Implementation of Goal Oriented Action Planning (GOAP) in First Person Shooter Video Game Titled Gunbreaker Using Godot Engine

Achmad Nabil Rivaldy Rachman[1)], Wahyuni[2)], dan **Ita Arfyanti[3)]**

[1,2,3]Teknik Informatika, STMIK Widya Cipta Dharma
[1,2,3]Jl. M. Yamin, Gn. Kelua, Kec. Samarinda Ulu, Kota Samarinda, Kalimantan Timur 75123
E-mail: nabilrivaldy@gmail.com[1)], wahyuni@wicida.ac.id[2)], ita@wicida.ac.id[3)]

ABSTRACT

Achmad Nabil Rivaldy Rachman, 2025, Implementation of Goal-Oriented Action Planning (GOAP) in a First-Person Shooter Video Game Titled Gunbreaker Using the Godot Engine, Widya Cipta Dharma College of Informatics and Computer Management. Supervisor (I) Wahyuni, S.Kom., M.Kom, Supervisor (II) Ita Arfyanti, S.Kom., M.M. This research was conducted to develop a system that supports video game development in creation of enemy characters. In this research, Goal Oriented Action Planning is used as the artificial intelligence system for characters so that they can perform sequences of actions. This research uses Godot Engine as the main tool for coding and integrating various assets and Blender for creating three dimensional Models. The research method used in this research is Multi Media Development Life Cycle.

***Keywords**: Goal-Oriented Action Planning, Video Game, First-Person Shooter, Godot Engine*

Penerapan *Goal Oriented Action Planning* Dalam *Video Game First Person Shooter* Berjudul Gunbreaker Menggunakan Godot Engine

ABSTRAK

Achmad Nabil Rivaldy Rachman, 2025, Penerapan *Goal Oriented Action Planning* (GOAP) Dalam *Video Game First Person Shooter* Berjudul Gunbreaker Menggunakan Godot Engine, Sekolah Tinggi Manajemen Informatika dan Komputer Widya Cipta Dharma, Pembimbing (I) Wahyuni, S.Kom., M.Kom, Pembimbing (II) Ita Arfyanti, S.Kom., M.M. Penelitian ini dilakukan untuk membuat sistem yang membantu pengembangan video game dalam pembuatan karakter musuh. Di dalam penelitian ini, *Goal Oriented Action Planning* dipakai sebagai sistem untuk *artificial intelligence* pada karakter untuk melakukan urutan aksi yang telah ditetapkan untuk setiap karakter yang dibuat. Penelitian ini menggunakan Godot Engine sebagai alat utama untuk membuat kode dan penggabungan berbagai macam asset dan Blender untuk membuat berbagai macam model tiga dimensi. Metode penelitian yang dipakai untuk penelitian ini adalah *Multimedia Development Life Cycle*.

Kata Kunci: *Goal Oriented Action Planning*, *Video Game*, *First Person Shooter*, Godot Engine

1. PENDAHULUAN

Video game adalah salah satu media hiburan yang banyak dimainkan berbagai kalangan. Ada bermacam-macam hal yang dapat dilakukan agar pemain menyukai dan melanjutkan *video game* tersebut. Salah satunya adalah membuat video game tersebut *replayable.* Cara yang dapat dilakukan adalah membuat tantangan yang cocok untuk pemain salah satu tantangan yang dapat dibuat adalah dengan mendesai *artificial intelligence* untuk *non playable character* yang dilawan oleh pemain.

Godot Engine adalah game engine yang dapat membuat video game dengan grafis 2D dan 3D. Godot Engine menggunakan GDScript sebagai bahasa utama pemrograman utamanya. Di dalam Godot Engine juga ada sistem GDNative yang dapat dipakai untuk menambah bahasa pemrograman lain.

Goal Oriented Action Planning adalah *Artificial Inteligence* (AI) agent untuk menentukan proses pengambilan keputusan yang digunakan didalam video game untuk membantu desainer dalam pembuatan karakter.

2. RUANG LINGKUP

2.1 Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan maka rumusan masalah yang dibahas adalah "Bagaimana cara menerapkan Sistem *Goal Oriented Action Planning* (GOAP) pada *non playable character* di dalam video game yang menggunakan Godot Engine?"

