PERGERAKAN NPC MENGGUNAKAN ALGORITMA *BOIDS* DAN *ARTIFICIAL BEE COLONY* PADA SIMULASI MENGELILINGI KA'BAH (THAWAF)

SKRIPSI

Oleh :
HERU SANTOSO DJAMALUDIN
NIM 12650048

JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERIMAULANA MALIK IBRAHIM
MALANG
2016

HALAMAN PENGAJUAN

PERGERAKAN NPC MENGGUNAKAN ALGORITMA *BOIDS* DAN *ARTIFICIAL BEE COLONY* PADA SIMULASI MENGELILINGI KA'BAH (THAWAF)

SKRIPSI

Diajukan Kepada:
Fakultas Sains dan Teknologi
Universitas Islam Negeri
Maulana Malik Ibrahim Malang
Untuk Memenuhi Salah Satu Persyaratan Dalam
Memperoleh Gelar Sarjana Komputer (S.Kom)

Oleh :
HERU SANTOSO DJAMALUDIN
NIM 12650048

JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2016

LEMBAR PERSETUJUAN

PERGERAKAN NPC MENGGUNAKAN ALGORITMA *BOIDS* DAN *ARTIFICIAL BEE COLONY* PADA SIMULASI MENGELILINGI KA'BAH (THAWAF)

SKRIPSI

Oleh:
Heru Santoso Djamaludin
NIM. 12650048

Telah Diperiksa dan Disetujui :
Tanggal : 15 Desember 2016

Pembimbing I	Pembimbing II
Fressy Nugroho, M.T	Hani Nurhayati, M.T
NIP. 19710722 201101 1 001	NIP. 19780625 200801 2 006

Mengetahui,
Ketua Jurusan Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang

Dr. Cahyo Crysdian
NIP. 19740424 200901 1 008

HALAMAN PENGESAHAN

PERGERAKAN NPC MENGGUNAKAN ALGORITMA BOIDS DAN ARTIFICIAL BEE COLONY PADA SIMULASI MENGELILINGI KA'BAH (THAWAF)

SKRIPSI

Oleh :

Heru Santoso Djamaludin

NIM : 12650048

Telah Dipertahankan di Depan Dewan Penguji Skripsi dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan Untuk Memperoleh Gelar Sarjana Komputer (S.Kom)

Tanggal Desember 2016

Susunan Dewan Penguji			Tanda Tangan
1. Penguji Utama	:	Fachrul Kurniawan, M.MT NIP. 19771020 200901 1 001	( )
2. Ketua	:	Dr. Muhammad Faisal, M.T NIP. 19740510 200501 1 007	( )
3. Sekretaris	:	Fressy Nugroho, M.T NIP. 19710722 201101 1 001	( )
4. Anggota	:	Hani Nurhayati, M.T NIP. 19780625 200801 2 006	( )

Mengetahui dan Mengesahkan
Ketua Jurusan Teknik Informatika

Dr. Cahyo Crysdian
NIP. 19740424 200901 1 008

HALAMAN PERNYATAAN
ORISINALITAS PENELITIAN

Nama : Heru Santoso Djamaludin
NIM : 12650048
Jurusan : Teknik Informatika
Fakultas : Sains dan Teknologi
Judul Skripsi : **PERGERAKAN NPC MENGGUNAKAN ALGORITMA *BOIDS* DAN *ARTIFICIAL BEE COLONY* PADA SIMULASI MENGELILINGI KA'BAH (THAWAF)**

Menyatakan dengan sebenarnya bahwa skripsi yang saya tulis ini benar-benar merupakan hasil karya saya sendiri, bukan merupakan pengambil alihan data, tulisan, atau pikiran orang lain yang saya akui sebagai hasil tulisan atau pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka. Apabila dikemudian hari terbukti atau dapat dibuktikan skripsi ini hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 10 Desember 2016
Yang membuat pernyataan,

Heru Santoso Djamaludin
NIM. 12650048

HALAMAN MOTTO

"Kehidupan pemuda – demi Allah – adalah dengan mencari ilmu dan bertaqwa, bila keduanya tak mewujud, maka tak ada yg menandai keberadaannya." (Imam Syafi'i)

HALAMAN PERSEMBAHAN

Bismillahirrohmanirrohim, kupersembahkan sebuah karya sederhanaku ini untuk orang-orang yang paling kusayangi, kubanggakan, dan selalu memberikan energi semangat untukku..

Seluruh keluarga besarku
khususnya Bapak dan Mama yang tercinta
Kasimanto dan Djaenab Djamaludin
yang selalu ikhlas mendoakan putra-putrinya
yang selalu mengarahkan kami dalam kebaikan
yang dengan sabar membimbing kami.
Semoga Allah SWT melindungi dan menjaga mereka dalam naungannya..
Aamiin

KATA PENGANTAR

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

Segala puji bagi Allah SWT yang Maha Pengasih lagi Maha Penyayang atas Rahmat dan Hidayah-Nya sehingga penulis dapat menyelesaikan penyusunan skripsi ini. Sholawat serta Salam tetap tercurahkan kepada junjungan kita, kekasih Allah, Nabi Muhammad SAW, sang pemberi syafaat kelak di hari akhir, beserta seluruh keluarga, sahabat, dan para pengikutnya.

Penelitian skripsi yang berjudul "**Pergerakan NPC Menggunakan Algoritma *Boids* Dan *Artificial Bee Colony* Pada Simulasi Mengelilingi Ka'bah (Thawaf)**" ini ditulis untuk memnuhi salah satu syarat guna memperoleh gelar Sarja Strata Satu (S1) Fakultas Sains dan Teknologi Universitas Maulana Malik Ibrahim Malang. Karya penelitian skripsi ini tidak akan pernah ada tanpa bantuan baik moral maupun spiritual dari berbagai pihak yang telah terlibat. Untuk itu dengan segala kerendahan hati, penulis mengucapkan rasa terimakasih yang sebesar-besarnya kepada:

1. Prof. Dr. H. Mudjia Rahardjo, M.Si, selaku rektor Universitas Islam Negeri Maulana Malik Ibrahim Malang.
2. Bapak Fressy Nugroho, M.T, selaku Dosen Pembimbing I yang telah bersedia meluangkan waktu, tenaga dan pikiran untuk memberikan bimbingan, berbagai pengalaman, arahan, nasihat, motivasi dan pengarahan dalam pembangunan program hingga penyusunan skripsi ini.
3. Bu Hani Nurhayati, M.T, selaku dosen pembimbing 2 yang selalu memberi masukan, serta pengarahan dalam penyusunan laporan skripsi ini.
4. Bapak H. Syahiduz Zaman, M.Kom, selaku dosen wali yang juga selalu memberi pengarahan terkait akademik selama masa study.
5. Dr. Cahyo Crysdian selaku ketua jurusan Teknik Informatika yang mendukung dan mengarahkan skripsi ini.
6. Segenap civitas akademika Fakultas Saintek, Universitas Islam Negeri Maulana Malik Ibrahim Malang terutama seluruh dosen, terimakasih atas segala ilmu dan bimbingannya.

7. Mama, Bapak dan seluruh keluarga yang selalu memberikan doa, kasih sayang, semangat, dukungan moril, serta motivasi sampai saat ini, terimakasih banyak.
8. Ilya Ismani M, Nur Halimatus S yang membantu internal maupun eksternal dalam bentuk semangat kepada penulis untuk menyelesaikan skripsi ini.
9. Teman-teman angkatan 2012, yang berjuang bersama-sama untuk meraih mimpi, terimakasih atas kenang-kenangan indah yang dirajut bersama.
10. Semua pihak yang tidak dapat penulis sebutkan satu-persatu atas bantuan, masukan, dukungan serta motivasi kepada penulis.

Harapan penulis semoga semua amal kebaikan dan jasa-jasa dari semua pihak yang telah membantu hingga skripsi ini selesai diterima oleh Allah SWT, serta mendapatkan balasan yang lebih baik dan berlipat ganda.

Penulis juga menyadari bahwa skripsi ini masih jauh dari kesempurnaan yang disebabkan keterbatasan Harapan penulis, semoga karya ini bermanfaat dan menambah ilmu pengetahuan bagi kita semua, Aamiin.

Malang, 10 Desember 2016
Penulis

Hera Santoso Djamaludin

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

ABSTRAK

Heru Santoso Djamaludin, **Pergerakan NPC Menggunakan Algoritma *Boids* Dan *Artificial Bee Colony* Pada Simulasi Mengelilingi Ka'bah (Thawaf)**
Pembimbing I: Fressy Nugroho, M.T
Pembimbing II: Hani Nurhayati, M.T
Kata Kunci : Simulasi, Thawaf, *Boids*, *Obstacle Avoidance*, *Artificial Bee Colony*

Simulasi perilaku NPC merupakan tema penelitian dalam bidang computer grafik yang memiliki banyak manfaat, antara lain untuk membuat simulasi mengelilingi Ka'bah (Thawaf) dan sebagai unsur pendukung realisme dalam game. Penelitian ini dilakukan dengan menggunakan metode yang dapat mendukung fitur AI dengan cara menambahkan perilaku kerumunan pada pergerakan NPC tersebut. Untuk mewujudkan hal tersebut pada simulasi animasi ini digunakan metode *Boids* dan *Artificial Bee Colony*. Algoritma *Boids* digunakan untuk mengorganisir pergerakan NPC dengan memperhitungkan jarak, kecepatan, dan pergerakan yang baik. Algoritma *Artificial Bee Colony* digunakan untuk persebaran NPC dalam mencari target. Dari pengujian telah menunjukkan hasil sebesar 100% terhadap pengujian 150 jemaah dalam keadaan bergerombol untuk mengelilingi Ka'bah (Thawaf). Sedangkan hasil untuk rata-rata waktu yang diperlukan dalam melakukan satu putaran simulasi akan meningkat dengan semakin meningkatnya jumlah jemaah. Dengan jumlah awal 15 jemaah kemudian ditingkatkan menjadi 150 jemaah, maka waktu yang diperlukan mengalami peningkatan sebesar 32.09%.

ABSTRACT

Heru Santoso Djamaluddin, **NPCs Movement Using Boids Algorithms And Artificial Bee Colony On Thawaf Simulation.**
Supervisor I: Fressy Nugroho, M.T.
Supervisor II : Hani Nurhayati, M.T.
Keywords : The simulation, Thawaf, Boids, Obstacle Avoidance, Artificial Bee Colony.

NPCs behavior simulation is a research theme on the field of computer graphics that has many benefits, among others, to create a Thawaf simulation and as a supporting element of realism in the game. This research was conducted using a method that can support the NPCs AI feature by adding the crowd behavior of the NPCs movement. To achieve this, the animated simulation method was using Boids and Artificial Bee Colony. Boids algorithm was used to organize the movement of NPCs well by taking into account distance, speed, and movement. Artificial Bee Colony Algorithm was used for the distribution of the NPCs to find the target. The result in the testing of 150 pilgrims in a state of clustered to do Thawaf is 100%. While the result of the average time needed to do a lap of simulation will increase with the increasing number of pilgrims. With the initial amount of 15 pilgrims, then increasing to 150 pilgrims, the time required increased by 32.09%.

الملخص

هيرو سانطوسو جمالدين، حركة غير فطرة اللاعب باستخدام حساب البوئيد و اصطناعي الثول على محاكاة طوّاف الكعبة

مشرفة الأولى : فرشي نوغراها الماجستر

مشرفة الثاني : هاني نورحياتي الماجستر

الكلمات الرائسية : محاكاة، طوّاف، بوئيد، إلغاء العقبة، اصطناعي الثول

محاكاة من صفة غير فطرة اللاعب هي موضوع البحث في مجال الحاسوب التخطيطي عنده كثير من الفوائد، منها لتصنع أن تطوف الكعبة وكعناصر المعاون الحقيقي في اللعبة . يعمّل هذا البحث باستخدام الطريقة التي تستطيع أن تساعد استيمارة المعلومات على غير فطرة اللاعب بطريقة الزيادة خلوقيا عقديا على حركتها . لتحقيق هذا الشيئ في محاكاة صناعة صور متحركة باستخدام الطريقة بوئيد واصطناعي الثول . يستخدم حساب البوئيد لينظّم حركتها بحساب الفضائي وسرعة وحسن الحركة. ويستخدم حساب الاصطناعي الثول لانتشارها في بحوث عن الهدف. وجد نتائج الاختبار ١٠٠% على ١٥٠ جماعة في محال مجموعة ليعملون الطواف. حيث أن نتائج معدّلة الوقت الذي يحتاج ليعامل طوّافا من محاكاة سيرفع بارتفاع الجماعة . بعدد الأول يعني ١٥ يكون ١٥٠ جماعة، فيحتاج الوقت يكون أن يرتفع ٣٢،٠٩%

BAB I

PENDAHULUAN

1.1 Latar Belakang

Thawaf merupakan salah satu dari rukun haji/umrah. Thawaf artinya mengitari/mengelilingi. Tawaf merupakan salah satu ibadah yang hanya dilakukan di Baitullah, yaitu mengelilingi Ka'bah sebanyak tujuh putaran yang dimulai dan diakhiri di Hajar Aswad.

Thawaf membawa pesan maknawi berputar pada poros bumi yang paling awal dan paling dasar. Perputaran tujuh keliling bisa diartikan sama dengan jumlah hari yang beredar mengelilingi kita dalam setiap minggu. Lingkaran pelataran Ka'bah merupakan arena pertemuan dan bertamu dengan Allah pertemuan dan bertamu dengan Allah Swt. yang dikemukakan dengan do'a dan dzikir dan selalu dikumandangkan selama mengelilingi Ka'bah.

Suci dari hadats dan najis, menutup aurat, berada di dalam Masjidil Haram, memulai dari Hajar Aswad, Ka'bah berada di sebelah kiri, di luar Ka'bah (tidak di dalam Hijir Ismail), mengelilingi Ka'bah sebanyak tujuh kali putaran, niat tersendiri, kalau thawafnya itu berdiri sendiri, tidak terkait dengan haji dan umrah merupakan syarat syahnya thawaf.

Dalam melakukan thawaf juga harus mengerti tata caranya yang benar. Masih banyak jemaah pada saat thawaf melakukan kesalahan-kesalahan yang tidak diperbolehkan saat thawaf. Oleh sebab itu, peneliti membuat simulasi thawaf untuk mengajarkan tatacara thawaf yang benar. Karena sebagai umat islam menyampaikan ilmu adalah sesuatu yang harus dilakukan. Seperti firman Allah pada Surat Ali-'Imraan ayat 104:

وَلْتَكُن مِّنكُمْ أُمَّةٌ يَدْعُونَ إِلَى الْخَيْرِ وَيَأْمُرُونَ بِالْمَعْرُوفِ وَيَنْهَوْنَ عَنِ الْمُنكَرِ وَأُوْلَئِكَ هُمُ الْمُفْلِحُونَ ﴿١٠٤﴾

Yang artinya: "*Dan hendaklah ada di antara kamu segolongan umat yang menyeru kepada kebajikan, menyuruh kepada yang ma`ruf dan mencegah dari yang munkar; merekalah orang-orang yang beruntung.*" (Qs. Ali-'Imraan : 104)

Dengan mengacu pada penelitian sebelumnya. Dalam penelitian (Reynold, 1987), pergerakan pasukan dengan melakukan pemisahan *(separation)*, menjaga keselarasan kecepatan *(alignment)* dan menyatu kembali *(cohesion)* adalah program kehidupan buatan, yang dikembangkan oleh Craig Reynolds pada tahun 1986, yang mensimulasikan perilaku berkelompok burung dan tiga aturan tersebut disebut dengan algoritma *Boids*.

Pada penelitian (Dewi, 2011), menggunakan metode *flocking* dan *obstacle avoidance* dengan *collision detection* untuk mensimulasikan pergerakan orang di mall keluar menuju pintu utama dalam menghindari hambatan statis berupa benda diam di dalam *mall* dan hambatan dinamis berupa benda bidang bergerak.

Menurut Jatiningsih untuk mengatur perpindahan kerumunan menuju suatu tujuan yang sama dapat menggunakan ABC *(Artificial Bee Colony)*. Pada penelitian yang dilakukan (Jatiningsih, 2014), menggunakan metode *Bee Colony* untuk menerapkan strategi penyerangan bergerombol dengan siklus terpendek, pada satu musuh yang diam. Metode ini memiliki prinsip seperti koloni lebah yang memiliki kerjasama team yang sangat erat dalam mencari sumber makanan.

Dalam penelitian ini, peneliti menerapkan algoritma *Boids* dan ABC *(Artificial Bee Colony)* sebagai metode dalam perilaku NPC untuk menghindari hambatan statis berupa benda diam, menjaga tidak saling tabrakan antar lainnya,

dan pencarian target yang akan disimulasikan dalam pergerakan jemaah yang sedang mengitari/mengelilingi Ka'bah (thawaf) di Baitullah. Maka, dengan simulasi ini diharapkan bisa memberikan kemudahan untuk memahami tatacara thawaf.

1.2 Identifikasi Masalah

Bagaimana membuat pergerakan NPC agar dapat menghindari hambatan statis berupa benda diam, tidak saling bertabrakan antar lainnya, dan pencarian target pada simulasi mengelilingi Ka'bah (thawaf), sehingga menghasilkan perilaku NPC yang fleksibel menggunakan algoritma *Boids* dan ABC *(Artificial Bee Colony).*

1.3 Batasan Masalah

Batasan masalah pada penelitian ini sebagai berikut :

a. Berbasis *desktop* dan diimplementasikan pada platform *Windows*.
b. Simulasi hanya dibatasi pada kasus pergerakan NPC saat mengitari/mengelilingi Ka'bah.
c. Simulasi ini fokus pada pembuatan pola bergerak realitas bagi NPC saat menuju target dengan *Boids* dan ABC.
d. Hambatan pada simulasi penelitian ini berupa hambatan statis yang ada disekitar area Ka'bah.
e. Target yang dituju pada simulasi ini yaitu *waypoint–waypoint* yang ada disekeliling Ka'bah.

1.4 Tujuan Penelitian

Tujuan dilakukannya penelitian ini adalah menghasilkan pergerakan NPC agar dapat menghindari hambatan statis berupa benda diam, tidak saling bertabrakan antar lainnya, dan pencarian target pada simulasi mengelilingi Ka'bah (thawaf), sehingga menghasilkan perilaku NPC yang fleksibel menggunakan algoritma *Boids* dan ABC (*Artificial Bee Colony*).

1.5 Manfaat Penelitian

Manfaat yang diharapkan dari penelitian ini adalah dapat memberikan kontribusi pada pengembangan deteksi tabrakan antar objek statis atau perilaku sekelompok agen dalam bidang animasi komputer, serta dapat dimanfaatkan sebagai salah satu metode simulasi kerumunan (*crowd simulation*).

BAB II

STUDI PUSTAKA

2.1 Landasan Teori

2.1.1 NPC (*Non Player Character*)

Non Player Character (NPC) disebut juga *autonomous character* yaitu karakter dalam *game* dimana memiliki kemampuan untuk melakukan gerakan secara otomatis atau tidak dikontrol oleh pemain. NPC merupakan komponen yang sangat penting dalam *game*, keberadannya dapat menciptakan agen cerdas yang nyata, jika tidak ada NPC yang berperilaku cerdas, maka *game* tidak akan menarik dan cenderung membosankan. Penelitian tentang AI (*Artificial Intelligence*) pada NPC (*Non Player Character*) dalam *game*, hingga saat ini masih terus dikembangkan. AI tersebut di kembangkan untuk merancang perilaku NPC (JimHyuk ,2005).

