TESIS-TE-142599

Pergerakan Kelompok NPC Menuju Target Berbasis *Artificial Fish Swarm Algorithm*

DENY SAFRIL
2213205703

DOSEN PEMBIMBING
Mochamad Hariadi, S.T., M.Sc., Ph.D.
Dr. Supeno Mardi Susiki Nugroho, S.T., M.T.

PROGRAM MAGISTER
BIDANG KEAHLIAN JARINGAN CERDAS MULTIMEDIA
KONSENTRASI TEKNOLOGI PERMAINAN
JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2015

THESIS-TE-142599

NPC Swarm Movement Towards Target Based On Artificial Fish Swarm Algorithm

DENY SAFRIL
2213205703

SUPERVISOR
Mochamad Hariadi, S.T., M.Sc., Ph.D.
Dr. Supeno Mardi Susiki Nugroho, S.T., M.T.

MAGISTER PROGRAM
INTELLIGENT NETWORK EXPERTISE MULTIMEDIA
PROGRAM GAME TECHNOLOGY
DEPARTMENT OF ELECTRICAL ENGINEERING
FACULTY OF INDUSTRIAL TECHNOLOGY
SEPULUH NOPEMBER INSTITUTE OF TECHNOLOGY
SURABAYA
2015

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

Oleh:
Deny Safril
NRP. 2213205703

Tanggal Ujian : 17 Juni 2015
Periode Wisuda : September 2015

Disetujui oleh :

1. Mochamad Hariadi, S.T., M. Sc., Ph. D. (Pembimbing 1)
   NIP. 196912091997031002

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. Eko Mulyanto Yuniarno, S.T., M.T. (Penguji)
   NIP : 196806011995121009

Direktorat Program Pasca Sarjana

Prof. Dr. Ir. Adi Soeprijanto, M.T
NIP. 196404051990021001

Pergerakan Kelompok NPC Menuju Target Berbasis Artificial Fish *Swarm* Algorithm

Nama Mahasiswa : Deny Safril

NRP : 2213205703

Calon Dosen Pembimbing : Mochamad Hariadi, S.T., M.Sc., Ph.D
Dr. Supeno Mardi Susiki, S.T, M.T.

ABSTRAK

Saat ini game RTS merupakan game yang paling diminati untuk dimainkan. Penelitian ini merupakan penelitian untuk membuat simulasi pergerakan kelompok NPC yang mampu mengatasi keterbatasan jangkaun visual agen mendeteksi atau mengetahui target yang dituju. Agen otonom bergerak secara acak untuk sampai menemukan target. Pada saat bergerak menuju target NPC dirancang bergerak kearah jangkauan visual yang dipunyai dengan step yang dipunyainya sehingga dapat menemukan targetnya, namun tetap memperhitungkan posisi NPC lain agar tidak terjadi tabrakan dan harus mampu menghindari halangan. *Artificial Fish Swarm Algorithm* dipilih dalam penelitian ini karena meniru konsep dari jangkauan visual dari ikan yang terbatas ketika bergerak dalam mencari makanan di sekitar lingkungan hidupnya.

Kata Kunci: agen otonom, NPC, jangkuan visual, step, *artificial fish swarm algorithm*

NPC Swarm Movement Towards Target Based On Artificial Fish Swarm Algorithm

By : Deny Safril

Student Identity Number : 2213205703

Supervisor : Mochamad Hariadi, S.T., M.Sc., Ph.D

Dr. Supeno Mardi Susiki, S.T., M.T.

ABSTRACT

Currently RTS games is the most attractive game to play. This research is to create a simulation of the movement of groups of NPCs are able to overcome the limitations of visual range agents detect or know the intended target. Autonomous agents move randomly for up to find the target. At the time of moving towards the target NPC is designed to move towards the visual range that belongs to the step that belongs to so that it can find its target, while still taking into account the position of the other NPC in order to avoid collisions and to be able to avoid obstacles. Artificial Fish Swarm Algorithm chosen in this study because it mimics the concept of visual range of fish is limited when the move in search of food in the surrounding environment.

Keywords: autonomous agents, NPC, visual range, step, artificial fish swarm algorithm

KATA PENGANTAR

Puji syukur penulis panjatkan kepada Tuhan Yang Maha Esa atas terselesaikannya Tesis dengan judul "Pergerakan Kelompok NPC Menuju Target Berbasis *Artificial Fish Swarm Algorithm*" sesuai waktu yang telah ditentukan. Tesis ini merupakan salah satu syarat kelulusan yang harus dikerjakan oleh setiap mahasiswa S-2 di jurusan Teknik Elektro FTI-ITS.

Pada kesempatan ini penulis juga mengucapkan terima kasih atas bantuan dan dukungan kepada:

1. Bapak Dr. Tri Arief Sardjono, ST., MT. selaku Ketua Jurusan Teknik Elektro FTI – ITS Surabaya.
2. Bapak Mochamad Hariadi, S.T., M.Sc., Ph.D. dan Bapak Dr. Supeno Mardi Susiki Nugroho, S.T., M.T selaku dosen pembimbing Tesis yang telah membimbing penulis dalam menyelesaikan Tesis ini.
3. Para dosen Bidang Keahlian Jaringan Cerdas Multimedia Konsentrasi Game Technology Jurusan Teknik Elektro FTI – ITS Surabaya yang telah membekali penulis ilmu dalam mengerjakan Tesis ini serta para dosen dan karyawan Teknik Elektro ITS pada umumnya.
4. Rekan-rekan S-2 Game Technology ITS 2013, dan seluruh penghuni A235, serta seluruh pihak jurusan Teknik Elektro FTI ITS yang sudah memberikan informasi dan senantiasa memberikan bantuan kepada penulis.

Akhir kata, penulis berharap Tesis ini dapat bemanfaat bagi pembaca.

Surabaya, 04 Juni 2015

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

Navigasi otonom karakter atau *Non Playable Character (*NPC) dalam sebuah *game* merupakan sesuatu yang penting untuk menjadikan *game* tersebut menarik. Pergerakan karakter ini akan menentukan seberapa menarik dan membuat pemain terus memainkan *game* tersebut.

Game yang bergenre RTS (*Real Time Strategy*) adalah game yang mempunyai ciri khas permainan perang. Dimana di game ini banyak dilakukan pembangunan kekuatan, pengumpulan sumberdaya, pembangunan dan pengaturan pasukan tempur dan pergerakan serta formasi pasukan tersebut.Salah satu contoh game RTS adalah *Clash of Clans* (CoC) yang dibuat dan disebarkan oleh Supercell. Pergerakan kelompok bisa digunakan untuk formasi menyerang maupun bertahan. ContohN PC yang melakukan serangan individu adalah *Archer* dan *Wizard*. *Archer* menyerang target dengan menggunakan panah. *Wizard* menembak target dengan bola api/cahaya. Sedangkan contoh NPC yang melakukan serangan berkelompok adalah *Barbarian, Giant, Ballon, Dragon*.

Gambar 1.1 Serangan barbarian secara berkelompok

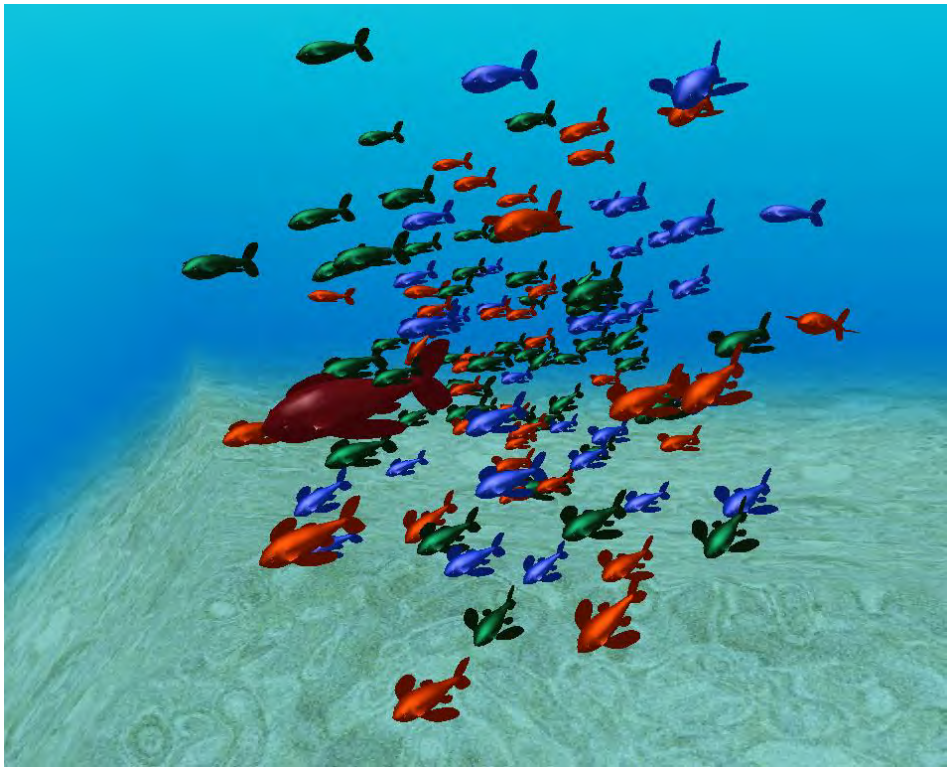
Gambar 1.1 adalah serangan barbarian yang bergerak secara berkelompok untuk menyerang target serangan. Terlihat bahwa seluruh individu bergerak bersama secara berkelompok kearah target yang dituju.

Pergerakan pada sebuah game bisa pergerakan individu maupun pergerakan kelompok. Pergerakan kelompok *Non Playable Character* (NPC) pada umumnya meniru pergerakan kelompok makhluk hidup yang ada di alam. Jika di daratan meniru pergerakan kelomok binantang darat seperti kawanan domba (*herd of sheeps*), di air meniru pergerakan kelompok kawanan ikan (*fish schools*), di udara meniru pergerakan kelompok dara kawan burung (*bird flock*). Pergerakan kelompok ini bisa digunakan untuk berbagai macam agen NPC dalam game, misalnya untuk pasukan tempur dalam melakukan formasinya dalam bergerak dalam situasi bertempur dan bertahan.

Penerapan *swarm intelligence* pada game biasanya digunakan untuk pergerakan sekelompok agen yang pergerakannya dapat ditebak, cenderung bergerombol menyerang musuh dari arah yang sama dan seolah olah telah mengetahui koordinat target yang akan dituju. Hal ini membuat pasukan agen itu mudah dilumpuhkan atau ditembak oleh lawan sehingga membuat game menjadi membosankan dan kurang menarik.

Untuk itu dibutuhkan agen yang dapat bergerak otonom dengan mengadaptasi perilaku makhluk hidup sesungguhnya. Perilaku makhluk hidup yang bergerak otonom dan berkelompok yang dapat ditiru adalah kawanan ikan. Dimana kawanan ikan tersebut bergerak dalam air untuk mencari makanan dari satu tempat ke tempat lain dengan keterbatasan visual, karena jangkauan mata ikan itu sendiri maupun karena tingkat kekeruhan air sebagai lingkungannya. Ikan akan terus bergerak acak ke arah mana saja sampai dia menemukan makanannya.

Gambar 1.2 adalah kawanan ikan yang bergerak di dalam lingkungannya, dimana setiap individu dalam kawanan tersebut selalu tau harus berkelompok dengan kawanannya kemanapun arah kawanan itu bergerak semua individu bergerak mengikuti dan terikat didalam kawananenghindarinya. Tidak ada kawanan ikan yang bergerak sendiri diluar kelompoknya. Ikan bergerak dalam kawanan berfungsi untuk memudahkan mendapatkan makanan maupun mengurangi kemungkinan dimangsa predator.



Gambar 1.2 Ikan yang bergerak dalam kawanan

Dalam penelitian ini akan membahas pergerakan agen NPC yang bergerak dalam kelompok, dalam hal ini agen berupa kawanan ikan dan tingkah lakunya ketika bergerak ke suatu target dan menghindari halangan dalam lingkungan 3 dimensi.Kawanan ikan memiliki pengetahuan yang lengkap tentang lingkungan , dan rencana pergerakannya didasarkan pada itu . Tapi secara umum, kawanan ikanhanya memiliki gagasan tentang tujuan dan harus mencapai itu menggunakan penglihatan visual sebagai sensor untuk mengumpulkan informasi tentang lingkungan yang ada di sekitarnya. Gambar 1.2 adalah kawanan ikan yang bergerak di dalam lingkungannya, dimana setiap individu dalam kawanan tersebut selalu tau harus berkelompok dengan kawanannya kemanapun arah kawanan itu bergerak semua individu bergerak mengikuti dan tberkerumun didalam kawanannya.

Perilaku sosial dari kawanan ikan dalam mencari , mengerumuni dan mengikuti, dijelaskan dalam *Artificial Fish Swarm Algorithm* (AFSA). Bagaimana kawanan ikan bergerak menuju sasaran dapat disimulasikan menggunakan *Artificial Fish Swarm Algorithm* (AFSA) . AFSA adalah salah satu metode optimasi

terbaik algoritma *swarm intelligence* . Algoritma ini terinspirasi oleh gerakan berkelompok ikan dan berbagai perilaku sosial mereka . Berdasarkan serangkaian perilaku naluriah , ikan selalu berusaha untuk mempertahankan koloni mereka dan menunjukkan perilaku cerdas . Mencari makanan , imigrasi dan berurusan dengan bahaya semua terjadi dalam bentuk sosial dan interaksi antara semua ikan dalam kelompok akan menghasilkan perilaku sosial cerdas [10].

1.2 Rumusan Masalah

Gerakan sekelompok NPC (*Non Playable Caharacter*) menuju target pada umumnya adalah mencari jarak terpendek dan seolah olah titik target sudah diketahui koordinatnya sehingga agen dengan mudah menemukan target.

Jangkauan visual dari tiap agen mempunyai keterbatasan sehingga agen yang bergerak dalam kelompoknya tidak dengan segera tahu dimana letak target yang akan menjadi tujuannya. Di dalam metode path finding yang lain, pergerakan agen hanya memperhitungkan jarak, arah dan adanya halangan saja serta koordinat target yang sudah diketahui, disaat agen bergerak menuju target tapi dengan keadaan target secara visual tidak kelihatan atau belum tahu koordinatnya menjadi sebuah permasalahan, dan hal ini juga akan memepengaruhi perilaku dari agen tersebut.

1.3 Batasan Masalah

1. *Artificial Fish Swarm Algorithm* digunakan untuk menentukan perilaku kelompok NPC yang bergerak menuju target statis dengan menghindari halangan statis yang ada disekitarnya dengan keterbatasan jangkauan visual agen.
2. Waktu yang dibutuhkan agen untuk mencapai target statis dengan ada atau tidak ada halangan statis yang berada disepanjang jalur agen menuju ke target tujuan.

1.4 Tujuan

Tujuan dari penelitian ini adalah mengembangkan pergerakan kelompok NPC menuju target, berbasis *Artificial Fish Swarm Algorithm* yang meniru perilaku kawanan ikan yang ada di alam.

1.5 Manfaat Penelitian

Mengembangkan pergerakan karakter secara otonom dengan *Artificial Intelligent* yang ditanamkan ke karakter tersebut, serta dapat di implementasikan ke dalam game.

BAB II
KAJIAN PUSTAKA DAN DASAR TEORI

2.1 Kecerdasan Buatan

Kecerdasan buatan adalah tentang membuat komputer mampu melakukan tugas-tugas berpikir yang mampu dilakukan manusia dan hewan . komputer sudah bisa diprogram untuk memiliki kemampuan super manusia dalam memecahkan banyak masalah : aritmatika , menyortir , mencari , dan sebagainya . Bahkan bisa mendapatkan komputer untuk memainkan beberapa *board game* yang lebih baik daripada manusia (mislanya :Reversi atau Connect) [4]. Kecerdasan dibuat dan dimasukkan kedalam mesin (komputer) denga tujuan mesin tersebut bisa bekerja sesuai dengan kecerdasan yang ditanamkan ke dalamnya.

Pada saat ini komputer sudah mampu diprogram untuk memecahkan masalah dengan meniru kemampuan yang dipunyai manusia dalam berbagai bidang. Pada penelitian ini kecerdasan buatan digunakan didalam game untuk menggerakkan karakter yang ada dalam game tersebut, sehingga karakter yang ada dalam dapat berperilaku cerdas sesuai dengan kecerdasan buatan yang akan ditanamkan di dalamnnya.

2.2 *Swarm* Intelligence

Swarm Intelligence (SI) merupakan cabang dari *Artificial Intelligence* (SI) yang digunakan untuk memodelkan perilaku kolektif dari hewan sosial di alam seperti koloni semut, lebah, burung atau kawanan hewan darat. 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 sosial antar anggota *swarm* dapat dilakukan secara langsung maupun tidak langsung.

Interakasi langsung dilakukan dengan melakukan kontak visual atau suara, sedangkan 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 [7].

Tidak semua kelompok agen dapat dikatakan memiliki kecerdasan (*intelligent*). Kelompok agen dapat dikatakan cerdas jika memiliki *self-organization* dan pembagian tugas yang jelas. *Self-organization* adalah ciri utama dari sistem *swarm* yang menghasilkan perilaku kolektif yang berasal dari interaksi lokal antar agen.

Tabel 2.1 *Swarm*ing dari burung, ikan, domba, lebah

No.	**Nama**	***Swarming***
1	Burung	
2	Ikan	
3	Domba	
4	Lebah	

Tabel 2.1 adalah contoh *swarming* beberapa hewan, baik yang di udara seperti burung, di air seperti ikan maupun di darat seperti domba dan juga koloni lebah. Perilaku hewan yang berkelompok ini bisa digunakan sebagai model perilaku berkelompok di game mislanya digunakan untuk pergerakan troop yang menyerang secara berkelompok ke sebuah target serangan.

2.3 Perilaku Kelompok

Perilaku berkelompok adalah perilaku yang ditunjukkan ketika sekelompok agen bergerak bersama sama menuju sebuah tujuan tertentu dalam foramsi tertentu. Pergerakan kelompok ini meniru pergerakan kelompok makhluk yang ada di alam. Ada persamaan dengan perilaku kawanan burung, perilaku kawanan ikan, perilaku kerumunan serangga, dan perilaku kawanan hewan darat.

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[11].

Dalam game pergerakan kelompok digunakan untuk berbagai macam pergerakan NPC maupun player yang bertujuan permainan dalam game lebih menarik. Menyerang maupun bertahan adalah salah satu aktifitas dalam game yang berjenis RTS seperti contohnya game Clash of Clan. Aktifitas ini membutuhkan formasi pasukan dalam bentuk kelompok untuk melakukan serangan terhadap musuh maupun bertahan terhadap musuh ataupun melarikan diri dari musuh. Agen didalam kelompok tersebut diharapkan mempunyai kecerdasan sehingga dapat menentukan pergerakannya dalam melakukan aktifitas menyerang, bertahan, maupun melarikan diri. Jika diambil contoh dari satu aktifitas yaitu menyerang maka dalam melakukan penyerangan agen bisa menyerang sendirian maupun menyerang dalam formasi kelompok. Dalam serangan kelompok, agar pergerakan

dapat sukses melakukan serangan, terlihat almiah dan tidak mudah ditebak, maka *swarm inteligent* perlu ditanamkan di agen dalam kelompok

Simulasi komputer dan model matematika yang telah dikembangkan untuk meniru perilaku berkelompok burung secara umum dapat diterapkan juga untuk perilaku berkelompok spesies lain.Algoritma Boid banyak digunakan untuk mensimulasikan pergerakan kelompok agen , yang sering terlihat dalam perilaku kawanan burung atau kawanan ikan dalam melakukan pergerakan dalam kelompoknya. Seperti halnya program simulasi sekelompok agen , maka setiap agen berbasis boid memiliki perilaku yang berbeda yang muncul akibat reaksi buatan yang dibuat dan ditanamkan didalamnya.

Dari perspektif modeller matematika, berbondong bondong adalah gerakan kolektif dari sejumlah besar individu dalam kelompok dan merupakan perilaku hewan kolektif yang dipunyai oleh banyak makhluk hidup seperti burung , ikan , domba, dan serangga

Dalam game strategi (RTS), player harus mengontrol pasukan untuk berperang melawan musuh. Emas, elixir atau tropi dikumpulkan player untuk menguatkan unit, membangun hall/barak dan merakit pesawat atau aset lain agar menang saat perang melawan musuh. Pengumpulan emas, elixir atau tropi dapat dilakukan dengan menyerang unit lawan atau menambang di tambang emas/elixir. Jika pengumpulan emas/elixir diperoleh dari hasil mengalahkan lawan, maka player harus mengkoordinir pasukan saat melakukan serangan pada lawan.

