Implementasi Algoritma A Stars, Tilebase Collision Dan Fuzzy Logic Pada Game Strategy

Harso Kurniadi*[1], Kusrini[2]
[1]STMIK AMIKOM Yogyakarta
[2]Magister TeknikInformatikaSTMIK AMIKOM Yogyakarta
E-mail: *[1]**harsok006@gmail.com**,[2]kusrini@amikom.ac.id

Abstrak

Implementasi algoritma A-Stars, Tilebase Collision dan Fuzzy Logic pada arsitektur dan perilaku tempur dalam game strategy menggunakan visual basic 6.0 dengan penelitian perilaku dan gambar karakter yang diambil dari game Red Alert2. Metode penelitian yang digunakanadalah Research and Development. Karena metode ini dapat membuktikan kemampuan algoritma yang digunakan dapat diterapkan pada game strategi. Kegiatan dalam mengukur kemampuan algoritma yang diterapkan pada aplikasi game strategi, dihasilkan formulasi untuk gerak multi direction dari pengembangan formulasi algoritma Steering Behavior. Formulasi multi direction dapat membuat karakter bergerak ke segala sudut mengikuti arah target. Perilaku cerdas karakter dalam game strategi merupakan bagian dari kecerdasan buatan (Artificial Intelligence) yang dibangun dari beberapa algoritma yaitu A-Stars sebagai solusi untuk menentukan pathfinding, Tilebase Collision sebagai solusi untuk mendeteksi tumbukan antara objek satu dengan objek lain, dan Fuzzy Logic menjadi solusi untuk membuat strategi perang dalam game.

Kata kunci: Game Strategy, Multi Direction, A-Stars, Tilebase Collision, Fuzzy Logic.

Abstract

The implementation of algorithms A-Stars, Tilebase Collision and Fuzzy Logic on the architecture and combat behavior in the strategy game using Visual Basic 6.0 with behavioral studies and character images taken from the game Red Alert2. The method used is the Research and Development. Because this method can prove the ability of the algorithm used can be applied to strategy game. Activity in measuring the ability of the algorithm is applied to the application of a strategy game, resulting formulations for motion (multi direction) on the development of algorithms Steering Behavior. Formulations multi direction can make the characters move in any corner following the direction of the target. Intelligent behavior of characters in the strategy game is part of the Artificial Intelligence which is built from multiple algorithms, namely A-Stars as a solution to determine the pathfinding, Tilebase Collision as a solution for detecting collisions between one with another object, and Fuzzy Logic to be the solution to create a war strategy in the game.

Keyword: Game Strategy, Multi Direction, A-Stars, Tilebase Collision, Fuzzy Logic.