Yunifa Miftahul Arif (2010) dalam penelitiannya yang berjudul Strategi Menyerang pada Game FPS Menggunakan *Hierarchy Finite State Machine* dan Logika Fuzzy menjelaskan bahwa NPC adalah jenis *otonomous agent* yang ditujukan untuk penggunaan komputer animasi dan media interaktif seperti *games* dan *virtual reality*. Agen ini mewakili tokoh dalam cerita atau permainan dan memiliki kemampuan untuk improvisasi tindakan mereka. Ini adalah kebalikan dari seorang tokoh dalam sebuah film animasi, yang tindakannya ditulis di muka, dan untuk "avatar" dalam sebuah permainan atau *virtual reality*, tindakan yang diarahkan secara *real time* oleh pemain. Dalam permainan, karakter otonom biasanya disebut *Non Player Character* (NPC).

Secara garis besar maka NPC dapat diartikan sebagai karakter yang sepenuhnya dikendalikan komputer dan tidak dapat dimainkan oleh pemain. Pengendalian NPC umumnya menggunakan bidang ilmu kecerdasan buatan.

Kecerdasan buatan dimasukkan dengan tujuan agar NPC dapat melakukan pekerjaan seperti yang dapat dilakukan manusia. Beberapa macam bidang yang menggunakan kecerdasan buatan antara lain sistem pakar, permainan komputer, logika fuzzy, jaringan syaraf tiruan, robotika dsb.

2.1.2 Algoritma *Boids*

Boids adalah program kehidupan buatan, yang dikembangkan oleh Craig Reynolds pada tahun 1986. *Boids* dari awal kata "*birds*" adalah sebuah algoritma yang merepresentasikan gerak dari sebuah kawanan. Perilaku yang dihasilkan sangat mirip dengan kumpulan ikan atau kawanan burung. Semua *boids* bisa bergerak dalam satu arah pada satu saat, dan kemudian formasi kawanan tersebut dapat berbelok dan yang lainnya akan mengikuti seperti gelombang yang mengubah penyebaran *boids*. Gerak *boids* yang dihasilkan cukup mengesankan, bahkan lebih mengesankan adalah kenyataan bahwa perilaku ini adalah hasil dari tiga aturan sederhana elegan yaitu *cohesion, alignment, separation*. Dimana sekelompok *boids* dapat saling menghindar dari *collision*, kemudian dapat menyesuaikan arah gerakan dari rata-rata gerak jumlah *boids* seluruhnya, dan yang terakhir adalah mampu bergerak ke arah pusat rata-rata dari kawanan tersebut (Alun dkk, 2011).

2.1.2.1 *Separation*

Pembatasan jika sebuah agen terlalu dekat dengan agen lainnya, dengan cara melakukan penyesuaian arah dan kecepatan untuk menghindari benturan (*collision*). Agen akan menjaga jarak agar tidak bertabrakan dengan *flocksmate* atau tetangganya. Aturan ini mengarahkan (*steering*) agar *boids* bergerak menghindari kondisi yang padat (*crowded*) oleh kawanan tetangganya. Hal ini memungkinkan *boids* dapat menghindari terjadinya tabrakan dan menjaga agar *boids* tetap terpisah pada jarak pisah tertentu yang realistis atau tidak terlalu berdekatan.

Gambar 2. 1 *Separation* (Reynold, 2010)

Separation dapat diformulasikan sebagai berikut;

$$\sum_{n \in N} Normalize\,(agent_{pos} - n_{pos})$$

Dimana :

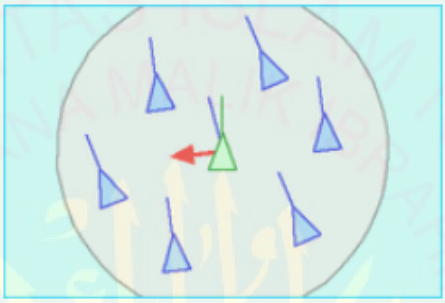
agentPos = Posisi agen

nPos = Posisi jumlah agen

2.1.2.2 *Alignment*

Mengambil rata-rata dari semua percepatan agen yang lain dan melakukan penyesuaian percepatan untuk pindah kearah kelompok. Mengarahkan agen

menuju posisi rata-rata tetangga. Aturan ini mengarahkan (*steering*) agar *boids* bergerak ke arah yang merupakan tujuan dari sebagian besar kawanan di tetangga lokalnya. *Boids* berusaha untuk menyesuaikan kecepatannya (arah, kecepatan bergerak) dengan kecepatan tetangga-tetangganya. Hal ini memungkinkan *boids* dapat mengimbangi pemisahan dan membuat *boids* bergerak pada satu arah tujuan yang sama.



Gambar 2. 2 *Alignment* (Reynold, 2010)

Aligment dapat diformulasikan sebagai berikut;

$$\sum_{n \in N} Normalize\,(n_{vel})$$

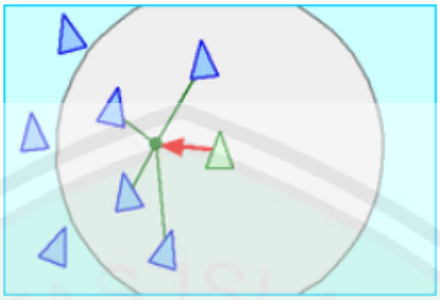
Dimana :

nVel = Kecepatan seluruh jumlah agen

2.1.2.3 *Cohesion*

Menghitung pusat keseluruhan kelompok dan mengarahkan agen ke arah titik pusatnya. Agen akan mencoba untuk tetap dekat dengan kelompoknya. Aturan ini mengarahkan (*steering*) agar *boids* (agen) bergerak maju ke arah yang merupakan tujuan dan sebagian besar kawanan di tetangga lokalnya. Hal ini memungkinkan *boids* dapat tetap bersama-sama dengan kawanan lokalnya dan

melakukan kegiatan pengumpulan beberapa kawanan maupun pemisahan kawanan ke dalam 2 kelompok.



Gambar 2. 3 *Cohesion* (Reynold, 2010)

Cohesion dapat diformulasikan sebagai berikut;

$$\frac{\sum_{n \in N} n_{pos}}{|N|}$$

Dimana :

nPos = Posisi jumlah agen

agentPos = Posisi agen

nVel = Kecepatan seluruh jumlah agen

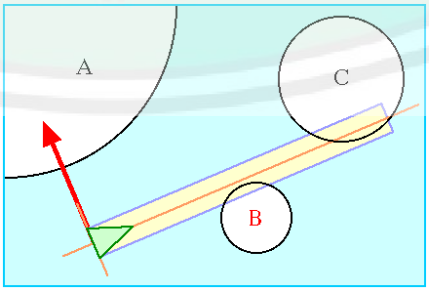
Reynolds (1999) telah menambah model *boids* termasuk aturan individu berbasis lebih dari kemudi perilaku, untuk memiliki individu yang lebih canggih yang mampu untuk menyelesaikan tugas tertentu atau beradaptasi dengan lingkungan yang kompleks. Beberapa perilaku ini adalah (Reynolds, 1999):

2.1.2.4 *Obstacle Avoidance*

Perilaku menghindari harnbatan *(obstacle avoidance)* memberikan kemampuan karakter untuk manuver di *environment* dengan menghindari harnbatan sekitarnya.

Ada perbedaan penting antara menghindari harnbatan (*obstacle avoidance*) dan perilaku melarikan diri (*flee*). *Flee* akan selalu mengarahkan karakter untuk menjauh dari lokasi tertentu, sedangkan *obstacle avoidance* tindakan akan diambil hanya jika suatu hambatan yang terdekat terletak tepat di depan karakter. Sebagai contoh, jika sebuah mobil mengemudi sejajar dengan dinding, *obstacle avoidance* akan mengambil tindakan korektif kemudi, tapi *flee* akan berusaha untuk berpaling dari dinding, akhinya mengemudi tegak lurus dengan dinding. Implementasi dari perilaku *obstacle avoidance* berhubungan dengan penghindaran rintangan dimana tidak harus terjadi tabrakan. Bayangkan sebuah pesawat berusaha untuk menghindari gunung (Mudhana dkk, 2014).

Tujuan dari perilaku *obstacle avoidance* adalah untuk menjaga sebuah silinder imajiner (sebagai hambatan) yang berada di depan karakter bola (lihat Gambar 2.4),. Silinder A dan C terletak disepanjang sumbu didepan karakter B (bola). Perilaku menghindari hambatan mempertimbangkan setiap kendala yang pada gilirannya mungkin menggunakan skema portioning spasial untuk menyisihkan jarak agar keluar dari hambatan dan menentukan apakah karakter bola bersinggungan dengan silinder (Reynolds, 2010).



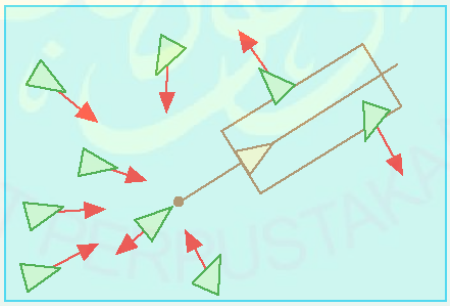
Gambar 2. 4 *Obstacle Avoidance* (Reynold, 2010)

2.1.2.5 *Leader Following*

Perilaku *leader following* menyebabkan satu atau lebih agen untuk mengikuti agen yang terpilih sebagai pemimpin dalam pergerakan berkelompok.

Keseluruhan proses *leader following* dijelaskan oleh langkah–langkah berikut ini (Nugroho dkk, 2010):

- Jika status *boid* "a leader" melakukan aktivitas berputar-putar sampai semua anggota berkumpul,
- anggota mengikuti kemana *leader*-nya berada,
- Kecepatan setiap anggota lebih besar daripada *leader*-nya,
- Jika keadaan *leader* dan anggota sejajar, maka kecepatan *leader* dan anggota disamakan,
- Semua *leader* mencari target berikutnya,
- Semua anggota mengikuti target *leader*-nya.



Gambar 2. 5 *Leader Following* (Reynold, 2010)

Pada gambar di atas merupakan langkah sederhana dalam memisahkan antara *leader* dan anggota *boid*. Dalam hal ini *leader* sangat berperan penting karena kemanapun *leader* pergi maka semua anggota akan selalu mengikutinya sampai semua dalam keadaan sama atau berkumpul bersama. Apabila semua anggota dan *leader*-nya telah berkumpul maka mereka akan mencari tujuan masing masing tanpa bertumpu pada *leader*-nya. Disini kecepatan antar agen

sangat mempengaruhi pergerakan, pada saat mencari *leader*, maka kecepatan anggota lebih tinggi dan apabila telah berkumpul semua, maka kecepatan dari semua anggota akan sama dengan kecepatan *leader*.

2.1.3 *Collision Detection*

Collision detection atau pendeteksian tumbukan adalah proses pengecekan apakah beberapa buah objek spasial saling bertumbuk atau tidak. Jika ternyata ada paling sedikit dua buah objek yang bertumbuk, maka kedua objek tersebut dikatakan saling bertumbukan pada ruang spasial dua dimensi objek yang bertumbuk berarti objek spasialnya beririsan. Teknik pendeteksian tumbukan bisa dikelompokkan menjadi dua macam yaitu *priori detection* dan *post detection*. *Priori detection* adalah pengecekan tumbukan sebelum tumbukan tersebut terjadi, sedangkan *post detection* adalah pengecekan tumbukan setelah tumbukan tersebut terjadi (Maulana, 2010).

Objek-objek ini bisa saja memiliki bentuk yang sangat bervariasi, ada yang berbentuk kotak, segi-n, sampai bentuk pesawat pemain yang sangat mendetail. Untuk mempercepat proses pada *collision detection*, umumnya objek-objek ini direpresentasikan secara *logic* dengan bentuk *primitive* seperti segiempat dan lingkaran (jika pada koordinat dua dimensi), atau kubus dan bola (jika pada koordinat tiga dimensi). Bentuk primitive yang merepresentasikan objek ini biasa disebut sebagai *Bounding Box* atau *Bounding Circle* (Ecky, 2011).

2.1.3.1 *Bounding Circle/Sphere*

Gambar 2. 6 *Bounding Circle* (Claudio de Sa, 2009)

Yang paling sederhana dari semua metode untuk medeteksi terjadinya tumpukan atau persimpangan antar objek adalah *bounding circle*. Pada dasarnya ini merupakan objek lingkaran atau bola. Setiap objek yang direpresentasikan dengan *bounding circle* memiliki titik pusat dan radius. Untuk menguji apakah terjadi tumpukan antar dua objek tersebut yang perlu dilakukan adalah membandingkan jarak antara dua titik pusat dengan jumlah dari dua jari-jari (r1 & r2).

Gambar 2. 7 *Cirecle Circle Colision* (Claudio de Sa, 2009)

- Jika jarak melebihi jumlah, lingkaran itu terlalu jauh untuk memotong.
- Jika jaraknya sama dengan jumlah, lingkaran tesebut saling bersentuhan.
- Jika jarak kurang dari jumlah jari-jari tersebut, lingkaran berpotongan.

2.1.3.2 *Bounding Box*

Gambar 2. 8 *Bounding Box* (Claudio de Sa, 2009)

Solusi yang kedua untuk masalah pendeteksian terjadi tumpukan antar objek adalah *bounding box*. Metode ini sangat ideal untuk objek yang lebih kecil seperti persegi panjang, dan juga sangat cepat untuk memproses.

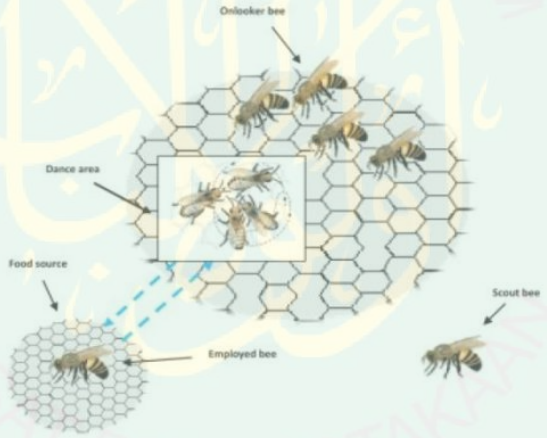
Persegi panjang dalam hal ini di tentukan dari kiri, kanan tepi, atas dan bawah. Untuk menentukan apakah dua persegi panjang berpotongan atau tidak, berikut adalah kondisi untuk memeriksa masalah tersebut :

- Tepi bawah *Rectangle* 1 lebih tinggi dari tepi atas *Rectangle* 2
- Tepi atas *Rectangle* 1 lebih rendah dari tepi bawah *Rectangle* 2
- Tepi kiri *Rectangle* 1 adalah sebelah kanan tepi kanan *Rectangle* 2
- Tepi kanan *Rectangle* 1 adalah sebelah kiri tepi kiri *Rectangle* 2

Gambar 2. 9 *Bounding Box Collision* (Claudio de Sa, 2009)

2.1.4 *Artificial Bee Colony*

Artificial bee colony (ABC) adalah algoritma optimasi yang terinspirasi berdasarkan kecerdasan perilaku lebah dalam mencari sumber makanan (Karaboga, 2005). ABC pertama kali diperkenalkan pada tahun 2005 oleh Dervis Karaboga, dengan mempelajari perilaku lebah dikehidupan nyata dalam mencari nektar sebanyak mungkin dan menyebarkan informasi sumber makanan kepada lebah lain di sarang. Para lebah mencoba memaksimalkan jumlah nektar yang akan di serahkan ke sarang mereka dengan melakukan pembagian tugas kepada setiap lebah. Dalam satu koloni ABC terbagi menjadi tiga jenis lebah; (1) *Employed Bee*, (2) *Onlooker Bee*, (3) *Scout Bee*.



Gambar 2. 10 Bagaimana lebah bekerja untuk menemukan sumber makanan.

(Jatiningsih dkk, 2014)

Dalam algoritma ABC sumber makanan menggambarkan kandidat solusi pada permasalahan optimasi dinotasikan sebagai NF, kemudian banyaknya nektar pada sumber makanan menggambarkan seberapa baik nilai *fitness* pada solusi tersebut. Penetapan nilai *fitness* ditentukan sesuai solusi yang diinginkan. Jumlah *Employed Bee* atau *Onlooker Bee* sama dengan jumlah sumber makanan (Jatiningsih dkk, 2014).

Berikut ini adalah langkah-langkah algoritma ABC menurut (Karaboga, 2005) dalam menyelesaikan permasalahan optimasi :

Initialization Phase

REPEAT

Employed Bees Phase

Onlooker Bees Phase

Sout Bees Phase

Memorize The Best Solutions achieved os far

UNTIL *(Cycle = Maximum Cycle Number bor Maximum CPU time)*

2.1.4.1 *Scouting Phase*

Dalam mencari sumber makanan, koloni lebah mengerahkan pasukan pencari (*scout*). Pasukan ini bertugas mencari sumber makanan baru jika sumber makanan yang ada saat ini telah habis. Pola pencarian oleh lebah pencari ini dilakukan secara acak dari posisi saat ini ke posisi berikutnya sehingga secara matematis dapat dijelaskan seperti pada persamaan (1).

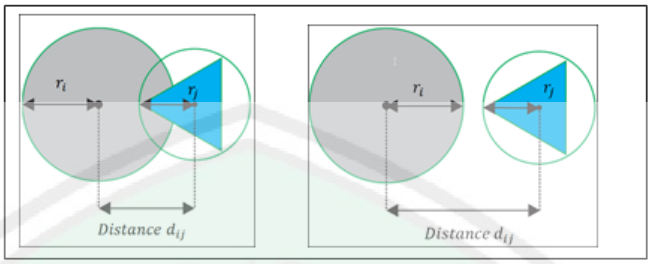
$$X_i' = X_i + rand[-1\ 1] \quad (1)$$

Jika pencarian lebah dibatasi dalam sebuah lingkup ruangan dengan radius (R) , maka persamaan tersebut berubah menjadi seperti persamaan (2).

$$X_i' = X_i + rand[-1\ 1] \times R \quad (2)$$

Saat pencarian, untuk setiap anggota pasukan diharapkan dapat meminimalkan terjadinya tabrakan pada lebah. Jika posisi yang akan dituju oleh lebah terhalang oleh adanya halangan, maka lebah tersebut harus mencari posisi yang baru lagi. Seekor lebah dikatakan bertabrakan jika jarak posisi lebah tersebut

terhadap halangan lebih kecil daripada jumlah kedua radius (r) objek tersebut (lihat Gambar 2.11).



Gambar 2. 11 Ilustrasi saat kedua objek bertabrakan (jarak < ∑radius) dan ketika tidak bertabrakan (jarak > ∑radius).

