RANCANG BANGUN *NON-PLAYABLE CHARACTER* DALAM *GAME* BERBASIS 3D MENGGUNAKAN METODE ALGORITMA A*

Ismail Abdurrozzaq Zulkarnain[1], Indah Puji Astuti[2], Ridho Fadhullah Lava Turuzi [3]

[1,2 3] Jurusan Teknik Informatika,Fakultas Teknik, Universitas Muhammadiyah Ponorogo
[1] ismail@umpo.ac.id, [2] indah@umpo.ac.id, [3] lvaturuzi@gmail.com

Abstrak

Game adalah sebuah permainan yang dimainkan dengan tujuan untuk bersenang-senang atau *refreshing* dengan memiliki beberapa aturan dalam melakukan permainan di dalamnya. Ada 2 jenis *game* yang dilihat dari sudut pandang pemain, yaitu NPC (*Non-Playable Character*) dan MC (*Main Character*). Dalam *game*, terdapat karakter *Non-Playable Character* yang berfugnsi untuk dapat membantu atau meramaikan suasana dalam permainan. Tidak dikendalikan oleh pemain, melainkan oleh program komputer yang disebut *Artificial Intelligence*. Dimana sistem ini memungkinkan *Non-Playable Character* melakukan tindakan atau pergerakan tertentu dalam suatu permainan. *Navmesh* merupakan sebuah komponen yang dapat mendukung dalam menentukan pencarian jalur untuk menuju ke target atau titik tujuannya. Algoritma A* merupakan sebuah algoritma yang digunakan untuk menentukan jalur terpendek dari suatu titik menuju titik lainnya dengan mengkalkulasikan seluruh jalur yang memungkinkan untuk dilalui kemudian mengambil nilai jalur terpendek dari salah satu jalur tersebut sebagai jalur yang akan di lalui oleh *Non-Playable Character*. Peforma *Non-Playable Character* akan sangat dipengaruhi oleh metode atau algoritma yang digunakan. Pemilihan metode atau algoritma yang kurang tepat didalam mengimplementasikan pada *Navmesh* akan mengakibatkan beberapa kegagalan atau keterlambatan dalam melakukan tindakan. Maka dari itu dibuatlah perbandingan Algoritma A* dengan metode lain pada *Navmesh* pada *game* 3D untuk menemukan metode atau algoritma terbaik.

Kata Kunci: *Game,* Algoritma A*, Navmesh

1. Pendahuluan

Game adalah sebuah permainan yang dimainkan dengan tujuan untuk bersenang-senang atau *refreshing* dengan memiliki beberapa aturan dalam melakukan permainan di dalamnya. Dalam *game*, terdapat karakter *Non-Playable Character* (NPC) yang dapat membantu atau meramaikan suasana dalam permainan (Junanto et al., 2020). *Game* telah menjadi bagian dari kehidupan sehari-hari kita, dan terus berkembang dari waktu ke waktu. Di era modern, industri *game* semakin berkembang dengan pesat dan menjadi kebutuhan esensial, terutama bagi anak-anak, remaja, dan pemuda (Junanto et al. (2020).

Meskipun begitu sering terdapat kejadian saat NPC pada *game* tersebut tidak dapat bergerak atau mengalami kondisi gagal dalam menjalankan tugasnya pada saat *game* sedang berjalan. NPC *(Non-Playable Character)* adalah suatu karakter dalam permainan yang memiliki peran dalam game, namun karakter ini tidak dapat dijalankan oleh player. Entitas ini dapat berbentuk manusia, hewan, atau robot dan mampu melakukan tindakan yang mirip dengan tindakan yang dilakukan oleh pemain. NPC tidak dikendalikan oleh pemain, melainkan oleh program komputer yang disebut *Artificial Intelligence* (kecerdasan buatan) (Safitra et al., 2020).

Performa NPC akan sangat dipengaruhi oleh metode atau algoritma yang digunakan pada NPC tersebut saat menerima perubahan kondisi, beradaptasi dengan kondisi yang sedang dihadapi.

Algoritma A* merupakan sebuah algoritma yang digunakan untuk menentukan jalur terpendek dari suatu titik menuju titik lainya dengan mengkalkulasikan seluruh jalur yang memungkinkan untuk dilalui yang kemudian diambil nilai terpendek dari salah satu jalur tersebut sebagai jakur yang di lalui (Wildan et al., 2020).

