IMPLEMENTASI ALGORITMA *PARTICLE SWARM OPTIMIZATION* (PSO) UNTUK MENENTUKAN *WAYPOINT* DALAM STUDI KASUS SIMULASI (THAWAF)

SKRIPSI

Oleh :
ALDILLA QURRATA A'YUN
NIM 12650090

JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERIMAULANA MALIK IBRAHIM
MALANG
2019

IMPLEMENTASI ALGORITMA *PARTICLE SWARM OPTIMIZATION* (PSO) UNTUK MENENTUKAN *WAYPOINT* DALAM STUDI KASUS SIMULASI (THAWAF)

SKRIPSI

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

Oleh :
ALDILLA QURRATA A'YUN
NIM 12650090

JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM MALANG
2019

LEMBAR PERSETUJUAN

IMPLEMENTASI ALGORITMA *PARTICLE SWARM OPTIMIZATION* (PSO) UNTUK MENENTUKAN *WAYPOINT* DALAM STUDI KASUS SIMULASI (THAWAF)

SKRIPSI

Oleh:
ALDILLA QURRATA A'YUN
NIM 12650090

Telah Diperiksa dan Disetujui untuk Diuji :
Tanggal : Mei 2019

Pembimbing I	Pembimbing II
Presy Nugroho, M.T	Dr.M.Amin Hariyadi M.T
NIP. 19710722 201101 1 001	NIP. 19670118 200501 1 001

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

Dr. Cahyo Crysdian
NIP. 19740424 200901 1 008

ii

HALAMAN PENGESAHAN

IMPLEMENTASI ALGORITMA *PARTICLE SWARM OPTIMIZATION* (PSO) UNTUK MENENTUKAN *WAYPOINT* DALAM STUDI KASUS SIMULASI THAWAF

SKRIPSI

Oleh :
ALDILLA QURRATA A'YUN
NIM. 12650090

Telah Dipertahankan di Depan Dewan Penguji Skripsi dan
Dinyatakan Diterima Sebagai Salah Satu Persyaratan untuk
Memperoleh Gelar Sarjana Komputer (S.Kom)
Tanggal : 13 Juni 2019

Susunan Dewan Penguji		**Tanda Tangan**
Penguji Utama	: M. Ainul Yaqin, M.Kom NIP. 19761013 200604 1 004	( )
Ketua Penguji	: Hani Nurhayati, M.T NIP. 19780625 200801 2 006	( )
Sekretaris Penguji	: Fresy Nugroho, M.T NIP. 19710722 201101 1 001	( )
Anggota Penguji	: Dr. M. Amin Hariyadi, M.T NIP. 19670118 200501 1 001	( )

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

Dr. Cahyo Crysdian
NIP. 19740424 200901 1 008

HALAMAN PERNYATAAN
ORISINALITAS PENELITIAN

Nama : Aldilla Qurrata A'yun
NIM : 12650090
Jurusan : Teknik Informatika
Fakultas : Sains dan Teknologi
Judul Skripsi : **IMPLEMENTASI ALGORITMA *PARTICLE SWARM OPTIMIZATION* (PSO) UNTUK MENENTUKAN *WAYPOINT* DALAM STUDI KASUS SIMULASI 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, Mei 2019
Yang membuat pernyataan,

Aldilla Qurrata A'yun
NIM. 12650090

iv

HALAMAN MOTTO

Setiap manusia mempunyai waktu dan porsi sendiri-sendiri. Jangan hanya menunggu hasil dari seseorang, tapi lihat bagaimana proses dan usaha dari seseorang tersebut untuk bisa berada ditempat dan keadaan yang sekarang.

-Aldilla Q. A'yun

HALAMAN PERSEMBAHAN

Bismillahirrohmanirrohim, kupersembahkan sebuah

karya sederhanaku ini untuk seluruh keluarga besarku

khususnya Ayah dan Mama yang tercinta

Bambang Wiyono dan Maimun fatimah

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

Tak lupa juga karya ini kupersembahkan untuk

Teman-teman yang selalu bertanya :

"kapan selesai"

"kapan lulus"

"kapan wisuda"

Wahai teman-teman tersayangku, akhirnya aku bisa menjawab

pertanyaan kalian dari jilidan ini.

Terimakasih semangat yang tidak henti kalian berikan disela-sela

ejekan kalian Hahaha .

Semoga Allah melindungi kalian. 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 "**Implementasi Algoritma *Particle Swarm Optimization* (PSO) Untuk Menentukan *Waypoint* Dalam Studi Kasus Simulasi 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. Abd. Haris, M.Ag selaku rektor Universitas Islam Negeri Maulana Malik Ibrahim Malang.
2. Dr. Sri Harini, M.Si selaku Dekan Fakultas Sains dan Teknologi UIN Maulana Malik Ibrahim.
3. Bapak Cahyo Crysdian, selaku Ketua Jurusan Teknik Informatika dan dosen wali yang selalu memberi pengarahan terkait akademik selama masa study dan selalu memberi motivasi dan juga semangat
4. 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.
5. Bapak Dr.M.Amin Hariyadi selaku dosen pembimbing 2 yang selalu memberi masukan, serta pengarahan dalam penyusunan laporan 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, Ayah dan seluruh keluarga yang selalu memberikan doa, kasih sayang, semangat, dukungan moril, serta motivasi sampai saat ini, terimakasih banyak.
8. Hartawan Abdillah dan juga Bapak Ibuk yang selalu membantu internal maupun eksternal dalam bentuk semangat kepada penulis untuk menyelesaikan skripsi ini.
9. Sahabat-sahabatku yang namanya terlalu banyak untuk disebut satu per satu, yang sudah selalu memberi motivasi selama menyelesaikan skripsi.
10. Teman-teman angkatan 2012, yang berjuang bersama-sama untuk meraih mimpi, terimakasih atas kenang-kenangan indah yang dilalui bersama.
11. 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, 13 Juni 2019
Penulis

Aldilla Qurrata A'yun

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

ABSTRAK

A'yun, Aldilla Qurrata. **Implementasi Algoritma *Particle Swarm Optimization* (PSO) Untuk Menentukan *Waypoint* Dalam Studi Kasus Simulasi Thawaf.** Skripsi. Jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.

