OPTIMISASI TOTAL TEMPUH *NPC* PADA *RTS GAME* MENGGUNAKAN *HARMONY SEARCH ALGORITHM*

Herti Miawarni[1*]
[1]Program Studi Teknik Elektro, Fakultas Teknik, Universitas Bhayangkara Surabaya
Jl. A. Yani 114, Surabaya 60231, Telp. 031-5602
[*]Email: herti_mia@ubhara.ac.id

Abstrak

Pada sebuah fighting game khususnya Real Time Strategy (RTS), perilaku menyerang yang dimiliki oleh autonomous NPC (Non Player Character) dituntut untuk lebih realistik dan natural. Perilaku menyerang NPC ditentukan berdasarkan penugasan pada tiap individu NPC (Assignment List), sehingga tiap NPC akan memiliki target serang masing-masing. Assignment List dapat disusun berdasarkan hasil analisa parameter jarak, sehingga dibutuhkan metode optimisasi untuk menganalisa parameter jarak, agar rata-rata total tempuh tiap NPC menjadi pendek (optimal). Pada penelitian ini, digunakan algoritma optimisasi Harmony Search Algorithm (HSA), yaitu merupakan salah satu metode optimisasi yang dapat digunakan dalam proses pencarian jalur terpendek (Shortest Path Finding). Dari uji coba yang dilakukan, perilaku pergerakan menyerang paling realistik diperoleh pada saat kedua kubu NPC saling menggunakan HSA, dengan waktu komputasi sebesar 0.69 detik, dan rata-rata total tempuh mencapai nilai terpendek sebesar 384.12.

***Kata kunci** : assignment list, Harmony Search Algorithm (HSA), Non Player Character (NPC), parameter jarak, rata-rata total tempuh.*

1. PENDAHULUAN

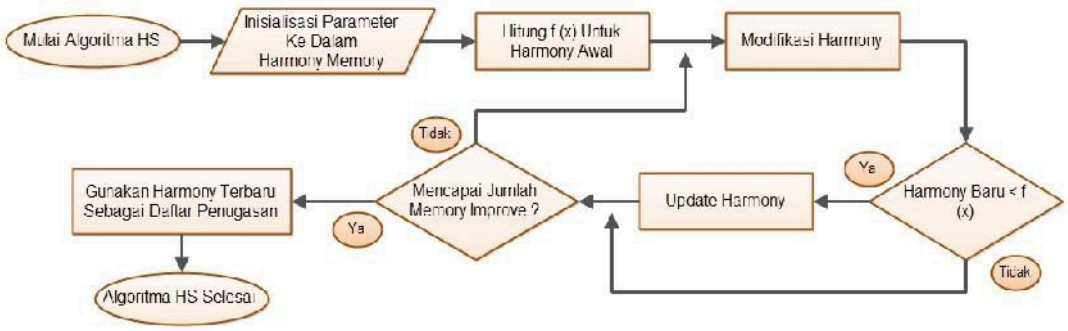
Harmony Search Algorithm (HSA) adalah suatu algoritma *metaheuristic* yang dinspirasi dari improvisasi permainan musik sejumlah musisi jazz, untuk mencari harmoni yang lebih baik. *Harmony Search* diperkenalkan oleh Zong Woo Geem pada tahun 2001. Pencarian harmoni pada proses improvisasi musik bertujuan untuk mendapatkan keadaan terbaik berdasarkan perkiraan estetika. Analogi impovisasi harmoni di dalam musik sama seperti mencari solusi yang optimal pada suatu masalah optimisasi.

Fighting game jenis *Real Time Strategy (RTS)* merupakan satu dari sekian banyak *game* yang melibatkan sekumpulan *NPC (Non Player Character)* dan bersifat *autonomy. NPC* dapat disebut juga dengan *agent.* Menurut Guralnik (1983), *agent* telah didefinisikan sebagai *A person or thing that acts or is capable of acting or is empowered to act, for another.* Bila dikaitkan dalam *game*, dapat diartikan sebagai karakter pada *game* yang memiliki kecerdasan perilaku dan kemampuan untuk melakukan suatu tugas atau pekerjaan demi kepentingan pengguna *game (user).* Kecerdasan *NPC* diperoleh dari algoritma *Artificial Intelligent (AI)* yang ada di dalam *game.* Sedangkan *autonomy* menurut Fanani (2012) adalah sifat *agent* yang dapat melakukan tugas mandiri tanpa dipengaruhi *user*, ataupun *agent* lainnya.