Navmesh merupakan package dari aplikasi game engine Unity, sebuah komponen yang dapat mendukung NPC dalam penentuan pencarian jalur untuk menuju ke titik tujuanya (Iskandar, n.d.).

Berdasarkan latar belakang di atas didapatkan sebuah permasalahan berupa metode manakah yang palik efektif untuk mendukung pergerakan untuk NPC dalam menjalankan fungsinya ketika menghadapi suatu perubahan kondisi. Untuk menyelesaikan masalah tersebut maka peneliti memutuskan utuk menggunakan algoritma A* dan membandingkan algoritma tersebut dengan komponen Navmesh sebagai media yang digunakan sebagai algoritma yang bertugas untuk menjadi penentu arah gerak NPC tersebut dalam sebuah game

3D yang akan di bangun melalui aplikasi game engine Unity dan berjalan pada Platform Windows.

2. Metode Penelitian

2.1 Studi Pustaka

Agar penelitian dapat berjalan dengan baik dengan disertai landasan dan teori yang kuat, pengumpulan data merupakan suatu hal yang penting guna memahami dasar penelitian.

2.2 *Concept*

Game ini direncanakan akan dapat dijalankan di sistem operasi windows dan akan dibangun dengan menggunakan beberapa aplikasi seperti Unity3D sebagai *game engine* dari *game* yang akan dibangun, Visual Studio *Code* sebagai aplikasi *text editor*, Blender sebagai sumber dari asset 3D, Krita sebagai pembuat *palette* warna pada asset.

2.3 *Design*

Perancangan pada penelitian ini akan dibagi menjadi beberapa bagian penting guna memudahkan pembangunan game ini. Bagian tersebut sebagai berikut:

1. *Player* (pemain) memiliki misi atau tugas yang harus diselesaikan, yaitu sebagai berikut:
   - Jika *player* telah melakukan kontak dengan NPC, kemudian dari kondisi tersebut didapati *hitpoint*/nyawa dari player lebih dari 0 maka *playe*r akan kembali kedalam kondisi untuk menyelesaikan misi. Jika misi masih tersedia maka misi akan dilanjutkan, namun bila misi sudah habis maka *player* dianggap menang.
   - Jika *player* telah melakukan kontak dengan NPC, kemudian dari kondisi tersebut didapati hitpoint/nyawa *player* sama dengan 0 maka *player* dianggap kalah.
2. NPC memiliki beberapa aturan, yaitu sebagai berikut:
   - Jika target tidak ada dalam jangkauan pandang dan serang, maka NPC akan masuk dalam kondisi berpatroli.
   - Jika target masuk dalam jangkauan pandang namun tidak masuk dalam jangkauan serang, maka NPC akan masuk dalam kondisi mengejar.
   - Jika target masuk dalam jangkauan pandang dan serang, maka NPC akan masuk dalam kondisi menyerang.
   - Jika kondisi hitpoint NPC lebih dari 0, maka NPC akan tetap menjalankan kondisi patrol, menngejar, dan menyerang.
   - Jika kondisi hitpoint NPC kurang atau sama dengan 0 maka NPC akan mati atau kalah.

2.4 Material Collecting

Perencanaan bahan yang akan dibuat dan dikumpulkan adalah objek 3D beserta audio, background, dan pendukung lain.

2.5 Assembly

Dalam tahap ini digunakan aplikasi Unity3D dan Visual Studio *Code*, yang mana keduanya memiliki peran utama berupa menggabungkan asset yang tersedia dan menggabungkanya sehingga menjadi sebuah *game.*

2.6 Testing

Dalam tahap ini peneliti menggunakan metode *whitebox* dan *blackbox*. Pengujian *whitebox* merupakan pengujian yang berfokus kepada fungsi internal yang terdapat dalam suatu algoritma, aplikasi, maupun objek dalam suatu aplikasi. *Blackbox* menguji sistem berdasarkan keluaran yang dihasilkan dari data masukan tanpa memperhatikan proses internal.

2.7 Distribution

Tahap yang bertujuan untuk menyebarkan dan menyampaikan produk yang telah dibuat dengan menggunakan langkah sebelumnya kepada pengguna.

