TESIS - TE142599

PERSEBARAN MULTI NPC MENGGUNAKAN ALGORITMA *BEE COLONY* UNTUK GAME 3D *SPACE SHOOTER*

JUNIARDI NUR FADILA
2214205003

DOSEN PEMBIMBING
Dr. EKO MULYANTO YUNIARNO, ST, MT.
Dr. SUPENO MARDI SUSIKI N, ST, MT.

PROGRAM MAGISTER
BIDANG KEAHLIAN JARINGAN CERDAS MULTIMEDIA
KONSENTRASI GAME TEKNOLOGI
JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2016

TESIS - TE142599

MULTI NPC SWARM USING BEE COLONY ALGORITM FOR 3D SPACE SHOOTER GAME

JUNIARDI NUR FADILA
2214205003

SUPERVISOR
Dr. EKO MULYANTO YUNIARNO, ST, MT.
Dr. SUPENO MARDI SUSIKI N, ST, MT.

MASTER PROGRAM
MULTIMEDIA INTELLIGENT NETWORK
PROGRAM GAME TECHNOLOGY
DEPARTMENT OF ELECTRICAL ENGINEERING
FACULTY OF INDUSTRIAL TECHNOLOGY
SEPULUH NOPEMBER INSTITUTE OF TECHNOLOGY
SURABAYA
2016

Tesis disusun untuk memenuhi salah satu syarat memperoleh gelar
Magister Teknik (M.T)
di
Institut Teknologi Sepuluh Nopember

Oleh:
Juniardi Nur Fadila
NRP. 2214205003

Tanggal Ujian : 20 Juni 2016
Periode Wisuda : September 2016

Disetujui oleh :

1. **Dr. Eko Mulyanto Yuniarno, S.T., M.T.** **(Pembimbing 1)**
   NIP : 196806011995121009
2. **Dr. Supeno Mardi Susiki N., S.T., M.T.** **(Pembimbing II)**
   NIP. 197003131995121001
3. **Dr. Surya Sumpeno, S.T., M.Sc.** **(Penguji)**
   NIP. 196906131997021003
4. **Dr. Ir. Yoyon Kusnendar Suprapto, M.Sc.** **(Penguji)**
   NIP. 195409251978031001

Direktorat Program Pasca Sarjana,

KEMENTERIAN RISET, TEKNOLOGI, DAN PENDIDIKAN TINGGI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
PROGRAM PASCASARJANA

Prof. Ir. Djauhar Manfaat, M.Sc., Ph.D.
NIP. 196012021987011001

Halaman ini sengaja dikosongkan

Persebaran Multi NPC Menggunakan Algoritma *Bee Colony* untuk Game 3D *Space Shooter*

Nama Mahasiswa	: Juniardi Nur Fadila
NRP	: 2214205003
Dosen Pembimbing 1	: Dr. Eko Mulyanto Yuniarno, ST., MT.
Dosen Pembimbing 2	: Dr. Supeno Mardi Susiki Nugroho, ST., MT.

ABSTRAK

NPC dalam sebuah game, merupakan sebuah objek yang banyak digunakan dalam penelitian baru-baru ini. Hal ini disebabkan karena NPC dalam game menyediakan peluang penelitian yang sangat luas. Sebagai salah satu contohnya yaitu pada bidang AI. Bidang AI sendiri dapat dibagi menjadi beberapa bidang seperti *path finding, decision making, learning logic,* dll. Penelitian ini dilakukan untuk mengembangkan sebuah metode yang dapat mendukung fitur AI pada NPC dengan cara menambahkan perilaku hewan pada pergerakan NPC tersebut. *Bee Colony* dipilih sebagai metode utama dalam penelitian ini karena algoritma tersebut telah diteliti pada beberapa penelitian sebelumnya dan terbukti efektif. Tetapi metode tersebut hanya di uji cobakan pada target yang statis. Selain hanya di uji cobakan pada target yang statis, tidak diketahui juga apa yang terjadi jika metode tersebut diaplikasikan pada NPC yang memiliki banyak target dan target tersebut bergerak bebas. Dalam penelitian ini akan diteliti bagaimana algoritma *bee colony* tersebut dapat mengorganisasi gerombolan NPC dalam permainan untuk bergerak menuju target yang tersebar dalam ruang lingkup game dan pergerakan NPC mengaplikasikan sistem kerja koloni lebah.

Kata Kunci: NPC, *Bee Colony*, *Artificial Intelligent, Swarming, Game.*

Halaman ini sengaja dikosongkan

Multi NPC Swarm Using *Bee Colony* Algoritm for 3D Space Shooter Game

Name	: Juniardi Nur Fadila
NRP	: 2214205003
Supervisor	: Dr. Eko Mulyanto Yuniarno, ST., MT.
Co-Supervisor :	: Dr. Supeno Mardi Susiki Nugroho, ST., MT.

ABSTRACT

NPC's in a game, is an object that to be trending topic in a recent study. This is because NPC's in the game provides a lot of research opportunities. For an example is research in NPC's Artificial Intelligent (AI). AI itself can be divided into several areas such as path finding, decision making, learning logic, etc. This research was conducted to develop a method that can support the AI feature by adding animal behavior at the NPC movement. Bee Colony has been selected as the main method in this study because the bee colony has been researched in several previous studies and has proven its effectiveness. But that method was only tested on a stationary target. However, the method only tested on a stationary target. Not yet known what would happen if such methods if applied in the NPC that has a wide range of targets and the target can move freely. In this study, will be researched how the bee colony method can organize a collection of NPC in a game to move to the targets that spread in the space of game and the NPC movement should be random like colony of bee.

***Keywords**: NPC, Bee Colony, Artificial Intelligent, Swarming, Game*

Halaman ini sengaja dikosongkan

KATA PENGANTAR

Segala puji bagi Allah SWT yang telah melimpahkan rahmat serta karuniaNya kepada penulis sehingga bisa menyelesaikan tesis dengan judul "Persebaran Multi NPC Menggunakan Algoritma *Bee Colony* untuk Game 3D *Space Shooter*" dengan baik.

Shalawat serta salam semoga tercurah kepada Nabi Agung Muhammad SAW yang telah membimbing umatnya dari gelapnya kekufuran menuju cahaya Islam yang terang benderang.

Penulis menyadari keterbatasan pengetahuan yang penulis miliki, karena itu tanpa keterlibatan dan sumbangsih dari berbagai pihak, sulit bagi penulis untuk menyelesaikan thesis ini. Maka dari itu dengan segenap kerendahan hati patutlah penulis ucapkan terima kasih kepada:

1. Ayah dan Ibu yang selalu memotivasi dan selalu mendoaakan yang terbaik dalam pengerjaan thesis ini.
2. Dr. Eko Mulyanto Yuniarno, S.T., M.T dan Dr. Supeno Mardi Susiki Nugroho, ST, MT selaku dosen pembimbing yang telah mengarahkan, memberi koreksi, dan motivasi dalam tesis ini.
3. Dr. Eko Mulyanto Yuniarno, S.T., M.T, selaku koordinator bidang keahlian Jaringan Cerdas Multimedia Program Studi Teknik Elektro.
4. Bapak Dewan penguji selaku dosen penguji yang telah memberikan saran dan kritik dalam tesis ini.
5. Bapak-bapak dosen pengajar di Program Studi Teknik Elektro, bidang keahlian Jaringan Cerdas Multimedia.
6. Kepada Calon istriku Izza Mabruroh yang selalu mendukung dan menyemangati dari kejauhan.
7. Kepada teman-teman seperjuangan S2 Jaringan Cerdas Multimedia ITS yang senior maupun yang satu angkatan.

8. Semua pihak yang tidak mungkin penulis sebutkan satu-persatu, atas segala yang telah diberikan kepada penulis dan dapat menjadi pelajaran.

Sebagai penutup, penulis menyadari dalam tesis ini masih banyak kekurangan dan jauh dari sempurna. Semoga apa yang menjadi kekurangan bisa disempurnakan oleh peneliti selanjutnya. Apa yang menjadi harapan penulis, semoga karya ini bermanfaat bagi kita semua. Amin.

Surabaya, 1 Juni 2016

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

Halaman ini sengaja dikosongkan

DAFTAR ISTILAH

φ	: Nilai acak
Atk	: Daya serang agen ke-i
Agen	: Kata ganti lebah atau NPC dalam metode
d	: Jarak
Def_i	: Daya tahan agen ke-i
$f_{collision}$	: Fungsi tumbukan
fit_{dist}	: Nilai fitness jarak
fit_{obs}	: Nilai fitness tabrakan
$fit_{strength}$	: Nilai fitness kekuatan
Hp	: Nyawa dari agen ke-i
Kr_i	: Nilai seberapa cepat agen ke-i membunuh musuh
MC	: *Maximum cycle* atau jumlah iterasi
NPC	: Non-Playable Character
n	: Jumlah nektar yang tersedia
P_i	: Probabilitas ke-i
r	: Radius tabrakan
$rand[0,1]$	: Nilai acak dari 0 hingga 1
TS	: *The Solution* atau solusi yang tersedia
x	: Posisi pada sumbu horizontal
y	: Posisi pada sumbu vertical
z	: Posisi pada sumbu depan

Halaman ini sengaja dikosongkan

BAB I

PENDAHULUAN

1.1 Latar Belakang

Teknologi pengembangan agen otonom yang diterapkan sebagai *non player character* (NPC) berperilaku menyerang pada game peperangan, makin banyak diteliti. Beberapa penelitian imenerapkan kecerdasan buatan pada NPC agar dapat merespon posisi serta keadaan player.

Mardi dkk., memodelkan kerumunan pada keadaan bencana berbasis boids, yang merupakan gabungan system partikel pada entitas (BOIDS). Dengan kemampuan yang sama antar boid, maka setiap boid akan menghindari halangan yang ada disekitarnya. Hasil yang diperoleh menunjukkan 60 boids mampu menghindari tabrakan antar boid atau halangan hingga 90%. Saat jumlah boids hanya 20, terjadi peningkatan hingga 100%. Sementara dari sisi waktu, 60 boids membutuhkan waktu lebih pendek untuk mencapai 80%, namun dengan waktu lebih lama, dapat mencapai 90%.(Mardi dkk., 2011)

Pada penelitian berikutnya, Nugroho dkk, menerapkan fuzzy logic dan Hierarchiecal Finite State Machine (HFSM) untuk non player character (NPC) agar dapat meniru strategi manusia untuk peperangan. Hasil yang dicapai antara lain, HFSM berhasil memodelkan perilaku manuver tiap NPC, penerapan fuzzy untuk perilaku NPC berhasil mengungguli NPC tanpa fuzzy hingga 80%.(Nugroho dkk., 2011)

Selanjutnya, Nugroho dkk, mengajukan metode fuzzy coordinator bagi pemimpin tim untuk mengkoordinasikan perilaku tim dengan memantau dan menganalisis kondisi kesehatan tiap anggota dalam tim. Sehingga saat menyerang bersama-sama, NPC selalu dalam kondisi bugar. Hasil eksperimen menunjukkan

bahwa pemimpin tim mampu menjaga kondisi kesehatan anggota tim selalu berada antara 75% hingga 50%.

Penelitian yang dilakukan Asmiatun, dkk, merupakan klasifikasi perilaku penyerangan NPC menggunakan algoritma Bayesian, untuk peperangan jarak dekat, kemudian hasil klasifikasi tersebut digunakan untuk menentukan perilaku penyerangan. Hasil penelitian awal menunjukkan bahwa NPC menjadi tidak mudah dikalahkan.(Asmiatun dkk., 2013)

Menyempurnakan penelitian tentang menyerang dalam formasi berkerumun, Jatiningsih dkk, meneliti tentang pembelajaran yang dilakukan NPC agar dapat menyerang musuh dalam formasi kerumunan. Dengan menggunakan empat fungsi benchmark yang umum untuk mencari jumlah siklus yang dibutuhkan lebah untuk menuju konvergen, ukuran populasi = 125, jangkauan parameter mulai dari -100 hingga +100, jumlah siklus dibatasi 50 kali, batas pengabaian = 10 dan posisi sumber makanan pada (0,0). Hasil yang diperoleh menunjukkan bahwa lebah menuju konvergen pada siklus 0-25(Jatiningsih dkk., 2014).

Namun penelitian yang telah dilakukan memiliki kelemahan, antara lain: hanya meneliti kemampuan menghindar dari tabrakan antar boid saat simulasi keadaan bahaya, memodelkan perilaku NPC dalam menyerang menggunakan fuzzy berbasis HFSM untuk melawan NPC tanpa fuzzy, menerapkan fuzzy coordinator untuk menjaga kondisi kesehatan anggota tim saat melakukan penyerangan, berdasarkan klasifikasi kondisi NPC, dapat ditentukan perilaku penyerangan, tetapi NPC merupakan agen tunggal dan menerapkan strategi penyerangan bergerombol dengan siklus terpendek berdasarkan metode *bee colony*, pada satu musuh yang diam.

Game simulator merupakan sebuah permainan yang memberikan sensasi nyata pada penggunanya sehingga seolah-olah sang pemain berada pada situasi game yang dia mainkan. Agar pemain dapat merasakan sensasi tersebut, game harus didesain senyata mungkin dari perilaku AI, hukum fisika, hingga desain interface

untuk pemainnya. Jika kita ambil sebuah contoh permainan simulasi berupa simulasi peperangan pesawat, maka dalam permainan tersebut pasti akan terdapat NPC yang bergerak secara otomatis. NPC harus dapat bergerak layaknya seorang player yang bermain secara independen, sehingga dalam NPC tersebut harus diberikan sebuah kecerdasan atau AI. Dalam permainan simulasi peperangan tersebut, kecerdasan yang diberikan kepada NPC merupakan AI yang dapat mengkoordinir setiap NPC untuk merespon keberadaan musuh dan bergerak sesuai dengan kondisi yang optimal dilihat dari segala aspek yang ada. NPC tersebut dapat juga di inisialisasikan sebagai sebuah agen dimana agen yang otomatis menurut The Maes Agent (Maes 1995), "Agen otomatis adalah sebuah system komputasi yang berjalan pada sebuah *environment* yang kompleks dan dinamis, yang dapat berpikir dan bertindak secara otomatis, dan dengan demikian dapat mewujudkan tujuan atau tigas yang telah dirancang". Selain itu, pengertian lain dari agen yang otomatis adalah sebuah sistem yang terletak dan merupakan bagian dari sebuah *environtment* yang menyatu dengan *environtmet* tersebut dan bertindak di atasnya, dari waktu ke waktu, dalam mengejar tujuannya sendiri sehingga dapat mempengaruhi apa yang terjadi selanjutnya (Franklin dkk., 1997).

Agar lebih menantang, NPC harus membuat pemain kesulitan dalam menghadapi dirinya. Salah satu cara untuk membuat pemain kesulitan adalah dengan membuat pergerakan yang acak saat NPC tersebut diserang atau menyerang. Hal tersebut juga berlaku pada saat NPC berjumlah lebih dari satu. Pergerakan mereka akan lebih baik jika acak akan tetapi masih terkoordinasi. Saat mereka bergerak ada kemungkinan jika mereka akan bertabrakan satu sama lain. Permasalahan ini harus dihindari karena dalam dunia nyata pesawat tidak akan dengan sengaja menabrakkan dirinya dengan pesawat lain. Sehingga dalam membuat kecerdasan NPC tersebut, kemungkinan bertabrakan tersebut harus diminimalkan. Permasalahan AI pada NPC ini adalah masalah yang dapat diteliti dengan beberapa metode pendekatan yang berbeda, misalnya: Neural Network, Particle Swarm Optimization, Algoritma Genetika, atau metode swarm yang lainnya. Hal tersebut dikarenakan permasalahan

pada NPC ini menyediakan model yang nyaman untuk evaluasi pendekatan baru dalam AI. Metode optimasi biasanya dapat diklasifikasikan sebagai berikut:

- Pencarian Langsung: metode pencarian langsung hanya menggunakan fungsi dan kendala obyektif nilai.
- Pencarian berbasis Gradient: teknik pencarian berbasis Gradient memerlukan informasi turunan dari fungsi dan kendala.

Karena informasi derivatif tidak digunakan, metode ini umumnya memerlukan sejumlah besar iterasi untuk konvergensi, tetapi di sisi lain berlaku untuk ruang masalah yang sangat luas. Swarm intelligence adalah salah satu metode dalam lingkup kecerdasan buatan. Metode ini berkaitan dengan model interaksi sosial antara makhluk sosial, lebah, terutama semut, dan burung-burung pada saat ini (Dasgupta dkk., 2008).

Berdasarkan review penelitian tersebut diatas, penulis mengusulkan penerapan metode *bee colony* sebagai metode dalam perilaku menyerang oleh NPC dalam sebuah tim yang padu dan solid untuk lawan yang berjumlah lebih dari dua serta selalu berpindah tempat.

1.2 Rumusan Masalah

Pergerakan sekelompok NPC yang terlihat monoton, membuat sekelompok NPC tersebut menjadi sangat mudah dikalahkan saat diserang oleh musuh. Selain itu saat terlibat pertempuran yang melibatkan banyak target, NPC biasanya memilih target yang pertama masuk kedalam jangkauan serangannya tanpa memperhatikan kemungkinan menangnya terhadap target tersebut. Permasalahan ini dapat membuat serangan NPC tersebut tidak optimal.

Pergerakan NPC yang mudah ditebak dan serangan yang tidak optimal dapat membuat sebuah permainan terlalu mudah untuk dimenangkan sehingga menjadi kurang menarik bagi usernya.

1.3 Tujuan

Diperoleh sekumpulan NPC yang memiliki perilaku kerja layaknya koloni lebah dan bergerak secara otonom menuju target yang paling baik menggunakan algoritma *Bee Colony*

1.4 Batasan Masalah

Batasan masalah yang digunakan dalam permasalahan ini adalah sebagai berikut:

- Metode digunakan pada permainan bergenre *flight combat simulator*.
- Metode akan berjalan ketika target masuk kedalam area pergerakan.
- Pergerakan NPC didasarkan pada sistem kerja koloni lebah.
- Kelompok NPC bergerak dalam bidang 3 dimensi
- Jumlah NPC dan Enemy dibatasi berjumlah 20 karakter.

Halaman ini sengaja dikosongkan

BAB II

KAJIAN PUSTAKA

Pada bab 2 ini akan dibahas dasar-dasar teori tentang metode yang digunakan dan teori-teori lain yang menjadi penunjan dalam penelitian ini.

2.1 Non-Playable Character(NPC)

Non Player Character atau disebut NPC adalah karakter dalam *computer game* yang tidak dikontrol oleh pemain. NPC dapat berupa agen cerdas yang dibuat dengan menanamkan *Artificial Intelegent* (AI) dan memiliki perilaku yang mirip seperti manusia seperti merubah strategi atau pola pikirnya dalam menanggapi tindakan lawan(Umarov dkk., 2012). NPC dapat juga dikatakan *belieavable* yang berarti memiliki kemiripan yang sangat tinggi terhadap karakter di *game* dengan yang ada di dunia nyata sehingga mampu membuat *game* lebih *immersive* (dimana pemain menjadi seakan terlibat dalam permainan dan menjadi fokus terhadap permainan tersebut dan kurang menyadari keadaan disekitarnya (Nugroho dkk., 2011). Kemiripan pada NPC dapat dilihat dari cara berjalan, cara berbicara dan tekstur karakter (Asmiatun dkk., 2013).

NPC sendiri dapat di bagi menjadi beberapa tipe.

- **Normal/Neutral NPC :** NPC yang digunakan hanya sebagai penambah latar dari game itu sediri. Biasanya NPC ini menjelaskan keadaan dari situasi atau kondisi pada latar scene pada game.
- **Dynamic NPC :** Berbeda dengan nomal NPC, NPC ini lebih berperan dalam game. NPC dengan tipe ini digunakan sebagai karakter tambahan dalam game. Biasanya dynamic NPC memiliki linear sendiri dan terkesan hidup dalam permainan.
- **Interacted NPC :** Interacted NPC hamper sama dengan Dynamic NPC. Perbedaannya adalah interacted NPC memiliki variable tertentu yang biasanya di nyatakan sebagai “Status”. Hal ini menyebabkan NPC ini dapat berubah-ubah sesuai dengan statusnya.

- **Dependent NPC :** Adalah NPC yang tidak bisa berinteraksi dengan player melainkan dengan Choices atau Tindakan Player.
- **Side NPC :** Side NPC merupakan NPC yang benar-benar tidak bisa berinteraksi. Biasanya digunakan hanya sebagai pelengkap latar saja.
- **Shopper :** Dalam game bergenre RPG atau MMORPG, biasanya terdapat NPC shopper. NPC ini bertindak sebagai pedagang barang yang dapat ditransaksikan ke player.

Merujuk pada definisi dari NPC maka pada penelitian ini di usulkan dua tipe NPC yaitu Dynamic NPC dan Side NPC.

2.2 Perilaku Agen Otonom

NPC atau *Non Playable Character* adalah karakter dalam game yang perilakunya tidak dikontrol oleh manusia/*player.* Perilaku NPC dibagi menjadi 3, yaitu strategis (*strategic*), taktik (*tactical*) dan reaktif (*reactive*). Perilaku stategi digunakan untuk mencapai tujuan jangka panjang, misalnya mengamankan wilayahnya. Setiap NPC selalu memiliki tujuan jangka panjang dan jangka pendek. Perilaku taktis digunakan untuk mencapai tujuan jangka pendek yang lebih spesifik lagi. Sedangkan perilaku reaktif adalah reaksi sederhana sesuai dengan persepsi audio visualnya pada saat itu, seperti melompat, berjalan, membidik atau menembak.