Beberapa Penelitian tentang perilaku *NPC* yang telah dilakukan, diantaranya adalah optimisasi strategi menyerang (Fanani, 2012; Noorman, 2012). Adapun keterkaitan penelitian ini dengan penelitian sebelumnya adalah pada permasalahan daftar penugasan *(Assignment List)* yang merupakan bagian dari strategi dalam *game.* Pada penelitian ini, dilakukan uji coba algoritma optimisasi *HSA* untuk menghasilkan daftar penugasan bagi tiap-tiap *NPC.* Sehingga pergerakan dan perilaku menyerang tiap *NPC* akan lebih terkoordinir dan optimal terutama dari segi total tempuh. Selain itu, pada penelitian ini juga dilakukan uji coba untuk menganalisa pengaruh penggunaan algoritma terhadap *NPC* lawan. Serta membandingkan bila *NPC* bergerak tanpa proses optimisasi.

2. METODOLOGI

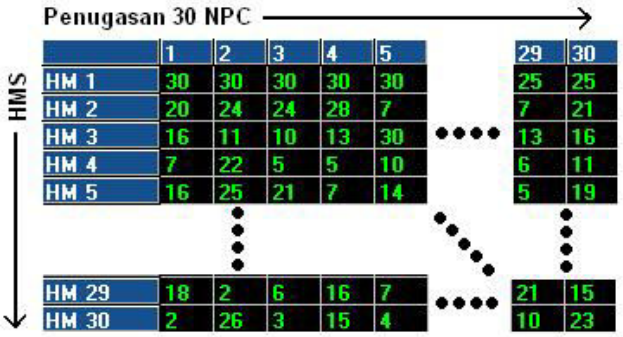
Pada penelitian ini digunakan *Harmony Search Algorithm (HSA)* dalam penyelesaian masalah optimisasi penugasan penyerangan NPC. Berikut gambaran bagaimana *HSA* digunakan dalam proses optimisasi.



Gambar 1. Diagram Alir Proses *Harmony Search Algorithm (HSA).*

2.1. Inisialisasi Parameter *HSA*

Langkah awal dalam proses optimisasi menggunakan *Harmony Search Algorithm* adalah inisialisasi parameter, dimana pembobotan parameter diaplikasikan langsung pada *Harmony Memory (HM). Harmony Memory* dapat diartikan sebagai matrix 30 x 30 [Penugasan x Harmony *Memory Size (HMS)*]. Dalam penelitian ini, penugasan dialokasikan untuk 30 *NPC player* dan 30 *NPC enemy*. Sedangkan *Harmony Memory Size (HMS)* adalah nilai variabel antara 10 hingga 30.



Gambar 2. *Harmony Memory (HM)* pada *(HSA).*

2.2. Harmony Awal Untuk mencari Penugasan Awal *(First Assignment List)*

Pada proses ini, penugasan diambil dari solusi yang terbaik diantara sekian banyak *Harmony Memory* yang ada, dengan aturan *objective function* sebagai berikut :

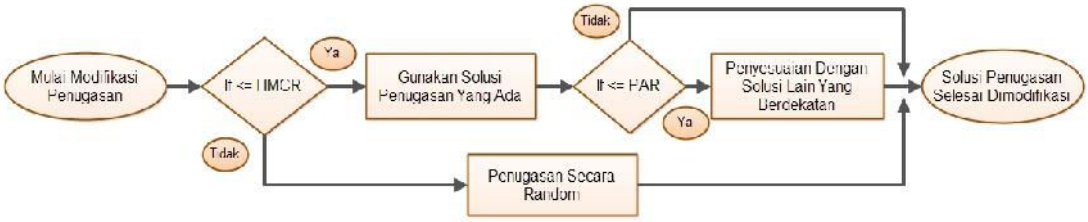
$$f(x)^{New} = min\, f(x)^{HMS} \qquad (1)$$

Dari aturan tersebut maka, solusi terbaik adalah solusi dengan jarak minimum dari *Harmony Memory*. Jarak minimum adalah pembobotan dari parameter jarak tiap NPC.