Pembimbing : (1) Fresy Nugroho, M.T
(2) Dr.M.Amin Hariyadi, M.T

Kata Kunci : Thawaf, *Boids*, *Particle Swarm Optimization*

Pada umumnya, ada beberapa jamaah ditengah-tengah saat melakukan ibadah thawaf berpisah dengan rombongannya dan juga jarak antara jemaah atau rombongan yang lain tidak berdekatan. Dengan permasalahan yang disebutkan, peneliti ingin membuat simulasi yang menghasilkan perilaku yang halus serta fleksibel. Penelitian ini dilakukan dengan menggunakan metode yang bisa mendukung fitur AI dengan cara menambahkan perilaku kerumunan pada pergerakan NPC tersebut. Untuk mewujudkan hal tersebut dalam simulasi animasi ini menggunakan algoritma *Particle Swarm Optimization* dan algoritma *Boids*. Algoritma *Boids* digunakan pada jemaah untuk mengatur pergerakan NPC dengan memperhitungkan jarak, kecepatan, dan pergerakan yang baik. Sedangkan Algoritma *Particle Swarm Optimization* digunakan pada *leader* untuk persebaran NPC dalam mencari target atau menentukan *waypoint* agar tetap optimal dalam artian tidak terlalu jauh dan juga tidak terlalu dekat antara jemaah satu dengan jemaah lain, antara rombongan satu dengan rombongan lain.

Dari pengujian yang sudah dilakukan dengan total jemaah 1-15 jemaah yang dibagi dalam 3 rombongan, berhasil melakukan thawaf dengan waktu yang meningkat setiap satu kali putaran. Akan tetapi, saat total jamaah 45 keatas jemaah, ada 1-2 jemaah yang tidak berhasil mengitari dan itu membuat penurunan waktu saat melakukan satu kali putaran thawaf. Hasil untuk rata-rata waktu yang diperlukan dalam melakukan satu putaran simulasi akan meningkat dan juga menurun dengan semakin meningkatnya jumlah jemaah.

ABSTRACT

A'yun, Aldilla Qurrata. **Implementation of Particle Swarm Optimization (PSO) Algorithm to Determine Waypoints in Thawaf Simulation Case Studies**. Thesis. Informatics Engineering Department. Faculty of Science and Technology. State Islamic University of Maulana Malik Ibrahim Malang.

Supervisor : (1) Fresy Nugroho, M.T
(2) Dr.M.Amin Hariyadi, M.T

Keyword : Thawaf, *Boids*, *Particle Swarm Optimization*

Generally, there are several pilgrims in the midst of doing Thawaf got separated from their group and the distance between the pilgrims are way too far. With the problems mentioned, researchers want to make simulations that produce smooth and flexible behavior. This research was conducted using a method that could support AI features by adding crowd behavior to the movement of the pilgrims as the non-Playable Character (NPC). This research simulation uses Particle Swarm Optimization algorithm and the Boids algorithm to realize the expected results. The Boids algorithm is placed on the NPC to regulate their movement by calculating the distance, speed and movement. Whereas Particle Swarm Optimization Algorithm is used on the leaders of the NPC for the distribution of NPCs on determining the waypoint to keep the most optimal distance between the NPCs, and the NPC groups.

The results show that with a total of 1-15 NPC divided into 3 groups gives the best time for each lap. Hence, if the total NPCs goes beyond 45 NPCs, there are 1-2 NPCs which were failed to do a lap and lengthen the time required. The time required to do a lap of thawaf is directly affected by the number of NPCs.

الملخص

أعين ، ألدلّة قرّة. تنفيذ خوارزمية ***Particle Swarm Optimization*(PSO)** لتحديد نقاط الطريق في دراسات الحالة لمحاكاة الطواف. البحث العلمي. قسم الهندسة المعلوماتية كلية العلوم والتكنولوجيا بجامعة الإسلامية الحكومية مولانا مالك إبراهيم مالانج.

المشرف الأول: فريسي نوغراها الماجستير

المشرف الثاني: الدكتور م. أمين هاريادي الماجستير

الكلمات الرئيسة: طواف، *Particle Swarm Optimization*، *Boids*

على شكل عام ، عديد من الحجاج في وسط القيام بعبادة الطواف متصرين عن حاشيتهم وكذلك المسافة بين الجماعة بالجماعات الأخرى بعيدة بعضها بعضا. معتمدا على المشكلات المذكورة، تريد الباحثة إجراء عمليات محاكاة تنتج سلوكًا خفيًا ومرنًا. تم إجراء هذا البحث باستخدام طريقة يمكن أن تدعم ميزات الذكاء الاصطناعي من خلال إضافة سلوك الجماهير لحركة *NPC*. لتحقيق محاكاة الرسوم المتحركة هذه، قامت الباحثة باستخدام خوارزمية تحسين الجسيمات والسرب وخوارزمية *Particle Swarm Optimization*(PSO) و خوارزمية *Boids*. يتم استخدام خوارزمية *Boids* في التجمعات لتنظيم حركة *NPC* عن طريق حساب المسافة والسرعة والحركة الجيدة. في حين تستخدم القادة خوارزمية *Particle Swarm Optimization*(PSO) من قبل القادة لتوزيع *NPC* في إيجاد الأهداف أو تحديد نقطة الطريق بحيث تظل مثالية بمعنى أنها ليست بعيدة جدًا وليست قريبة جدًا بين الجماعة والمجموعات الأخرى.

من الاختبارات التي أجريت مع 1-15 الجمعات مقسمة إلى 3 مجموعات ، تمكنوا من تنفيذ الطواف مع زيادة الوقت كل جولة 1 مرة. ومع ذلك ، عندما يكون إجمالي عدد الحجاج 45 وأعلى من الجماعة ، هناك 1-2 حاج لا يستطيعون الالتفاف حوله ويؤدي ذلك إلى تقليل الوقت عند القيام بجولة واحدة من الطواف. ستزداد نتائج متوسط الوقت اللازم لإجراء جولة محاكاة وستنخفض أيضًا مع زيادة عدد الجماعة.

BAB I

LATAR BELAKANG

1.1 Latar Belakang

Sebagai muslim kita mengenal rukun islam,dimana umat muslim wajib melakukannya. Namun ada salah satu rukun Islam yang dilakukan bagi yang mampu. yaitu Haji. Dalam ibadah Haji sendiri ada rukun-rukun haji yang harus dilakukan selama proses ibadah, salah satunya yaitu Thawaf.

