Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN: 2548-964X
Vol. 2, No. 10, Oktober 2018, hlm. 3381-3390 http://j-ptiik.ub.ac.id

Penerapan Algoritme Basic Theta* Pada *Game* Hexaconquest

Ilman Naafian Firmansyah[1], Eriq Muh. Adams Jonemaro[2], Muhammad Aminul Akbar[3]

Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Brawijaya
Email: [1]illnavyforce@gmail.com, [2]eriq.adams@ub.ac.id, [3]muhammad.aminul@ub.ac.id

Abstrak

Pada zaman sekarang hampir semua *Game Turn-based Strategy* memberikan fitur singleplayer pada tipe permainan yang dapat dilakukan pemainnya. Jika pemain manusia hanya satu orang, maka pemain lainnya harus digerakkan oleh komputer. Pada kondisi seperti ini peran AI (Artificial Intelligence) atau kecerdasan buatan dibutuhkan. Kecerdasan buatan digunakan pada *Game* agar pemain manusia dapat merasa seakan-akan melawan manusia sehingga ia dapat melatih kemampuan bermainnya terlebih dahulu dengan melawan komputer sebelum melawan pemain manusia lain. Algoritme yang sering digunakan oleh AI pada *Game* untuk mencari jalan terbaik menuju lokasi tujuannya adalah algoritme A*. Namun, algoritme A* tidak selalu merupakan solusi terbaik dalam pathfinding. Maka dari itu, penerapan algoritme Basic Theta* dilakukan oleh penulis pada *Game* strategi berbasis giliran atau *Turn-based Strategy* yang bernama Hexaconquest. Algoritme pathfinding Basic Theta* akan dibandingkan dengan algoritme pathfinding dasar pada *Game* Hexaconquest yakni algoritme A*. Perbandingan performa kedua algoritme dilakukan dengan melihat jumlah frame per second, waktu eksekusi, dan jumlah cost node yang dilewati oleh agen algoritme. Dari hasil penelitian ini dapat disimpulkan bahwa algoritme Basic Theta* dapat memberikan solusi dengan jarak terpendek, sedangkan A* dapat memberikan solusi dengan lebih cepat dan ringan.

Kata kunci: *algoritme pencari, A*(A star), basic theta*(basic theta star), penerapan, perbandingan*

Abstract

Nearly every Turn-based Strategy Games today have a singleplayer feature in their Game modes. If there is only one human player, then every other player must be controlled by computer. This is where AI(Artificial Intelligence) used. AI used so the Game can be played by a human player as if they play against another human. Human player can then use this feature to train their playing skill while playing against computer player before they play against another human player. Commonly used algorithm to search for most optimal way AI can use to reach their destination is A algorithm. But A* is not always the best solution for pathfinding. In this study we try to implement Basic Theta* algorithm on a Turn-based Strategy Game called Hexaconquest. Basic Theta* algorithm performance will then be compared to Hexaconquest's original pathfinding algorithm, A*. Both algorithm frame per second, running time, and node total cost performance will be compared. From result of this study, we can conclude that Basic Theta* algorithm can give solution with shortest route, but A* can give lighter and faster solution.*

Keywords: *search algorithm, A*(A star), Basic Theta*(Basic Theta star), implementat, compare*

1. PENDAHULUAN

Game adalah permainan yang terstruktur, biasanya digunakan untuk bersenang-senang, maupun untuk edukasi(Mark, 2002). *Game* berbeda dari pekerjaan, dimana pekerjaan bertujuan untuk mendapatkan upah. *Game* juga berbeda juga dari seni, dimana seni lebih menjadi sebagai ekspresi dari sebuah elemen keindahan atau ideologis. Namun, perbedaan ini juga tidak memiliki batas yang jelas. Banyak *Game* yang dianggap sebagai pekerjaan (pemain professional dari *Game* olahraga-elektronik) dan juga sebagai seni (beberapa *Game* mementingkan unsur artistik). *Turn-based Strategy* adalah salah satu *genre Game*. Dalam *genre Game* tersebut dua atau lebih pemain yang saling bermusuhan bergantian mengendalikan pasukan atau kerajaan mereka sesuai dengan urutan giliran mereka(Sánchez-Ruiz, et al., 2007). Kendali yang dilakukan pemain dapat berupa memberi perintah pada pasukan, alokasi

sumber daya, dan melatih pasukan baru. *Game* dengan jenis ini dapat meningkatkan kemampuan membuat keputusan pemainnya karena cara bermainnya yang membutuhkan pembuatan keputusan dan merespon apa yang kira-kira akan dilakukan oleh pemain lawan.

Hampir semua *Game Turn-based Strategy* zaman sekarang memberikan fitur singleplayer pada tipe permainan yang dapat dilakukan pemainnya. Jika pemain manusia hanya satu orang, maka pemain lainnya harus digerakkan oleh komputer. Disinilah peran AI(Artificial Intelligence) atau kecerdasan buatan. Kecerdasan buatan digunakan pada *Game* agar pemain manusia dapat merasa seakan-akan melawan manusia sehingga ia dapat melatih kemampuan bermainnya terlebih dahulu dengan komputer sebelum melawan pemain manusia lain.