Ada 2 jenis game strategi, yaitu *turn based* dan *real time*. Pada *turn-base strategy games* player dan lawan secara bergantian mengeluarkan perintah untuk unit mereka, seperti bermain catur. Baik player maupun lawan bergantian menjalankan pasukannya. Sedangkan *real time strategy game player* dan computer mengkoordinir pasukannya secara bersamaan (real-time).

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 [6].

Pada umumnya, untuk mengarahkan unit NPC player harus men-deploy mereka ke lokasi yang ditentukan oleh player. Sebagai contoh kasus, pada game CoC untuk mengirimkan pasukan menyerbu musuh yang memasuki wilayah, player harus mendeploy pasukan dulu ke lokasi musuh. Sehingga, jika player tidak mendeploy unit NPC untuk menyerang musuh yang masuk ke wilayahnya maka dengan mudah unit NPC musuh dapat menghancurkan wilayah itu. Gambar 2.1 adalah gerakan tap saat mengirim pasukannya secara manual ke lokasi yang dia inginkan.

Gambar 2.1. Tap Saat Deploy Pasukan

Untuk menghindari hal tersebut, dibutuhkan unit NPC berbasis agen otonom yang dapat bereaksi secara otonom terhadap setiap serangan yang masuk

kewilayah player. Artinya, dengan adanya NPC berbasis agen otonom player tidak perlu secara manual mendeploy pasukan ke lokasi musuh, karena unit NPC player akan menyerang secara otomatis setiap NPC lawan yang masuk ke wilayahnya.

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 dan Graesser 1997). [9]

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 dan Varela 1980, Varela 1991). [9]

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 [6].
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

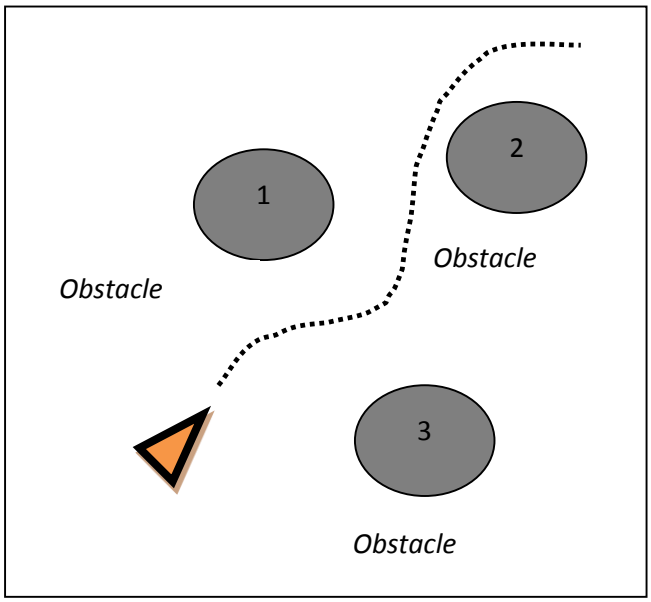
halangan, mendengarkan suara. Dan agen juga dapat bertindak (*acting*) sesuai perubahan lingkungan yang ditemuinya, seperti : melompat untuk menghindari halangan, makan. agen diberi semacam indera untuk dapat mengindera lingkungannya dan untuk berkomunikasi dengan agen lain.

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.

2.4 Model Boid

Model boid dikenal sebagai metode yang berguna untuk mensimulasikan gerakan kelompok kawanan makhluk hidup yang dikembangkan oleh Craig W Reynolds pada tahun 1986.

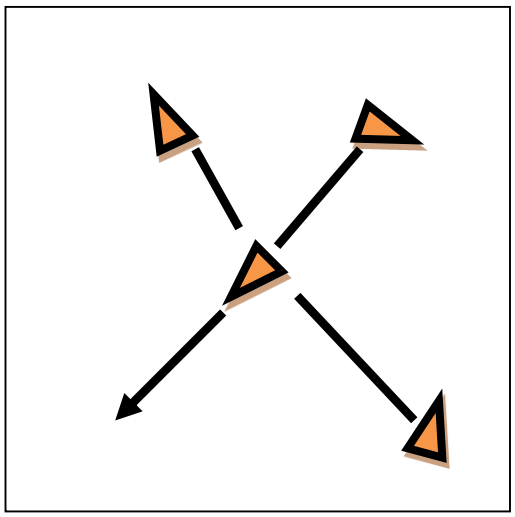
Craig W Reynold membagi steering behavior menjadi beberapa perilaku. Antara lain *seek, flee, pursuit, evasion, offset pursuit, arrival, obstacle avoidance, wander, path following, unligned collision avoidance, separation, cohesion* dan *alignment* [5]. Tetapi pada penelitian ini hanya akan digunakan *obstacle avoidance*, dan *separation* saja.



Gambar 2.2 *Obstacle Avoidance*

Obstacle avoidance adalah perilaku agen melakukan manuver untuk mengindari benturan dengan hambatan (*obstacle*). Ada perbedaan mendasar antara *obstacleavoidance* dengan *flee*. *Flee* akan selalu bergerak menjauh dari target sedangkan *obstacle avoidance* hanya bergerak menghindari rintangan jika menemukan rintangan di depannya. Gambar 2.2 adalah ilustrasi gerakan agen saat menghindari halangan (*obstacle avoidance*). Agen akan menghindari halangan yang ada disepanjang jalur agen tersebut menuju ke tujuannya.

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*). Agen akan menghitung jarak paling optimal saat agentersebut berkelompok bersama agen alain. Jarak tidak boleh terlalu dekat mauun terlalu jauh. Jika jarak terlalu dekat dengan agen lain maka agen menjauh dari agen yang lain sejauh jarak yang diperbolehkan.



Gambar 2.3 *Separation*

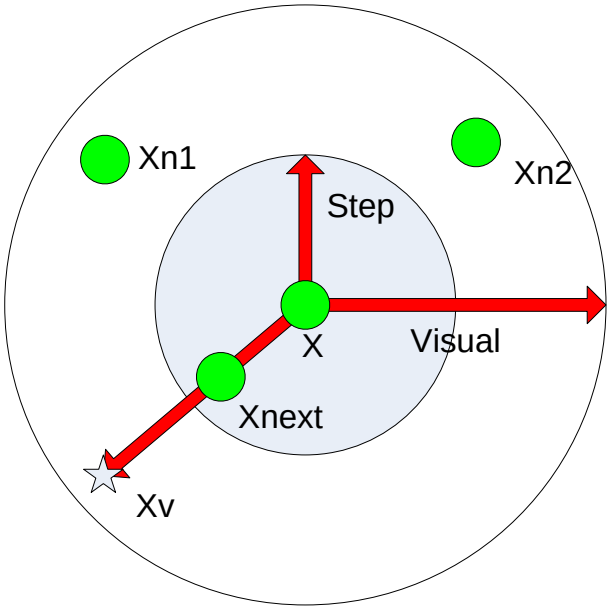
2.5 Artificial Fish *Swarm* Algorithm(AFSA)

Artificial fish *Swarm* Algorithm adalah metode baru untuk mencari global optimum, ide dasar dari AFSA adalah meniru perilaku sosial kawanan ikan di alam, saat mereka mencari, berkerumun, dan saling mengikuti di dalam satu kelompok. Seekor ikan dalam kawanan ikan dapat dengan cepat menanggapi perubahan arah

dan kecepatan ikan lain dalam kelompok. Informasi perilaku ini membantu ikan tetap berada kawanannya [2].

Di dalam limgkungan airnya, ikan selalu dapat menemukan makanan di tempat di mana ada banyak makanan , dimana banyak makanan maka disitu akn ada ikan yang berada disitu . Menurut fenomena ini , AFSA membangun beberapa *Articial Fish* (AF) , yang mencari solusi optimal dalam ruang solusi (lingkungan di mana AF hidup) dengan meniru perilaku kawanan ikan [1] .AFSA mempunyai 4 tingkah laku dasar:

1. Memangsa (*prey*) : Ini adalah perilaku biologis dasar mencari makan dengan merasakan dan melihat konsentrasi makanan dalam air untuk menentukan gerakan kearah makanan
2. Berkerumun(*swarm*): Ikan berada dalam kelompok alami dalam proses bergerak , yang merupakan semacam kebiasaan hidup untuk menjamin keberadaan koloni dan menghindari bahaya
3. Mengikuti (*follow*) : Dalam proses bergerak dari kawanan ikan , ketika ikan menemukan makanan , maka ikan lain dalam kawanan itu akan dengan cepat menemukan makanan yang sama
4. Bergerak (*move*) : ikan berenang acak dalam air ; pada kenyataannya , mereka mencari makanan atau sesamanya dalam jangkauan yang luas



Gambar 2.4 Konsep visual *artificial fish*

Gambar 2.4 adalah konsep penglihatan (visual) agen, X adalah keadaan agen saat ini , Visual adalah jarak jangkauan penglihatan agen yang dihitung dari koordinat posisi agen sampai ke koordinat titik penglihatan terjauh yang bisa dilihat oleh agen, dan X_v adalah target visual pada beberapa saat . Jika keadaan pada target visual lebih baik dari kondisi saat ini, ia pergi ke arah ini , dan menuju keadaan X_{next} serta terus melakukan pengamatan visual. Semakin banyak melakukan pengamatan visual maka semakin banyak pengetahuan yang didapat ikan. $X = (x_1, x_2,...., x_n)$ dan $X_v = (x^v_1, x^v_2,...., x^v_n)$ maka proses ini dapat diekspresikan sebagai berikut:

$$x_i^v = x_i + \text{Visual}.\,\text{rand}()\ i \in (0, n] \quad (2.1)$$

$$\text{Xnext} = X + \frac{X_v - X}{\|X_v - X\|}.\,\text{Step}.\,\text{rand}() \quad (2.2)$$

di mana rand ( ) menghasilkan angka acak antara nol dan 1 , Step adalah panjang langkah ketika bergerak ke arah target visual X_v , dan x_i adalah variabel optimasi , n adalah jumlah variabel [8] .

Algoritma *Artificial Fish Swarm* mempunyai dua bagian yaitu variabel dan fungsi. Yang termasuk varibael adalah X (posisi agen saat ini), Step adalah panjang langkah agen ketika bergerak, Visual adalah jarak jangkauan penglihatan agen yang dihitung dari koordinat posisi agen sampai ke koordinat titik penglihatan terjauh yang bisa dilihat oleh agen, try_number (interasi maksimum), crowd factor (δ) ($0<\delta<1$). Sedangkan Fungsi adalah : perilaku memangsa *(prey)*, perilaku bergerak bebas (*free move)*, perilaku berkerumun *(swarm)*, perilaku mengikuti *(follow)* [3]. Agen mencari lokasi dengan nilai fitness paling baik dalam masalah mencari ruang dengan melakukan keempat perilaku ini dengan menerapkan prosedur algoritma.

Algoritma *Articial Fish Swarm* dapat dilaksanakan sebagai berikut :

Langkah 1 Inisialisasi parameter dari agen : Step, Visual, maksimum iterasi, crowd dan n ikan secara acak

Langkah 2 Set temporari untuk merekam keadaan saat ini dari tiap agen dan pilih nilai optimal yang terekam

Langkah 3 Implementasi dari perilaku memangsa (*prey*), berkerumun (*swarm*) dan mengikuti (*follow*) dan bergerak bebas (*free move)*

Langkah 4 Nilai optimal temporari di perbarui

Langkah 5 Jika kondisi akhir sudah tercapai, keluarkan hasilnya.; jika tidak kembali ke langkah 2.

2.6 Jarak Euclidian

Jarak Euclidian adalah perhitungan jarak antara dua buah titik didalam ruang euclidian. Penghitungan jarak ini bisa dilakukan untuk 1 deimensi 2 dimensi 3 dimensi maupun n dimensi. Perhitungan jarak euclidian dilakukan dengan cara sebagai berikut. Jika ada 2 buah titik misalkan titik pertama x dan titik kedua x_1, maka jarak euclidian dapat dithitung dengan mengurangkan x_1 dengan x dan cari nilai absolutnya, kemudian mengkuadratkan hasilnya setelah itu hasilnya diakarkan. Jika pada ruang 3 dimensi ada titik pertama dengan koordinat (x_i,y_i,z_i) dan titik kedua (x_j, y_j,z_j) maka jarak euclidian dapat dihitung sebagai berikut :

$$D_{i-j} = \sqrt{\left(x_j - x_i\right)^2 + (y_j - y_i)^2 + (z_j + z_i)^2} \qquad (2.3)$$

2.7 Clash of Clans

Penelitian ini difokuskan pada pencarian rute pergerakan sekelompok agen saat menuju target tertentu dan Clash of Clans (CoC) menjadi salah satu rujukan penelitian ini. Pasukan dalam CoC dikenal dengan istilah *troops*. Pasukan atau *troops* digunakan untuk mempertahankan wilayahnya (desa) atau menghancurkan *base* lawan, mengumpulkan *Tropi, Gold* dan *Elixir*. *Troop* diciptakan di *Barracks* kemudian disimpan di *Army Camps*.. Setiap pasukan yang ada pada CoC memiliki kemampuan dan ciri khas yang berbeda-beda. *Player* dapat meningkatkan kemampuan *troop*nya namun hal ini akan membuat *Elixir* yang dikumpulkan berkurang sesuai dengan jumlah *Elixir* yang dibutuhkan untuk meng*upgradetroop*nya. Pasukan dalam CoC dikelompokkan dalam beberapa kelompok yaitu : Tier 1, Tier 2, Tier 3, Dark Elixir dan Heroes

Gambar 2.5 Troop pada Tier 1 CoC
(http://1.bp.blogspot.com/)

Gambar 2.5 adalah gampar troops yang ada di kelompok Tier 1. *Troop s*pada Tier 1 mudah dilatih dan harganya murah. Mereka memiliki *hitpoint* yang rendah dan tidak mampu membuat kerusakan yang fatal jika bekerja secara individu. Mereka lebih mampu membuat kerusakan besar jika bekerja dalam kelompok saat menyerang pertahanan tunggal.*Troop* pada Tier 1 mudah dikalahkan oleh Wizards Tower dan Mortars hanya dengan sekali tembak. Barbarian, Archer dan Goblin adalah *troop* yang ada di Tier 1.

Gambar 2.6 Troop pada Tier 2 CoC
(http://1.bp.blogspot.com/)

Gambar 2.6 adalah gambar *troops* yang ada di kelompok Tier 2. *Troop* yang ada di Tier 2 lebih kuat daripada *troop* di Tier 1. Sama pada *troop* di Tier 1, *troops* pada Tier 2 juga memiliki keahlian khusus. Seperti Wall Breaker yang hanya bertugas menghancurkan dinding, Wizards meskipun tidak terlalu kuat namun mampu menembakkan api pada musuh, Giant dan Ballons dirancang untuk

mempertahankan wilayah. Ballons merupakan unit terbang pertama yang tersedia di CoC.

Gambar 2.7 Troop pada Tier 3 CoC
(http://1.bp.blogspot.com/)

Gambar 2.7 adalah sekumpulan *troops* yang ada di Tier 3. *Troops* Tier 3 merupakan *troop non hero* yang paling kuat dan paling bertenaga dalam CoC. Mereka mampu menghancurkan semua desa hanya dengan sedikit pasukan. Namun, *troops* di Tier 3 membutuhkan biaya yang mahal dan waktu yang lama untuk melatihnya. Pada umumnya, kekuatan 6 Barbarian atau Archer di *level* 6 setara dengan kemampuan *troop* di Tier 3 ini. Keunggulan lain *troop* di Tier 3 adalah mereka tidak mudah dikalahkan oleh Mortars dan Wizard Tower.

Gambar 2.8 adalah sekumpulan *troops* yang ada di Dark Elixir. Kelompok *troop* Dark Elixir adalah *troops* yang memiliki fungsi khusus yang tidak dimiliki oleh *troop* biasa, seperti *troop* terbang, perusak dinding, *troop* yang bertugas membuat kerusakan darat dengan menggunakan senjata yang dilempar dan kekuatan besar. Troop yang menjadi anggota kelompok Dark Elixir adalah Minions, Hog Riders, Valkyries, Golems, Witches dan Lava Hounds.

Gambar 2.8 Dark Elixir Troop
(http://clashofclans.wikia.com/wiki/)

Berikutnya adalah kelompok *troop* Heroes yang terdiri dari Barbarian King dan Archer Queen. Mereka adalah *troop* yang kekal atau*immortal*. Mereka memiliki kekuatan terbesar namun setiap *player* hanya memiliki 1 *troop* Heroes saja. Meskipun Heroes tidak dapat mati, tapi mereka tetap dapat terluka dan membutuhkan waktu istirahat untuk menyembuhkan luka mereka. Saat melakukan serangan dan untuk menjaga agar Heroes tidak terluka, Heroes perlu di*back up* oleh *troop* lain. *Troop* yang biasa digunakan untuk membackup serangan Heroes adalah Barbarian dan Archer. Setiap Barbarian King dan Archer Queen dapat mengeluarkan Barbarian dan Archer untuk membantu mereka menyerang lawan.

Setiap *troop* mempunyai ciri khas yang tidak dimiliki oleh *troop* lain. Mereka juga memiliki cara bertahan dan menyerang yang berbeda-beda. Berikut ini adalah beberapa *troop* yang pada game Clash of Clans yang melakukan serangan secara berkelompok :

1. Barbarian

Barbarian adalah unit pertama yang dimiliki *player*. Penampilan *troop* ini digambarkan sebagai seorang pria dengan ekspresi marah, siap tempur, rambut kuning dan kumis kuning panjang. Karakter ini memakai rok coklat dengan sabuk kulit serta membawa perisai, *hand band* berduri pada kedua lengannya, memakai sandal dan membawa pedang di tangan kanannya. Ketika di*upgrade* pada *level* 3, Barbarian akan menggunakan *headband*. Pada *level* 5, Barbarian akan menggunakan atribut helm Viking dan pada *level* 6 akan pedangnya akan lebih

tajam serta memakai helm tanduk raksasa. Gambar 2.9 adalah perubahan penampilan Barbarian di beberapa level.

Gambar 2.9 Karakter Barbarian di Beberapa Level
(http://clashofclans.wikia.com/)

Archer dan Giants dirancang untuk mendukung serangan yang dilakukan oleh Barbarian. Karena Barbarian mudah dikalahkan oleh Mortar dan Wizard Tower yang melakukan serangan *spash damage*, maka pada umumnya*player* menyebar Barbarian untuk menyerang secara berkelompok dari titik yang berbeda. Barbarian secara brutal serta tanpa menghitung kekuatan target, akan menyerang dan merusak gedung dengan posisi terdekat dengan dirinya. Jika Mortar dalam kondisi lemah dan kehabisan amunisi, Barbarian dapat dengan mudah menghancurkan Mortar meskipun Barbarian menyerangnya sendirian. Bersama Archer, Barbarian secara berkelompok akan melindungi Barbarian King atau Archer Queen.

2. Archer

Archer merupakan unit pemanah. *Troops* ini digambarkan sebagai seorang wanita dengan rambut dan mata berwarna merah muda yang tajam, gaun hijau muda, sabuk dengan tas kecil, cincin emas di bahunya, dan bulu merah muda pada anak panahnya. Archer tidak memiliki target yang spesifik, menyerang apa saja yang di hadapan dan jangkauannya.

Archer mudah kalah jika diserang dengan api tapi mampu melakukan serangan jarak jauh dengan panahnya dan dapat menyerang dari balik dinding. Karena mudah kalah melawan api, Archer harus menghindari serangan Mortars, Wizard Towers dan Bomb. Jika Archer melawan sendirian, maka dia akan dengan mudah dikalahkan. Oleh karena itu, pada umumnya *player* akan mengkondisikan

Archer melakukan serangan secara berkelompok untuk memperbesar kemungkinan menang saat melawan target.