Gambar 3. Penugasan awal diambil dari *best solution* pada *Harmony Memory (HM).*

2.3. Modifikasi Harmony Untuk Membentuk Penugasan Baru *(New Assignment List)*

Proses pembentukan penugasan baru dilakukan dengan cara mengoptimisasi penugasan *NPC* menggunakan *HSA,* kemudian dilanjutkan dengan modifikasi Harmony. Pada penelitian ini, modifikasi Harmony dimaksudkan untuk memodifikasi penugasan dalam mencari solusi yang lebih baik. Adapun atribut algoritma *HSA* yang berperan penting pada proses ini adalah *HMCR* dan *PAR,* seperti diagram alir pada Gambar 4.



Gambar 4. Diagram Alir Proses Modifikasi Penugasan Pada *HSA*.

Pada Gambar 4, proses modifikasi diawali dengan seleksi *HMCR. HMCR* adalah variabel dengan nilai *0<=HMCR<=1.* Kemudian sebuah variabel bilangan *(n)* di-*generate* secara random dengan nilai *0<=n<=1.* Jika *n>HMCR,* maka *Harmony Memory* akan diisi dengan bilangan random. Dan bila sebaliknya, maka solusi terbaik pada *Harmony Memory* akan diambil untuk diproses dengan seleksi *PAR.*

Pitch Adjustment Rate (PAR) adalah variabel dengan nilai *0<=PAR<=1.* Kemudian sebuah variabel bilangan *(m)* di-*generate* secara random dengan nilai *0<=m<=1.* Jika *m>PAR,* maka solusi penugasan akan dijadikan solusi baru. Dan bila sebaliknya maka, solusi tersebut akan dimodifikasi dengan solusi nilai tetangga yang berdekatan. Dalam penelitian ini, yang dimaksud solusi nilai tetangga adalah solusi yang paling mendekati terbaik.

2.4. Update Memory

Pada proses ini, solusi hasil modifikasi Harmony dinyatakan sebagai solusi baru *X new*. Kemudian solusi yang sudah ada pada *Harmony Memory* dinyatakan sebagai solusi lama *X old* , dan solusi terburuk pada *Harmony Memory* dinyatakan sebagai *X worst*, seperti yang terlihat pada Gambar 5.

Gambar 5. Proses *Update Memory* Pada *HSA*.

Pada Gambar 5, jika *X new* lebih baik dari *X worst (HM 5),* maka semua solusi penugasan yang ada pada *X worst (HM 5)* akan diganti dengan solusi *X new* .

2.5. Proses Pemberhentian Optimisasi

Proses optimisasi untuk memperoleh solusi penugasan akan berhenti jika sudah mencapai batas *Memory Improve.* Dalam hal ini, solusi penugasan diambil dari *Harmony Memory* yang terbaik. Namun proses optimisasi masih akan tetap berlanjut, karena perubahan kondisi *Real Time*. Perubahan kondisi akan berakibat pada perubahan *Harmony Memory.*

3. HASIL DAN PEMBAHASAN

Pada penelitian ini, dilakukan uji coba kinerja *HSA* sebagai algoritma penyelesaian masalah penugasan *NPC,* yaitu dengan membandingkan 3 mode penggunaan algoritma, antara lain :

1) *NPC Player (Non Optimisasi) vs NPC Enemy (Non Optimisasi)*
2) *NPC Player (HSA) vs NPC Enemy (Non Optimisasi)*
3) *NPC Player (HSA) vs NPC Enemy (HSA)*