2.2 Batasan Masalah

Dalam penelitian ini ada beberapa batasan yang diberikan yaitu, penelitian ini menggunakan Godot

Engine versi 4; *Goal Oriented Action Planning* hanya diimplementasikan kepada *non playable character* musuh; pembuatan karakter akan dilakukan di grafik 3D; level yang dibuat akan berjumlah 4 level.

2.3 Tujuan Penelitian

Untuk penelitian ini ada beberapa tujuan yang ingin dicapai. Berdasarkan rumusan masalah yang telah ditetapkan, tujuan yang dicapai adalah melakukan penerapan *Goal Oriented Action Planning* untuk *non playable* character pada *video game* menggunakan Godot Engine dari tahap pembuatan hingga menghasilkan *video game* yang dapat dimainkan; membuat sistem *non playable character* yang menggunakan arsitektur *Goal Oriented Action Planning*; menerapkan *non playable character* menggunakan *Goal Oriented Action Planning*; mengeksplorasi konsep *video game design* dengan cara membuat level environment yang akan dipakai untuk percobaan *non playable character* yang telah dibuat.

2.4 Manfaat Penelitian

Manfaat yang diharapkan penelitian ini adalah hasil yang diperoleh dapat dipakai sebagai referensi untuk membantu dalam proses membangun *non playable character* menggunakan Godot Engine di masa depan.

3. BAHAN DAN METODE

Dalam penelitian ini diperlukan suatu landasan konseptual dalam merumuskan definisi yang menunjang kegiatan penelitian, baik berupa teori dasar maupun teori umum.

3.1 Video Game

Menurut Roos dan Dharmawan (2023), *video game* dapat didefinisikan sebagai sebuah permainan yang dimainkan dengan peralatan audiovisual dan dapat didasarkan pada sebuah cerita fiksi. Terlepas dari semua itu video game mempunyai arti adalah sebuat permainan. Jadi singkatnya *video game* adalah perkembangan dari sebuah permainan yang bisa kita mainkan melalui mesin komputer, konsol game, maupun ponsel.

Menurut Roos dan Dharmawan (2023), *First Person Shooter* (FPS) adalah sesuatu game yang ditunjukkan untuk memberikan pemainnya suatu pengalaman baru, biasanya dimainkan dengan betuk *video game* 3D yang bersifat "*first-person-perspective*" atau yang sering diketahui oleh para pemain menampilkan perspektif penglihatan karakter pemain dalam game ke layar monitor. Sering sekali game dalam genre ini hanya menampilkan kedua tangan saja, dengan senjata pilihan para pemain ditunjukan dalam tampilan layar, senjata-senjata inilah yang menjadi bintang utama dan "karakter" dalam *video game* jenis ini.

3.2 Goal Oriented Action Planning

Menurut Nurudin (2024), *Goal Oriented Action Planning* merupakan kecerdasan buatan yang dapat menghasilkan variasi perilaku yang lebih dinamis, kecerdasan buatan ini mengambil keputusan yang mampu membuat karakter tidak hanya melakukan apa yang akan dilakukan tetapi juga menentukan bagaimana karakter melakukannya. *Goal Oriented Action Planning* (GOAP) berfungsi sebagai AI planning yang mendukung *decision-making*.

3.3 Game Engine

Menurut Tero Salmela (2022), konsep *game engine* sudah dipakai tahun 90an saat waktu rilis Doom dari id Software pada tahun 1993. konsep memisahkan fungsi inti dari video game dari konten adalah cara baru pada waktu itu. *Game engine* akan menyediakan sebuah kerangka kerja dari fungsi yang biasanya memakan waktu lebih lama jika dibuat ulang.

3.4 Godot Engine

Menurut Tero Salmela (2022), Godot adalah *game engine* gratis dan *open-source* yang dikembangakan dan dikelola oleh komunitas yang terdiri dari kontributor independen. Dikarenakan sifatnya yang gratis dan open source, Banyak pengguna menggunakan Godot sebagai engine pertama mereka. Sebab itu, banyak pengguna yang mempelajari proses pengembangan video game menggunakan Godot berbagai pengetahuan antar sesama pengembang.