Simulasi pertempuran angkasa adalah sejenis permainan (mirip dengan simulator penerbangan atau software simulasi penerbangan amatir) yang digunakan untuk mensimulasikan pesawat militer dan operasi mereka. Dalam game *combat flight simulator,* NPC harus dapat bergerak selincah mungkin agar tidak terkena serangan oleh musuh. NPC tersebut harus dapat bergerak lincah tanpa menabrak pesawat lainnya maupun benda-benda yang dapat menjadi halangan pada saat terbang. Koordinasi pergerakan NPC ini dilakukan secara *realtime* karena pada game tipe ini, tidak ada waktu untuk memikirkan rute pengejaran maupun pelarian.

Ada 2 jenis NPC dalam game strategy, yaitu *strategic* NPC dan *unit* NPC. Strategic NPC digunakan untuk mengendalikan tentara lawan. Mereka harus bisa mengatur

strategi sama seperti yang player lakukan. Mereka juga mengumpulkan sumber daya yang ada dalam environment selama game dijalankan. Stategic NPC harus dapat melakukan melee attack sebagus player, tetapi di akhir game mereka harus membiarkan player yang menang.

Sedangkan Unit NPC adalah tentara atau karakter tunggal yang menjadi anggota pasukan player maupun strategic NPC. Artinya, unit NPC ini pasukan yang digerakkan oleh player maupun strategic NPC. Unit NPC harus memiliki kecerdasan agar dapat melaksanakan perintah player maupun perintah strategic NPC. Perintah itu dapat berupa perintah menyerang, mengumpulkan sumber daya atau membangun gedung. Unit NPC juga harus mampu merencanakan rute jalan dan mengikuti rute itu untuk mencapai target mereka dan untuk melaksanakan tugas mereka secara efektif sementara pada saat yang bersamaa mereka juga harus bereaksi terhadap perubahan lingkungan.

Agen otomon adalah sebuah sistem komputer yang hidup di dalam lingkungan buatan. Dia beraksi sesuai dengan agenda yang ditanamkan padanya sehingga dia bisa tahu harus melakukan apa jika menerima suatu rangsangan. (Franklin dkk., 1997).

Agen otonom harus mampu beraksi sesuai dengan sensor yang dia terima. Artinya agen tidak hanya berada dan menjadi bagian dari sebuah lingkungan buatan, tetapi menjadi pasangan dari lingkungan buatan itu. Arsitektur dan mekanisme agen harus terhubung dengan lingkungannya sehingga dapat merasakan setiap tujuan yang dirancang untuknya dan beraksi untuk memenuhi tujuan itu. (Maturana 1975, Maturana dkk., 1980, Varela 1991).

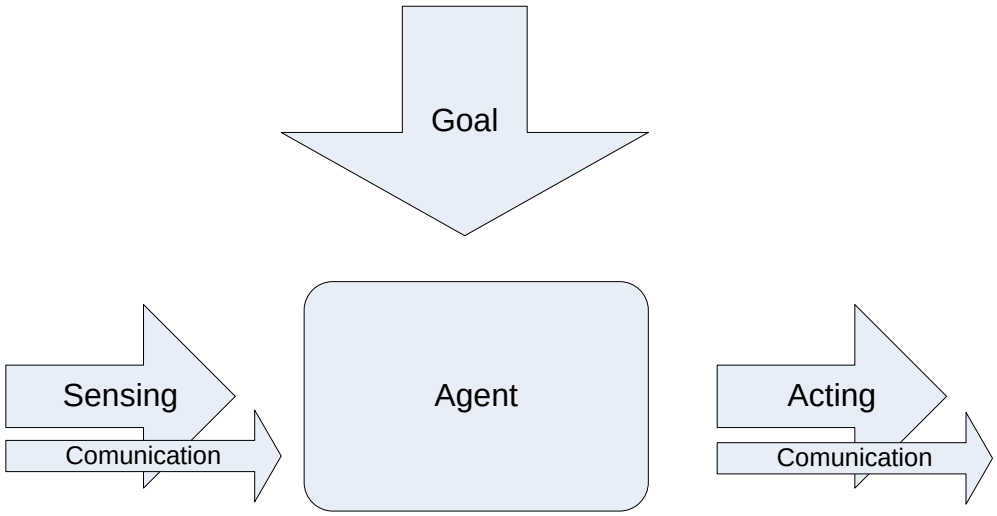
Agen otonom sesuai dengan namanya harus bersifat otonom, mandiri, reaktif, proaktif. Biasanya agen otonom terhubung dengan suatu server, sehingga mereka dapat berkomunikasi dengan agen lain dengan bahasa yang telah disepakati antar agen itu (ACL = *Agent Communication Language)* . Maksud dari sifat otonom adalah agen dapat bertindak tanpa adanya control dari manusia atau intervensi lain sesuai dengan kecerdasan buatan yang ditanamkan padanya. Agen juga harus

bersifat reaktif, artinya agen mampu secara terus-menerus berinteraksi dengan lingkungannya dan mampu memberikan respon yang tepat terhadap setiap perubahan yang terjadi di lingkungannya. Selain bersifat reaktif agen juga harus pro-aktif, artinya agen harus mampu mengambil inisiatif sendiri, tidak menunggu sesuatu terjadi sesuatu terlebih dahulu baru bertindak. Sebagai contoh sifat pro aktif agen adalah jika ada musuh memasuki wilayahnya, agen tidak perlu menunggu musuh melepaskan tembakan dulu baru menyerang, tetapi agen akan proaktif menyerang setiap musuh yang memasuki wilayahnya (Ingham, 1997). Selanjutnya pada penelitian ini akan digunakan istilah agen untuk menyebut NPC berbasis agen otonom.

Setiap agen dalam game selalu memiliki tujuan (*goal*), seperti : menyerang musuh, tetap hidup, menangkap avatar. Agen atau aotonomous NPC juga mampu merasakan (*sensing*) perubahan pada lingkungannya, seperti kemampuan melihat halangan, mendengarkan suara. Dan agen juga dapat bertindak (*acting*) sesuai perubahan lingkungan yang ditemuinya, seperti : melompat untuk menghindari halangan, makan. agen diberi semacam indera untuk dapat mengindera lingkungannya dan untuk berkomunikasi dengan agen lain. Gambar 2. 1 adalah ilustrasi interaksi agen/NPC dengan lingkungannya.

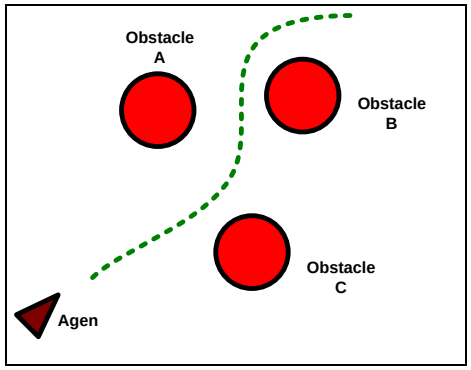
Saat bergerak di dalam lingkungannya, sekelompok agen harus mengatur pergerakaannya agar tidak berbenturan dengan agen lain. *Steering behavior* bertujuan untuk membantu agen bergerak secara realistik di lingkungan buatan di mana agen hidup. Pergerakan ini diciptakan mengacu pada informasi lokal tentang posisi agen lain yang berdekatan dengannya.

Penelitian ini lebih difokuskan pada bagian *steering* yang didalamnya berisi perilaku agen menentukan rencana-rencana untuk menyerang predator secara berkelompok.



Gambar 2. 1 Ilustrasi Interaksi Agen Terhadap Lingkungannya

Craig W Reynold membagi steering behavior menjadi beberapa perilaku. Antara lain *seek, flee, pursuit, evasion, offset pursuit, arrival, obstacle avoidance, wander, path following, unligned collision avoidance, separation, cohesion* dan *alignment*. Tetapi pada penelitian ini hanya akan digunakan *obstacle avoidance*, dan *separation* saja.

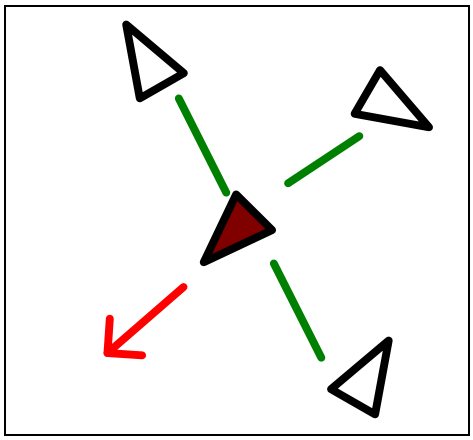


Gambar 2. 2 Obstacle Avoidance

Gambar 2. 2 adalah ilustrasi gerakan agen saat menghindari halangan (*obstacle avoidance*). *Obstacle avoidance* adalah perilaku agen melakukan maneuver untuk mengindari benturan dengan hambatan (*obstacle*). Ada perbedaan mendasar antara *obstacle avoidance* dengan *flee*. *Flee* akan selalu bergerak menjauh dari target

sedangkan *obstacle avoidance* hanya bergerak menghindari rintangan jika menemukan rintangan di depannya (Jatiningsih dkk., 2014).

Separation adalah perilaku agen untuk menjaga jarak dengan agen lain saat dalam kerumunan/kelompok. Hal ini untuk menghindari agar agen-agen tidak menumpuk di satu tempat. Gambar 2. 3 adalah ilustrasi agen saat menjaga jarak dengan agen lain agar tidak saling bertrabakan (*separation*).



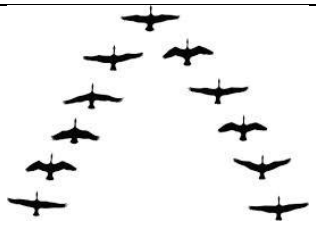
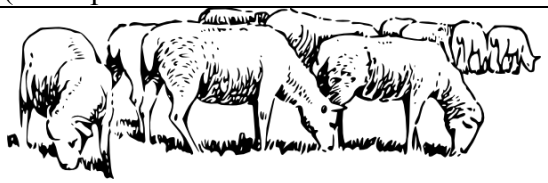
Gambar 2. 3 Separation

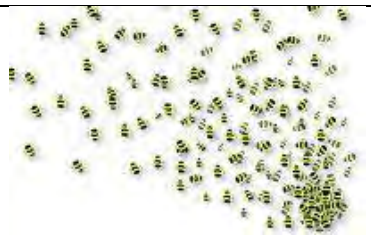
2.3 NPC Swarming

Jika kita mencermati lingkungan dan alam sekitar, ada banyak fenomena yang menarik untuk dicermati. Manusia memanfaatkan hal tersebut untuk belajar sesuatu dan melahirkan ilmu-ilmu tertentu yang terus dipelajari dan dikembangkan sampai sekarang ini seperti : fisika, astronomi, pertanian, dan sebagainya. Dari fenomena tersebut, manusia akhirnya membuat teori, metoda, formula, dalil dan sebagainya untuk mencoba mendefinisikan sesuatu ataupun memberi pengertian kepada sesamanya. Di ilmu komputer, salah satu nya adalah kecerdasan berkoloni dari binatang. Tingkah laku hewan tersebut memberi inspirasi kepada para peneliti dan ilmuwan untuk melahirkan sebuah teori atau pun algoritma. Beberapa algoritma tersebut antara lain adalah ACO (Ant Colony Optimization), PSO (*particle swarm optimization*), BCO (*bee colony optimization*), CSO_a (*Cat swarm optimization*),

dan CSO_b (*cockroach swarm optimization*). Tabel 2. 1 menunjukkan jenis algoritma dan sumber inspirasinya.

Tabel 2. 1 Jenis hewan yang menjadi inspirasi dari algoritma

No.	**Nama**	***Swarming***
1	Burung	 (http://www.istockphoto.com/illustrations/birds+flying+in+v-formation)
2	Ikan	 (www.pinterest.com/taniamoliveira/ilustrations/)
3	Domba	 (http://www.freepik.com/free-vector/grazing-sheep-clip-art_381081.htm)
4	Semut	 (http://blog.timesunion.com/marymartin/fresh-new-typing-of-creepy-crawlies/1992/)

5	Lebah	 (http://www.shutterstock.com/s/swarming/search.html)

Dalam kehidupan, makhluk hidup seperti burung, ikan maupun serangga, dilengkapi dengan fungsi biolgis sederhana dalam dirinya. Populasi mereka memiliki kecerdasan untuk melakukan persebaran untuk tiap individu dalam populasi tersebut, sehingga mereka dapat beradaptasi dengan lingkungan sekitar dan dapat hidup lebih lama. Contohnya seperti koloni lebah pada Gambar 2. 4 Jika saja sebuah NPC dalam game mendapatkan kecerdasan layaknya makhluk hidup tersebut maka dampaknya dalam game yaitu game akan lebih hidup dan menjadi semakin menarik.

Gambar 2. 4 Makhluk hidup memiliki kemampuan biologis sendiri - sendiri contohnya lebah yang hidup dalam sebuah colony memiliki tugas masing-masing tiap individu

Satu diantara banyak algoritma yang menggunakan teknologi *swarm* yaitu

algoritma *particle swarm optimization* (PSO) yang pertama kali diperkenalkan oleh Kennedy dan R. Eberhart pada tahun 1995. Algoritma PSO ini merupakan salah satu hasil perkembangan dari kecerdasan buatan yang terinspirasi dari perilaku pergerakan kawanan hewan seperti ikan (*school of fish*), burung (*flock*), dan kawanan serangga seperti semut rayap dan lebah. Tiap objek hewan disederhanakan menjadi partikel. Algoritma ini menerapkan sifat masing masing individu dalam satu kelompok besar kemudian menggabungkan sifat sifat tersebut untuk menyelesaikan masalah. Algoritma PSO adalah salah satu metode dari teknik kecerdasan buatan *swarm intelligence* (SI) yang berlandaskan perilaku kolektif (*collective beharviour*) yang dapat mengatur dirinya sendiri (*self-organizing*), dalam SI algoritma *particle swarm optimization* ini juga merupakan metode yang digunakan untuk penentuan posisi partikel terbaik (Nugroho dkk., 2011).

Swarm Intelligence (SI) merupakan cabang dari *Artificial Intelligence* (SI) yang digunakan untuk memodelkan perilaku kolektif dari hewan social di alam seperti koloni semut, lebah atau burung. Meskipun setiap agen dalam swarm cenderung memiliki kemampuan yang sederhana tetapi mereka saling berkomunikasi dan melakukan pembagian tugas yang jelas untuk kelangsungan hidup mereka. Interaksi social antar anggota swarm dapat dilakukan secara langsung maupun tidak langsung. Interakasi langsung dilakukan dengan melakukan kontak visual atau suara, seperti pada saat lebah pekerja (*employed bee*) mempresentasikan informasi tentang jumlah *nectar* pada *food source* yang dikunjunginya di lantai dansa (*dance area*) dengan melakukan tarian (*wagle dance*) yang ditujukan pada lebah pengamat (*onlooker bee*). Sedangkan interaksi tidak langsung terjadi pada saat ada anggota kelompok yang melakukan perubahan pada lingkungannya dan anggota yang lain berperilaku sesuai perubahan lingkungan itu. Contoh interaksi tidak langsung adalah ketika semut berkomunikasi dengan temannya dengan meninggalkan jejak feromon pada jalur yang dia lewati saat menuju sumber makanan, kemudian semut lain akan bergerak mengikuti jejak feromon itu.

Beberapa penelitian yang mengimplementasikan algoritma PSO (particle

swarm optimization) pernah dilakukan, salah satunya penelitian yang di lakukan oleh Safril Rizki Waluyo, Mochamad Hariadi dan I ketut Eddy purnama. Penelitian yang dilakukan adalah tentang pencarian jalur terbaik menggunakan algoritma PSO untuk mengoptimasi lalu lintas kendaraan, pada penelitian yang dilakukan mereka mensimulasikan permasalahan lalu lintas dan mencoba mengimplementasikan algoritma PSO kepada kelompok obyek yang ada dalam simulasi. Untuk meringankan beban kerja sistem, obyek sederhana yang digunakan adalah cube. Obyek inilah yang dijadikan sebagai partikel partikel dimana algoritma PSO ini diimplementasikan terhadap sifat dari obyek atau partikel tersebut, partikel partikel yang ada memiliki sifat, perilaku dan tujuan yang sama. Dari penelitian yang dilakukan oleh Safril Rizki Waluyo, Mochamad Hariadi dan I ketut Eddy purnama menunjukan algoritma PSO yang diimplementasikan dapat mengoptimasi lalu lintas kendaraan .

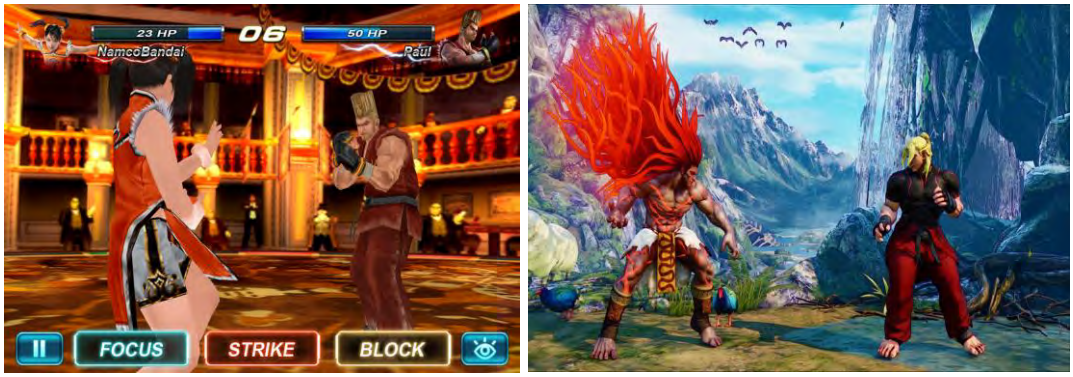
Dengan saran dari penelitian yang dilakukan oleh Safril Rizki Waluyo, Mochamad Hariadi dan I ketut Eddy purnama, penelitian kali ini akan mencoba mengimplementasikan algoritma PSO untuk model perilaku non-player character (NPC) menyerang secara berkelompok pada game action-RPG. Model perilaku NPC menyerang secara berkelompok ini merupakan model perilaku multi NPC dimana setiap NPC dalam suatu kelompok NPC tersebut dapat saling berkomunikasi dalam menyerang karakter player. Perilaku menyerang secara berkelompok dalam game ini ditunjukkan dari bagaimana kelompok NPC tersebut mengejar karakter player untuk menghalangi karakter player tersebut dalam mencapai misi tertentu dalam game.

Dalam mengejar karakter player setiap NPC dalam kelompok NPC ini akan berkomunikasi untuk menentukan NPC mana yang memiliki posisi terbaik, NPC yang memiliki posisi terbaik ini akan menjadi pemimpin dari NPC yang lainnya, untuk kemudian di ikuti oleh NPC lainnya tersebut dalam mengejar karakter player. Dengan itu timbul masalah bahwa setiap NPC dalam kelompok NPC tidak akan bisa begitu saja memilih NPC mana yang memiliki posisi terbaik untuk dijadikan pemimpin yang akan diikuti NPC lain yang bukan merupakan pemimpin dalam

mengejar karakter player.

Untuk itu dibutuhkan algoritma yang dapat membuat setiap NPC dalam kelompok NPC tersebut untuk dapat memilih NPC mana yang menjadi pemimpin agar setiap NPC yang bukan merupakan pimimpin dapat mengikuti NPC pemimpin dalam mengejar karakter player.

Ditinjau dari jumlah agen saat melakukan serangan, serangan dalam game dibedakan menjadi 2, yaitu serangan tunggal dan serangan dalam kelompok. Yang dimaksud dengan serangan tunggal adalah dimana player dan musuh berhadapan satu lawan satu. NPC hanya berfokus strategi penyerangan single target saja tidak harus memikirkan formasi penyerangan musuh maupun formasi rekan. Biasanya serangan tunggal ini digunakan pada game bergenre *fighting* game contohnya saja *Street Fighter, Teken,* dan *Bloody Roar* seperti yang terlihat pada Gambar 2. 5.



Gambar 2. 5 Game bergenre fighting merupakan contoh game yang menggunakan serangan tunggal

Diperlukan kecerdasan buatan yang ditanamkan pada agen agar terbentuk sekelompok agen yang dapat menyerang bersama-sama. Berkelompok atau *Swarming* pada dasarnya meniru perilaku hewan sosial, seperti kawanan ikan, burung dan serangga misalnya semut, rayap dan lebah. Setiap anggota kelompok berperilaku tanpa supervisi dan memiliki perilaku stokastik karena persepsi dirinya terhadap lingkungan sekitarnya. *Swarm intelligence* (SI) adalah perilaku menyebar secara

kolektif dan memiliki aturan pengorganisasian sendiri (*self-organized*).

2.4 Bee Colony

The Artificial Bee Colony (ABC), diperkenalkan oleh Dervis Karaboga pada tahun 2005. ABC dapat memecahkan permasalahan optimasi yang terbatas. Rekayasa desain, optimasi struktural, ekonomi, desain VLSI, alokasi dan lokasi masalah hanya beberapa dari bidang ilmiah yang Dibatasi masalah Optimasi sering bertemu. ABC merupakan kecerdasan buatan yang menirukan koloni lebah dalam mencari sumber nektar (sari bunga). Kemampuan koloni lebah dalam menentukan sumber makanan terbagi menjadi tiga kelompok yaitu lebah pekerja, lebah penjelajah dan lebah pengintai.