1. PENDAHULUAN

Perkembangan *game* tidak lepas dari teknologi yang maju pesat mulai dari fasilitas desain grafis yang semakin baik dan menampilkan kemampuan karakter game berinteraksi dengan pengguna, sehingga memiliki daya tarik tersendiri bagi para pengguna aplikasi *game*. *Game* merupakan sarana dalam interaksi antar manusia atau dengan komputer yang di rangkai dalam suatu aturan tertentu untuk menentukan pemenang. Dalam pembuatan aturan aturan tersebut di perlukan algoritma tertentu yang membentuk kecerdasan buatan pada game. Kegiatan dalam mengukur kemampuan algoritma yang diterapkan pada aplikasi *game* strategidi perlukan beberapa penelitian untuk menggali ide yaitu pada penelitian lain yang berjudul **"**Perancangan *Framework Real Time Strategy Games"* pada penelitian tersebut di latar belakangi dari "informasi mengenai game *Real Time Strategi* yang bersifat *close source* sehingga terdapat kesulitan bagi para *developer game* untuk menggali informasi *source code* tersebut." tujuannya menghasilkan sebuah *framework* untuk *game* berjenis *Real Time Strategy* dengan kecerdasan buatan, dan menghasilkan *game* dari *framework* yang sudah dibuat[1].*Real Time Strategy* (RTS) merupakan jenis game yang menjadi domain yang ideal dalam pengujian kemampuan kecerdasan buatan. *Gamereal time strategy* menunjukkan perlunya heterogen pada pengembang arsitektur *game*, karena mengurangi keputusan kompleksitas domain *game* adalah bukan hal yang sepele [2]. Rangkaian strategi tersebut digunakan untuk membuat kecerdasan buatan pada game strategi. Kecerdasan buatan atau dalam bahasa inggris adalah Artificial Intelligence yang "merupakan proses dimana peralatan mekanik dapat melaksanakan kejadian-kejadian dengan menggunakan pemikiran atau kecerdasan seperti manusia"[3]. Kecerdasan buatan dapat membuat tampilan karakter pada *game* yang memiliki perilaku seperti karakter yang hidup di dunia nyata. Menurut Benufinit, dkk, 2015, dalam paper yang berjudul "Manuver Kelompok NPC Berbasis Boids Pengembangan Game Real Time Strategy**"**menjelaskan bahwa "taktik merupakan tingkat terendah dalam perencanaan, melibatkan unit-unit kecil mulai dari satu karakter hingga beberapa ratus karakter *game*. Model yang di gunakan pada penelitian tersebut dalam bermanuver adalah model *flocking*, dimana tim *Army* terdiri dari tiga kompi, kompi yang pertama mengumpan musuh keluar dari markasnya, kompi yang kedua melakukan penyerangan ke daerah pertahanan musuh, sedangkan kompi yang ketiga melindungi pasukan dari serangan musuh"[4].Dalam melakukan penyerangan ke daerah pertahanan musuh diperlukan algoritma untuk menentukan rute terpendek. Menurut Rahadiansyah, 2015, dalam paper yang berjudul "Kompleksitas Algoritma A* Pada Implementasi PassiveAI Untuk *Game Mobile AI*",menjelaskan tentang algoritma A-Stars dalam pencarian rute terpendek dan juga dapat menggunakan pencarian heuristic[5].*Pathfinding* dengan algoritma A * (A Star) dalam menentukan jalan terpendek pada grafik komputer tidak sama dengan jalur terpendek di lingkungan di kehidupan nyata. Isu lain yang terkait dengan A * adalah bahwa, ketika ukuran peta secara signifikan lebih besar, algoritma A * tidak dapat menemukan jalan minimum untuk tujuan dalam jumlah terbatas waktu[6]. Dalam kegiatan menemukan jalan minimum untuk tujuan, diperlukan pendeteksian tumbukan pada tiap karakter yang akan di buat. Deteksi adanya *collision* dari satu objek dengan objek lainnya yang saling berpapasan akan terlihat seperti bertubrukan maka game trampak lebih realistis[7]. Kemampuan pada karakter *game* strategi dalam mendeteksi halangan dan menentukan arah untuk menghindar terdapat dalam Algoritma*Steering Behavior. "*Algoritma *Steering Behavior* merupakan algoritma yang akan di terapkan dalam penelitian ini bertujuan untuk membantu karakter bergerak secara realistis, dengan menggunakan kekuatan sederhana yang dikombinasikan untuk menghasilkan seperti pada kehidupan nyata, navigasii mprovisasi di sekitar lingkungan karakter"[8]. Faktor yang menentukan perilaku adalah jarak musuh dengan pemain, dengan menghasilkan *output* yaitu musuh yang memiliki kepintaran dalam bertindak"[9]. Algoritma *Fuzzy logic* menjadi solusi dalam membuat perilaku cerdas, menurut Shaout, dkk, 2006,dengan judul: "*Real Time Game Design of Pacman using Fuzzy Logic*". Menjelaskan tentang pembuatan perila kukarakter musuh dalam *game* Pacman menggunakan "*Fuzzy* IF-THEN rule"[10].

2. METODE PENELITIAN

Metode penelitian yang digunakan penulis dalam menyelesaikan masalah adalah *Research and Development*, karena *Research and Development* adalah "metode penelitian yang digunakan untuk menghasilkan produk tertentu, dan menguji keektifan produk tersebut"[11]. Penelitian untuk menghasilkan suatu produk dibutuhkan penelitian yang bersifat analisis kebutuhan dan melakukan pengujian pada kefektifan produk tersebut. Penelitian dilakukan pada beberapa algoritma yang digunakan untuk membuat perilaku cerdas pada *game* strategi antara lain :

- Penerapan algoritma A-Stars untuk menentukan rute perjalanan karakter *game.*
- Penerapan algoritmaTilebase Collision untuk mendeteksi tabrakan pada karakter *game* yang akan di buat.
- Penerapan algoritma*Fuzzy Logic* untuk pembuatan strategi dalam mengatur taktik pertempuran.

Algoritma tersebut akan diterapkan pada tiap karakter *game* stretegi baik karakter yang di gunakan pemain maupun karakter pada pihak lawan. Pihak lawan yang pada penelitian ini adalah karakter yang di jalankan komputer. Pada penerapanya algoritma akan di sesuaikan dengan kebutuhan dalam pembuatan *game* strategi.

2.1. Pengumpulan Data

Metode Observasi digunakan untuk mengumpulkan referensi dari berbagai penelitian antara lain:

- Penelitian dengan judul "Perancangan *Framework Real Time Strategy Games".*
- Penelitian dengan judul "Building Human-Level AI for Real-Time Strategy Games".
- Penelitian dengan judul "Simulasi Perilaku Tempur Pada Sekumpulan NPC Berbasis Boid".
- Penelitian dengan judul "Manuver Kelompok NPC Berbasis Boids Pengembangan Game Real Time Strategy".
- Penelitian dengan judul "Kompleksitas Algoritma A* Pada Implementasi Passive AI Untuk *Game Mobile AI".*
- Penelitian terhadap *game* jenis *real time strategy* seperti Red Alert 2.

