainan Video cukup sederhana namun juga memiliki efektifitas yang tinggi. Karena bentuk *Tree* sudah ditentukan secara *Hardcoded*, maka untuk implementasinya menggunakan *Nested if-else.*

```
1) If (jarak player<10)
   a) If(jarak player < 0.5)
      i. Serang
   b) Else
      i. Kejar()
2) Else
   a) Patrol()
...
```

Gambar 10: *Pseudocode* rancangan *Decision Tree* AI musuh.

3.5. Analisis Kebutuhan Perangkat Lunak

Perangkat lunak atau software merupakan hal yang terpenting dalam mendukung kinerja sebuah sistem. Perangkat lunak digunakan dalam sebuah sistem merupakan perintah-perintah yang diberikan kepada perangkat keras agar dapat saling berinteraksi diantara keduanya. Perangkat lunak yang dibutuhkan sebagai system operasi yang ada pada komputer dan sebagai media penelitian dan perancangan game *third person shooter* ini antara lain.

Tabel 1: Kebutuhan Perangkat lunak.

Software	**Keterangan**
Sistem Operasi	Windows 7
Graphics API	DirectX 10

3.6. Analisis Kebutuhan Perangkat Keras

Untuk menjalankan suatu aplikasi maka diperlukan perangkat keras yang dapat mendukung proses kerja dari sistem itu sendiri. Pada dasarnya

game ini dapat dijalankan di semua perangkat Komputer namun untuk kenyamanan sebaiknya dijalankan di perangkat yang mempunyai spesifikasi sebagai berikut:

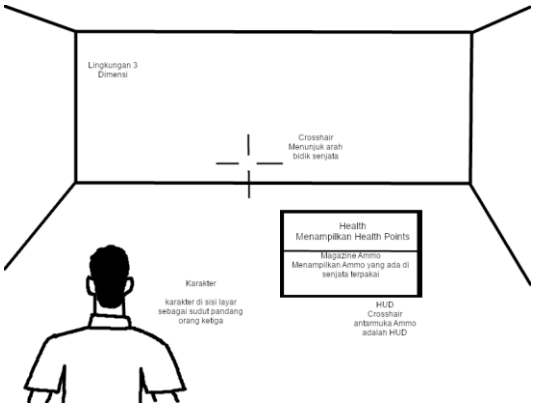
Tabel 2: Kebutuhan Perangkat Keras

Hardware	**Keterangan**
Processor	Dual Core 2.6 GHz
RAM	4GB
Storage	Tersissa 1 GB
Graphic card	1 GB VRAM dengan pixel shader 3.0
Sound	Stereo Channel

3.7. Rancangan Layar

Menu utama merupakan menu awal yang di tampilkan pertama kali pada saat *user* menjalankan *game* ini. Dan Pada Saat permainan ada sebuah sistem antarmuka yang disebut *Head-Up Display* (HUD), sekumpulan informasi tentang progress permainan.

Gambar 11: Rancangan Layar *Menu* Utama



Gambar 12: Rancangan Layar HUD

3.8. Tampilan Layar

Pada bagian ini akan dijelaskan mengenai tampilan layar pada permainan ini mulai dari tampilan pertama dijalankan hingga selesai dijalankan. Selanjutnya akan diberikan penjelasan beserta gambar mengenai tampilan-tampilan yang ada pada game yang telah dibuat.

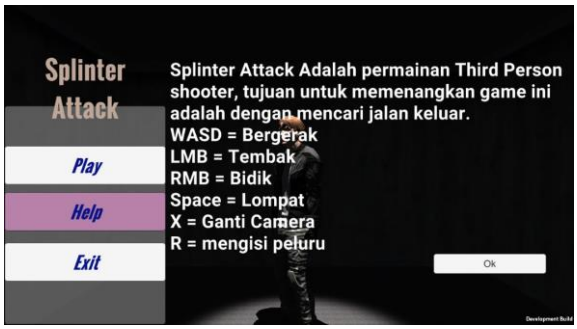
a. Tampilan Layar *Menu* Utama

Tampilan layar dari menu utama akan muncul ketika pertama kali aplikasi dijalankan. Pada *menu* utama terdapat tiga buah menu diantaranya *play* dan *exit*. *Menu Play* berfungsi untuk pindah *scene* menuju permainan. *Menu Help* berfungsi untuk menampilkan informasi terkait cara bermain, *Menu exit* yang berfungsi untuk keluar dari permainan.