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

$$d_{ij} > (r_{bee} + r_{obs}) \quad (3)$$

$$f_{collision} = d_{ij} - (r_{bee} + r_{obs}) > 0 \quad (4)$$

Persamaan (4) dibandingkan terhadap nilai 0 mana yang lebih rendah sehingga menghasilkan nilai C.

$$C = \min_0 \left(d_{ij} - (r_{bee} + r_{obs})\right) \quad (5)$$

Persamaan (5) tersebut menjadi persyaratan dasar untuk pemilihan posisi untuk persamaan (2). Jika nilai dari C kurang dari 0, maka lebah diharuskan melakukan pencarian ulang posisi yang baru sesuai dengan persamaan (6).

$$X_i' = f(x) = \begin{cases} X_i + rand[-1\ 1] \times R, & C < 0 \\ X_i', & C \geq 0 \end{cases} \quad (6)$$

2.1.4.2 *Onlooker Phase*

Lebah pengintai yang telah mendapatkan posisi, kembali ke sarang dan menginformasikan data yang telah dia dapat pada posisi tersebut kepada lebah

pengamat yang berada di sarang. Dalam penelitian ini, data yang digunakan yaitu diambil dari parameter yang melekat pada target dan parameter yang melekat pada lebah itu sendiri. Parameter yang diambil dan disebarluaskan ke lebah yang lain yaitu adalah jarak lebah terhadap target saat ini. Untuk mengukur jarak lebah ke target, digunakan persamaan *Euclidian*.

$$d^2 = (x_j - x_i)^2 + (y_j - y_i)^2 \quad (7)$$

$$d = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2} \quad (8)$$

2.1.4.3 *Employed Phase*

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

$$n = \begin{cases} X_i + rand[-1\ 1] \times R, & n < 0 \\ n - 1, & n \geq 0 \end{cases} \quad (9)$$

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

2.2 Penelitian Terkait

2.2.1 Pergerakan Otonom Pasukan Berbasis Algoritma *Boids* Menggunakan Metode *Particle Swarm Optimization*

Penelitian ini dilakukan oleh Syahri Mu'min, Mochammad Hariadi, dan Supeno Mardi Susiki Nugroho Program Studi Teknik Elektro, Fakultas Teknologi Industri, Institut Teknologi Sepuluh Nopember Surabaya pada tahun 2015. Pada penelitian ini peneliti membahas bagaimana memodelkan algoritma *Boids* dengan memperi kemampuan PSO (*Particle Swarm Optimization*) untuk mencapai posisi optimum menciptakan kemungkinan untuk secara otomatis menghasilkan jalan *non deterministic* kerumunan pasukan dari satu posisi tertentu. Penelitian ini fokus pada pembuatan pola bergerak yang halus dan fleksibel realistis bagi pasukan virtual dengan memanfaatkan fasilitas komputasi yang ditawarkan oleh PSO.

Pada penelitian ini peneliti berhasil mewujudkan representasi dari kehidupan buatan secara realitas dari pergerakan. Pergerakan pasukan dihitung dengan menggabungkan semua vektor *steering behaviour*. Pasukan melakukan perlambatan dan percepatan untuk menyelaraskan posisi antar agen atau pasukan. Algoritma PSO digunakan untuk mengoptimalkan model *boids* dengan mengoptimalkan koefisien dari vektor bergerak untuk meminimalkan fungsi *cost* dan *flocking* terlihat lebih bagus.

2.2.2 *Simulating The Movement Of The Crowd In An Environment Using Flocking*

Penelitian ini dilakukan oleh Meilany Dewi Program Studi Teknik Elektro, Fakultas Teknologi Industri, Institut Teknologi Sepuluh Nopember Surabaya pada tahun 2011. Pada penelitian ini peneliti membahas bagaimana memodelkan algoritma *Flocking* dan *Obstacle Avoidance* dengan *Collision Detection* untuk mensimulasikan pergerakan kerumunan orang dan meminalkan frekuensi jumlah tabrakan antar pengunjung serta terhadap hambatan statis dan dinamis.

Pada penelitian ini peneliti berhasil mensimulasikan perkerakan kerumunan orang dan meminalkan frekuensi tabrakan antar pengunjung. Tetapi meningkatnya jumlah populasi akan meningkatkan jumlah frekuensi tabrakan yang terjadi, dengan 0.03% pada populasi 100 dan 1.6% pada populasi 500. Meningkatnya jumlah populasi juga akan menurunkan tingkat kecepatan rata–rata pengunjung untuk mencapai target pintu keluar utama. Dengan jumlah populasi awal 10 kemudian ditingkatkan menjadi 50 kali, kecepatan rata–rata mengalami penurunan sebesar 98%. Meningkatnya waktu yang diperlukan oleh pengunjung untuk mencapai target pintu keluar utama dengan semakin meningkatnya populasi. Dengan jumlah populasi awal 10 kemudian ditingkatkan menjadi 50 kalinya, maka waktu yang diperlukan mengalami peningkatan sebesar 97.9%.

2.2.3 Simulasi Pergerakan Evakuasi Bencana Tsunami Menggunakan Algoritma *Boids* dan *Pathfinding*

Penelitian ini dilakukan oleh I Made Pasek Mudhana, Mauridhi Hery Purnomo, dan Supeno Mardi Susiki Nugroho Program Studi Teknik Elektro,

Institut Teknologi Sepuluh Nopember Surabaya pada tahun 2014. Pada penelitian ini peneliti membahas bagaimana memodelkan algoritma *Boids* dan *Pathfinding* untuk mensimulasikan pergerakan kerumunan orang untuk bergerak menuju suatu titik evakuasi pada saat terjadinya gempa bumi yang diperkirakan menimbulkan terjadinya tsunami. Simulasi ini menentukan jarak terdekat/terpendek dari posisi individu, meminimalkan terjadinya tabrakan dalam menghindari segala hambatan yang ditemui baik hambatan statis maupun dinamis yang ditemui pada saat melewati rute jalan yang telah ditentukan.

Penggunaan algoritma *boids* dan *pathfinding* berhasil diterapkan sebagai algoritma untuk simulasi kerumunan menuju target tertentu. Peubah simulasi adalah jumlah penduduk dan keceparan rata-rata menuju target. Pertambahan jumlah kerumunan membutuhkan selang waktu yang lebih menuju target yang diinginkan.

Algoritma *boids* dan *pathfinding* berhasil memenuhi waktu maksimal yang diperlukan unutk melakukan evakuasi yaitu kurang dari 59 menit.

2.2.4 Perilaku Agen Cerdas Berbasis *BOIDS* Untuk Simulasi Kerumunan Pada Keadaan Bahaya

Penelitian ini dilakukan oleh Supeno Mardi Susiki Nugroho, Ervan Wahyudi, Diah Puspito W, Christyowidiasmoro, Mochamad Hariadi, Mauridhi H Purnomo Program Studi Teknik Elektro, Institut Teknologi Sepuluh Nopember Surabaya pada tahun 2010. Pada penelitian ini peneliti membahas bagaimana memodelkan algoritma *Boids* untuk mensimulasikan pergerakan kerumunan manusia dalam kondisi panik agar dapat segera keluar menuju pintu utama dari

gedung bila terjadi kebakaran. Skenario yang dimodelkan dalam penelitian ini adalah bagaimana menjaga jarak antar orang, jika berpapasan dengan orang lain disekitarnya atau pada saat menemui halangan dinding maka pergerakan individu dalam kerumunan harus bisa segera memberikan aksi dalam menghadapi kondisi tersebut untuk menuju aksi berikutnya.

Pada penelitian ini peneliti berhasil membuktikan bahwa agen berhasil menghindari tabrakan antar agen dan agen lain, berhasil menghindari tabrakan antar agen dan objek. Pada pengujian keadaan bahaya para agen berhasil menghindari pintu keluar yang menandakan adanya sinyal bahaya, maka agen tidak akan melewatinya dan akan mencari jalan keluar lainnya yang paling dekat selain pintu keluar yang ada tanda bahayanya tadi.

2.2.5 Penerapan Algoritma *Collision Detection* dan *Boids* Pada *Game Dokkaebi Shooter*

Penelitian ini dilakukan oleh Lia Musfiroh, Ahmad Jazuli, Anastasya Latubessy Program Studi Teknik Informatika, Fakultas Teknik, Universitas Muria Kudus Gondangmanis pada tahun 2014. Penelitian ini bertujuan untuk melatih ketangkasan dengan penerapan-penerapan algoritma *boids* pada pergerakan musuh dan algoritma *collision detection* pada serangan musuh.

Game ini merupakan tembakan secara vertikal dengan berfokus pada pengumpulan skor dan melawan musuh utama yaitu *dokkaebi*. *Dokkaebi* adalah hantu korea berwujud monster. *Game* ini tidak hanya bergantung pada matinya musuh namun juga tergantung *timer* untuk memenangkan *game* ini, musuh dapat menyerang pemain, dan dapat menyimpan *score*.

2.2.6 *Autonomous Agent Based NPC Swarm Attack Behaviour Using Bee Colony Algorithm*

Penelitian ini dilakukan oleh Wilujeng Jatiningsih, Eko Mulyanto Yuniarno, dan Mochamad Hariadi Program Studi Teknik Elektro, Institut Teknologi Sepuluh Nopember Surabaya pada tahun 2014. Pada penelitian kali ini peneliti membahas bagaimana memodelkan algoritma *Bee Colony* untuk pelajaran yang dilakukan NPC agar dapat menyerang musuh dalam formasi kerumunan. Dengan menggunakan empat fungsi *benchmark* yang umum untuk mencari jumlah siklus yang dibutuhkan lebah untuk menuju konvergen, ukuran populasi = 125, jangkauan parameter mulai dari -100 hingga +100, jumlah siklus dibatasi 50 kali, batas pengabaian = 10 dan posisi sumber makanan pada (0,0). Hasil yang diperoleh menunjukkan bahwa lebah menuju konvergen pada siklus 0-25.

2.2.7 Penerapan Algoritma *Artificial Bee Colony* dalam Aplikasi Penjadwalan Pelajaran untuk Sekolah Menengah Pertama

Penelitian ini dilakukan oleh Rakhmad Fajar Nugroho dan Mewati Ayub, Jurusan Teknik Informatika, Fakultas Teknologi Informasi, Universitas Kristen Maranatha pada tahun 2013. Pada penelitian ini peneliti membahas bagaimana menerapkan algoritma ABC pada proses penjadwalan dengan membuat sebuah aplikasi yang dapat menghasilkan jadwal pelajaran dengan kasus bentrok semisal mungkin.

Pada penelitian ini peneliti telah berhasil membuat aplikasi yang mampu menghasilkan jadwal pelajaran dengan menggunakan algoritma *Artificial Bee Colony* (ABC) untuk mengurangi jumlah bentrokan. Algoritma ABC sangat

dipengaruhi oleh fungsi random yang digunakan, terutama untuk penerapan eksploitasi dengan metode *random*, sehingga hasil yang didapatkan dari beberapa pengujian tidak akan sama persis walaupun parameter yang diinput tidak berbeda.

2.2.8 Optimasi Rute Kendaraan dengan Kapasitas Menggunakan Modifikasi Algoritma *Artificial Bee Colony*

Penelitian ini dilakukan oleh Muhammad Zidny Alam, Program Studi Matematika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Syarif Hidayatullah Jakarta pada tahun 2013. Pada penelitian ini peneliti membahas bagaimana menerapkan algoritma ABC untuk menyelesaikan *Capacitated Vehicle Routing Problem* (CVRP) agar didapatkan hasil terbaik yaitu biaya total perjalanan yang minimum.

Pada penelitian ini metode optimasi ABC merupakan metaheuristik yang dapat menjadi salah satu alternatif dan mampu diterapkan untuk menyelesaikan permasalahan CVRP dengan baik. Percobaan yang dilakukan untuk menyelesaikan data permasalahan untuk pelanggan kurang dari 100, algoritma ABC mampu menghasilkan solusi yang lebih baik dari solusi algoritma *Branch and Cut*. Sedangkan untuk data permasalahan yang memiliki pelanggan lebih dari 100 algortima ABC mampu menyelesaikan namun tidak lebih baik dari algoritma pembanding.

2.2.9 Implementasi Algoritma Lebah Untuk Pencarian Jalur Terpendek Dengan Mempertimbangkan Heuristik

Penelitian ini dilakukan oleh Dian Nurdiana yang diterbitkan pada Jurnal Pendidikan Matematika tahun 2015. Pada penelitian ini peneliti membahas bagaimana menerapkan algoritma ABC sebagai metode pencarian jalur terpendek yang tidak hanya mempertimbangkan jarak saja, tetapi mempertimbangkan heuristik lainnya seperti kemacetan, lampu jalan, jalan tol, rawan bencana dan keamanan. Sehingga rute yan dihasilkan merupakan rute yang optimum.

Penelitian ini telah berhasil mengimplementasikan algoritma lebah untuk menyelesaiakan pencarian jalur terpendek secara optimal. Perangkat lunak yang dibangun berhasil merekomendasikan jalur mudik yang paling optimum dari kota pemberangkatan ke kota tujuan kepada pengguna.

Pencarian jalur terpendek yang dilakukan untuk pengujian sebanyak 100 kali percobaan. Dari 100 kali percobaan tersebut, 90 kali percobaaan menghasilkan rute dan panjang jalur terpendek, sedangkan 10 kali percobaan lainnya tidak menghasilkan rute dan panjang jalur terpendek. Modus pencarian jalur terpendek hasil simulasi adalah 1044,9907 Km. Dengan kemunculan sebesar 26 kali. Nilai rata–rata sebesar 1058,395 Km dengan standar deviasi sebesar 20,33801 Km. Dari standar deviasi menunjukan hasil simulasi yang didapat mempunyai nilai terbesar 1078,7330 Km dan nilai terkecil 1038,0569 Km.

2.2.10 Penyelesaian *Travelling Salesman Problem* (TSP) Dengan Menggunakan *Artificial Bee Colony*

Penelitian ini dilakukan oleh Rendra Firma Pratama, Purwanto, dan Mohammad Yasin, Universitas Negeri Malang. Pada penelitian ini peneliti membahas bagaimana menerapkan algoritma ABC untuk menyelesaikan permasalahan TSP. ABC sebagai alat bantu perhitungan sangat membantu para *sales* untuk menentukan rute perjalanan yang lebih minimum dalam setiap permasalahan *Travelling Salesman Problem* yang ditemui. Program *Artificial Bee Colony* ini mampu menyelesaikan masalah *Travellin Salesman Problem* dengan jumlah konsumen yang besar.

BAB III

DESAIN DAN RANCANGAN SIMULASI

3.1 Tahapan Penelitian

Tahapan-tahapan penelitian yang dilakukan pada penelitian ini sesuai dengan diagram alur penelitian (lihat Gambar 3.1).

Gambar 3. 1 Diagram alur penelitian

3.1.1 Studi literatur

Pada tahap ini dilakukan berbagai pengumpulan informasi terkait beberapa hal, yakni :

- Pengumpulan informasi tentang thawaf
- Pengumpulan informasi tentang pembuatan simulasi 3D dengan Unity
- Pengumpulan informasi tentang metode *Boids* dan *Artificial Bee Colony*
- Pengumpulan informasi tentang penelitian terkait

3.1.2 Desain skenario

Pada tahap ini dirancang bagaimana simulasi akan dijalankan dan metode akan bekerja.

3.1.3 Desain NPC dan *Enviroment*

Pada tahap ini dirancang bagaimana desain *interface* dan desain NPC-nya.

3.1.4 Pembuatan simulasi

Proses pembuatan simulasi dibangun dengan memanfaatkan unity3d *engine* dengan menggunakan bahasa C# dan Javascript. Scripting dilakukan dengan fasilitas Visual Studio.

3.1.5 Uji coba metode

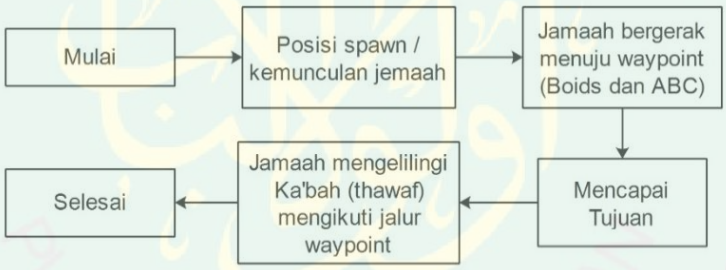
Uji coba dalam penelitian ini dilakukan pada simulasi dan karakter jemaah yang telah diimplementasi algoritma *Boids* dan ABC, kemudian dicek apakah output yang keluar sudah sesuai dengan rule yang dibuat.

3.1.6 Penyusunan laporan

Penyusunan laporan akhir merupakan bagian dokumentasi dari keseluruhan pelaksanaan penelitian yang diharapkan dapat bermanfaat bagi penelitian selanjutnya.

3.2 Analisis dan Perancangan Simulasi

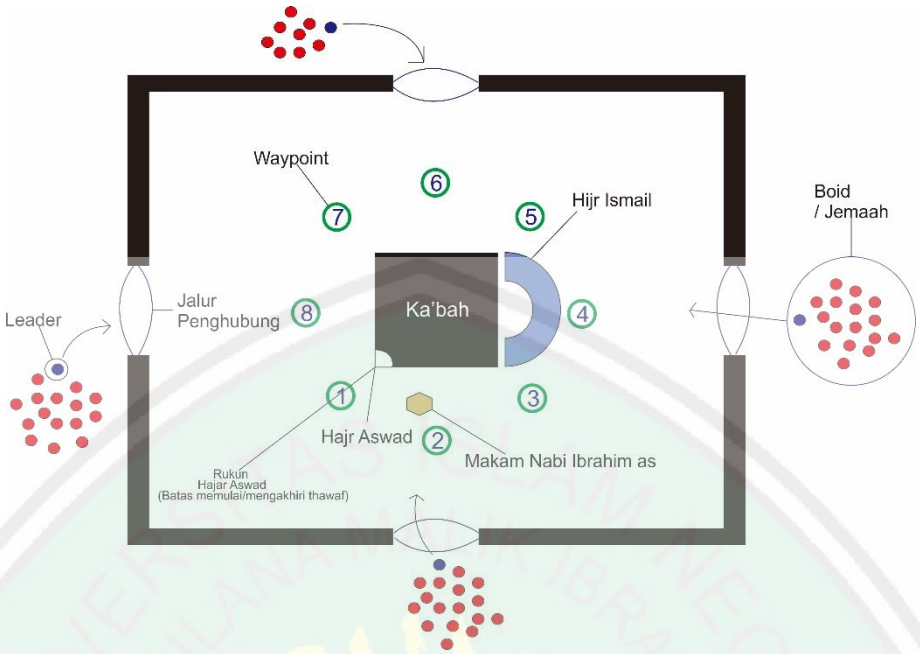
Simulasi ini dirancang untuk menerapkan algortima *Boids* dan *Artificial Bee Colony* pada kerumunan jemaah yang sedang mengitari atau mengelilingi Ka'bah (thawaf). Terdapat beberapa karakter pada simulasi ini yang di desain agar simulasi sangat menarik dilihat. Karakter tersebut berupa beberapa karakter NPC (*Non Player Character*). NPC tersebut adalah jemaah yang sedang mengelilingi Ka'bah dan akan diberikan sebuah perilaku cerdas dengan menginisiasi algoritma *Boids* dan ABC. Metode ini digunakan untuk perubahan perilaku jemaah saat bergerombol dan menuju target. Pada simulasi, saat jemaah muncul maka algoritma *Boids* dan ABC otomatis akan bekerja (lihat Gambar 3.2). Simulasi ini dirancang pada PC dan dibangun dengan grafis 3 dimensi.



Gambar 3. 2 Diagram alur kerja simulasi

3.2.1 Keterangan Umum Simulasi

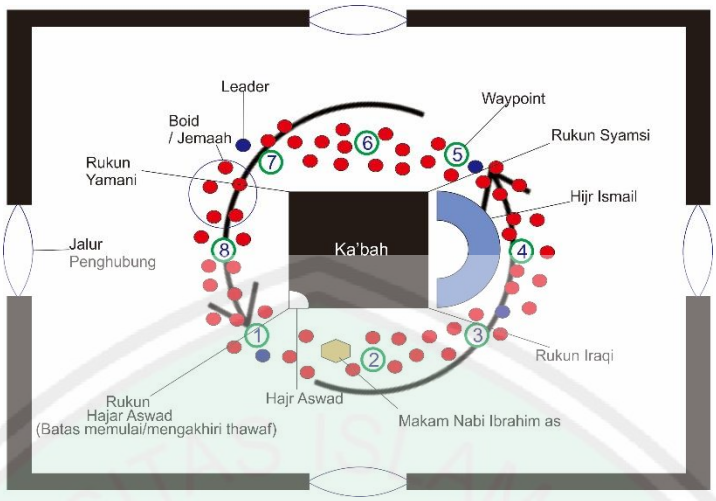
Didalam simulasi ini jemaah harus mengitari/mengelilingi Ka'bah. Berikut adalah skenario latar simulasi yang disajikan dalam bentuk gambar.