Data penelitian yang di butuhkan untuk melakukan pengamatan sebagai berikut :

- Data perilaku karakter dalam berbagai kondisi medan permainan.
- Data strategi tempur dalam *game* strategi.
- Data aturan main dalam *game* strategi.
- Data algoritma yang digunakan untuk membuat kecerdasan buatan pada *game* strategi.
- Data resource tentang sprite dan sound untuk menyelesaikan dan mengembangkan aplikasi game strategi.

2.2. Analisa

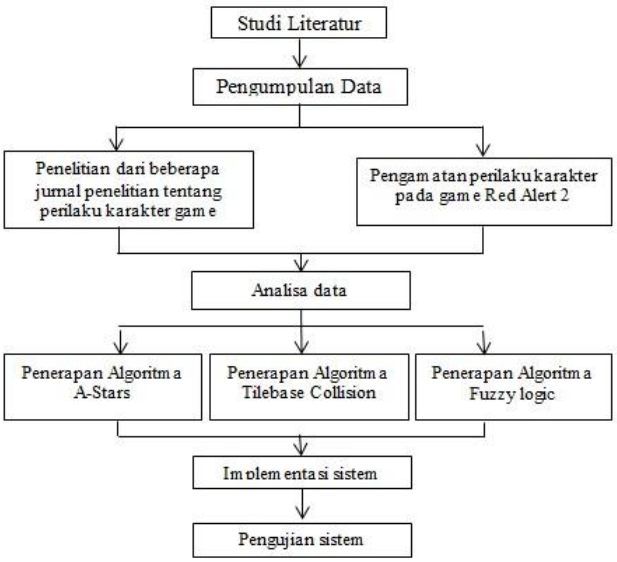
Metode analisis digunakan untuk melakukan analisa terhadap kebutuhan algoritma untuk membuat kecerdasan buatan pada game strategi antara lain:

- Perilaku karakter dalam *game* strategi
- Strategi tempur dan aturan main dalam *game* strategi.
- Algoritma A-Stars untuk menentukan rute perjalanan karakter game strategi
- Algoritma Tilebase Collision untuk mendeteksi tabrakan pada karakter *game* yang akan di buat.
- Algoritma *Fuzzy Logic* untuk pembuatan strategi dalam mengatur taktik pertempuran.
- Data resource tentang sprite dan sound untuk menyelesaikan dan mengembangkan aplikasi game strategi

2.3. Alur Penelitian

Alur penelitian dalam penulisan ini merupakan diagram alur dari penelitian di mulai dari studi literatur dari buku, jurnal dan berbagai referensi penelitian lain yang relevan dengan penelitian ini, kemudian pengumpulan data penelitian untuk melakukan pengamatan, diteruskan dengan analisa data untuk mengolah data yang adapadapengumpulan data, Data resource tentang sprite dan sound

untuk menyelesaikan dan mengembangkan aplikasi game strategiimplementasi dilakukan dengan membuat aplikasi *game* strategi dan terakhir adalah pengujian dilakukan dengan meminta 3 programer dari Tahoegames yang merupakan salah satu game developer di Kota Kediri.Diagram alur penelitian ada pada gambar 2.1. sebagai berikut :



Gambar 1. Alur Penelitian

3. HASIL DAN PEMBAHASAN

3.1. Pengujian Algoritma A Star

Dalam pengujian algoritma A Star ini akan dilakukan pengujian fungsi pencarian rute dalam menghindari rintangan. Pada pembuktian ini di buat empat buah benda sebagai *start*, *path*, *goal*, balok dan lima buah benda sebagai node yang telah di set pada setiap sudut balok yang menghalangi jalur dari start. Pembuktian ini akan dijalankan pada simulasi, kemudian dilakukan pengaturan titik awal dengan koordinat bebas dan pengaturan pada titik akhir dengan koordinat bebas. Langkah awal dalam menentukan rute karakter dengan algoritma A Star adalah membuat gerak *Multi Direction* untuk menentukan karakter bergerak secara realistis. Algoritma *Multi Direction* adalah hasil pengembangan dari algoritma *Steering Behavior* dengan formulasi sebagai berikut:

X1 = abs (X_target - X_position)
Y1 = abs (Y_target - Y_position)
NX1= X1 / (X1 + Y1)
NY1= Y1 / (X1 + Y1)
Gerak_X = Gerak_X + (max_velocity * NX1)
Gerak_Y = Gerak_Y + (max_velocity * NY1)