3.5 Multimedia Development Life Cycle

Menurut Riyanto dan Singgih (2015), *Multimedia Development life Cycle* merupakan metode pengembangan sistem yang cocok untuk pengembangan sistem berbasis multimedia. Metode pengembangan sistem *Multimedia Development Life Cycle* terdiri dari enam tahap yaitu:

1. Tahap pengonsepan

   Tahap pengonsepan adalah tahap untuk menentukan tujuan dan kepada siapa multimedia ditujukan (*audience identification*). Selain itu menentukan jenis aplikasi (presentasi, interaktif, dan lain-lain) dan tujuan aplikasi (hiburan, pembelajaran, dan lain-lain).
2. Perancangan

   Perancangan adalah tahap pembuatan spesifikasi yang meliputi arsitektur proyek, gaya, tampilan dan kebutuhan material atau bahan untuk program yang ingin dibuat. Spesifikasi dibuat serinci mungkin sehingga pada tahap berikutnya yaitu material collecting dan assembly, pengambilan keputusan baru tidak akan diperlukan lagi, cukup ini biasanya dengan cara menggunakan *storyboard* atau menggambarkan deskripsi tiap *scene* dengan mencantumkan semua obyek multimedia.
3. Pengumpulan bahan

   Pengumpulan bahan adalah tahap pengumpulan bahan yang sesuai dengan kebutuhan yang dikerjakan. Bahan-bahan tersebut antara lain seprti *clip-art*, *graphic*, animasi, video, audio. Tahap ini dapat dikerjakan secara parallel dengan tahap *assembly*. Namun dapat juga tahap

material collecting dan *tahap assembly* akan dikerjakan secara *linear* dan tidak *parallel*.

4. Pembuatan

Pembuatan adalah tahap pembuatan semua obyek atau bahan multimedia dibuat. Pembuatan proyek didasarkan pada tahap design. Seperti storyboard, bagan alir atau struktur navigasi.

5. Pengujian

Pengujian Akan dilakukan setelah selesai pembuatan (*assembly*) dengan menjalankan proyek apakah ada kesalahan atau tidak. Tahap ini disebut dengan tahap pengujian alpha (*alpha test*) dimana pengujian dilakukan oleh pembuat. Fungsi dari tahap ini adalah melihat hasil pembuatan proyek apakah sesuai dengan yang diharapkan atau tidak, maka akan dibuat tabel pengujian untuk menguji kriteria proyek tersebut.

6. Pendistribusian

Pada tahap ini proyek akan disimpan dalam suatu media penyimpanan. Jika media penyimpanan tidak cukup menampung proyeknya, maka kompresi terhadap proyek itu akan dilakukan. Tahap ini juga dapat disebut sebagai tahap evaluasi untuk pengembangan produk yang sudah jadi agar menjadi lebih baik. Hasil evaluasi ini dapat digunakan sebagai masukan untuk tahap konsep pada produk selanjutnya.

3.6 White Box Testing

Menurut Rony Setiawan (2021), *White box testing* atau yang dapat diartikan menjadi "pengujian kotak putih" adalah metode pengujian yang dilakukan untuk menguji perangkat lunak. Lain halnya dengan *black box testing* yang hanya melihat hasil input dan output dari perangkat lunak, pengujian *white box testing* berfokus pada aliran *input* dan *output* dari perangkat lunak.

Untuk melakukan pengujian ini, penguji/*tester* perlu memiliki kemampuan dalam memahami kode dari suatu program sehingga pengujian ini tidak bisa dilakukan oleh sembarang orang. Berikut adalah teknik yang dipakai untuk melakukan metode *white box testing* untuk perangkat lunak:

1. Basis Path Testing

Basis path testing merupakan metode yang memungkinkan perancang *test case* untuk membuat pengukuran kompleksitas logika dari rancangan prosedural dan menggunakan pengukuran ini sebagai panduan untuk mendefinisikan himpunan basis dari jalur eksekusi. *Test case* yang dibuat untuk menguji himpunan basis dijamin akan meneksekusi setip statement didalam program sekurangnya sekali pada saat pengujian.

2. Flow Graph

Flow graph merupakan notasi sederhana untuk merepresentasi *control flow*.

3. Cyclomatic Complexity

Cyclomatic complexity digunakan untuk mengetahui jumlah jalur yang perlu dicari. Cyclomatic complexity adalah metric software yang menyediakan ukuran kuantitatif dari kompleksitas logika program. Nilai yang dihitung bagi *cyclomatic complexity* menentukan jumlah jalur yang independen dalam kumpulan basis suatu program dan memberikan jumlah tes minimal yang harus dilakukan untuk memastikan bahwa semua pernyataan telah dieksekusi sekurangnya satu kali.

