Algoritma A* dalam Pencarian Rute Terpendek pada *Game The Legend of Zelda: Breath Of The Wild*

Muhammad Furqon 13519184
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): 13519184@std.stei.itb.ac.id

***Abstrak*—*Game* The Legend of Zelda: Breath of the Wild sebagai *game* bergenre *open world*, dimana lingkunan pada *game* dapat dieksplorasi dan berinteraksi dengan pemain secara bebas. Untuk menempuh suatu titik pada peta ke titik lainnya diperlukan sebuah rute. Pencarian rute atau *path planning* yang terpendek dapat dicari dengan menerapkan algoritma A* dengan representasi peta pada *game* yang direpresentasikan sebagai graf. Algoritma A* akan menemukan solusi rute terpendek yang optimal sesuai dengan graf yang dibangun dengan asumsi yang sudah dibuat.**

Kata Kunci*—*A**; Breath of The Wild; graf; *path planning

I. PENDAHULUAN

Game dengan genre *open world* merupakan genre dimana dapat melakukan eksplorasi pada dunia virtual yang luas. Untuk eksplorasi tersebut diperlukan rute yang untuk berpindah dari suatu titik di peta ke titik lainnya. Pada pencarian rute terdekat dapat menggunakan algoritma A* untuk menemukan rute yang optimal. Pada makalah ini, *game* yang digunakan untuk menjadi media implementasi tersebut adalah The Legend of Zelda: Breath of the Wild (Gambar 1).

The Legend of Zelda: Breath of the Wild dirilis bersamaan pada *console* Nintendo Switch dan Wii U pada tanggal 3 Maret 2017. *Game* ini termasuk ke dalam *game* "open-world action-adventure" yang dikembangkan dan di-*publish* sebagai bagian dari *franchise* Zelda, yang telah ada sejak tahun 1986[1].

Gambar 1 Poster The Legend of Zelda: Breath of the Wild(Sumber: https://www.nintendo.co.uk/Games/Nintendo-Switch/The-Legend-of-Zelda-Breath-of-the-Wild-1173609.html)

Player dapat menjelajah dunia bernama Hyrule sebagai pahlawan bernama Link. Link terbangun setelah 100 tahun kehancuran Hyrule oleh Calamity Ganon. Setelah 100 tahun, Ganon yang sedang disegel oleh putri Zelda di Hyrule Castle, mengancam kembali keselamatan Hyrule. Link perlu mengembalikan kembali kontrol terhadap empat Divine Beasts, robot raksaksa yang masih dalam kendali Ganon. Link perlu mengumpulkan kembali kekuatannya, membebaskan para Divine Beast, dan membantu Zelda pada Hyrule Castle untuk mengalahkan Calamity Ganon untuk selamanya [1]. Saat menjelajah dunia *game*, *player* dapat dibantu dengan fitur pencarian rute terpendek untuk memenuhi permainan.

Makalah ini bertujuan untuk menjelaskan penerapan algoritma A* dalam mencari rute terpendek pada *game* The Legend of Zelda: Breath of the Wild. Sistematika penulisan masalah terdiri dari tujuh bagian. Bagian pertama berisi pendahuluan, bagian kedua landasan teori, bagian ketiga berisi analisis masalah dan implementasi, bagian keempat berisi pengujian algoritma A*, bagian kelima berisi kesimpulan dan saran, bagian keenam berisi tautan program dan video, dan bagian ketujuh berisi ucapan terima kasih.

II. LANDASAN TEORI

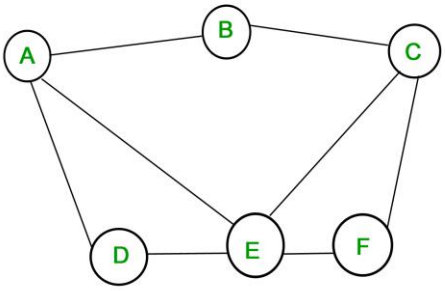
Untuk mendapatkan pencarian rute atau *path planning* dibutuhkan landasan teori, yaitu algoritma A* dan *game* The Legend of Zelda: Breath of the Wild.

A. *Algoritma A**

Pencarian rute atau *path planning* dapat menggunakan algoritma A*, yaitu algoritma yang termasuk ke dalam algoritma *informed search* yang mampu mendapatkan rute dengan jarak terpendek dari dua titik dengan informasi *cost* atau bobot sehingga menghindari titik yang akan membangkitkan *cost* yang lebih besar[2].