Gambar 3. 3 Posisi kemunculan jemaah bergerak menuju area Ka'bah

Pada bagian ini (lihat Gambar 3.3) posisi kemunculan jemaah akan tersebar secara acak pada titik–titik area persebaran yang akan ditentukan. Pada jemaah terdapat satu atau lebih agen yang terpilih sebagai *leader* (pemimpin) gerombolan tersebut. Kemana pemimpinnya bergerak anggotanya akan mengikutinya.

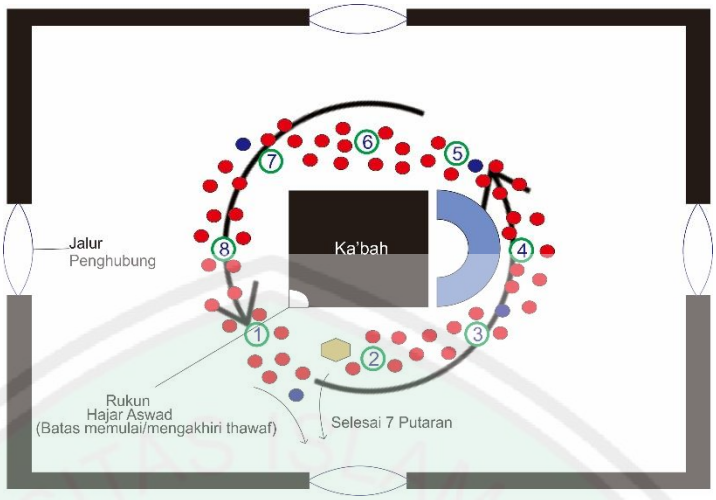
Jemaah akan bergerak menuju *waypoint* terdekat yang ada disekeliling Ka'bah. Pada saat kemunculan dimulai, algoritma *Boids* dan ABC akan bekerja. Algoritma ABC akan diberikan kepada *leader* yang berfungsi untuk bergerak menuju target (*waypoint*) yang ada disekeliling Ka'bah. Dan algoritma *boids* akan menjaga agar tidak saling bertabrakan antar jemaah dengan jemaah lainnya, menjaga kecepatan agar saling bergerak bersamaan, dan menjaga agar jemaah tetap dengan kelompoknya saat menuju target.



Gambar 3. 4 Jemaah mengelilingi Ka'bah

Setalah jemaah berada disekitar area Ka'bah, pada bagian ini (lihat Gambar 3.4) jemaah akan mengitari/mengelilingi Ka'bah (thawaf) mengikuti jalur-jalur *waypoint* yang ada disekeliling Ka'bah. Arah melakukan mengelilingi Ka'bah (thawaf) adalah berlawan dengan arah jarum jam. Bila *leader* menuju titik *waypoint* kesatu, maka anggota jemaah lainnya akan mengikutinya dan seterusnya. Disekitar area Ka'bah terdapat hambatan statis, yaitu Hijr Ismail & Makam Nabi Ibrahim as. Setiap jemaah akan menghindari rintangan tersebut agar tidak terjadi tabrakan dengan *obstacle*.

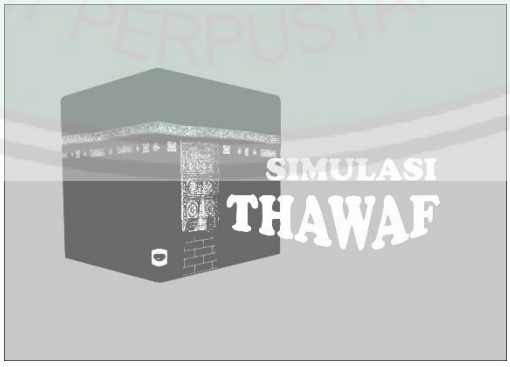
Jemaah akan mengelilingi Ka'bah sebanyak 7 kali putaran. Pada setiap awal putaran (pertama s.d ketujuh) dimulai dari Rukun Hajar Aswad dengan menghadapkan muka sambil mengangkat tangan, mengkecup dan membaca do'a. Jemaah akan menbaca do'a mulai Rukun Hajar Aswad sampai Rukun Yamani. Pada saat setiap kali sampai di Rukun Yamani mengangkat tangan tanpa dikecup sambil membaca do'a. Diantara Rukun Yamani dan Rukun Hajar Aswad terdapat do'a yang dibaca oleh jemaah.



Gambar 3. 5 Jemaah keluar dari area Ka'bah

Setelah jemaah selesai mengelilingi Ka'bah sebanyak tujuh putaran, jemaah akan keluar dari area Ka'bah yang menandakan telah selesai melakukan thawaf. Di sini *leader* yang akan berperan mengecek apakah telah melakukan tujuh putaran. Jika tujuh putaran telah selesai yang diawali dan diakhiri dari Rukun Hajar Aswad, *leader* akan bergerak keluar dari area Ka'bah dan jemaah anggotanya akan mengikutinya.

3.2.2 Desain *Interface* Simulasi



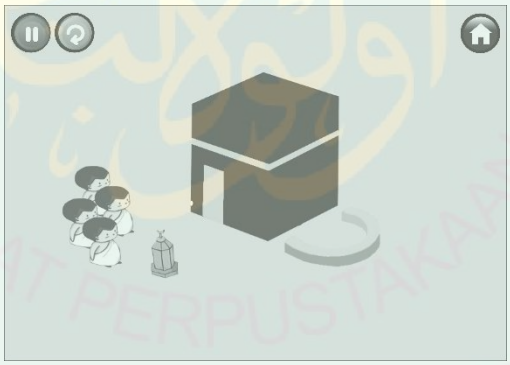
Gambar 3. 6 Tampilan *Splash screen*

Gambar 3.6 merupakan tampilan awal simulasi. Dimana tiap kata akan muncul satu-persatu mulai dari kata paling atas. Tiap kata berselang 2 detik

sampai akhir terlihat seperti gambar diatas. Sehingga *splash screen* diatas akan muncul selama 4 detik. Terdapat pula sound saat *splash screen* ini berjalan.

Gambar 3. 7 Tampilan Menu

Gambar 3.7 merupakan tampilan menu dalam simulasi yang akan muncul setelah tampilan *splash screen*. Di dalam menu terdapat 3 tombol, yaitu tombol mulai untuk memulai simulasi, tombol informasi yang berisi tentang simulasi, dan tombol keluar untuk menutup aplikasi.



Gambar 3. 8 Tampilan Simulasi

Gambar 3.8 merupakan tampilan ketika simulasi berjalan yang akan muncul ketika tombol mulai dipilih. Jemaah akan mengelilingi Ka'bah untuk melakukan tawaf.

Gambar 3. 9 Tampilan Petunjuk Do'a

Gambar 3.9 merupakan tampilan petunjuk do'a yang muncul ketika jemaah mengelilingi Ka'bah (thawaf). Ketika jemaah mengelilingi Ka'bah, akan muncul tampilan *text box* yang berisi bacaan do'a yang harus dibaca oleh jemaah.

3.2.3 Deskripsi NPC

Gambar 3. 10 Jemaah (NPC) Laki–laki

Gambar 3.10 merupakan karakter jemaah laki-laki yang terdapat dalam simulasi yang akan mengelilingi Ka'bah.

3.3 Perancangan Algoritma *Boids*

Di dalam pergerakan kelompok *boids* terdapat beberapa hal yang diperhitungkan. Yang pertama, agen atau individu *boids* sebagai satuan entitas di dalam kelompok. Kedua, vektor kecepatan yang merupakan kecepatan agen pada

saat tertentu. Ketiga, *neighborhood* yang merupakan lingkungan dalam radius tertentu dari *boids* yang menjadi jangkauan pengamatan *boids* terhadap *boids* lain.

Di dalam pemrograman perilaku *boids* pada penelitian ini terdapat beberapa variabel yang diatur atau dikomputasi pada NPC jemaah. Variabel-variabel tersebut adalah vektor posisi, kecepatan, gaya maksimum dan kecepatan maksimum.

- Vektor Posisi (*Vector Position*)

  Vektor posisi menyatakan koordinat dari unit NPC jemaah. Vektor posisi ini bisa berdimensi dua maupun dimensi tiga. Pada penelitian ini yang digunakan adalah vektor posisi berdimensi tiga. Nilai dari vektor posisi tergantung dari titik acuan atau referensi yang digunakan. Jika menggunakan acuan lokal (*local position*) terhadap diri sendiri, maka unit NPC jemaah selalu berada pada koordinat (0,0,0). Sedangkan untuk acuan global, maka koordinat unit NPC jemaah dihitung dari titik referensi global pada *game scene* Unity 3D.

- Kecepatan (*Velocity*)

  Kecepatan unit NPC jemaah menentukan laju pergerakan. Semakin besar kecepatan, maka pergerakan semakin cepat. Di dalam simulasi ini kecepatan yang dimiliki oleh unit NPC jemaah nilainya tidak statis. Nilai kecepatan unit NPC dipengaruhi oleh besarnya gaya yang diterima dan berubah-ubah menyesuaikan dengan kondisi lingkungan sekitarnya. Variabel kecepatan ini nilainya berupa vektor.

- Gaya Maksimum (*Maximum Force*)

Gaya maksimum digunakan untuk membatasi besarnya gaya yang diterima oleh unit NPC jemaah. Pembatasan ini dilakukan agar pergerakan unit NPC jemaah dapat selaras dengan unit NPC jemaah lain disekitarnya. Nilai gaya maksimum berupa besaran skalar.

- Kecepatan Maksimum (*Maximum Velocity*)

Sebagaimana nilai kecepatan yang dipengaruhi oleh besarnya gaya yang diterima, maka besarnya kecepatan maksimum dipengaruhi oleh besarnya gaya maksimum yang dapat diterima.

Implementasi algoritma *boids* melibatkan tiga perilaku sederhana : *alignment, cohesion, separation*. Tiga perilaku tersebut dipengaruhi oleh lingkungan di mana setiap *boids* berada. Lingkungan tersebut disebut juga *neighborhood*, yaitu suatu area di sekitar setiap NPC jemaah yang digunakan untuk mencari NPC jemaah lain dan mengambil nilai variabel–variabelnya. Variabel-variabel tersebut nantinya akan dikomputasi untuk menghasilkan tiga perilaku *boids* tersebut.

Perilaku *alignment* dipengaruhi oleh kecepatan dari agen NPC jemaah dengan NPC jemaah lain. Gaya yang digunakan untuk bergerak menyesuaikan pergerakan lingkungan dihitung dengan mengambil rata–rata kecepatan NPC jemaah lain. Rata–rata ini kemudian dinormalisasi dan dikalikan dengan kecepatan maksimum NPC jemaah. Hasil perkalian tersebut adalah kecepatan dari perilaku *alignment* yang besar dan arahnya selaras antara satu NPC jemaah dengan NPC jemaah yang lain sehingga secara matematis dapat dijelaskan seperti pada persamaan (1).

$$\sum_{n \in N} Normalize\,(n_{vel}) \quad (1)$$

Perilaku *separation* dihitung berdasarkan posisi dari agen NPC jemaah dengan NPC jemaah lain yang ada di sekitarnya. Pertama dihitung selisih dari posisi agen NPC jemaah dengan NPC jemaah lain. Kemudian semua selisih tersebut dijumlahkan. Hasil penjumlahan tersebut dibagi dengan jumlah NPC jemaah lain untuk mendapatkan nilai rata-ratanya. Nilai rata-rata yang didapatkan kemudian dinormalisasi dan dikalikan dengan kecepatan maksimal dari NPC jemaah. Hasil akhir dari perhitungan ini merupakan kecepatan *separation* yang mengatur agar NPC jemaah bergerak saling menjauh ketika berada pada posisi yang terlalu dekat satu sama lain. Secara matematis dapat dijelaskan seperti pada persamaan (2).

$$\sum_{n \in N} Normalize\,(agent_{pos} - n_{pos}) \quad (2)$$

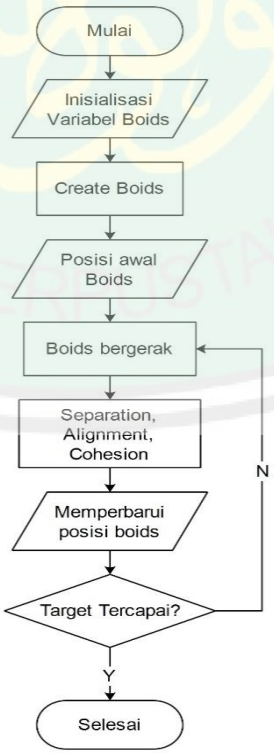
Sama halnya dengan perilaku *separation*, perilaku *cohesion* dihitung berdasarkan posisi dari agen NPC jemaah dengan NPC jemaah lain yang ada di sekitarnya. Yang berbeda adalah arah dari gerak NPC jemaah. Arah gerak dari perilaku *cohesion* NPC jemaah berlawanan dengan arah gerak dari perilaku *separation* NPC jemaah. Karena itu langkah pertama dibuat berkebalikan. Pertama dihitung selisih dari posisi NPC jemaah lain dengan agent NPC jemaah. Kemudian semua selisih tersebut dijumlahkan. Hasil penjumlahan tersebut dibagi dengan jumlah NPC jemaah lain untuk mendapatkan nilai rata-ratanya. Nilai rata-rata yang didapatkan kemudian dinormalisasi dan dikalikan dengan kecepatan maksimal dari NPC jemaah. Hasil akhir dari perhitungan ini merupakan kecepatan

cohesion yang mengatur agar jemaah bergerak saling mendekat ketika berada pada posisi yang terlalu jauh satu sama lain. Secara matematis dapat dijelaskan seperti pada persamaan (3).

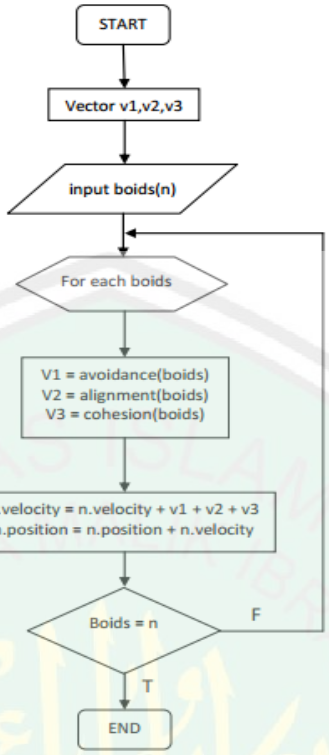
$$\frac{\sum_{n \in N} n_{pos}}{|N|} \quad (3)$$

Untuk menentukan kapan jemaah berlaku *separation* dan kapan berlaku *cohesion* digunakan *threshold* atau ambang batas jarak untuk masing-masing perilaku. *Threshold* untuk *cohesion* nilainya lebih besar dari pada *threshold separation*. Perilaku *cohesion* dimulai ketika posisi NPC jemaah lebih dari *threshold cohesion*. Perilaku *separation* dimulai ketika posisi NPC jemaah kurang dari *threshold separation*.

3.3.1 *Flowchart* Algoritma *Boids*



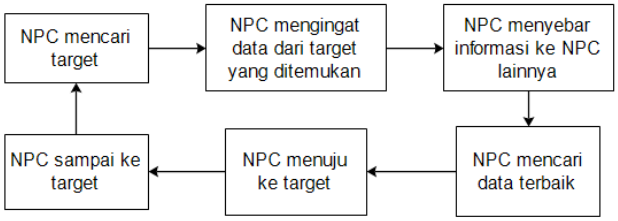
Gambar 3. 11 *Flowchart* Algoritma *Boids*



Gambar 3. 12 *Flowchart* Gerak Posisi *Boids*

3.4 Perancangan Algoritma ABC (*Artificial Bee Colony*)

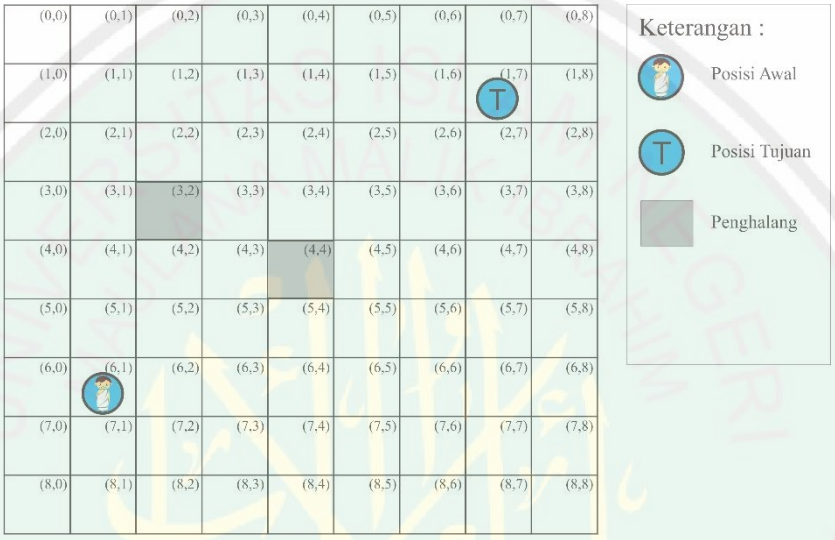
Dalam kehidupan nyata, koloni lebah memiliki kerja sama *team* yang sangat erat. Setiap lebah dalam koloni saling terkoordinasi antara satu sama lainnya dan memiliki tugasnya masing-masing sesuai dengan tipenya. Kemudian cara kerja lebah tersebut diadopsi ke dalam perilaku NPC. NPC dianalogikan sebagai lebah dan makanan dianalogikan sebagai target yang di cari oleh NPC (lihat Gambar 3.14).



Gambar 3. 13 Diagram kerja dalam koloni lebah

3.4.1 Simulasi Perhitungan Algoritma ABC

Untuk lebih memahami algoritma ABC, bisa dilihat dari contoh kasus seperti terlihat pada Gambar 3.12. Diasumsikan dalam simulasi terdapat 2 objek yaitu NPC dan target (T), selain itu juga ada halangan. Objek NPC dan T digunakan untuk menunjukkan lokasi/posisi awal dan tujuan.



Gambar 3. 14 Kondisi awal pencarian ABC

Proses pertama yang dilakukan NPC adalah mencari tahu jalan menuju target, kemudian mencari dan memutuskan langkah yang akan diambil. Apabila langkah menuju target terhalang penghalang, maka ulangi pencarian dengan X'_{ij} baru, sampai menemukan jalan terpendek menuju target. Untuk lebih jelas, berikut adalah langkah-langkah perhitungan dalam pencarian jalur menuju target. Rumus untuk perhitungan X'_{ij} , *C* dan fit_{jarak} dapat dilihat pada persamaan 2, 6, dan 8.