Thawaf memiliki makna mengitari. Dimana putaran thawaf dimulai serta diakhiri dengan arah sejajar pada hajar aswad dan Ka'bah selalu terletak disebelah kirinya dan hanya dilaksanakan di Baitullah sebagaimana Ka'bah kiblat umat muslim dan dilakukan sebanyak tujuh (7) kali putaran.

Dalam hadist riwayat muslim dalam shahihnya dari Ibnu Abbas RA ,

لاَيَنْفِرَنَّ أَحَدٌمِنءكُم حَتَّى يَكُوْنَ آخِرُ عَهْدَهُ بِالْبَيْتِ

Yang artinya : "*Janganlah seseorang diantara kamu pulang melainkan akhir yang dilakukan adalah thawaf di Baitullah*"

Dan dalam Shahih Bukhari dan Shahih Muslim terdapat riwayat dari Ibnu Abbas Radhiallahu anhu, ia berkata.

أُمِرَ النَّاسُ أَنْ يَكُوْنَ آخِرُ عَهْدِهِمْ بِالْبَيْتِ إِلاَّ أَنَّهُ خُفِّفَ عَنِ الْحَائِضِ

Yang artinya : "*Nabi Shallallahu 'alaihi wa sallam memerintahkan manusia (yang haji) agar akhir yang dilakukannya adalah thawaf di Baitullah. Tetapi beliau memberikan keringanan kepada wanita yang haidh*" [Muttafaqun 'alaih].

Dan Rasulullah Shallallahu ‘alaihi wa sallam thawaf ketika selesai dari semua amal hajinya dalam waji wada’ ketika akan pulang ke Madinah, dan beliau bersabda : “*Ambilah dariku manasik hajimu*”. Beberapa hadits tersebut menunjukkan wajibnya thawaf wada’ kecuali bagi wanita yang sedang haidh dan nifas. Maka siapa yang meninggalkannya dari orang-orang yang haji, dia wajib menyembelih kurban karena dia melanggar sunnah Nabi Shallallahu ‘alaihi wa sallam dam meninggalkan ibadah wajib dalam haji. Ini adalah yang benar dari pendapat-pendapat ulama. Sebab terdapat riwayat shahih dari Ibnu Abbas Radhiallahu ‘anhu, ia berkata :

مَنْ تَرَكَ نُسُكًاأَوْنَسِيَهُ فَلْيُرِقْ دَمًا

Artinya : “*Barangsiapa meninggalkan suatu ibadah wajib dalam haji atau lupa, maka dia wajib menyembelih kurban*” [Hadits Riwayat Malik]

Dari penjelasan hadist diatas , disebutkan secara jelas bahwa Thawaf wajib dilakukan saat ibadah haji, sebab apabila tidak melakukan thawaf atau meninggalkan satu kali putaran maka haji tersebut tidak sah kecuali jika jamaah tersebut sedang haid/ nifas bagi perempuan dan membayar dam di Mekkah apabila jamaah tersebut meninggalkan / keluar Mekkah karena sakit atau halangan lainnya.

Ada juga beberapa jamaah yang keliru dalam melaksanakan thawaf saat haji, oleh karena itu kurangnya praktek thawaf dalam manasik haji yang dilakukan calon jamaah haji berakibat kurangnya pengetahuan tentang tata cara thawaf yang benar. Saat melaksanakan thawaf juga perlu dikertahui bagaimana langkah-langkahnya dengan benar, dikarenakan masih ada beberapa jamaah yang

melakukan kesalahan maka perlu simulasi thawaf yang bisa di perkenalkan dengan situasi yang sama dengan kondisi yang ada. Simulasi thawaf sendiri sekarang sudah ada dalam berbagai bentuk, meliputi dalam bentuk game ,*virtual reality, unreal engine,* dan android.

Thawaf dapat dilakukan secara sendirian atau berkelompok. Namun pada penelitian ini simulasi dibuat untuk secara berkelompok yang didalamnya terdapat leader (pemimpin) dan follower (pengikut). Dalam menjalankan thawaf saat ibadah haji sendiri ada kemungkinan terjadi berbenturan dari kelompok satu dengan kelompok yang lainnya. Dari kerumunan satu dengan kerumunan yang lain, karena thawaf sendiri dilakukan oleh jutaan orang muslim didunia yang sedang melakukan ibadah Haji di tanah Suci, Mekkah. Kerumunan sendiripun perlu simulasi guna menghindari hambatan-hambatan statis yang akan terjadi.

Berdasarkan penelitian sebelumnya, simulasi kerumunan sendiri sudah pernah dilakukan oleh peneliti Heru Djamaludin. Dalam penelitiannya ia menggunakan Artificial Bee Colony dan Algoritma Boids untuk pergerakan NPC pada simulasi Thawaf. Namun penerapan metode *artificial bee colony(ABC)* dalam penelitian ini masih kurang fleksibel untuk simulasi kerumanan dalam menentukan waypoint / pencarian jalur posisi. Kekurangan *artificial bee colony(ABC)* sendiri yaitu apabila dimensi meningkat, pertukaran informasi dalam satu dimensi menjadi terbatas dan yang kedua apabila dimensi dipilih secara random (acak)dimananilai *fitness* tinggi maka ada kemungkinan tidak akan dipilih.

Dalam penelitian kali ini, penelitiakan memperbaiki metode *artificial bee colony(ABC)* dengan metode *particle swarm optimization* karena dalam menentukan waypoint dengan metode *particle swarm optimization*menjadilebih cepat dibandingkan dengan *artificial bee colony*. Selain waktu penyelesaian yang lebih cepat, perhitungan yang ada didalam PSO sangat sederhana dibandingkan dengan perhitungan lainnya, dan PSO mengadopsi kode bilangan *real* yang telah ditetapkan secara langsung oleh solusi (Boi,2010.)

1.2 Pernyataan Masalah

Bagaimana menentukan waypoint secara tepat pada pergerakan NPC jamaah agar menghasilkan perilaku yang halus serta fleksibel dengan menerapkan algoritma PSO.

1.3 Batasan Masalah

Batasan masalah pada penelitian ini mencakup :

a. Simulasi diimplementasikan pada platform *Windows* dengan berbasis dekstop.
b. Pada simulasi ini hambatannya berupa hambatan statis yang ada di sekitar Ka'bah.
c. Objek yang dituju di simulasi ini yaitu waypoint yang ada di sekitar Ka'bah.
d. Kecepatan pergerakan secara statis.
e. Penentuan pada titik masuk dan titik keluar pada simulasi.