Graf adalah kumpulan titik dan garis yang menghubungkan sebagian titik (ada kemungkinan kosong). Titik-titik pada sebuah graf disebut sebagai vertices graph, istilah lainnya titik, simpul, atau *node*. Garis yang menghubungkan kedua *vertice* tadi adalah *edges*, istilah lainnya garis atau sisi[3].

Graf yang digunakan adalah graf tidak berarah (*undirected graph*) sehingga peta dapat direpresentasikan sebagai simpul terhubung yang tidak memiliki arah. Graf tidak berarah dapat dilihat sesuai dengan Gambar 2.



Gambar 2 Graf tidak berarah(Sumber: https://www.geeksforgeeks.org/mathematics-graph-theory-basics/)

Penentuan bobot didasari perhitungan *cost*, perhitungan tersebut dinyatakan dalam formula(1)

$$f(n) = g(n) + h(n) \qquad (1)$$

$f(n)$ = estimasi terkecil total *cost* solusi dari *path* yang melalui n.

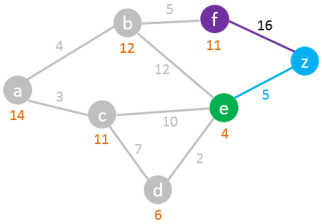
$g(n)$ = *cost* dari *path* sampai dengan simpul n.

$h(n)$ = estimasi *cost* terkecil *path* dari n ke tujuan atau *goal.*

Euclidean distance dipilih sebagai fungsi heuristik karena nilai $h(n)$ hasil heuristik *admissible* dan tidak akan melebihi atau *overestimate* serta konsisten. Nilai $h(n) \leq h^*(n)$ dimana $h^*(n)$ merupakan *cost* sesungguhnya. *Euclidean distance* antara dua titik didefinisikan sebagai berikut pada formula 2,

$$h(n) = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2} \qquad (2)$$

Dengan menggunakan nilai f dari formula 1, algoritma A* dapat mencari rute terdekat seperti contoh pada Gambar 3.



Node	Status	Shortest Distance From A	Heuristic Distance to Z	Total Distance*	Previous Node
A	Visited	0	14	14	
B	Visited	4	12	16	A
C	Visited	3	11	14	A
D	Visited	10	6	16	C
E	Visited	12	4	16	D
F		9	11	20	B
Z	Current	17	0	17	E

* Total Distance = Shortest Distance from A + Heuristic Distance to Z

Gambar 3 Contoh penggunaan A pada graf tidak berarah untuk pencarian rute terdekat (Sumber: https://www.101computing.net/a-star-search-algorithm/ diakses pada tanggal 8 Mei 2021)*

B. *The Legend of Zelda: Breath Of The Wild*

Game ini termasuk ke dalam genre *open world*, dunia dalam *game* tidak dibatasi lewat *loading screen* antara lokasi. Pemain dapat melihat lokasi yang jauh dan dapat bergerak secara bebas. Pemain dalam *game* ini dapat berenang, memanjat dinding, dan *paragliding* untuk mencapai lokasi pada suatu peta[4].

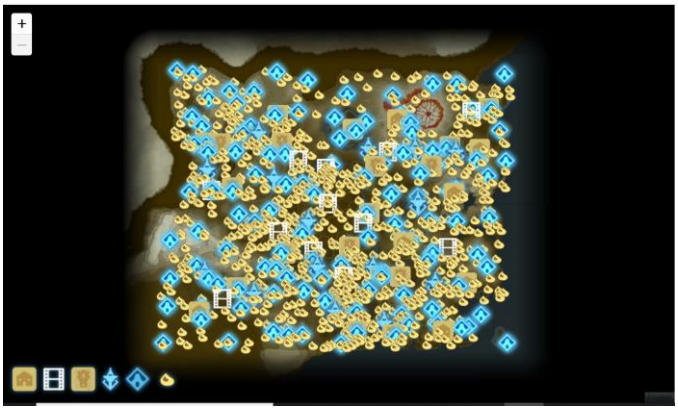
Peta dibagi ke dalam delapan daerah dengan kondisi yang berbeda sesuai dengan Gambar 4. Dalam permainan peta juga dibagi ke dalam 15 bagian yang masing-masing bagiannya memiliki sebuah menara atau *tower*. Luas peta tercatat sebagai peta pada peringkat ke-22 peta *game* terbesar dengan interaksi pemain dengan lingkungan *game* yang sangat bebas[4].