3. Goblin

Karakter Goblin digambarkan sebagai mahluk hijau kecil dengan telinga runcing lebar, mata merah dengan pupil berwarna hijau dan hidung merah. Goblin memakai celana dan sepatu coklat serta membawa karung yang berisi sumber daya yang dicuri dari lawan. Goblin merupakan karakter tercepat dalam CoC. Bangunan sumber daya adalah target utama Goblin, seperti Gold, Elixir dan penyimpanan. Goblin mampu memberikan kerusakan 2 kali lipat lebih parah dibandingkan kerusakan yang dibuat oleh Barbarian dan 3 kali lipat dibandingkan Archer untuk serangan yang ditujukan pada bangunan. Karena Goblin mengutamakan menyerang bangunan sumber daya, maka mereka akan mengabaikan semua pertahanan lawan, rentan terhadap serangan dan lemah jika digunakan tanpa *back up* dari *troop* lain. Gambar 2.10 adalah perubahan karakter Goblin di beberapa level game CoC.

Gambar 2.10 Karakter Goblin di Beberapa Level
(http://clashofclans.wikia.com/)

Saat mencuri sumber daya lawan Goblin membutuhkan bantuan Wall Breaker untuk menghancurkan dinding lawan. Barbarian, Giant atau troop dengan nyawa yang lebih tinggi untuk menghancurkan Mortar dan Wizard Tower yang melindungi sumber daya lawan. sama seperti Barbarian, Goblin sangat rentan terhadap Mortar dan Wizard Tower. Untuk mengatisipasi hal ini, pada umumnya Goblin dikirim dalam jumlah yang besar agar kemungkinan Goblin yang mampu mencuri sumber daya juga semakin besar. Goblin juga dapat dimunculkan secara massal karena kecepatan dan *power*nya dapat menghancurkan banyak gedung. Ketika semua penyimpanan sumber daya telah dihancurkan, mereka akan menghancurkan apapun yang bisa mereka serang.

4. Giant

Giant adalah *troop* berbadan besar yang mampu memporakporandakan lawan. Troop ini mengincar pertahanan musush seperti Canon, Wizard Tower dan Archer Tower. Dalam jumlah yang besar mereka dapat menghancurkan sebuah desa. Karena *hitpoint* Giant sangat besar, pada umumnya *player* menempatkan mereka terlebih dahulu untuk melindungi pasukan lain yang lebih lemah. Gambar 2.11 adalah perbedaan perubahan Giant di setiap level.

Gambar 2.11 Karakter Goblin di Beberapa Level
(http://clashofclans.wikia.com/)

5. Wall Breaker

Wall Breaker digambarkan seperti tengkorak kecil yang memakai topi coklat dan memiliki luka kecil melintasi mata kanannya dengan membawa bom. Targetnya adalah menghancurkan bangunan dan dinding lawan terdekat yang dilindungi *troop* lawan dengan cara meledakkan diri sendiri menggunakan bom yang dibawanya. Selama belum mati mereka akan terus berusaha menghancurkan gedung dan dinding lawan.Wall Breaker menghancurkan dinding lebih cepat dari unit lain dan selalu menjaga jarak dengan *troop* lain agar ketika meledak troop itu tidak ikut meledak. Mereka akan mencari gedung terdekat yang dikelilingi oleh dinding dan menghancurkan semua dinding yang mengelilingi gedung itu. Saat menyerang, Giant dan Wall Breaker adalah tim yang tepat. Giant maju terlebih dahulu untuk menghancurkan troop pertahanan lawan, agar Wall Breaker tidak mati sebelum berhasil menghancurkan gedung/dinding..

6. Ballon

Ballon merupakan unit terbang yang bergerak lambat menyerang lawan yang ada di darat dengan cara menjatuhkan bom dari atas. Kadar kerusakan yang diciptakan oleh Ballon cukup besar. Target utama Ballons adalah bangunan pertahanan

lawan.Disarankan untuk menghancurkan pertahanan lawan seperti Archer Tower, Wizards Towers dan Air Defense sebelum mengirim Ballon ke *base* lawan. Meskipun mampu membuat kerusakan yang parah, Ballons harus tetap waspada dengan Archers, Minions, Wizards dan Dragon yang bersembunyi di Clan Castle.

7. Wizard

Wizard digambarkan sebagai lelaki berjubah biru, memakai sabuk kulit dengan gasper emas dan sepatu *boot* coklat. Serangan Wizard masuk dalam kategori serangan jarak jauh dan dapat menyerang pertahanan lawan meskipun terhalang dinding. Bola api yang keluar dari tangannya mampu membuat kerusakan yang cukup parah namun *troop* ini lemah sehingga mudah untuk dikalahkan lawan.

8. Healler

Healler adalah unit terbang yang tidak memiliki kemampuan untuk menyerang lawa, tetapi sesuai dengan namanya, Healer dapat menyembuhkan troop darat lain. Jika semua troop dalam kondisi sehat, Healler tidak akan melakukan apa-apa. Healler akan terbang di atas troop dan melakukan proses penyembuhan jika ada troop yang terluka. Meskipun sebagai troop penyembuh, namun Healler tidak dapat menyembuhkan unit terbang lain seperti Dragon, Ballon, Minion atau Healler sendiri.

9. Dragon

Dragon adalah unit terbang yang menakutkan dan mampu melakukan serangan udara maupun darat. Berbeda dengan Ballon, Dragon mampu menciptakan kerusakan yang lebih luas dan lebih besar, seperti kerusakan yang disebabkan oleh Wizard. Berbeda juga dengan Ballon, Dragon hanya menyerang target apapun yang paling dekat dengannya.

10. PEKKA

Sesuai dengan namanya, "*Perfect Enraged Knight Killer of Assassins*",P.E.K.K.A merupakan salah satu pasukan kuat yang mampu membuat kerusakan besar. Karakter PEKKA digambarkan sebagai kesatria berbaju besi lengkap dengan helm *fullface* bertanduk dan pedang di tangan kanan.Karena kemampuan merusak dan kesehatan PEKKA sangat tinggi dibandingkan dengan unit lain, dianjurkan untuk mengirim sepasang PEKKA di setiap sudut desa lawan atau di bangunan pertahanan lawan. Bahkan Mortar tidak akan mampu menahan sepasang PEKKA.

11. Minions

Gambar 2.12 adalah karakter Minion di beberapa level, Minions merupakan *troop* yang mampu terbang cepat, murah tapi cenderung lemah. Berbentuk seperti *gargoyle* dengan tanduk besar, sayap pendek dan bertangan besar dengan cakar tajam. Karena Minion adalah unit terbang, Minions tidak dapat diserang oleh Cannons, Mortars atau X-Bows jika dalam Ground Mode, tapi sangat rentan terhadap Air Defense dan Air Bombs. Gambar 2.12 adalah perubahan tampilan Minions di beberapa level CoC.

Gambar 2.12 Karakter Minions di Beberapa Level
(http://clashofclans.wikia.com/)

Meskipun Minion merupakan *troop* yang lemah namum mereka cukup mampu membuat kerusakan jika menyerang secara berkelompok. Oleh karena itu *player* disarankan untuk mengirim 10 atau lebih Minions untuk menyerang bangunan yang hanya dilindungi oleh *ground defenses*, seperti Mortars, Cannons dan X-Bows. Minions relatif ekonomis dan tidak membutuhkan banyak ruang sehingga cocok digunakan untuk strategi menyerang cepat secara berkelompok. Minions dapat dengan mudah dikalahkan oleh Air Bomb, oleh karena itu sebaiknya Minion dikirim ke lokasi yang tidak ada AirBomb-nya secara masal. Sebelum mengirim Minion, semua unit terbang lawan seperti Archer Tower, Wizards Tower dan Air Defenses harus hancurkan dahulu. Oleh karena itu pemilihan Giants sebagai *back up* serangan Minion merupakan strategi yang tepat. Giant akan menghancurkan bangunan pertahanan dan saat lawan sibuk melawan Giant, Minions akan membuat kerusakan pada bangunan yang tidak pemiliki perlindungan. Minions harus waspada terhadap Wizard Towers, karena Wizard Towers dapat menghancukan sekelompok Minions.

12. Hog Riders

Gambar 2.13 adalah perubahan penampilan Hog Rider di beberapa level. Hog Riders digambarkan sebagai pria berkulit gelap, kasar, menaiki babi besar, bertelanjang dada, mengenakan celana pendek kulit berwarna coklat dengan sabuk merah, memakai sandal kulit, memakai dua gelang dan anting-anting emas, bersenjatakan Warhammer besar. Bersama babinya, troop ini mampu melompati dinding lawan dan bergerak menghancurkan *base* lawan dengan cepat

Gambar 2.13 Karakter Minions di Beberapa Level
(http://clashofclans.wikia.com/)

Hog Riger akan menyerang pertahanan lawan yang terdekat darinya. Hog Rider dapat digunakan sebagai pengganti Giant karena keduanya sama-sama memiliki tubuh yang kuat. Babi tunggangan Hog Riders mampu melompati semua *level* Wall sehingga Hog Riders dapat menghancurkan semua targetnya meskipun target itu ada di dalam dinding yang tinggi. Namun meskipun Hog Riders dapat melompati semua dinding, tetapi jika Hog Riders membutuhkan *back up* dari *ground troop* lain, maka *player* harus mengirimkan Wall Breaker sebagai pembuka jalan bagi *troop* lain yang akan membantu Hog Riders. Hog Rider yang menyerang secara berkelompok akan sangat berbahaya bagi lawan, karena *troop* ini sangat kuat dan memiliki *hit point* yang tinggi. Musuh utama Hog Riders adalah Giant Bomb, oleh karena itu *player* disarankan menjauhkan diri dari Giant Bomb atau mengorbankan Barbarian atau satu Hog Rider untuk menghancurkan Giant Bomb sebelum mengirimkan Hog Rider.

13. Valkyries

Valkyrie digambarkan sebagai gadis perkasa berambut merah, mengenakan celana kulit pendek dengan juntai panjang di bagian tengahnya dan bra kulit serta

memakai sepatu boot setinggi lutut. *Troop* ini memegang kapak ganda besar. Jika pada saat awal menyerang Valkirie ditempatkan dekat dengan Barbarian King lawan, maka dia akan membuat kerusakan yang cukup besar pada Barbarian King sehingga akan memudahkan *troop* lain untuk memberikan kerusakan yang lebih parah di wilayah lawan. Valkirie mampu merusak beberapa gedung sekaligus, oleh karena itu sebaiknya player tidak menempakan beberapa gedung di satu lokasi.

14. Golems

Golem merupakan unit darat yang sangat kuat. Pada saat Golem mati, troop akan berubah menjadi dua Golemites, yaitu Golem kecil yang memiliki kekuatan seperlima dari Golem sebenarnya. Golemites akan terus melakukan kerusakan pada base lawan. Sifat serangan Golem adalah splash damage seperti Ballon.Serangannya mampu memberikan efek hancur ke banyak daerah lawan. Target utama Golem adalah pasukan perahanan lawan seperti Giant, Hog Rider dan Ballons.Cara menyerang terbaik menggunakan Golems adalah melepaskan Golem dan biarkan Golems menghancurkan pertahanan *splash damage* seperti Mortar dan Wizard Tower, kemudian kirim Archer dan Barbarian untuk menyelesaikan serangan.

15. Witches

Witch mempunyai kekuatan dapat memanggil pasukan yang sudah lama mati(Skeleton), sehingga ketika dibangunkan pasukan itu berwujud kerangka saja. Pasukan mati itu dapat digunakan sebagai pasukan pendukung (*back up*). Mereka dapat juga menghancukan lawan dengan mudah jika digunakan dengan benar. Karakter Witch adalah perempuan dengan rambut berwarna ungu yang menjuntai sampai ke bahu. Witch menggunakan hiasan bahu keemasan, sabuk emas, rok dan jubah ungu usang serta memegang tongkat berkepala tengkorak kambing gunung. Karena Witch memiliki *hit point* yang rendah, maka Witch direkomendasikan hanya sebagai *troop* pendukung dan harus diposisikan di belakang serangan. Menempatkan sekelompok Witch pada saat pertempuran terjadi akan memudahkan Mortar menghancurkan semua Witch dalam kelompok itu. Skeleton yang dihidupkan oleh Witch dapat digunakan untuk menarik perhatian bangunan pertahanan.

16. Lava Hounds.

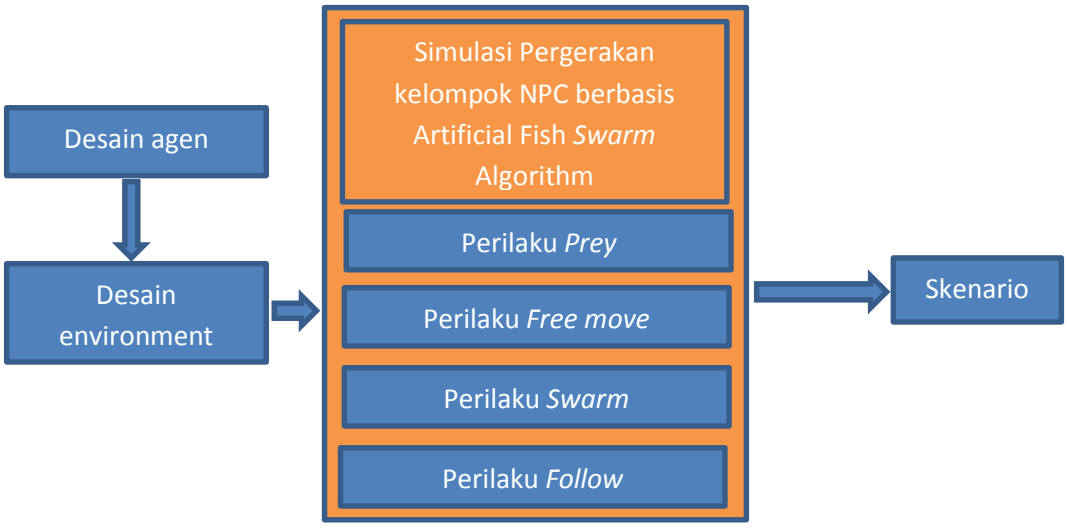
Target Lava Hounds adalah menghancurkan Air Defense *base* lawan. Ketika Lava Hound mati, dia akan pecah menjadi 8 Lava Pup di level 1, pecah menjadi 10 Lava Pup di level 2 dan menjadi 12 Lave Pup di level 3. Lava Pup mempunyai kekuatan yang cukup kuat untuk melakukan serangan. Karena Lava Hound adalah unit terbang maka Cannon dan Mortar tidak dapat menyerang mereka. Target utama Lava Hounds adalah Air Defense

BAB III
METODOLOGI PENELITIAN

3.1 Metodologi

Untuk menyelesaikan penelitian ini, diawali dengan mempelajari dan memahami beberapa teori yang berhubungan dengan konsep *artificial inteligent* dari tema dari tesis yaitu meliputi tingkah laku berkelompok kawanan ikan yang sesuai dengan *Artificial Fish Swarm Algorithm*. Pergerakan agen dalam game dibutuhkan untuk menuju sebuah sasaran baik itu berkelompok maupun bergerak individu. Penelitian ini dilakukan unutk membuat pergerakan agen yang berkelompok sesuai dengan *Artificial Fish Swarm Algorithm*. Selain untuk mensimulasikan perilaku kawanan ikan yang ada di alam, hasil dari penelitian ini dapat digunakan di dalam game untuk menggerakkan troops secara berkelompok baik dalam formasi menyerang dan menuju target maupun menghindari serangan musuh.

Gambar 3.1 adalah alur diagram alur langkah langkah pelaksanaan penelitian ini



Gambar 3.1 Diagram alur penelitian

3.2 Perancangan Sistem

Hasil dari penelitian ini dapat diterapkan untuk game yang membutuhkan pergerakan sekelompok agen yang berkelompok menuju atau bergerak ke target dan menghindari halangan ataupun musuh yang ada di sepanjang perjalanannya menuju ke tujuan. Agen akan bergerak berdasarkan perilaku yang terkondisikan karena pengaruh dari jangkuan visual dan step dari tiap agen. Perilaku akan selalu berubah berdasarkan kondisi sekitar agen, apakah ada halangan ada target.

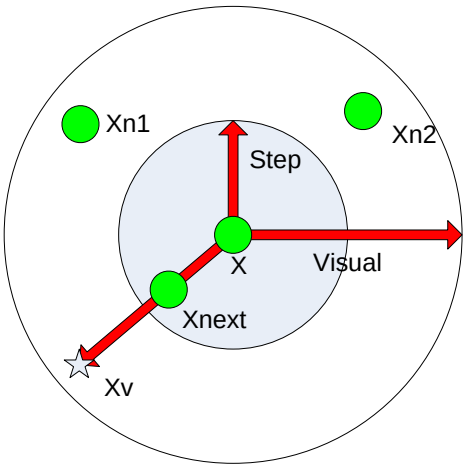
Agen di penelitian ini dalam bertingkah laku berbasis *Articial Fish Swarm Algorithm*, yang mana algoritma ini meniru tingkah laku dari ikan yang ada di alam.

Perancangan sistem dalam penelitian ini meliputi perancangan perilaku agen dalam kelompok yaitu :

1. *Prey*
2. *Swarm*
3. *Follow*
4. *Free move*

3.3 Perilaku agen berdasar Artificial Fish *Swarm* Algorithm

Pada penelitian ini agen berperilaku kelompok menurut *Articial Fish Swarm Algorithm*. Agen bergerak dan berkelompok mengikuti perilaku kelompok didalam algoritma tersebut. Proses pergerakan dari agen dalam kelompok di representasikan sebagai berikut :



Gambar 3.2 Konsep Visual Agen *Artificial Fish*

$$X_v = X_i + Visual \times rand(), i \in (0, n] \quad (3.1)$$

$$X_{next} = X + \frac{X_v - X}{||X_v - X||} \times Step \times rand() \quad (3.2)$$

Visual adalah jarak jangkauan penglihatan dari agen yang dihitung dari koordinat posisi agen sampai ke koordinat titik penglihatan terjauh yang bisa dilihat oleh agen. Step adalah panjang langkah dari agen untuk menuju ke arah target visual yang telah ditentukan, agen akan bergerak kearah target visual atau jangkauan visual maksimum yang bisa dilihat oleh agen dengan cara melangkah sesuai panjang langkahnya. Crowd (δ) adalah faktor kejenuhan kerumunan yang merupakan faktor yang menentukan banyaknya individu agen yang menjadi anggota kerumunan. Jika kerumumunan sudah mempunyai anggota maksimum yang ditentukan oleh δ maka individu agen lain yang belum masuk ke kerumumanan tidak akan bisa menjadi anggota kerumunan tersebut. X_v adalah posisi visual pada saat tertentu. X_{next} adalah posisi terbaru dari agen saat melangkah sebanyak n Step menuju jangkauan maksimum visual dari agen. Variabel rand() menghasilkan angka acak antara 0 dan 1. X_{n1} dan X_{n2} adalah agen lain yangberada disekitar agen X

Perilaku kelompok ikan dalam sebuah kawanan ikan, menurut AFSA meliputi perilaku sebagai berikut :

1. Perilaku memangsa (*prey*)
2. Perilaku berkerumun (*swarm*)
3. Perilaku mengikuti (*follow*)
4. Perilaku bergerak (move)

3.3.1 Perilaku memangsa (*prey*)

Perilaku ini adalah perilaku dasar untuk mendapatkan makanan. Jika X_i adalah keadaan agen saat ini dan pilih keadaan X_j secara random pada jarak visualnya. Dan Y adalah konsentrasi makanan. Jika X_j lebih unggul dari X_i maka agen akan bergerak ke X_j, kebalikannya, jika dipilih keadaan X_j secara random dan tentukan apakah memenuhi kondisi maju, ulangi beberapa kali, jika tetap tidak memenuhi kondisi maju, maka agen akan bergerak satu langkah secara acak [8].

$$X_j = X_i + Visual \times rand() \quad (3.3)$$

Jika $Y_i < Y_j$ maka akan bergerak selangkah ke arah ini

$$X_i^{i+1} = X_i^t + \frac{X_j - X_i^t}{||X_j - X_i^t||} \times Step \times rand() \quad (3.4)$$

3.3.2 Perilaku Berkerumun (*swarm*)

Saat X_i adalah keadaan saat ini dari agen, X_c adalah posisi pusat dan n_f adalah jumlah dari agen yang berada pada jangkauan visual agen yang lain atau agen yang saling bisa melihat agen yang lain. ($d_{ij} <$ Visual), n adalah jumlah total agen seluruhnya. δ adalah faktor kejenuhan kerumunan yaitu jumlah agen maksimum yang akan berada pada kerumumanan tersebut. δ bernilai antara 0 dan 1, semakin kecil nilai δ maka semakin sedikit agen yang bisa bersama sama menjadi anggota kerumunan.