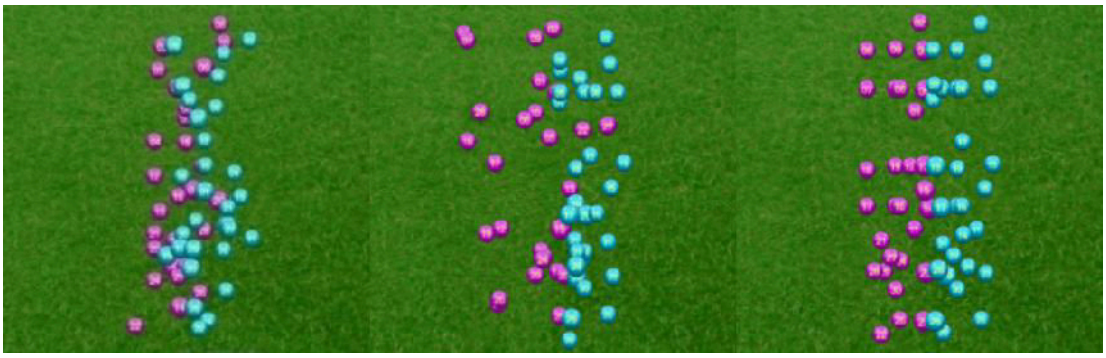
Dengan demikian, berapa besar peran *HSA* sebagai algoritma optimisasi dapat dianalisa secara detail. Sedangkan Non Optimisasi adalah suatu kondisi dimana setiap *NPC* bebas memilih lawan secara acak *(Random Asignment)* tanpa ada proses optimisasi. Adapun *setting* beberapa atribut pada algoritma *HSA* adalah sebagai berikut :

1) *Memory Improvisation* = 10
2) *Harmony Memory Size (HMS)* = 30
3) *Harmony Memory Considration Rate (HMCR)* = 0.6
4) *Pitch Adjustment Rate (PAR)* = 0.5

3.1. Pengaruh Mode Kecerdasan Pada Penugasan *NPC* Terhadap Perilaku Pertempuran

Perilaku *NPC* dalam pertempuran sangat dipengaruhi oleh penugasannya. Penugasan tidak hanya berpengaruh terhadap siapa target yang harus dilawan oleh NPC, tetapi penugasan juga berpengaruh pada perilaku pertempuran.

Pada penelitian ini, 30 unit *NPC player* disimulasikan dengan 30 buah bola berwarna biru. Sedangkan 30 unit *NPC enemy* disimulasikan dengan 30 bola berwarna ungu. Selanjutnya, kedua tim *NPC* akan bergerak menyerang secara otomatis berdasarkan penugasan hasil dari algoritma yang digunakan, seperti yang terlihat pada Gambar 6.



(a) **(b)** **(c)**

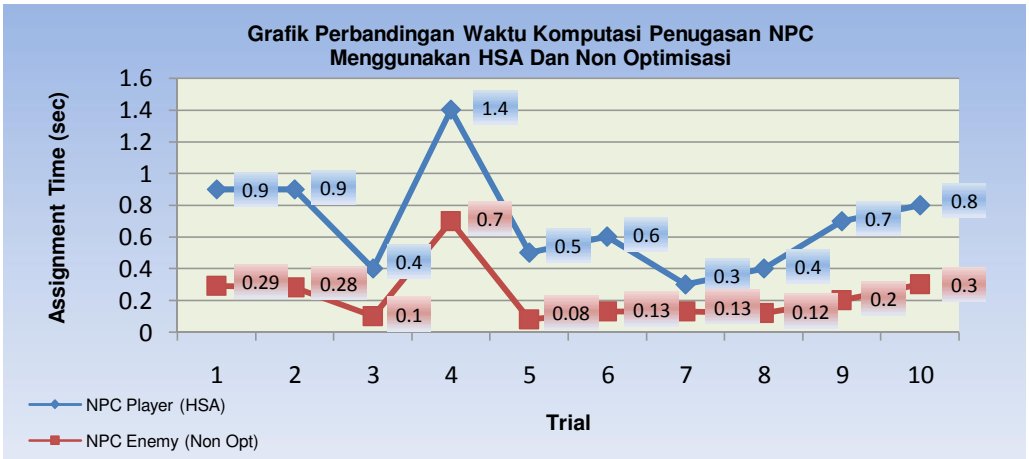
Gambar 6. *Capture* Simulasi Pertempuran *NPC*.
(a) *NPC player (Non Optimisasi) vs NPC enemy (Non Optimisasi).*
(b) *NPC player (HSA) vs NPC enemy (Non Optimisasi).*
(c) *NPC player (HSA) vs NPC enemy (HSA).*