3. Pembahasan

3.1 Assembly

3.1.1 Pembuatan Hierarki

Hierarki pada *game* dimaksudkan untuk menata susunan *object* yang ada dalam *game* supaya lebih tersusun rapi, yaitu sebagai berikut:

a. Hierarki A*

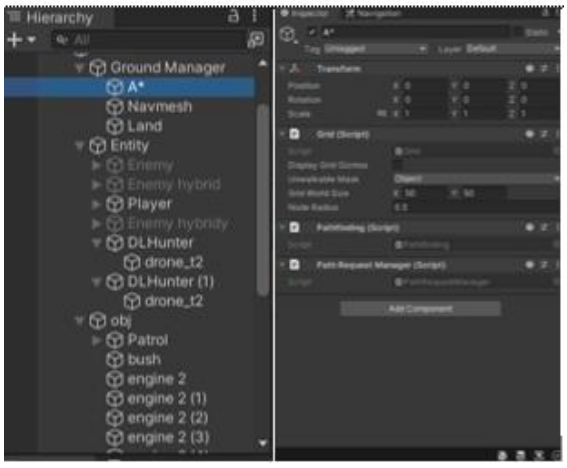
   Gambar 1 adalah gambar hierarki dari algoritma A* yang tampil pada *Unity.*

b. Hierarki Navmesh

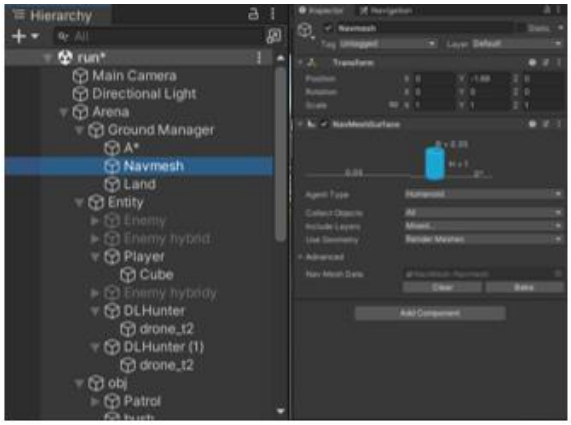
   Berikut Gambar 2 adalah gambar hierarki dari Navmesh yang tampil pada aplikasi *Unity*.

c. Hierarki NPC

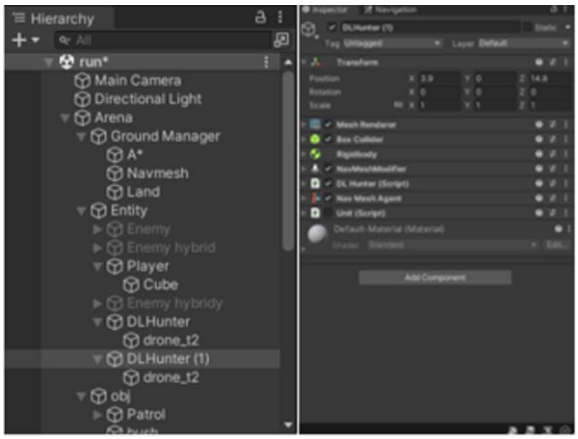
   Gambar 3 merupakan gambar hierarki dari Non-Playable Character.



Gambar 1. Hierarki A*



Gambar 2. Hierarki Navmesh



Gambar 3. Hierarki NPC

3.1.2 Pembuatan *Script*

Script pada *game* merupakan kumpulan perintah, kondisi, dan aksi yang berada dalam satu *file* yang akan digunakan untuk menjalankan *game* dan algoritma yang akan berjalan dalam *game* tersebut. Berikut adalah *script game* yang digunakan, yaitu sebagai berikut:

a. *Script* pada *Player*
   Script pada *player* berfungsi untuk menggerakan objek yang bertindak sebagai player dengan menggunakan keyboard.
b. *Script* pada NPC
   Script pada NPC merupakan *script* yang digunakan oleh untuk dapat memainkan peranya dalam *game* ini. *Script* ini memiliki beberapa *statement*(kondisi) yang berupa *Patrol* (Berpatroli), *Chase* (mengejar), dan *Attack* (menyerang).
c. *Script* pada A*
   Merupakan *script* yang berfungsi untuk melakukan perhitungan menggunakan algoritma A* dengan memindai semua jalur yang dapat dilalui dari titik berangkat sampai ke titik tujuan yang kemudian dikalkulasikan untuk mendapatkan jalur yang terbaik dan nantinya jalur tersebut akan digunakan oleh NPC.