Algoritme yang sering digunakan oleh AI pada *Game* untuk mencari jalan terbaik menuju lokasi tujuannya adalah algoritme A*. Algoritme ini bekerja dengan membuat jalur efisien yang dapat dijelajahi poin demi poin yang disebut dengan Node(Norsati, et al.,2012). A* mulai dikembangkan pada tahun 1968 dengan menggabungkan konsep dari algoritme Djikstra dan Greedy Best First Search(Firmansyah, et al.,2016). Saat algoritme A* menjelajahi sebuah graph, ia mengikuti jalur dengan nilai bobot terkecil sambil mengingat jalur alternatif yang ada di sepanjang jalan. Jika ada jalur node yang memiliki nilai bobot yang lebih besar dari jalur lain yang sudah ditemukan, maka ia akan membuang jalur yang nilai bobotnya lebih besar dan mengikuti jalur dengan bobot yang lebih kecil. A* akan memilih jalur terbaik pertama yang ditemukan sebagai hasil pencariannya.

Pada tahun 2016, Firmansyah, et al., pada penelitiannya membandingkan performa algoritme pathfinding Basic Theta* dengan A* di sebuah *Game* pathfinding berbasis Android. Algoritme Basic Theta* merupakan algoritme variasi A* yang dipublikasikan pada tahun 2007 oleh Alex Nash, Kenny Daniel, Sven Koenig, dan Ariel Felner(Firmansyah, et al, 2016). Algoritme ini digunakan untuk mengatasi kelemahan A* dimana rute terpendek bukanlah rute yang benar-benar terpendek. Hal ini karena algoritme A* dibatasi dengan mencari rute terpendek dalam suatu grid. Theta* mengganti hal ini agar dapat mencari rute dari segala sudut. Perbedaan algoritme ini dibanding dengan algoritme A* adalah diamana setelah menemukan rute, rute tersebut dicek ulang apakah ada node yang parent dari node tersebut dapat bersambung ke node selanjutnya dari node tersebut. Hasil dari penelitian Firmansyah, et al., adalah performa Basic Theta* lebih baik dibanding dengan A*.

Dalam penelitian ini algoritme Basic Theta* diterapkan pada *Game* Hexaconquest yang merupakan *Game Turn-based Strategy*. Setelah diterapkan, maka performa algoritme akan dibandingkan dengan algoritme A* untuk mengetahui apakah lebih baik. Ada beberapa kriteria yang dapat digunakan untuk mengukur performa algoritme pada pathfinding *Game* yaitu kompleksitas waktu, kinerja sistem, dan opimalitas. Kinerja sistem akan diukur menggunakan rata-rata jumlah frame per waktu atau frame per second selama *Game* berjalan. Kompleksitas waktu akan diukur dengan rata-rata waktu yang dibutuhkan agen AI dalam mencari jalur menuju tujuan yang disebut waktu eksekusi. Dan optimalitas diukur dari jumlah total cost dari node yang dilalui oleh agen AI.

2. HEXACONQUEST

Hexaconquest adalah *Game* ber-*genre Turn-based Strategy*. Pada *Game* ini kedua pemain diberikan satu set pasukan. Kedua pemain kemudian bermain dengan menjalankan pasukan mereka untuk menyerang pasukan lawan pada area permainan yang berupa kumpulan segienam atau hexagon. Pada area permainan Hexaconquest terdapat jenis area yang memiliki halangan sehingga tidak dapat dilewati oleh pasukan pemain. Halangan tersebut dapat berupa batu, pohon atau jurang. Pemain dapat mengalahkan pasukan musuh dengan memberikan damage pada hit point pasukan musuh dengan menyerang pasukan tersebut. Permainan berakhir jika pasukan salah satu pemain habis. Pemain yang menang adalah yang pasukannya masih tersisa.

Pasukan yang dapat digunakan pemain dibagi menjadi beberapa jenis yaitu Hero, Archer, Spearman, dan Paladin. Hero merupakan pasukan terkuat milik pemain. Spearman dapat memberikan damage lebih banyak pada hit point pasukan musuh ketika menyerang pasukan berjenis Paladin. Archer dapat memberikan damage lebih banyak pada hit point pasukan musuh ketika menyerang pasukan berjenis Spearman. Sedangkan Paladin dapat dapat memberikan damage lebih banyak pada hit point pasukan musuh ketika menyerang pasukan berjenis Archer.