Dari Gambar 6 (b) dan (c), terlihat jika penugasan salah satu tim dioptimisasi menggunakan *HSA*, maka perilaku pertempuran tim tersebut akan mengikuti pengaruh optimisasi. Di sisi lain, lawan akan beradaptasi dengan perilaku tersebut. Lain halnya jika pada suatu pertempuran, tidak satupun tim yang menggunakan optimisasi penugasan, maka pertempuran menjadi tidak terkoordinir seperti yang terlihat pada Gambar 6 (a).

3.2. Perbandingan Waktu Komputasi

Waktu komputasi adalah waktu yang dibutuhkan untuk menghasilkan daftar penugasan pada tiap *NPC* baik *NPC player* maupun *enemy*. Pada uji coba ini, untuk membandingkan waktu komputasi, cukup dengan 1 mode variasi algoritma, yaitu penugasan pada *NPC player* menggunakan *HSA,* sedangkan penugasan pada *NPC enemy* tidak menggunakan algoritma optimisasi (Non Optimisasi).

Adapun hasil uji coba, dapat divisualisasikan ke dalam sebuah grafik, seperti yang terlihat pada Gambar 7.



Gambar 7. Grafik Perbandingan Waktu Komputasi NPC Menggunakan *HSA* Dan Non Optimisasi.

Dari Gambar 7, dapat diambil suatu analisa bahwa, waktu komputasi penugasan *HSA* pada *NPC player* lebih lama dibandingkan dengan waktu komputasi Non Optimisasi pada *NPC enemy.* Hal ini disebabkan karena, proses optimisasi *HSA* membutuhkan waktu sesuai jumlah *Memory Improvisation.* Dari 10 kali trial, maka dapat dirata-rata waktu komputasi *HSA* sebesar 0.69 detik, sedangkan waktu komputasi Non Optimisasi sebesar 0,23 detik.

3.3. Pengaruh Algoritma Terhadap Total Tempuh

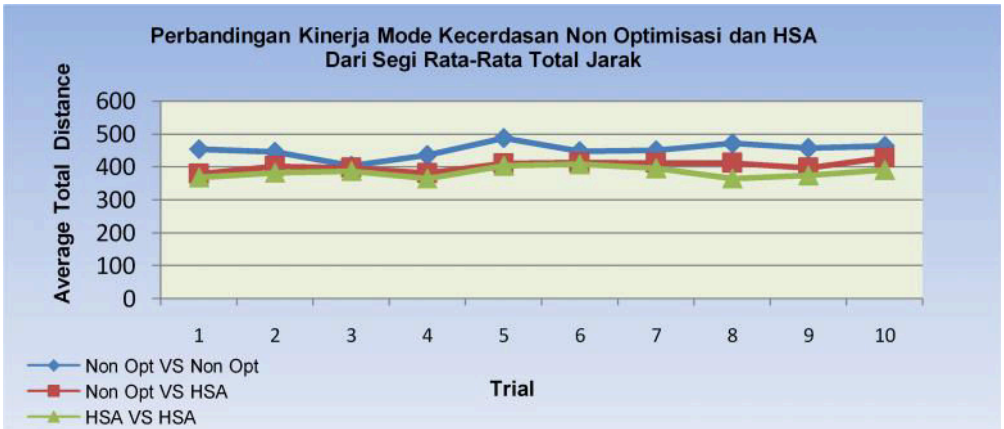
Uji coba bertujuan untuk menguji kinerja ketiga mode penggunaan algoritma dan pengaruhnya terhadap rata-rata total tempuh *NPC.* Jika mengacu pada uji coba sebelumnya (sub bab 3.1), maka rata-rata total tempuh *NPC* tidak hanya dipengaruhi oleh kecerdasan *NPC player,* namun juga dipengaruhi oleh kecerdasan *NPC enemy.* Maka dari itu, uji coba dilakukan cukup dengan mengamati rata-rata total tempuh *(Average Total Distance)* pada *NPC player* saja.