3.7 *Beta Testing*

Menurut T, Menora., dkk (2023) menyatakan bahwa beta testing merupakan pengujian yang dilakukan dari perspektif pengguna. Pengujian ini dilakukan atas dasar ingin mengetahui seberapa besar tingkat penerimaan pengguna sebelum aplikasi benar-benar dirilis. Hasil perhitungan tingkat penerimaan pengguna tersebut nantinya akan digunakan sebagai masukan untuk melakukan perbaikan aplikasi di masa mendatang.

4. PEMBAHASAN

Bagian ini menjelaskan hasil berdasarkan metode penelitian yang dipakai yaitu *Multi Media Development Life Cycle*. Hasil dari metode yang dipakai terdiri dari tahap pengonsepan, perancangan, pengumpulan bahan, pembuatan, pengujian dan pendistribusian.

4.1 Tahap Pengonsepan

Pada fase ini dilakukan penrancangan konsep untuk karakter yang ingin dibuat. Dalam video game Gunbreaker dengan genre *first person shooter*, pemain akan diberikan tantangan dalam bentuk karakter musuh yang akan berusahan untuk memusnahkan pemain dengan cara yang berbeda. Untuk penelitian ini dibuat 6 jenis karakter musuh yang akan dilawan oleh pemain. 4 Karakter ini diberikan aksi (*action*) dan tujuan (*goal*) yang akan diatur oleh sistem *Goal Oriented Action Planning*. Sementara 2 karakter lainnya tidak menggunakan *Goal Oriented Action Planning*.

4.2 Perancangan

Penggunaan GOAP dalam *video game* ini adalah agar karakter musuh dapat bergerak dan melakukan bermacam aksi yang akan dipilih secara otomatis berdasarkan nilai yang ada didalam *variabel state*. Di dalam variabel ini ada *array* yang terdiri dari nilai boolean yang akan berubah sesuai dengan kondisi karakter musuh. Berdasarkan *variabel* tersebut akan dipilih *class goal* yang akan menyusun *action* apa yang akan dijalankan. *action* yang telah disusun berdasarkan *goal* yang terpilih akan membuat sebuah *plan*. Dapat dilihat pada tabel 4.1 karakter apa saja yang dibuat menggunakan GOAP.

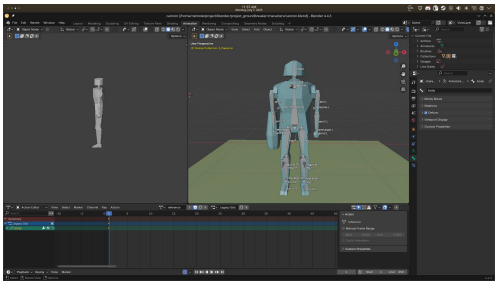
Tabel 4.1 Karakter yang dibuat menggunakan GOAP

No.	Nama Karakter	Goal	Action
1.	Cannon Shooter	- Idle - Patrol - ShootTarget - AttackTarget	-GoToTarget -LocateTarget -Shoot -GoToPosition

2.	Chaser	- Idle - Patrol - AttackTarget	-GoToTarget -Attack -GoToPosition
3.	Shotgunner	- Idle - Patrol - ShootTarget - PunchTarget	-GoToTarget -LocateTarget -Shoot -GoToPosition
4.	Giant	- Idle - Patrol - ShootTarget	-GoToTarget -LocateTarget -Shoot -GoToPosition

4.3 Pengumpulan Bahan

Untuk melengkapi visual yang didalam *video game* maka di dalam fase ini, akan ada beberapa model tiga dimensi yang dibuat. Model-model ini akan terdiri dari beberapa elemen seperti model tiga dimensi untuk karakter, senjata yang dipakai oleh pemain dan level yang akan dilewati oleh pemain. Model yang sudah dibuat juga diberikan animasi agar *video game* yang dibuat dapat lebih dinikmati oleh pemain. Semua proses pembuatan model, *rigging*, *material* dibuat menggunakan aplikasi Blender yang dapat dilihat pada gambar 4.1.



Gambar 4.1 Proses Pembuat Model

4.4 Pembuatan

Pada fase ini akan dilakukan penggabungan semua bahan yang telah di siapkan. Untuk memulai penggabungan, semua bahan model tiga dimensi akan di-*import* ke dalam Godot Engine agar dapat dipakai dalam *script* yang telah dibuat. *File* 3d yang telah dibuat akan dipindahkan ke folder projek. Kemudian file tersebut akan diproses oleh Godot Engine.