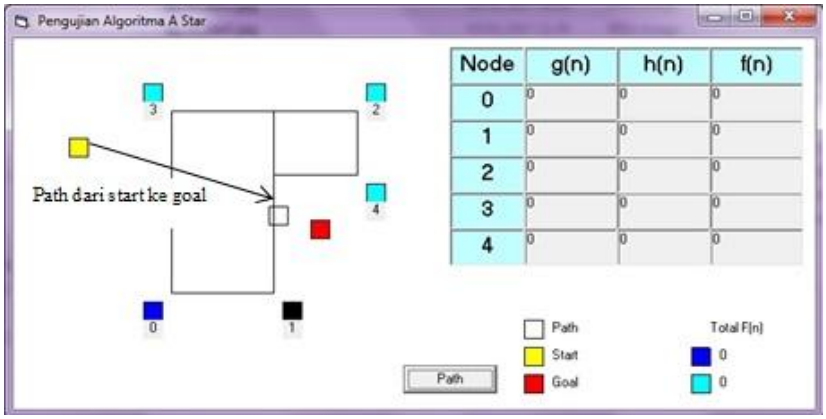
Dimana :

X1 adalah nilai absolut dari perhitungan jarak antara target dengan posisi pada koodinat X.
Y1 adalah nilai absolut dari perhitungan jarak antara target dengan posisi pada koodinat Y.
NX1 adalah nilai X yang dapat berubah untuk mengikuti titik X target.
NY1 adalah nilai Y yang dapat berubah untuk mengikuti titik Y target.

Dengan adanya algoritma *Multi Direction* maka pada penelitian ini cukup mengatur node ke sudut sudut objek penghalang kemudian memasukan rumus f(n) = g(n)+h(n) untuk mencari jarak terpendek. Penggunaan algoritma *Multi Direction* untuk menggerakkan path dari *start* menuju *goal* dengan sudut bebas. Kemudian algoritma *Multi Direction* di masukkan pada pseudocode sebagai berikut :

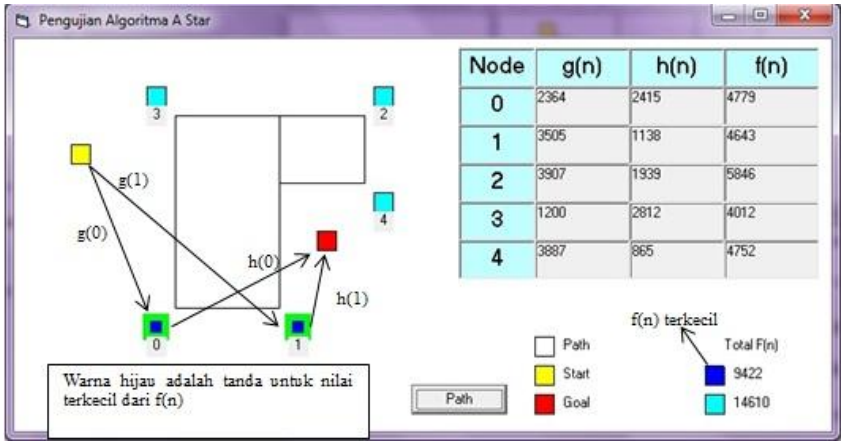
```
X1 = Abs(goal.Left - Path.Left)
Y1 = Abs(goal.Top - Path.Top)
NX1 = X1 / (X1 + Y1)
NY1 = Y1 / (X1 + Y1)
  Select Case X
      Case 1
         Path.Left = Path.Left + (60 * NX1)
      Case 2
         Path.Left = Path.Left - (60 * NX1)
    End Select
    Select Case Y
      Case 1
         Path.Top = Path.Top + (60 * NY1)
      Case 2
         Path.Top = Path.Top - (60 * NY1)
    End Select
If Path.Left < goal.Left - 20 Then X = 1
If Path.Left > goal.Left + 20 Then X = 2
If Path.Top < goal.Top - 20 Then Y = 1
If Path.Top > goal.Top + 20 Then Y = 2
If Path.Top > goal.Top - 20 And Path.Top < goal.Top + 20 _
And Path.Left > goal.Left - 20 And Path.Left < goal.Left + 20 Then _
Y = 0: X = 0
```

Perjalanan *path* yang di maksudkan adalah dari *start* ke *goal* untuk proses split grup dari node pada gambar 2 sebagai berikut :



Gambar 2. Proses *split node* A Star

Proses *split node* adalah untuk memisahkan kelompok node dari sisi kanan karakter dan sisi kiri karakter, pada pengujian ini dibedakan dengan dua warna node yaitu *blue node* dan *cyan node*. Langkah berikutnya adalah memasukan rumus f(n) = g(n)+h(n) pada masing-masing kelompok untuk mencari jalur terpendek dengan gambar 3 sebagai berikut :



Gambar 3. Proses pencarian jalur terpendek

- Langkah awal dalam proses mencari jalur terpendek adalah menghitung nilai g(n) adalah nilai dari hasil perhitungan jarak dari *start* ke *node*, kemudian menghitung nilai h(n) adalah nilai dari hasil perhitungan jarak dari *node* ke *goal*, selanjutnya adalahmencari nilai f(n) dari hasil g(n) + h(n).
- Langkah kedua adalah menghitung kelompok node yang sudah dilakukan split dengan fungsi dari varibel (fn) untuk menghitung kelompok *cyan node*dan (fm) untuk menghitung kelompok *blue node*.
- Langkah ketiga adalah mencari nilai terkecil dengan membandingkan nilai fn dan fm, kemudian memberi tanda hijau pada kelompok node dengan nilai terkecil.