Adapun hasil uji coba yang telah dilakukan dari ketiga mode penggunaan algoritma, dapat diuraikan seperti pada Tabel 1.

Tabel 1. Perbandingan Total Tempuh Penggunaan Tiga Mode Algoritma.

Trial	Non Optimisasi vs Non Optimisasi	HSA vs Non Optimisasi	HSA vs HSA
1	454.6	379.6	368.53
2	445.33	402	382.5
3	403.06	396.42	387.22
4	435.4	380.15	365.33
5.	488.23	410.6	402.8
6	447.15	412.72	409
7	451.08	412.36	395.15
8	472.3	412.22	364.33
9	456.61	397.43	375.12
10	463.1	427.71	391.26
Rata-Rata *(Average)*	451.68	403.12	384.12

Dari data percobaan pada Tabel 1, bila divisualisasikan ke dalam sebuah grafik, seperti yang terlihat pada Gambar 8.



Gambar 8. Grafik Perbandingan Total Tempuh Ketiga Mode Penggunaan Algoritma

Grafik pada Gambar 8 dapat disimpulkan bahwa, dari 10 kali trial, penggunaan mode algoritma HSA vs HSA memiliki rata-rata total tempuh terpendek, yaitu pada angka 384.12. Sedangkan HSA vs Non Optimisasi memiliki rata-rata total tempuh 403.12, sedangkan mode Non Optimisasi vs Non Optimisasi memiliki rata-rata terjauh yaitu 451.68.

4. KESIMPULAN

Berdasarkan hasil analisa dan uji coba pada penelitian ini, permasalahan penugasan *NPC* yang diselesaikan menggunakan metode *Harmony Search Algorithm (HSA),* diperoleh beberapa kesimpulan sebagai berikut:

(1) Optimisasi permasalahan penugasan pada *NPC player*, selain berpengaruh terhadap penugasan, juga berpengaruh terhadap perilaku pertempuran. *NPC player* cenderung memprioritaskan penyerangan terhadap musuh paling depan, yang merupakan jarak terpendek. Disisi lain, *NPC enemy* beradaptasi dengan perilaku serang *NPC player,* begitu pula sebaliknya. Pergerakan *NPC* yang paling realistik adalah ketika *NPC player* dan *enemy* bergerak secara teratur dan terkoordinir. Hal ini terlihat pada mode dimana kedua tim *NPC* saling menggunakan metode optimisasi *HSA*.

(2) Waktu komputasi pada mode *HSA* lebih lama dibanding Non Optimisasi. Dari 10 kali trial, algoritma *HSA* memiliki rata-rata waktu komputasi penugasan 0.69 detik dan Non Optimisasi memiliki waktu komputasi lebih singkat yaitu 0.23 detik.

(3) Kinerja algoritma *HSA* dalam meminimumkan rata-rata total tempuh sangat baik. Hal ini ditunjukkan dari hasil rata-rata total tempuh terpendek sebesar 384.12 pada saat menggunakan mode HSA vs HSA. Sedangkan mode HSA vs Non Optimisasi mencapai nilai rata-rata total tempuh sebesar 403.12. Dan rata-rata total tempuh terjauh didapat dari mode Non Optimisasi vs Non Optimisasi yaitu sebesar 451.68.

DAFTAR PUSTAKA

Fanani Nurul Zainal. (2012). "*Optimisasi Strategi Menyerang Kelompok NPC Pada Game Pertarungan Jarak Dekat Menggunakan Algoritma Harmony Search"*. Tesis. Teknik Elektro, Fakultas Teknologi Industri. Institut Teknologi Sepuluh Nopember, Surabaya.

Geem ZW, Kim JH and Loganathan GV (2001) "*A new heuristic optimization algorithm*": Harmony search. Simulation, 76:60-68.

Guralnik David B. (1983). "*Webster's New World Dictionary*". Student Edition., Pearson Prentice Hall Press, New Jersey.

Noorman Rinanto. (2012). *Optimisasi Strategi Menyerang NPC Group Jarak Dekat Menggunakan Simulated Annealing*. Tesis. Teknik Elektro, Fakultas Teknologi Industri. Institut Teknologi Sepuluh Nopember, Surabaya.