Gambar 4 Peta The Legend of Zelda: Breath of the Wild(Sumber: https://objmap.zeldamods.org/#/map/z2,0,-136)

Pada peta terdapat titik-titik yang relevan dalam *game* sehingga dapat dijadikan acuan oleh pemain. Terdapat enam tipe titik yang masing-masing memiliki simbol *icon* pada peta. Tipe titik pada Gambar 5 dari kiri ke kanan adalah "town", "memory", "stable", "tower", "shrine", dan "korok".

Gambar 5 Simbol semua tipe lokasi

Lokasi atau titik yang bertipe "town" sebanyak sembilan titik, tipe "memory" sebanyak 12, tipe "stable" sebanyak 15, "tower" sebanyak 15, "shrine" sebanyak 120, dan "korok" sebanyak 900 titik. Total semua titik sebanyak 1062 titik yang terdapat peta yang dapat digunakan sebagai acuan. Representasi visual sesuai pada Gambar 6.



Gambar 6 Tampilan Website Program dengan semua tipe lokasi

III. Analisis Masalah dan Implementasi

Asumsi yang digunakan pada peta yang digunakan untuk algoritma A*, yaitu

1. Peta dianggap datar atau tidak memiliki ketinggian yang akan mempengaruhi jarak.
2. Jarak antara kedua titik tidak dipengaruhi oleh objek lain yang terdapat dalam peta.

Asumsi yang digunakan pada graf untuk merepresentasikan titik untuk algoritma A*, yaitu

1. Simpul atau titik dapat terhubung satu sama lain tanpa membandingkan bentuk peta.
2. Simpul atau titik pada peta yang digunakan untuk membangun rute adalah titik yang tipe lokasi dalam keadaan aktif.
3. Ketetanggan antar simpul ditentukan berdasarkan jarak *euclidean* antar titik pada peta.
4. Graf direpresentasikan sebagai graf tidak berarah dengan bobot sisi antara dua simpul sama secara bolak-balik.

A. *Preproses (kalkulasi jarak euclidean dan penentuan tetangga)*

Gambar 7 Tampilan Website Program dengan tipe lokasi aktif "memory"

Pada contoh kasus penerapan algoritma A* tipe lokasi yang digunakan adalah tipe "memory". Titik atau simpul pada peta dengan tipe lokasi aktif "memory" diberi penomoran dan singkatan sebagai berikut,

1. Lake Kolomo (LK)
2. Kara Kara Bazaar (KB)
3. Spring of Power (SoP)
4. Sanidin Park Ruins (SP)
5. Lanaryu Road – East Gate (LR)
6. Ancient Columns (AC)
7. Irch Plain (IP)
8. Eldin Canyon (EC)
9. West Necluda (WN)
10. Hyrule Field (HF)
11. Sacred Ground Ruins (SG)
12. Hyrule Castle (HC)

Perhitungan jarak antara simpul dengan menggunakan *euclidean distance* sesuai dengan formula 2 antara kedua simpul tersebut. Jarak pada Tabel 1 dibulatkan menjadi bilangan bulat.

	1	2	3	4	5	6	7	8	9	10	11	12
1	-	49	84	18	69	54	20	43	18	26	14	19
2	49	-	134	36	116	31	52	92	63	76	61	63
3	84	134	-	100	27	133	89	43	73	58	74	75
4	18	36	100	-	87	35	17	58	36	43	26	27
5	69	116	27	87	-	122	79	36	54	44	62	65
6	54	31	133	35	122	-	45	91	71	78	60	59
7	20	52	89	17	79	45	-	46	36	35	17	14
8	43	92	43	58	36	91	46	-	36	18	32	32
9	18	63	73	36	54	71	36	36	-	18	21	28
10	26	76	58	43	44	78	35	18	18	-	18	22
11	14	61	74	26	62	60	17	32	21	18	-	6
12	19	63	75	27	65	59	14	32	28	22	6	-

Tabel 1 jarak antar simpul

Setelah perhitungan simpul, setiap simpul ditentukan ketetangaan berdasarkan jaraknya. Suatu simpul bertetangga dengan empat simpul lainnya dengan jarak euclidean terdekat. Hasil dari ketetanggan antar simpul ditunjukkan pada Gambar 8