1. Tentukan posisi awal NPC, posisi target, posisi halangan.

   Posisi awal NPC = (6,1)

   Posisi target = (1,7)

Posisi halangan = (3,2), (4,4)

2. Hitung pola pencarian NPC posisi berikutnya dari posisi awal yang dilakukan secara acak pada ruang lingkup (R).

$$X_i' = X_i + rand[-1\ 1] \times R$$

X_i' = 6 + (-1) * 2 Solusi kemungkinan posisi baru :

= 6 + (-2) = 4 $X_{ij}' = (4,1)$

X_j' = 1 + 0 * 2

= 1 + 0 = 1

3. Dicek pada posisi itu terdapat halangan atau tidak. Jika nilai dari C kurang dari 0, maka NPC diharuskan melakukan pencarian ulang posisi baru. Karena terdapat dua halangan, maka dicek semua posisi baru NPC (4,1) ke halangan (3,2) dan posisi baru NPC (4,1) ke halangan (4,4).

$$C = \min_0 \left(d_{ij} - (r_{bee} + r_{obs})\right)$$

$$d_{ij} = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}$$

$d_{ij} = \sqrt{(3-4)^2 + (2-1)^2}$ => jarak NPC ke halangan (3,2)

$d_{ij} = \sqrt{(-1)^2 + 1^2}$

$d_{ij} = \sqrt{2}$

$d_{ij} = 1{,}414214$

rBee = 0,5 ; rObs = 0,5

$$C = \min_0 (1{,}414214 - (0{,}5 + 0{,}5))$$

$$C = \min_{0}(1{,}414214 - 1)$$

$$C = \min_{0}(0{,}414214)$$

$d_{ij} = \sqrt{(4-4)^2 + (4-1)^2}$ => jarak NPC ke halangan (4,4)

$d_{ij} = \sqrt{0^2 + 3^2}$

$d_{ij} = \sqrt{9}$

$d_{ij} = 3$

rBee = 0,5 ; rObs = 0,5

$$C = \min_{0}(3 - (0{,}5 + 0{,}5))$$

$$C = \min_{0}(3 - 1)$$

$$C = \min_{0}(2)$$

4. Mencari nilai optimal dari jarak NPC awal (6,1) ke target (1,7) dan jarak NPC baru (4,1) ke target (1,7). Jika nilai jarak NPC baru lebih kecil daripada jarak NPC awal, maka NPC pindah ke posisi baru.

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

$d = \sqrt{(1-6)^2 + (7-1)^2}$ => jarak posisi awal NPC ke target

$d = \sqrt{(-5)^2 + 6^2}$

$d = \sqrt{25 + 36}$

$d = \sqrt{61}$

$d = 7{,}81025$

$d = \sqrt{(1-4)^2 + (7-1)^2}$ => jarak posisi baru NPC ke target

$d = \sqrt{(-3)^2 + 6^2}$

$d = \sqrt{9 + 36}$

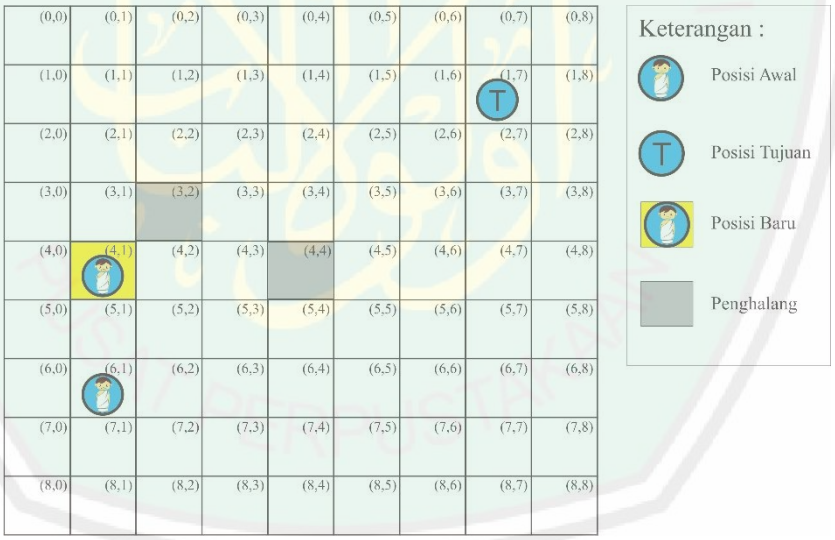
$d = \sqrt{45}$

$d = 6{,}708204$

Nilai jarak posisi baru ke target < Nilai jarak posisi awal ke target

6,708204 < 7,81025

Karena nilai jarak NPC baru lebih kecil daripada NPC awal, posisi baru menjadi posisi awal untuk pencarian posisi selanjutnya.



Gambar 3. 15 Iterasi ke-1 pencarian ABC

5. Karena tujuan belum ditemukan, maka pencarian dilanjutkan untuk menghitung pola pencarian NPC posisi berikutnya yang dilakukan secara acak pada ruang lingkup (R) dari posisi awal yang telah diperbarui.

$X_i' = X_i + rand[-1\ 1] \times R$

X_i' $= 4 + 0 * 2$ Solusi kemungkinan posisi baru :

$= 4 + 0 = 4$ $X_{ij}' = (4,3)$

X_j' $= 1 + 1 * 2$

$= 1 + 2 = 3$

6. Dicek pada posisi itu terdapat halangan atau tidak. Jika nilai dari C kurang dari 0, maka NPC diharuskan melakukan pencarian ulang posisi baru. Karena terdapat dua halangan, maka dicek semua posisi baru NPC (4,3) ke halangan (3,2) dan posisi baru NPC (4,3) ke halangan (4,4).

$$C = \min_0 \left(d_{ij} - (r_{bee} + r_{obs})\right)$$

$$d_{ij} = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}$$

$d_{ij} = \sqrt{(3-4)^2 + (2-3)^2}$ => jarak NPC ke halangan (3,2)

$d_{ij} = \sqrt{(-1)^2 + (-1)^2}$

$d_{ij} = \sqrt{2}$

$d_{ij} = 1{,}414214$

rBee = 0,5 ; rObs = 0,5

$$C = \min_0 (1{,}414214 - (0{,}5 + 0{,}5))$$

$$C = \min_0 (1{,}414214 - 1)$$

$$C = \min_0 (0{,}414214)$$

$d_{ij} = \sqrt{(4-4)^2 + (4-3)^2}$ => jarak NPC ke halangan (4,4)

$d_{ij} = \sqrt{0^2 + 1^2}$

$d_{ij} = \sqrt{1}$

$d_{ij} = 1$

rBee = 0,5 ; rObs = 0,5

$C = \min_0(1 - (0{,}5 + 0{,}5))$

$C = \min_0(1 - 1)$

$C = \min_0(0)$

7. Mencari nilai optimal dari jarak NPC awal (4,1) ke target (1,7) dan jarak NPC baru (4,3) ke target (1,7). Jika nilai jarak NPC baru lebih kecil daripada jarak NPC awal, maka NPC pindah ke posisi baru.

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

$d = \sqrt{(1-4)^2 + (7-1)^2}$ => jarak posisi awal NPC ke target

$d = \sqrt{(-3)^2 + 6^2}$

$d = \sqrt{9 + 36}$

$d = \sqrt{45}$

$d = 6{,}708204$

$d = \sqrt{(1-4)^2 + (7-3)^2}$ => jarak posisi baru NPC ke target

$d = \sqrt{(-3)^2 + 4^2}$

$$d = \sqrt{9 + 16}$$

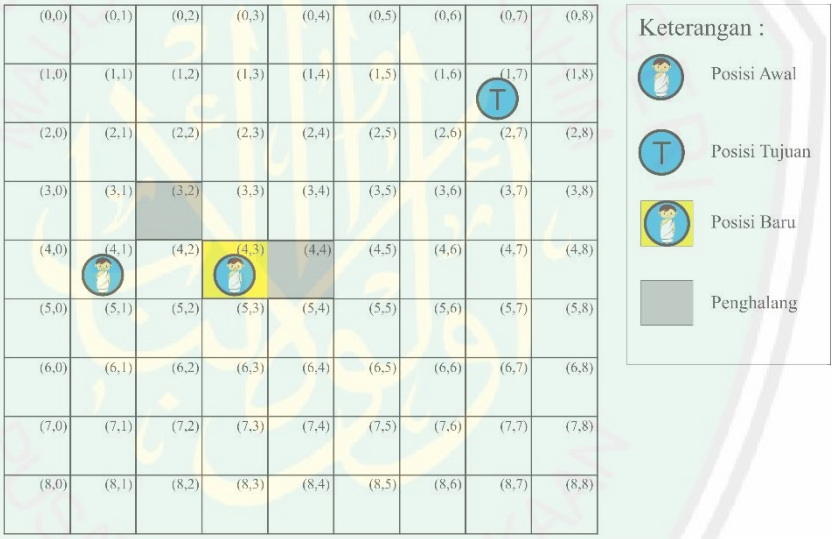
$$d = \sqrt{25}$$

$$d = 5$$

Nilai jarak posisi baru ke target < Nilai jarak posisi awal ke target

5 < 6,708204

Karena nilai jarak NPC baru lebih kecil daripada NPC awal, posisi baru menjadi posisi awal untuk pencarian posisi selanjutnya.



Gambar 3. 16 Iterasi ke-2 pencarian ABC

8. Karena tujuan belum ditemukan, maka pencarian dilanjutkan untuk menghitung pola pencarian NPC posisi berikutnya yang dilakukan secara acak pada ruang lingkup (R) dari posisi awal yang telah diperbarui.

$$X_i' = X_i + rand[-1\ 1] \times R$$

X_i' = 4 + (-1) * 2 = 4 + (-2) = 2 Solusi kemungkinan posisi baru : $X_{ij}' = (2,1)$

X_j' = 3 + (-1) * 2

= 3 + (-2) = 1

9. Dicek pada posisi itu terdapat halangan atau tidak. Jika nilai dari C kurang dari 0, maka NPC diharuskan melakukan pencarian ulang posisi baru. Karena terdapat dua halangan, maka dicek semua posisi baru NPC (2,1) ke halangan (3,2) dan posisi baru NPC (2,1) ke halangan (4,4).

$$C = \min_0 \left(d_{ij} - (r_{bee} + r_{obs})\right)$$

$$d_{ij} = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}$$

$d_{ij} = \sqrt{(3-2)^2 + (2-1)^2}$ => jarak NPC ke halangan (3,2)

$d_{ij} = \sqrt{1^2 + 1^2}$

$d_{ij} = \sqrt{2}$

$d_{ij} = 1{,}414214$

rBee = 0,5 ; rObs = 0,5

$C = \min_0 (1{,}414214 - (0{,}5 + 0{,}5))$

$C = \min_0 (1{,}414214 - 1)$

$C = \min_0 (0{,}414214)$

$d_{ij} = \sqrt{(4-2)^2 + (4-1)^2}$ => jarak NPC ke halangan (4,4)

$d_{ij} = \sqrt{2^2 + 3^2}$

$d_{ij} = \sqrt{13}$

$d_{ij} = 3{,}60555$

rBee = 0,5 ; rObs = 0,5

$C = \min_{0}(3{,}60555 - (0{,}5 + 0{,}5))$

$C = \min_{0}(3{,}60555 - 1)$

$C = \min_{0}(2{,}60555)$

10. Mencari nilai optimal dari jarak NPC awal (4,3) ke target (1,7) dan jarak NPC baru (2,1) ke target (1,7). Jika nilai jarak NPC baru lebih kecil daripada jarak NPC awal, maka NPC pindah ke posisi baru.

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

$d = \sqrt{(1-4)^2 + (7-3)^2}$ => jarak posisi awal NPC ke target

$d = \sqrt{(-3)^2 + 4^2}$

$d = \sqrt{9 + 16}$

$d = \sqrt{25}$

$d = 5$

$d = \sqrt{(1-2)^2 + (7-1)^2}$ => jarak posisi baru NPC ke target

$d = \sqrt{(-1)^2 + 6^2}$

$d = \sqrt{1 + 36}$

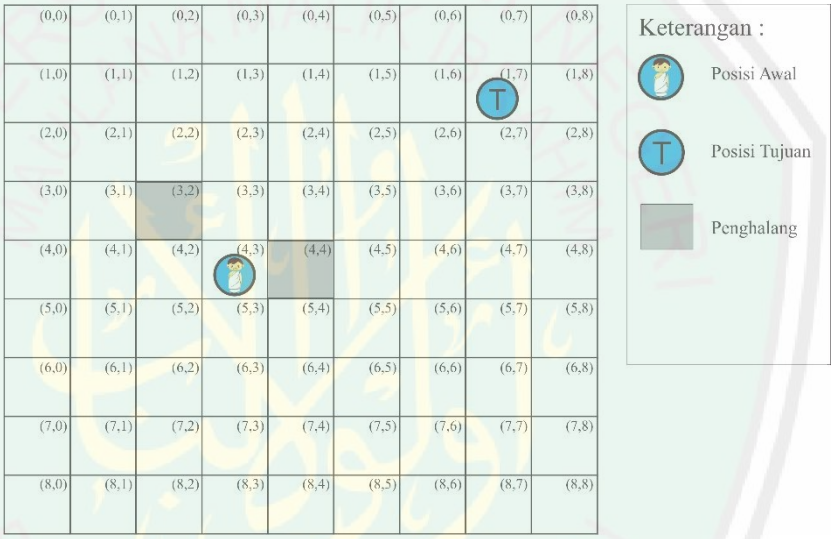
$d = \sqrt{37}$

$d = 6{,}08276$

Nilai jarak posisi baru ke target < Nilai jarak posisi awal ke target

6,08276 > 5

Karena nilai jarak NPC baru lebih besar daripada NPC awal, maka NPC tidak pindah ke posisi baru dan melakukan kembali perhitungan posisi selanjutnya dari posisi awal sebelumnya.



Gambar 3. 17 Iterasi ke-3 pencarian ABC

11. Karena tujuan belum ditemukan, maka pencarian dilanjutkan untuk menghitung pola pencarian NPC posisi berikutnya yang dilakukan secara acak pada ruang lingkup (R) dari posisi awal yang telah diperbarui.

$X_i' = X_i + rand[-1\ 1] \times R$

X_i' $= 4 + (-1) * 2$ Solusi kemungkinan posisi baru :

$= 4 + (-2) = 2$ $X_{ij}' = (2,3)$

X_j' $= 3 + 0 * 2$

$= 3 + 0 = 3$

12. Dicek pada posisi itu terdapat halangan atau tidak. Jika nilai dari C kurang dari 0, maka NPC diharuskan melakukan pencarian ulang posisi baru. Karena terdapat dua halangan, maka dicek semua posisi baru NPC (2,3) ke halangan (3,2) dan posisi baru NPC (2,3) ke halangan (4,4).

$$C = \min_{0}\left(d_{ij} - (r_{bee} + r_{obs})\right)$$

$$d_{ij} = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}$$

$d_{ij} = \sqrt{(3-2)^2 + (2-3)^2}$ => jarak NPC ke halangan (3,2)

$d_{ij} = \sqrt{1^2 + (-1)^2}$

$d_{ij} = \sqrt{2}$

$d_{ij} = 1{,}414214$

rBee = 0,5 ; rObs = 0,5

$C = \min_{0}(1{,}414214 - (0{,}5 + 0{,}5))$

$C = \min_{0}(1{,}414214 - 1)$

$C = \min_{0}(0{,}414214)$

$d_{ij} = \sqrt{(4-2)^2 + (4-3)^2}$ => jarak NPC ke halangan (4,4)

$d_{ij} = \sqrt{2^2 + 1^2}$

$d_{ij} = \sqrt{5}$

$d_{ij} = 2{,}23606$

rBee = 0,5 ; rObs = 0,5

$C = \min_{0}(2{,}23606 - (0{,}5 + 0{,}5))$

$C = \min_{0}(2{,}23606 - 1)$

$C = \min_{0}(1{,}23606)$

13. Mencari nilai optimal dari jarak NPC awal (4,3) ke target (1,7) dan jarak NPC baru (2,3) ke target (1,7). Jika nilai jarak NPC baru lebih kecil daripada jarak NPC awal, maka NPC pindah ke posisi baru.

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

$d = \sqrt{(1-4)^2 + (7-3)^2}$ => jarak posisi awal NPC ke target

$d = \sqrt{(-3)^2 + 4^2}$

$d = \sqrt{9 + 16}$

$d = \sqrt{25}$

$d = 5$

$d = \sqrt{(1-2)^2 + (7-3)^2}$ => jarak posisi baru NPC ke target

$d = \sqrt{(-1)^2 + 4^2}$

$d = \sqrt{1 + 16}$

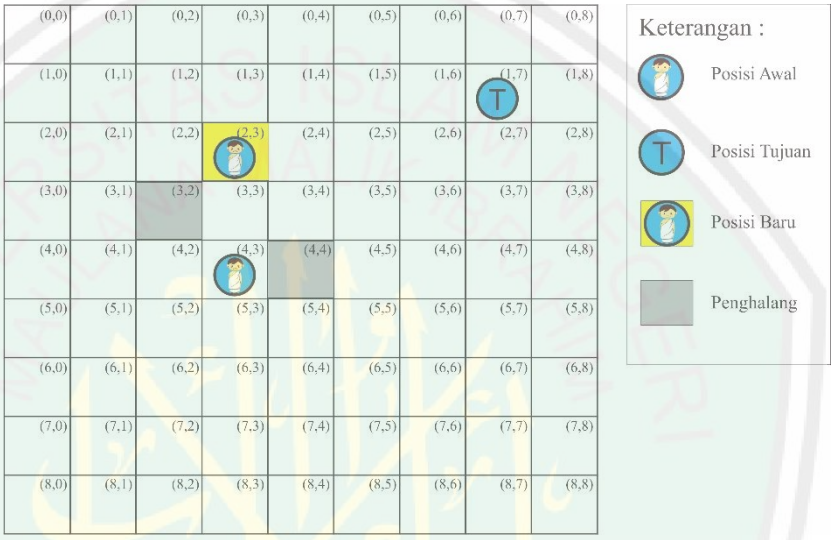
$d = \sqrt{17}$

$d = 4{,}12310$

Nilai jarak posisi baru ke target < Nilai jarak posisi awal ke target

5 < 4,12310

Karena nilai jarak NPC baru lebih kecil daripada NPC awal, posisi baru menjadi posisi awal untuk pencarian posisi selanjutnya.



Gambar 3. 18 Iterasi ke-4 pencarian ABC

14. Karena tujuan belum ditemukan, maka pencarian dilanjutkan untuk menghitung pola pencarian NPC posisi berikutnya yang dilakukan secara acak pada ruang lingkup (R) dari posisi awal yang telah diperbarui.

$$X_i' = X_i + rand[-1\ 1] \times R$$

X_i' $= 2 + 0 * 2$ Solusi kemungkinan posisi baru :

$= 2 + 0 = 2$ $X_{ij}' = (2,5)$

X_j' $= 3 + 1 * 2$

$= 3 + 2 = 5$

15. Dicek pada posisi itu terdapat halangan atau tidak. Jika nilai dari C kurang dari 0, maka NPC diharuskan melakukan pencarian ulang posisi baru. Karena terdapat dua halangan, maka dicek semua posisi baru NPC (2,5) ke halangan (3,2) dan posisi baru NPC (2,5) ke halangan (4,4).

$$C = \min_{0}\left(d_{ij} - (r_{bee} + r_{obs})\right)$$

$$d_{ij} = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}$$

$d_{ij} = \sqrt{(3-2)^2 + (2-5)^2}$ => jarak NPC ke halangan (3,2)

$d_{ij} = \sqrt{1^2 + (-3)^2}$

$d_{ij} = \sqrt{10}$

$d_{ij} = 3{,}162277$

rBee = 0,5 ; rObs = 0,5

$C = \min_{0}(3{,}162277 - (0{,}5 + 0{,}5))$

$C = \min_{0}(3{,}162277 - 1)$

$C = \min_{0}(2{,}162277)$

$d_{ij} = \sqrt{(4-2)^2 + (4-5)^2}$ => jarak NPC ke halangan (4,4)

$d_{ij} = \sqrt{2^2 + (-1)^2}$

$d_{ij} = \sqrt{5}$

$d_{ij} = 2{,}23606$

rBee = 0,5 ; rObs = 0,5

$$C = \min_{0}(2{,}23606 - (0{,}5 + 0{,}5))$$

$$C = \min_{0}(2{,}23606 - 1)$$

$$C = \min_{0}(1{,}23606)$$

16. Mencari nilai optimal dari jarak NPC awal (2,3) ke target (1,7) dan jarak NPC baru (2,5) ke target (1,7). Jika nilai jarak NPC baru lebih kecil daripada jarak NPC awal, maka NPC pindah ke posisi baru.