Jika $Y_c > Y_i$ dan $\frac{n_f}{n} < \delta$

maka berarti pusat dari kerumunan akan mempunyai makanan yang lebih banyak Dan tidak terlalu penuh sesak, maka agen ikan akan menuju ke pusat kerumunan. Kebalikan dari itu adalah mengeksekusi perilaku memangsa

$$X_i^{i+1} = X_i^t + \frac{X_c - X_i^t}{||X_c - X_i^t||} \times Step \times rand() \quad (3.5)$$

3.3.3 Perilaku mengikuti (*follow*)

Jika X_i adalah keadaan saat ini dari agen dan agen ini mengeksplorasi agen mitra dalam kawanannya yg disebut X_j yang berada di sekitarnya ($d_{ij} <$ Visual),

Jika $Y_j > Y_i$ dan $\frac{n_f}{n} < \delta$

Yang berarti keadaan agen mitra di dalam kerumunannya mempunyai konsentrasi makanan yang tinggi (nilai fungsi fitnes yang lebih tinggi) dan disekitarnya tidak terlalu sesak, maka agen akan maju selangkah ke agen mitra X_j.

$$X_i^{i+1} = X_i^t + \frac{X_j - X_i^t}{||X_j - X_i^t||} \times Step \times rand() \quad (3.6)$$

3.3.4 Perilaku bergerak bebas (*free move*)

Perilaku ini dioperasikan dengan cara memilih keadaan atau posisi acak di penglihatan agen ikan. Dan menggerakkan agen tersebut menuju keadaaan atau posisi tersebut. Perilaku ini mirip dengan perilaku memangsa (*prey*).

$$X_j = X_i + Visual \times rand() \quad (3.7)$$

3.4 Penentuan nilai fitness

Penentuan nilai fitness adalah penentuan jarak terdekat dari posisi agen disetiap iterasi, perhitungan jarak ini menggunaka perhitungan jarak euclidian

Di setiap implementasi perilaku agen selalu melakukan proses scan untuk menentukan posisi terbaik berikutnya dalam jangkaun visualnya. Agen akan selalu menentukan target visualnya dengan posisi yang random. Jika target visual sudah ditentukan maka agen akan bergerak ke arah target visual sesuai panjang Proses ini dilakukan dengan cara membandingkan fitness posisi baru dengan posisi lama. Hanya posisi yang memiliki fitness lebih bagus saja yang akan dipilih oleh agen. Setiap posisi dengan fitness rendah akan diabaikan oleh agen dan agen akan diam ditempat hingga N kali. Untuk mengukur sebagus apa setiap posisi baru yang diciptakan, maka *fitness function* pada penelitian ini mempunyai 2 komponen, yaitu :

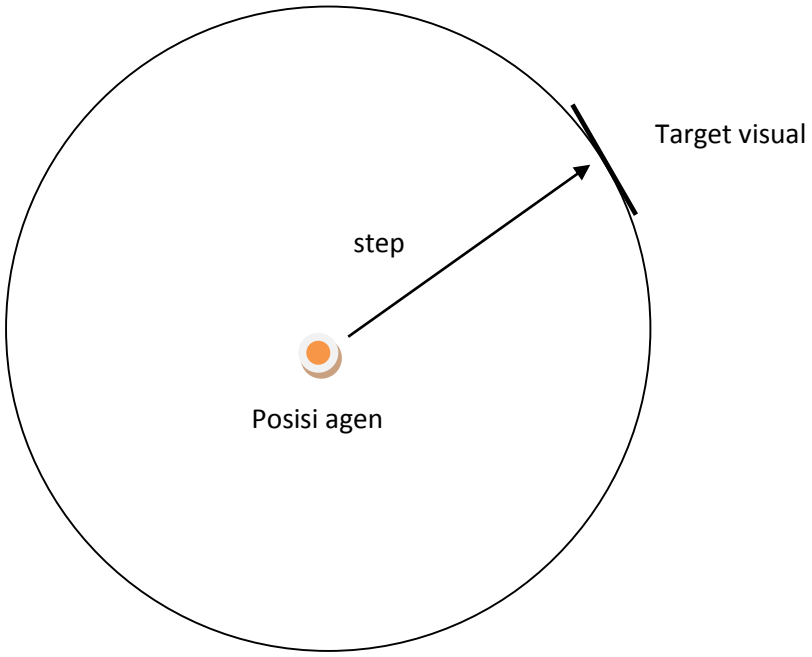
1. Fungsi objektif untuk menentukan jarak terdekat menuju target visual
2. *Constrain*, untuk menghindari tabrakan dengan agen lain dan *obstacle* statis

Untuk memenuhi fungsi objektif, maka agen harus menghitung jarak posisi baru (x'_i, y'_i, z'_i) ke posisi target (x_{ig}, y_{ig}, z_{ig}).

$D_{i\prime-g}$ adalah jarak antara posisi baru dengan posisi target visual, maka untuk mencari besarnya jarak antara posisi baru dengan posisi target adalah

$$D_{i\prime-g} = \sqrt{\left(x'_i - x_g\right)^2 + \left(y'_i - y_g\right)^2 + \left(z'_i - z_g\right)^2} \quad (3.8)$$

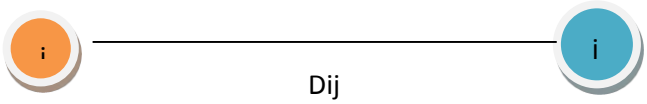
Gambar 3.3 adalah jarak ke target visual, jarak inilah yang akan menjadi fitness dari posisi agen, di mana semakin kecil jarak antara agen dengan target visual maka fitness akan dianggap semakin bagus. Jadi untuk bergerak ke arah target visual maka dilakukan minimasi nilai fitness ini.



Gambar 3.3 Jarak ke target visual

Sedangkan untuk menghindari tabrakan dengan agen lain, dievaluasi posisi tiap agen lain apakah dalam jarak yang cukup jauh atau terlalu dekat. Dua agen dikatakan bertabrakan jika jarak keduanya lebih kecil dari jumlah radius (R) ukuran 2 agen tersebut. D_{i-j}adalah jarak antara agen ke-*i* dan agen ke-*j*. Untuk menghitung jarak antara agen ke-*i* dengan agen ke-*j* adalah sebagai berikut.

$$D_{i-j} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2} \tag{3.9}$$

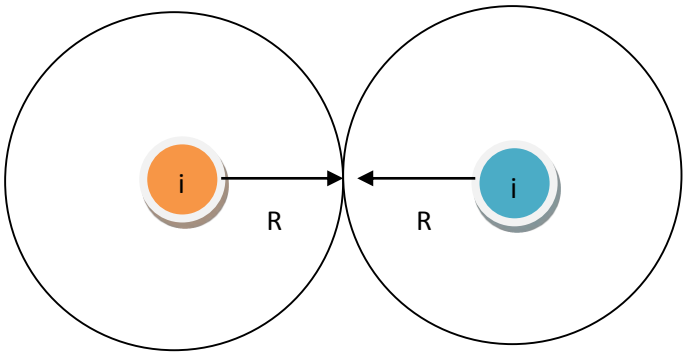


Gambar 3.4 Jarak Agen ke-i dengan Agen ke-j

Gambar 3.4 adalah ilustrasi jarak antara dua buah agen ke i dan agen ke j , Agar

dua agen tidak bertabrakan, maka jarak kedua agen itu jika dikurangkan dengan radius atau ukuran keduannya, nilainya harus lebih besar dari 0. Sehingga, jika R adalah radius atau ukuran dari agen, maka

$$D_{i-j} - 2R > 0$$



Gambar 3.5 Dua Agen Berhimpitan

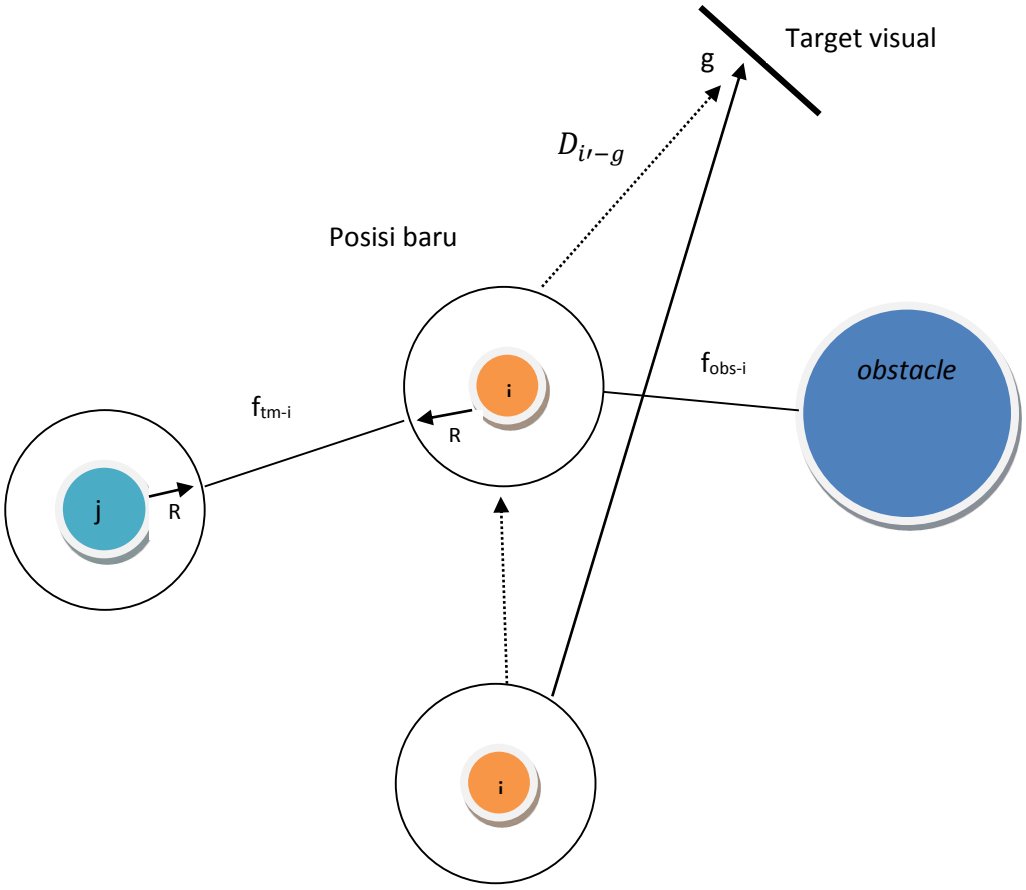
Gambar 3.5 adalah dua agen yang saling berhimpitan, dua gen dikatan berhimiptan jika $D_{i-j} - 2R = 0$ dua agen tidak boleh dalam kondisi berhimpitan karena akan bertabrakan. Untuk menghindari tabrakan dengan semua agen lain, agen ke-*i* harus mengevaluasi jaraknya dengan semua agen. Fungsi untuk menghindari agen lain (f_{tm-i}) adalah sebagai berikut

$$f_{tm-i} = \sum_{j=1}^{n} (\min(0, (D_{i-j} - 2R)))^2 \qquad (3.10)$$

Jika $(\min(0, (D_{i-j} - 2R)))$ bernilai negative itu berarti kedua agen bertabrakan. Sehingga ketika nilai itu dikuadratkan, akan menghasilkan nilai positif. Ketika nilai ini dijumlahkan dengan fungsi obyektif akan menyebabkan nilai fitness function menjadi besar. Karena nilai fitness function lebih besar dari nilai sebelumnya, maka agen tidak akan memilih posisi yang memiliki nilai fitness yang besar ini.

f_{tm-i} bernilai 0 jika semua agen berada pada jarak yang cukup berjauhan, sebaliknya bernilai positif jika ada agen yang bertabrakan. Pada penelitian ini, setiap radius (R) setiap agen diberi nilai sama dengan step. Untuk menghindari *obstacle*, hal yang sama akan diberlakukan sama seperti saat menghindari agen lain. Agar simulasi berjalan baik, diberikan bobot Δ yang bisa diset.

$$f_{obs-i} = \sum_{j=1}^{n\_obs} \frac{\Delta}{(D_{i-obs})} \quad (3.11)$$



Gambar 3.6 Ilustrasi Penghitungan Fitnes

Setelah fungsi objektif menghitung jarak target visual diketahui dan fungsi untuk menghindari agen lain dan *obstacle* telah ditentukan, maka untuk mencari nilai fitness dari posisi baru adalah sebagai berikut :

$$fit(p_i') = D_{i\prime-g} + f_{tm-i} + f_{obs-i} \quad (3.12)$$

dimana, p_i' adalah posisi baru agen ke-i, $D_{i\prime-g}$ menghitung jarak posisi baru agen ke target, f_{tm-i} untuk mengevaluasi apakah agen bertabrakan dengan agen lain dan f_{obs-i} untuk mengevaluasi apakah posisi baru agen adalah posisi *obstacle*. Semakin rendah nilai *fitness* menandakan agen dengan posisi barunya sudah mendekati target sekaligus dan posisi baru bukan posisi agen lain atau posisi

obsctacle. Agen akan selalu memilih fitness yang rendah karena fitness yang rendah adalah posisi yang bebas dalam arti tidak menabrak agen lain dan *obstacle*

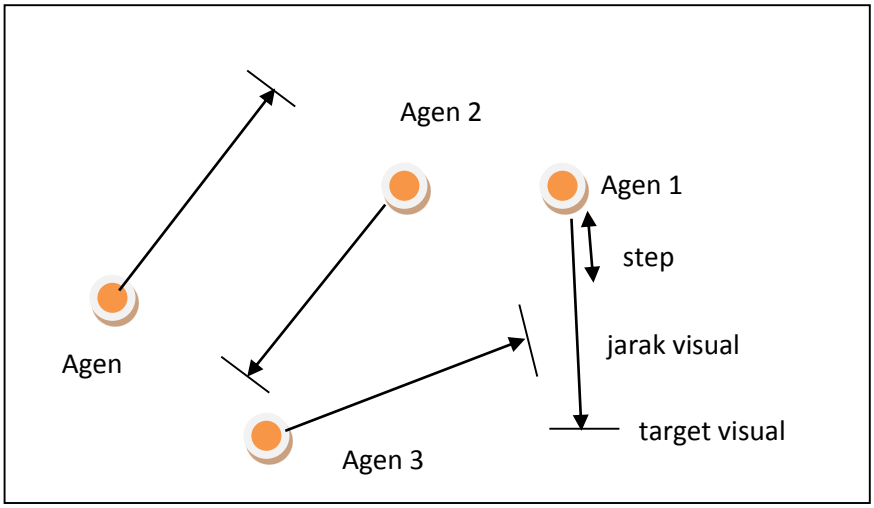
3.5 Desain Agen

Hasil dari penelitian ini selain untuk menirukan pergerakan kawanan ikan di alam juga dapat dapat diterapkan untuk pergerakan misalnya troops dalam game. NPC lain yang membutuhkan pergerakan sekelompok agen saat menyerang atau menuju target dapat menggunakan algoritma ini. Simulasi untuk agen yang berperilaku sesuai *Artificial Fish Swarm Algorithm* dilakukan dengan merubah parameter yang didesain pada agen, Tabel 3.1 adalah parameter yang didesain untuk agen, dimana perubahan parameter akan mensimulasikan agen yang berbeda beda kemampuannya. Perubahan parameter akan mempengaruhi bagaimana agen tersebut akan berperilaku selama mencari target dan menghindari halangan yang ada disekitarnya.

Tabel 3.1 Desain Agen

Parameter	**Value**	**Keterangan**
Fish Number	0-100	Jumlah agen
Limit	0-100	Jumlah percobaan pencarian jarak visual
Step	0-10	Untuk menentukan seberapa jauh agen melangkah jika mendapatkan nilai fitness terbaik
Visual	0-150	Jangkauan dari penglihatan ikan
Behavior	*Swarm* *Follow* *Free move* *Prey*	Berkerumun dalam kelompok Mengikuti bergerak dalam kerumunan Bergerak bebas jika target belum masuk jangkauan visual agen Bergerak menuju target jika target sudah masuk dalam jangkauan visual agen
Crowd factor	0-1	Faktor kerumumanan

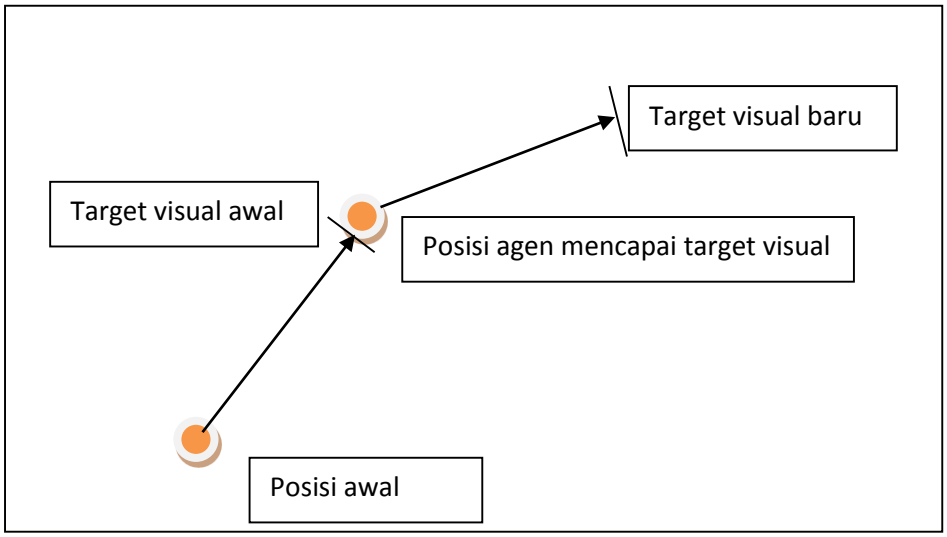
Pada tahap inisialisasi, agen akan diberi posisi acak, target visual tiap agen juga akan berada pada posisi acak. Setelah inisialisasi maka agen akan mulai bergerak kearah target visual masing denga langkah step yang sudah diset. Pada fase ini perilaku *free move* bekerja di seluruh agen.



Gambar 3.7 Inisialisasi awal agen dan perilaku freemove

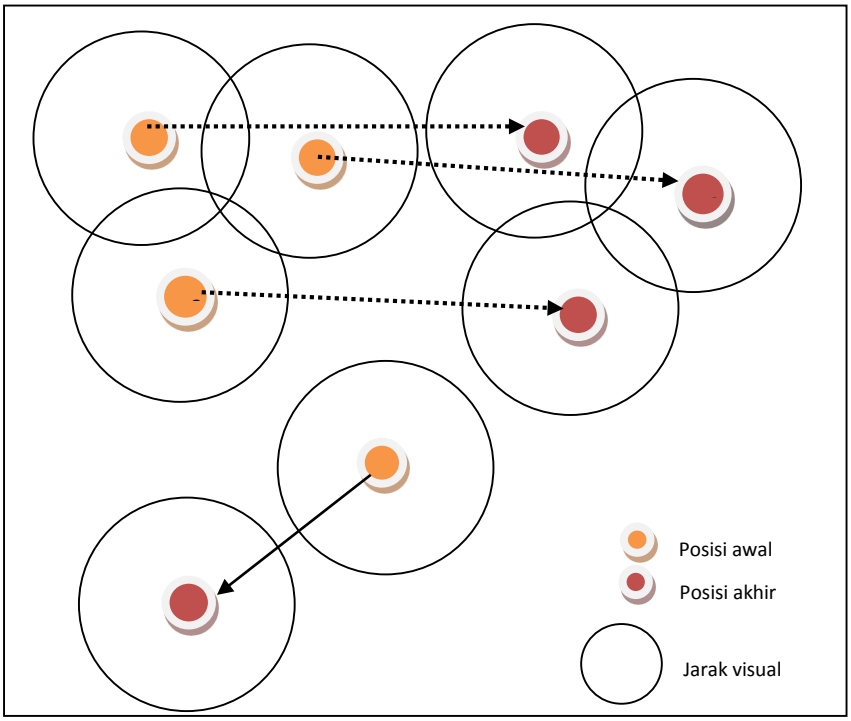
Gambar 3.7 adalah inisialisasi awal agen, dimana agen akan mendapatkan posisi yang acak, tidak sama antara individu agen yang satu dengan individu agen yang lain. Begitu juga arah penglihatan agen, yang di tandai dengan target visual, juga akan berda tiap individu agen. Agen akan bergerak acak berperilaku *free move* kearah dimana target visual tiap agen berada. Agen bergerak dengan langkah step, langkah demi langkah menuju target visual saat agen sudah mencapai target visual maka target visual akan berpindah acak ke posisi baru.