Proses ini akan memerlukan fitur yang sudah ada dalam aplikasi Blender yaitu proses *import* menggunakan GLTF. *File format* ini dapat membantu dalam proses penggabungan asset karena adanya sistem drag and drop yang sangat leluasa. Semua animasi dan gambar *texture* yang sudah di-*mapping* ke model dapat dimasukkan kedalam Godot Engine. Selain itu Blender juga membantu dalam pembuatan level karena fitur fiturnya yang sangat banyak dan fleksibel. Animasi yang sudah di-import kemudian dapat dimainkan didalam Godot Engine sesuai dengan animasi yang sudah dibuat di dalam Blender.

Untuk pembuatan class *Goal Oriented Action Planning*. *Class* yang dibuat adalah *class goal* dan *action*. *Class goal* akan menyimpan data yang digunakan untuk membuat susunan dari *action*. Kemudian *class action* akan berisi *script* yang akan menjalankan karakter. Kemudian akan dibuat class utama *Goal Oriented Action Planning* yang berisi algoritma untuk mencari goal yang valid dan menyusun action berdasarkan goal tersebut. Akan dibuat node didalam obyek karakter seperti terlihat di gambar 4.2

Gambar 4.2 *Class Goal* dan *Action* Dalam Obyek Karakter

Setelah semua sistem telah digabungkan dengan asset yang sesuai. Dapat dilanjutkan untuk mulai memainkan game dari level awal hingga level akhir. Level – level ini di desain dengan cara menaruh karakter musuh di tempat tertentu agar membuat alur yang dapat memberikan tantangan kepada pemain. Pada gambar 4.3 dapat dilihat saat *video game* sedang dimainkan.

Gambar 4.3 Semua Asset yang Telah Digabungkan

4.5 Pengujian

Setelah semua bahan telah dimasukkan kedalam Godot Engine dan pembuatan *script* berdasarkan konsep dan rancangan telah selesai. Maka, akan dilakukan percobaan atau testing untuk menjamin kualitas dari penelitian ini. Akan dilakukan pengujian *white box* dan *black box*.

4.5.1 Pengujian *White Box*

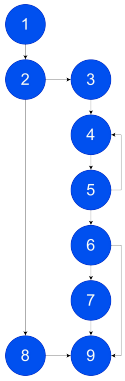
Dalam *white box* testing akan dilihat *script* yang dibuat dan dicoba apakah *script* itu berjalan dengan baik. Perlu dilihat apakah *script* yang telah disambungkan ke obyek dalam *video game* memberikan timpal balik yang sesuai. Juga perlu diperhatikan jika muncul *error* yang penting. Logika yang ada dalam *script* akan diteliti untuk memastikan apakah fungsi secara keseluruhan telah berjalan sesuai ekspektasi. Untuk memastikan *script* berjalan dengan lancar, diperlukan testing proses yang pasti akan berinteraksi dengan pemain. Hal ini dilakukan

karena ada beberapa variasi dari perilaku pemain yang pasti interaksinya dengan proses yang sedang berjalan dapat diperhitungkan.

```
func _physics_process(delta:float) -> void:
1   if state['is_in_animation'] == false:
2       if goal != current_goal || current_goal ==
        null:
            current_goal = get_best_goal()

3       _plan_step = 0
        blackboard = {
              "position" : position,
              "health" : $Health.health,
              "target" : $Detection._target,
        }
4       for s in state:
5           blackboard[s] = state[s]

6       if current_goal != null:
7           _plan =
            get_plan(current_goal,blackboard)


    elif current_goal != null:
8       follow_plan(_plan,delta)
9   end
```

Gambar 4.4 Proses Utama Script GOAP

Proses pada gambar 4.4 adalah proses utama yang berjalan pada *Goal Oriented Action Planning.* Proses ini akan selalu diproses selama karakter masih aktif. Berdasarkan script ini akan dibuat *flow graph* yang terlihat pada gambar 4.5.