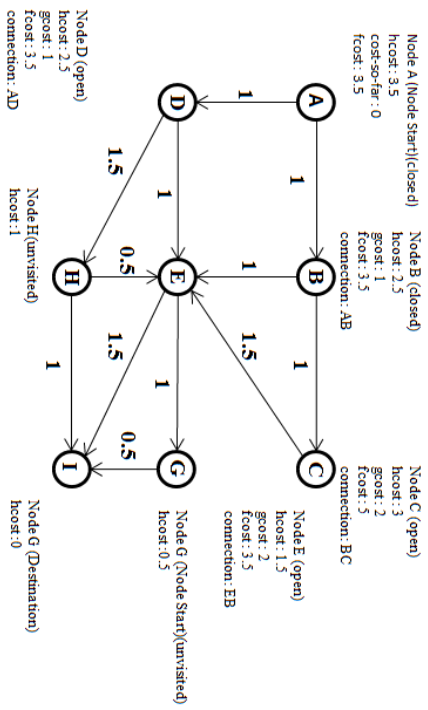
3. ALGORITME A*

Algoritme A* dikembangkan pada tahun 1968 dengan menggabungkan konsep Algoritme Djikstra dan Algoritme Greedy Best First Search(Firmansyah, et al.,2016). A* menghitung cost menggunakan fungsi heuristik untuk memprioritaskan node yang akan dilewati pada peta dibentuk menjadi sebuah graph. Yang menjadi cost-nya adalah jarak yang ditempuh dari satu node ke node lainnya. Algoritme ini efisien terhadap waktu yang digunakan, karena jarak yang dibutuhkan untuk menuju tujuan tak perlu dihitung lagi. Algoritme ini dapat memberikan solusi yang cukup baik untuk pencarian rute pathfinding dan umum digunakan pada pembuatan *Game*.

3.1 Langkah Algoritme

Pertama algoritme akan memastikan letak node awal sebagai Current Node dan node tujuan sebagai Destination. Tiap iterasi, node selanjutnya yang ditelusuri sebagai node yang perlu dilewati untuk mencapai tujuan ditandai sebagai node Current sedangkan node Current yang lama akan ditandai sebagai parent-nya. Pada setiap node akan memiliki niai jarak yang ditempuh dari node sebelumnya yang disebut sebagai gcost. Pada tiap node juga terdapat jarak antara node tersebut dengan node tujuan yang disebut dengan heuristic cost atau hcost yang tidak dapat bernilai negatif. Hasil penjumlahannya akan digunakan sebagai acuan dalam pathfinding. Hasil dari penjumlahan ini biasa disebut fcost seperti dilihat pada persamaan (1).

fcost=gcost+hcost .....Persamaan (1)



Gambar 1. Tahapan Algoritme A*

4. ALGORITME BASIC THETA*

Algoritme Basic Theta* merupakan algoritme variasi A* yang dipublikasikan pada tahun 2007 oleh Alex Nash, Kenny Daniel, Sven Koenig, dan Ariel Felner(Firmansyah, et al, 2016). Algoritme ini digunakan untuk mengatasi kelemahan A* dimana rute terpendek bukanlah rute yang benar-benar terpendek. Hal ini karena algoritme A* dibatasi dengan mencari rute terpendek dalam suatu grid. Theta* mengganti hal ini agar dapat mencari rute dari segala sudut.

Perbedaan algoritme ini dibanding dengan algoritme A* adalah dimana setelah menemukan rute, rute tersebut dicek ulang apakah ada node yang *parent* dari node tersebut yang dapat bersambung ke node selanjutnya dari node tersebut. Contohnya, jika ada node A dengan node selanjutnya adalah node B, maka algoritme akan mencari apakah node *parent* dari A dapat langsung menuju node B tanpa harus melalui A. Jika hal tersebut dapat dilakukan, maka rute yang baru adalah dari *parent* A langsung menuju node B..

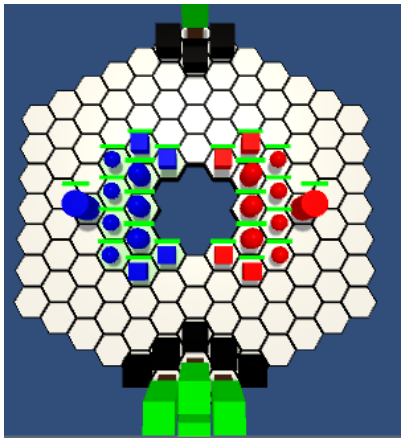
4.1 Langkah Algoritme

Langkah awal pada algoritme Basic Theta* sama dengan algoritme A*. Perbedaannya adalah, setelah iterasi selesai dan rute didapatkan algoritme Basic Theta* akan

melakukan pengecekkan ulang terhadap tiap node yang ada pada rute yang didapat. Parent dari node yang sedang dicek akan dilihat apakah bisa langsung menuju ke node selanjutnya dari node yang sedang dicek tersebut. Jika bisa, maka jalur baru adalah dari parent node yang dicek disambungkan dengan node selanjutnya dari node yang dicek tersebut. Setelah itu node yang dicek akan dikeluarkan dari list node pada rute. Iterasi ini berakhir ketika sebua node sudah dicek.