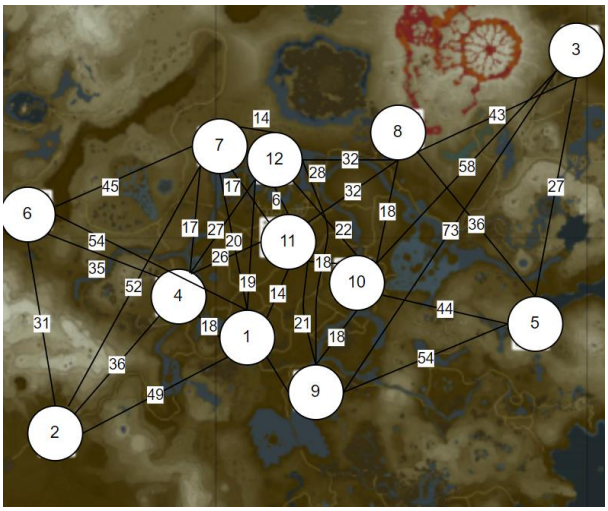
Gambar 8 Hasil console program dalam penentuan ketetanggaan antar simpul

Ketetanggaan dari sebuah simpul direpresentasikan sebagai matriks dengan nilai yang sesuai dengan jarak antar simpul. Nilai "-" menyatakan kedua simpul tidak bertetangga. Matriks ketetanggan tersebut direpresentasikan dalam Tabel 2.

	1	2	3	4	5	6	7	8	9	10	11	12
1	-	49	-	18	-	54	20	-	18	-	14	19
2	49	-	-	36	-	31	52	-	-	-	-	-
3	-	-	-	-	27	-	-	43	73	58	-	-
4	18	36	-	-	-	35	17	-	-	-	26	27
5	-	-	27	-	-	-	-	36	54	44	-	-
6	54	31	-	35	-	-	45	-	-	-	-	-
7	20	52	-	17	-	45	-	-	-	-	17	14
8	-	-	43	-	36	-	-	-	-	18	32	32
9	18	-	73	-	54	-	-	-	-	18	21	28
10	-	-	58	-	44	-	-	18	18	-	18	22
11	14	-	-	26	-	-	17	32	21	18	-	6
12	19	-	-	27	-	-	14	32	28	22	6	-

Tabel 2 Matriks Ketetanggaan

Graf dibuat dengan jarak dan ketetanggaan yang sesuai dengan matriks ketetanggaan yang telah dibuat. Graf yang telah dibuat ditunjukkan pada gambar graf pada gambar 9.



Gambar 9 Graf

*B. Langkah-Langkah Algoritma A**

Algoritma A* menggunakan *open list* untuk menyatakan simpul hidup yang diekspan dengan *priority queue* dengan nilai elemen berindeks lebih kecil memiliki nilai bobot f yang lebih kecil. Simpul yang telah diekspan menggunakan *closed list* sehingga tidak akan diekspan lagi[6].

Langkah-langkah yang dilakukan pada pencarian A* adalah sebagai berikut:

1. Tambahkan simpul awal ke *open list* (simpul hidup).
2. Elemen pertama dari simpul hidup yang memiliki nilai f terkecil dipilih sebagai simpul ekspan.
3. Simpul ekspan dipindahkan dari simpul hidup menjadi simpul yang sudah diekspan (*closed list*).
4. Untuk setiap simpul tetangga pada simpul jika sudah ada dalam *closed list* abaikan dan lanjut ke simpul tetangga selanjutnya
5. Untuk setiap simpul tetangga Jika belum ada pada simpul hidup, tambahkan pada simpul hidup. Jika sudah, pilih simpul dengan nilai g yang lebih kecil karena memiliki *cost* yang lebih baik.
6. Ulangi langkah 2 sampai satu di antara dua kondisi terpenuhi. Kondisi pertama simpul tujuan atau *goal* telah diekspan atau masuk ke *closed* list sehingga *path* telah ditemukan. Kondisi kedua jika tidak ada
7. Path yang didapat merupakan path dari simpul tujuan ke simpul sebelumnya yang mengekspan *goal* (*parent node*), proses ini diulang sampai mencapai simpul awal.

C. Penerapan A pada Program Berbasis Web*

Program mengembangkan dari peta Zelda[5]. Program menambahkan pencarian rute terdekat dengan menerapkan algoritma A*. Program berbasiskan pemrograman berorientasi objek. Program yang digunakan berbasis react app dengan menggunakan react leaflet sebagai representasi dan interaksi peta *game* dan react redux untuk menyimpan informasi berupa kondisi atau *state* dari peta. Program dibuat dengan menggunakan bahasa JavaScript.