1.4 Tujuan Penelitian

Tujuan dari penelitian ini ialah mengoptimasi pergerakan NPC guna menghasilkan pergerakan jamaah yang dapat menghindari obstacle dan tidak saling bertabrakan antar rombongan lainnya ,sehingga dapat menghasilkan perilaku yang halus dan fleksibel.

1.5 Manfaat Penelitian

Manfaat dari penelitian ini ialah bisa memberikan dedikasi terhadap perkembangan penemuan tabrakan yang terjadi terhadap objek statis pada animasi komputer, dan metode simulasi kerumunan (*crowd simulation*) bisa dijadikan sebagai salah satu metode yang dapat digunakan untuk kasus demikian.

BAB II

STUDI PUSTAKA

2.1 Dasar Teori

Thawaf merupakan salah satu amal ibadah yang dilakukan muslim pada saat melaksanakan haji dan umrah. Dalam ibadah haji, thawaf wajib dilakukan, sebagaimana yang sudah tertuang dalam Surat Al-Hajj ayat 29 yang artinya :

"*Hendaknya mereka melakukan thawaf sekililing rumah yang tua itu (Baitullah)".*

Adapun syarat thawaf menurut Musthafa Said Al-Khin dan Musthafa Diyeb Al-Bugha dalam *Al-Fiqhul Manhaji 'ala Madzhabil Imamis Syafi'i* juz I, halaman 396 menjelaskan :

1. melaksanakan semua syarat sah shalat , kecuali dalam thawaf kita masih diperkenankan berkomunikasi dengan orang lain.
2. Pundak kiri lurus terus ke arah kiblat, tidak menoleh kearah lainnya .
3. Putaran berlawanan arah jarum jam, dan dimulai dari titik hajar aswad.
4. Putaran dilakukan sebanyak 7 kali.

Banyak calon jamaah yang masih tidak tau pasti syarat thawaf dan bagaimana pelaksanaannya saat manasik haji, sehingga peneliti membuat simulasi thawaf agar nantinya memudahkan para calon jamaah yang mungkin ketika praktik manasik haji masih bingung .

2.1.1 Non Player Character *(NPC)*

Autonomous character disebut juga *Non Player Character* (NPC) yang artinya objek pada sebuah permainan yang berkemampuan dapat bergerak tanpa

dikendalikan oleh PC. NPC merupakan komponen yang sangat penting didalam game, karena keberadaannya dapat menciptakan agen cerdas seperti nyata. Kecerdasan NPC didapat dari AI yang ada didalam game. Tidak adanya NPC yang cerdas akan membuat *game* tidak menarik dan membosankan.

Yoga Baskara dalam penelitiannya menyebutkan bahwa NPC adalah salah satu karakter dalam game yang tidak dikontrol oleh komputer melainkan dikontrol dengan kecerdasan buatan. Dia juga menerangkan bahwa implementasi PSO didalam game yang dia buat yaitu *action*-RPG berjalan dengan baik ketika menentukan karakter NPC saat mendekati *player* dan menentukan NPC yang akan menjadi pemimpin.

Maka dapat disimpulkan, NPC diartikan sebagai karakter yang sepenuhnya dikendalikan oleh komputer atau kecerdasan buatan dan tidak dikendalikan atau dimainkan oleh pemain.

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

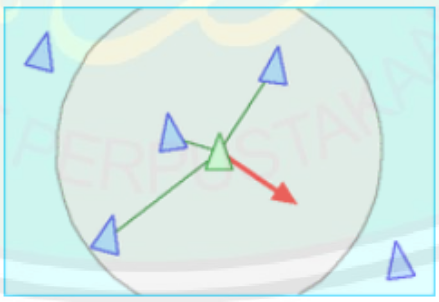
2.1.2 Algoritma *Boids*

Boids merupakan program kehidupan buatan yang dikembangkan oleh Craig Reynolds di tahun 1986. *Boids* berawal dari kata **"***birds***"** dimana ia merupakan algoritma yang merepresentasikan gerak dari sebuah kawanan.Algoritma *boids* merupakan algoritma yang mendukung *flocking behavior* untuk mensimulasikan suatu kelompok agen. Hasil gerak boids pun

cukup menakjubkan, bahkan yang lebih menakjubkan adalah kenyataan bahwa perilaku merupakan hasil dari tiga aturan elegan sederhana yaitu *cohesion, alignment, separation*. Dimana bisa saling menghindar dari *collision*, bisa menyesuaikan arah gerakan dari jumlah rata-rata gerak *boids* seluruhnya, dan yang terakhir memiliki kemampuan untuk bergerak ke arah pusat dari rata-rata 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;

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

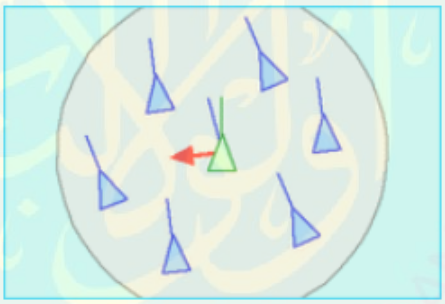
Dimana :

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;

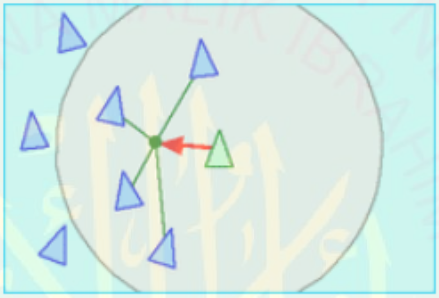
$$\sum_{n \in N} Normalize\,(n_{vel})$$

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

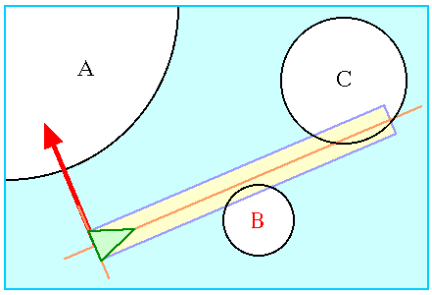
Reynolds (1999) telah menambah model *boids* termasuk aturan individu berbasis lebih dari kemudi perilaku, untuk memiliki individu yang lebih canggih yang mampu untuk menyelesaikan tugas tertentu atau beradaptasi dengan lingkungan yang kompleks. Beberapa perilaku ini adalah (Reynolds, 1999):

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).