5. PENERAPAN

Penerapan dilakukan dengan cara manual yaitu melalui antarmuka Unity 5.6.1f1. Pada Gambar 2 merupakan peta simulasi yang telah dibuat. Pada peta tersebut terdapat halangan berupa pohon dan batu. Pasukan kedua pemain akan diletakkan di masing-masing bagian kiri dan kanan dari peta.



Gambar 2. Penerapan Peta Simulasi

AI yang dibuat untuk pengujian diterapkan pada peta. Dua AI yang kemudian akan memainkan *Game* sebagai kedua pemain.

5.1 Penerapan Basic Theta*

Berikut ini merupakan tabel *Pseudocode* dari algoritme Basic Theta*.

Tabel 1. Pseudocode Basic Theta*

Penerapan algoritme Basic Theta*	
Deklarasi **List found, open, close** **Node current, origin**	
	Proses
1	**def FindPath(Node Start, Node Target):**
2	open.Add(start)
3	while open.count > 0 do
4	current = open[0]
5	foreach node in open do
6	if node.fCost < current.fCost then
7	origin = current
8	current = node
9	end if
10	end foreach
11	open.Remove(current)
12	close.add(current)
13	if current.Equals(target) then
14	found = RetracePath(start,current)
15	Break
16	end if
17	foreach neighbor in GetNeighbour(current) do
18	if close.Contains(neighbor) != true then
19	cost=current.gCost+ GetDistance(current,neighbor)
20	if cost < neighbor.fCost || !open.Contains(neighbor) then
20	neighbor.gCost = cost;
21	neighbor.hCost = GetDistance(neighbor,target)
22	neighbor.parent = current
23	if open.Contains(neighbor) != true then
24	open.Add(neighbor)
25	end if
26	end if
27	end if
28	end foreach
29	end while
30	return found

5.2 Penerapan A*

Berikut ini merupakan tabel *Pseudocode* dari algoritme A*

Tabel 2. *Pseudocode* A*

Penerapan algoritme A*	
Deklarasi List found, open, close Node current	
	Proses
1	**def FindPath(Node Start,Node Target):**
2	open.Add(start)
3	while open.count > 0 do
4	current = open[0]
5	foreach node in open do
6	if node.fCost < current.fCost then
7	current = node
8	end if
9	end foreach
10	open.Remove(current)
11	close.Remove(current)
12	if current.Equals(target) then
13	found = RetracePath(start,current)
14	Break
15	end if
16	foreach neighbor in GetNeighbour(current) do
17	if close.Contains(neighbor) != true then
18	cost=current.gCost+ GetDistance(current,neighbor)
19	if cost < neighbor.fCost || !open.Contains(neighbor) then
20	neighbor.gCost = cost;
21	neighbor.hCost = GetDistance(neighbor,target)
22	neighbor.parent = current
23	if open.Contains(neighbor) != true then
24	open.Add(neighbor)
25	end if
26	end if
27	end if
28	end foreach
29	end while
30	return found

6. PENGUJIAN

Pengujian 1 dilakukan dengan melakukan satu sesi *Game*. Kedua pemain akan dikendalikan oleh AI yang menggunakan algoritme yang sama. Pada setiap pengujian digunakan salah satu dari dua jenis algoritme pathfinding yang digunakan yakni A* atau Basic Theta*. Pengujian dilakukan lima kali untuk setiap algoritme dengan menggunakan parameter pengujian yang sama yaitu FPS, waktu eksekusi, total jumlah cost node yang dilewati. Data waktu eksekusi, dan total jumlah cost node yang didapatkan adalah rata-rata dari semua agen pasukan yang bergerak.

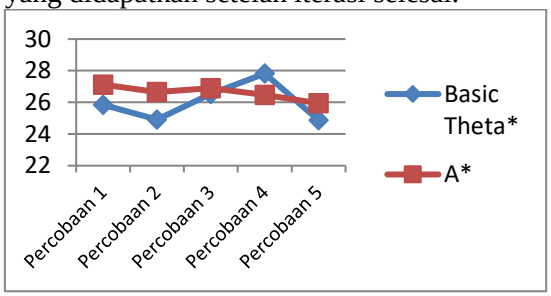
6.1 Hasil Pengujian 1

Pada Tabel 3 Merupakan hasil dari pengujian yang dilakukan.