3.1.3 Penyusunan *Component*

Penyusunan *component* merupakan penataan terhadap komponen apa saja yang akan di lekatkan pada suatu objek pada *game*, hal ini dapat berupa *Mesh Renderer*, *RigdBody*, *Script* atau lain sebagainya.

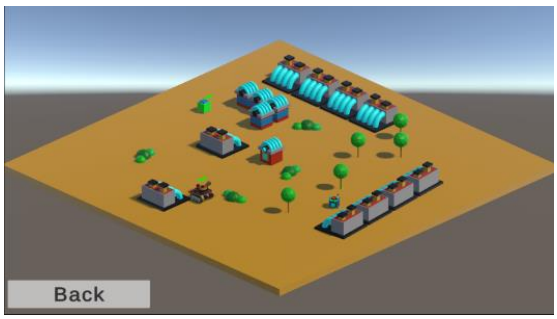
a. Susunan komponen *Player*
   Pada susunan *component* player terdapat beberapa *component*, yaitu *Mesh filter*, *Box Collider*, *RigidBody*, *NavMesh Modifier*, *Navmesh Agent*, *Transform Function (script)*.
b. Susunan komponen NPC
   Komponen pada NPC adalah sebagai berikut, yaitu: *Mesh filter*, *Box Collider*, *RigidBody*, *NavMesh Modifier*, *DLHunter (Script)*, *Navmesh Agent*, *Unit (Script)*
c. Susunan komponen A*
   Susunan pada A* adalah sebagai berikut, yaitu *Grid (Script)*, *node* (kotak) *Pathfinding (Script)*, *Path Request Manager (Script)*.

3.1.4 *Tampilan UI pada Game*

Pada tampilan awal sebagai dashboard game ini adalah ada pemilihan menu game yang terdiri dari *Play*, *Level Select*, dan *Quit.*

Gambar 4. Tampilan *Main menu*

Gambar 4 merupakan tampilan menu utama yang nantinya akan memberikan beberapa pilihan kepada player saat masuk ke dalam *game* seperti tombol "*Play"* yang langsnug mengarahkan kepada permainan, *"Level Select"* yang mengarahkan kepada beberapa pilihan level yang berbeda, "*Help*" untuk info lebih lanjut mengenai permainan, dan *"Quit"* untuk keluar. Dan berikut Gambar 5 merupakan tampilan saat dalam permainan:



Gambar 5. Tampilan Gameplay

3.2 *Testing*

3.2.1 Pengujian *Whitebox*

Pada bagian ini akan disajikan hasil pengujian yang telah dilakukan, baik hasil dari metode pengujian *whitebox.*

a. Pengujian *Whitebox* NPC

Tabel 1 menunjukkan skenario pengujian *whitebox* yang telah dilakukan terhadap karakter NPC.

Tabel 1. Pengujian *Whitebox* NPC

No.	Pengujian	Keterangan
1.	`private void Patroling(){ Debug.Log($"Patroling"); gameObject.GetComponent<Unit>().enabled = false; agent.enabled = true;`	NPC melakukan patrol kea rah yang telah ditentukan saat tidak ada musuh masuk ke jangkauan pandangnya (jangkauan chase). Pada saat statement ini berjalan, script ini akan mengaktifkan component Navmesh agent dan menonaktifkan komponen Unit (Script) dan pada console akan muncul tuisan "Patroling"
2.	`private void ChasePlayer(){ //State Debug.Log($"DLHunter is Chasing"); agent.enabled = false; gameObject.GetComponent<Unit>().enabled = true;`	NPC mematikan Component Navmesh agent yang ada padanya dan menghidupkan component Unit (Script) saat player memasuki area chasenya. Pada saat statement ini berjalan, script ini akan menonaktifkan component Navmesh agent dan mengaktifkan komponen Unit (Script) dan pada console akan muncul tuisan "DLHunter is Chasing"
3.	`private void AttackPlayer(){ //State Debug.Log($"DLHunter is Attacking"); gameObject.GetComponent<Unit>().enabled = false; agent.enabled = true; agent.SetDestination(transform.position); transform.LookAt(player);`	NPC akan melakukan penyerangan kepada player saat player memasuki area jangkauan serangnya. Pada saat statement ini berjalan, script ini akan mengaktifkan component Navmesh agent dan menonaktifkan komponen Unit (Script) dan pada console akan muncul tuisan "DLHunter is Attacking"