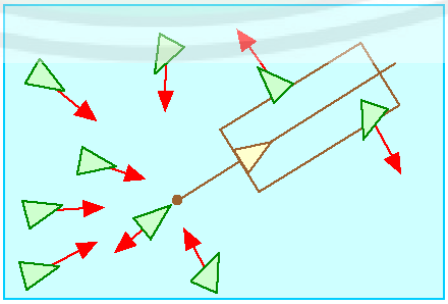
Gambar 2. 4 *Obstacle Avoidance* (Reynold, 2010)

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 Algoritma *Particle Swarm Optimization (*PSO*)*

Di tahun 1995, Dr. Eberhart dan Dr.Kennedy mengembangkan tentang algoritma *Particle Swarm Optimization* (PSO) yang terinspirasi oleh perilaku social dari sekelompok burung (*flock of bird*) atau sekumpulan ikan (*school of fish*).Dari pengembangannya tersebut, Dr. Eberhart dan Dr.Kennedy menyimpulkan bahwa *Particle Swarm Optimization* (PSO) merupakan suatu populasi yang berdasarkan pada teknik optimasi stokastik.

Particle Swarm Optimization (PSO) didasarkan pada perilaku sekelompok burung atau ikan. Algoritma PSO meniru pada perilaku sosial organisme. Perilaku sosial sendiri terdiri dari tindakan individu dan pengaruh dari individu-individu yang lain dalam suatu kelompok. Kata partikel diartikan, misalnya, seekor burung

Dalam kelompok burung. Tiap individu atau partikel akan berperilaku menggunakan kecerdasannya(*intelligence*) sendiri yang juga dipengaruhi akan perilaku kelompok kolektifnya. Dengan begitu, jika satu partikel atau seekor

burung menemukan rute yang tepat atau pendek menuju sumber makanan, maka sisa kelompok lainnya akan segera mengikuti rute tersebut meski lokasi mereka jauh dari kelompok tersebut.

Dalam *Particle Swarm Optimization* (PSO), kawanan diasumsikan mempunyai ukuran tertentu dengan posisi awal terletak di lokasi yang acak dalam ruang multidimensi. Tiap partikel diasumsikan memiliki dua karakteristik, yaitu posisi dan kecepatan. Tiap partikel yang bergerak dalam ruang/space tertentu akan mengingat posisi terbaik yang pernah dilalui terhadap sumber makanan atau nilai fungsi objektif. Tiap partikel juga akan menyampaikan informasi atau posisi terbaiknya kepada partikel yang lain dan menyesuaikan posisi serta kecepatan masing-masing berdasarkan informasi yang telah diterima terkait posisi tersebut. Sebagai contoh, misalkan perilaku burung-burung dalam dalam sekelompok burung.

Meskipun setiap burung mempunyai keterbatasan dalam hal kecerdasan, biasanya ia akan mengikuti aturan (rule) sebagai berikut :

1. Seekor burung berada tidak terlalu dekat dengan burung yang lain.
2. Seekor burung akan mengarahkan terbangnya ke arah rata-rata keseluruhan burung
3. Seekor burung akan memposisikan diri dengan jumlah rata-rata dari posisi burung yang lain guna menjaga jarak antar burung sehingga didalam kawanan tersebut agar tidak terlalu jauh.

Dengan demikian, perilaku kawanan burung yang berdasarkan kombinasi 3 sederhana meliputi :

1. Kohesi - terbang bersama
2. Separasi – jarak tidak terlalu dekat
3. Penyesuaian (*alignment*) - mengikuti arah secara bersama

Pada algoritma PSO, pencarian solusi akan dilakukan oleh suatu populasi yang terdiri dari beberapa partikel. Populasi tersebut dibangkitkan secara randomdengan batasan dari nilai terkecil sampai nilai terbesar. Setiap partikel merepresentasikan posisi atau solusi dari permasalahan yang dihadapi. Setiap partikel juga melakukan pencarian solusi yang sangat optimal dengan melintasi ruang pencarian (*search space*). Hal ini dilakukan dengan cara tiap partikel melakukan penyesuaian terhadap posisi terbaik dari partikel tersebut (*local best*) dan penyesuaian terhadap posisi partikel terbaik dari seluruh kawanan (*global best*) selama melintasi ruang pencarian. Jadi, penyebaran informasi terjadi di dalam partikel itu sendiri dan antar suatu partikel dengan partikel terbaik dari seluruh kawanan. Selanjutnya, dilakukan proses pencarian untuk mencari posisi terbaik tiap partikel dalam jumlah iterasi tertentu sampai didapatkan posisi yang relatif steady atau mencapai batas iterasi yang telah ditentukan. Pada setiap iterasi, setiap solusi yang direpresentasikan oleh posisi partikel, akan dievaluasi performansinya dengan cara memasukkan Solusi tersebut kedalam *fitness function*.

Setiap partikel diperlakukan seperti sebuah titik pada suatu dimensi ruang tertentu. Kemudian terdapat dua faktor yang memberikan karakter terhadap status partikel pada ruang pencarian yaitu posisi partikel dan kecepatan partikel [Kennedy and Eberhart, 1995].

Berikut ini merupakan rumus matematika yang menjelaskan posisi dan kecepatan partikel pada suatu dimensi ruang tertentu :

$$Xi(t) = xi1(t),xi2(t),..., xiN(t) \quad (1)$$

$$Vi(t) = vi1(t),vi2(t),..., viN(t) \quad (2)$$

Dimana :

X = posisi partikel

V = kecepatan partikel

i = indeks partikel

t = iterasi ke-t

N = ukuran dimensi ruang

Rumus dibawah ini merupakan model matematika yang menggambarkan mekanisme pembaruan status partikel (Kennedy and Eberhart [1995]):

$$Vi(t) = Vi\ (t-1) + c1r1(Xi\ L- Xi(t-1))+c2r2(XG-Xi(t-1)) \quad (3)$$

$$Xi(t) = Vi(t)+Xi(t-1) \quad (4)$$

Berikut langkah-langkah perhitungan algoritma PSO :

- Tentukan posisi awal sejumlah partikel sekaligus kecepatan awalnya secara random.
- Evaluasikan *fitness* dari tiap partikel berdasarkan letak posisinya.

- Tentukan partikel dengan *fitness* terbaik, dan tetapkan sebagai *Gbest*. Untuk setiap partikel, nilai *Pbest* awal akan sama dengan posisi awal.