Gambar 13: Tampilan Layar *Menu* Utama

b. Tampilan Layar *Help*

Berikut ini adalah tampilan dari *Help*, Teks akan muncul di layar sebagai informasi terkait cara bermain.



Gambar 14: Tampilan Layar *Help*

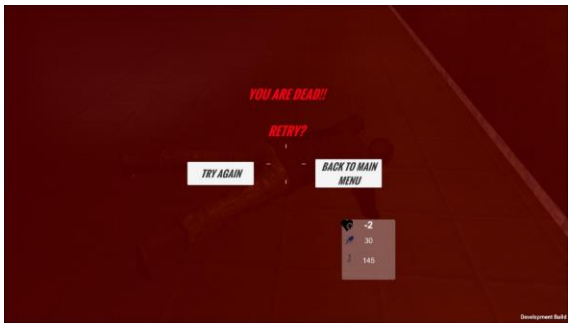
c. Tampilan Layar *Gameplay*

Tampilan Layar *Gameplay* adalah tampilan pada saat *scene gameplay* atau saat permainan berlangsung, menampilkan *user interface* atau *head-up display* yang berisi informasi terkait jalannya permainan, yaitu *Health point*, *magazine ammo,* dan *ammo* cadangan.

Gambar 15: Tampilan Layar *Gameplay* dengan musuh sedang mengejar.

d. Tampilan Layar *Game Over*

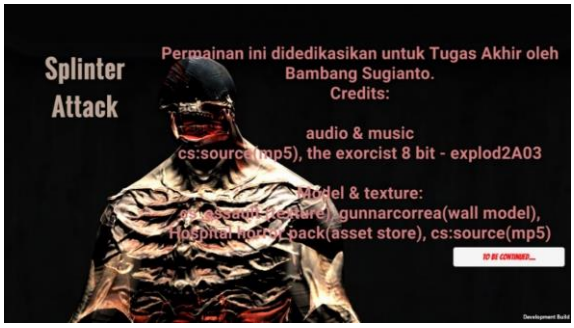
Tampilan ketika pemain kehilangan semua *health points*, maka karakter akan jatuh dengan animasi *Ragdoll* dan menampilkan pilihan pada layar yaitu *try again* dan *back to menu*.



Gambar 16: Tampilan Layar *Game Over*

e. Tampilan Layar *Ending*

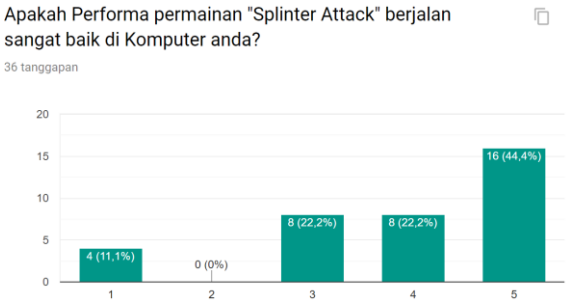
Tampilan ini akan muncul ketika pemain menjangkau pintu keluar. Tampilan ini berisi credits, yaitu daftar orang atau pemilik yang menciptakan Asset untuk permaianan Splinter Attack.