b. Pengujian *Whitebox* Algoritma A*

Tabel 2 menunjukkan skenario pengujian *whitebox* yang telah dilakukan terhadap penerapan algoritma A*.

Tabel 2. Pengujian Whitebox Algoritma A*

No.	Pengujian	Test Case benar
1.	`using UnityEngine; using System.Collections; public class Unit : MonoBehaviour { public Transform target; public float speed; Vector3[] path; int targetIndex; void Update() { PathRequestManager.RequestPath(transform.position,target.position, OnPathFound);} public void OnPathFound(Vector3[] newPath, bool pathSuccessful) { if (pathSuccessful) { path = newPath; targetIndex = 0; StopCoroutine("FollowPath"); StartCoroutine("FollowPath");}} IEnumerator FollowPath() { Vector3 currentWaypoint = path[0]; while (true){if (transform.position == currentWaypoint) { targetIndex ++; if (targetIndex >= path.Length) {yield break;} currentWaypoint = path[targetIndex];} transform.position = Vector3.MoveTowards(transform.position,currentWaypoint,speed * Time.deltaTime); yield return null;}} public void OnDrawGizmos() {if (path != null) {for (int i = targetIndex; i < path.Length; i ++) { Gizmos.color = Color.black; Gizmos.DrawCube(path[i], Vector3.one); if (i == targetIndex) { Gizmos.DrawLine(transform.position, path[i]);} else{Gizmos.DrawLine(path[i-1],path[i]);}}}}}`	Objek dapat sampai ke titik tujuan dan Objek dapat beradaptasi dengan perubahan titik tujuan dan mendapatkan jalur baru dengan menggunakna algoritma A-Star.

3.2.2 Pengujian Blackbox

Pada bagian ini akan disajikan hasil pengujian yang telah dilakukan, baik hasil dari metode pengujian *whitebox.*

a. Pengujian *Blackbox* NPC

Tabel 3 menunjukkan skenario pengujian *blackbox* yang telah dilakukan terhadap karakter NPC

Tabel 3. Pengujian Blackbox NPC

No.	Deskripsi pengujian	Hasil yang didapatkan	Kesimpulan
1.	NPC untuk *game*	NPC dapat bekerja sesuai dengan 3 *statement* yang diberikan	Berhasil
2.	NPC untuk pengujian Algoritma A*	NPC bergerak dengan menggunakan algoritma A-Star	Berhasil
3.	NPC untuk pengujian Navmesh	NPC bergerak menggunakan Navmesh agent dan bergerak melalui *NavMesh surface* yang telah ditentukan	Berhasil

b. Pengujian Blackbox Algoritma NPC

Tabel 4 menunjukkan skenario pengujian *blackbox* yang telah dilakukan terhadap penerapan algoritma NPC.

Tabel 4. Pengujian *Blackbox* Algoritma NPC

No.	Deskripsi pengujian	Hasil yang didapatkan	Kesimpulan
1.	Patroli	NPC bergerak ke titik *patrol* saat tidak ada musuh terdeteksi	Berhasil
2.	Mengejar	NPC mengejar taget saat target yang telah memasuki jangkauan pandang NPC	Berhasil
3.	Menyerang	NPC dapat melakukan aksi penyerangan saat player memasuki jangkauan serang	Berhasil

c. Pengujian Blackbox Algoritma A*

Tabel 5 menunjukkan skenario pengujian *blackbox* yang telah dilakukan terhadap penerapan algoritma A*.