- *Employe* : seekor lebah yang pergi ke sumber makanan yang telah dikunjungi sebelumnya dengan sendirinya dinamakan *Employe*.
- *Onlookers*: Sebuah lebah yang menunggu di ruang lingkup- pencarian untuk membuat keputusan untuk memilih sumber makanan.
- *Scout* : Lebah yang melakukan pencarian acak disebut *scout*.

Lebah-lebah ini melakukan suatu fungsi untuk menentukan letak dan besar suatu sumber nektar kemudian mengingat dan membandingkan dengan sumber lain. Pada akhir fungsi dipilih suatu lokasi dengan sumber nectar yang paling optimal. Haiyan Quan dan Xinling Shi mendiskusikan berbagai macam permasalahan ABC. Perbaikan dalam ABC disarankan dengan menganalisis beberapa fungsi oleh haiyan Quan dan Xinling Shi. Sebuah paper dari Li-Pei, Wong Malcolm Yoke Hean Low dan Chin Soon Chong melakukan penelitian terhadap suatu kegiatan menggunakan pendekatan ABC dengan menyelesaikan suatu permasalahan perjalanan seorang sales berdasarkan pada analisis algoritma *Bee Colony*. Pengembangan suatu desain filter IIR (Infinite Input Respon) oleh Nurhan Karaboga, dengan algoritma *Bee Colony* menjadi dasar pada pengembangan performansi dari filter IIR agar lebih optimal.

2.4.1 Pencarian Madu oleh serangga lebah

Setiap sumber bunga memiliki keragaman dan perbedaan dalam hal kualitas. Sehingga seseorang mungkin memiliki pikiran bahwa keputusan tetang jumlah lebah yang harus dikirim ke setiap tempat sumber bunga dan lama lebah disumber tersebut merupakan sebuah perkumpulan madu bunga (nectar) sebanyak mungkin. Berkat sistem kerja kawanan lebah yang sangat baik maka lebah mampu memecahkan permasalahan ini tanpa mengalami kesulitan.

Kawanan lebah yang berada di dalam sebuah sarang ada yang bertugas sebagi pengumpul nectar. Mereka memiliki tugas untuk berkeliling diantara bunga-bunga dan mengumpulkan nectar sebanyak mungkin. Ketika kembali ke sarang, mereka menyerahkan seua nectar yang mereka bawa kepada lebah-lebah yang bertugas menjaga sarang dan menyimpan bahan makanan. Lebah-lebah penyimpan bahan makanan ini kemudian menyimpan nectar didalam sel-sel madu. Seekor lebah pengumpul nectar juga dibantu oleh lebah pengumpul nectar lain dalam menentukan kualitas sumber bunga yang ditentukannya. Lebah pengumpulan nectar tersebut mengamati dan menunggu untuk bertemu dengan seekor lebah penyimpan makanan yang telah siap menerima nectar.

Selanjutnya ketika menunggu, lebah memiliki waktu tunggu sehingga jika waktu tunggu ini berlangsung lama, maka sang lebah pengumpul nectar tersebut memahami hal tersebut. Hal ini adalah sebagai isyarat bahwa sumber bunga yang dia temukan bukan dari mutu yang baik dan sumber bunga terbaik telah ditemukan oleh lebah-lebah lain. Sebaliknya jika lebah pengumpul makanan tersebut disambut oleh sejumlah besar lebah-lebah penyimpanan makanan maka sebagian besar kemungkinan bahwa muatan nectar yang dia temukan memiliki kualitas yang baik.

Lebah yang memperoleh informasi ini memutuskan apakah sumber bunganya senilai dengan kerja keras yang akan dilakukan pada tingkat selanjutnya. Jika iya, maka lebah ini melakukan tarian getarnya agar dipahami oleh lebah-lebah lain. Lama

tarian ini memperlihatkan seberapa besar keuntungan yang mungkin dapat diperoleh dari sumber bunga ini.

2.4.2 Penyimpanan maksimum dengan bahan minimum

Sarang lebah dibangun dari dinding lilin lebah sarang madu dan diarang tersebut terdapat ratusan sel-sel kecil di tiap-tiap permukaan. Semua sel sarang lebah madu mempunyai ukuran yang sama tepat. Keajaiban teknik ini didapat dari kerja sama ribuan lebah. Lebah menggunakan sel tersebut untuk meletakkan makanan dan mengatur lebah muda.

Lebah madu menggunakan struktur heksagonal sebagai konstruksi pembangunan sarang mereka. Para ahli matematika memberikan alas an bahwa struktur heksagonal memiliki bentuk geometric yang paling cocok untuk penggunaan maksimum dari suatu area. Jika sel sarang lebah dibentuk dengan bentuk lain seperti segiempat atau segitiga makan ada area yang tidak digunakan kemudian hanya sedikit madu yang akan disimpan dan semakin sedikit lebah yang mendapatkan manfaat dari penyimpanan madu tersebut.

Untuk kedalaman sarng yang sama, sarang berbentuk sel segitiga atau segiempat akan menampung jumlah madu yang sama dengan sel heksagonal mempunyai lingkaran yang paling dekat, dengan demikian jumlah lilin yang dibutuhkan untuk membangun sel sarang berbentuk heksagonal lebih sedikit daripada jumlah lilin yang dibutuhkan untuk sel segitiga atau segiempat. Sehingga dapat diambil kesimpulan bahwa sel-sel heksagonal membutuhkan jumlah minimal dari lilin dalam kontruksi ketika mereka menyimpan jumlah madu yang maksimal.

Selain dalam pembuatan dinding sel, kawanan lebah ini juga membawa prinsip penyimpanan makanan dalam pertimbangan ketika mereka membangun bagian bawah. Sarang dibuat dengan bentuk pipih dengan dua baris sel yang saling membelakangi. Dalam kasus ini masalah dari titik pertemuan dari dua baris muncul.

Lebah membangun permukaan dasar sel-sel dengan mengkombinasikan tiga segiempat untuk menyelesaikan masalah ini. Ketiga sel dibangun disisi sarang, permukaan dasar dari satu sel yang berada disisi yang lain secara otomatis juga terbentuk. Karena permukaan bawah tersusun dari plat-plat lilin bujur sangkar, maka bagian bawah sel-sel yang dibuat jadi bertambah dalam sehingga menambah volume sel dan juga jumlah madu yang disimpan.

2.4.3 Cara lebah membagi informasi sumber makanan

Dalam pencarian sumber makanan, lebah menempuh jarak yang jauh dan menjelajah are yang luas untuk menentukan sumber makanan. Lebah yang menentukan bunga kembali kesarang untuk member tahu letak sumber makanan yang telah ditemukan, lebah tersebut menari untuk mendeskripsikan lokasi bunga tersebut kepada lebah lain didalam sarang.

Tarian ini berarti ekspresi yang digunakan untuk menginformasikan kepada lebah yang lain tentang lokasi dari bunga. tarian ini dilakukan berulang0ulang oleh lebah, termasuk untuk semua informasi kemiringan, arah, jarak, detail lain dari sumber makanan yang membuat lebah lain mampu untuk mencapai lokasi yang diperoleh.

Tarian ini membentuk angka “8” diulang-ulang secara konstan oleh lebah. Lebah membentuk bagian tengah dari angka 8 dengan menggoyangkan ekornya dan berzigzag. Sudut diantara zigzag dan garis diantara matahari serta sarang memberikan arah yang tepat dari sumber makanan.

2.4.4 Metode penandaan bunga

Lebah madu dapat mengetahui kalau bunga yang dia temui telah didatangi dan nectar yang ada telah diambil lebih dahulu oleh lebah lain, maka dia segera maninggalkan lokasi tersebut dan langsung mencari lokasi yang lain. Hal ini terjadi karena lebah lain yang mendatangi letak bunga terlebih dahulu menandai dengan tetesan berbau khas. Begitu seekor lebah baru mengunjungi bunga yang sama, dia

akan mencium bau tersebut dan mengetahui bahwa bunga tersebut sudah tidak berguna dan langsung pergi kebunga lain. Dengan demikian lebah tersebut dapat menghemat waktu dan tenaga dalam menemukan sumber makanan.

2.4.5 Pemakaian konsep lebah untuk optimasi

Dua konsep dasar untuk kinerja kolektif konsep swarm disajikan dibawah ini, yaitu organisasi diri (*self organization*) dan pembagian kerja. keduanya diperlukan sebagai property untuk mendapatkan perilaku kecerdasan swarm seperti halnya sistem pemecahan masalah terdistribusi, yang mengatur dirinya sendiri dan beradaptasi dengan lingkungan tertentu. Dua konsep dasar tersebut :

1. Organisasi mandiri (*self organization*) dapat didefinisikan sebagai seperangkat mekanisme dinamis, yang menghasilkan struktur ditingkat global dari sistem memulai interaksi diantara komponen tingkat rendah. mekanisme ini menetapkan aturan dasar untuk interaksi antar komponen-komponen sistem. Aturan tersebut memastikan bahwa interaksi dijalankan atas dasar informasi local murni tanpa ada hubungannya dengan pola global. Bonabeau telah menandai empat dasar organisasi diri yaitu umpan balik positif, umpan balik negative, fluktuasi dan multiple interactions.
2. Di dalam ABC dan swarm ada tugas berbeda yang dilakukan secara bersamaan oleh individu-individu khusus. Fenomena semacam ini disebut sebagai pembagian kerja. Performa tugas simultan melalui kerja sama diantara individu-individu khusus tersebut diyakini lebih efisisen dari performa tugas secara berurutan oleh individu tanpa spesialisasi. pembagian kerja juga memungkinkan swarm untuk merespon perubahan kondisi dalam ruang pencarian.

2.4.6 Pemodelan perilaku kawanan lebah madu

Model minimal dari seleksi mencari makan yang mengarah pada munculnya kecerdasan kolektif kawanan lebaha madu, terdiri dari tiga komponen penting yaitu sumber makanan, lebah pekerja dan lebah *unemployed*. Model tersebut

mendefinisikan dua modus perilaku yang paling penting yaitu rekrutmen ke sumber nektar dan ditinggalkan ke sumber.

1. **Sumber makanan**. Nilai sumber makanan tergantung pada banyak faktor yaitu jarak terdekat kesarang, kekayaan atau konsentrasi dari energi sumber makanan tersebut dan tingkat kemudahan dalam pengambilan energi makanan. Untuk penyederhanaan keuntungan dari sumber makanan dapat diwakili dari satu kuantitas.
2. **Lebah pekerja**. Mereka dikaitkan dengan sumber makanan tertentu yang sedang mereka eksploitasi atau tempat mereka dipekerjakan. Mereka juga membawa informasi tentang letak dari sumber makanan, karak dan arah dari sarang. Profitabilitas atau keuntungan sumber makanan tersebut dan membagikan informasi ini dengan nilai keuntungan tertentu.
3. **Lebah *unemployed***. Mereka secara terus menerus keluar mencari sumber makanan untuk dieksploitasi. Ada dua jenis lebah unemployed yaitu lebah *scout* yang bertugas mencari lingkungan disekitar sarang untuk mendapatkan sumber makanan baru dan lebah *onlooker* yang bertugas menunggu disarang dan mendapatkan sumber makanan melalui informasi yang dibagikan oleh lebah pekerja. Jumlah rata-rata lebah *scout* sekitar 5-12% dari jumlah lebah keseluruhan.

Pertukaran informasi diantara lebah adalah kejadian yang paling penting dalam pembentukan pengetahuan kolektif. Saat memeriksa selurug sarang, dimungkinkan untuk dapat memebedakan beberapa bagian yang umumnya ada disemua sarang. Dan bagian paling penting dari sarang yang berkaitan dengan pertukaran informasi adalah dancing area. Komunikasi diantara lebah yang berkaitan dengan mutu sumber makanan terjadi di dancing area. Tarian lebah ini disebut dengan *waggle dance*.

Karena informasi tentang semua sumber yang kaya makanan tersedia untuk lebah *onlooker* di dancing area, memungkinkan lebah tersebut untuk dapat menonton berbagai tarian lebah dan menutuskan untuk mempekejakan dirinya pada sumber yang paling menguntungkan.

Ada peluang yang lebih besar bagi lebah *onlooker* untuk memilih sumber-sumber makanan yang lebih menguntungkan, karena lebih banyak informasi yang beredar tentang sumber-sumber makanan yang lebih menguntungkan tersebut. Lebah pekerja membagi informasi sesuai dengan probabilitas yang sebanding dengan profitability dari sumber makanan, dan membagi informasi melalui *waggle dancing* dengan durasi yang lebih lama. oleh karena itu, rekruitmen sebanding dengan profitability dari sumber makanan.

Diasumsikan bahwa ada dua sumber makanan yang ditemukan, yaitu A dan b yang pada mulanya lebah pencari makan yang potensial akan mulai sebagai lebah unemployed. lebah tersebut tidak memiliki pengetahuan tentang sumber makanan disekitar ruang. Ada dua opsi pilihan untuk lebah tersebut yaitu:

1. Bisa menjadi scout dan mulai mencari-cari sumber makanan disekitar sarang secara spontan karena adanya motifasi internal atau petunjuk eksternal yang mungkin
2. Bisa menjadi rekrutan setelah menonton *waggle dance* dan mulai mencari sumber makanan

Setelah menemukan sumber makanan, lebah tersebut menggunakan kemampuannya sendiri untuk mengingat lokasi sumber makanan dan kemudian mulai mengeksploitasinya segera. Oleh karena itu, lebah tersebut akan menjadi lebah pekerja. lebah pekerja mengambil nectar lalu kembali kesarang dan membongkar nectar pada tempat persediaan makanan. Setelah pembongkaran makanan, lebah pekerja tersebut memiliki tiga opsi yaitu menjadi pengikut tidak terikat setelah meninggalakan sumber makanan, melakukan waggle dance dan kemudian merekrut lebah lainnya sebelum kembali kesumber makanan yang sama atau meneruskan untuk mencari makanan disumber makanan semula tanpa merekrut lebah lainnya.

Dan penting dicatat bahwa tidak semua lebah mulai mencari makan secara bersamaan. Percobaan yang telah dilakukan menegaskan bahwa lebah yang baru

mulai mencari makan pada tingkat yang sebanding dengan perbedaan antara jumlah akhir lebah dan jumlah lebah yang sedang mencari makan.

Dalam kasus lebah madu, sifat dasar organisasi yang mengandalkan diri sendiri (*self organization*) adalah sebagai berikut :

1. umpan balik positif. ketika jumlah nectar sumber makanan meningkat, jumlah lebah onlooker yang mengunjungi sumber makanan tersebut meningkat juga.
2. Umpan balik negative. Proses eksplorasi sumber makanan yang ditinggalkan oleh lebah dihentikan.
3. *Fluktuasi*. Lebah scout melakukan proses pencarian acak untuk menentukan sumber makanan baru.
4. *Multiple interactions*, lebah pekerja membagi informasi tentang posisis sumber makanan dengan lebah-lebah lainnya pada area dance.

Dalam algoritma koloni lebah buatan, posisi sumber makanan merupakan solusi yang mungkin untuk masalah optimasi dan jumlah nektar dari sumber makanan merujuk pada kualitas (*fitness*) dari solusi terkait. Jumlah lebah yang bekerja atau *onlooker* sama dengan jumlah solusi dalam populasi. Algoritma *bee colony*, terdiri dari langkah-langkah berikut.

Artificial Bee Colony menghasilkan populasi awal secara acak dari solusi TS(The Solution). Dengan "i" adalah representasi dari solusi ke- (i = 1,2, ..., TS) dan "j" adalah (j = 1,2, ..., ND) D-dimensi. Setiap solusi adalah sebuah xi (i = 1,2, ..., TS) vektor D-dimensi. TS menunjukkan ukuran populasi. Langkah pertama, Setelah inisialisasi, populasi posisi (solusi) terkena siklus berulang dari (siklus = 1,2, ..., MC), mencari *employe*, lebah onlooker dan lebah pramuka yang bekerja. MC adalah jumlah siklus. Sebuah *employe* buatan atau lebah *onlooker* secara probabilistik memodifikasi posisi (solusi) dalam memori untuk menemukan sumber makanan baru dan tes jumlah nektar (nilai fitness) dari sumber baru (solusi baru). Jika jumlah nektar yang baru lebih banyak dari sebelumnya, lebah melupakan yang lama dan menghafal

posisi baru. Jika tidak, dalam ingatannya, dia terus posisi sebelumnya. Mereka berbagi informasi nektar dari sumber makanan (solusi) dan informasi posisi mereka dengan lebah onlooker di ruang lingkup . Setelah semua *employe* menyelesaikan proses pencarian, Seperti dalam kasus lebah digunakan, hal tersebut menghasilkan modifikasi pada posisi (solusi) dalam memori dan memeriksa jumlah nektar dari sumber calon (solusi), dan lebah onlooker mengevaluasi informasi nektar yang diambil dari semua lebah bekerja dan kemudian memilih sumber makanan dengan probabilitas yang terkait dengan jumlah nektar nya. Jika nomor nektar yang baru adalah lebih tinggi dari sebelumnya, lebah melupakan yang lama dan menghafal posisi baru. Seekor lebah onlooker buatan memilih sumber makanan tergantung pada nilai probabilitas metode roulette wheel yang terkait dengan sumber makanan (Bhattacharjee dkk., 2011), Onlooker akan memilih foodsource berdasarkan pada nilai probabilitas foodsource-nya (p_i) yang dihitung menggunakan persamaan 2. 1:

$$P_i = \frac{f_i}{\sum_i^{TS} f_n} \qquad 2.1$$

Ket. :

P_i = Probabilitas ke-i

f_i = Nilai fitness dari solusi ke-i

f_n = Total nilai fitness

TS = Jumlah sumber makanan (solusi)

f_i adalah nilai fitness dari solusi *i* yang telah dievaluasi oleh lebah *employe*. Nilainya sebanding dengan jumlah nektar dari sumber makanan di posisi *i*. TS adalah jumlah sumber makanan yang berjumlah sama dengan jumlah lebah yang pekerja. Sedangkan untuk menentukan kandidat posisi foodsource baru, algoritma ABC menggunakan persamaan 2. 2:

$$x_{ij}' = x_{ij} + \varphi_{ij}(x_{ij} - x_{kj}) \quad 2.2$$

Ket.:

x_{ij}' = Posisi baru

x_{ij} = Posisi lama

φ_{ij} = Nilai random dari -1 hingga 1

x_{kj} = Posisi target

Dalam persamaan, k ∈ {1, 2, ..., TS} dan j ∈ {1,2, ..., ND} adalah indeks yang dipilih secara acak. K ini adalah SN dari yang dipilih secara acak, tetapi harus berbeda dari i. Dimana *Φij* adalah nomor acak antara [-1,1]. Nilai tersebut mengontrol produksi sumber makanan tetangga *xij* terdekat dan merupakan perbandingan dari dua posisi makanan secara visual oleh lebah. Jika kita mencari pendekatan untuk solusi optimal dalam ruang pencarian, maka panjang langkah berkurang secara adaptif. Jika nilainya (*Φij*) yang dihasilkan oleh operasi ini melebihi batas yang telah ditetapkan ([-1,1]), parameter dapat diatur ke nilai yang dapat diterima dan nilai parameter diatur ke nilai batasnya. Oleh lebah *scout*, sumber makanan yang nektarnya ditinggalkan oleh lebah diganti dengan sumber makanan baru. Hal ini disimulasikan dengan memproduksi posisi acak dan menggantinya dengan nectar yang ditinggalkan tadi di *Artificial Bee Colony*.

Setelah masing-masing posisi calon sumber x_{ij}' diproduksi, kemudian dievaluasi oleh lebah buatan, kinerjanya dibandingkan dengan yang lama. Dimana *Φij* adalah nomor acak antara [0,1]. Jika sumber makanan baru memiliki nektar yang sama atau lebih baik dari yang lama, sumber makanan tersebut diganti dengan yang lama dalam memori. Jika tidak, yang lama masih dipertahankan dalam memori. Mekanisme seleksi kasar ini digunakan sebagai operasi seleksi antara kandidat lama dan baru. Dalam kasus apapun, dijelaskan di atas bahwa ada tiga bagian yang

digunakan dalam *Artificial Bee Colony*: Jumlah sumber makanan yang sama dengan jumlah lebah yang bekerja atau penonton (TS), nilai limit, siklus maksimum (MC) (Bhattacharjee dkk., 2011).

2.5 Fungsi Fitness

Fungsi fitness atau fungsi objektif/tujuan merupakan bagian terpenting dalam algoritma yang berbasis algoritma genetika. Fungsi objektif, sesuai dengan namanya berfungsi untuk mengevaluasi apakah sebuah nilai yang didapat kan dari sebuah proses, cocok dengan yang diharapkan atau tidak. Fungsi fitness yang digunakan dalam algoritma harus menyesuaikan dengan apa maksud dan tujuan alggoritma tesebut dibuat. Contoh kasus jika sebuah algoritma yang berbasis algoritma genetika dibuat untuk menentukan jarak terpendek, maka fungsi fitness yang digunakan adalah sebuah fungsi jarak yang digunakan untuk mengevaluasi apakah posisi saat ini telah mendekati targetnya atau belum.