Berdasarkan pengujian didapatkan hasil sebagai berikut :
Proses menghitung kelompok *cyan node.*

f(2) = g(2) + h(2)
f(2) = 3907 + 1939
f(2) = 5846
f(3) = g(3) + h(3)
f(3) = 1200 + 2812
f(3) = 4012
f(4) = g(4) + h(4)
f(4) = 3887 + 865
f(4) = 4752

Maka perhitungan variabel fn adalah sebagai berikut:

fn = f(2) + f(3) + f(4)
fn = 5846 + 4012 + 4752
fn = 14610

Proses menghitung kelompok *blue node.*

f(0) = g(0) +h(0)
f(0) = 2364 +2415
f(0) = 4779
f(1) = g(1) +h(1)
f(1) = 3505 +1138
f(1) = 4643

Maka perhitungan variabel fm adalah sebagai berikut:

fm = f(0) + f(1)
fm = 4779 + 4643
fm = 9422

Dari proses perhitungan kedua kelompok node tersebut didapatkan nilai dari variabel fn = 14610 dan fm = 9422, maka nilai terkecil dari kelompok node adalah variabel fm.

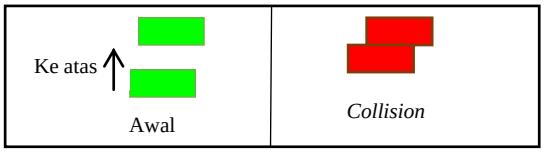
Dari pengujian tersebut dibuktikan bahwa dalam menentukan rute karakter tidak di perlukan perhitungan node pada tiap satu langkah karakter.Pada tabel 1 perbandingan pengujian algoritma A* terlihat proses pencarian rute hanya dengan menghitung node pada tiap sudut penghalang. Kelemahanya adalah objek pada start membutuhkan objek lain sebagai path untuk melakukan split node.

Tabel 1. Perbandingan Pengujian Algoritma A*

Penelitian	Gerak	Mencari rute terpendek
Game Strategi	Multi Direction	Hanya menghitung node pada sudut penghalang
Ena Burena	Menghitung node tiap 1 langkah	Menghitung node tiap 1 langkah

3.2. Pengujian algoritma Tilebase Collision

Dalam pengujian algoritma *Tilebase Collision* dilakukan dengan penerapan algoritma *multi direction* pada objek untuk dapat bergerak ke segala sudut mengikuti arah target.Jika terjadi *collision detection* maka algoritma *Fuzzy Logic* akan mengatur objek keluar dari *collision*. Pengujian ini untuk mengetahui ketepatan objek dalam menghindari *collision* dari satu objek dengan objek lainnya. Dalam proses *collision detection* terdapat pada gambar 4 sebagai berikut :



Gambar 4. Gambaran proses *collision detection*

Pada simulasi *collision detection* menggunakan *Bounding Box*, saat terjadi *collision* objek akan berubah warna menjadi merah, jika bebas dari collision maka objek akan berwarna hijau. Langkah awal dalam proses *collision detection* adalah menentukan bagian atau sisi yang terjadi *collision* dengan pseudocode sebagai berikut :

```
For a = 0 to 8
For b = 0 to 8
        If Collision Detection = True Then
        If P1(a).Left < P1(b).Left - (P1(a).Width - 20) Then Side(a) = 1
        If P1(a).Left > P1(b).Left + (P1(b).Width - 20) Then Side(a) = 2
        If P1(a).Top < P1(b).Top - (P1(a).Height - 30) Then Side(a) = 3
        If P1(a).Top > P1(b).Top + (P1(b).Height - 20) Then Side(a) = 4
        End If
Next
Next
```