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

$d = \sqrt{(1-2)^2 + (7-3)^2}$ => jarak posisi awal NPC ke target

$$d = \sqrt{(-1)^2 + 4^2}$$

$$d = \sqrt{1 + 16}$$

$$d = \sqrt{17}$$

$$d = 4{,}12310$$

$d = \sqrt{(1-2)^2 + (7-5)^2}$ => jarak posisi baru NPC ke target

$$d = \sqrt{(-1)^2 + 2^2}$$

$$d = \sqrt{1 + 4}$$

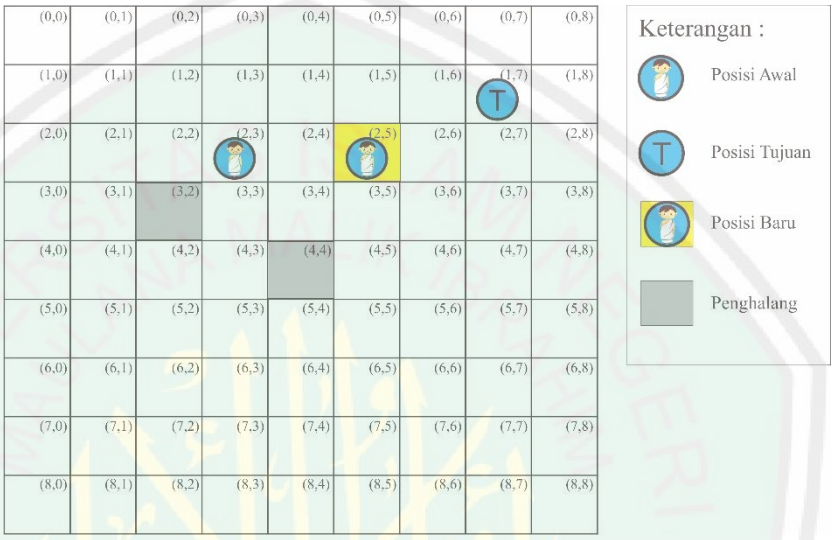
$$d = \sqrt{5}$$

$$d = 2{,}236068$$

Nilai jarak posisi baru ke target < Nilai jarak posisi awal ke target

2,236068 < 4,12310

Karena nilai jarak NPC baru lebih kecil daripada NPC awal, posisi baru menjadi posisi awal untuk pencarian posisi selanjutnya.



Gambar 3. 19 Iterasi ke-5 pencarian ABC

17. Karena tujuan belum ditemukan, maka pencarian dilanjutkan untuk menghitung pola pencarian NPC posisi berikutnya yang dilakukan secara acak pada ruang lingkup (R) dari posisi awal yang telah diperbarui.

$X_i' = X_i + rand[-1\ 1] \times R$

X_i' = 2 + (-0,5) * 2 = 2 + (-1) = 1 Solusi kemungkinan posisi baru : $X_{ij}' = (1,7)$

X_j' = 5 + 1 * 2 = 5 + 2 = 7

18. Dicek pada posisi itu terdapat halangan atau tidak. Jika nilai dari C kurang dari 0, maka NPC diharuskan melakukan pencarian ulang

posisi baru. Karena terdapat dua halangan, maka dicek semua posisi baru NPC (1,7) ke halangan (3,2) dan posisi baru NPC (1,7) ke halangan (4,4).

$$C = \min_{0}\left(d_{ij} - (r_{bee} + r_{obs})\right)$$

$$d_{ij} = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}$$

$d_{ij} = \sqrt{(3-1)^2 + (2-7)^2}$ => jarak NPC ke halangan (3,2)

$d_{ij} = \sqrt{2^2 + (-5)^2}$

$d_{ij} = \sqrt{29}$

$d_{ij} = 5{,}38516$

rBee = 0,5 ; rObs = 0,5

$C = \min_{0}(5{,}38516 - (0{,}5 + 0{,}5))$

$C = \min_{0}(5{,}38516 - 1)$

$C = \min_{0}(4{,}38516)$

$d_{ij} = \sqrt{(4-1)^2 + (4-7)^2}$ => jarak NPC ke halangan (4,4)

$d_{ij} = \sqrt{3^2 + (-3)^2}$

$d_{ij} = \sqrt{18}$

$d_{ij} = 4{,}24264$

rBee = 0,5 ; rObs = 0,5

$$C = \min_{0}(4{,}24264 - (0{,}5 + 0{,}5))$$

$$C = \min_{0}(4{,}24264 - 1)$$

$$C = \min_{0}(3{,}24264)$$

19. Mencari nilai optimal dari jarak NPC awal (2,5) ke target (1,7) dan jarak NPC baru (1,7) ke target (1,7). Jika nilai jarak NPC baru lebih kecil daripada jarak NPC awal, maka NPC pindah ke posisi baru.

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

$d = \sqrt{(1-2)^2 + (7-5)^2}$ => jarak posisi awal NPC ke target

$d = \sqrt{(-1)^2 + 2^2}$

$d = \sqrt{1+4}$

$d = \sqrt{5}$

$d = 2{,}236068$

$d = \sqrt{(1-1)^2 + (7-7)^2}$ => jarak posisi baru NPC ke target

$d = \sqrt{0^2 + 0^2}$

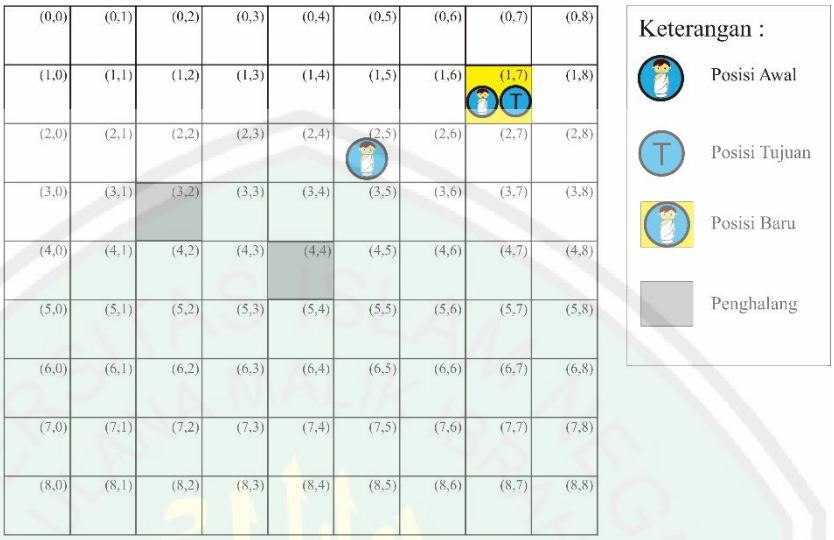
$d = \sqrt{0}$

$\boldsymbol{d = 0}$

Nilai jarak posisi baru ke target < Nilai jarak posisi awal ke target

0 < 2,236068

Karena nilai jarak NPC baru lebih kecil daripada NPC awal, posisi baru menjadi posisi awal untuk pencarian posisi selanjutnya.



Gambar 3. 20 Iterasi ke-6 pencarian ABC

20. Ketika NPC telah sampai pada posisi target (jarak NPC terhadap target = 0), maka perhitungan akan dihentikan. Jika terdapat dua atau lebih target yang berada dalam ruang lingkup (R), maka posisi akhir dari NPC sebagai posisi awal untuk mencari target berikutnya yang telah ditentukan.

BAB IV

IMPLEMENTASI DAN PEMBAHASAN

4.1 Implementasi

Bab ini membahas mengenai implementasi dari perencanaan yang telah diajukan. Selain itu pada bab ini dilakukan pengujian terhadap simulasi untuk mengetahui apakah simulasi tersebut telah berjalan sesuai dengan tujuan penelitian yang ingin dicapai.

4.1.1 Kebutuhan Perangkat Keras

Perangkat keras yang diperlukan untuk mengimplementasikan aplikasi simulasi ini menggunakan Laptop ASUS A455L Processor Intel(R) Core(TM) i5 - 5200U CPU @ 2.20 GHZ, RAM 4.00 GB

4.1.2 Kebutuhan Perangkat Lunak

Perangkat lunak yang diperlukan untuk mengimplementasikan aplikasi simulasi ini, sebagai berikut:

- Sistem Operasi : Windows 8 64 Bit
- *Game Engine* : Unity3D 5.3.5f1
- Konsep desain 2D : Corel Draw X6 dan Adobe Photoshop CS6
- Desain 3D : Blender 2.7
- *Script Writer* : Visual Studio 2015

4.2 Pengujian Algoritma *Boids*

Pengujian *Boids* pada simulasi ini, terdapat beberapa proses yakni *Separation, Alignment, Cohesion,* dan *Obstacle Avoidance*.

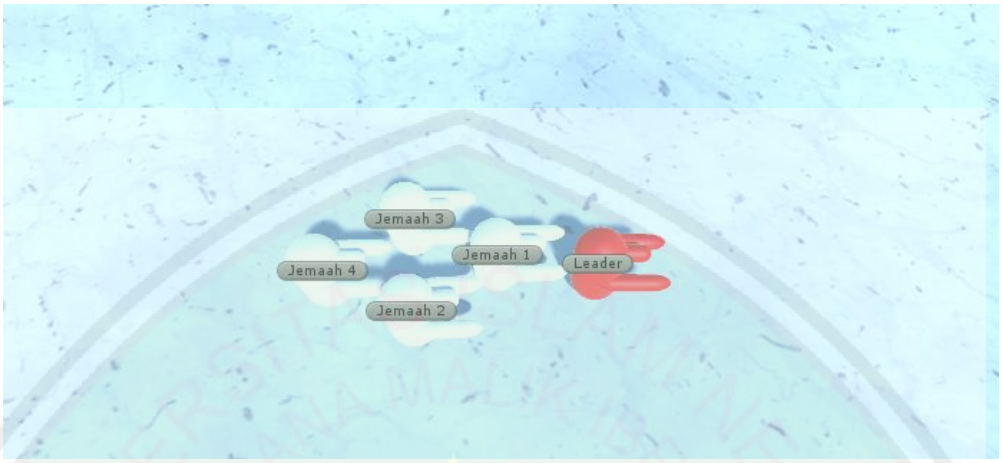
4.2.1 Pengujian *Separation*

Pengujian ini dilakukan untuk melihat bagaimana proses menjaga jarak agar tidak terjadi tabrakan antar jemaah yang dilakukan oleh *boids*. Pengujian ini dilakukan ketika jemaah berada pada titik–titik koordinat yang sudah ditentukan (lihat Gambar 4.1). Setiap jemaah mempunyai kecepatan maksimal bergerak 2.0f untuk mengikuti *leader* yang menuju target, dan setiap jemaah memiliki ambang batas radius 2.0f untuk menjaga terjadinya tabrakan. Lingkungan yang digunakan dalam pengujian ini tanpa penghalang (*obstacle*) dengan menggunakan 5 jemaah dan 1 sebagai *leader*.

Gambar 4. 1 Contoh tata letak awal jemaah

Dengan algoritma *boids* diharapkan jemaah dapat menjaga tabrakan antar jemaah lainnya. Pengujian dilakukan untuk mengecek apakah algoritma telah berfungsi sehingga dapat membuat jemaah dapat menjaga jarak agar tidak terjadi tabrakan. Pengujian dilakukan dengan jumlah iterasi sebanyak 30 kali. Hasilnya

jemaah dapat bergerombol dan saling menjaga jarak antar jemaah (lihat Gambar 4.2).



Gambar 4. 2 Jemaah bergerombol dan menjaga jarak dengan jemaah lainnya

Jemaah dapat bergerombol dan saling menjaga jarak. Dari uji coba didapat jarak jemaah ketika mendekati jemaah lainnya (lihat Tabel 4.1). Jika jarak tersebut bernilai lebih dari 0 semua, maka artinya tidak ada jemaah yang bertabrakan dengan jemaah lainnya. Dari data tersebut diketahui bahwa jarak terkecil dari jemaah untuk menjaga jarak dengan jemaah lainnya adalah 1.96f pada iterasi 26-27. Data "Jemaah 1" diambil sebagai data pengujian *separation*.

Tabel 4. 1 Pengujian *Separation*

Iterasi	Jarak Jemaah 1 Terhadap Jemaah Lainnya			
	Leader	Jemaah 2	Jemaah 3	Jemaah 4
1	13.81	5.39	6	9.22
2	12.94	4.75	5.97	8.74
3	11.92	4.13	5.94	8.28
4	10.88	3.57	5.93	7.89
5	9.87	3.08	5.92	7.58
6	8.89	2.68	5.91	7.33
7	7.87	2.34	5.91	7.13
8	6.87	2.08	5.9	6.98
9	6.1	2.01	5.69	6.69
10	5.35	2	5.53	6.5
11	4.59	2.01	5.66	6.6

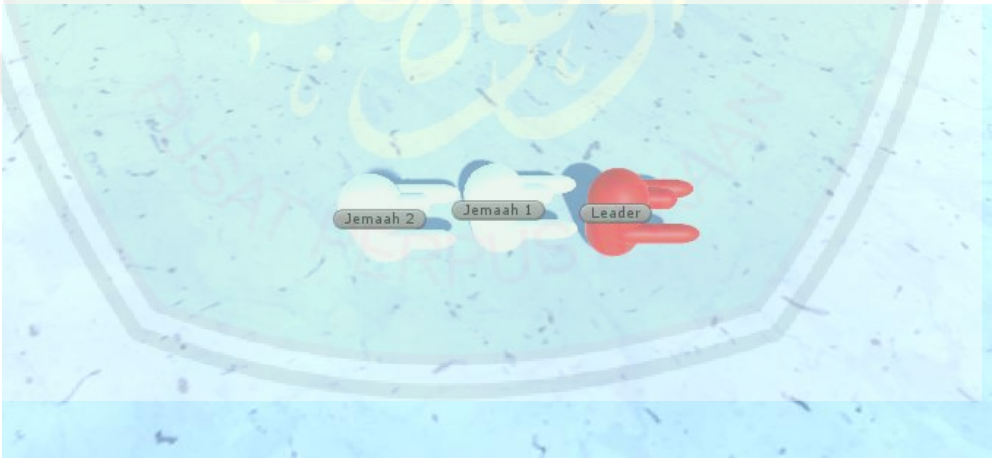
Iterasi	Jarak Jemaah 1 Terhadap Jemaah Lainnya			
	Leader	Jemaah 2	Jemaah 3	Jemaah 4
12	3.84	2	5.8	6.73
13	3.21	1.98	5.86	6.77
14	2.84	2.03	5.59	6.56
15	2.61	2.02	5.28	6.26
16	2.43	2.02	4.87	5.9
17	2.33	2.01	4.42	5.51
18	2.24	2	3.95	5.13
19	2.15	2	3.74	4.87
20	2.04	1.99	3.62	4.64
21	2.01	2	3.48	4.31
22	1.99	1.98	3.34	3.99
23	1.99	2	3.18	3.91
24	2	1.99	3.1	3.94
25	1.98	1.99	3.03	3.9
26	1.96	2	2.9	3.88
27	1.96	1.99	2.74	3.84
28	1.99	2	2.52	3.75
29	2.03	1.99	2.29	3.6
30	2.06	2	2.06	3.54

4.2.2 Pengujian *Alignment*

Pengujian ini dilakukan untuk melihat bagaimana proses menjaga kecepatan jemaah saat bergerombol yang dilakukan oleh *boids*. Pengujian ini dilakukan ketika jemaah berada pada titik–titik koordinat yang sudah ditentukan (lihat Gambar 4.3). Setiap jemaah mempunyai kecepatan maksimal bergerak 2.0f untuk mengikuti *leader* yang menuju target dengan kecepatan *leader* 1.0f, dan setiap jemaah memiliki ambang batas radius 2.0f untuk menyesuaikan kecepatan dengan kelompok. Lingkungan yang digunakan dalam pengujian ini tanpa penghalang (*obstacle*) dengan menggunakan 3 jemaah dan 1 sebagai *leader*.

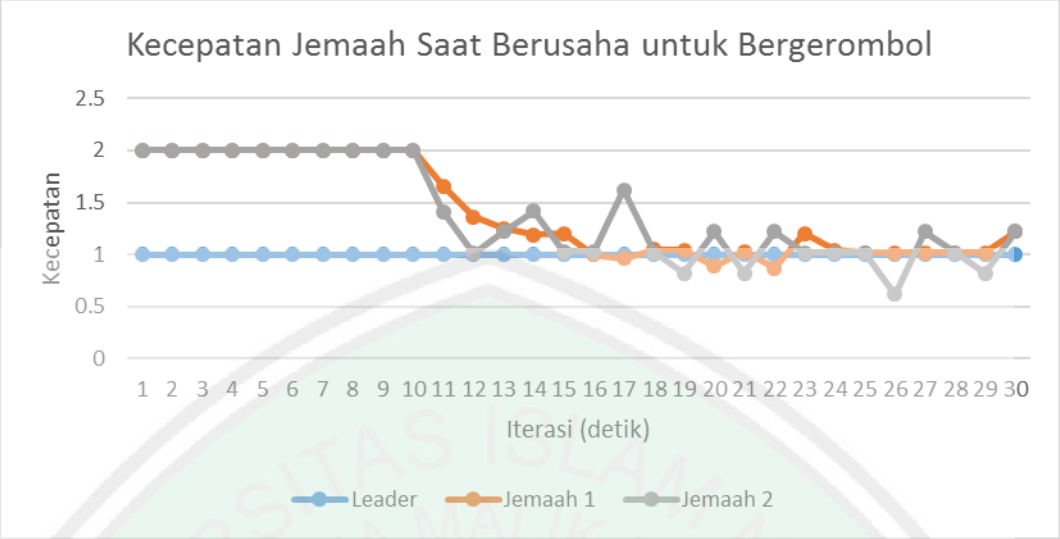
Gambar 4. 3 Contoh tata letak awal jemaah

Dengan algoritma *boids* diharapkan jemaah dapat menyesuaikan kecepatan dengan kelompok. Pengujian dilakukan untuk mengecek apakah algoritma telah berfungsi sehingga dapat membuat jemaah dapat menyesuaikan kecepatan dengan kelompok. Pengujian dilakukan dengan jumlah iterasi sebanyak 30 kali. Hasilnya jemaah dapat bergerombol dan saling menyesuaikan kecepatan dengan kelompok (lihat Gambar 4.4).



Gambar 4. 4 Jemaah menyesuaikan kecepatan dengan jemaah lainnya

Jemaah dapat bergerombol dan berusaha menyesuaikan kecepatan dengan kelompoknya. Dari uji coba didapat kecepatan jemaah ketika berusaha untuk bergerombol saat mengikuti *leader* (lihat Gambar 4.5).



Gambar 4. 5 Grafik kecepatan jemaah saat berusaha untuk bergerombol

Jika dilihat dari grafik kecepatan jemaah saat berusaha untuk bergerombol mengikuti *leader* (Gambar 4.5). Pada grafik terlihat bahwa jemaah pada detik awal menggunakan kecepatan maksimal, karena jemaah mencoba untuk bergerombol mendekati *leader*. Tetapi pada detik ke-11 terjadi penurunan kecepatan. Dikarenakan jemaah berusaha menyesuaikan kecepatan dengan kelompok saat bergerombol.