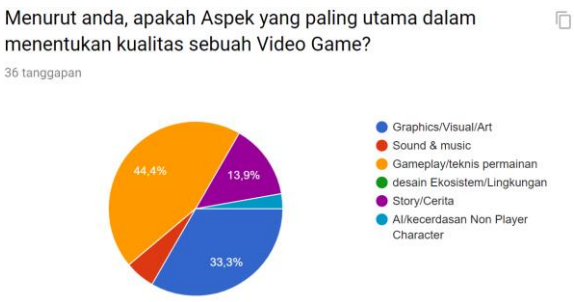
Gambar 17: Tampilan Layar *Ending*

3.9. Hasil Uji Coba

Permainan diuji dengan menggunakan teknik *user experience*, yaitu dengan menggunakan pihak ketiga untuk menjalankan permainan dan memberikan *feedback* dengan cara mengisi kuisioner *User Test Applicant*. Berikut ditampilkan beberapa point pertanyaan untuk menjawab rumusan masalah:



Gambar 18: Hasil UAT mengenai Performa Permainan.



Gambar 19: UAT Mengenai aspek penentu kualias Game

Pada Gambar 19 ditampilkan data mengenai aspek utama penentu kualitas dari sebuah *Video Game*. Pertanyaan pada UAT ini bertujuan untuk menentukan Aspek utama dalam industri hiburan dibidang *Video Game*.

4. KESIMPULAN

Bedasarkan hasil implementasi dan analisis dari Bab sebelumnya, maka dapat disimpulkan bahwa:

1) NPC berjalan dengan bantuan metode *Pathfinding* dengan perhitungan algoritma A*
2) Setiap aksi atau *Behaviour* NPC ditentukan dengan kondisi pada model *Tree* yang sudah dibuat, dengan konsep model *Decision Tree*.
3) Metode *Decision Tree* untuk permainan hanya bersifat *Nested if-else*.
4) Penentuan target spesifikasi untuk menjalankan *game* sebaiknya ditargetkan dengan standar umum spesifikasi komputer pengguna, agar semakin banyak yang dapat memainkan *game* yang artinya membuat cangkupan pengguna menjadi lebih banyak.
5) Mekanisme game third person shooter yang paling utama adalah sudut pandang dan arah gerakan.

5. SARAN

Penelitian Penelitian ini masih banyak kekurangannya, untuk itu diberikan saran agar jika ingin mengembangkan hal serupa dengan penelitian Penelitian ini dapat mencapai lebih baik lagi, berikut sarannya:

1) Perpadukan Algoritma lain dalam menentukan Node untuk melengkapi algoritma A*
2) Buat lebih banyak klasifikasi *Tree* pada metode *Decision Tree* agar *AI Behaviour* lebih bervariasi
3) Buat agar *AI* dapat melewati rintangan dengan melakukan Aksi, hal ini membutuhkan keterkaitan *AI Behaviour* dan *Pathfinding* yang lebih dalam lagi, seperti contohnya AI harus melewati rintangan dengan cara melompat

TERIMA KASIH

Penulis berterima kasih kepada Komunitas Kotak Kreatif atas diberikannya kesempatan

DAFTAR PUSTAKA

[1] Erwinda Y, 2014, Penerapan Metode Heuristic Dalam Pembuatan Game Ping Pong. Penerapan Metode Heuristic Dalam Pembuatan Game Ping Pong, Medan, Pelita Informatika, 168

[2] Cui X & Shi H, 2011, A * -based Pathfinding in Modern Computer Games A * -based Pathfinding in Modern Computer Games, Melbourne, *ResearchGate*, 125

[3] Nilsson Nils J, 1998, Artificial Intelligence: A New Synthesis, San Francisco, Morgan Kaufmann Publishers

[4] Azmi Z. & Dahria M, 2013, Decision Tree Berbasis Algoritma Untuk Pengambilan Keputusan, Medan, Saintikom, 157–164.
[4] Azmi Z. & Dahria M, 2013, Decision Tree Berbasis Algoritma Untuk Pengambilan Keputusan, Medan, Saintikom, 157–164.
[5] Rasmussen J, 2016, Are Behavior Trees a Things of the Past?, [Online] Available: https://www.gamasutra.com/blogs/JakobRasmussen/20160427/271188/Are_Behavior_Trees_a_Thing_of_the_Past.php/ [Accessed 17 May 2019].