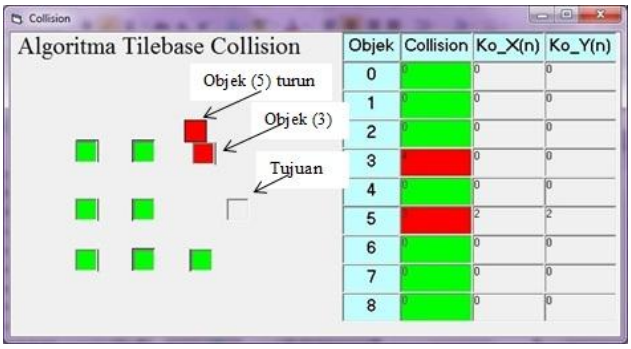
Dalam proses tersebut berfungsi untuk mengetahui sisi objek yang telah terjadi *collision*, pada simulasi ini ditentukan empat sisi yaitu :

a. Side (a) = 1 adalah kondisi jika objek (a) berada disebelah kiri objek (b).
b. Side (a) = 2 adalah kondisi jika objek (a) berada disebelah kanan objek (b).
c. Side (a) = 3 adalah kondisi jika objek (a) berada disebelah atas objek (b).
d. Side (a) = 4 adalah kondisi jika objek (a) berada disebelah bawah objek (b).

Fungsi dari empat sisi tersebut untuk menentukan arah objek keluar dari *collision* dan kembali menuju target. Arah objek ditentukan dengan aturan sebagai berikut :

Jika X(a) = 1 maka arah objek kekiri.
Jika X(a) = 2 maka arah objek kekanan.
Jika Y(a) = 1 maka arah objek keatas.
Jika Y(a) = 2 maka arah objek kebawah.

Pengujian algoritma *Tilebase Collision* untuk mendeteksi *collision* dan proses objek keluar dari collision pada gambar 5 sebagai berikut :



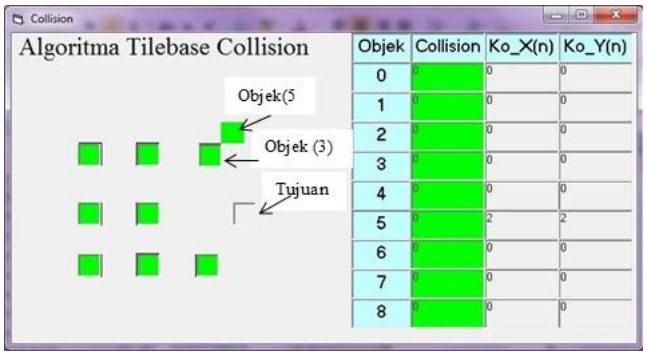
Gambar 5. Proses mendeteksi *collision*

Proses objek keluar dari *collision* yang terjadi adalah objek(5) bergerak turun dan terjadi collision dengan objek(3), diketahui sisi *coliision* objek (5) adalah side (5) = 3, X(5) = 2 dan Y(5) = 2, maka dapat ditentukan pseudocode sebagai berikut :

If Side(5) = 3 And X(5) = 2 And Y(5) = 2 Then P1(5).Left = P1(5).Left + 50

Dimana P1(5) = objek(5).

Maka didapatkan hasil pada gambar 6 sebagai berikut :



Gambar 6. Proses keluar dari *collision*

Objek (5) diatur oleh rule bergerak kekanan sampai warna dari objek kembali hijau yang menandakan objek keluar dari *collision*.

Pengujian dengan simulasi antara objek (5) dengan objek (3)diketahui bahwa objek (5) secara tepat mampu menghidar dari *collision* sebelum terjadi *collision* yang semakin dalam. Kelemahannya penggunaan *bounding box* adalah pendeteksian *collision* tidak maksimal pada objek dengan bentuk selain kotak. Pada tabel 2 perbandingan pengujian algoritma *Tilebase Collision*dengan simulasi yang dilakukan, *collision* dapat terdeteksi.

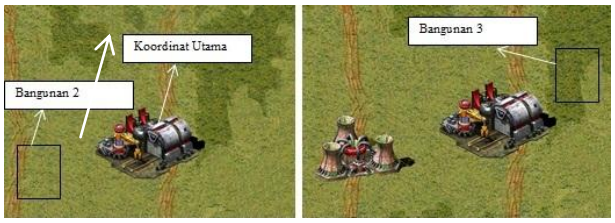
Tabel 2. Perbandingan Pengujian Algoritma *Tilebase Collision*

Penelitian	*Collision detection*	Simulasi 2 objek	Simulasi 1 ke banyak objek
Game Strategi	*Bounding Box*	ya	ya
Ena Burena	*Bounding Capsules*	ya	ya

3.3. Pengujian algoritma *Fuzzy Logic*

3.3.1 Pengujian *Fuzzy Logic* pada pembangunan pangkalan dan pertahanan

Pada Non Player Character (NPC),*Fuzzy Logic* akan mengatur langkah awal dalam membangun pangkalan pertahanan dengan parameter koordinat bangunan utama dan *mission*. koordinat bangunan utama digunakan sebagai acuan koordinat bangunan lain dan *mission* digunakan untuk menentukan koordinat sesuai dengan *map*. Proses awal pada gambar 7 sebagai berikut :



Gambar 7. Proses pembuatan pangkalan pertahanan NPC

Koordinat bangunan ditentukan oleh algoritma *fuzzy logic* dengan pengaturan koordinatsesuai *Mission*dan secara acak. Pada tabel 3 hasil pengujian *Fuzzy Logic* pada pembangunan pangkalan dan pertahanan dengan *map* yang berbeda.