Table 3 Tabel hasil pengujian 1

No.	Algoritme	*FPS*	Waktu Eksekusi (ms)	Total *Cost* Node
1	A*	27,1	20,7	2,29
	Basic Theta*	25,8	23,0	2,34
2	A*	26,7	17,4	2,07
	Basic Theta*	24,9	21,3	2,16
3	A*	26,9	18,4	2,47
	Basic Theta*	26,5	15,2	2,14
4	A*	26,5	19,4	2,36
	Basic Theta*	27,8	19,2	2,23
5	A*	25,9	20,1	2,14
	Basic Theta*	24,9	20,8	2,12
Rata-rata A*		26,6	19,2	2,25
Rata-rata Basic Theta *		26,0	19,9	2,19

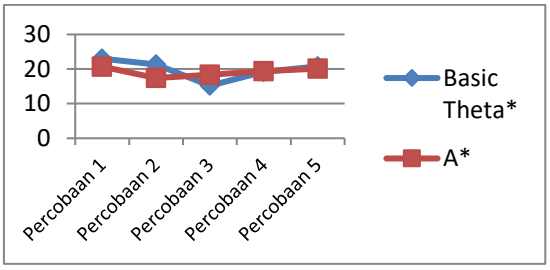
Pada Tabel 3 terdapat fps dan rata-rata fps dari setiap pengujian. Dari hasil ini, terlihat bahwa rata-rata fps dari pengujian menggunakan algoritme Theta* lebih kecil dari hasil yang didapatkan dari pengujian menggunakan algoritme A*. Menurut analisis penulis hal ini dikarenakan satu bagian dari pencarian Theta* harus melalui tahapan pengecekan ulang rute yang didapatkan. Tahap ini membutuhkan lebih banyak komputasi dari algoritme A* karena pada A* tidak dibutuhkan pengecekan ulang rute yang didapatkan setelah iterasi selesai.



Gambar 3. Perbandingan hasil pengujian FPS

Pada Gambar 3 ditunjukkan perbandingan FPS dari kedua algoritme. Pada gambar tersebut dapat dilihat bahwa FPS yang didapatkan dari hasil pengujian kedua algoritme memiliki pola yang berbeda. Pada grafik untuk algoritme A* terlihat pola FPS yang stabil. Namun pada grafik untuk algoritme Basic Theta* terlihat bahwa FPS yang didapatkan dari kelima percobaan tidak stabil. Hal ini menunjukkan bahwa FPS yang didapatkan dari penggunaan algoritme A* lebih stabil dari menggunakan algoritme Basic Theta*.

Lalu pada Tabel 3 juga terdapat waktu eksekusi dan rata-rata waktu eksekusi dari setiap pengujian. Dari hasil ini, terlihat bahwa rata-rata waktu eksekusi dari pengujian menggunakan algoritme Theta* lebih besar dari hasil yang didapatkan dari pengujian menggunakan algoritme A*. Menurut analisis penulis hal ini dikarenakan satu bagian dari pencarian Theta* harus melalui tahapan pengecekkan ulang rute yang didapatkan. Tahap ini membutuhkan lebih banyak komputasi dari algoritme A* karena pada A* tidak dibutuhkan pengecekan ulang rute yang didapatkan setelah iterasi selesai. Karena hal ini maka waktu eksekusi yang dibutuhkan oleh algoritme Theta* akan lebih lama.

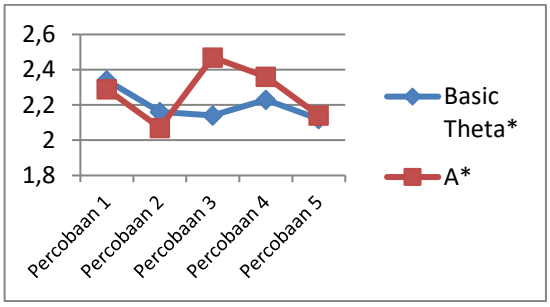


Gambar 4. Perbandingan Hasil Pengujian Waktu Eksekusi

Pada Gambar 4 ditunjukkan perbandingan waktu eksekusi dari kedua algoritme. Pada gambar tersebut dapat dilihat bahwa perbedaan waktu eksekusi yang didapatkan dari hasil pengujian kedua algoritme sangat sedikit. Hal ini ditunjukan oleh kerapatan antar kedua grafik. Dapat dilihat bahwa walaupun pada percobaan ketiga algoritme Basic Theta* mampu lebih cepat dari algoritme A*, algoritme A* tetap unggul dan stabil secara rata-rata performanya.

6.1.3. Hasil Pengujian Jumlah Cost Node Dilewati

Selanjutnya pada Tabel 3 juga terdapat jumlah cost node dan rata-rata jumlah cost node dari setiap pengujian. Dari hasil ini, terlihat bahwa rata-rata jumlah cost node dari pengujian menggunakan algoritme Theta* lebih kecil dari hasil yang didapatkan dari pengujian menggunakan algoritme A*. Menurut analisis penulis hal ini berarti algoritme Theta* berhasil memberikan solusi pathfinding yang lebih baik dari A*. Namun, perbedaan hasil yang didapatkan tidak signifikan. Kemungkinan hal ini disebabkan oleh jenis *Game* yang merupakan *Turn-based Strategy* yang menggunakan tile berjenis hex. Pada jenis tile seperti ini tidak banyak pengoptimasian rute yang dapat dilakukan oleh Theta*.