Gambar 3.8 adalah perubahan arah target visual secara acak yang akan menentukan arah pergerakan dari individu agen dan juga menentukan agen menemukan target maupun menemukan agen lain. Agen akan terus bergerak menuju target visual kemanapun target visual mendapatkan posisi barunya. Target visual berubah posisinya secara acakunutk mencari target, jika dari beberapa kali perubahan target visual belum menemukan target maka target visual akan terus berubah posisi hingga menemukan target.



Gambar 3.8 Perubahan target visual

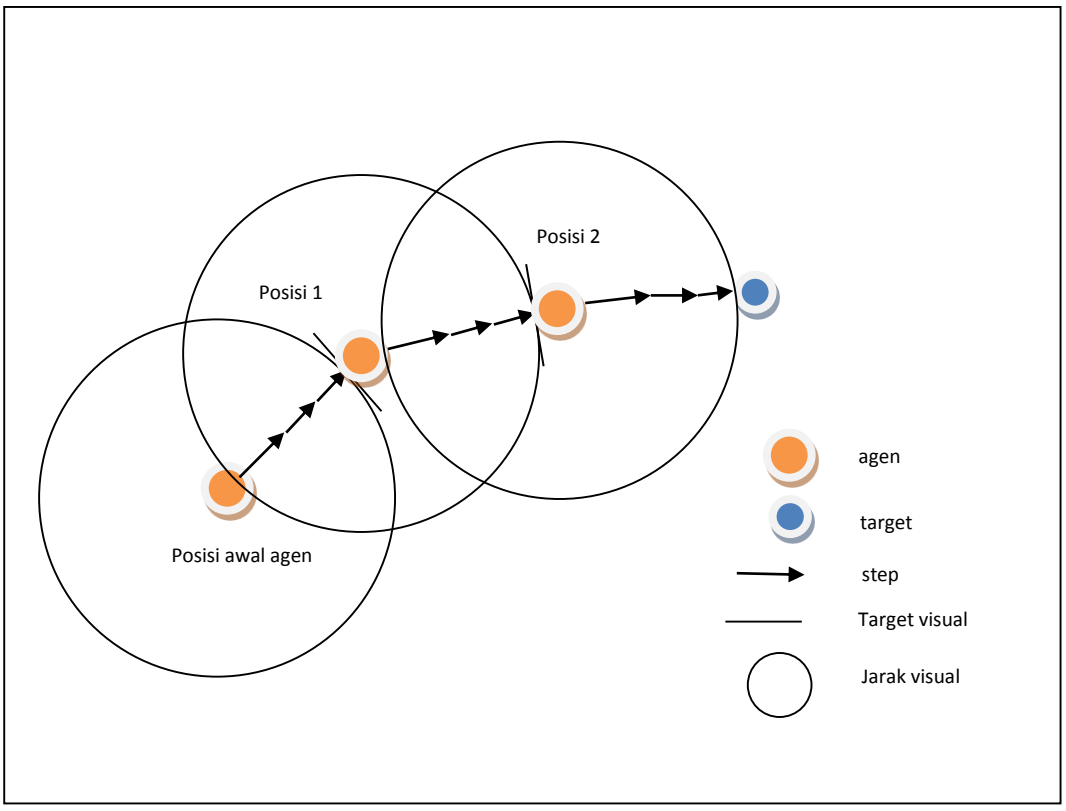
Jika dalam gerakan *free move*, jarak visual agen saling bersinggungan dengan agen lain maka agen akan berperilaku *swarm* dengan agen lain yang saling bersinggungan jarak visualnya.agen yang berperilaku *swarm* akan cenderung tertarik ke pusat kerumunan dan akan bergerak bersama-sama dalam kerumunan. Agen yang tidak bersinggungan dengan agen lain akan tetap berperilaku *free move*



Gambar 3.9 Agen yang berperilaku *swarm*

Gambar 3.9 adalah ilustrasi perilaku *swarm* saat agen bersinggungan jarak visualnya. Agen 1, 2, 3 adalah agen yang masing masing bersinggungan jarak visualnya, yang berarti agen tersebut bisa melihat agen lain dalam kerumuman di dekatnya. Jika Hal ini terjadi maka agen akan bergerak bersama- dalam kerumiumannya dengan agen lain yang berada dalam jarak visual. Agen 4 tidak berada jarak visual agen yang lain dan agen lain juga tidak berada pada jarak bvisual agen 4. Sehingga agen 4 akan tetap bergerak sendiri.

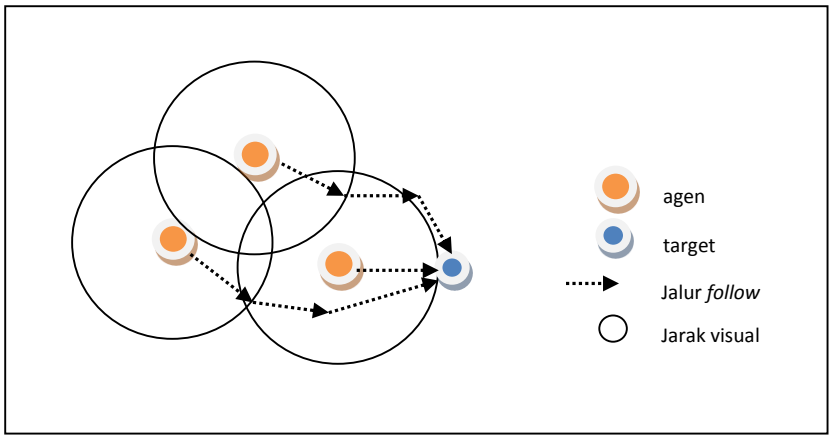
Jika ada target yang masuk ke jarak visual agen maka agen akan berperilaku *prey*. Jika target sudah masuk ke jarak visual maka target visual akan berhenti mencari posisi baru, agen akan bergerak ke arah target.



Gambar 3.10 Agen yang berperilaku *prey*

Gambar 3.10 adalal ilustrasi dari agen yang bergerak dan menemukan targetnya sehingga agen akan berperilaku *prey*. Dari posisi 1 agen bergerak ke arah target visual, ketika sampai ke target visual maka target visual akan merubah posisinya ke posisi 2. Jika pada posisi ini tareget visual akan terus bergerak acak mencari posisi

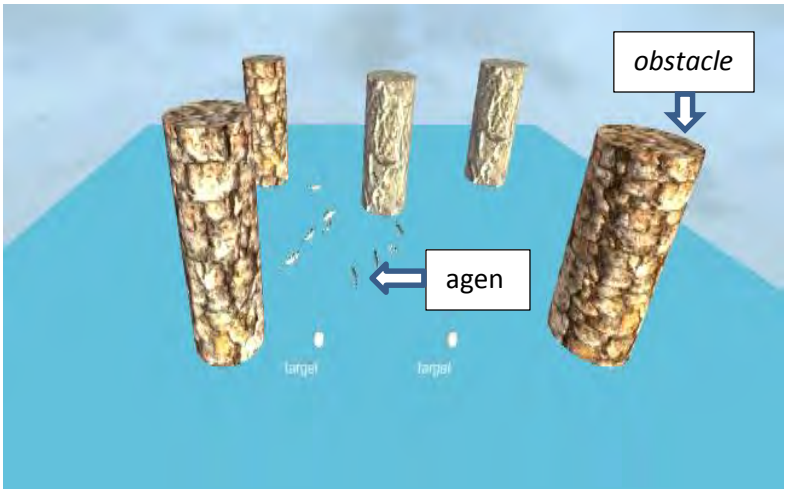
barunya dan agen akan bergerak langkah demi langkah ke arah target visual ini. Agen akan terus bergerak keposisi barunya secara acak sampai menemukan target. Jika pada posisi 2 jarak visual agen sudah menemukan target, maka target visual akan berhenti mencari posisi baru dan agen akan bergerak ke arah posisi target visual dimana disitu terdapat target tujuan.



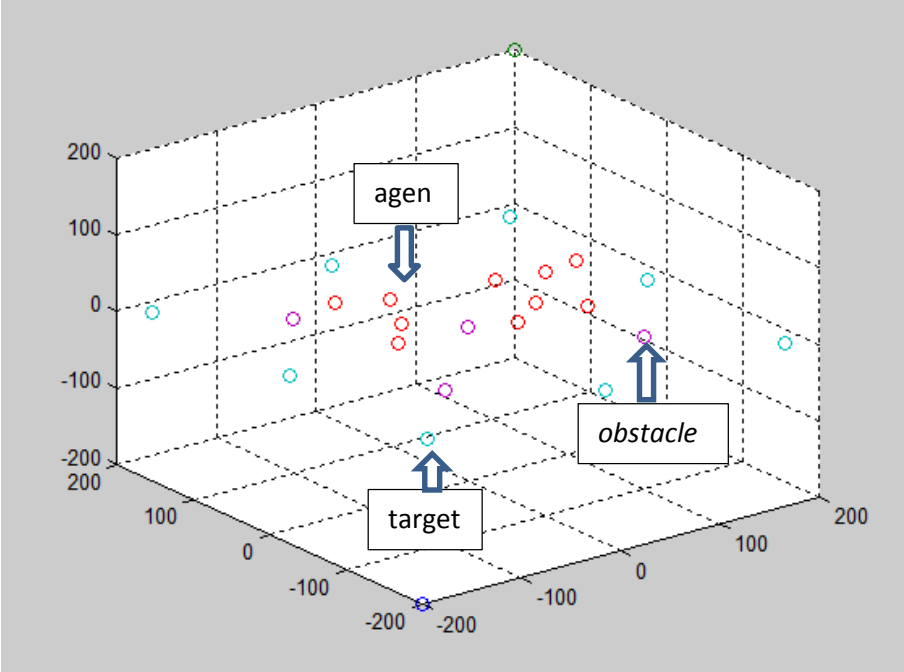
Gambar 3.11 Agen yang berperilaku *follow*

Gambar 3. 11 adalah ilustrasi dari agen yang berperilaku *follow*. Peilaku foloow akan terjadi jika sejumlah agen sedang berkerumun dan salah satu atau beberapa agen jarak visualnya menemukan target. Jika target sudah ditemukan maka agen tersebut akan menuju ke arah target dan diikuti oleh agen lain yang berada di kerumunan

3.6 Desain Environment



Gambar 3.12 Desain Environment dalam Unity 3D



Gambar 3.13 Desain Environment dalam matlab

Gambar 3.12 adalah agen bergerak di ruang 3 dimensi bergerak didalam environmentnya unutk simulasi di unity 3D dan gambar 3.13 adalah desain environment di Matlab. Dalam desain environmet tersebut, didalamnya terdapat *obstacle* yang bersifat statis dan ada beberapa target yang juga bersifat statis. Agen akan bergerak mencari target secara acak dan menemukan target yang pertama kali dilihatnya dan menhindari *obstacle* yang ada di sepanjang jalur pencarian target.

3.7 Skenario Percobaan

Tujuan dari penelitian ini adalah memperoleh sekelompok NPC yang mampu bergerak otonom menuju target dengan dengan tanpa menabrak agen lain dalam keterbatasan jangkauan visual agen dan mampu menghindari halangan (*obstacle*) statis yang ada di jalur pencarian targetnya. Untuk memenuhi tujuan tersebut, penelitian ini melakukan percobaan dengan 3 skenario yaitu, pergerakan agen dari posisi asal menuju ke posisi target tanpa ada halangan, pergerakan agen dari posisi asal menuju ke posisi target dengan ada halangan, pergerakan agen dari posisi asal menuju ke posisi target penambahan jumlah agen 10-100.

BAB IV
HASIL PENELITIAN DAN PEMBAHASAN

Pergerakan sekelompok agen yang telah diuraikan pada Bab 3 diimplementasikan dalam simulasi komputer dengan menggunakan Matlab R2013 untuk menganalisis data dan implementasi dan visualisasi dalam game menggunakan unity 5.0.1f1. Simulasi ini melibatkan 10 agen (Fish_Number), beberapa target diam di beberapa koordinat, dan beberapa *obstacle* statis. Limit pencarian posisi baru untuk agen adalah 5, artinya jika lebih dari 5 kali percobaan pencarian posisi baru tidak ditemukan kandidat posisi baru untuk agen, maka pencarian jarak visual akan mencarikan posisi baru secara acak.

4.1 Pergerakan agen tanpa halangan (*obstacle*)

Percobaan pertama dilakukan untuk menguji kemampuan agen dalam bergerak dan mencari target dengan formasi acak menggunakan *Articial Fish Swarm Algorithm*. Dalam simulasi ini, lingkungan yang digunakan adalah lingkungan 3 Dimensi berupa sebuah kubus yang didalamnya terdapat 10 agen, dan 8 target. Tujuan utama dari simulasi ini adalah menggerakkan sekelompok agen dari posisi awal dan bergerak secara acak sampai menemukan target.

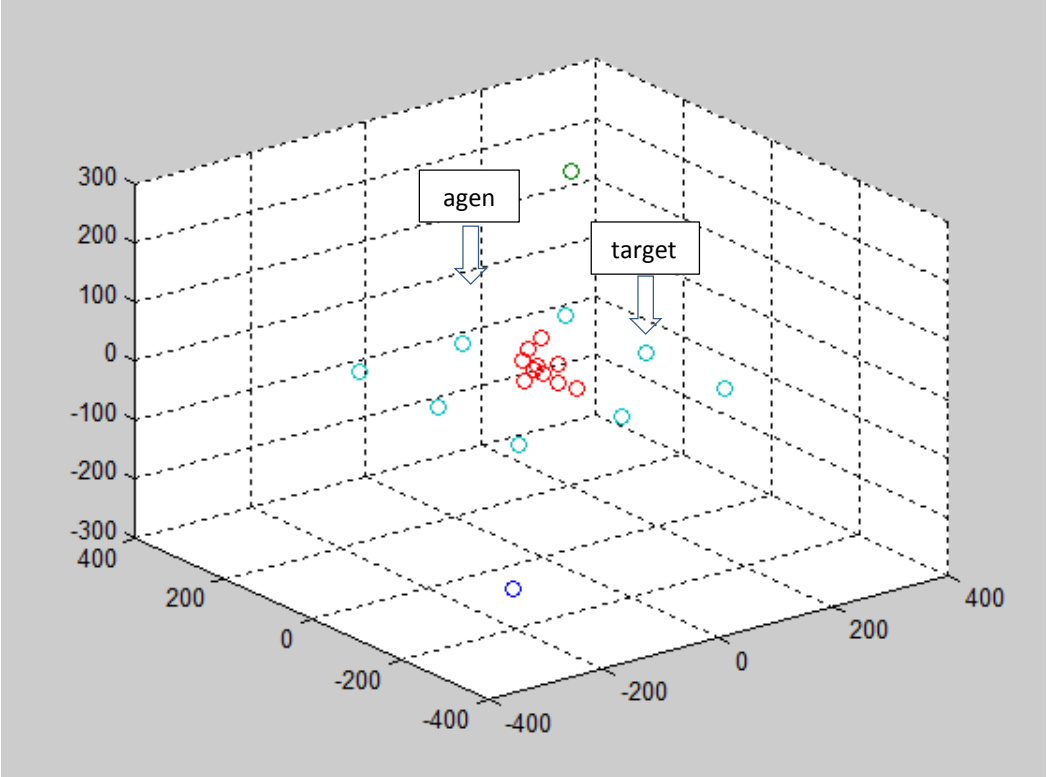
Untuk pengujian agen diberi nilai parameter seperti yang ditunjukkan pada tabel 4.1. Agen dibuat menjadi Agen A, Agen B, Agen C. dari nilai parameter yang diberi akan diamati hasilnya.

Tabel 4.1 Parameter Pengujian

Parameter	Agen A	Agen B	Agen C
Fish Number	10	10	10
Step	1	2	2
Visual	100	100	100
Crowd factor	0,4	0,4	0.2
Max Iterasi	1000	1000	1000

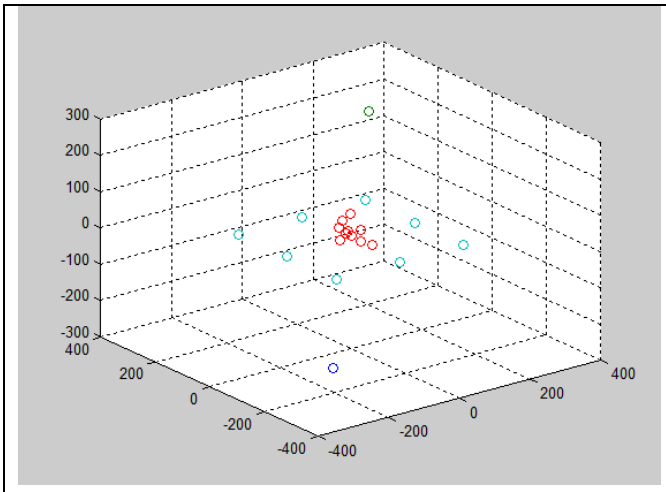
Tabel 4.1 adalah nilai parameter pengujian yang diberikan untuk agen. Agen dibedakan menjadi Agen A, Agen B, Agen C. Dari nilai parameter yang diberikan ke masing masing jenis agen akan diuji dan diamati hasilnya.

Untuk mendapatkan data dan mengolahnya digunakan simulasi dengan menggunakan matlab. Agen dalam simulasi matlab divisualisasikan dengan lingkaran berwarna merah. Target divisualisasikan dengan warna cyan. Agen untuk percobaan ini berjumlah sepuluh dengan inisialisasi awal secara acak.

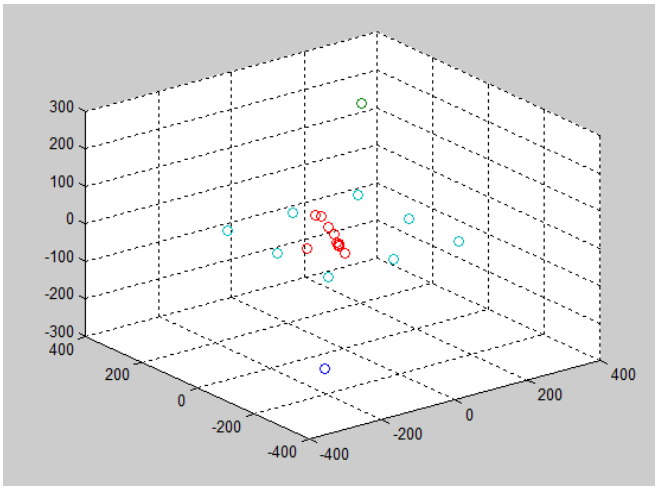


Gambar 4.1 Lingkungan 3D Simulasi Agen

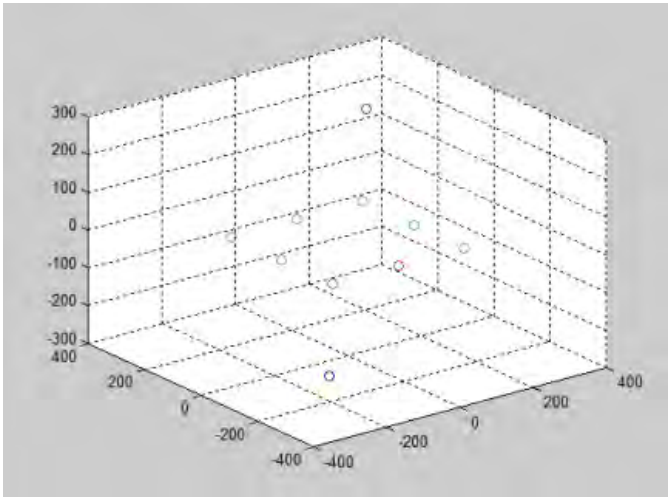
Gambar 4.1 adalah hasil dari simulasi lingkungan 3D pengujian agen yang bergerak ke target tanpa ada halangan. Agen untuk percobaan ini berjumlah sepuluh dengan inisialisasi awal secara acak. Agen divisualisasikan dengan lingkaran berwarna merah. Target divisualisasikan dengan warna cyan. Pewarnaan dimaksudkan untuk memudahkan mengamati perbedaan antara agen dan target.



a. Posisi awal agen



b. Posisi bergerak mencari target



c. Posisi menemukan target

Gambar 4.2 Simulasi pergerakan agen tanpa halangan (*obstacle*)

Gambar 4.2a adalah posisi awal agen diinisialisasi dengan posisi acak berdasarkan perilaku *free move*, ketika agen mulai bergerak agen berperilaku *free move* dan *swarm*. Selama agen secara visual atau jankauan visualnya belum menemukan target maka agen akan terus bergerak dengan perilaku *free move*. Gambar 4.2b adalah agen yang mulai bergerak mencari target dan menuju kearah target. Agen akan melakukan *swarm* dengan agen yang berada pada jangkauan visualnya, sehingga ketika ada beberapa agen yang berada pada jangkauan visual dari agen-agen tersebut maka perilaku *swarm* akan berlaku. Agen yang berada di luar jangkauan visual akan tetap bergerak sendiri dengan perilaku *free move*. Jika salah satu agen atau beberapa agen menemukan target maka agen tersebut akan menuju ke target dan diikuti oleh agen lain yang berada dalam jangkauan visualnya atau yang berada dalam kerumunannya. Disaat seperti ini maka perilaku yang berlaku adalah perilaku *prey* dan *follow*. Jika agen menemukan target tanpa ada agen lain yang berada dalam jangkaiuan visualnya maka agen tersebut akan tetap berperilaku *prey* sendiri tanpa diikuti agen yang lain. Gambar 4.2c adalah agen yang sudah menemukan target dan memposisikan dirinya mengerumuni target yang ditemukannya. Agen akan menemukan target yang paling cepat ditemukan oleh jangkauan visualnya. Target ditemukan saat jarak target ke agen lebih kecil dari parameter visual agen dan sudah masuk ke jarakvisual agen. Jika jarak target sudah lebih kecil dari visual agen tapi belum masuk ke jarak visual agen maka target belum teridentifikasi.