Ulangi langkah berikut sampai stopping criteria dipenuhi :

1. Gunakan *Pbest* dan *Gbest* yang sudah ada, kemudian perbarui nilai kecepatan tiap partikel menggunakan persamaan (3). Lalu hasil dari kecepatan yang sudah didapat, perbarui posisi tiap partikel menggunakan persamaan (4).
2. Evaluasi *fitness* dari tiap partikel.
3. Tentukan partikel dengan *fitness* terbaik, dan tetapkan sebagai *Gbest*. Untuk setiap partikel, tentukan *Pbest* dengan cara membandingkan posisi yang sekarang dengan *Pbest* dari iterasi sebelumnya.
4. Cek *stopping criteria*. Jika terpenuhi, maka berhenti. Jika tidak, maka kembali ke persamaan (1).

2.2 Penelitian Terkait

2.2.1 Implementasi Algoritma *Particle Swarm Optimization* untuk Penentuan Posisi Strategis Agent pada Simulasi Robot Sepak Bola Dua Dimensi

Penelitian ini dilakukan oleh Galih Hermawan,2012. Pada penelitian ini, peneliti menerapkan algoritma *particle swarm optimization* (PSO) untuk menentukan posisi robot ketika bermain sepak bola. Hasil pengujian pada simulasi RoboCup Soccer dua dimensi menunjukkan bahwa tim robot sepak bola yang menggunakan algoritma PSO memiliki performa bermain lebih baik daripada sebelum menggunakan algoritma PSO.

Dalam hal ini, peneliti ingin mengembangkan sebuah tim sepak bola virtual dengan menggunakan algoritma PSO guna menentukan posisi *(agent positioning*) dan pergerakan agent (pemain sepak bola virtual) ketika menyerang ke area lawan dan bertahan di daerah sendiri.

2.2.2 *Using Hybrid Artificial Bee Colony Algorithm and Particle Swarm Optimization For Training Feed-Forward Neural Networks*

Pada penilitian ini dilakukan oleh M. Ghanem , dkk, 2014. Pada penilitian ini, peniliti menggunakan beberapa algoritma. Salah satunya *Ant Bee Colony (*ABC) dan *Particle Swarm Optimization* (PSO). Peneliti membandingkan antara metode PSO dengan metode ABC untuk Rangkaian *Training Neural Feed-Forward.* Dan hasil pengujian menghasilkan bahwa metode PSO menghasilkan waktu tempuh lebih cepat dibandingkan dengan algoritma ABC.

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

Penelitian ini dilakukan oleh Jatiningsih, dkk, 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.

2.2.4 Implementasi Prilaku Berkelompok pada Swarm Robots Menggunakan Teknik Logika *Fuzzy-Particle Swarm Optimization*

Penelitian ini dilakukan oleh Nurmaini, 2014.Pada penelitian ini menggunakan 3 robot sederhana yang identik dengan 3 sensor infra-red, sensor kompas dan *X-Bee*. Untuk mencapai target dan menentukan posisi dari masing-masing robot digunakan sebuah sensor kamera yang terhubung dengan computer yang berfungsi sebagai pusat informasi dan penyimpan data.

Untuk menghasilkan kinerja yang baik dalam penelitian ini menggnunakan teknik Logika *Fuzzy-Particle Swarm Optimization* (PSO). Dari pengujian yang telah dilakukan diperoleh hasil dari robot yang ketiga dapat menemukan posisi terbaik dan menghasilkan pergerakan yang halus serta mampu mencapai target yang telah ditentukan dengan menggunakan algoritma PSO.

2.2.5 Penerapan Kecerdasan Kelompok untuk Penyelesaian Teka Teki Sudoku dengan Metode *Particle Swarm Optimization*

Penelitian ini dilakukan oleh Wardani,dkk, 2015.Dalam penelitian ini peneliti menguji kaedah PSO untuk menyelesaikan teka-teki sudoku dengan beberapa tahap dimensi 4x4 sehingga 25x25. Peneliti menggunakan algoritma PSO karena PSO mempunyai konsep bahwa tiap individu akan menyimpan jejak jejak posisinya dalam problem space. Nilainya, yaitu *fitness value* , yang disebut *pbest*. Selain *pbest* yang merupakan milik individu yang bersangkutan, ikut disimpan juga nilai terbaik milik individu yang ada disekitarnya (*local best)* yang disebut *lbest.*

Penelitian yang telah dihasilkan menunjukkan bahwa nilai *pbest* pada iterasi selanjutnya terkadang lebih kecil dari *pbest* sebelumnya. Hal ini terjadi karena nilai kecepatan berfungsi sebagai perhitungan pengacakan terhadap partikel. Penggunaan nilai acak juga terkadang menghasilkan nilai yang lebih kecil, sehingga partikel bisa bergerak maju atau mundur.

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

Penelitian ini dilakukan oleh Mu'min,dkk, 2015. Pada penelitian ini peneliti membahas bagaimana memodelkan algoritma *Boids*yang diberikanalgoritma PSO (*Particle Swarm Optimization*) guna mencapai posisi optimum untuk menciptakan kemungkinan secara otomatis untuk menghasilkan jalan *non deterministic* kerumunan pasukan dari suatu posisi tertentu. Penelitian ini fokus pada pembuatan bagaimana pola akan bergerak secara halus dan fleksibel realistis bagi pasukan virtual dengan memanfaatkan fasilitas komputasi yang ditawarkan oleh PSO.

2.2.7 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%.

2.2.8 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.

2.2.9 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 menghasilkan representasi dari kehidupan buatan secara realitas dari suatugerakan. Pergerakan pasukan dihitung dengan menggabungkan semua vektor *steering behaviour*. Pasukan akan

melakukan perlambatan dan percepatan untuk menyelaraskan posisi antar agen atau pasukan. Algoritma PSO disini digunakan untuk mengoptimalkan model *boids* dengan mengoptimalkan koefisien dari vektor bergerak untuk meminimalkan fungsi *cost* dan *flocking*agar terlihat lebih bagus.