Gambar 5. Perbandingan Hasil Pengujian Jumlah Cost Node yang Dilewati

Pada Gambar 5 ditunjukkan perbandingan jumlah cost node dari kedua algoritme. Pada gambar tersebut dapat dilihat bahwa jumlah cost node yang didapatkan dari

hasil pengujian kedua algoritme memiliki pola yang berbeda. Berbeda dari percobaan FPS, pada grafik untuk algoritme Basic Theta* terlihat pola jumlah cost node yang stabil dengan rata-rata yang lebih baik dari A*. Namun pada grafik untuk algoritme A* terlihat bahwa jumlah cost node yang didapatkan dari kelima percobaan tidak stabil. Hal ini menunjukkan bahwa jumlah cost node yang didapatkan dari penggunaan algoritme Basic Theta* lebih baik daripada menggunakan algoritme A*.

6.2 Hasil Pengujian 2

Pada pengujian ini parameter yang akan diuji adalah jumlah *cost* node yang dibutuhkan oleh pasukan-pasukan untuk mencapai node tujuan yang sama. Peta yang digunakan pada pengujian ini adalah sama dengan peta yang digunakan pada pengujian sebelumnya. Pada pengujian ini pemain diberikan satu set pasukan seperti pada pengujian sebelumnya. Namun, pada pengujian ini hanya akan digunakan satu pemain. Pemain tersebut akan menggerakkan pasukannya ke node tujuan yang sudah ditentukan di awal tiap pengujian. Pengujian akan dilakukan sebanyak lima kali untuk setiap algoritma.

Tabel 4. Hasil Pengujian 2

No.	Algoritme	Banyak *Cost* Node
1	Theta*	64
	A*	72
2	Theta*	51
	A*	56
3	Theta*	73
	A*	82
4	Theta*	34
	A*	34
5	Theta*	62
	A*	68

Tabel 4 merupakan hasil dari pengujian 2. Pada percobaan pertama node yang dituju terletak agak jauh dari node awal pasukan. Pasukan dengan algoritma *pathfinding* Basic Theta* membutuhkan 64 *cost* node untuk mencapai tujuannya sedangkan pasukan yang menggunakan A* membutuhkan 72 *cost* node. Pada percobaan kedua node tujuan berjarak cukup dekat dari node awal pasukan. Pasukan dengan algoritma *pathfinding* Basic Theta* membutuhkan 51 *cost* node untuk mencapai tujuannya sedangkan pasukan yang menggunakan A* membutuhkan 56 *cost* node. Pada percobaan ketiga node tujuan berjarak cukup jauh dari node awal pasukan. Pasukan dengan algoritma *pathfinding* Basic Theta* membutuhkan 73 *cost* node untuk mencapai tujuannya sedangkan pasukan yang menggunakan A* membutuhkan 82 *cost* node. Pada percobaan keempat node tujuan berjarak sangat dekat dari node awal pasukan. Pasukan dengan algoritma *pathfinding* Basic Theta* membutuhkan 34 *cost* node untuk mencapai tujuannya sedangkan pasukan yang menggunakan A* membutuhkan 34 *cost* node. Pada percobaan kedua node tujuan berjarak cukup dekat dari node awal pasukan. Pasukan dengan algoritma *pathfinding* Basic Theta* membutuhkan 62 *cost* node untuk mencapai tujuannya sedangkan pasukan yang menggunakan A* membutuhkan 68 *cost* node. Dari semua hasil percobaan-percobaan ini terlihat bahwa Basic Theta* dapat memberikan solusi untuk mencapai tujuan dengan menggunakan *cost* node yang lebih sedikit dibanding dengan rute yang didapatkan dari *pathfinding* A*.

7. ANALISIS HASIL PENGUJIAN

Dari hasil pengujian dari penerapan algoritme pathfinding Basic Theta* dan Algoritme A*. Analisis dilakukan terhadap hasil pengujian FPS, rata-rata cost node yang dilewati, dan waktu eksekusi algoritme.

Pengujian dilakukan menggunakan satu peta, yakni peta standar permainan Hexaconquest. Pada hasil pengujian FPS yang dapat dilihat di Tabel 1, FPS yang dapat dicapai oleh pengujian menggunakan algoritme Basic Theta* adalah tertinggi 27,82 FPS pada percobaan ke 4, terendah 24,87 FPS pada percobaan ke-5, sedangkan rata-ratanya adalah 25,99 FPS. Untuk algoritme A* FPS tertingginya adalah 27,12 FPS pada percobaan ke-1, terendahnya 25,94 pada percobaan ke-5, dan rata-ratanya adalah 26,62 FPS. Dari data hasil pengujian ini, terlihat bahwa walaupun FPS tertinggi dicapai oleh algoritme Basic Theta* pada salah satu percobaanya, namun nilai FPS