Agen bergerak dengan jangkauan visual yang terbatas. Setelah jangkauan visual ditentukan maka agen akan bergerak kearah titik visual denga step yang telah ditentukan. Setelah sampai di titik visual maka jangkauan visual akan kembali mencari posisi baru secara acak. Agen kembali mengulangi langkah bergerak dengan step dan begitu seterusnya berulang ulang. Jika dalam perjalanannya agen bertemu denga agen lain dalam jangkauan visual maka agen akan melakukan *swarm*. Agen akan terus melakukan pengulangan posisi jangkauan visual sampai menemukan target yang berada di jangkauan visualnya.

Untuk pengujian tanpa halangan, target dipasang statis pada posisi seprti berikut :

Tabel 4.2 Posisi target

Target	posX	posY	posZ
1	180	180	0
2	180	-180	0
3	-180	-180	0
4	-180	180	0
5	0	180	0
6	0	-180	0
7	-180	0	0
8	180	0	0

Dari hasil inisilisasi posisi awal didapat posisi agen seperti tabel 4.3, tabel 4.4 tabel 4.5

Tabel 4.3 Posisi awal agen A

Agen	posX	posY	posZ
1	69,176232	47,434894	23,242491
2	-60,729251	-23,862772	-0,500176
3	-28,240797	-48,552305	11,273561
4	-66,050805	65,033770	-41,558542
5	5,903907	-69,868423	32,093328
6	-65,754993	-43,247956	-39,074809
7	14,281096	-27,212678	-42,108607
8	9,430767	57,477197	-32,795960
9	56,490230	25,737924	-17,064530
10	2.758998	42,225816	-10,771631

Tabel 4.3 adalah hasil inisialisasi awal posisi agen A. Dari hasil inisialisasi terlihat bahwa tiap agen mempunyai posisi awal yang acak dan berbeda-beda antara individu agen yang satu dengan individu agen yang lain. Tidak ada individu agen yang mempunyai posisi sama sehingga tidak ada agen yang berhimpit atau agen yang bertabrakan pada saat inisialisai awal

Tabel 4.4 Posisi awal agen B

Agen	posX	posY	posZ
1	46,189622	65,932213	-38,961546
2	45,849890	16,831198	56,540123
3	-12,219638	-10,799615	27,384634
4	14,534491	-18,750745	-72,952232
5	-50,768327	-19,538635	51,173186
6	-54,130684	55,182915	-48,079904
7	-20,809752	-24,164896	-45,174961
8	1,083506	53,589753	65,156795
9	45,490863	53,745508	-66,398303
10	-35,249829	59,768007	-15,990129

Tabel 4.4 adalah hasil inisialisasi awal posisi agen B. Dari hasil inisialisasi terlihat bahwa tiap agen mempunyai posisi awal yang acak dan berbeda-beda antara individu agen yang satu dengan individu agen yang lain.

Tabel 4.5 Posisi awal agen C

Agen	posX	posY	posZ
1	58,348244	-67,305600	-66,998085
2	50,353133	-1,845857	48,136674
3	-51,808783	66,514228	72,945012
4	27,716664	63,048042	32,031712
5	-0,652920	-41,787995	-64,595738
6	-69,797098	18,842374	53,923926
7	-9,016972	-33,651818	-63,663185
8	18,804943	-25,673948	60,046871
9	-54,859254	-13,690683	-32,878646
10	11,720863	-8,774686	36,197393

Tabel 4.5 adalah hasil inisialisasi awal posisi agen C. Dari hasil inisialisasi terlihat bahwa tiap agen mempunyai posisi awal yang acak dan berbeda-beda antara individu agen yang satu dengan individu agen yang lain.

Dari hasil inisialisasi awal terlihat agen mendapatkan posisi awal yang acak. Tidak ada agen yang mempunyai posisi awal yang sama.

Agen dianggap mulai menemukan target dengan ditandai jarak visual dan jarak ke target yang sama. Jarak visual adalah jarak yang masuk didalam jangkauan penglihatan agen, agen akan mengenali atau mulai menemukan target jika jarak visual lebih kecil dari nilai Visual. Jarak visual akan selalu <= nilai parameter Visual. Pada pengujian ini Visual di set pada nilai 100.

Tabel 4.6 Agen A ketika mulai menemukan target

Agen	Jarak Visual	Jarak Target	Iterasi ke
1	99,817036	99,817036	260
2	99,722411	99,722411	458
3	99,685495	99,685495	485
4	99,300470	99,300470	485
5	99,348108	99,348108	831
6	96,176942	96,176942	818
7	99,288941	99,288941	288
8	60,469366	60,469366	752
9	99,346366	99,346366	413
10	98,964889	98,964889	67

Tabel 4.6 menunjukkan agen A yang mulai menemukan target dengan jarak visual yang lebih kecil dari nilai visual yang diset pada nilai 100, Jarak target juga akan sama dengan jarak visual, yang menanndakan target sudah dideteksi sehingga agen A akan mengunci posisi target dan bergerak mendekati posisi target

Tabel 4.7 Agen B ketika mulai menemukan target

Agen	Jarak Visual	Jarak Target	Iterasi ke
1	99,926369	99,926369	227
2	97,769351	97,769351	85
3	95,432774	95,432774	226
4	98,692377	98,692377	234
5	98,953764	98,953764	190
6	99,689248	99,689248	185
7	99,490728	99,490728	197
8	99,940747	99,940747	271
9	98,041942	98,041942	309
10	77,261143	77,261143	39

Tabel 4.7 menunjukkan agen B yang mulai menemukan target dengan jarak visual yang lebih kecil dari nilai visual yang diset pada nilai 100, Jarak target juga akan sama dengan jarak visual, yang menandakan target sudah dideteksi sehingga agen B akan mengunci posisi target dan bergerak mendekati posisi target

Tabel 4.8 Agen C ketika mulai menemukan target

Agen	Jarak Visual	Jarak Target	Iterasi ke
1	99,817036	99,817036	260
2	99,722411	99,722411	458
3	99,685495	99,685495	485
4	99,300470	99,300470	485
5	99,348108	99,348108	831
6	96,176942	96,176942	818
7	99,288941	99,288941	288
8	60,469366	60,469366	752
9	99,346366	99,346366	413
10	98,964889	98,964889	67

Tabel 4.8 menunjukkan agen C yang mulai menemukan target dengan jarak visual yang lebih kecil dari nilai visual yang diset pada nilai 100, Jarak target juga akan sama dengan jarak visual, yang menandakan target sudah dideteksi sehingga agen C akan mengunci posisi target dan bergerak mendekati posisi target

Dari hasil pengujian terlihat bahwa tiap agen akan berbeda beda mulai menemukan target. Dimulai dari yang pertama kali menemukan target adalah agen ke 10 pada iterasi ke 67 dan terkhair adalah agen ke 5 pada iterasi ke 835.

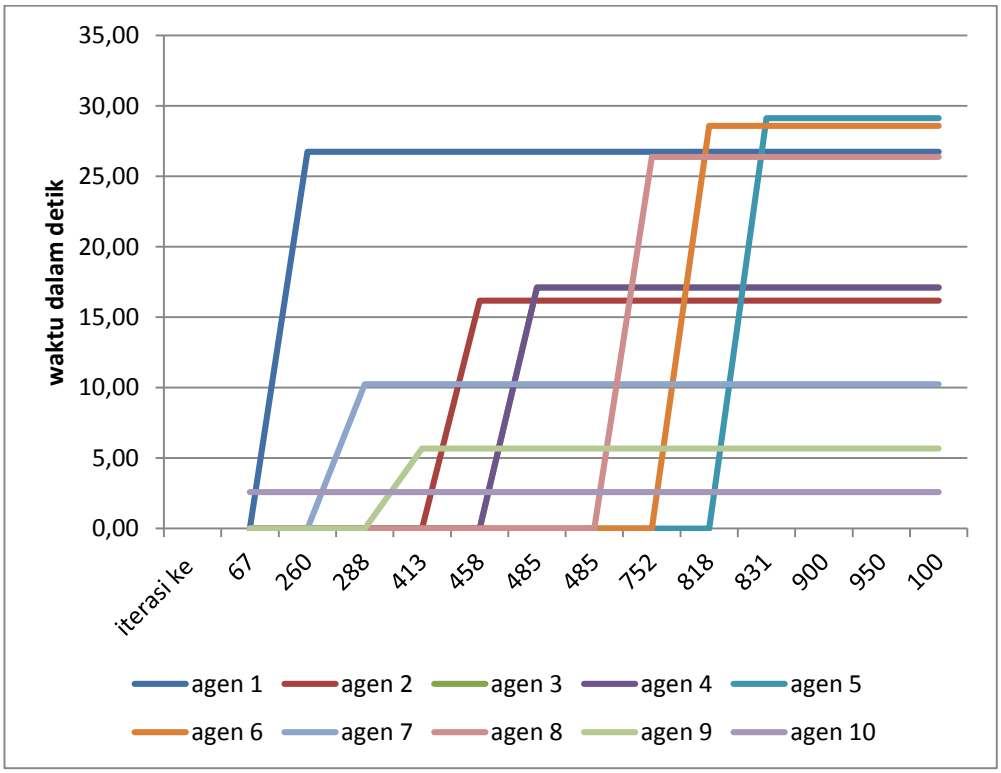
4.2 Penghitungan waktu agen menuju target keadaan tanpa halangan (*obstacle*)

Dari hasil percobaan terlihat bahwa tiap agen akan mempunyai waktu yang berbeda beda ketika menemukan target. Waktu agen menemukan taget aik secara individu maupun yang berkelompok berbeda beda tergantung seberap cepat agen melihat target yang paling cepat dilihat oleh visualnya.

Tabel 4.9 Penghitungan waktu agen A menuju target

Agen	Jarak Visual	Jarak Target	Waktu
1	84,543053	78,884921	26,729978
2	0,061078	0,061078	16,165541
3	82,206008	82,206008	17,102057
4	0,035185	0,035185	17,102073
5	0,251584	0,251584	29,121016
6	0,198989	0,198989	28,572533
7	83,170688	83,170688	10,254109
8	0,181096	0,181096	26,378879
9	95,793625	82,603153	5,682195
10	81,886548	81,886548	2,587650

Dari tabel 4.9 terlihat tidak semua agen mampu sampai ke titik target pada akhir iterasi ke 1000, agen 1, 3, 7, 9, 10 tidak mampu mendekat ke target, tapi agen agen tersebut semua sudah mampu menemukan target yang terlihat oleh jangkauan visual. Ini berarti untuk setting parameter seperti di agen A, untuk mencapai target agen membutuhkan iterasi dan waktu yang lebih banyak lagi. waktu minimum 2,587650 detik dan waktu maksimum 29,121016 detik.



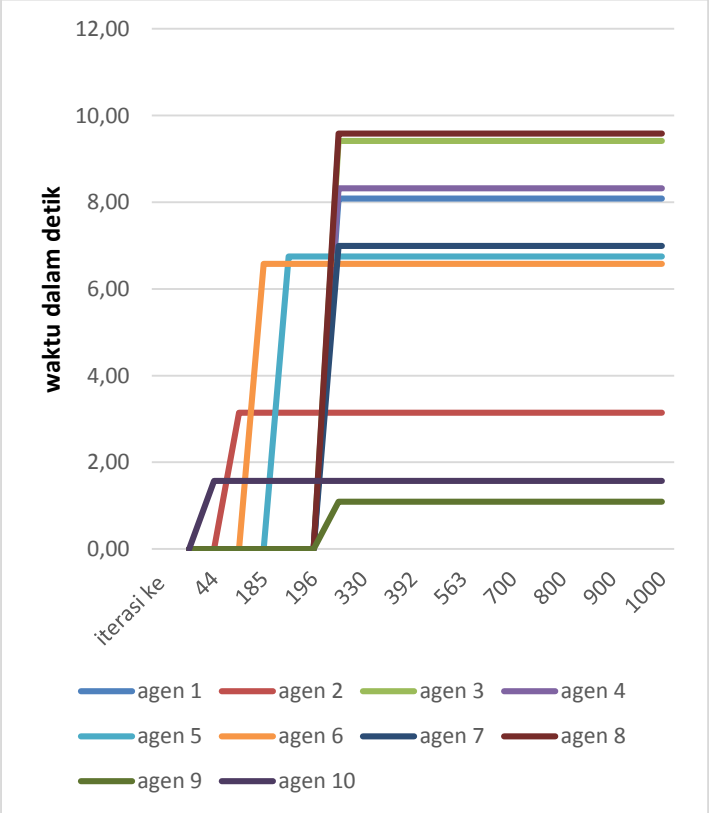
Gambar 4.3 Grafik waktu agen A menemukan target

Gambar 4.3 adalah grafik waktu agen A pertama kali menemukan targetnya, terlihat bahwa agen 10 pertama kali menemukan target pada iterasi ke 67. Pada iterasi ke 831 semua agen sudah menemukan target.

Tabel 4.10 Penghitungan waktu agen B menuju target

Agen	Jarak Visual	Jarak Target	Waktu
1	0,203576	0,203576	8,083864
2	0,098770	0,098770	3,147624
3	0,113170	0,113170	9,415153
4	0,101664	0,101664	8,323225
5	0,135662	0,135662	6,745736
6	0,208639	0,208639	6,576508
7	0,172742	0,172742	6,995317
8	0,117530	0,117530	9,583430
9	0,182293	0,182293	0,908368
10	0,079649	0,079649	1,574404

Dari tabel 4.10 terlihat semua agen mampu sampai ke titik target degan ditandai jarak visual dan jarak target yang mendekati 0 pada akhir iterasi ke 1000. Dengan waktu minimum 0,908368 detik dan maksimum 9,415153 detik.



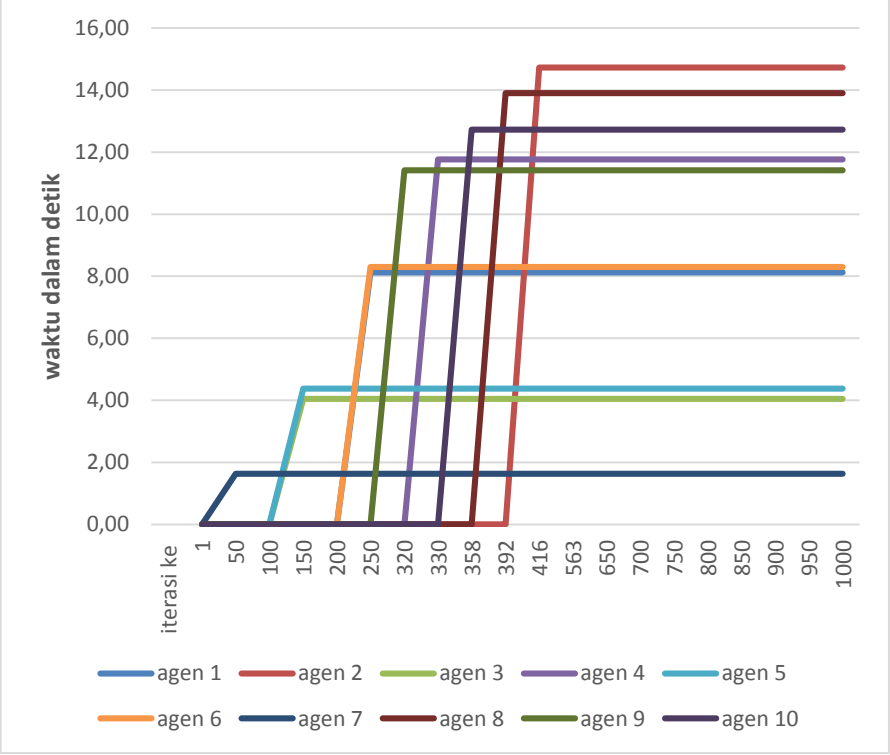
Gambar 4.4 Grafik waktu agen B menemukan target

Gambar 4.4 adalah grafik waktu agen B pertama kali menemukan targetnya, terlihat bahwa agen 10 pertama kali menemukan target pada iterasi ke 44. Pada iterasi ke 268 semua agen sudah menemukan target.

Tabel 4.11 Penghitungan waktu agen C menuju target

Agen	Jarak Visual	Jarak Target	Waktu
1	0,050631	0,050631	8,123018
2	0,15481	0,15481	14,72455
3	0,089482	0,089482	4,047305
4	0,126349	0,126349	11,76306
5	0,114498	0,114498	4,380678
6	0,17408	0,17408	8,295295
7	0,129542	0,129542	1,62867
8	0,091805	0,091805	13,89988
9	0,064622	0,064622	11,41907
10	0,056687	0,056687	12,72971

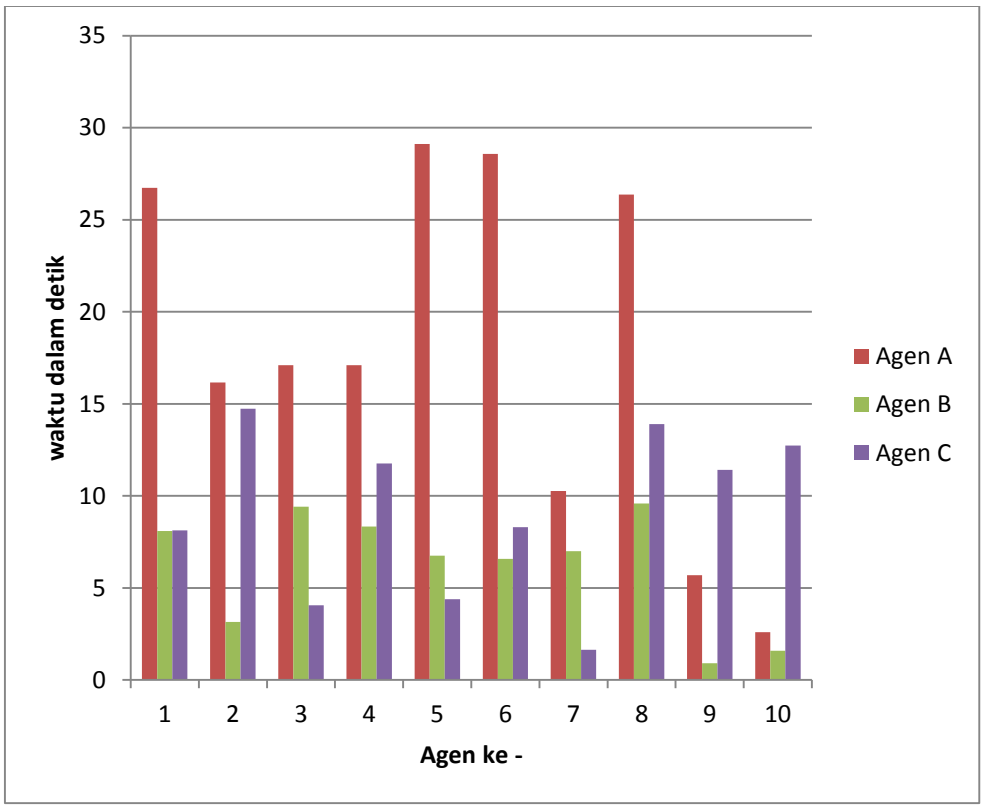
Dari tabel 4.11 terlihat semua individu agen C mampu sampai ke titik target dengan ditandai jarak visual dan jarak target yang mendekati 0 pada akhir iterasi ke 1000. Dengan waktu minimum 4,047305 detik dan maksimum 13,89988



Gambar 4.5 Grafik waktu agen C menemukan target

Gambar 4.5 adalah grafik waktu agen C pertama kali menemukan targetnya, terlihat bahwa agen 10 pertama kali menemukan target pada iterasi ke 50. Pada iterasi ke 416 semua agen sudah menemukan target.