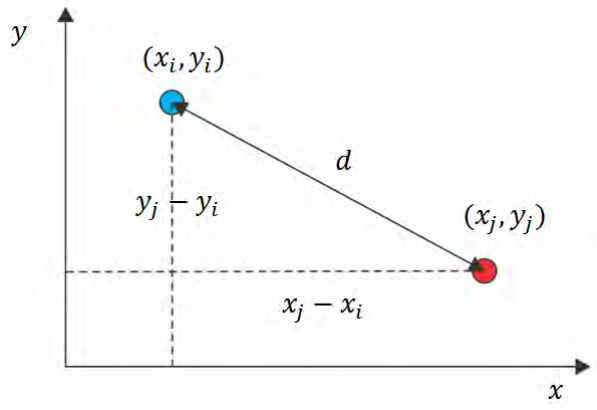
Dalam pembuatan algoritma game tidak sedikit kemungkinan digunakannya bermacam - macam fungsi fitness diantaranya fungsi jarak, fungsi halangan, fungsi optimal target, dan lain sebagainya. Fungsi-fungsi fitness tersebut menggunakan berbagai metode. Contoh kasus untuk menghitung jarak digunakan algoritma *Euclidian Distance* atau *Manhattan Distance*. untuk fungsi tabrakan, digunakan algoritma *Collision Detection*:

- *Euclidian Distance:*

Euclidean distance adalah perhitungan jarak dari 2 buah titik dalam *Euclidean space*. *Euclidean space* diperkenalkan oleh Euclid, seorang matematikawan dari Yunani sekitar tahun 300 B.C.E. untuk mempelajari hubungan antara sudut dan jarak. Euclidean ini berkaitan dengan Teorema Phytagoras dan biasanya diterapkan pada 1, 2 dan 3 dimensi. Tapi juga sederhana jika diterapkan pada dimensi yang lebih tinggi.

Metode Euclidean adalah suatu metode pencarian kedekatan nilai jarak dari 2 buah variabel, selain mudah metode ini juga tidak memakan waktu, dan proses yang cepat. Euclidean adalah fungsi heuristik yang diperoleh berdasarkan jarak langsung

bebas hambatan seperti untuk mendapatkan nilai dari panjang garis diagonal pada segitiga. Tetapi sebelum mendapatkan hasil kedua titik harus direpresentasikan ke dalam koordinat 2 dimensi (x, y) seperti yang terlihat pada Gambar 2. 6.



Gambar 2. 6 Ilustrasi *Euclidian distance*

Cara kerja algoritma ini lebih sederhana di bandingkan dengan euclid, karena tidak perlu menghitung berdasarkan bentuk bumi yang bulat dan dalam Geometri Euclid hanya satu lintasan yang merupakan jarak terpendek.

- *City Block Distance/Manhattan Distance*

Taxicab geometry, diusulkan oleh Hermann Minkowski di Jerman abad ke-19, adalah bentuk geometri di mana fungsi jarak yang merupakan geometri metrik atau Euclidean digantikan oleh metrik baru di mana jarak antara dua titik adalah jumlah dari perbedaan absolut koordinat Cartesian mereka.

Taxicab metric juga dikenal sebagai *rectilinear distance*, L_1 *distance* atau ℓ_1, *snake distance, city dlock distance*, atau *manhattan distance*, dengan variasi yang sesuai dalam nama geometri. Nama Manhattan diambil dari sebuah pulau di amerika, karena algoritma ini mengacu pada jalan-jalan di pulau Manhattan yang menyebabkan jalur terpendek mobil bisa memakan waktu antara dua persimpangan yang memiliki panjang yang sama dengan *Taxicab Geometry*.

Fungsi yang digunakan pada *manhattan distance* yaitu persamaan 2. 3

$$d_{pq} = \|p - q\|_1 = \sum_{i=1}^{n} |p_i - q_i| \qquad 2.\ 3$$

Dimana (p, q) adalah vector, $p = (p_1, p_2, p_3, \dots, p_n)$ dan $q = (q_1, q_2, q_3, \dots, q_n)$. Contohnya pada bidang datar, jarak manhattan distance antara (p_1, q_1) dan (p_2, q_2) adalah $|p_1 - q_1| + |q_1 - q_2|$

- *Collision Detection*

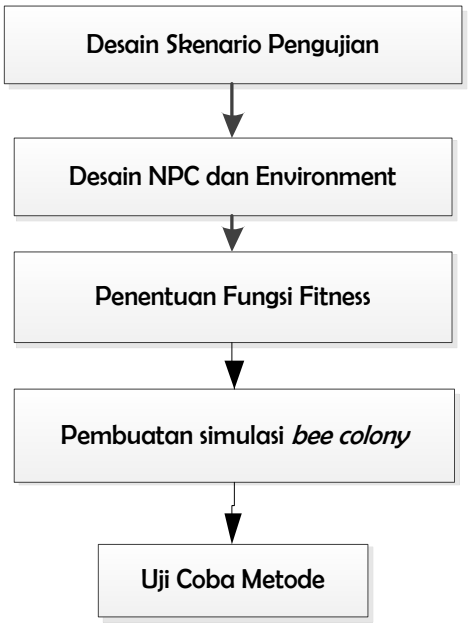
Algoritma *collision detection* adalah proses pengecekan apakah beberapa buah objek spasial saling bertumpuk atau tidak. Jika ternyata ada paling sedikit dua buah objek yang bertumpuk, maka kedua objek tersebut dikatakan saling bertumpukkan. Pada ruang spasial dua dimensi.Objek yang bertumpuk berarti objek spasialnya beririsan (Nugroho dkk., 2011).

Permasalahan *collision detection* adalah tentang simulasi computer, simulasi fisik, motion planning robot, dan banyak penelitian dengan permasalahan rumit lainnya. Sebuah algoritma *collision detection* yang akurat dan *real-time* dapat menambah keaslian dari lingkungan virtual dan *immersive. Collision detection* dibagi menjadi ruang tabrakan dua dimensi dan ruang tabrakan tiga dimensi. Pada bidang 2D, sebuah object dapat dapat di representasikan dengan bentuk polygon. Collision detection menggunakan irisan dari 2 buah polygon untuk mendeteksi adanya tabrakan pada object tersebut (Fan, 2012).

BAB III

METODE PENELITIAN

Pada penelitian ini, terdapat beberapa langkah dalam pembuatan simulasi permainan. Sebelum melakukan uji coba, terlebih dahulu menentukan desain skenario untuk pengujian. Gambar 3. 1 merupakan diagram aliran kerja yang digunakan pada penelitian ini dari perancangan desain scenario hingga pengujian metode dan visualisasi pergerakan.



Gambar 3. 1 Skema Metodologi Penelitian

Untuk pembuatan simulasi dan pengujian, digunakan program *Matlab* karena dengan program tersebut, data yang muncul bisa di monitor dengan jelas dan proses *debuging*dapat dilakukan dengan mudah.

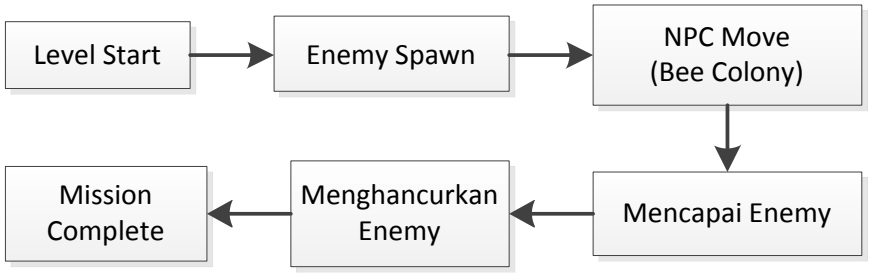
3.1 Desain Skenario

Wadah penelitian yang digunakan dalam penelitian ini berupa sebuah game *space shooter* 3D yang mengacu seperti permainan *galaxy on fire* seperti yang terlihat pada Gambar 3. 2 dimana dalam game tersebut dapat ditambahkan NPC yang akan menjadi aktor dalam penelitian ini. Game ini memiliki ruang gerak bebas secara 3D dan tidak ada gravitasi yang mempengaruhi character karena setting lokasi berada diluar angkasa. Setiap character memiliki parameter yang menyatakan status dari karakter tersebut. Parameter yang melekat tersebut diantaranya *Health Point* (HP), *Attack(ATK), defence(DEF), level, weapon* , *fire rate*, dll. Akan tetapi yang digunakan dalam perhitungan dalam penelitian ini hanya HP, ATK, dan DEF saja. Hasil dari penelitian ini dapat digunakan sebagai acuan untuk membuat gerombolan NPC dalam game bergerak secara harmonik menuju target yang telah dipilih.

Gambar 3. 2 Permainan *galaxy on fire*

Pergerakan NPC pada permainan space shooter ini pada awalnya belum memiliki kemampuan untuk bergerak menyebar ataupun menghindari serangan. Saat NPC tersebut *spawning* dan menemukan target, mereka hanya akan bergerak menuju target secara langsung. hal ini menimbulkan sebuah kejenuhan pada pemain

karena pergerakan NPC sudah dapat ditebak. Selain itu, NPC hanya akan mentargetkan pada enemy yang pertama kali masuk pada jarak serang dari NPC tersebut, sehingga pada saat banyak musuh yang masuk pada jarak serang, NPC hanya akan menyerang ke musuh yang pertama kali masuk tanpa memperhatikan bahwa kemungkinan menang melawan musuh tersebut lebih kecil dibandingkan melawan musuh yang lainnya. Dalam penelitian ini, metode digunakan untuk pencarian musuh. Pada permainan, berdasarkan pada Gambar 3. 3, saat NPC muncul maka algoritma *bee colony* otomatis akan bekerja.



Gambar 3. 3 Diagram alur kerja permainan

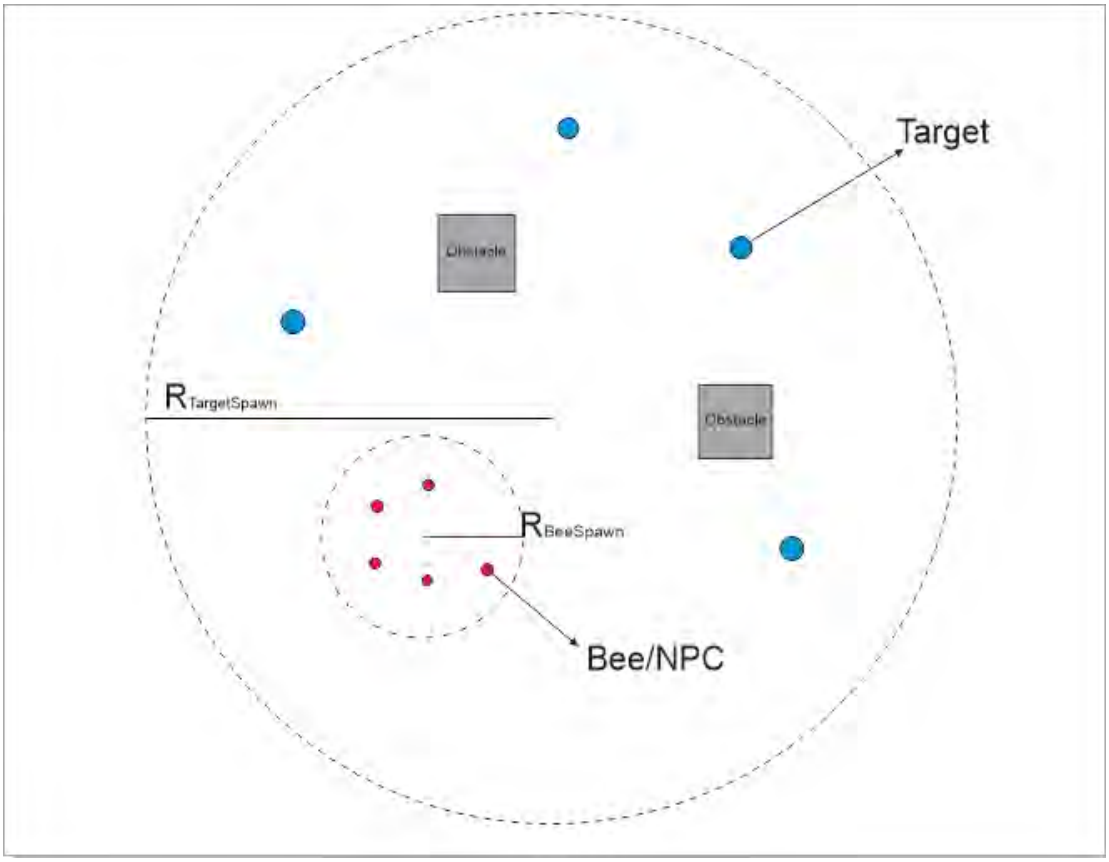
Pada penelitian ini, nantinya akan dilakukan pengujian dalam beberapa kondisi seperti yang tertera pada Tabel 3. 1.

Tabel 3. 1 Tabel Perlakuan Uji Coba

Uji Coba ke-n	**Posisi Target**	**Jumlah Target**	**Obstacle**
1	Statis	Single	No
2	Statis	Multiple	No
3	Dinamis	Multiple	No
4	Dinamis	Multiple	Yes

Visualisasi dilakukan dalam sebuah ruang virtual 3D dengan rentang jarak sumbu X, Y, dan Z antara -1000px sampai 1000px. Agen, *obstacle* dan target di

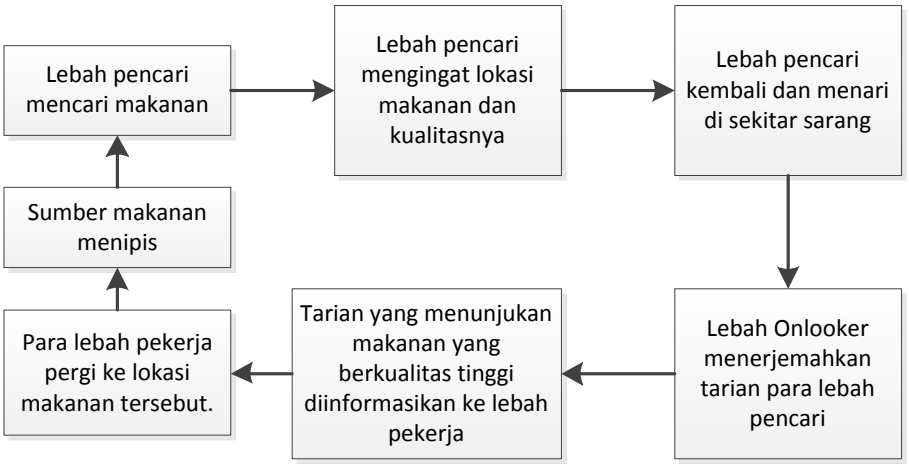
simbolkan dengan objek bulat dengan warna dan ukuran yang berbeda-beda. Seperti yang terlihat pada Gambar 3. 4



Gambar 3. 4 Skema penempatan aset dsaat pengujian

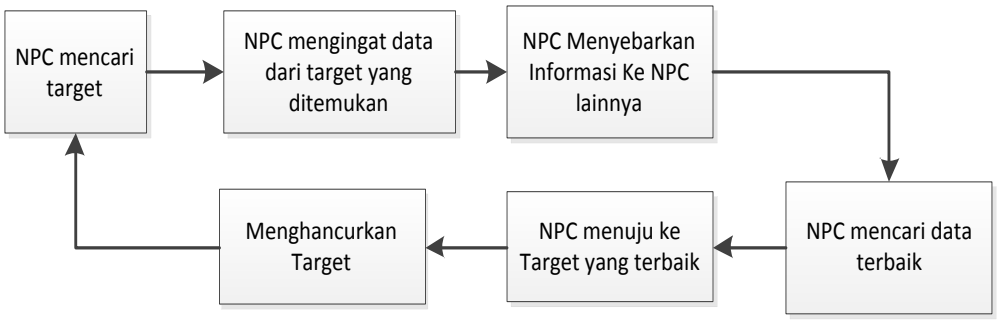
3.2 Bee Colony

Dalam kehidupan nyata, koloni lebah memiliki kerjasama team yang sangat erat. Setiap lebah dalam koloni saling terkoordinasi antara satu sama lainnya dan memiliki tugasnya masing-masing sesuai dengan tipenya. Dalam satu koloni tersebut terdapat lebah yang bekerja mencari sumber makanan, lebah yang bertugas sebagai pengamat, dan juga terdapat lebah yang hanya bekerja untuk mengambil makanan dari sumbernya. Diagram kerja dalam koloni lebah tersebut dapat dilihat dalam diagram kerja lebah pada Gambar 3. 5



Gambar 3. 5 Diagram kerja dalam koloni lebah

Cara kerja lebah tersebut kemudian diadopsi ke dalam permainan. Berdasarkan diagram kerja NPC pada Gambar 3. 6 NPC dianalogikan sebagai lebah dan makanan dianalogikan sebagai musuh yang di cari oleh NPC. NPC menyebar mencari target yang ada di sekitarnya, kemudian NPC tersebut menyimpan data dari target tersebut di rekam dalam memori NPC tersebut kemudian NPC tersebut menyebarkan data dari target yang dia datangi kepada NPC lainnya.



Gambar 3. 6 Diagram kinerja NPC

3.2.1 Pencarian Target (*Scouting*)

- Pengacakan Posisi Awal

Dalam mencari sumber makanan, koloni lebah mengerahkan pasukan pencari (*scout*). Pasukan ini bertugas mencari sumber makanan baru jika sumber makanan yang ada saat ini telah habis. Karena *scout* ini bersifat mencari, maka lokasi target yang sesungguhnya dianggap belum diketahui sehingga lebah *scout* ini melakukan pergerakan secara acak. Lebah pencari ini melakukan pengacakan posisi berikutnya yang akan dituju berdasarkan posisi saat ini sehingga secara matematis dapat dijelaskan seperti pada persamaan 3. 1.

$$X_i' = X_i + rand[-1\ 1] \quad 3.\ 1$$

Karena lebah memiliki jarak maksimum ($d_{maximum}$) untuk melakukan pencarian, maka persamaan tersebut berubah menjadi persamaan 3. 2

$$X_i' = X_i + rand[-1\ 1]\ \times d_{maximum} \quad 3.\ 2$$

Keterangan,

X_i' = Posisi selanjutnya

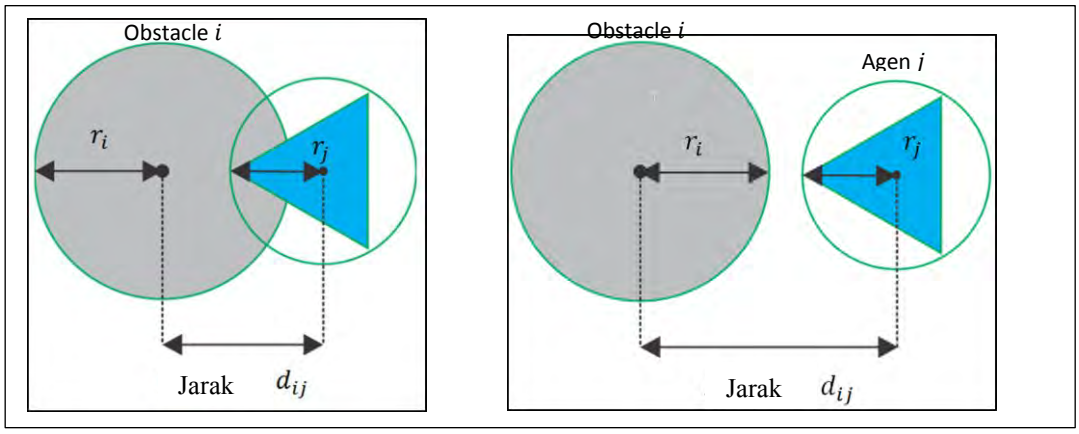
Xi = Posisi saat ini

$d_{maximum}$ = Jarak maksimum lebah melakukan pencarian

Pencarian dibatasi dengan jarak pencarian $d_{maximum}$ karena ibarat lebah yang asli pasti memiliki kapasitas energi untuk melakukan pergerakan. Mustahil untuk lebah pencari melakukan perjalanan ber kilo-kilo tanpa henti.

- Penghindaran *Obstacle*

Saat pencarian, untuk setiap anggota pasukan diharapkan dapat meminimalkan terjadinya tabrakan pada dirinya. Jika posisi yang akan dituju oleh lebah terdapat lebah yang lain ataupun terhalang oleh adanya *obstacle*, maka lebah tersebut harus mencari posisi yang baru lagi. Seekor lebah dikatakan bertabrakan jika jarak posisi lebah tersebut terhadap lebah yang lain maupun halangan lebih kecil daripada jumlah kedua radius (r) objek tersebut. seperti yang terlihat pada Gambar 3. 7



Gambar 3. 7 Ilustrasi saat kedua objek bertabrakan (jarak < ∑radius) dan ketika tidak bertabrakan (jarak > ∑radius).

Sehingga jarak (d_{ij}) minimal lebah terhadap halangan atau lebah yang lain harus sebesar atau lebih besar dari $r_i + r_j$

$$d_{ij} = (r_i + r_j) \quad 3.3$$

$$f_{collision} = d_{ij} - (r_i + r_j) \quad 3.4$$

Keterangan,

d_{ij} = Jarak dari titik pusat agen ke titik pusat *obstacle*

r_i = *Radius collider* dari obstacle i

r_j = *Radius collider* dari agen j

Persamaan 3. 4 tersebut menjadi persyaratan dasar untuk pemilihan posisi untuk persamaan 3. 5. Jika nilai dari $f_{collision}$ kurang dari 0, maka lebah diharuskan melakukan pencarian ulang posisi yang baru sesuai dengan persamaan (3.6).

$$X_i' = f(x) = \begin{cases} X_i + rand[-1\ 1] \times d_{maximum}, & f_{collision} < 0 \\ X_i', & f_{collision} \geq 0 \end{cases} \quad 3.5$$

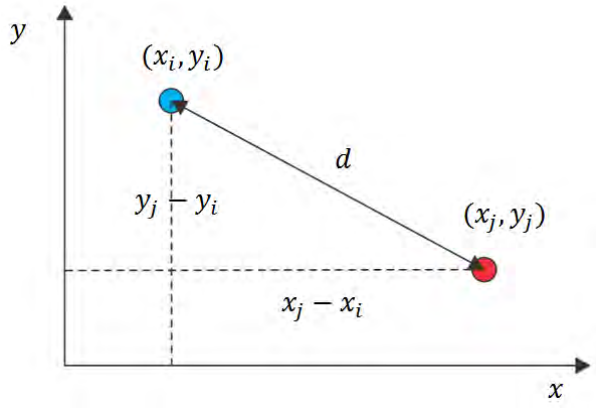
3.2.2 Fase Pengamatan (*Onlooker*)

Lebah pengintai yang telah mendapatkan posisi, kembali ke sarang dan menginformasikan data yang telah dia dapat pada posisi tersebut kepada lebah pengamat yang berada di sarang. Dalam penelitian ini, data yang digunakan yaitu diambil dari parameter yang melekat pada target dan parameter yang melekat pada lebah itu sendiri. Parameter yang diambil dan disebarluaskan ke lebah yang lain yaitu adalah jarak lebah terhadap target saat ini, selisih nyawa antara target dengan lebah, selisih daya serang, dan selisih daya tahan.