Representasi dari sebuah simpul merupakan kelas Simpul Jarak yang menyimpan beberapa informasi, yaitu nama simpul, koordinat simpul, tetangga simpul, nilai beban simpul berupa nilai f, g, dan, h, serta orang tua dari simpul tersebut bila simpul diekspan dari simpul lain.

Penerapan A* pada program dilakukan pada fungi astar(start,end) yang akan mengembalikan rute berupa array dari simpul dengan simpul awal start dan simpul akhir end. Simpul hidup yang akan diekspan direpresentasikan dengan open_list dan simpul yang sudah diekspan direpresentasikan dengan closed_list.

Program dideploy pada Heroku dan dapat diakses lewat *internet browser* baik dengan menggunakan komputer ataupun *smartphone.* Pengguna dapat memilih tipe lokasi yang aktif,

lalu menekan *icon* lokasi sebagai titik mulai dan selesai. (tautan: https://zelda-makalah-stima.herokuapp.com/).

IV. PENGUJIAN ALGORITMA A*

Algoritma A* memerlukan dua simpul sebagai simpul awal dan simpul akhir. Pencarian rute akan dimulai dari simpul awal, lalu ke simpul yang akan membangkitkan bobot terkecil. Hasil akhir adalah rute dengan jarak terkecil selama simpul awal dan akhir tidak saling terisolasi atau dapat terhubung satu sama lain.

Kasus 1:

Tipe lokasi yang aktif: "memory"

Rute dimulai dari titik pada simpul 2 dengan nama "Kara-Kara Bazaar" (KB) sebagai simpul awal ke simpul 8 dengan nama "Eldin Canyon" (EC). Penamaan dari singkatan sesuai pada bagian 3.A bagian preproses. Iterasi dari A* dicatat dalam Tabel 3.

Iterasi	Simpul Ekspan	Simpul Hidup	f(n)
1	KB	LK[KB]	92,43
		SP[KB]	93,86
		IP[KB]	98,40
		AC[KB]	121,69
2	LK	SG[LK,KB]	92,64
		SP[KB]	93,86
		HC[LK,KB]	94,85
		IP[KB]	98,40
		WN[LK,KB]	98,91
		AC[KB]	121,69
3	SG[LK,KB]	EC[SG,LK,KB]	92,20
		HF[SG,LK,KB]	93,73
		SP[KB]	93,86
		HC[LK,KB]	94,85
		IP[KB]	98,40
		WN[LK,KB]	98,91
		AC[KB]	121,69
4	EC[SG,LK,KB]	Solusi Ditemukan	92,20

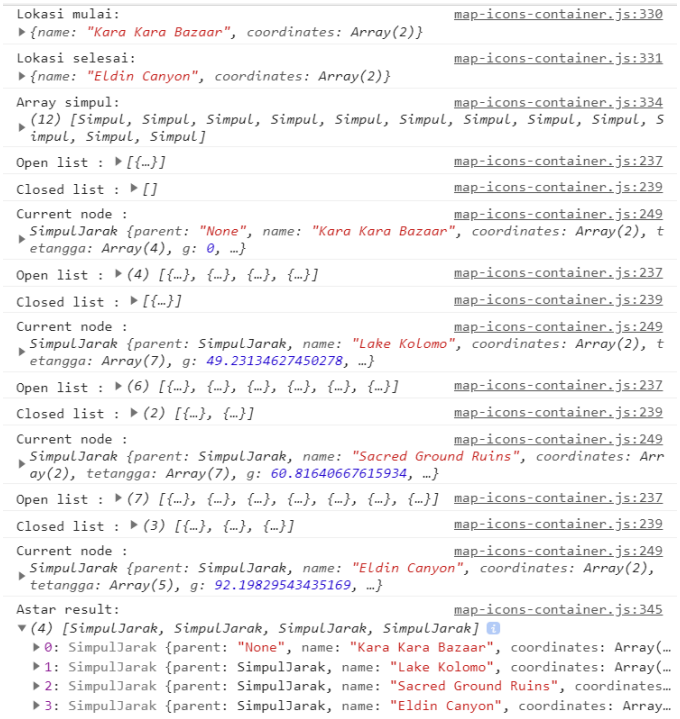
Tabel 3 Iterasi A Kasus 1 sesuai program*

Hasil dari algoritma A* sesuai dengan tabel 3 yang mengacu pada program sesuai dengan Gambar 10 dan direpresentasikan secara visual oleh program sesuai Gambar 11.