4.2.3 Pengujian *Cohesion*

Pengujian ini dilakukan untuk menjaga jemaah tetap dekat dengan jemaah lainnya yang dilakukan oleh *boids*. Pengujian ini dilakukan ketika jemaah berada pada titik–titik koordinat yang sudah ditentukan (lihat Gambar 4.6). Setiap jemaah mempunyai kecepatan maksimal bergerak 2.0f untuk mengikuti *leader*, dan setiap jemaah memiliki ambang batas radius 2.0f untuk mengarahkan jemaah ke arah jemaah lainnya. Lingkungan yang digunakan dalam pengujian ini tanpa penghalang (*obstacle*) dengan menggunakan 5 jemaah dan 1 sebagai *leader*.

Gambar 4. 6 Contoh tata letak awal jemaah

Dengan algoritma *boids* diharapkan jemaah dapat menjaga jarak agar jemaah tetap dekat dengan jemaah lainnya. Pengujian dilakukan untuk mengecek apakah algoritma telah berfungsi (lihat Gambar 4.7).

Gambar 4. 7 Jemaah menjaga jarak tetap dekat dengan jemaah lainnya

Jemaah dapat bergerombol dan saat jemaah mendekati jemaah lainnya, jemaah akan mencoba untuk tetap dekat dengan jamaah lainnya. Dari uji coba didapat jarak jemaah ketika mendekati jemaah lainnya (lihat Tabel 4.2). Dari data tersebut diketahui bahwa jarak terkecil dari jemaah untuk mencoba tetap dekat dengan jemaah lainnya adalah 1.96f.

Tabel 4. 2 Pengujian *Cohesion*

Iterasi	Jarak Jemaah 1 Terhadap Jemaah Lainnya			
	Leader	Jemaah 2	Jemaah 3	Jemaah 4
1	13.81	5.39	6	9.22
2	12.94	4.75	5.97	8.74
3	11.92	4.13	5.94	8.28
4	10.88	3.57	5.93	7.89
5	9.87	3.08	5.92	7.58
6	8.89	2.68	5.91	7.33
7	7.87	2.34	5.91	7.13
8	6.87	2.08	5.9	6.98
9	6.1	2.01	5.69	6.69
10	5.35	2	5.53	6.5
11	4.59	2.01	5.66	6.6
12	3.84	2	5.8	6.73
13	3.21	1.98	5.86	6.77
14	2.84	2.03	5.59	6.56
15	2.61	2.02	5.28	6.26
16	2.43	2.02	4.87	5.9
17	2.33	2.01	4.42	5.51
18	2.24	2	3.95	5.13
19	2.15	2	3.74	4.87
20	2.04	1.99	3.62	4.64
21	2.01	2	3.48	4.31
22	1.99	1.98	3.34	3.99
23	1.99	2	3.18	3.91
24	2	1.99	3.1	3.94
25	1.98	1.99	3.03	3.9
26	1.96	2	2.9	3.88
27	1.96	1.99	2.74	3.84
28	1.99	2	2.52	3.75
29	2.03	1.99	2.29	3.6
30	2.06	2	2.06	3.54

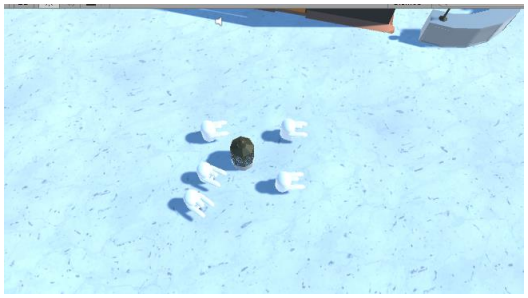
4.2.4 *Obstacle Avoidance*

Pengujian ini dilakukan untuk melihat bagaimana proses menghindari *obstacle* yang dilakukan oleh *boids*. Pengujian ini dilakukan ketika jemaah berada pada titik–titik koordinat yang sudah ditentukan (lihat Gambar 4.8). Setiap jemaah mempunyai kecepatan maksimal berjalan 2.0f dan setiap jemaah memiliki ambang batas radius 3.0f untuk mendeteksi *obstacle*. Lingkungan yang digunakan dalam pengujian ini diberi halangan (*obstacle*) dengan menggunakan 5 jemaah.

Gambar 4. 8 Contoh tata letak awal jemaah

Pengujian dilakukan untuk mengecek apakah algoritma *boids* telah berfungsi sehingga dapat membuat jemaah dapat bergerak menghindari *obstacle* yang berada didepannya. Pengujian dilakukan dengan jumlah iterasi sebanyak 20 kali. Hasilnya jemaah dapat bergerak menjauh dari *obstacle* dan bergerombol kembali untuk setiap iterasi menuju target (lihat Gambar 4.9).

(a)



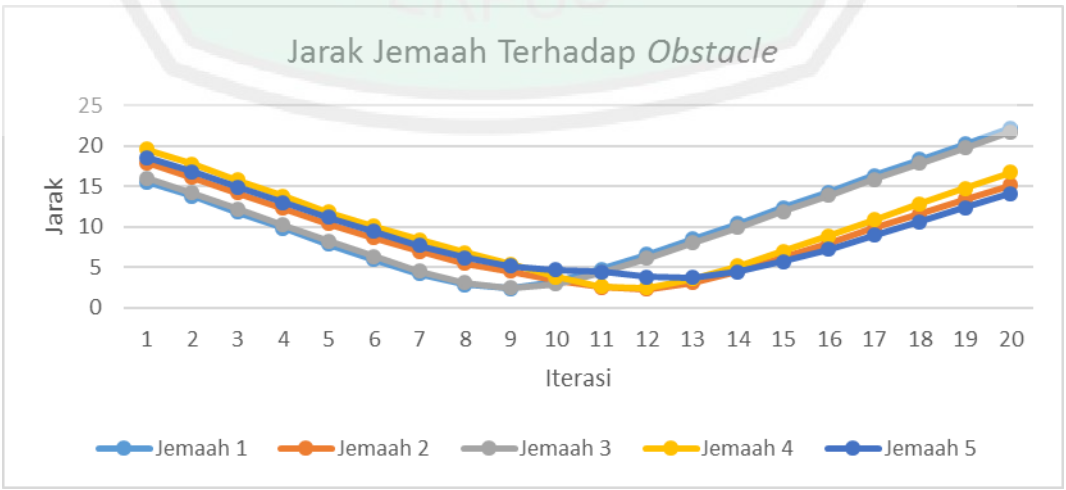
(b)

(c)

Gambar 4. 9 Uji coba pada iterasi ke-1(a), ke-9(b), dan ke-20(c).

Untuk mengetahui apakah jemaah bertabrakan dengan *obstacle*, kita gunakan jarak *euclide* untuk setiap jemaah terhadap *obstacle*. Jika jarak tersebut bernilai lebih dari 0 semua, maka artinya tidak ada jemaah yang bertabrakan dengan *obstacle*.

Jika dilihat dari grafik jarak *obstacle* (lihat Gambar 4.10), berdasarkan tata letak posisi awal jemaah (lihat Gambar 4.8), maka jemaah ketika menuju target, akan bertemu sebuah *obstacle*. Sehingga pada grafik jarak terlihat jarak jemaah yang semula jauh dari *obstacle* mendekati *obstacle* tetapi masih menjaga jarak, kemudian jemaah setelah aman dari *obstacle*, maka akan melanjutkan perjalanan menuju target. Jarak ke *obstacle* setelah jemaah bebas pun semakin lama semakin naik menjauh dari *obstacle*.



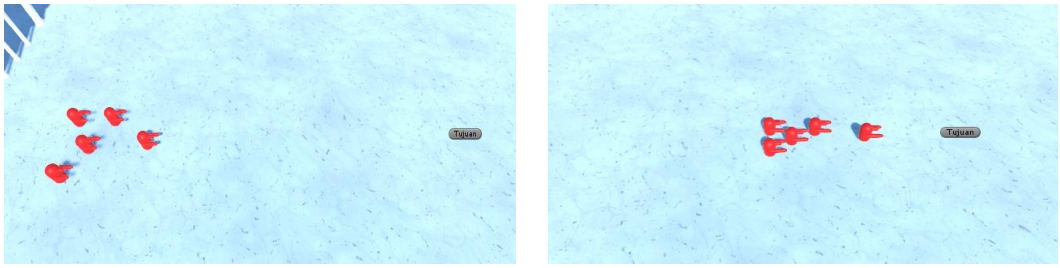
Gambar 4. 10 Grafik jarak jemaah terhadap *obstacle*

4.3 Pengujian Algoritma *Bee Colony*

Pengujian ini dilakukan untuk melihat bagaimana proses menuju target yang dilakukan oleh *Bee Colony*. Pengujian ini dilakukan menggunakan 5 jemaah dan posisi awal jemaah berada pada titik–titik koordinat yang sudah ditentukan (lihat Gambar 4.11). Setiap jemaah mempunyai kecepatan maksimal bergerak 1.0f. Posisi pengacakan awal ini didisarkan pada fase *scouting bee colony*. Dimana jemaah dianggap menyebar untuk menuju target.

Gambar 4. 11 Contoh tata letak awal jemaah dan target

Dengan algoritma *Bee Colony* diharapkan jemaah tersebut dapat bergerak bergerombol menuju target yang telah ditentapkan. Pengujian ini dilakukan untuk mengecek apakah algoritma telah berfungsi sehingga dapat membuat jemaah dapat bergerak seperti lebah menuju sebuah target. Pengujian dilakukan dengan jumlah iterasi sebanyak 35 kali. Hasilnya jemaah dapat bergerombol untuk setiap iterasi menuju target (lihat Gambar 4.12).

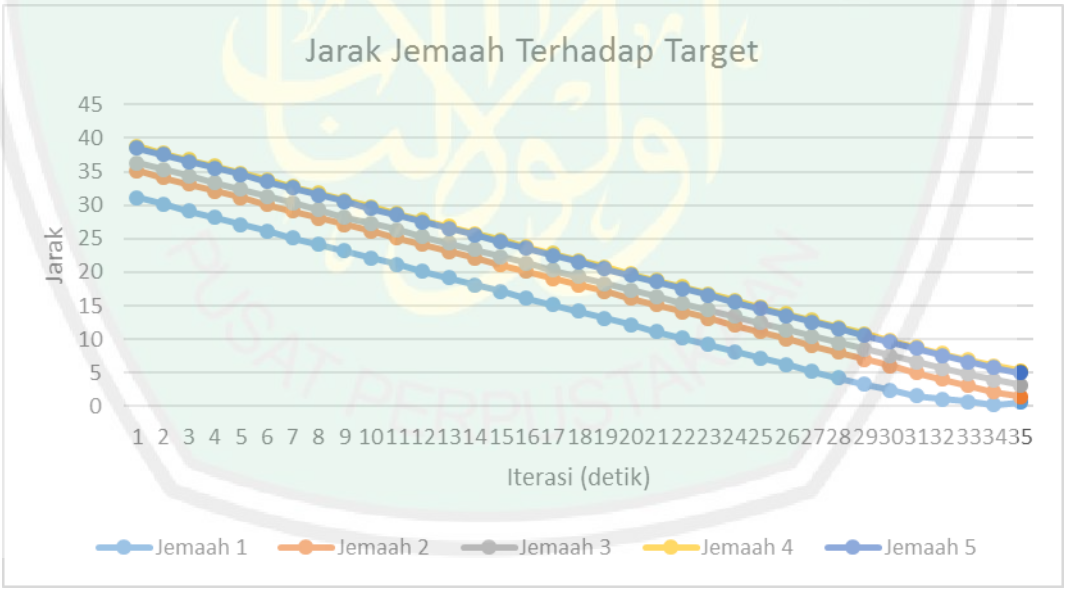


(a) (b)

(c)

Gambar 4. 12 Uji coba pada iterasi ke-1(a), ke-23(b), dan ke-35(c).

Jemaah dapat bergerombol layaknya lebah dan jemaah dapat mencapai target yang dituju. Dari uji coba didapatkan jarak terhadap target yang terus menurun setiap dilakukan iterasi seperti yang terpampang pada grafik (lihat Gambar 4.13). Hal ini menunjukkan bahwa untuk setiap iterasinya, jemaah telah berpindah menuju posisi yang lebih optimal dari posisi sebelumnya.



Gambar 4. 13 Grafik jarak jemaah terhadap target

4.4 Pengujian Algoritma *Boids* dan *Bee Colony* pada Simulasi

Pengujian ini dilakukan untuk melihat apakah metode *Boids* dan *Bee Colony* dapat bekerja dengan baik pada simulasi. Pengujian ini dilakukan ketika jemaah

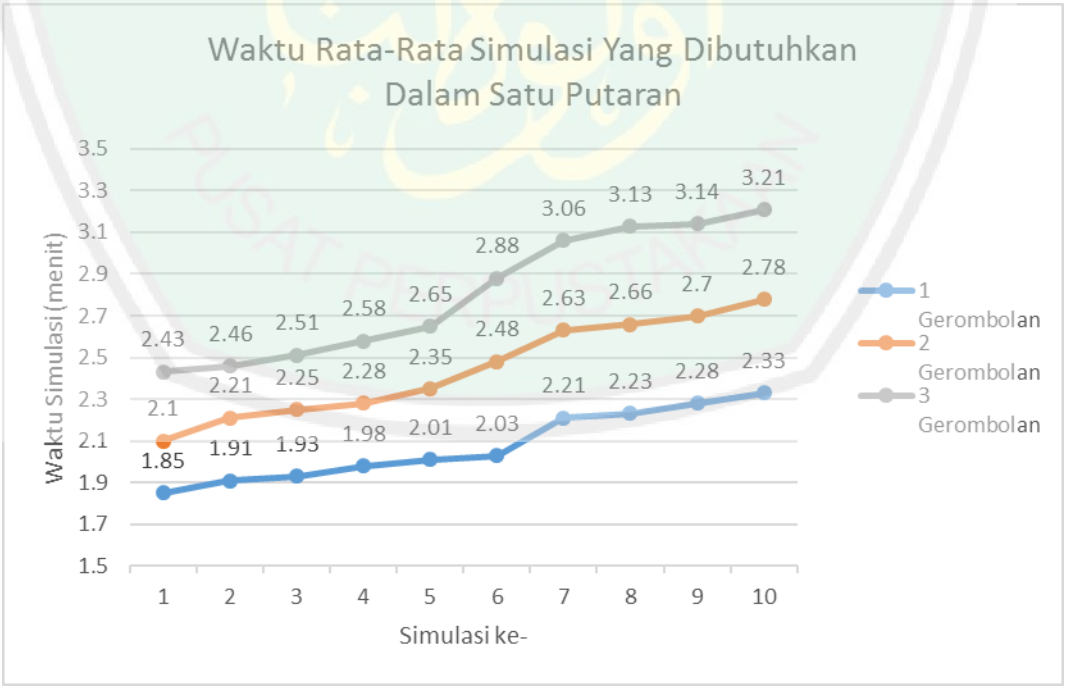
berada pada titik–titik koordinat yang sudah ditentukan. Dalam pengujian dilakukan 10 kali simulasi dengan 3 kali uji coba pada setiap simulasinya. Pada setiap uji coba dibedakan dengan jumlah gerombolan dan jumlah jemaahnya. Dari uji coba didapat waktu rata-rata yang diperlukan jemaah untuk melakukan 1 putaran thawaf dan presentasi jumlah jemaah yang berhasil melakukan putaran thawaf (lihat Tabel 4.3). Dari data menunjukkan bahwa pada setiap simulasi dan uji coba presentasi jemaah yang berhasil mengelilingi Ka'bah hasilnya 100%.

Tabel 4. 3 Pengujian Algoritma *Boids* dan ABC pada Simulasi

Simulasi ke-	Uji coba ke-	Jumlah Gerombolan	Jumlah Jemaah	Jumlah jemaah yang berhasil mengelilingi Ka'bah	Presentasi jemaah yang berhasil mengelilingi Ka'bah	Waktu rata-rata Simulasi (menit)
1	1	1	5	5	100	1.85
	2	2	10	10	100	2.1
	3	3	15	15	100	2.43
2	1	1	10	10	100	1.91
	2	2	20	20	100	2.21
	3	3	30	30	100	2.46
3	1	1	15	15	100	1.93
	2	2	30	30	100	2.25
	3	3	45	45	100	2.51
4	1	1	20	20	100	1.98
	2	2	40	40	100	2.28
	3	3	60	60	100	2.58
5	1	1	25	25	100	2.01
	2	2	50	50	100	2.35
	3	3	75	75	100	2.65
6	1	1	30	30	100	2.03
	2	2	60	60	100	2.48
	3	3	90	90	100	2.88
7	1	1	35	35	100	2.21
	2	2	70	70	100	2.63
	3	3	105	105	100	3.06
8	1	1	40	40	100	2.23
	2	2	80	80	100	2.66
	3	3	120	120	100	3.13

Simulasi ke-	Uji coba ke-	Jumlah Gerombolan	Jumlah Jemaah	Jumlah jemaah yang berhasil mengelilingi Ka'bah	Presentasi jemaah yang berhasil mengelilingi Ka'bah	Waktu rata-rata Simulasi (menit)
9	1	1	45	45	100	2.28
	2	2	90	90	100	2.7
	3	3	135	135	100	3.14
10	1	1	50	50	100	2.33
	2	2	100	100	100	2.78
	3	3	150	150	100	3.21

Jika dilihat dari grafik waktu rata-rata yang diperlukan jemaah dalam melakukan 1 putaran thawaf dengan jumlah gerombolan dan jumlah jemaah yang berbeda (lihat Gambar 4.14). Pada grafik terlihat meningkatnya waktu yang diperlukan jemaah dalam melakukan 1 putaran thawaf dengan meningkatnya jumlah jemaah dan rata-rata peningkatan waktu sebesar 32.09%.



Gambar 4. 14 Waktu rata-rata simulasi dalam satu putaran

4.5 Implementasi Simulasi

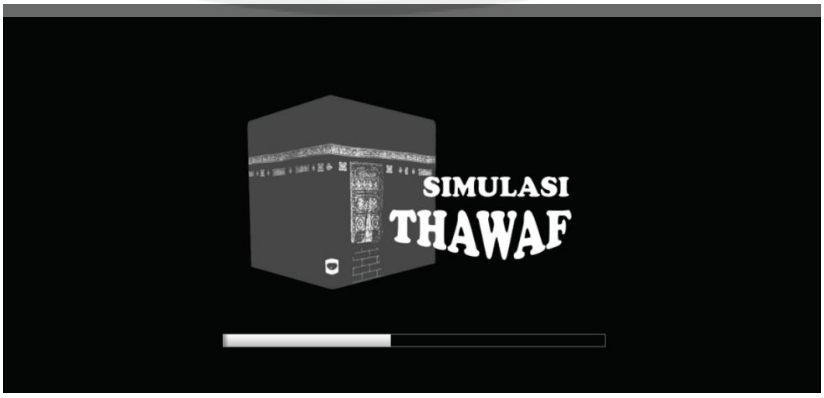
Implementasi merupakan proses pembangunan komponen-komponen pokok suatu sistem. Komponen tersebut dibangun berdasarkan desain dan rancangan yang telah dibuat sebelumnya.

4.5.1 Tampilan Menu Awal

Gambar 4. 15 Tampilan Menu Awal

Tampilan menu awal merupakan tampilan yang pertama kali akan muncul ketika aplikasi simulasi dijalankan. Menu simulasi dirancang menampilkan dua pilihan yaitu *play* : untuk memainkan simulasi, dan *exit* : memberikan pilihan untuk keluar dari simulasi.