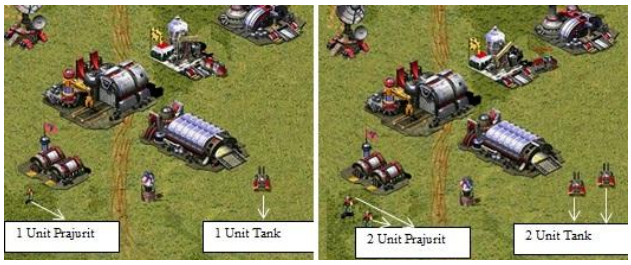
Tabel 3. Hasil Pengujian *Fuzzy Logic* pada pembangunan pangkalan dan pertahanan

Uji	Koordinat	Map 1		Map 2		Map 3	
		P1	P2	P1	P2	P1	P2
1	X	800	3050	800	750	800	3050
	Y	350	1700	1990	550	350	1700
2	X	3100	750	800	3050	800	750
	Y	1990	1900	350	550	350	1900

Dari tabel hasil pengujian pada tabel 3, terlihat bahwa koordinat P1(pengguna) dan koordinat P2(NPC)beda tempat dan tidak ketemu dalam satu tempat.

3.3.2. Pengujian *Fuzzy Logic* pada pembuatan unit penyerang

Pada pembuatan unit penyerang, *Fuzzy Logic* akan mengatur proses pembuatan unit penyerang dengan parameter jumlah dan *way point*. Dalam membuat unit penyerang diatur secara bertahap, yaitu 1 unit di buat pada masing masing *factory*. Proses dalam membuat unit penyerang terdapat pada gambar 8 sebagai berikut :



Gambar 8. Proses pembuatan unit penyerang

Penentuan way point digunakan untuk menghindari penumpukan karakter pada satu tempat.

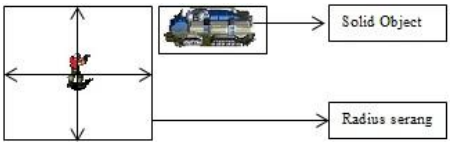
Tabel 4. Pengujian *Fuzzy Logic* pada pembuatan unit penyerang

Jumlah unit	Way Point X	Way Point Y
1	342	205
2	284	275
3	284	205
4	226	270

Dari tabel hasil pengujian pada tabel 4, terlihat bahwa way point beda tempat dan tidak terjadi penumpukan karakter.

3.3.3. Pengujian *Fuzzy Logic* padapengaturan perilaku karakter dalam strategi perang

Pada pengaturan perilaku karakter dalam strategi perang,*Fuzzy Logic* akan mengatur perilaku karakterdengan parameter karakter, target,aksi dan prioritas target.Dalam mengatur jumlah penyerang awal dari unit prajurit ditentukan empat orang dan dari unit tank adalah empat unit, kemudian mengatur target tim adalah daerah lawan dengan mengaktifkan radar unit terdapat pada gambar 10 sebagai berikut:



Gambar 10. Proses mengatur perilaku tempur pada karakter.

Dalam mengatur perilaku tempur pada karakter dalam strategi perang dengan pseudocode sebagai berikut :

```
Karakter orang :
    If (Army_near_Solid_Object) then
        If (enemy = enemy) then defense : attack
        If (enemy = destroy) then Next_Target
        If (Solid_Object <> enemy) then Call Tilebase Collision
    End If
Karakter tank :
    If (Tank_near_Solid_Object) then
        If (Solid_Object = enemy) then attack
        If (enemy = destroy) then Next_Target
        If (Solid_Object <> enemy) then Call Tilebase Collision
    End If
Karakter menara pertahanan :
    If (Tower_near_enemy) then
        If (enemy = tank) then attack
    End If
```

Tabel 5. Pengujian *Fuzzy Logic* padapengaturan perilaku karakter dalam strategi perang

Tipe karakter	Target	Aksi	Prioritas target
orang	Orang + tank + gedung	Bertahan + menyerang	Menyerang orang
tank	Orang + tank + gedung	menyerang	Menyerang orang
menara	tank	Bertahan	tank

Dari tabel hasil pengujian pada tabel 5, terlihat bahwa *fuzzy logic* dapat mengatur perilaku karakter dalam medan perang dengan menentukan aksi sesuai tipe karakter dan menentukan prioritas target.

Dalam menentukan prioritas target digunakan jarak musuh yang terdekat dengan mengutamakan target orang kemudian tank dan terakhir adalah menghancurkan gedung.