Gambar 4.6 adalah grafik perbandingan waktu ketiga agen dalam bergerak menmukan target di lingkungannya. Terlihat bahwa agen A mempunyai waktu yang lebih lama dibanding agen B dan C. Parameter step agen A yang bernilai satu dan parameter step agen B, C bernilai 2 mempengaruhi lama dari agen untuk bergerak menemukan target. Agen A akan lebih lama karena melangkah lebih banyak dengan step yang lebih kecil



Gambar 4.6 Grafik perbandingan waktu agen A, B, C

4.3 Penentuan posisi agen pada akhir maksimum iterasi keadaan tanpa halangan (*obstacle*)

Agen akan berhenti bergerak ketika sudah menemukan target atau iterasi maksimum sudah tercapai. Agen akan menuju ke target yang paling cepat di

jangkau oleh jangkauan visualnya. Agen akan bergerak ke arah target yang terdekat dengan agen tersebut.

Tabel 4.12 Posisi akhir agen A pada saat menemukan target

posX	posY	posZ	Target ke
-175,067194	78,638115	-3,813790	7
-179,977405	179,977477	0,052084	4
-172,475767	81,379755	8,862794	7
-180,022457	180,004989	0,026623	4
-180,002514	180,098397	0,231530	4
-179,822876	179,951474	0,076608	4
-81,640739	164,173186	1,290363	5
-179,959148	179,947739	0,168510	4
-80,160412	163,575282	11,305659	5
-171,952002	80,974098	9,155979	7

Tabel 4.12 menunjukkan bahwa pada posisi akhir maksimum iterasi ke 1000 agen A berkerumun pada tiga posisi target yaitu pada target ke 4, 5, 7. Disini terlihat bahwa agen berperilaku *swarm* dan *follow* saat bergerak mencari target. Agen A akan cenderung mengikuti agen lain yang terjangkau oleh jangauan visualnya, sehingga membentuk kerumunan. Agen ke 2, 4,5,6,8 berkerumun di posisi target ke 4, Agen ke 7, 9 berkerumun di posisi target 5. Agen ke 1, 3, 10 berkerumun di posisi target 7. Agen cenderung lebih menyebar ke berbagai target.

Tabel 4.13 Posisi akhir agen B pada saat menemukan target

posX	posY	posZ	Target ke
179,868133	0,154648	-0,011762	8
179,905389	-0,013999	0,024665	8
179,968359	-0,080197	-0,073312	8
179,902989	0,002960	0,030260	8
180,009937	-0,115751	0,070050	8
179,938067	-0,164985	0,111689	8
180,045534	0,084637	-0,143537	8
180,014490	-0,079046	0,085762	8
179,932464	-0,058669	0,158832	8
179,988722	-0,046729	0,063507	8

Tabel 4.13 adalah tabel yang menunjukkan posisi akhir dari agen B yang semua agennya berhasil menuju target. Dengan step yang lebih besar dibanding agen A terlihat bahwa agen B akan cenderung berkerumun menjadi 1 dan menuju ke arah satu target yaitu target ke 8.

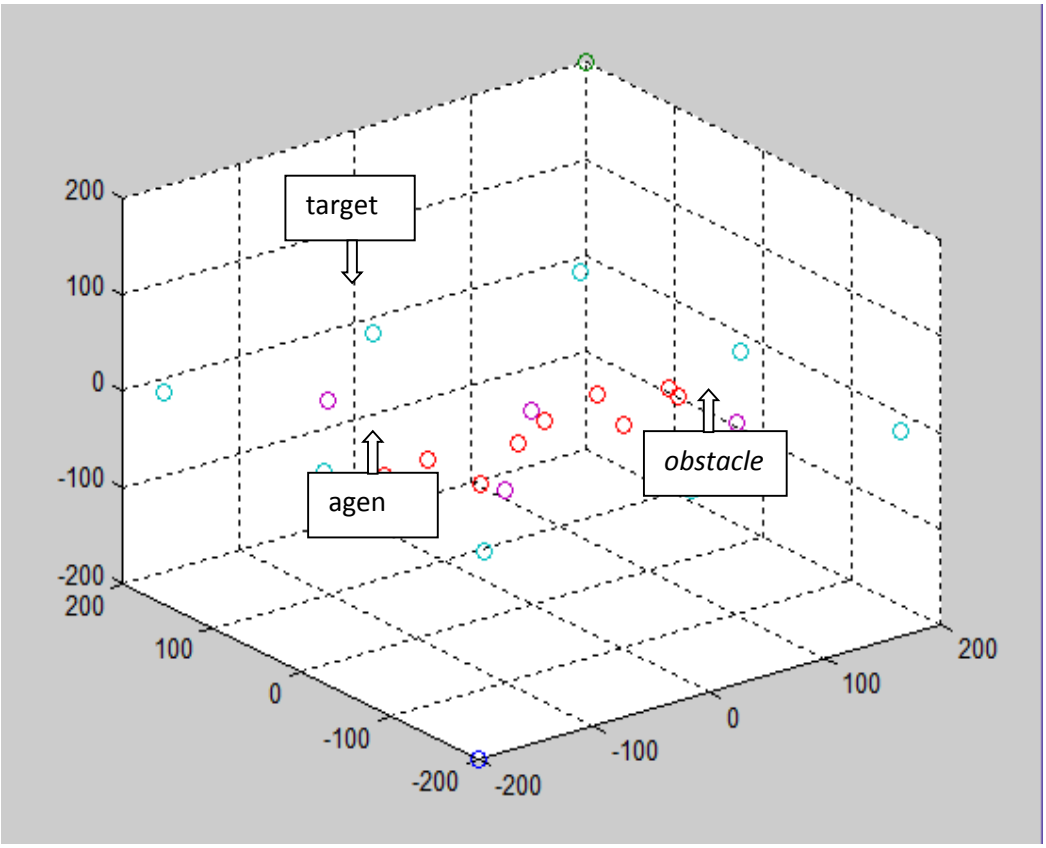
Tabel 4.14 Posisi akhir agen C pada saat menemukan target

posX	posY	posZ	Target ke
-0,027849	-180,001624	-0,042253	6
0,133348	-179,940572	0,051505	6
-0,087635	-179,981952	0,001181	6
0,036892	-179,920959	0,091409	6
-0,014411	-179,977614	-0,111359	6
0,027122	-180,169831	-0,026941	6
-179,896578	-0,076435	0,015583	7
-0,028235	-179,923245	-0,041709	6
-0,037414	-180,03965	-0,034701	6
-0,01277	-179,947704	-0,017761	6

Tabel 4.14 adalah tabel yang menunjukkan posisi akhir dari agen C yang semua agennya berhasil menuju target. Dengan crowd yang lebih kecil dibanding agen B terlihat bahwa agen C akan cenderung tidak berkerumun menjadi 1 dan menuju ke arah bebeberapa target yaitu target ke 6 dan 7.

4.4 Pergerakan agen dengan halangan (*obstacle*)

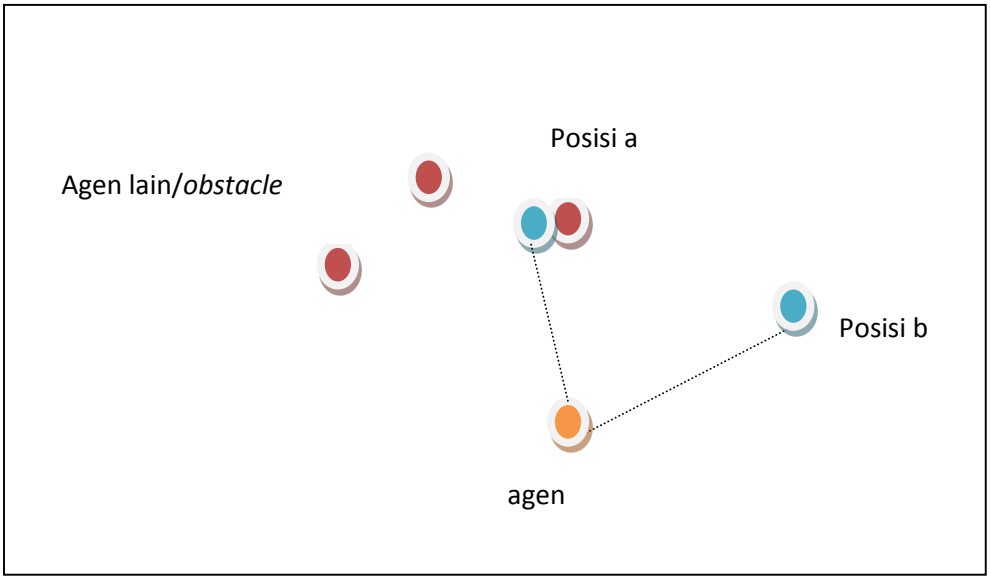
Dalam perjalanannya menuju ke target agen akan berhadapan atau dihalangi oleh *obstacle*. Agen harus mendatangi target dengan memperhatikan adanya *obstacle* dan berusaha menghindarinya. Saat bergerak menghindari *obstacle*, agen juga harus tetap memperhitungkan posisi agen lain agar tidak terjadi benturan antar agen. Tiap agen harus menjaga jarak dirinya dari agen agen lain dan juga *obstacle* agar tidak terjadi tabrakan.



Gambar 4.7 Lingkungan 3 dimensi agen dengan 4 *obstacle*

Gambar .4.7 adalah lingkungan pengujian agen yang bergerak *obstacle* di dalamnya. Agen adalah bulatan yang bewarna merah dan *obstacle* adalah bulatan berwarna ungu, target bewarna cyan. Agen dirancang untuk dapat bergerak mencar target tanpa menabrak *obstacle* yang dilewatinya sepanjang pencariannya terhadap target. Agen juga tirancang untuk tidak bertabrakan dengan sesama agen.

Prosedur pengecekan tabrakan denga agen lain maupun dengan *obstacle* adalah sebagai berikut, agar tidak terjadi tabrakan adalah dengan mengecek posisi baru suatu agen sama dengan posisi agen lain atau tidak. Jika posisinya sama maka agen tersebut tertahan dan memperbaharui posisinya dan menunggu iterasi berikutnya untuk menentukan calon posisi baru yang tentunya juga harus dicek apakah menabrak agen lain atau tidak. Jika calon posisi baru benar-benar kosong maka agen tersebut dapat memperbaharui posisinya ke tempat yang baru.



Gambar 4.8 Pengecekan tabrakan dengan agen lain /*obstacle*

Gambar 4.8 adalah prosedur pengecekan pemilihan posisi agen jika akan berpindah posisi dengan adanya agen atau *obstacle*. Pada posisi a posisi baru agen akan bertabrakan dengan agen lain atau *obstacle* yang berada di posisi situ. Agen akan tetap berhenti disitu dan melnjutkan melakukan iterasi sampai diperoleh posisi yang tidak bertabrakan dengan agen lain atau *obstacle*. Pada posisi b agen tidak akan bertabrakan dengan agen lain atau *obstacle* maka agen berpindah ke posisis b.

Untuk pengujian Agen A, Agen B, Agen C diberi nilai parameter seperti yang ditunjukkan pada tabel 4.15. Dari parameter tersebut akan diamati hasil pengujian.

Tabel 4.15 Parameter Pengujian dengan *obstacle*

Parameter	**Agen A**	**Agen B**	**Agen C**
Fish Number	10	10	10
Step	2	2	2
Visual	50	100	150
Crowd factor	0,4	0,4	0.4
Max Iterasi	1000	1000	1000

Tabel 4.15 adalah parameter pengujian dengan memperhitungkan *obstacle* yang ada di sepanjang jalur pencarian.

Tabel 4.16 Posisi target pada pengujian dengan *obstacle*

Target	**posX**	**posY**	**posZ**
1	180	180	0
2	180	-180	0
3	-180	-180	0
4	-180	180	0
5	0	180	0
6	0	-180	0
7	-180	0	0
8	180	0	0

Tabel 4.16 adalah posisi target pada pengujian dengan obstacele. Ada 8 posisi target yang dipasang pada lingkungan dimana agen bergerak. Agen dirancang untuk bergerak menuju target yang terjangkau oeh jangkauan visualnya. Jadi agen tidak akan pernah tahu kemana target yang akan dituju, agen hanya menuju target berdasar target yang pertama kali atau paling cepat masuk ke jangkauan visual agen.

Tabel 4.17 Posisi *obstacle* pada pengujian dengan *obstacle*

Obstacle	**posX**	**posY**	**posZ**
1	0	0	0
2	100	-100	0
3	-100	-100	0
4	-100	100	0

Tabel 4.17 adalah posisi *obstacle* yang diletakkan pada lingkungan dimana agen bergerak mencari target.

Dari hasil inisialisasi posisi awal didapat posisi agen seperti tabel 4.18, tabel 4.19 tabel 4.20

Tabel 4.18 Posisi awal agen A

Agen	posX	posY	posZ
1	-13,9623	-4,16591	-63,4585
2	12,20828	-39,7549	63,51646
3	-28,867	-60,794	38,61405
4	0,684249	63,8507	-39,5207
5	-59,7802	45,21039	36,9491
6	-52,6733	-8,8567	72,59504
7	39,24631	-3,937	20,63036
8	20,46009	-39,6806	-17,7879
9	16,57874	2,500911	-14,9415
10	-35,4544	51,67029	58,30526

Tabel 4.19 Posisi awal agen B

Agen	posX	posY	posZ
1	-20,6537	-55,0655	60,63664
2	-52,5048	53,31954	-30,0828
3	20,38318	10,15899	-12,5333
4	-29,4974	-67,9015	-36,1206
5	-65,8704	14,83949	37,64535
6	-10,0484	-37,7654	16,216
7	33,823	-43,1647	-58,2389
8	18,59721	-60,8843	-57,5801
9	52,57228	-55,1096	-35,5911
10	12,38488	10,11788	23,49069

Tabel 4.20 Posisi awal agen C

Agen	posX	posY	posZ
1	-69,1818	56,22061	-17,0706
2	-32,163	30,56818	-44,5379
3	42,25014	58,94604	51,31588
4	50,01306	-13,8251	-6,54155
5	-16,781	-60,6691	-21,1205
6	30,9257	-4,11448	-13,5157
7	63,93087	41,04998	-29,5247
8	20,02953	68,42105	62,09506
9	41,87464	38,25802	18,31308
10	-45,7987	55,99192	54,63159

4.5 Penentuan posisi agen pada akhir iterasi maksimum dengan halangan (*obstacle*)

Agen akan berhenti bergerak ketika sudah menemukan target atau iterasi maksimum sudah tercapai. Agen akan menuju ke target yang paling cepat di jangkau oleh jangkauan visualnya.

Tabel 4.21 Posisi akhir agen A pada iterasi maksimum

posX	posY	posZ	Target ke
-163,732	371,3007	66,228	0
-165,348	297,5831	80,89337	0
-115,731	365,6771	100,4389	0
-149,975	381,7454	78,1128	0
-133,241	306,9443	74,89981	0
-160,371	367,9946	90,57498	0
-156,216	352,1853	89,85374	0
-161,347	349,9017	88,87428	0
-156,127	328,8999	108,1409	0
-165,249	358,1103	100,2956	0

Dari tabel 4.19 terlihat bahwa pada posisi akhir maksimum iterasi ke 1000 agen sama sekali tidak ada yang berhasil mencapai target, ini berarti agen membutuhkan itersi atau waktu yang lebih banyak lagi untuk bergerak mecari target. Hal ini dipengaruhi oleh jangkauan visualnya yang nilainya kecil (visual=50). Sehingga agen hanya dapat menjangkau jarak yang pendek. Dan kemungkinan untuk melihat target membutuhkan percobaan berkali kali.

Tabel 4.22 Posisi akhir agen B pada iterasi maksimum

posX	posY	posZ	Target ke
0,520868	180,3721	0,214079	5
0,044355	179,6706	-3,71767	5
-0,91067	175,5506	-1,25846	5
-3,35889	177,569	-4,01	5
-0,10396	180,8745	4,206762	5
-2,65378	182,6197	1,368636	5
-3,18108	178,8849	-0,15692	5
2,73327	177,0808	-0,33461	5
-0,3848	183,4504	-2,17483	5
-0,57165	177,1366	2,436109	5

Tabel 4.21 adalah tabel yang menunjukkan posisi akhir dari agen B yang semua agennya berhasil menuju target. Dengan visual yang lebih besar dibanding agen A (visual=100) terlihat bahwa agen B akan bisa menemukan target sebelum iterasi maksimum tercapai.

Tabel 4.23 Posisi akhir agen C pada iterasi maksimum

posX	posY	posZ	Target ke
176,975	2,456275	1,255842	8
181,0894	2,851615	-2,6925	8
180,9931	-3,74006	-2,13668	8
179,0468	-0,54957	3,475178	8
182,021	-2,64703	1,770021	8
183,4707	-0,10961	-1,02531	8
179,4329	-0,12103	-0,58705	8
177,9918	-3,55482	0,977668	8
181,6028	2,356987	1,686012	8
179,162	5,685027	-0,000052	8

Tabel 4.22 adalah tabel yang menunjukkan posisi akhir dari agen C yang semua agennya berhasil menuju target. Dengan visual yang lebih besar dibanding agen A (visual=100) terlihat bahwa agen B akan bisa menemukan target sebelum iterasi maksimum tercapai.

4.6 Penghitungan waktu agen mencapai iterasi maksimum dengan halangan (*obstacle*)

Dari hasil percobaan terlihat bahwa tiap agen akan mempunyai waktu yang berbeda beda ketika menemukan target pada iterasi maksimum baik agen secara individu maupun agen dalam kelompok mempunyai waktu yang berbeda-beda.