- **Pengukuran Jarak**

Dalam penelitian ini pengukuran jarak dari lebah menuju target menggunakan metode *Euclidian distance*. Dengan Euclidian distance, posisi lebah dan target dinormalisasi ke dalam ruang *metric* 3D. Dengan begitu, setiap objek yang ada memiliki nilai vector pada sumbu X,Y, dan Z.

.



Gambar 3. 8 Teorema Phytagoras yang berlaku pada lebah dan target

Dari Gambar 3. 8 tersebut maka untuk menghitung nilai d digunakan persamaan 3. 6.

$$d^2 = (x_j - x_i)^2 + (y_j - y_i)^2 \qquad 3.6$$

$$d = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2} \quad 3.7$$

Keterangan :

d = Jarak

x_i = posisi pada sumbu x agen i

x_j = posisi pada sumbu x agen j

y_i = posisi pada sumbu y agen i

y_j = posisi pada sumbu y agen j

Jarak d tersebut masih dalam koordinat 2D karena lebah bergerak pada koordinat 3D, maka kita harus menambahkan juga koordinat objek pada sumbu z sehingga dari persamaan 3. 8 akan menjadi seperti persamaan

$$d = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2 + (z_j - z_i)^2} \quad 3.8$$

- **Pengukuran tingkat kekuatan musuh**

Target yang menjadi tujuan NPC, terkadang tidak hanya satu target. Akan tetapi bisa dua atau lebih. Target tersebut memiliki kekuatannya masing-masing. Seekor lebah yang mewakili NPC harus dapat memilih target yang optimal untuk dihancurkan. Optimal di sini yaitu yang dapat cepat dihancurkan. Semakin cepat target dihancurkan maka semakin sedikit musuh yang menyerang sehingga tingkat kemungkinan untuk tetap hidup akan semakin tinggi juga. Dalam karakter yang digunakan terdapat parameter yang melekat dalam karakter tersebut. Diantaranya yaitu nyawa (Hp), daya serang (Atk), daya tahan (Def). Jika target yang dituju berupa karakter juga, maka dapat dibandingkan antara agen dan target mana yang lebih kuat dan lemah.

Untuk menghitung seberapa kuat serangan agen terhadap target, maka kita harus mengukur nilai ΔAtk dari jumlah damage yang dikeluarkan agen (Atk_i) kemudian di kurangkan dengan pertahanan yang dimiliki musuh Def_j sehingga dari penjelasan tersebut didapatkan persamaan 3. 9

$$\Delta Atk_{ij} = Atk_i - Def_j \quad 3.9$$

Nilai Negatif dari ΔAtk_{ij} menandakan bahwa target lebih kuat dari agen dan sebaliknya jika ΔAtk_{ij} bernilai positif maka agen lebih kuat dari target. Karena nilai ΔAtk_{ij} yang diinginkan adalah nilai positif yang menandakanagen lebih kuat, maka nilai pertahanan (Def) target yang digunakan untuk persamaan hanya separuhnya. Sehingga persamaan 3. 9 berubah menjadi persamaan 3. 10

$$\Delta Atk_{ij} = Atk_i - \frac{1}{2} Def_j \quad 3.\ 10$$

Keterangan,

ΔAtk_{ij} = daya serang agen i terhadap musuh j

Atk_i = nilai serangan agen i

Def_j = daya tahan dari musuh j

Kemudian setelah mendapat kan nilai damage yang didapatkan, harus diukur seberapa cepat musuh dapat dihancurkan dengan cara membandingkan nyawa (Hp) dari kedua objek sehingga didapatkan persamaan 3. 11.

$$Kr_{ij} = \frac{Hp_j}{(\Delta Atk_{ij})} \quad 3.\ 11$$

Keterangan,

Kr_{ij} = kecepatan membunuh agen i terhadap musuh j

Hp_j = nilai parameter nyawa musuh j

ΔAtk_{ij} = daya serang agen i terhadap musuh j

Sehingga jika persamaan 3. 10 disubsitusikan dengan persamaan 3. 11, maka kita akan mendapatkan persamaan 3. 12.

$$Kr_{ij} = \frac{Hp_j}{(Atk_i - \frac{1}{2} Def_j)} \quad 3.\ 12$$

Karena target tidak hanya berjumlah satu, maka *DestroyTime* (Kr) harus dihitung untuk setiap target dan dicari yang paling maksimum

$$\Delta Kr_{ij} = Kr_i - Kr_j \quad 3.\ 13$$

$$Kr_{best} = \min_{1-n}(\Delta Kr_{in}) \quad 3.\ 14$$

Maka, NPC akan menyerang target dengan Kr yang terbaik dimana akan membuatnya menyerang target yang mudah dihancurkan dan dapat bertahan lebih lama.

- **Penentuan fungsi fitness**

Setelah pengamat mendapat data yang diperlukan maka pengamat menghitung nilai fitness (fit) untuk menentukan kemana lebah pekerja harus mengambil makanan. fit_{jarak} didapat dari persamaan *Euclidian* yaitu persamaan 3. 8. Persamaan ini di normalisasi supaya rentang nilainya tidak terpaut terlalu jauh dengan nilai fitness yang lainnya. $fit_{strength}$ merupakan nilai fitness untuk kekuatan lawan diambil dari data Kr yang terbaik. fit_{obs} diambil dari nilai $fit_{collision}$ yang didapat dengan menggunakan Persamaan 3. 4. Fungsi ini untuk mendeteksi adanya tabrakan pada langkah selanjutnya. Dari ketiga persamaan tersebut, didapatkan nilai fitness yaitu fit_{total} dengan cara menjumlahkan ketiga nilai fitness yang ada.

$$fit_{jarak} = \frac{\sqrt{(x_j - x_i)^2 + (y_j - y_i)^2 + (z_j - z_i)^2}}{R_i} \quad 3.\ 15$$

$$fit_{strength} = Kr_{best} = \min_{1-n}(\Delta Kr_{in}) \quad 3.\ 16$$

$$fit_{obs} = d_{ij} - (r_i + r_j) \quad 3.\ 17$$

$$fit_{total} = fit_{jarak} + fit_{strength} + fit_{obs} \quad 3.\ 18$$

Ket:

fit_{dist} Didapatkan dari rumus *Euclidian distance*. nilai fitness ini di normalisasikan sehingga rentang nilainya tidak terlalu jauh dari nilai fitness lainnya.

$fit_{strength}$ merupakan nilai kekuatan dari target yang ada. Nilai ini digunakan untuk memilih target yang akan diserang.

fit_{obs} merupakan nilai fitness yang digunakan untuk agen agar dapat menghindari *obstacle*

3.2.3 Fase Pekerja (Employed)

Ketika lebah pencari telah menginformasikan data yang telah diperolehnya kepada lebah pengamat, dan lebah pengamat sudah menganalisa data yang diterima dari lebah pengintai maka informasi yang telah diolah lebah pengamat diteruskan kepada lebah pekerja. Oleh lebah pekerja, data tersebut menjadi acuan untuk bergerak mengambil makanan yang berada pada target. Setiap kali lebah pekerja kembali ke sarang dengan membawa makanan, lebah pekerja tersebut menginformasikan kepada pengamat bahwa makanan yang tersedia pada target telah berkurang.

$$n = \begin{cases} X_i + rand[-1\ 1]\ \times d_{maximum}, & n < 0 \\ n - 1, & n \geq 0 \end{cases} \qquad 3.\ 19$$

n = jumlah nectar / makanan saat ini

Jika nectar (n) pada target telah habis, maka koloni mengerahkan kembali lebah pencari untuk memperbaharui posisi sumber makanan dengan menggunakan persamaan 3. 2.

BAB IV

HASIL PENGUJIAN

Hal pertama yang dilakukan adalah menguji algoritma *bee colony* dari penelitian sebelumnya. Algoritma dari penelitian sebelumnya yang menjadi uji coba adalah algoritma milik Wilujeng Jatiningsih (Jatiningsih dkk., 2014), karena Wilujeng juga menerapkan algoritma *bee colony* tersebut dalam sebuah permainan sehingga masih ada keterkaitan antara penelitian Wilujeng dengan penelitian ini.

4.1 Pengujian metode sebelumnya

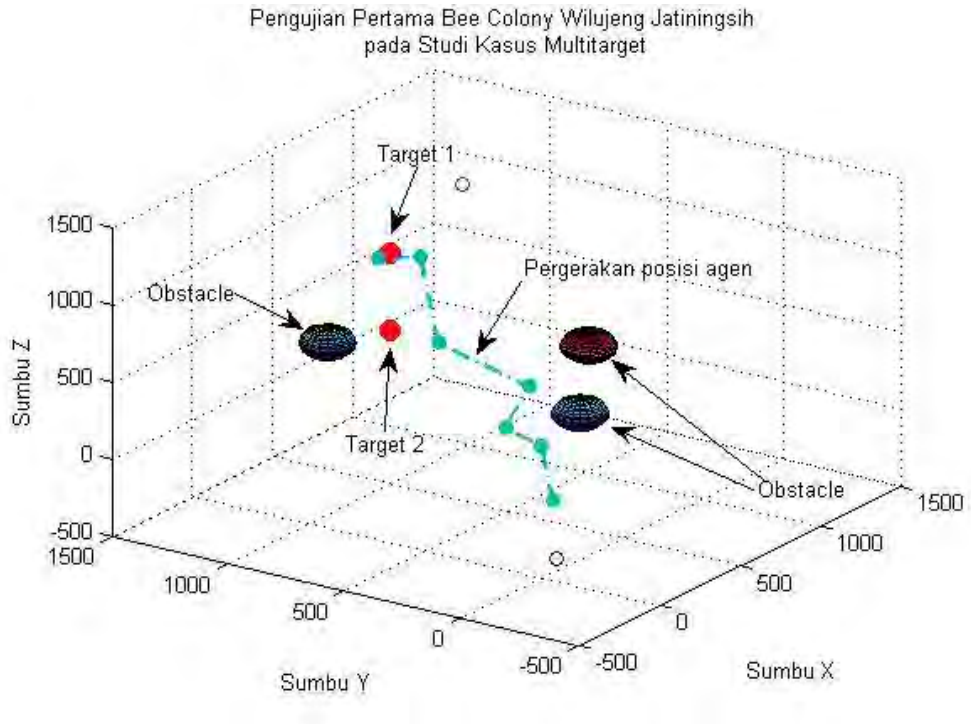
Pada algoritma milik Wilujeng, agen tidak memiliki cukup banyak parameter yang bisa dimanfaatkan. Terdapat beberapa parameter kontrol yang dapat diubah sesuai selera sehingga bisa kita bisa mengatur skenario yang kita inginkan parameter tersebut diantaranya jumlah agen, besar agen, jumlah halangan, ukuran halangan, dan berapa banyak perulangan dapat dilakukan akan tetapi tidak banyak parameter yang melekat agen yang bergerak. Agen hanya dibekali dengan sebuah parameter jarak terhadap target yang mana artinya agen hanya bergerak menuju target yang memiliki jarak yang optimal terhadap agen tersebut saja. Sedangkan pada algoritma yang sedang diteliti ini telah ditambahkan beberapa parameter baru seperti daya serang, nyawa, pertahanan dan lain-lain yang dapat diatur sehingga bisa dimanfaatkan untuk membuat fungsi fitsess yang lebih baik. Perbandingan parameter kontrol dapat dilihat pada Tabel 4. 1.

Tabel 4. 1 Perbandingan parameter yang digunakan pada penelitian sebelumnya dan penelitian yang sedang dilakukan

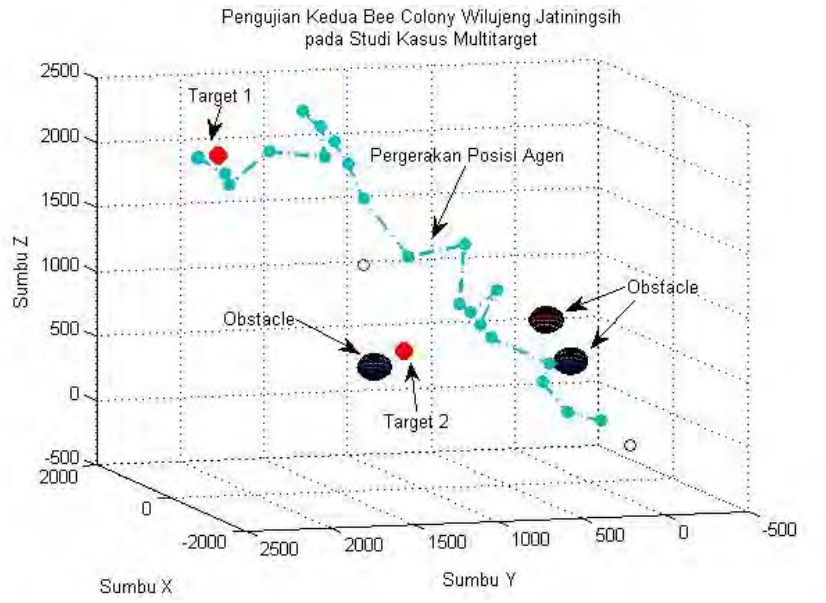
Parameter	Penelitian sebelumnya	Penelitian saat ini
Jumlah Iterasi yang dilakukan	✓	✓
Jumlah Agen	✓	✓
Posisi Agen	✓	✓
Ukuran Agen	✓	✓

Parameter	Penelitian sebelumnya	Penelitian saat ini
Daya Serang Agen	-	✓
Daya Tahan Agen	-	✓
Nyawa Agen		✓
Jumlah Obstacle	✓	✓
Posisi Obstacle	✓	✓
Jumlah Target	-	✓
Posisi Target	✓	✓
Daya Serang Target	-	✓
Daya Tahan Target	-	✓
Nyawa Target	-	✓

Dengan hanya menggunakan parameter letak agen, maka algoritma *bee colony* milik Wilujeng hanya sebatas menyerang target tanpa memperhatikan bahwa target yang diserang itu optimal atau tidak. Hal itu dibuktikan dengan hasil pengujian algoritma milik Wilujeng dengan penambahan beberapa target.



Gambar 4. 1 Hasil pengujian pertama dari algoritma terdahulu dengan penambahan target



Gambar 4. 2 Hasil pengujian kedua dari algoritma terdahulu dengan penambahan target

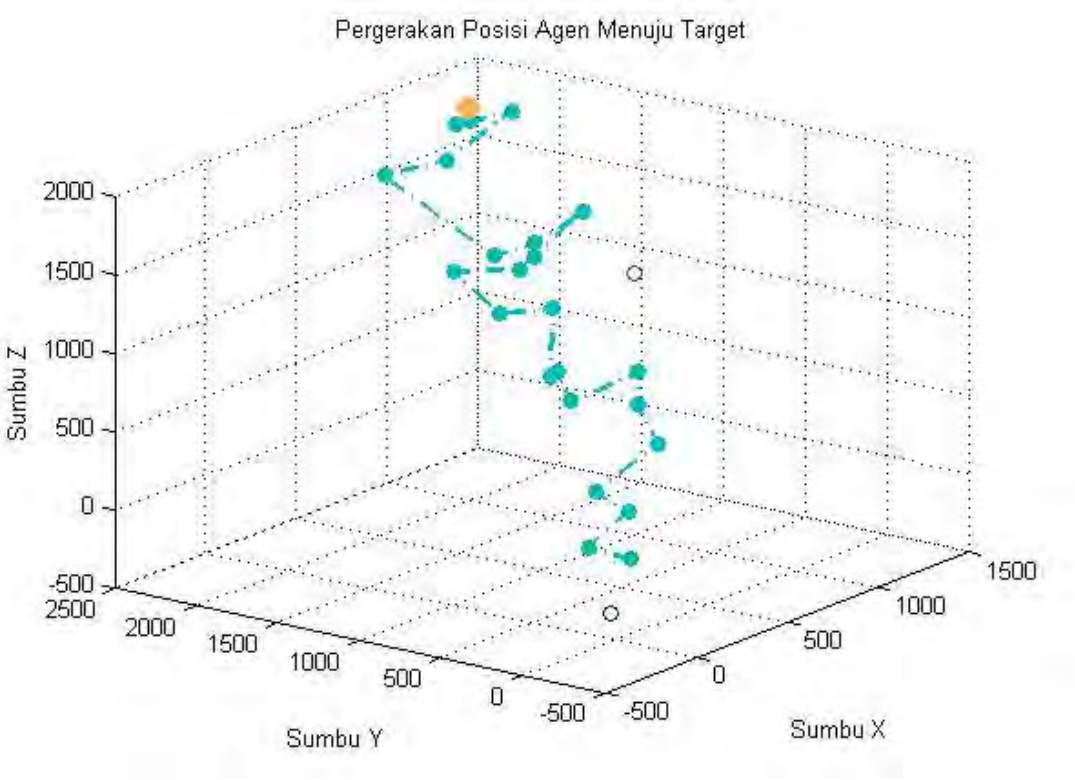
Dua sample hasil pengujian pada Gambar 4. 1 dan Gambar 4. 2 membuktikan bahwa dengan penambahan target dalam prosesnya, algoritma terdahulu belum dapat memilih target yang tepat untuk agen. Agen tetap memilih mengejar target pertama walaupun terdapat target yang lebih dekat di sekitarnya.

Selanjutnya pengujian dilakukan beberapa kali dengan bermacam-macam skenario kondisi pengujian seperti pada Tabel 4. 2.

Tabel 4. 2 Skenario pengujian algoritma

Skenario No.	Jumlah Bee	Sifat Target	Obstacle
1	1	Tidak	Tidak
2	5	Tidak	Tidak
3	1	Bergerak	Tidak
4	5	Bergerak	Tidak
5	1	Tidak	Ya
6	5	Tidak	Ya
7	1	Bergerak	Ya
8	5	Bergerak	Ya

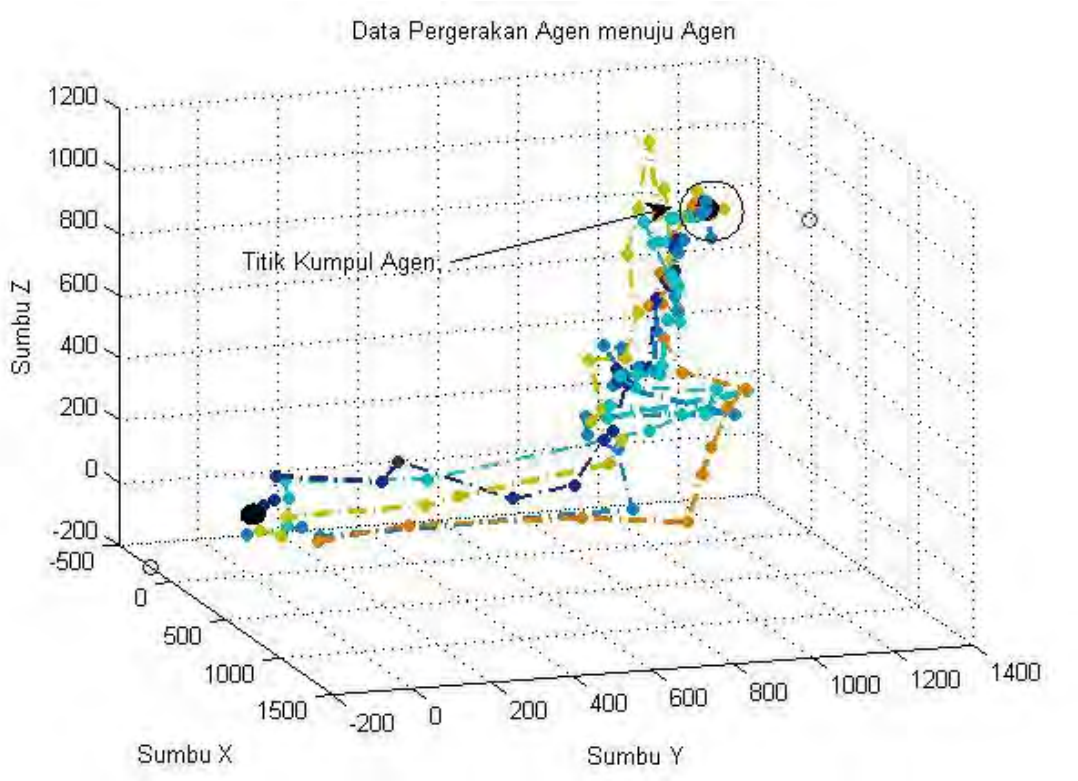
Hasil dari pengujian dari sekenario 1 dan 2, algortitma berjalan lancar sebagaimana mestinya. Agen bergerak dari posisi kemunculannya yang acak, kemudian bergerak tak beraturan menuju target.