BAB III
DESAIN DAN IMPLEMENTASI

3.1 Tahapan Penlitian

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 Desain skenario

Pada simulasi thawaf ini ada 2 karakter yang berjalan. Yang pertama leader (pemimpin) , yang kedua follower (jamaah). Pada saat simulasi dimainkan maka leader akan berjalan mencari titik waypoint menggunakan metode *Particle Swarm Optimization* yang kemudian diikuti oleh jamaah yang sudah diterapkan metode *Boids.*

3.1.2 Desain NPC dan *Enviroment*

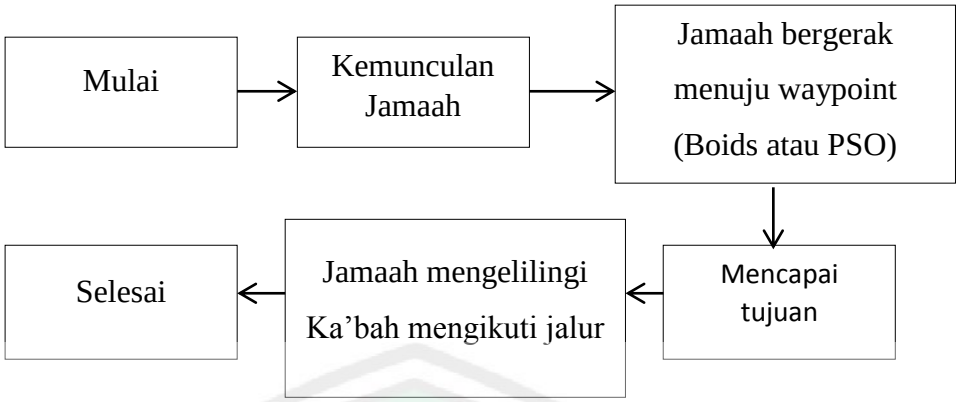
Gambaran NPC pada simulasi ini berupa jamaah laki-laki dan untuk enviromentnya berupa lingkungan sekitar ka'bah yang dimana nantinya ketika simulasi ini dimainkan ada lantunan Talbiyah yang mengiringi.

3.1.3 Pembuatan simulasi

Proses pembentukan simulasi dibangun dengan menggunakan unity3d *engine* yang menggunakan bahasa C# dan Javascript serta scripting dilakukan dengan fasilitas Visual Studio.

3.2 Analisis dan Pembuatan Simulasi

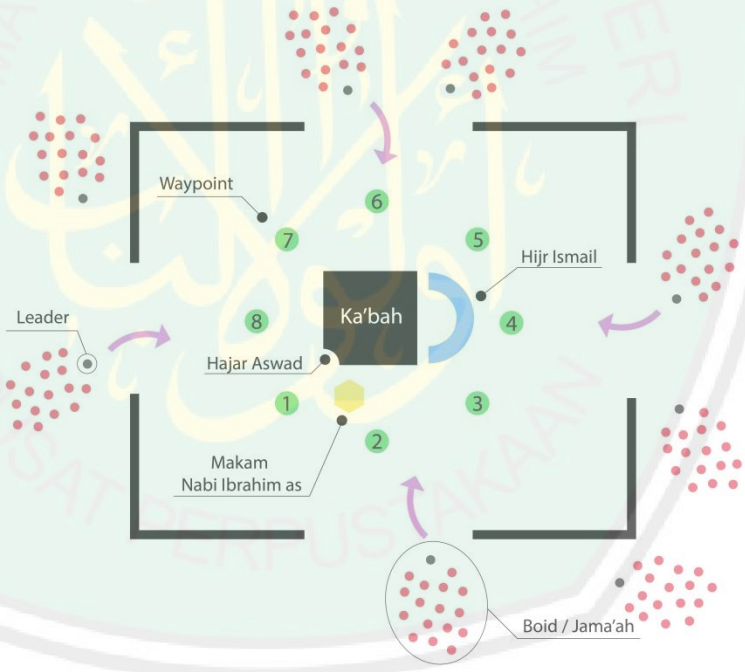
Simulasi ini dibuat dengan mengimplementasikan 2 algoritma, yaitu *Boids* dan *Particle Swarm Optimization. Pada simulasi ini terdapat karakter yang di desain agar simulasi terlihat menarik kertika dimainkan.* Karakter tersebut merupakan karakter NPC *(Non Player Character* dimana NPC tersebut ialah jemaah yang sedang mengiatari Ka'bah dan sudah diberikan perilaku cerdas dengan mengimplementasi algoritma *Boids* dan *PSO*. Metode ini diterapkan untuk perubahan perilaku jemaah saat bergerombol ketika menuju target. Pada simulasi ini, saat jemaah muncul maka algoritma *Boids* dan *PSO* otomatis akan bekerja (lihat Gambar 3.2). Simulasi dirancang pada PC dan dibangun dengan grafis 3 dimensi



Gambar 3.2 Diagram alur Simulasi

3.2.1 Gambaran Umum Simulasi

Pada simulasi ini jemaah harus mengitari Ka'bah. Berikut gambaran latar simulasi yang ditunjukkan dalam bentuk gambar.

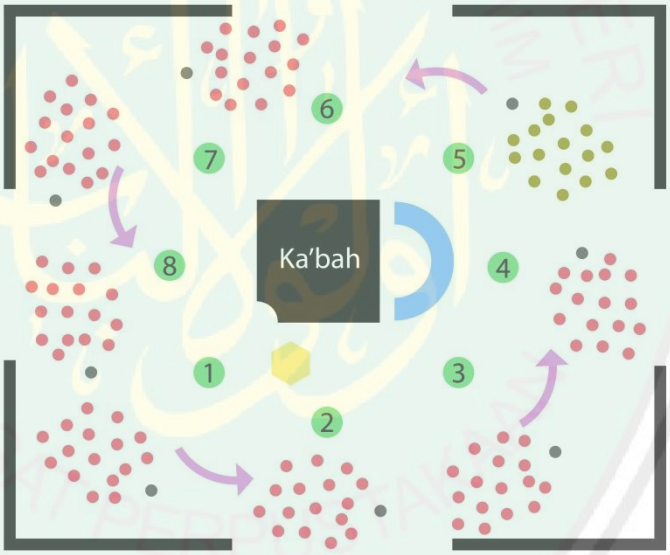


Gambar 3.3 Posisi munculnya jamaah yang bergerak menuju Ka'bah

Pada bagian ini (lihat gambar 3.3) , digambarkan posisi munculnya jemaah yang akan tersebar secara random di titik-titik area persebaran yang telah ditentukan. Tiap jemaah terdapat satu atau lebih agen yang dipilih sebagai *leader*

(pemimpin) digerombolan tersebut. Kemanapun pemimpinya bergerak maka anggotanya akan mengikutinya.