Rute terdiri dari simpul-simpul sebagai berikut: KB – LK – SG – EC

Rute Optimal:

"Kara Kara Bazaar" – "Lake Kolomo" – "Sacred Ground Ruins" – "Eldin Canyon" dengan jarak 92,20



Gambar 10 Hasil Console Program rute dari "Kara Kara Bazaar" ke "Eldin Canyon"

Gambar 11 Hasil Rute pada Program dari "Kara Kara Bazaar" ke "Eldin Canyon"

Kasus 2:

Tipe lokasi yang aktif: "memory","stable","tower"

Dilakukan preproses dengan total 42 simpul (Gambar 12). Setelah itu, dicari rute terpendek dimulai dari titik pada simpul dengan nama "Rito Stable" sebagai simpul awal ke simpul dengan nama "Faron Tower".

Penamaan singkatan untuk simpul yang relevan pada kasus pencarian rute dengan A*

1. Rito Stable (RS) 14. Central Tower

2. Faron Tower (FT)
3. Tabantha Bridge Stable (TB)
4. Hebra Tower (HT)
5. Ancient Columns (AC)
6. Tabantha Tower (TT)
7. Gerudo Canyon Stable (GC)
8. Ridgeland Tower (RT)
9. Serenne Stable(SS)
10. Snowfield Stable(SnS)
11. Sanidin Park Ruins(SP)
12. Irch Plain(IP)
13. Outskirt Stable (OS)
14. Central Tower (CT)
15. Wasteland Tower (WT)
16. Great Plateau Tower (GP)
17. Lake Kolomo (LK)
18. West Necluda (WN)
19. Lake Tower (LT)
20. Sacred Ground Ruins (SG)
21. Hyrule Castle (HC)
22. Highland Stable (HS)
23. Riverside Stable (RvS)
24. Dueling Peaks Tower (DT)
25. Lakeside Stable (LS)

Gambar 12 Tampilan Program dengan tipe lokasi aktif "memory","stable", dan "tower"

Iterasi	Simpul Ekspan	Simpul Hidup	f(n)
1	RS	TB[RS]	97,26
		HT[RS]	100,80
		AC[RS]	106,21
		TT[RS]	108,52
2	TB[RS]	HT[RS]	100,80
		AC[RS]	106,21
		TT[RS]	108,52
		GC[TB,RS]	111,22
3	HT[RS]	RT[HT,RS]	99,04
		SS[HT,RS]	102,80
		SnS[HT,RS]	104,46
		AC[RS]	106,21
		TT[RS]	108,52
		GC[TB,RS]	111,22
4	RT[HT,RS]	SP[RT,HT,RS]	96,84
		SS[HT,RS]	102,80
		IP[RT,HT,RS]	103,73
		SnS[HT,RS]	104,46
		AC[RS]	106,21
		TT[RS]	108,52
		GC[TB,RS]	111,22
5	SP[RT,HT,RS]	OS[SP,RT,HT,RS]	96,94
		CT[SP,RT,HT,RS]	99,16
		SS[HT,RS]	102,80
		IP[RT,HT,RS]	103,73
		SnS[HT,RS]	104,46
		AC[RS]	106,21
		WT[SP,RT,HT,RS]	107,70
		TT[RS]	108,52
		GC[TB,RS]	111,22
6	OS[SP,RT,HT,RS]	GP [OS,SP,RT,HT,RS]	96,90
		LK [OS,SP,RT,HT,RS]	97,44
		CT[SP,RT,HT,RS]	99,16
		SS[HT,RS]	102,80
		IP[RT,HT,RS]	103,73
		SnS[HT,RS]	104,46
		AC[RS]	106,21
		WT[SP,RT,HT,RS]	107,70
		TT[RS]	108,52
		GC[TB,RS]	111,22
7	GP [OS,SP,RT,HT,RS]	LK [OS,SP,RT,HT,RS]	97,44
		WN [GP,OS,SP,RT,HT,RS]	97,75
		LT [GP,OS,SP,RT,HT,RS]	98,68
		CT[SP,RT,HT,RS]	99,16
		SS[HT,RS]	102,80
		IP[RT,HT,RS]	103,73
		SnS[HT,RS]	104,46
		AC[RS]	106,21
		WT[SP,RT,HT,RS]	107,70
		TT[RS]	108,52
		GC[TB,RS]	111,22
8	LK [OS,SP,RT,HT,RS]	WN [GP,OS,SP,RT,HT,RS]	97,75
		LT [GP,OS,SP,RT,HT,RS]	98,68
		CT[SP,RT,HT,RS]	99,16
		SS[HT,RS]	102,80
		IP[RT,HT,RS]	103,73
		SG [LK,OS,SP,RT,HT,RS]	104,26
		SnS[HT,RS]	104,46
		AC[RS]	106,21
		WT[SP,RT,HT,RS]	107,70
		HC [LK,OS,SP,RT,HT,RS]	108,16
		TT[RS]	108,52
		GC[TB,RS]	111,22
9	WN [GP,OS,SP,RT,HT, RS]	LT [GP,OS,SP,RT,HT,RS]	98,68
		CT[SP,RT,HT,RS]	99,16
		HS [WN ,GP,OS,SP,RT,HT,RS]	101,30
		RvS [WN ,GP,OS,SP,RT,HT,RS]	102,55
		SS[HT,RS]	102,80
		IP[RT,HT,RS]	103,73
		SG [LK,OS,SP,RT,HT,RS]	104,26
		SnS[HT,RS]	104,46
		DT[WN ,GP,OS,SP,RT,HT,RS]	105,13
		AC[RS]	106,21
		WT[SP,RT,HT,RS]	107,70
		HC [LK,OS,SP,RT,HT,RS]	108,16
		TT[RS]	108,52
		GC[TB,RS]	111,22
10	LT [GP,OS,SP,RT,HT, RS]	CT[SP,RT,HT,RS]	99,16
		HS [WN ,GP,OS,SP,RT,HT,RS]	101,30
		RvS [WN ,GP,OS,SP,RT,HT,RS]	102,55
		SS[HT,RS]	102,80
		IP[RT,HT,RS]	103,73
		SG [LK,OS,SP,RT,HT,RS]	104,26