4.5.2 Tampilan *Splashscreen*



Gambar 4. 16 Tampilan *Splashscreen*

Tampilan *splashscreen* merupakan tampilan yang akan muncul setelah tombol *play* pada menu awal diklik. *Splashscreen* ini bertujuan untuk memberi jeda waktu sebelum simulasi berjalan. *Splashscreen* akan muncul sekitar 3 detik.

4.5.3 Tampilan Simulasi pada Bagian Awal

Gambar 4. 17 Tampilan Simulasi pada Bagian Awal

Tampilan simulasi pada bagian awal merupakan tampilan awal ketika simulasi ini berjalan. Jemaah tersebar pada posisi yang telah ditentukan di sekitar area Masjidil Haram. Jemaah akan menuju Ka'bah untuk melekukan putaran thawaf.

4.5.4 Tampilan Simulasi Saat Sejajar dengan Hajar Aswad

Gambar 4. 18 Tampilan Simulasi Saat Sejajar dengan Hajar Aswad

Tampilan simulasi saat sejajar dengan Hajar Aswad merupakan tampilan ketika jemaah berada sejajar dengan Hajar Aswad. Pada simulasi ini akan muncul konten informasi mengenai apa yang harus dilakukan dan dibaca ketika berada sejajar dengan Hajar Aswad.

4.5.5 Tampilan Simulasi Saat Melakukan Putaran Thawaf

Gambar 4. 19 Tampilan Simulasi Saat Melakukan Putaran Thawaf

Tampilan simulasi saat melakukan putaran thawaf merupakan tampilan ketika jemaah sudah melewati Hajar Aswad. Pada tampilan ini akan muncul konten yang berisi doa yang harus dibaca ketika melakukan putaran thawaf, dibaca mulai dari Hajar Aswad sampai Rukun Yamani. Putaran pertama sampai putaran ketujuh memiliki bacaan doa yang berbeda. Konten tersebut bisa dihilangkan dengan meng-klik tombol “Doa”.

4.5.6 Tampilan Simulasi Saat Sejajar dengan Rukun Yamani

Gambar 4. 20 Tampilan Simulasi Saat Sejajar dengan Rukun Yamani

Tampilan simulasi saat sejajar dengan Rukun Yamani merupakan tampilan ketika jemaah berada sejajar dengan Rukun Yamani. Pada simulasi ini akan muncul konten informasi mengenai apa yang harus dilakukan dan dibaca ketika berada sejajar dengan Rukun Yamani.

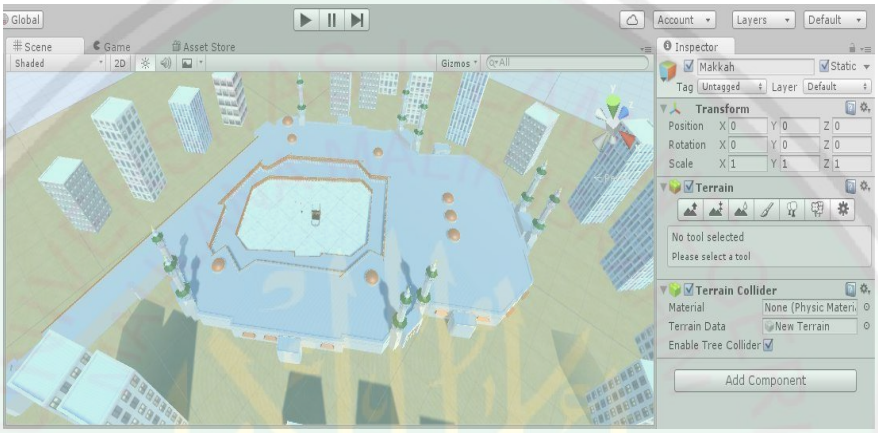
4.5.7 Tampilan Simulasi Saat Berjalan dari Rukun Yamani ke Hajar Aswad

Gambar 4. 21 Tampilan Simulasi Saat Berjalan dari Rukun Yamani ke Hajar Aswad

Tampilan simulasi saat berjalan dari Rukun Yamani ke Hajar Aswad merupakan tampilan ketika jemaah berjalan di antara Rukun Yamani dan Hajar

Aswad. Pada tampilan ini akan muncul konten yang berisi doa yang harus dibaca saat berjalan di antara Rukun Yamani dan Hajar Aswad. Konten tersebut bisa dihilangkan dengan meng-klik tombol “Doa”.

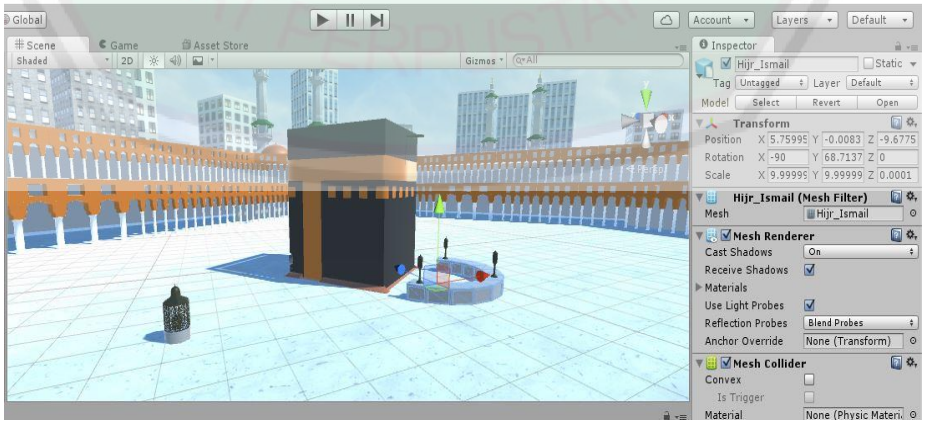
4.5.8 Pembangunan *Enviroment*



Gambar 4. 22 Tampilan Pembangunan Enviroment

Pembangunan *Enviroment* merupakan tampilan *editor unity* saat membuat *enviroment* Masjidil Haram.

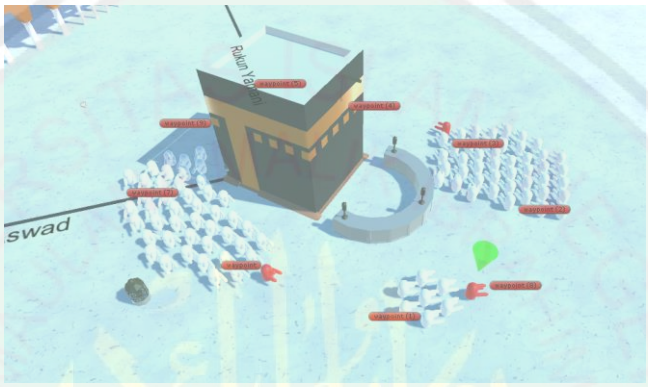
4.5.9 Hambatan Statis padas Sekitar Area Ka’bah



Gambar 4. 23 Hambatan Statis padas Sekitar Area Ka’bah (Maqam Ibrahim dan Hijr Ismail)

Hambatan statis pada sekitar area Ka'bah merupakan hambatan yang harus dihindari jemaah saat melewatinya. Hambatan–hambatan tersebut, yaitu Hijr Ismail dan Maqam Ibrahim.

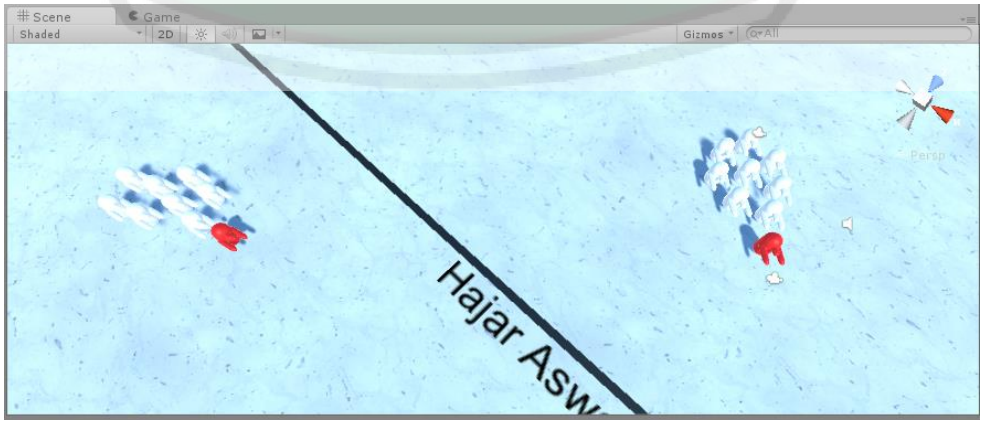
4.5.10 Lintasan Jemaah saat Mengelilingi Ka'bah



Gambar 4. 24 Lintasan Jemaah saat Mengelilingi Ka'bah

Lintasan Jemaah saat mengelilingi Ka'bah merupakan lintasan yang telah ditentukan dengan titik–titik *waypoint* yang tersebar disekeliling Ka'bah. Pada setiap jemaah terdapat *leader*. *Leader* memiliki sifat mencari titik–titik *waypoint* tersebut dengan menggunakan algoritma *Bee Colony*.

4.5.11 Jemaah



Gambar 4. 25 Jemaah

Jemaah adalah NPC yang berkumpul menjadi kerumunan/gerombolan. Pada setiap gerombolan terdapat *leader* (berwarna merah) dan jemaah lainnya (berwarna putih). Jemaah lainnya memiliki beberapa sifat, yaitu menjaga terjadinya tabrakan antar jemaah lainnya, menjaga kecepatan dengan kelompok, menjaga tetap bersama dengan kelompok, menghindari hambatan, dan mengikuti *leader*. Sedangkan *leader* memiliki sifat untuk mencari target disekeliling Ka'bah agar jemaah dapat mengitari Ka'bah.

4.6 Integrasi Penelitian dengan Islam

Islam memandang, menyampaikan ilmu adalah sesuatu yang harus dilakukan bagi seluruh umat islam. Seperti firman Allah dibawah ini:

وَلْتَكُن مِّنكُمْ أُمَّةٌ يَدْعُونَ إِلَى الْخَيْرِ وَيَأْمُرُونَ بِالْمَعْرُوفِ وَيَنْهَوْنَ عَنِ الْمُنكَرِ وَأُوْلَئِكَ هُمُ الْمُفْلِحُونَ ﴿١٠٤﴾

Yang artinya: "*Dan hendaklah ada di antara kamu segolongan umat yang menyeru kepada kebajikan, menyuruh kepada yang ma`ruf dan mencegah dari yang munkar; merekalah orang-orang yang beruntung.*" (Qs. Ali-'Imraan : 104)

Berdasarkan ayat Al-Qur'an diatas, yaitu tentang kewajiban bagi setiap orang yang berilmu haruslah menyampaikan ilmunya itu kepada setiap orang yang membutuhkan, agar orang tersebut tidak tersesat.

Bersabda Rasulullah SAW :"Sedekah yang paling utama adalah orang islam yang belajar suatu ilmu kemudian ia ajarkan ilmu itu kepada saudaranya muslim" (H.R; Ibnu Majah)

Mengajarkan ilmu dianggap sebagai sedekah, karena mengajarkan suatu ilmu khususnya ilmu agama berarti menanam amal *muta'addi* (dapat

berkembang) dimana kemanfaatannya bukan hanya dikenyam oleh orang yang diajar itu sendiri, melainkan dapat dinikmati oleh orang lain.

Dari penjelasan dan penjabaran diatas, menjadi langkah awal penelitian ini yang mempunyai konten tata cara thawaf yang benar. Kemudian ketika *user* melihat simulasi ini, maka secara otomatis akan menambah ilmu atau wawasa pada *user* tentang tata cara thawaf.

Banyak orang yang baru belajar/mengetahui tata cara thawaf hanya sesuai dari yang mereka baca tanpa mengetahui dengan melihat secara visualnya apa yang harus mereka lakukan dan mereka baca(doa) agar mereka tidak melakukan kesalahan. Padahal banyak dalil yang menyebutkan bahwa beramal tanpa ilmu adalah sia-sia, salah satu dalilnya adalah sebagai berikut :

وَلاَ تَقْفُ مَا لَيْسَ لَكَ بِهِ عِلْمٌ إِنَّ السَّمْعَ وَالْبَصَرَ وَالْفُؤَادَ كُلُّ أُولئِكَ كَانَ عَنْهُ مَسْؤُولاً ﴿٣٦﴾

Artinya: "*Dan janganlah kamu mengikuti apa yang kamu tidak mempunyai pengetahuan tentangnya. Sesungguhnya pendengaran, penglihatan dan hati, semuanya itu akan diminta pertanggungan jawabnya..*" (Qs. Al-Israa': 36)

Penjelasan diatas cukuplah menjadi alasan penulis melakukan penelitian dalam pembuatan media simulasi tentang tata cara thawaf untuk membantu memberikan pengetahuan dasar tentang hal–hal yang berhubungan dengan thawaf.

BAB V

PENUTUP

5.1 Kesimpulan

Dari penelitian yang dilakukan dengan melakukan simulasi kerumunan untuk pergerakan jemaah yang sedang melakukan Thawaf, dapat disimpulkan bahwa dengan menggunakan metode *Boids* dan *Artificial Bee Colony* untuk NPC jemaah agar berperilaku dinamis yakni mempunyai perilaku yang fleksibel sesuai dengan keadaan lingkungan.

Dari pengujian telah menunjukan hasil sebesar 100% terhadap pengujian 150 jemaah dalam keadaan bergerombol untuk mengelilingi Ka'bah (Thawaf). Sedangkan hasil untuk rata-rata waktu yang diperlukan dalam melakukan satu putaran simulasi akan meningkat dengan semakin meningkatnya jumlah jemaah. Dengan jumlah awal 15 jemaah kemudian ditingkatkan menjadi 150 jemaah, maka waktu yang diperlukan mengalami peningkatan sebesar 32.09%.

5.2 Saran

Dalam simulasi ini tentu masih banyak kekurangan dalam penelitian dan pembuatan. Oleh karena itu, penulis menyarankan nantinya perlu untuk dilakukan pengembangan, diantaranya:

1. Penyempurnaan karakter jemaah dan *visualisasi* dalam simulasi perlu dikembangkan agar terlihat lebih nyata.
2. Simulasi ini tidak hanya disajikan berbasis *desktop* saja, namun juga bisa dikembangkan pada *platform* android agar lebih mudah dinikmati.

3. Simulasi ini bisa dikembangkan untuk permasalahan simulasi kerumunan jemaah dengan menerapkan pada *enviroment* lebih dari satu lantai dengan jumlah jemaah yang lebih banyak lagi, sehingga simulasi dapat lebih mewakili keadaan sebenarnya didalam dunia nyata.

DAFTAR PUSTAKA

Al-Qur'an dan Terjemahannya. (2003). Semarang: Toha Putera.

Alam, Muhammad Zidny. 2013. *Optimasi Rute Kendaraan Dengan Kapasitas Mengunakan Modifikasi Algoritma Artificial Bee Colony*. Skripsi. Jakarta : Program Studi Matematika, UIN Syarif Hidayatullah.

Arif, Yunifa Miftachul. 2010. *Strategi Menyerang pada Game FPS Menggunakan Hierarchy Finite State Machine dan Logika Fuzzy*. Thesis. Surabaya: Pasca Sarjana Teknik Elektro ITS.

Benufinit, Yonly Andrianus., Hariadi, Moch., Mardi S.N, Supeno. 2014. *Manuver Kelompok NPC Berbasis Boids Pengembangan Game Real Time Strategy*. Jurusan Teknik Elektro, Fakultas Teknik Industri Institut Teknologi Sepuluh Nopember: Surabaya.

Dewi, Meilany., Hariadi, Moch., Purnomo, Mauridhi Hery. 2011. *Simulating The Movement Of The Crowd In An Enviroment Using Flocking*. Jurusan Teknik Elektro, Institut Teknologi Sepuluh November: Surabaya.

Fadila, Juniardi Nur., Yuniarno, Eko Mulyanto., Nugroho, Supeno Mardi Susiki., Nugroho, Fresy. 2016. *NPCs Multi Enemy Attack Formation Using Bee Colony Algorithm*. Jurnal IRECOS. Vol. 11, No. 7.

Jatiningsih, Wilujeng., Yuniarno, Eko Mulyanto., Hariadi, Mochamad. 2014 *Autonomous Agent Based NPC Swarm Attack Behavior Using Bee Colony*

Algorithm. Proceedings of the Seminar on Intelligent Technology and Its Applications (ISITIA), May 22th.

JimHyuk Hong. 2005. *Evolving Reactive NPCs for the Real-Time Simulation Game. CIG*

Karaboga, D. *Artificial Bee Colony Algorithm. Scholarpedia 2010,5,6915*. Availableonline: http://www.scholarpedia.org/article/Artificial_bee_colony_algorithm/. (di akses tanggal 01 Mei 2016).

Kementrian Agama. 2014. *Tuntunan Manasik Haji dan Umrah*. Direktorat Jenderal penyelenggaraan Haji dan Umrah Jakarta.

_____. 2014. *Do'a dan Dzikir Manasik Haji dan Umrah*. Direktorat Jenderal penyelenggaraan Haji dan Umrah Jakarta.

Mudhana, I Made Pasek., Purnomo, Mauridhi Hery., Mugroho, Supeno Mardi Susiki. 2014. *Simulasi Pergerakan Evakuasi Bencana Tsunami Menggunakan Algoritma Boids dan Panthfinding*. Jurusan Teknik Elektro, Institut Teknologi Sepuluh November : Surabaya.

Mu'min, Syahri., Hariadi, Mochammad., Nugroho, Supeno Mardi Susiki. 2015. *Pergerakan Otonom Pasukan Berbasis Algoritma Boids Menggunakan Metode Particle Swarm Optimazition*. Journal of Animation and Games Studies, Vol. 1 No. 1 – April 2015 ISSN 2460-5662.

Maulana, S, N. 2010. *Penggunaan Struktur Data Quad-Tree dalam Algoritma Collision Detection pada Vertical Shooter Game*. Makalah IF3051 Strategi Algoritma Sem.I Tahun 2010/2011. Institut Teknologi Bandung. Bandung.

Nurdiana, Dian. 2015. *Implementasi Algoritma Lebah Untuk Pencarian Jalur Terpendek Dengan Mempertimbangkan Heuristik*. Jurnal Pendidikan Matematika Volume 5, Nomor 2, April 2015.

Nugroho, Rakhmad Fajar dan Ayub, Mewati. 2013. *Penerapan Algoritma Artificial Bee Colony dalam Aplikasi Penjadwalan Pelajaran untuk Sekolah Menengah Pertama*. Jurnal Informatika, Vol. 9 No. 1, Juni 2013: 1 – 17.

Nugroho, Supeno Mardi Susiki dkk. 2010. *Perilaku Agen Cerdas Berbasis BOIDS Untuk Simulasi Kerumunan Pada Keadaan Bahaya*. Journal of Electrical and Electronics Engineering, Vol. 8, No.1, Apr . 2010, ISSN 1412-8306.

Putrady, Ecky. 2010. *Optimasi Collision Detection Menggunakan Quadtree*. Jurusan Teknik Informatika, Institut Teknologi Bandung. Bandung.

Sa, Claudio de. 2009. *Basic Collision Detection in 2D – Part 1*. http://devmag.org.za/2009/04/13/basic-collision-detection-in-2d-part-1/. (diakses tanggal 30 April 2016).

Sujjada, Alun., Hariadi, Mochamad., Mardi, Supeno. 2011. *Formasi Pasukan Perang Menggunakan Algoritma Boids*. Teknik Elektro. ITS Surabaya.

Reynolds, C.W. 2010. *Steering Behaviors For Autonomous Characters*. http://www.red3d.com/cwr/steer/gdc99/. (diakses tanggal 01 Mei 2016).