Tabel 5. Pengujian *Blackbox* Algoritma A*

No.	Deskripsi pengujian	Hasil yang didapatkan	Kesimpulan
1.	Pencarian jalur menuju titik tujuan	NPC dapat mencari jalur terpendek padasuatu arena untuk sampai ke tujuan	Berhasil
2.	Beradaptasi kepada perubahan lokasi	NPC Menemukan jalur baru untuk menuju tujuan apabila target atau tujuan berpindah lokasi.	Berhasil

3.2.3 Pengujian Perbandingan

Tabel 6. Pengujian *Blackbox* Algoritma NPC

No.	Kecepatan Gerak	Bidang (Lokasi)	PathFinding	
			Algoritma A*	NavMesh
1.	20	Map 1	0.5 detik	2.5 detik
2.	20	Map 2	1.4 detik	4.15 detik
3.	20	Map 3	2.5 detik	5.4 detik
4.	20	Map 4	4 detik	10.7 detik
5.	20	Map 5	6.11 detik	12.46 detik
6.	40	Map 1	0.25 detik	1.25 detik
7.	40	Map 2	0.98 detik	3.3 detik
8.	40	Map 3	1.7 detik	5.6 detik
9.	40	Map 4	2.85 detik	13.5 detik
10.	40	Map 5	4.19 detik	23.14 detik

Seperti pada table 6, pengujian dilakukan dengan cara membandingkan peforma antara Algoritma A* dengan Algoritma NavMesh yang merupakan bawaan dari unity, dilakukan dengan cara menguji peforma akurasi dan kecepatan masing masing algoritma untuk mencapai tujuan. Pengujian ini akan dilakukan dengan menggunakan arena yang bervariasi. Dapat diambil kesimpulan dari tabel, yang merupakan hasil dari perbandingan algoritma A* dengan navmesh bahwa algoritma A* memiliki kualitas yang lebih baik dalam pencarian jalurnya dibandingkan menggunakan Navmesh.

3.3 *Distribution*

Tahap terakhir (Tahap pendistribusian). tahap yang bertujuan untuk menyebarkan dan menyampaikan produk yang telah dibuat dengan menggunakan langkah sebelumnya kepada pengguna. *Game* ini akan didistribusikan melalui halaman website https://itch.io/ .

4. Kesimpulan dan Saran

Algoritma A* (A-Star) memiliki kecepatan dan akurasi yang lebih tinggi bila dibandingkan dengan menggunakan Navmesh. Algoritma ini memiliki potensi yang baik dalam menjalankan peranya untuk mendukung jalanya sebuah NPC.

Kombinasi antara Algoritma A*(A-Star) sebagai metode pencarian jalur dengan Navmesh sebagai metode untuk menjaga pemain atau objek lainya tetap berada dalam arena, merupakan kombinasi yang dinilai cukup baik untuk mendukung jalanya sebuah NPC untuk dalam menjalankan peranya saat game berlangsung.

Berdasarkan hasil dari penelitian yang telah dilakukan dan diuji coba, peneliti memiliki beberapa saran yang mungkin bermanfaat bagi pembaca, beberapa saran tesebut yaitu sebagai berikut:

1. Penelitian ini menggunakan area dengan tipe datar sehingga kalkulasi atau perhitungan yang dilakukan hanya di pengaruhi dari luas arena, banyaknya halangan, dan sifat entitas yang ada dalam arena tersebut. Saran dari peneliti adalah untuk mengembangkan penelitian ini menuju tingkat yang lebih tinggi seperti menambahkan ketinggian atau kedalaman pada arena atau bahkan penambahan sifat pada rintangan.
2. Penelitian ini dilakukan dengan menggunakan prespektif layaknya kita sedang berjalan menyusuri suatu lokasi dengan menggunakan rute darat atau dua dimensi (terbatas pada bidang X dan Z pada unity). Saran dari peneliti adalah untuk mengembangkan penelitian ini dengan menambahkan aspek ketinggian dalam pencarian

jalur, misalkan pencarian jalur untuk pesawat, burung, misil kendali, atau lain sebagainya.

5. Daftar Pustaka:

Agung, E. G., Eridani, D., & Fauzi, A. (2022). Implementasi Metode Pathfinding dengan Algoritma A * pada Game Rogue-like menggunakan Unity. *Journal on Computing*, *7*(December), 81–94.