Gambar 4. 3 Skenario 1 agen bergerak tak beraturan menuju target seperti lebah yang sedang mencari makanan

Pergerakan agen yang tak beraturan dan tak dapat ditebak seperti yang terlihat pada Gambar 4. 3 mengindikasikan bahwa posisi target belum diketahui oleh agen sehingga agen berusaha mencarinya . Tidak ada masalah yang terjadi pada pengujian dengan skenatio 1. Hasil yang didapatkan ckup memuaskan. Fungsi fitness yang membuat agen bergerak menuju target dapat bekerja sebagai semestinya sehingga agen dapat mencapai target dalam pengujian pertaman ini. Akan tetapi pada skenario 2, pada saat menggunakan agen lebih dari 1 terdapat permasalahan saat agen mencapai target.

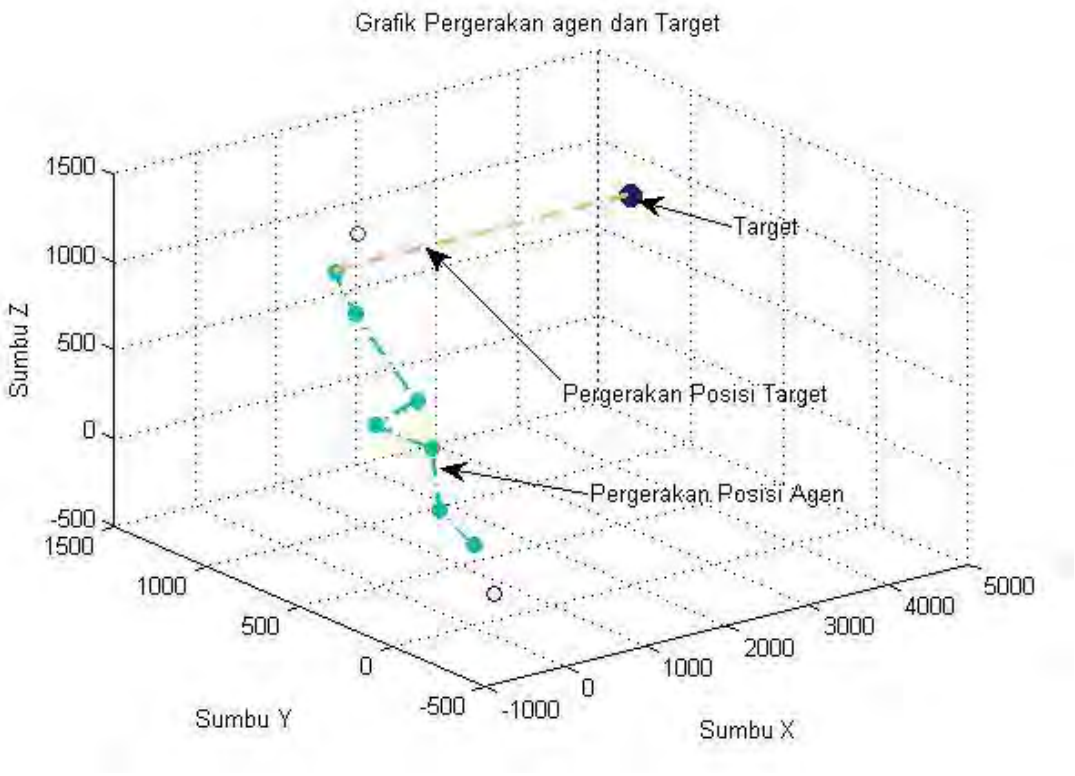
Para agen awalnya muncul pada koordinat yang acak. Kemudian agen-agen tersebut beergerak tak beraturan. Sama seperti pengujian skenario 1, sampai proses ini algoritma menjalankan tugasnya dengan cukup baik. Tetapi pada saat agen-agen mencapai target yang dituju, agen-agen menunjukan tanda-tanda adanya permasalahan.



Gambar 4. 4 Skenario 2 menggunkan agen lebih dari satu. Agen berkerumun pada target

Fungsi yang seharusnya menjaga agar para agen tidak bertabrakan satu sama lainnya tidak berjalan semestinya sehingga agen-agen yang ada mulai bertabrakan satu sama lainnya. Kondisi tersebut membuat para agen berkumpul menjadi satu titik saja seperti yang terlihat pada Gambar 4. 4. Permasalahan ini tidak bisa di tolerir mengingat dalam permainan tabrakan maupun tumpukan antara agen dapat dianggap sebagai *bug* dalam game. Selain permasalahan tersebut, ternyata terdapat juga

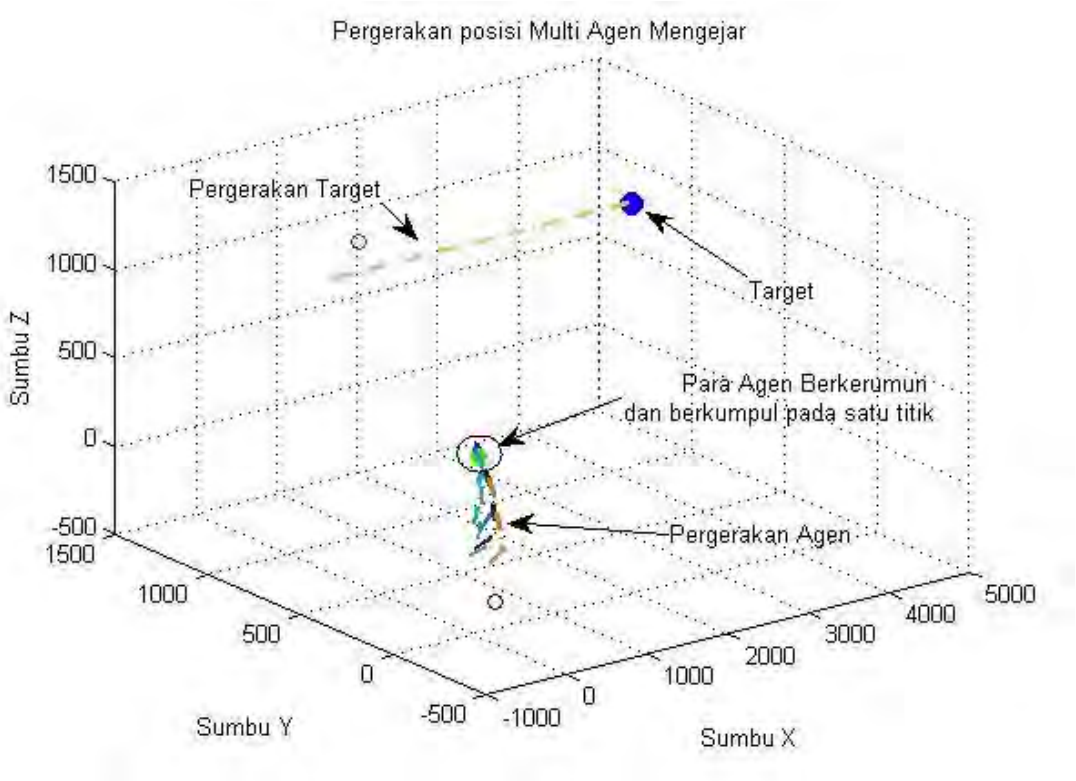
permasalahan yang muncul ketika target berpindah tempat. Pada skenario pengujian ke-3, target dibuat berpindah tempat setiap waktunya. Saat target bergerak, agen tetap bergerak menuju posisi awal target. Hal ini mengindikasikan bahwa agen tidak melakukan *update* posisi target sehingga posisi yang dijadikan tujuan adalah posisi awal target saja seperti yang ditunjukan pada Gambar 4. 5.



Gambar 4. 5 Agen bergerak menuju target tetapi agen tidak dapat mengikuti pergerakan yang diakukan target

Jika agen tidak bisa mengikuti pergerakan agen, maka algoritma tersebut masih belum bisa diterapkan secara penuh dalam game, karena dalam game target bisa saja berganti-ganti maupun berpindah tempat. Jika target yang dituju merupakan karakter, target tesebut dapat bergerak sehingga algoritma yang diterapkan harus mampu mengimbangi pergerakan target tersebut. Pada skenario pengujian ke-4 para agen bukan hanya tidak dapat mengikuti pergerakan target, para agen juga tidak ada yang

berhasil mencapai titik awal target sepert yang terlihat pada Gambar 4. 6. Tidak banyak pergerakan yang dilakukan oleh agen. Mereka hanya berkumpul di sebuah titik dan tidak bergerak menuju target yang ada. Jika permasalahan ini terjadi dalam sebuah game, ini akan menjadi permasalahan yang serius karena hal ini menyangkut pada *gameplay* sebuah permainan.

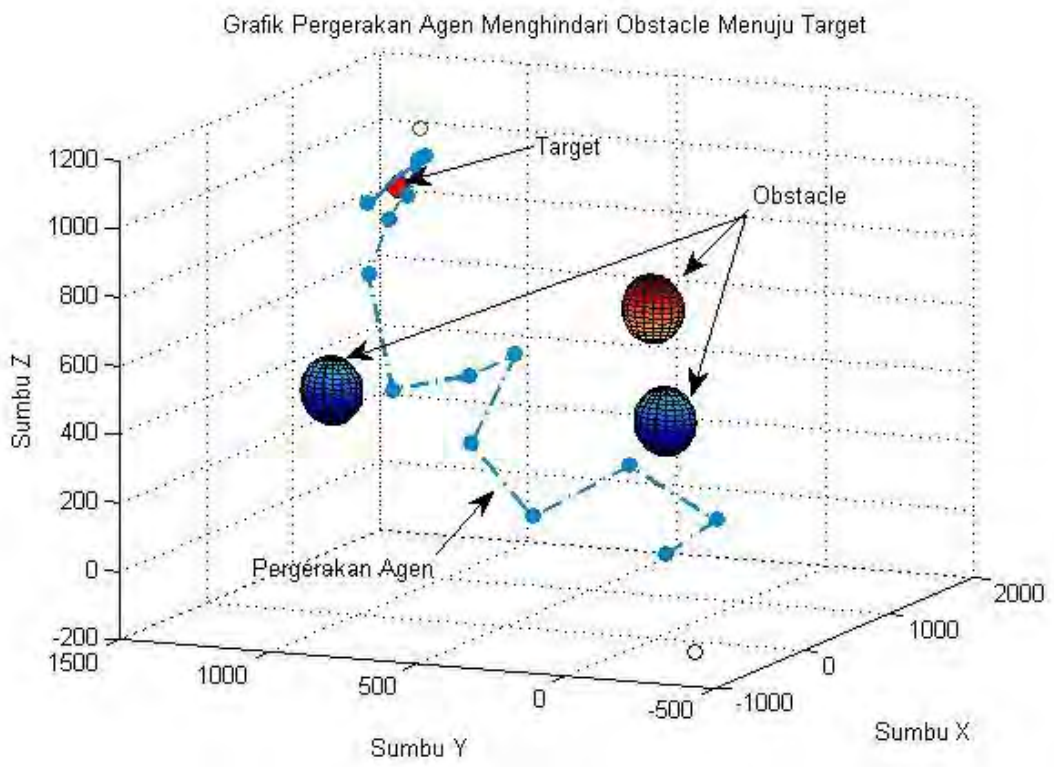


Gambar 4. 6 Pengujian dengan skenario nomor 4, agen hanya berkerumun dan tidak ada yang dapat mencapai target

Pengujian dengan skenario 5 dan 6 dilakukan untuk mengtahui apakah algoritma yang dibuat sudah dapat menghindari obstacle yang ada pada ruang permainan atau belum. Uji coba skenario 5 dilakukan dengan jumlah agen hanya 1 dan target dibuat tetap pada koordinat (650,650,650). Kemudian terdapat 3 obstacle yang diletakan pada koordinat (500, 150, 700) , (300, 50, 400), dan (400, 1200, 400). Hasil

yang didapatkan dari pengujian sekenario 5 pada algoritma yang ada, dapat dilihat pada Gambar 4. 7.

Para agen awalnya bergerak tak beraturan menuju target. Pergerakan agen tersebut selain acak, para agen juga bisa menghindari *obstacle* yang telah disiapkan. Akan tetapi belum sampai pada target, para agen tidak melanjutkan pergerakannya dan berkumpul menjadi satu. Sama halnya seperti skenario ke dua, para agen berkumpul tanpa menghiraukan tabrakan yang terjadi pada agen tersebut.

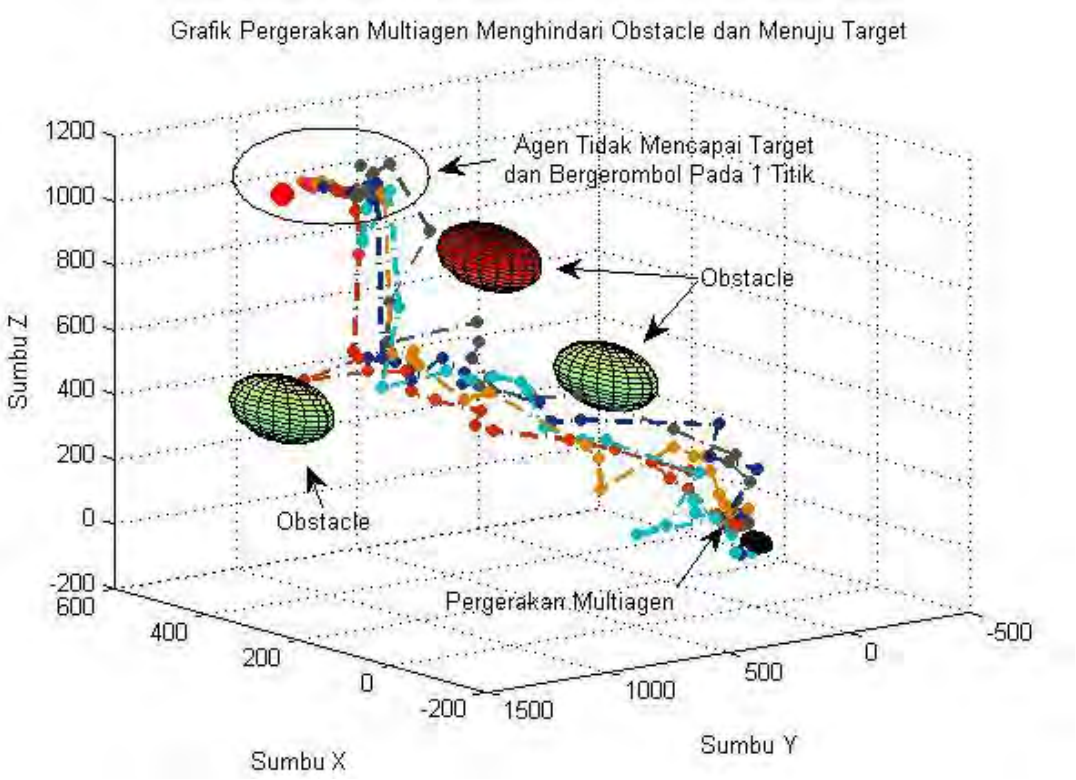


Gambar 4. 7 Hasil pengujian dengan skenario 5

Pada Skenario 6, agen yang dipakai berjumlah 5 agen. Pada proses pengujian agen muncul pada koordinat acak. Kemudian para agen tersebut bergerak tak beraturan. Pergerakan agen tersebut walaupun tak beraturan akan tetapi tetap konvergen menuju target. Akan tetapi setelah mendekati target, sama seperti pengujian seelumnya, agen berkumpul menjadi satu dalam sebuah titik. Dan agen-agen tersebut

saling bertabrakan satu sama lain. Tetapi walaupun begitu dilihat dari rute pergerakan yang dihasilkan para agen pada Gambar 4. 8, agen dapat menghindari obstacle yang telah disediakan.

Setelah mengetahui kelemahan dari penelitian sebelumnya, maka untuk algoritma *bee colony* yang diteliti ini dilakukan pengujian untuk mengetahui apakah algoritma yang diteliti saat ini dapat menutupi kelemahan yang terdapat pada penelitian sebelumnya.



Gambar 4. 8 Pengujian skenario 6 pada algoritma sebelumnya menunjukan agen tidak dapat mencapai target

4.2 Pengujian metode yang diteliti

Pengujian dilakukan menggunakan 10 buah agen, yang bergerak menuju beberapa target yang memiliki karakteristik yang berbeda-beda. Setiap agen memiliki radius sebesar 20 px. Pegujian dilakukan dengan skenario yang sama seperti skenario

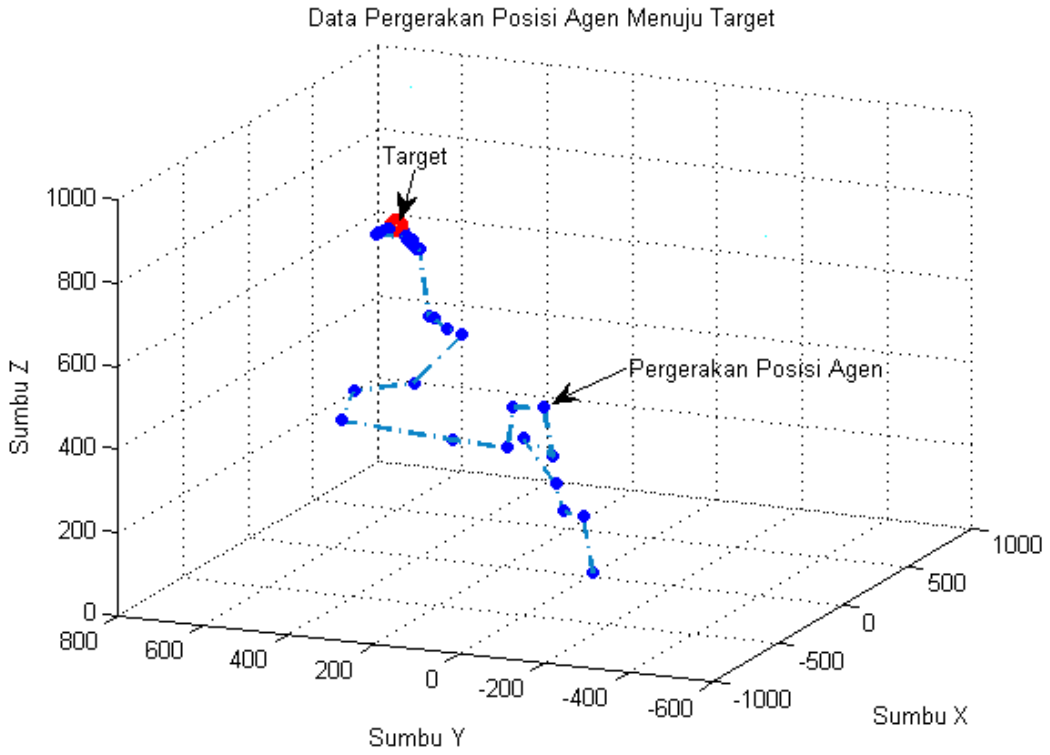
yang diuji cobakan pada algoritma sebelumnya dan parameter yang digunakan seperti Tabel 4. 3 . Sedangkan posisi awal agen di pilih secara acak dalam radius 200px. Posisi pengacakan awal ini didasarkan pada fase scouting *bee colony*. Dimana koloni agen dianggap menyebarkan lebah pengintai kesegala arah untuk mencari sumber makanan baru.

Tabel 4. 3 Data Parameter Agen Lebah

ID Agen	Health Power	Attack	Defense
1	100	11	6
2	95	12	7
3	90	13	8
4	85	14	9
5	80	15	10
6	75	16	6
7	70	17	7
8	65	18	8
9	60	19	9
10	55	20	10

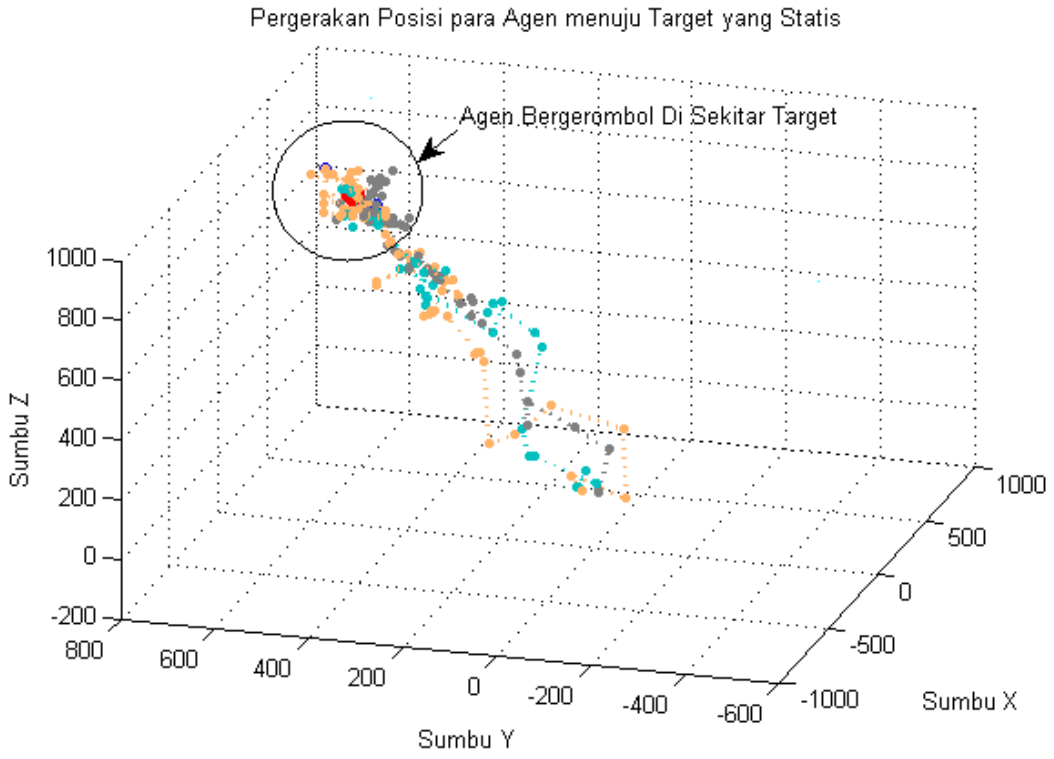
Pada uji coba pertama, jumlah agen lebah yang digunakan hanya satu agen lebah dengan posisi target yang telah ditentukan pada koordinat (650,650,650). Agen muncul pada koordinat (2, -12, 10). Kemudian seperti halnya pada algoritma sebelumnya, agen mulai bergerak menuju target secara tak beraturan. Agen mencari-cari jalan yang cocok sesuai dengan fungsi fitnessnya.