		SnS[HT,RS]	104,46
		DT[WN ,GP,OS,SP,RT,HT,RS]	105,13
		AC[RS]	106,21
		WT[SP,RT,HT,RS]	107,70
		HC [LK,OS,SP,RT,HT,RS]	108,16
		TT[RS]	108,52
		GC[TB,RS]	111,22
11	CT[SP,RT,HT,RS]	HS [WN ,GP,OS,SP,RT,HT,RS]	101,30
		RvS [WN ,GP,OS,SP,RT,HT,RS]	102,55
		SS[HT,RS]	102,80
		IP[RT,HT,RS]	103,73
		SG [LK,OS,SP,RT,HT,RS]	104,26
		SnS[HT,RS]	104,46
		DT[WN ,GP,OS,SP,RT,HT,RS]	105,13
		AC[RS]	106,21
		WT[SP,RT,HT,RS]	107,70
		HC [LK,OS,SP,RT,HT,RS]	108,16
		TT[RS]	108,52
		GC[TB,RS]	111,22
12	HS[WN,GP,OS,SP, RT,HT,RS]	FT[HS,WN,GP,OS,SP,RT,H T,RS]	96,82
		RvS [WN ,GP,OS,SP,RT,HT,RS]	102,55
		SS[HT,RS]	102,80
		IP[RT,HT,RS]	103,73
		SG [LK,OS,SP,RT,HT,RS]	104,26
		SnS[HT,RS]	104,46
		DT[WN,GP,OS,SP,RT,HT,R S]	105,13
		AC[RS]	106,21
		WT[SP,RT,HT,RS]	107,70
		HC [LK,OS,SP,RT,HT,RS]	108,16
		TT[RS]	108,52
		LS[HS,WN,GP,OS,SP,RT,H T,RS]	108,97
		GC[TB,RS]	111,22
13	FT[HS,WN,GP,OS, SP,RT,HT,RS]	Solusi Ditemukan	96,82

Tabel 4 Iterasi A Kasus 2 sesuai program*

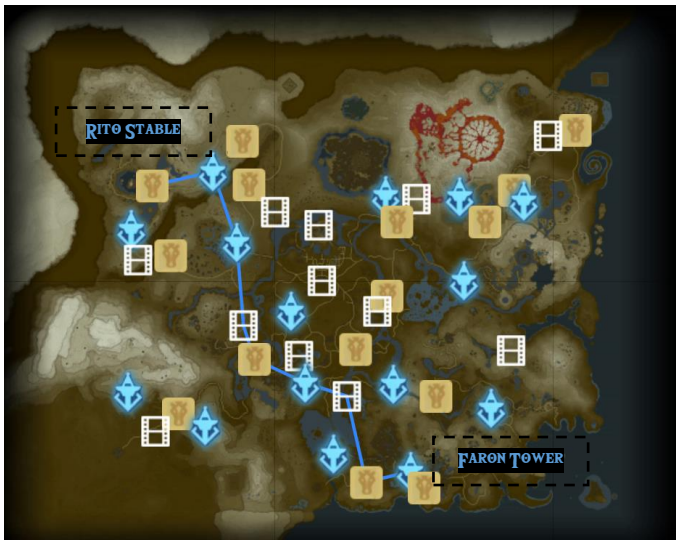
Hasil dari algoritma A* sesuai dengan tabel 4 yang mengacu pada program sesuai dengan Gambar 13 dan direpresentasikan secara visual oleh program sesuai Gambar 14.