Agustini, & Kurniawan, W. J. (2019). Sistem E-Learning Do'a dan Iqro' dalam Peningkatan Proses Pembelajaran pada TK Amal Ikhlas. *Jurnal Mahasiswa Aplikasi Teknologi Komputer Dan Informasi*, *1*(3), 154–159.

Angkat, M. Y., Osmond, A. B., & Ansori, A. S. R. (2020). Membandingkan Algoritma Djikstra Dengan Algoritma a * Star Menggunakan Metode Logika Fuzzy Di Unity 3D Comparing the Djikstra Algorithm With the a * Star Algorithm Using the Fuzzy Logic Method in Unity 3D. *E-Proceeding of Engineering*, *7*(1), 1484–1490.

Borman, R. I., & Purwanto, Y. (2019). Implementasi Multimedia Development Life Cycle pada Pengembangan Game Edukasi. *Jurnal Edukasi Dan Penelitian Informatika*, *5*(2), 119–124.

Idayat, R., & Handayani, I. (2022). Penerapan Algoritma A*Star Menggunakan Graph Untuk Menentukan Rute Terpendek Berbasis Web. *Pendidikan Dan Informatika*, *1*(1), 7–14.

Iskandar, R. J., Antonius, & Edwinyo. (2019). Penggunaan Unity Engine pada Perancangan Game The Cient dengan Navigation Mesh. *InTekSis*, *8*(2), 61–72.

Junanto, E., Andrew Brian Osmond, S.T., M. T., & Anton Siswo Raharjo Ansori, S.T., M. T. (2020). Membuat Pergerakan Non-Player Character (Npc) Menggunakan Metode a Star Making Non-Player Character (Npc) Movement Using the a Star Method. *EProceeding of Engineering*, *7*(1), 1491–1497.

Liman, H. L., Pragantha, J., & Haris, D. A. (2022). Pembuatan Game Hack-and-Slash dengan Deck Building 2D "Need More Gold." *Jurnal Ilmu Komputer Dan Sistem Informasi*, *10*(1), 1–7.

Mutaqin, G., Fadilah, J. N., & Nugroho, F. (2021). Implementasi Metode Path Finding dengan Penerapan Algoritma A-Star untuk Mencari Jalur Terpendek pada Game "Jumrah Launch Story." *Walisongo Journal of Information Technology*, *3*(1), 43–48.

Pasaribu1, J., Osmond2, A. B., & Saputra3, R. E. (2019). Pengembangan Perilaku Karakter Lalat Pada Game Menjaga Makanan Menggunakan Algoritma a* (a Star) Develompment of Flies Character Behavior on Game Keeping Food Using Algorithm a* (a Star). *EProceedings of Engineering*, *6*(2), 5715–5722.

Putra, M. M. I., Sompie, S. R. U. A., Paturusi, S., Elektro, T., & Sam, U. (2020). Implementasi Speech Recognition pada Aplikasi Pembelajaran Bahasa Inggris untuk Anak. *Jurnal Teknik Informatika*, *15*(4), 247–256.

Ramadhan, A. W. R., & Udjulawa, D. (2020). Perbandingan Algoritma Dijkstra dan Algoritma A Star pada permainan Pac-Man. *Jurnal Algoritme*, *1*(1), 12–20.

Safitra, W., Faisol, A., & Adi Wibowo, S. (2020). Application of the Finite State Machine Method to Non Player Character (NPC) Action Strategy Game "Ouroboros." *JATI (Jurnal Mahasiswa Teknik Informatika)*, *4*(2), 292–297.

Saleh, A., & P, D. W. (2021). Implementasi Kecerdasan Buatan Menggunakan Algoritma A-Star Dan Repulsive Field Pada Simulasi Game 3D. *Prosiding Seminar Nasional Riset Dan Teknologi Terapan (RITEKTRA)*, C1--C1.

Wibowo, M. C. (2022). Pemodelan Dengan Blender 3D. *Penerbit Yayasan Prima Agus Teknik*, *8*(1 SE-Judul Buku).

Yohanes, D. N., & Rochmawati, N. (2022). Implementasi Algoritma Collision Detection dan A*(A Star) pada Non Player Character Game World Of New Normal. *Journal of Informatics and Computer Science (JINACS)*, *3*(03), 322–333.