Gambar 4.5 *Flow Graph* Proses GOAP

Berdasarkan *flow graph* tersebut, jumlah *node* adalah 9 dan jumlah *edge* adalah berjumlah 11. Maka didapat rumus V(g) = 11 – 9 + 2. Hasil perhitungan dari rumus tersebut adalah V(g) = 4. Hal ini berarti ada 4 jalur yang perlu diuji untuk memastikan semua kemungkinan alur logika. berdasarkan jumlah jalur yang ditentukan. Dapat dibuat tabel uji coba pada tabel 4.2

Tabel 4.2 Test Case Proses GOAP

Jalur yang diuji	Input	Hasil yang diharapkan	Keterangan
1-2-3-4-5-6-7-9	Variabel goal dan current_goal	Proses pencarian goal dijalankan	valid
1-2-3-4-5-4-5-6-7-8-9	Variabel state	For loop untuk memasukkan array state kedalam array blackboard	valid
1-2-3-4-5-6-9	Variable current_goal	Melewati proses pembuatan *plan* dalam fungsi get_*plan*()	valid
1-2-8-9	Variabel goal dan current_goal	Melewati semua pencarian goal dan pembuatan *plan*	valid

4.5.2 Pengujian Black Box

Tabel 4.3 Kuisioner Beta Test

No	Pertanyaan	Penilaian				
		SM	M	CM	TM	STM
1.	Apakah *game* berjalan dengan baik?	11	2	1		
2.	Apakah karakter musuh pada *game* berjalan dengan baik?	11	3			
3.	Bagaimana Penilaian anda untuk *gameplay* keseluruhannya?	6	8			
4.	Bagaimana Penilaian anda untuk sistem movement playernya?	11	3			
5.	Bagaimana peneliaian anda untuk pilihan senjata yang diberikan?	6	6	2		
6.	Bagaimana anda untuk animasi dan suara dari senjata yang ada?	8	3	3		
7.	Bagaimana penilaian anda untuk level design-nya?	8	5	1		
8.	Bagaimana penilaian anda untuk variasi musuh yang dilawan?	5	5	3	1	
9.	Bagaimana penilaian anda untuk sound effect yang dipakai di dalam *game*?	6	6	2		
10.	Bagaimana penilaian anda untuk music yang ada di dalam *game*?	7	6	1		

Berdasarkan tabel 4.3, Jumlah pertanyaan yang diberikan pada kuisioner berjumlah 10 pertanyaan dengan jumlah penilaian tertinggi yaitu 5 dan terendah yaitu 1 yang di representasikan dengan kata Sangat Memuaskan, Memuaskan, Cukup Memuaskan, Tidak Memuaskan, Sangat Tidak Memuaskan. Rumus yang dipakai untuk menghitung kuisioner menggunakan skala likert (Rumus Index % = Total Nilai / Nilai Tertinggi * 100). Dari hasil kuisioner dapat dihasikan perhitungan sebagai berikut :

Total Nilai = (Total Pemilih * Nilai):

= (79 * 5) + (47 * 4) + (12 * 3) + (1 * 2) + (0 * 1) = 395 + 188 + 36 + 2 + 0 = 621

Skor Tertinggi = (Nilai Tertinggi * Jumlah Pertanyaan * Jumlah Responden) :

= 5 * 10 * 14 = 700

Hasil Akhir = Total Nilai / Skor Tertinggi * 100 = 621/700 * 100 = 88.7

Dari hasil kuisioner yang telah dilakukan diperoleh nilai hasil akhir berjumlah 88,7.

4.6 Pendistribusian

Setelah melakukan langkah pengujian dan memastikan tidak ada *error* yang fatal, dapat lanjutkan untuk melakukan pembagian *video game* Groundbreaker untuk dapat dimainkan oleh pemain yang lebih luas. Sekarang *file video game* yang telah ter-*compile* akan dibagikan menggunakan link google drive yang dapat dibuka secara publik (https://drive.google.com/file/d/1NMK-kAKfgSnNfdj2xV6IzU5l9tTJsDYK/view?usp=sharing).

5. KESIMPULAN

Berdasarkan pembahasan yang telah dijelaskan dari bab sebelumnya, dapat diambil kesimpulan sebagai berikut :

1. *Goal Oriented Action Planning* dapat membantu dalam pembuatan karakter yang simpel tapi memiliki aksi yang banyak.
2. Adanya kesulitan saat diawal pembuatan karakter karena susunan aksi tidak berjalan sesuai yang diinginkan.
3. Ada penurunan performa ketika memakai untuk karakter yang banyak.