Tabel 4.24 Penghitungan waktu agen A mencapai target pada iterasi maskimum = 2000

Agen	Jarak Visual	Jarak Target	Waktu
1	3,157874	3,157874	68,22707
2	4,363136	4,363136	68,8105
3	3,977531	3,977531	68,69494
4	2,86949	2,86949	67,29073
5	4,96795	4,96795	69,27882
6	4,123969	4,123969	68,40165
7	4,836593	4,836593	68,86953
8	1,450247	1,450247	67,46534
9	3,52636	3,52636	69,27902
10	3,917101	3,917101	68,69528

Tabel 4.24 adalah penghiutngan waktu agen A dengan iterasi maksimum yang dinaikkan menjadi 2000. Terlihat pada iteraksi maksimum = 2000 semua agen mampu mencapai target. Dengan waktu minimum 67,27902 detik dan maksimum 69,27882 detik

Tabel 4.25 Penghitungan waktu agen B mencapai target pada iterasi maksimum = 1000

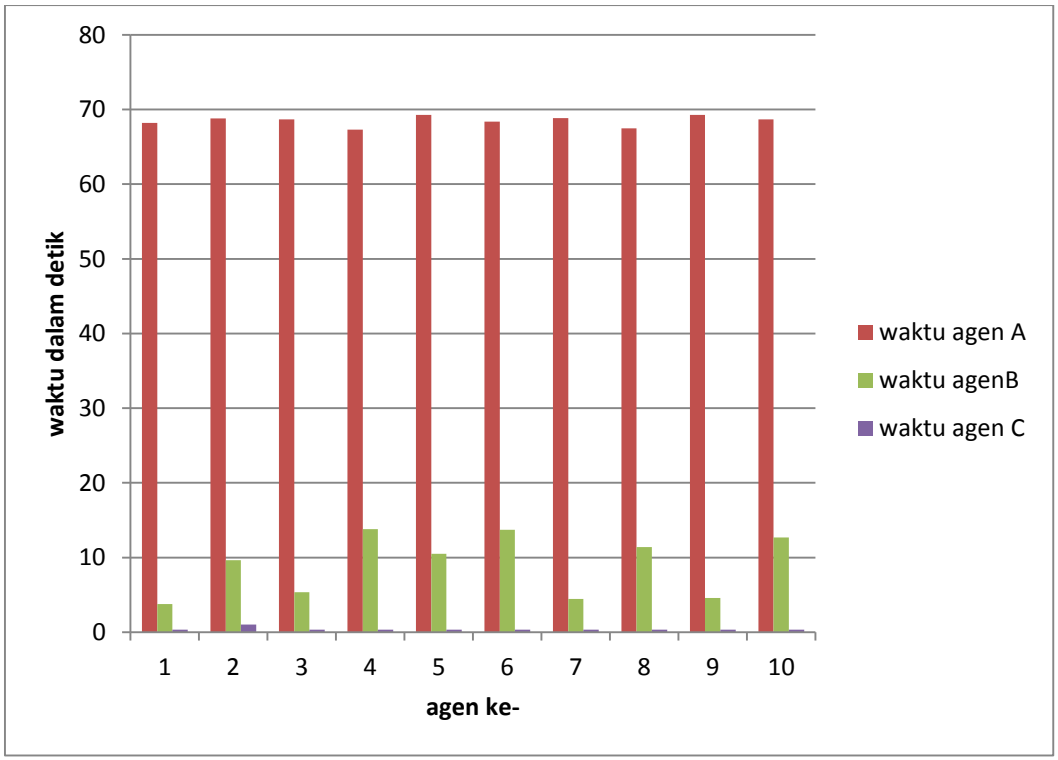
Agen	Jarak Visual	Jarak Target	Waktu
1	0,674959	0,674959	3,750057
2	3,732494	3,732494	9,626239
3	4,71274	4,71274	5,330859
4	5,768196	5,768196	13,78361
5	4,297956	4,297956	10,50767
6	3,972244	3,972244	13,72555
7	3,374513	3,374513	4,454654
8	4,013063	4,013063	11,37679
9	4,096739	4,096739	4,574252
10	3,80266	3,80266	12,66784

Dari tabel 4.24 terlihat semua agen mampu sampai ke titik target pada akhir iterasi ke 1000. Dengan waktu minimum 4,574252 detik dan maksimum 13,78361 detik

Tabel 4.26 Penghitungan waktu agen C mencapai target pada iterasi maskimum = 1000

Agen	Jarak Visual	Jarak Target	Waktu
1	4,094049	4,094049	0,321676
2	4,070392	4,070392	1,000104
3	4,420377	4,420377	0,321785
4	3,64519	3,64519	0,32184
5	3,771487	3,771487	0,321889
6	3,620678	3,620678	0,32194
7	0,825146	0,825146	0,321991
8	4,198272	4,198272	0,322039
9	3,311659	3,311659	0,32209
10	5,746459	5,746459	0,322143

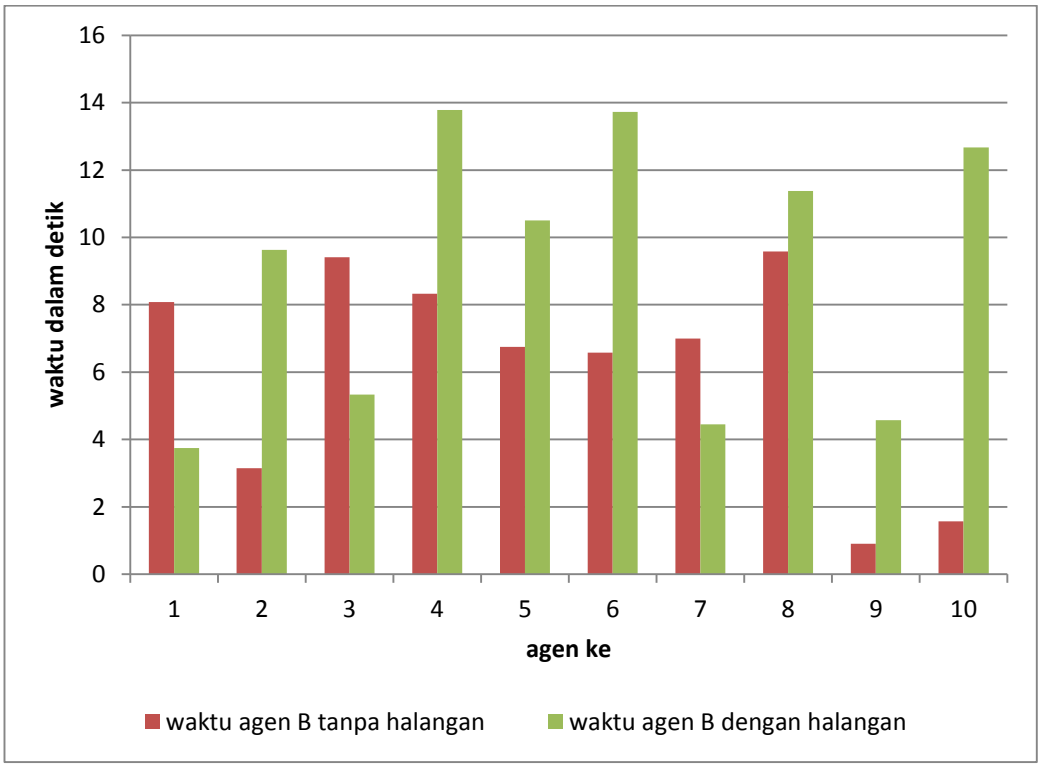
Dari tabel 4.25 terlihat semua agen mampu sampai ke titik akhir iterasi ke 1000. Dengan waktu minimum 0, 322039 detik dan maksimum 1,000104 detik



Gambar 4.9 Grafik perbandingan waktu agen A, B, C ada halangan

Gambar 4.9 adalah grafik perbandingan waktu ketiga agen dalam bergerak menemukan target di lingkungannya. Terlihat bahwa agen A mempunyai waktu

yang lebih lama dibanding agen B dan C. Agen C adalah yang tercepat mencapai target. Jangkauan visual sangat mempengaruhi kecepatan agen untuk menemukan target. Semakin besar jangkuan visual (nilai visual besar) maka kemungkinan agen untuk segera menemukan target akan lebih besar.



Gambar 4.10 Grafik perbandingan waktu agen tanpa halangan dan dengan ada halangan

Gambar 4.10 adalah perbandingan waktu yang diperlukan agen B pada parameter yang yaitu visual = 100, step = 2, crowd = 0,4, iterasi maksimum 1000 pada keadaan ada halangan dan tanpa halangan. Terlihat bahwa keadaan tanpa halangan memerlukan waktu yang lebih singkat untuk menemukan target dengan rata rata waktu rata-rata 6,1353629 detik dibanding dengan keadaan dengan adanya halangan yaitu ditempuh dengan waktu rata- rata 8,9797521 detik

4.7 Penghitungan waktu agen menemukan target dengan halangan (*obstacle*)

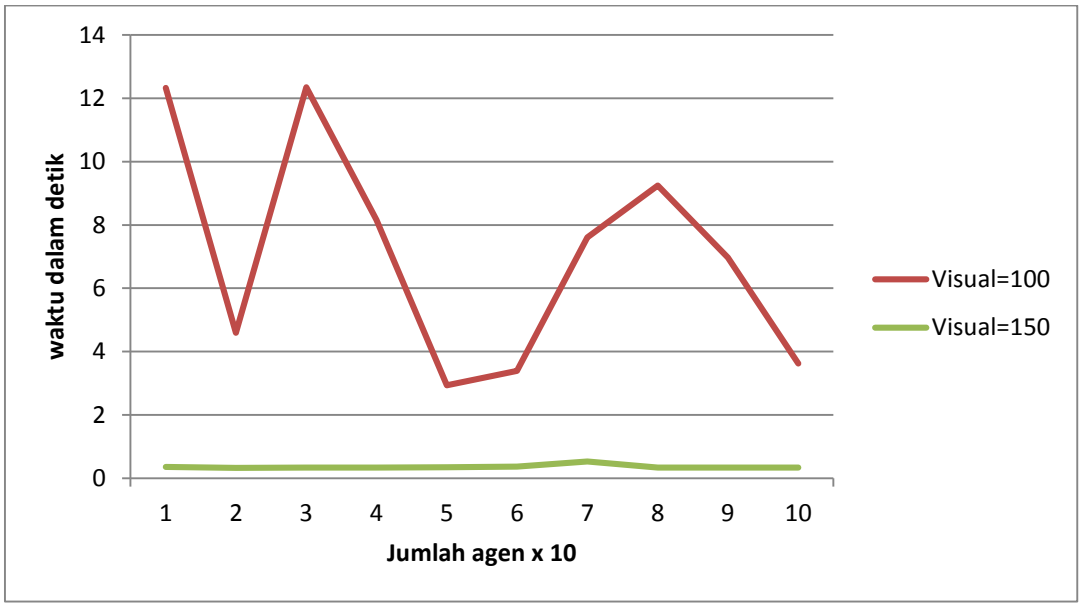
Pada percobaan ini akan dihitung waktu tempuh agen menemukan target, dari posisi awal. Agen akan melakukan pencarian jalur dan menghindari halangan yang berada di sepanjang. Agen yang mempunyai jangkauan visual yang terbatas akan mencari target dengan melakukan pencarian acak sampai menemukan target.

Agen di uji dengan parameter visual yang berbeda-beda sesuai dengan tabel.4.26. Untuk parameter yang lain ditentukan sebagai berikut Step = 1, Crowd = 0.1, Jumlah *obstacle* = 4.

Tabel 4.27 Pengujian agen dengan visual berbeda

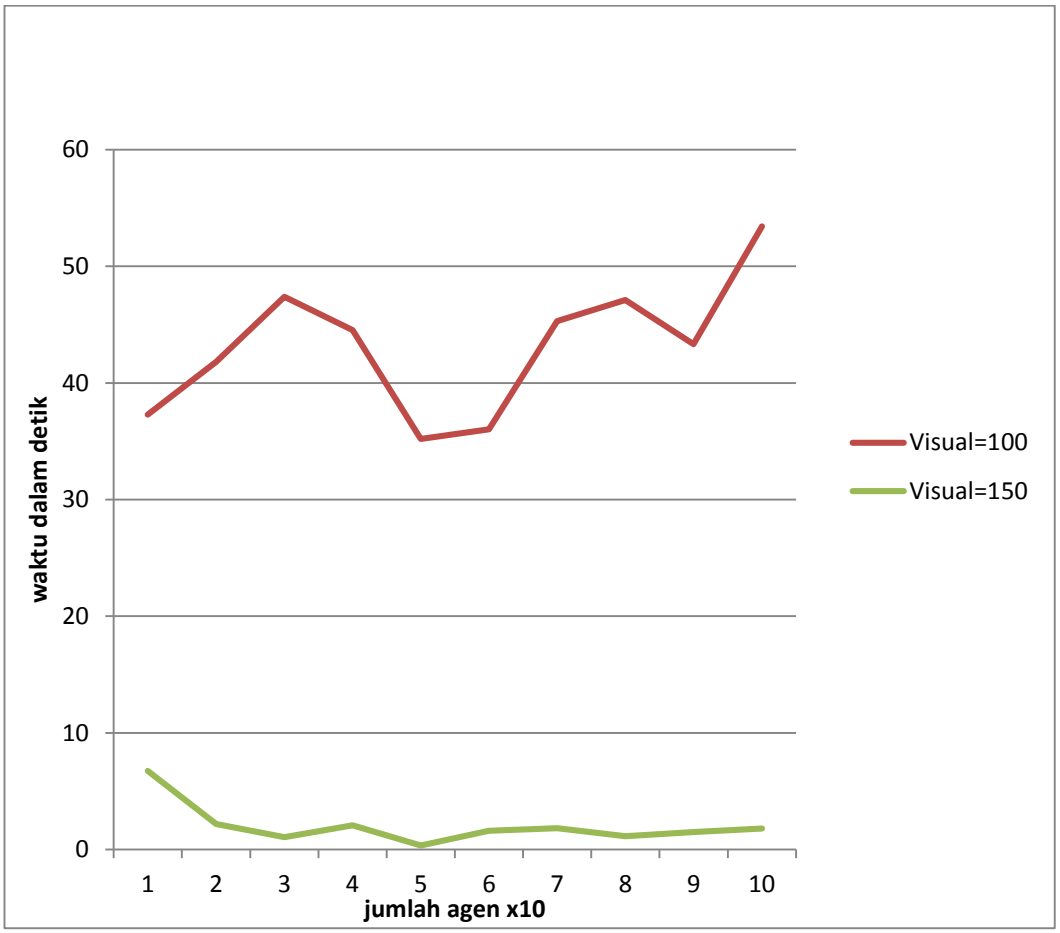
Fish Number	Waktu	
	Visual = 100	Visual =150
10	12,332315	0,362197
20	4,595692	0,326103
30	12,34593	0,333126
40	8,162023	0,333519
50	2,935326	0,344154
60	3,392442	0,370637
70	7,607009	0,525787
80	9,248949	0,337715
90	6,972585	0,335493
100	3,622679	0,333634

Tabel 4.26 adalah hasil pengujian agen dengan waktu tercepat untuk jumlah agen 10 sampai 100 agen. Dengan perubahan parameter visual=100 dan visual=150.



Gambar 4.11 Grafik waktu tercepat menemukan target

Gambar 4.11 Menunjukkan waktu yang dibutuhkan oleh agen dengan jumlah 10 sampai dengan 100 untuk menemukan target. Terlihat bahwa untuk jarak visual yang sama, penambahan jumlah agen tidak mempengaruhi waktu yang dibutuhkan agen menemukan target. Perubahan nilai visual dari 100 ke 150 mempengaruhi secara signifikan waktu yang ditempuh oleh agen menemukan targetnya.



Gambar 4.12 Grafik waktu rata-rata menemukan target

Gambar 4.12 adalah grafik waktu rata-rata yang dibutuhkan oleh agen dengan junmlah 10 sampai dengan 100 untuk menemukan target. Untuk nilai visual= 100 waktu rata rata minimal adalah 35,20750076 detik yang dibutuhkan agen berjumlah 50 dan waktu rata-rata maksimal adalah 53,41820419 yang dibutuhkan agen berjumlah 100. Untuk nilai visual= 150 waktu rata rata minimal adalah 0,3441785 detik yang dibutuhkan agen berjumlah 50 dan waktu rata-rata maksimal adalah 6,7335342 detik yang dibutuhkan agen berjumlah 10.

BAB 5
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dari hasil pengujian yang dilakukan telah berhasil membuktikan bahwa *Artificial Fish Swarm Algorithm* mampu membuat simulasi pergerakan sekelompok agen otonom yang dapat menemukan target dengan keterbatasan jangkauan visual dan berhasil melewati halangan. Agen akan bergerak terus secara acak sampai menemukan target yang masuk ke jangkauan visual. Perubahan perilaku agen dalam kelompok akan terjadi saat agen mulai bergerak awal. Pada saat agen belum menemukan target maka agen bergerak dengan perilaku *free move* pada fase perilaku ini agen akan terus merubah posisi target visualnya sampai menemukan target tujuan dan ketika agen menemukan target maka agen akan berperilaku *prey*, pada fase perilaku ini agen akan mengunci posisi target visualnya pada target tujuan dengan ditandai jarak visual yang sama dengan jarak target. Agen akan berkelompok atau berperilaku *swarm* jika menemukan agen lain yang berada dalam jangkauan visualnya. dan jika ada agen lain yang berada pada jangkauan agen yang menemukan target maka agen akan berperilaku *follow*. Dari hasil pengamatan di salah satu agen yaitu agen A didapatkan hasil agen berkerumun pada tiga posisi target yang berjumlah delapan target yaitu pada target ke 4, 5, 7. Agen akan cenderung mengikuti agen lain yang terjangkau oleh jangauan visualnya, sehingga membentuk kerumunan. Agen ke 2,4,5,6,8 berkerumun di posisi target ke 4, Agen ke 7, 9 berkerumun di posisi target 5. Agen ke 1, 3, 10 berkerumun di posisi target 7. Disini terlihat bahwa agen berperilaku *swarm* dan *follow* saat bergerak mencari dan menuju target.

Dari hasil pengamatan pengujian yang dilakukan dengan jumlah agen 10 sampai dengan 100 , perubahan parameter visual sangat mempengaruhi waktu agen untuk menemukan target. Untuk nilai visual= 100 waktu rata-rata maksimal adalah 53,4 detik. Untuk nilai visual= 150 waktu rata-rata maksimal adalah 6,7 detik. Semakin besar visual maka agen akan lebih cepat menemukan target.

Penambahan halangan (*obstacle*) akan mempengaruhi waktu tempuh agen menemukan target karena agen harus menghitung posisi agar tidak berbenturan dengan *obstacle* sehingga dibutuhkan iterasi yang lebih banyak dan waktu komputasi yang lebih banyak. Keadaan tanpa halangan memerlukan waktu yang lebih singkat untuk menemukan target dengan waktu rata-rata 6,1 detik dibanding dengan keadaan dengan adanya halangan yaitu ditempuh dengan waktu rata- rata 8,9 detik

5.2 Saran

Penelitian ini masih pada tahap awal implementasi Artificial Fish *Swarm* Algorith sebagai algoritma untuk menggerakkan karakter virtual. Agar lebih kompleks maka penelitian ini dapat dilanjutkan dengan menambahkan *obstacle* dan target yang dinamis, karena pada game agar lebih menarik, *obstacle* yang bisa berperan sebagai musuh sebaiknya harus bisa bergerak. Demikian juga dengan target. Target bisa berperan sebagai makanan yang akan dimakan predator jika masih dalam keadaan hidup maka makanan tersebut seharusnya juga bergerak.

DAFTAR PUSTAKA

[1] Cai, Yun, "Artificial Fish School Algorithm Applied in a Combinatorial Optimization Problem", I,J, Intelligent Systems and Applications, 2010, 1, 37-43.

[2] Chen, Yi, "*Swarm* Fish The Artificial Fish *Swarm* Algorithm (AFSA)", *Swarm*sLAB, 2011.

[3] Yazdani, B dan Saman, B, "A new Algorithm Based on Improved Artificial Fish *Swarm* Algorithm for Data Clustering", International Journal of Artifciial Intelligence Vol. 11 No. A13, 2013.

[4] Millington, Ian, "Artificial Intelligent for Games", Morgan Kauffman Publisher, 2006.

[5] Reynolds, Craig W, "Flocks, Herds, and Schools: A Distributed Behavioral Model", ACM SIGGRAPH '87 Conference Proceedingscs, Computer Graphics, 21(4), July 1987, pp, 25-34.

[6] Ingham, James, "What is an Agent?", Centre for Software Maintenance University of Durham,1997.

[7] Ahmed, Hazem dan Glasgow, Janice, "*Swarm* Intelligence : Concepts, Models and Applications", School of Computing Queen's University Kingston, Ontario, Canada K7L3N6, 2012-585.

[8] Huang, Zhehuang dan Chen, Yidong, "An Improved Artificial Fish *Swarm* Algorithm based on Hybrid Behavior Selection ", International Journal of Control and Automation Vol.6, No.5, 2013, pp.103-116.

[9] Franklin, Stan, "Autonomous Agents as Embodied AI", Cybernetics and System, 28 : 6 (1997) 499-520.

[10] Neshat, M, Sepidnam, G, Sargolzaei, M., Najaran Toosi, A, "Artificial Fish *swarm* algorithm: a survey of the state of the-art, hybridization, combinatorial and indicative applications", Springer Science+Business Media B.V., 2012.

[11] Doherty, Darren dan O'Riordan, Colm, "The Design and Implementation of AI in Modern Computer Games", Department of Information Technology National University of Ireland Galway

[12] http://clashofclans.wikia.com/wiki/

BIOGRAFI

Deny Safril, lahir di Tulungagung, 1 April 1976. Menempuh pendidikan dasar dan menengah di Tulungagung, Jawa Timur. Tahun 1988 menyelesaikan Sekolah Dasar di SD Negeri Kampungdalem 2 Tulungagung, tahun 1991 menyelesaikan Sekolah Menengah Pertama di SMP Negeri 1 Tulungagung, dan tahun 1994 menyelesaikan Sekolah Mengah Atas di SMA Negeri 1 Kedungwaru, Tulungagung. Tahun 2001 lulus jenjang S1 dari Jurusan Teknik Elektro Universitas Brawijaya Malang. Saat ini bekerja sebagai staf pengajar di SMKN 1 Blitar dan Akademi Komunitas Negeri Putra Sang Fajar Blitar, mengajar di program studi Teknologi Informasi dan Multimedia. Hobi bermain musik dan menggeluti dunia musik diluar kesibukan sehari hari sebagai pendidik dan pengajar.

denysafril@gmail.com