Sama seperti lebah pencari yang mencari makanan berdasarkan indera penciumannya sehingga rute yang dihasilkan bukan merupakan garis lurus akan tetapi jalur yang berbelok-belok seperti yang ditampilkan pada Gambar 4. 9. Hasil tersebut berarti bahwa fungsi fitness untuk pencarian target berjalan dengan baik. Pada uji coba pertama ini juga dapat dilihat bahwa saat mencapai target, agen tidak hanya sekedar mencapainya akan tetapi terbang mengitari target tersebut. Berbeda dengan algoritma sebelumnya dimana agen jika telah mencapai target, maka akan berhenti.



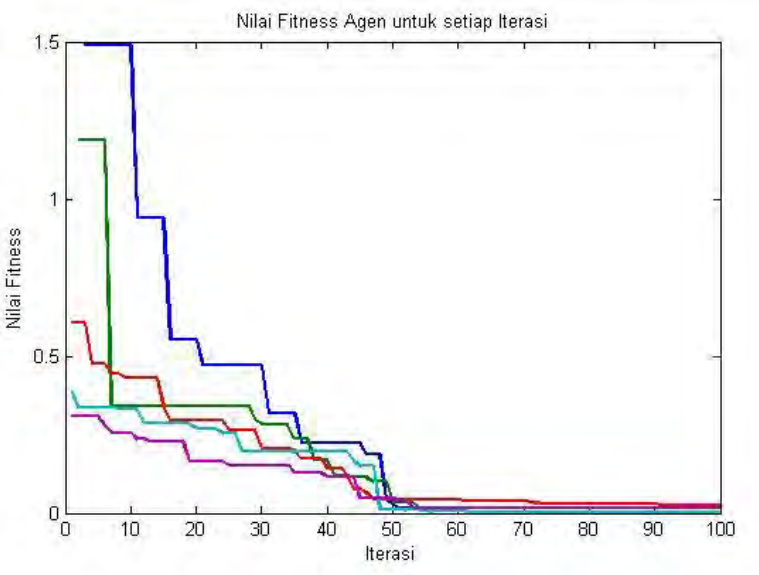
Gambar 4. 9 Hasil pengujian dengan skenario ke-1 pada algoritma yang di teliti.

Kemudian pengujian dilanjutkan untuk skenario ke-2 dimana pada skenario ini agen yang digunakan berjumlah lebih dari satu dengan posisi target tetap pada koordinat (650, 650, 650). Jika pada penelitian sebelumnya agen dapat bergerak menuju target tetapi pada akhirnya agen berkerumun menjadi 1dan terjadi tabrakan antar agen maka pada algoritma baru ini hal tersebut tidak terjadi. Pada algoritma yang baru ini, agen dapat bergerak menuju target, kemudian pada saat berada pada target, para agen mengitarinya dan tetap menjaga jarak antar agen. Sehingga tabrakan pada agen dapat di minimalisir.



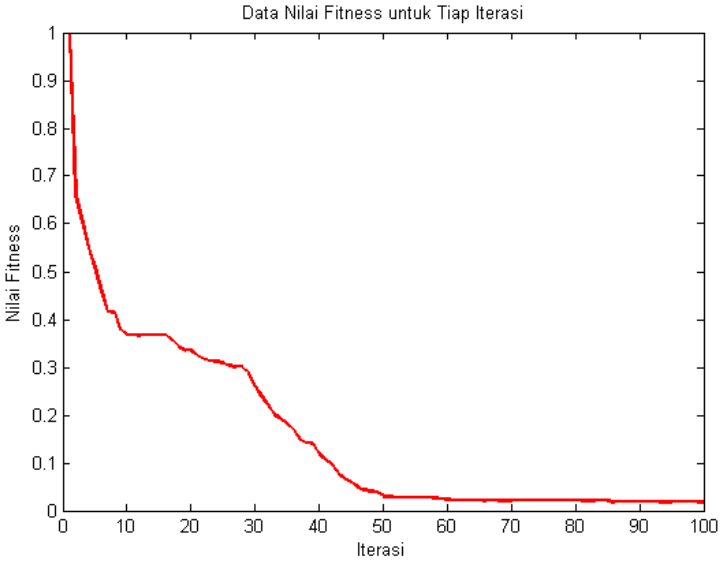
Gambar 4. 10 Rute agen dari *spawn* menuju target (a). Algoritma *bee colony* membuat rute yang berbeda-beda menuju target untuk setiap agennya.

Berdasarkan dari Gambar 4. 10, para agen muncul secara acak di sekitar koordinat (0,0, 0). Kemudian, masing-masing dari agen bergerak secara tak beraturan mendekati target dan ketika telah dekat dengan target mereka berputar putar mengelilingi target tersebut. Dari pengujian pertama ini, didapatkan rata-rata fitness yang semakin menurun setiap iterasinya seperti yang tertera pada grafik pada Gambar 4. 11.



Gambar 4. 11 Grafik nilai fitness tiap agen

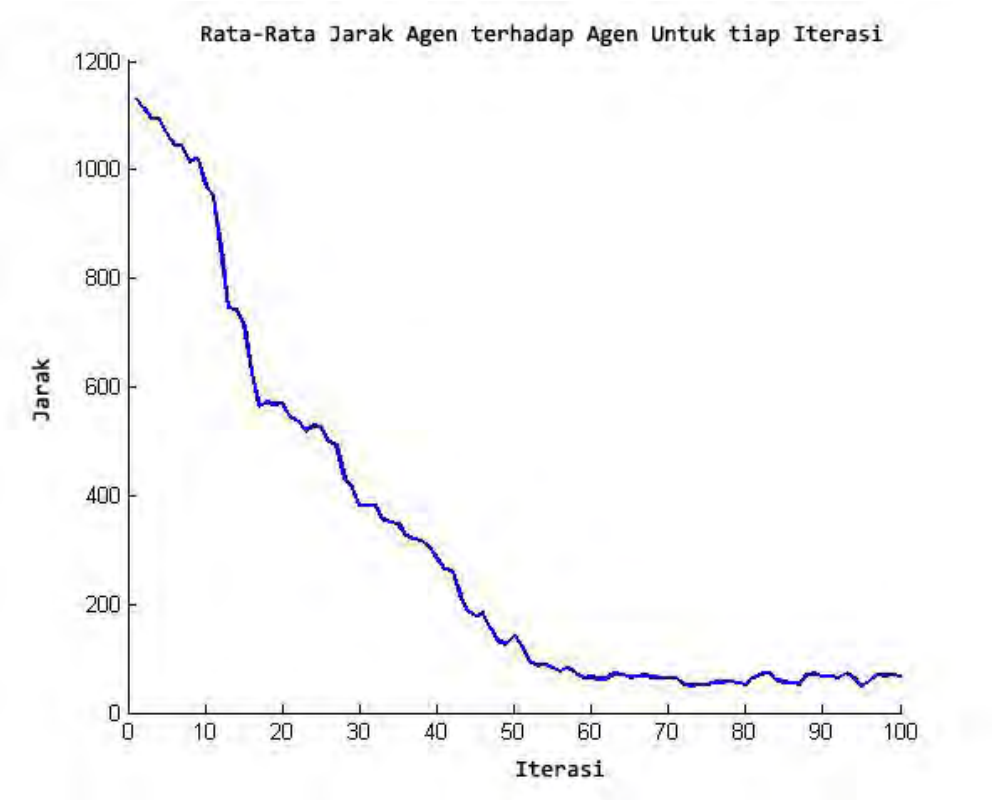
Karena dalam fungsi fitness yang dipakai terdapat fungsi fitness jarak dimana semakin kecil jarak menunjukkan agen mendekati target, maka dengan menurunnya grafik fitness seperti pada Gambar 4. 11 mengindikasikan bahwa agen telah dapat bergerak menuju targetnya.



Gambar 4. 12 Rata-rata nilai fitness agen

Dan jika dilihat dari data rata-rata nilai fitness keseluruhan agen pada Gambar 4. 12, diketahui bahwa pada sekitar iterasi ke-50, para agen telah mendapatkan fitness yang paling rendah dimana artinya pada saat itu pula, agen telah mencapai targetnya.

Pernyataan tersebut didukung dengan grafik jarak agen terhadap target yang ada pada Gambar 4. 13. pada grafik tersebut diketahui bahwa jarak agen menuju target semakin lama semakin berkurang yang artinya agen pada setiap iterasinya semakin mendekat ke posisi target.



Gambar 4. 13 Grafik jarak antara agen dengan target yang dituju

Selain itu setelah dibandingkan dengan metode sebelumnya, dari 20 kali pengujian menggunakan posisi target yang sama, percobaan dengan studi kasus multi agen ini menghasilkan Tabel 4. 4

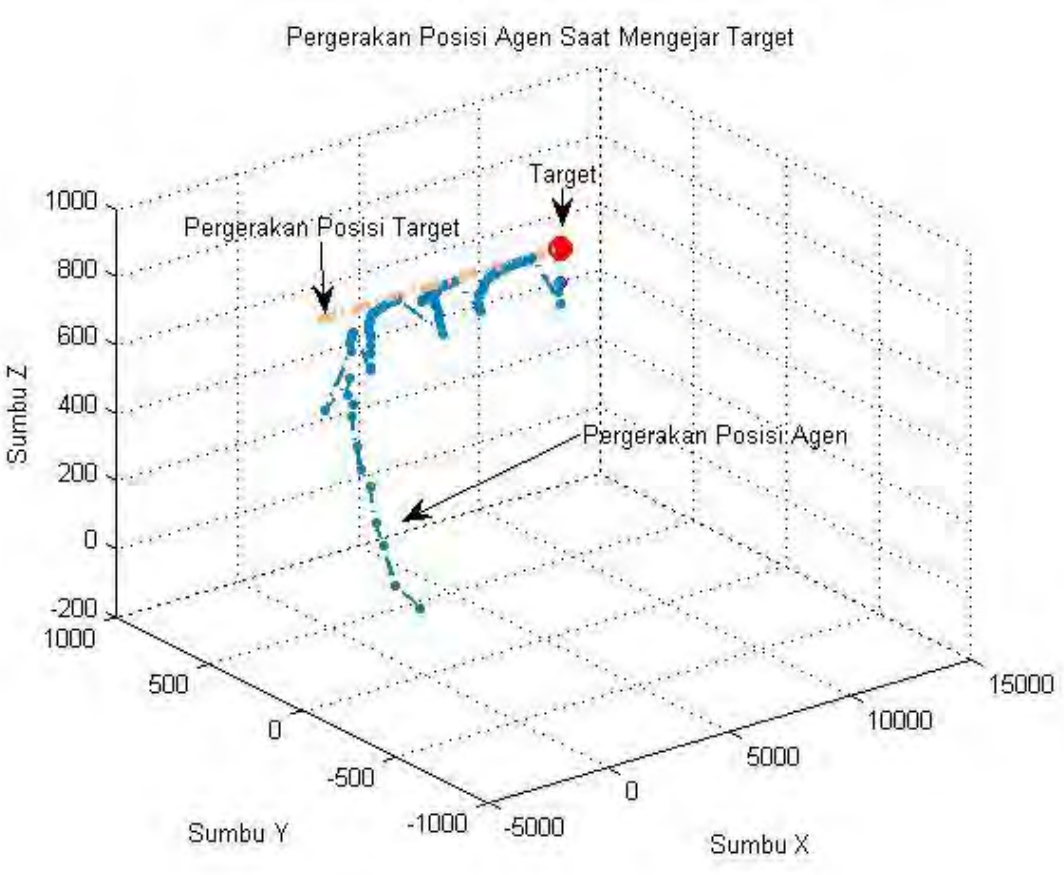
Tabel 4. 4 Tabel tingkat keberhasilan metode

Pengujian ke-n	**Metode Sebelumnya**	**Metode yang diteliti**
1	Berhasil	Berhasil
2	Berhasil	Berhasil
3	Berhasil	Berhasil
4	Tidak Berhasil	Berhasil
5	Berhasil	Berhasil
6	Berhasil	Berhasil
7	Berhasil	Berhasil
8	Berhasil	Berhasil
9	Tidak Berhasil	Tidak Berhasil
10	Tidak Berhasil	Berhasil
11	Berhasil	Berhasil
12	Tidak Berhasil	Berhasil
13	Berhasil	Berhasil
14	Tidak Berhasil	Berhasil
15	Berhasil	Berhasil
16	Tidak Berhasil	Berhasil
17	Tidak Berhasil	Berhasil
18	Berhasil	Berhasil
19	Berhasil	Berhasil
20	Tidak Berhasil	Berhasil

Dari data Tabel 4. 4 tersebut diketahui bahwa metode sebelumnya, agen hanya berhasil mencapai target 12 dari 20 kali pengujian atau bisa dikatakan tingkat keberhasilannya hanya 60% sedangkan dengan metode yang diteliti tingkat keberhasilannya dapat mencapai 95%.

Kemudian pada skenario ke-3 pengujian dilakukan untuk menguji bagaimana jika target yang menjadi tujuan agen, posisinya tidak tetap akan tetapi target

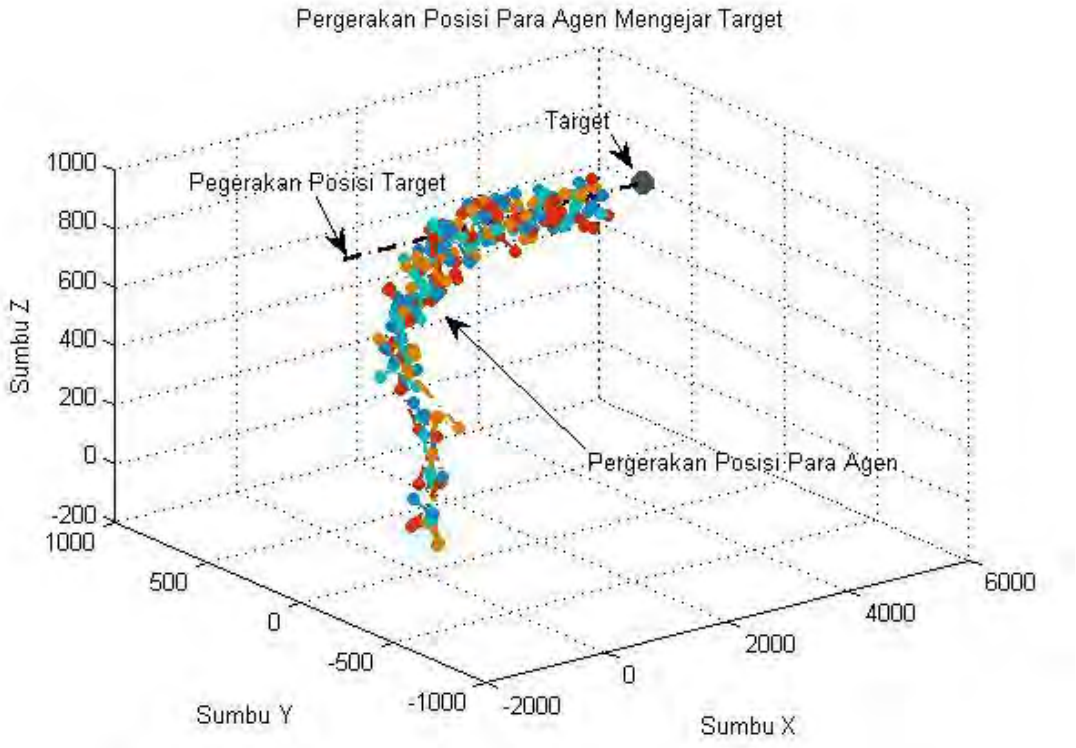
merupakan objek yang dapat bergerak secara bebas. Target muncul pada koordinat (650, 650, 650) kemudian pada tiap iterasinya, target bergerak pada sumbu x-nya sejauh 50px. Hanya 1 agen saja yang digunakan pada skenario ini. Pada algoritma sebelumnya, agen belum berhasil mencapai target dan agen hanya bergerak menuju koordinat munculnya target saja. Pada algoritma yang diteliti ini, rute pergerakan agen menggunakan skenario ke-3 ini ditunjukan dengan Gambar 4. 14.



Gambar 4. 14 Rute pergerakan agen menuju target yang bergerak

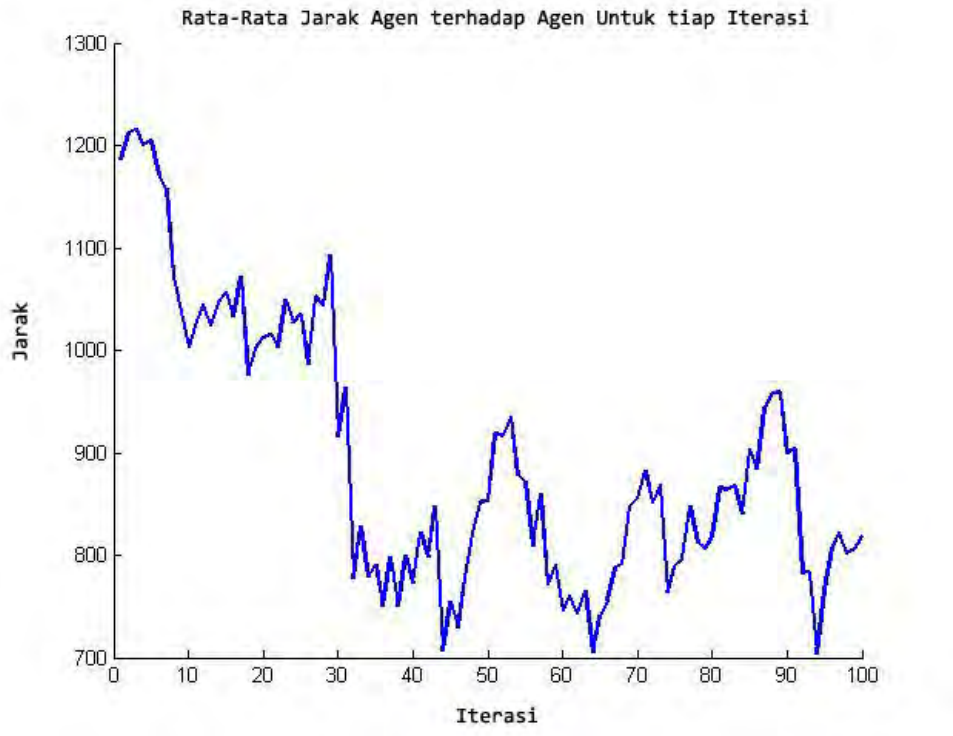
Pada gambar tersebut, terlihat agen muncul pada koordinat disekitar (0, 0, 0). Kemudian secara perlahan-lahan agen bergerak mengikuti kemana target berpindah tempat. Jika dilihat rutenya, walaupun mengekor target, tetapi agen juga masih tetap melakukan pergerakan acak. Kemudian pengujian dilanjutkan ke skenario ke-4 dimana

jumlah agen yang bergerak ditambah. Hal ini untuk mengetahui apakah dengan penambahan agen ini masih terdapat *error* dalam prosesnya dan agen masih tetap berkumpul seperti yang terjadi pada algoritma sebelumnya. Agen yang digunakan kali ini berjumlah lima agen sesuai skenario yang ditunjukan pada Tabel 4. 2. hasil dari pengujian skenario ke 4 ini, dapat dilihat pada Gambar 4. 15. Pada gambar tersebut, agen yang ada bergerak tak beraturan. Akan tetapi walaupun pergerakannya tidak beraturan, ke 5 agen tersebut sama-sama menuju target dan ketika target berpindah ke 5 agen tersebut dapat mengimbangi pergerakan dari target tersebut.



Gambar 4. 15 Rute pergerakan multi agen menuju target yang bergerak

Dilihat dari grafik rata-rata jarak agen terhadap target pada Gambar 4. 16 grafik yang didapat berupa grafik gelombang naik turun. Berbeda dengan grafik rata-rata jarak pada skenario 1 dan 2 yang cenderung stabil seperti pada Gambar 4. 13.



Gambar 4. 16 Rata-rata jarak agen terhadap target pada skenario ke-4

Terjadinya gelombang naik turun berarti jarak agen dan target terkadang menjauh kemudian mendekat lagi. Saat target bergerak menjauh, terjadi peningkatan jarak sehingga menimbulkan gelombang naik pada grafik. Kemudian, mengetahui target menjauh, para agen berusaha mencari posisi yang lebih dekat dengan target lagi. Sehingga jarak menuju target berkurang yang akhirnya menimbulkan gelombang turun pada grafik. Ampitudo tertinggi terletak di awal iterasi karena pada awal iterasi merupakan jarak terjauh antara target dengan agen kemuddian amplitude gemlombang berikutnya lebih pendek dkarenakan pada saat itu agen sudah mendekat menuju target. Dari hasil pengujian tersebut didapatkan kesimpulan bahwa pada algoritma yang diteliti ini, sudah dapat memperbaiki kelemahan dari algoritma sebelumnya yang masih belum bisa mengikuti target yang bergerak.