6. SARAN

Adapun saran-saran yang dapat diberikan untuk penelitian selanjutnya. Saran tersebut adalah sebagai berikut :

1. Dari salah satu kuisioner memberikan nilai kurang dalam variasi musuh, hal ini dapat dijadikan saran untuk meneruskan pengembangan untuk variasi musuh.
2. Agar pembuatan karakter diawal lebih mudah, perlu cara agar ada susuan aksi yang pasti berjalan sesuai dengan urutan yang diinginkan
3. Dikarenakan proses perulangan yang sangat banyak saat GOAP berjalan, akan ada beban dalam performa jika karakter yang dijalankan berjumlah banyak. Hal ini perlu ditangani dengan *optimization* agar fungsi yang ada berjalan secara efisien.

7. REFERENSI

Salmela, T. (2022). *GameDevelopment Using the Open-Source Godot GameEngine*.

Kusuma, R. C., Liliana, L., & Juwiantho, H. (2021). *Penerapan Metode Goal Oriented Action Planning untuk Agent AI pada Turn Based Tactics Video Game*.*Jurnal Infra*,*9*(2), Article 2.

Wolf, M. J. P., & Baer, R. H. (2002). *The Medium of the Video Game*(1st edition). University of TexasPress.

Studiawan, R., Hariadi, M., & Sumpeno, D. S. (2018). *Strategi Adaptif Kelompok di Permainan Taktik Menggunakan Goal-Oriented Action Planning*.

Orkin, J. (2006). *Three States and a Plan: The A.I. of F.E.A.R.*

Orkin, J. (t.t.). *Applying Goal-Oriented Action Planning to Games*.

Rabin, S. (2008). *AIGameProgramming Wisdom 4*. Course Technology, Cengage Learning.*https://books.google.co.id/books?id=N3AJGgAACAAJ*

Bradfield, C. (2018). Godot Engine *GameDevelopment Projects: Build five cross-platform 2D and 3D games with Godot 3.0*. Packt Publishing Ltd.

Santos, R. (2025).*Godot C# Package move to .NET 8.*https://godotengine.org/article/godotsharp-packages-net8/

Booker, B. M. (2021). *Common Terms New Dungeons & Dragons Players Should Learn.*https://www.dndbeyond.com/posts/1026-common-terms-new-dungeons-dragons-players-should

Slavicsek, B., & Baker, R. (2005). *Dungeons & Dragons For Dummies (1st edition)*.

Setiawan, R. (2021, November 19). White Box Testing untuk Menguji Perangkat Lunak. Dicoding Blog. https://www.dicoding.com/blog/white-box-testing/

Godot Docs(2025).GDScript reference. https://docs.godotengine.org/en/stable/tutorials/*scripting*/gdscript/gdscript_basics.html

Ponuthorai,P. K., & Loeliger, J. (2022). *Version Control with Git: Powerful Tools and Techniques for Collaborative Software Development (3rd edition). O'Reilly Media.*

Stallman, R. (n.d.). *Linux and GNU - GNU Project—Free Software Foundation. Retrieved May 6, 2025, from https://www.gnu.org/gnu/linux-and-gnu.en.html*

Blender Manual (2025). SupportedVideo & Audio Formats. https://docs.blender.org/manual/en/4.4/*files*/media/video_formats.html

Menora, T., Primasari, C. H., Wibisono, Y. P., Sidhi, T. A. P., Setyohadi, D. B., & Cininta, M. (2023). Implementasi Pengujian Alpha dan Beta Testing Pada Aplikasi *Game*lan Virtual Reality. KONSTELASI: Konvergensi Teknologi Dan Sistem Informasi, 3(1), 48–60. https://doi.org/10.24002/konstelasi.v3i1.6625

Dharmawan, E. A., & Roos, J. R. M. (2023). RANCANG BANGUN APLIKASI VIDEO GAME FIRST PERSON SHOOTER MENGGUNAKAN ENGINE UNITY. *JURNAL SIMETRIK*, *13*(1), 661–668. https://doi.org/10.31959/js.v13i1.1506

Sampe, D., Tulenan, V., & Paturusi, S. (2023). Aplikasi Game First Person Shooter Multiplayer Pertempuran Manado 1942: Multiplayer First Person Shooter Application Battle Game. *Jurnal Teknik Informatika*,*18*(4), 181–190. https://doi.org/10.35793/jti.v18i4.51338