terendah dari hasil pengujian algoritme Basic Theta* memiliki perbedaan yang cukup jauh dengan hasil terendah yang didapatkan dari pengujian menggunakan algoritme A*. Selain itu, dari rata-rata hasil kedua pengujian terlihat bahwa rata-rata FPS yang dihasilkan oleh penggunaan algoritme A* lebih baik dibandingkan dengan rata-rata hasil dari penggunaan algoritme Basic Theta*. FPS lebih rendah yang ditunjukkan oleh hasil pengujian menggunakan algoritme Basic Theta* dapat diakibatkan oleh cara kerja algoritme Basic Theta*, dimana ia melakukan pengecekan ulang terhadap rute yang sudah didapatkan dari iterasi awal untuk mencari jalur yang lebih optimal. Dengan data-data hasil pengujian tersebut, dapat disimpulkan bahwa pada *Game* Hexaconquest penggunaan algoritme A* dapat menghasilkan FPS yang lebih baik dari menggunakan algoritme Basic Theta*.

Untuk hasil pengujian waktu eksekusi yang dapat dilihat pada Tabel 3, waktu eksekusi yang dapat dicapai oleh algoritme Basic Theta* adalah terlambat 23,0 ms pada percobaan ke-1, tercepat adalah 15,2 ms pada percobaan ke-3, sedangkan rata-ratanya 19,9 ms. Untuk algoritme A*, waktu eksekusi terlambatnya adalah 20,8 ms pada percobaan ke-5, tercepatnya adalah 17,4 ms pada pengujian ke-2, dengan rata-rata 19,2 ms. Dari data hasil pengujian ini, terlihat bahwa algoritme Basic Theta* dapat menghasilkan waktu eksekusi jauh lebih cepat dari algoritme A* pada percobaan ke-3. Namun pada percobaan lainnya terlihat bahwa waktu eksekusi A* secara stabil lebih cepat dari algoritme Basic Theta*. Pada rata-rata dari hasil pengujian yang telah dilakukan terlihat bahwa algoritme A* sedikit lebih unggul dalam waktu eksekusi. Hal ini dikarenakan pada algoritme Basic Theta* diperlukan pengecekan ulang rute yang membuat waktu yang diperlukan untuk mengeksekusi algoritme menjadi lebih tinggi. Dari data hasil pengujian tersebut dapat disimpulkan bahwa pada *Game* Hexaconquest penggunaan algoritme A* dapat memberikan waktu eksekusi sedikit lebih cepat dibanding dengan algoritme Basic Theta*.

Untuk hasil pengujian rata-rata jumlah cost node yang dilewati yang dapat dilihat pada Tabel 3, rata-rata jumlah cost node yang dapat dicapai oleh algoritme Basic Theta* adalah terbanyak 2,34 node pada percobaan ke-1, terendah adalah 2,12 node pada percobaan ke-5, sedangkan rata-ratanya 2,19 node. Untuk algoritme A*, rata-rata jumlah cost node terbanyak adalah 2,47 node pada percobaan ke-3, terrendahnya adalah 2,07 node pada pengujian ke-2, dengan rata-rata 2,25 node. Hasil rata-rata jumlah cost node terbanyak dan terendah ada pada percobaan menggunakan algoritme A*, namun pada rata-ratanya terlihat bahwa algoritme Theta* lebih unggul. Hal ini dikarenkan oleh cara kerja algoritme Theta* dimana algoritme tersebut melakukan pengecekkan ulang pada rute yang telah didapat dari pencarian awal untuk mencari rute lebih optimal. Dari data hasil pengujian tersebut dapat disimpulkan bahwa algoritme Basic Theta* mampu menghasilkan jalur yang lebih pendek dari algoritme A*.

Pada pengujian kedua terlihat bahwa semua hasilnya Basic Theta* dapat memberikan banyak *cost* node yang digunakan lebih sedikit dibanding dengan A*. Pada percobaan kedua dan keempat dapat terlihat bahwa perbedaan banyak *cost* yang digunakan kedua algoritma tidak terlalu berbeda. Hal ini dikarenakan jarak node awal dengan node tujuan yang tidak jauh sehingga kelebihan Basic Theta* bila dibandingkan dengan A* tidak terlalu terlihat. Namun pada hasil percobaan lainnya dapat terlihat bahwa Basic Theta* dapat memberikan hasil rute jalur menuju node tujuan dengan *cost* yang lebih sedikit. Dari hasil pengujian ini dapat disimpulkan bahwa Algoritme Basic Theta* dapat memberikan *cost* yang lebih sedikit dibandingkan dengan Algoritme A* dalam pencarian jalur pada game Hexaconquest.