Tabel 6. Perbandingan Pengujian Algoritma *Fuzzy Logic*

Penelitian	Pengujian	Parameter
Game Strategi	1. Pembangunan	1.a. Koordinat b. Mission
	2. Pembuatan Unit	2.a. Jumlah b. Way Point
	3. Perilaku	3.a. Karakter b. Target c. Aksi d. Prioritas Target
Action RPG	1. Musuh Penyerang	1.a. Range b. Life
	2. Musuh Pemanah	2.a. Range b. Ammo
	3. Musuh Bos	3.a. Range b. Life

Pada tabel 6 perbandingan pengujian algoritma *Fuzzy Logic* terlihat dalam penerapan pada game strategi bahwa *fuzzy logic*menggunakan parameter pengujian lebih banyakdibandingkan dengan pengujian pada *game Action RPG*untuk penerapan kecerdasan buatan.

4. KESIMPULAN

Hasil dari implementasi dan proses pengujian algoritma yangdilakukan, dapat diambil kesimpulan sebagai berikut:

- Penggabungan Algoritma A Star, Tilebase Collision dan Fuzzy Logic digunakan untuk mendapatkan parameter kecerdasan buatan.
- Penerapan Algoritma A Star dalam menentukan rute karakter pada *game* strategi dibutuhkan gerak *multi direction.*
- Algoritma *multi direction* adalah dari hasil pengembangan algoritma *Steering Behavior.*
- Algoritma *Tilebase Collision* dapat digunakan untuk mendeteksi tabrakan antara semua objek dan menjalankan algoritma *Fuzzy Logic* untuk mengantar karakter keluar dari *collision* dan kembali menuju target
- Algoritma *Fuzzy Logic*menggunakan parameter pengujian lebih banyak dibandingkan dengan pengujian pada *game Action RPG* untuk penerapan kecerdasan buatan.dan mengatur ruang lingkup tugas yang akan dilakukan sistem.

5. SARAN

Formulasi algoritma *Multi Direction* adalah hasil pengembangan algoritma *Steering Behavior,* dari formulasi tersebut diharapkan dapat diterapkan pada game jenis RTS atau RPG. Pada *game* strategi ini masih menggunakan desain orang lain sehingga perlu dikembangkan untuk hasil desain sendiri.

DAFTAR PUSTAKA

[1] Ratno Kustiawan,Kusrini, Hanif al Fatta.2013.*PerancanganFramework Real Time Strategy Games.* Yogyakarta:*Seminar Nasional Teknologi Informasi dan Komunikasi 2013* (SENTIKA 2013) ISSN: 2089-9815.

[2] Ben G. Weber, Michael Mateas,Arnav Jhala.2011.*Building Human-Level AI for Real-Time Strategy Games*. Association for the Advancement of Artificial Intelligence (www.aaai.org).

[3] Siswanto,(2010). *Kecerdasan Tiruan*, edisi 2, Yogyakarta:Graha Ilmu.

[4] Yonly Adrianus Benufinit, Moch. Hariadi, Supeno Mardi S. N (2015). Manuver Kelompok NPC Berbasis Boids.Yogyakarta: Seminar Nasional ke – 9: Rekayasa Teknologi Industri dan Informasi Sekolah Tinggi Teknologi Nasional (STTNAS)

[5] Dandy Akhmad Rahadiansyah (2015).*Kompleksitas Algoritma A* Pada Implementasi PassiveAI Untuk Game Mobile AI*. Makalah IF5110 Teori Komputasi – Sem. I Tahun 2015/2016

[6] GeethuElizebeth Mathew. (2015).*Direction Based Heuristic ForPathfinding In Video Games*. Elsevier.(http://creativecommons.org/licenses/by-nc-nd/4.0/).

[7] Patah Herwanto, Trisna Sonjaya (2016).*Rancang Bangun Game 3d "Ena Burena" Dengan Algoritma A* Dan Collision Detection Menggunakan Unity 3d Berbasis Desktop Dan Android*.Jurnal Informasi Volume VIII No.1 / Februari / 2016

[8] Bevilacqua, Fernando., "*Understanding Steering Behaviors: Seek*". <URL: http://gamedev.tutsplus.com/tutorials/implementation/understanding-steering-behaviors-seek/ >. 2013.

[9] Kristo Radion Purba, Rini Nur Hasanah dan M. Azis Muslim (2013).*Implementasi Logika Fuzzy Untuk Mengatur Perilaku Musuh dalam Game Bertipe Action-RPG*. Jurnal EECCIS Vol. 7, No. 1, Juni 2013.

[10] Shaout, Adnan, B. King dan L. Reisner.(2006). "Real Time Game Design of Pacman using Fuzzy Logic". The International Arab Journal of Information Technology, 2006, vol. 3 no. 4. p.320

[11] Sugiyono,(2016). *Metode Penelitian Pendidikan*. edisi 23, Bandung: Alfabeta