Pengujian dengan target bergerak ini juga dibandingkan dengan metode sebelumnya. Dari 20 kali pengujian, didapatkan data seperti pada

Tabel 4. 5 Tabel tingkat keberhasilan metode untuk studi kasus target yang bergerak

Pengujian ke-n	**Metode Sebelumnya**	**Metode yang diteliti**
1	Tidak Berhasil	Berhasil
2	Tidak Berhasil	Berhasil
3	Tidak Berhasil	Berhasil
4	Tidak Berhasil	Berhasil
5	Tidak Berhasil	Berhasil
6	Tidak Berhasil	Berhasil
7	Berhasil	Berhasil
8	Tidak Berhasil	Berhasil
9	Tidak Berhasil	Berhasil
10	Tidak Berhasil	Berhasil
11	Berhasil	Berhasil
12	Tidak Berhasil	Berhasil
13	Tidak Berhasil	Berhasil
14	Tidak Berhasil	Berhasil
15	Tidak Berhasil	Berhasil
16	Tidak Berhasil	Berhasil
17	Tidak Berhasil	Berhasil
18	Berhasil	Berhasil
19	Tidak Berhasil	Berhasil
20	Tidak Berhasil	Berhasil

Data Tabel 4. 5 menunjukan bahwa metode sebelumnya tidak mampu menyelesaikan pengujian dengan studi kasus target yang bergerak. Dengan tingkat keberhasilan 3 dari 20 kali pengujian atau hanya 15% tidak memungkinkan menggunakan metode sebelumnya untuk menyelesaikan permasalahan target yang

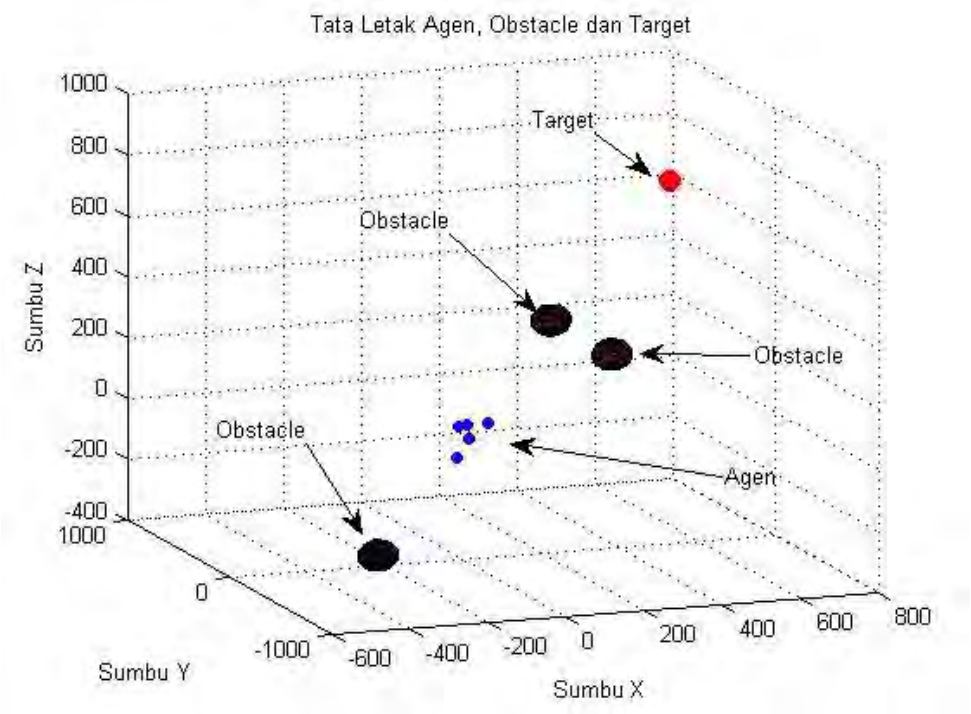
selalu bergerak. Lain halnya dengan metode yang sedang diteliti. Metode tersebut dapat menyelesaikan 20 uji coba dari 20 kali pengujian yang diberikan.

Pengujian selanjutnya dilakukan untuk menguji apakah dengan penambahan obstacle pada ruang 3D dapat diatasi oleh algoritma *bee colony* yang diteliti. Pengujian dilakukan dengan menempatkan beberapa obstacle statis dan komponen lainnya pada koordinat seperti pada Tabel 4. 6.

Tabel 4. 6 Data posisi awal komponen

Objek	**X**	**Y**	**Z**
Obstacle 1	300	300	300
Obstacle 2	-300	-300	-300
Obstacle 3	300	-300	300
Agen 1	-19.87	-27.40	42.33
Agen 2	3.80	-34.15	2.92
Agen 3	-26.55	-28.21	-60.09
Agen 4	60.42	-1.96	40.48
Agen 5	1.32	-21.59	48.11
Target	700	650	650

Dari tabel tersebut penempatan agen, obstacle dan target jika divisualisasikan dapat terlihat seperti pada Gambar 4. 17

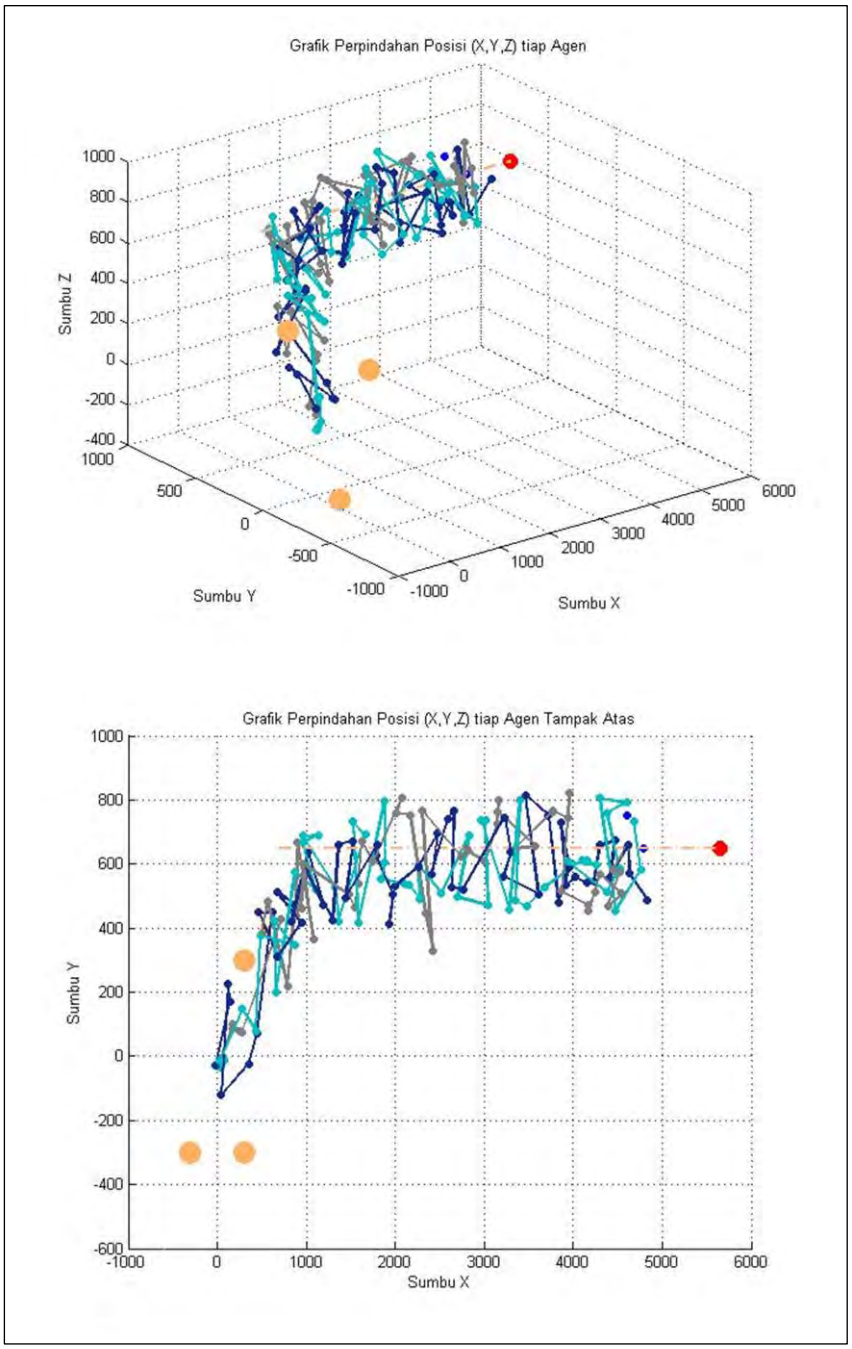


Gambar 4. 17 Lokasi awal penempatan komponen

Setelah diproses,agen dapat bergeraktak beraturan menuju target. Pergerakan tersebut tanpa menabrak obstacle yang ada. Rute pergerakan tersebut divisualisasikan seperti pada Gambar 4. 18.

Para agen bergerak melintasi obstacle tanpa menabrakya. Agen hanya bergerak di pinggir obstacle tetapi tidak menembusnya. Rute agen juga acak akan tetapi secara konvergen menuju target yang ada.

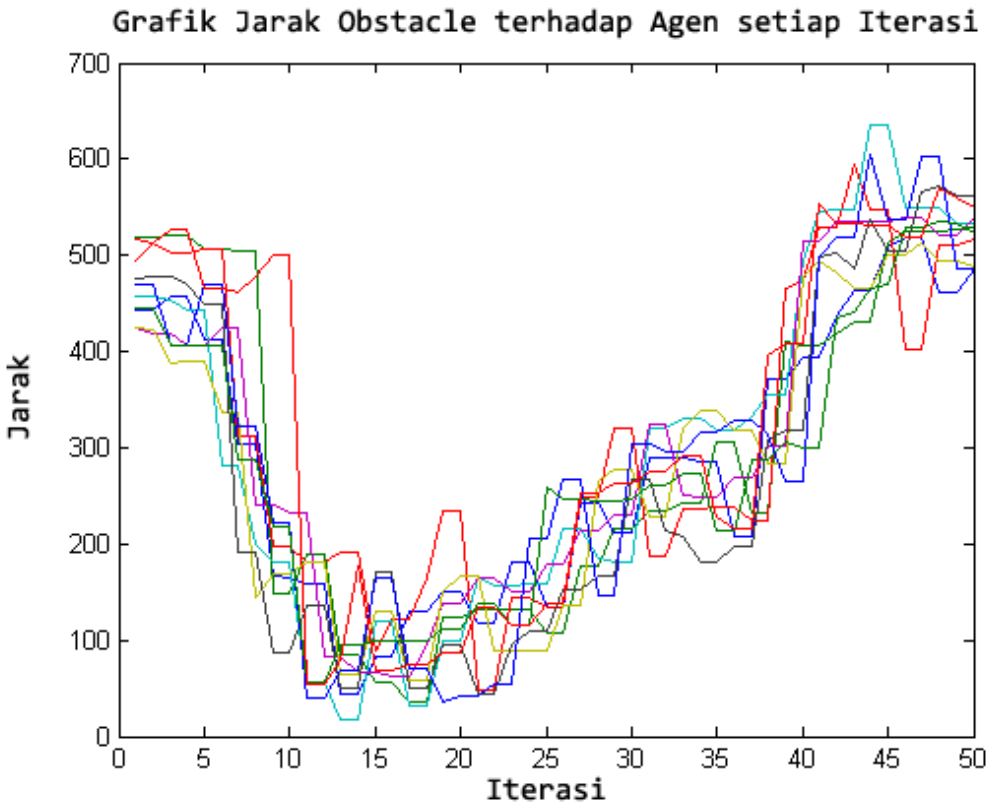
Untuk mengetahui apakah agen bertabrakan dengan obstacle, kita gunakan jarak euclide untuk setiap agen terhadap obstacle. Jika jarak tersebut bernilai lebih dari 0 semua, maka artinya tidak ada agen yang bertabrakan dengan obstacle seperti yang dijelaskan pada sub bab *scouting fase*.



Gambar 4. 18 Visualisasi rute agen menuju target

Setelah dilakukan beberapa kali pengujian, diadapatkan jarak terkecil dari agen yaitu sekitar 17. Hal itu menandakan tidak ada tabrakan antara Agen dengan obstacle.

Jika dilihat dari grafik jarak obstacle Gambar 4. 19, berdasarkan tata letak simulasi yang dijelaskan sebelumnya pada Gambar 4. 17, maka para agen ketika menuju target, akan bertemu sebuah obstacle. Sehingga pada grafik jarak terlihat jarak para agen yang semula jauh dari obstacle mendekati obstacle tetapi masih menjaga jarak sehingga masih ada jarak walaupun sedikit, kemudian para agen setelah aman dari obstacle, maka akan melanjutkan perjalanan menuju target. Jarak ke obstacle setelah agen bebas pun semakin lama semakin naik menjauh dari obstacle.



Gambar 4. 19 Grafik jarak agen terhadap obstacle yang ada

Dari beberapa pengujian yang telah dilakukan, algoritma *bee colony* yang diteliti, telah diujikan pada skenario dengan kasus agen lebih dari satu, target yang dinamis, dan adanya obstacle yang menghalangi agen. Pada pengujian selanjutnya, pengujian dilakukan untuk menguji apakah algoritma *bee colony* dapat menangani

target yang berjumlah lebih dari satu. Pertama, diberikan data target yang memiliki parameter yang berbeda – beda seperti yang ada pada Tabel 4. 7.

Tabel 4. 7 Data Parameter Target

Target ID	Target Color	HP	Attack	Defense
1		100	12	6
2		95	14	7
3		90	16	8
4		85	18	6
5		80	20	7

Data parameter target tersebut jika kita hitung nilai fitness kekuatannya menggunakan persamaan (3.14) maka akan menghasilkan data fitness yang bervariasi seperti Tabel 4. 8. Dari data tersebut diketahui bahwa rata-rata fitness dengan target berindeks 1 memiliki rata-rata fitness yang lebih tinggi dibandingkan dengan target lainnya. Sehingga nantinya para agen akan bergerak menuju ke target 1.

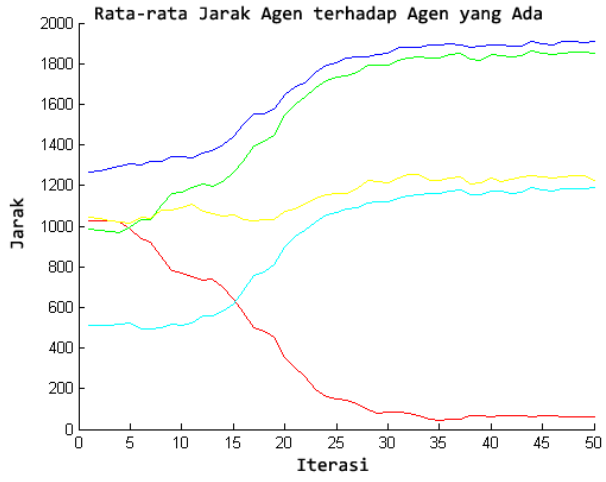
Tabel 4. 8 Data Nilai Fitnes Kekuatan antara Agen dengan Target

Agent ID	Target ID	Fitness
1	1	-1.38889
1	2	-3.57576
1	3	-5.16484
1	4	-3.95833
1	5	-4.78431
2	1	0.065359
2	2	-2.12885
2	3	-3.65
2	4	-2.89272
2	5	-3.65419
3	1	1.25
3	2	-1
3	3	-2.5
3	4	-2.07143
3	5	-2.79605

Lanjutan Tabel 4. 8

Agent ID	Target ID	Fitness
4	1	2.242424
4	2	-0.10025
4	3	-1.6087
4	4	-1.43098
4	5	-2.13518
5	1	3.095238
5	2	0.628019
5	3	-0.90909
5	4	-0.92949
5	5	-1.62319

Setelah dilakukan beberapa kali uji coba dengan menggunakan target sesuai dengan Tabel 4. 8, didapatkan hasil sesuai dengan prediksi yaitu agen-agen bergerak menuju target 1 yang berwarna merah. Hal ini ditunjukan dengan rata-rata jarak antara agen dengan target yang semakin kecil pada tiap iterasinya seperti yang terlihat pada Gambar 4. 20.



Gambar 4. 20 Rata-rata jarak antara agen ke target

Halaman ini sengaja dikosongkan

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dari 4 studi kasus yang telah di uji cobakan, yaitu kasus untuk mencapai target tanpa bertabrakan antar agen, kasus untuk mencapai target yang bergerak, kasus untuk bergerak menuju target tanpa bertabrakan dengan obstacle, dan kasus untuk multi target, metode *bee colony* yang terdahulu hanya dapat mengatasi 2 dari 4 permasalahan tersebut. Sedangkan metode *bee colony* yang diteliti, dapat menyelasaikan ke-4 studi kasus yang diberikan tersebut.

Selain itu metode yang diteliti ini memiliki tingkat keberhasilan 95% untuk kasus multi agen dengan target yang statis, 35% lebih tinggi dari metode sebelumnya. Sedangkan untuk target yang bergerak, metode yang diteliti tingkat keberhasilannya 100% sedangkan metode sebelumnya hanya 15%.

Fungsi fitness yang diusulkan dalam *bee colony* yang di teliti, memainkan peran besar dalam penentuan posisi agen. Fungsi fitness yang diusulkan dapat membuat agen tidak bertabrakan antar sesama agen dan dengan obstacle yang ada. Selain itu fungsi fitness tersebut dapat membuat agen memilih dan mengejar target yang dituju.

5.2 Saran

Penelitian ini telah terbukti dapat memperbaiki kelemahan algoritma *bee colony* sebelumnya. Yaitu dalam hal *multitargetting* dan target yang dinamis. Akan tetapi pada penelitian kali ini, agen masih bergerak secara berkelompok dan hanya bergerak menuju satu target yang optimal saja.

Saran dari penulis adalah kompleksitas pada penelitian ini dapat ditambah. Sehingga agen dapat berpisah dan menyebar secara otonom kemudian menuju target yang sesuai dengan nilai optimal fitness dari tiap-tiap agen. Akan tetapi walau

terpisah, para agen tetap bergerak seperti lebah dan menjadi colony-colony yang lebih kecil.

DAFTAR PUSTAKA

Asmiatun, S., Hermawan, L., & Daryatni, T. (2013). Strategi Menyerang Jarak Dekat Menggunakan Klasifikasi Bayesian Pada NPC (Non Player Character), *2013*(November), 351–357.

Bhattacharjee, P., Rakshit, P., Chakraborty, G., & Konar, A. (2011). Multi-Robot Path-Planning Using Artificial Bee Colony Optimization Algorithm. *Third World Congress on Nature and Biologically Inspired Computing*, 219–224.

Dasgupta, P. (2008). A Multiagent Swarming System for Distributed Automatic Target Recognition Using Unmanned Aerial Vehicles. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, *38*(3), 549–563. http://doi.org/10.1109/TSMCA.2008.918619

Fan, L. (2012). Study on Improved Collision Detection Algorithm, *2*, 589–594.

Franklin, S., & Graesser, A. (1997). Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents. *Intelligent Agents III Agent Theories, Architectures, and Languages*, 21–35. http://doi.org/10.1007/BFb0013570

Ingham, J., & Ingham, J. (1997). What is an Agent?, *99*, 1–16.

Jatiningsih, W., Yuniarno, E., & Hariadi, M. (2014). Autonomous Agent Based NPC Swarm Attack Behaviour Using Bee Colony Algorithm, (ii), 1–5.

Mardi, S., Nugroho, S., Arif, Y. M., Hariadi, M., & Purnomo, M. H. (2011). Perilaku Taktis Untuk Non - Player Characters Di Game Peperangan Meniru Strategi Manusia Menggunakan Fuzzy, *6*(1), 55–64.

Nugroho, F., Mardi, S., & Hariadi, M. (2011). Simulasi Permasalahan Multiobyektif Berbasis Agen Pada Kasus Economic dan Emission Dispatch (EED) Dengan Metode Neuro Fuzzy System di Power Plant, 1–8.

Umarov, I., & Mozgovoy, M. (2012). Believable and effective AI agents in virtual worlds: Current state and future perspectives. *International Journal of Gaming and Computer-Mediated Simulations*, *4*(2), 37–59. http://doi.org/10.4018/jgcms.2012040103

BIOGRAFI PENULIS

Penulis dilahirkan di Malang, 5 Juni 1992, merupakan anak ke empat dari 4 bersaudara. Penulis telah menempuh pendidikan formal yaitu: TK BA Restu Malang, MIN Malang 1, SMPN 1 Malang, SMAN 8 Malang. Setelah lulus SMA tahun 2010, penulis diterima di Universitas Islam Negeri Maulana Malik Ibrahim Malang jurusan Teknik Informatika di tahun yang sama. Setelah menyelesaikan jenjang S1 pada tahun 2014, penulis kembali melanjutkan pendidikan pada jenjang S2 di ITS Surabaya jurusan Teknik Elektro bidang keahlian Jaringan Cerdas Multimedia konsentrasi Game Teknologi. Bidang penelitian yang diambil penulis dalam pengerjaan tesis yaitu Artificial Intelligent untuk Game