Rute terdiri dari simpul-simpul sebagai berikut: RS-HT-RT-SP-OS-GP-WN-HS-FT

Rute optimal:

"Rito Stable" - "Hebra Tower" – "Ridgeland Tower" – "Sanidin Park Ruins" – "Outskirt Stable" – "Great Plateau Tower" – "West Necluda" – "Highland Stable" – "Faron Tower" dengan jarak 96,82.

Gambar 13 Hasil Console bagian hasil program rute dari "Rito Stable" ke "Faron Tower"



Gambar 14 Hasil rute pada Program dari "Rito Stable" ke "Faron Tower"

V. KESIMPULAN DAN SARAN

A. *Kesimpulan*

Hasil penerapan Algoritma A* untuk *game* The Legend of Zelda: Breath of the Wild dapat menentukan rute terpendek dari lokasi pada peta ke lokasi lainnya. Peta yang digunakan dapat direpresentasikan sebagai graf dengan asumsi yang telah ditetapkan sebelumnya.

Algoritma A* dapat digunakan untuk mencari rute terpendek dari suatu titik ke titik lainnya yang terhubung dalam sebuah graf.

B. *Saran*

Penyelesaian dari rute terpendek dapat dipengaruhi oleh faktor lain jika asumsi dihilangkan dan dilakukan secara nyata. Penggunaan algoritma A* dapat menjadi lebih baik jika penentuan ketetanggan yang lebih baik dan menggunakan fungsi heuristik yang memperhitungkan kondisi dunia *game* sehingga dapat menemukan rute terpendek yang sesuai dengan kondisi pada *game.*

VI. Tautan

Penulis membuat program yang dapat diakses melalui tautan berikut. Penulis juga membuat video yang dapat diakses melalui tautan Youtube:

Tautan Program/Website

https://zelda-makalah-stima.herokuapp.com/

Tautan Video Youtube

https://youtu.be/zkmqQpa5Tj4

VII. Ucapan Terima Kasih

Segala puji dan syukur penulis panjatkan ke hadirat Allah SWT karena berkat rahmat, hidayah, dan karunia-Nya, penulis dapat menyelesaikan makalah ini tepat waktu dengan baik. Terima kasih kepada kedua orang tua yang telah memberi dukungan kepada penulis.

Terima kasih yang sebesar-besarnya kepada semua pihak yang telah membantu penulis dalam menyelesaikan makalah ini. Terima kasih kepada Pak Rinaldi Munir selaku dosen pengampu mata kuliah Strategi Algoritma IF2211 kelas K4, dan juga kepada seluruh tim pengajar mata kuliah IF2211 yang telah mengajarkan ilmunya kepada penulis sehingga penulis mampu menyelesaikan makalah ini.

Penulis juga menyampaikan terima kasih kepada semua teman penulis yang selalu memberikan motivasi, masukan, dan bantuan dalam pengerjaan tugas makalah ini.

Daftar Referensi

[1] Vidqvist, Joel. 2019. Open-world Game Design Case Study the Legend of Zelda: Breath of the Wild. Tampere:Tampere University of Applied Sciences.

[2] Munir, Rinaldi, https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Route-Planning-Bagian2-2021.pdf, diakses tanggal 8 Mei 2021.

[3] Munir, Rinaldi, https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf, diakses tanggal 8 Mei 2021.

[4] Codex Gamicus, Open-world video games, https://gamicus.gamepedia.com/Open-world_video_games, diakses tanggal 8 Mei 2021.

[5] Duffy, Willman, https://github.com/willmanduffy/breath-of-the-wild-map, diakses tanggal 8 Mei 2021.

[6] Swift, Nicholas, https://medium.com/@nicholas.w.swift/easy-a-star-pathfinding-7e6689c7f7b2, diakses tanggal 8 Mei 2021.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 10 Mei 2021

Muhammad Furqon
13519184