8. KESIMPULAN

Berdasarkan hasil pengujian dan analisis yang dilakukan pada penerapan algoritme Basic Theta* pada bab sebelumnya, didapatkan kesimpulan sebagai berikut:

1. Penerapan algoritme Basic Theta* pada *Game* Hexaconquest dapat dilakukan menggunakan *Game* engine Unity dengan memasukkannya pada kode untuk melakukan patfinding.
2. Hasil FPS yang didapatkan dari algoritme A* lebih stabil dibanding dengan yang didapat dari Basic Theta*. Rata-rata FPS yang didapat dari pengujian algoritme Basic Theta* lebih rendah dibandingkan dengan algoritme

A*. Hal ini berarti bahwa algoritme Basic Theta* memberikan performa *Game* lebih buruk dibandingkan dengan algoritme A*. Namun perbedaan ini hanya sekitar 2,367%, tidak dapat dikatakan performa yang jauh lebih buruk. Rata-rata waktu eksekusi algoritme Basic Theta* lebih rendah dari A* adalah hal yang wajar karena tahapan yang dilakukan pada algoritme Basic Theta* lebih banyak dari algoritme A*. Namun, perbedaan hasil pengujian waktu eksekusi yang hanya sekitar 3,518% menunjukkan bahwa perbedaan waktu eksekusi yang terjadi tidak signifikan. Pada hasil pengujian jumlah cost node yang dilewati terlihat bahwa hasil yang didapatkan dari pengujian menggunakan algoritme Basic Theta* lebih baik dibandingkan dengan menggunakan algoritme A*. Pada pengujian kedua terlihat bahwa algoritme Basic Theta* dapat memberikan banyak *cost* node yang lebih sedikit bila dibandingkan dengan algoritme A*.

Tidak seperti pada hasil penelitian ini, hasil penelitian yang dilakukan Firmansyah, et al, yang merupakan dasar penggunaan algoritme Basic Theta* pada penelitian ini, terlihat bahwa hasil pengujian menggunakan algoritme Basic Theta* jauh lebih baik dari penggunaan algoritme A*. Hal ini kemungkinan disebabkan oleh beberapa faktor yakni jenis *Game* dan ukuran peta *Game*. Pada penelitian yang dilakukan Firmansyah, et al, *Game* Finding The Bones bukan merupakan *Game Turn-based Strategy*. Ini mempengaruhi batas pergerakkan agen yang melakukan pathfinding. Selain itu, ukuran peta yang digunakan juga lebih kecil. Ukuran peta mempengaruhi seberapa luas bagian peta yang harus ditelusuri untuk mencapai tempat yang dituju oleh agen.

Maka dari itu, dapat disimpulkan bahwa performa secara keseluruhan dari algoritme Basic Theta* lebih buruk dibandingkan dengan algoritme A* jika diterapkan pada game Hexaconquest. Namun, algoritme Basic Theta* memiliki kelebihan dalam mencari jalur terpendek untuk pergerakan pasukan bila dibandingkan dengan algoritma A*.

9. SARAN

Untuk meningkatkan hasil yang didapat dari penelitian ini, diharapkan pada penelitian selanjutnya dapat melakukan beberapa perbaikan sebagai berikut:

1. Diharapkan pada penelitian salanjutnya untuk mengoptimasi algoritme Basic Theta* dapat digunakan dengan lebih cocok pada *Game Turn-based Strategy* yang berbasis Hex.
2. Diharapkan agar melakukan percobaan menggunakan *Game* lain dengan peta yang lebih kecil dan besar sehingga dapat terlihat jelas perbedaan performa dari algoritme A* dan Basic Theta*.

DAFTAR PUSTAKA

BAIER, ET AL., 2015. *Fast Algorithm for Catching a Prey Quickly in Known and Partially Known Game Maps.* IEEE Transactions on Computational Intelligence and Ai in *Game*s.

CLAYPOOL K. T. & CLAYPOOL, 2007. *On frame rate and player performance in first person shooter Games.* Multimedia Systems.

ELIZABETH M. & MALATHY, 2015. *Direction Based Heuristic for Pathfinding in Video Game*. IEEE Sponsored Second International Conference on Electronics and Communication Systems.

GRAHAM, ET AL. 2003. *Pathfinding in Computer Games.* The ITB Journal.

HAGELBÄCK, J. 2016. *Hybrid Pathfinding in StarCraft*. IEEE Transactions on Computational Intelligence and Ai in *Game*s.

MARK. 2002 *The Educational Benefits of VideoGame.* Education and Health Journal Vol.20 No.3.

NORSATI, ET AL., 2012. *Investigation of the * (Star) Search Algorithms: Characteristics, Methods and Approaches.* World Applied Programming

FIRMANSYAH, et al., 2016. *Comparative Analysis of A* and Basic Theta* Algorithm in Android-Based Pathfinding Games.* 2016 6th International Conference on Information and Communication Technology for The Muslim World.

SÁNCHEZ-RUIZ, ET AL., 2007. *Game AI for a Turn-based Strategy Game with Plan Adaptation and Ontology-based retrieval ⁎.* Spanish Committee of Science & Technology.