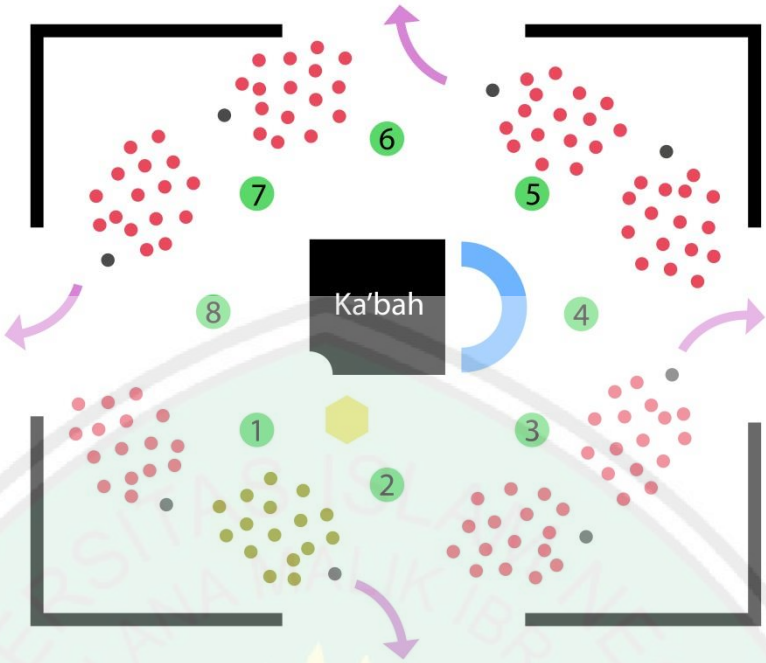
Jemaah akan bergerak menuju *waypoint* terdekat yang ada di area Ka'bah . Saat kemunculan dimulai, maka disinilah algoritma *Boids* dan PSO akan bekerja. Algoritma PSO diberikan kepada *leader* yang berfungsi untuk bergerak menuju target (*waypoint)* yang ada di area Ka'bah. Sedangkan algoritma *boids* berfungsi untuk menjaga agar tidak saling bertabrakan antar jemaah satu dengan jemaah lainnya, menjaga kecepatan agar bergerak secara bersamaan dan menjaga agar jemaah tetap berada dengan gerombolannya ketika menuju target.



Gambar 3.4 Jamaah Mengitari Ka'bah

Setelah jamaah berada di sekitar area Ka'bah , pada bagian ini (lihat gambar 3.4) jemaah akan thawaf dengan mengikuti jalur *waypoints* yang ada di sekitar are Ka'bah. Untuk arah melakukan thawaf sendiri berlawan dengan arah jarum jam. Apabila leader menuju titik *waypoints* kesatu, maka anggota jemaah yang lainnya akan mengikutinya hingga titik terakhir. Disekitar area Ka'bah akan terdapat hambatan statis, yaitu Hijr Ismail & Makam Nabi Ibrahim as. Dan tiap jemaah harus menghiindari hambatan tersebut untuk menghindari tabrakan dengan *obstacle*.

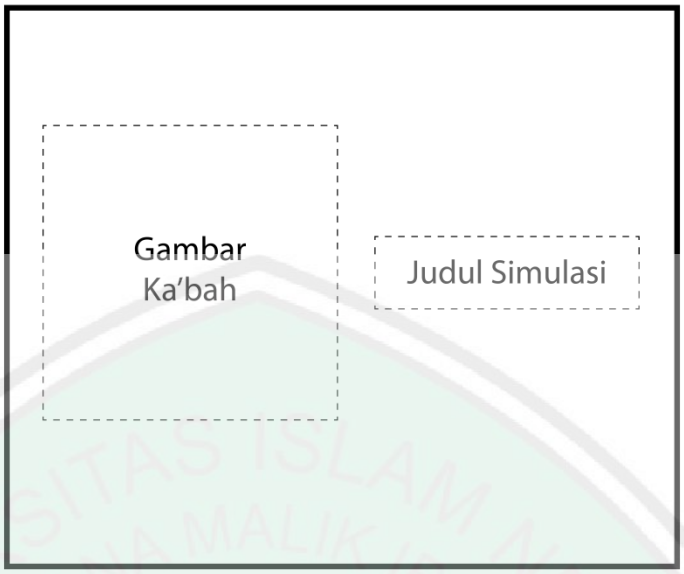
Jemaah melakukan thawaf sebanyak 7 kali putaran. Setiap putaran (pertama s.d ketujuh) dimulai dari Rukun Hajar Aswad dengan menghadapkan muka sambil mengangkat tangan, kemudian dilanjut mengecup dan membaca do'a. Jemaah membaca do'a mulai dari Rukun Hajar Aswad hingga Rukun Yamani. Pada saat setiap kali sampai Rukun Yamani jemaah mengangkat tangan tanpa mengecup sambil membaca do'a. Dan diantara Rukun Yamani dan Rukun Hajar Aswad terdapat do'a yang harus dibaca oleh jemaah.



Gambar 3.5 Jamaah keluar dari area Ka'bah

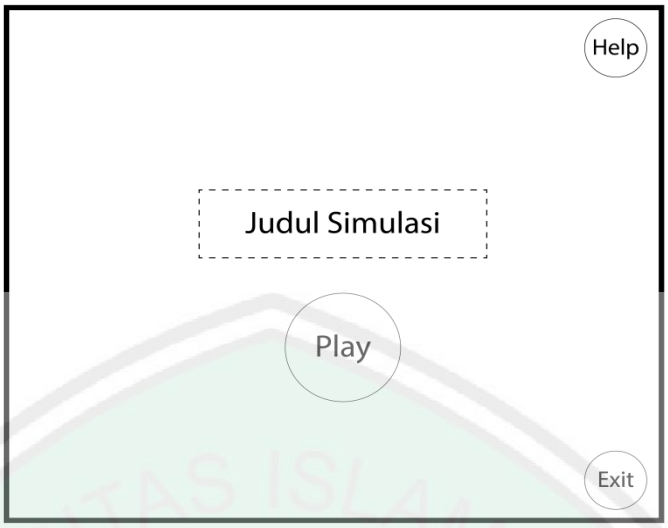
Setelah jemaah selesai melakukan thawaf sebanyak tujuh putaran, jemaah keluar dari area Ka'bah yang menandakan bahwa pelaksanaan thawaf telah selesai. Di sini, *leader* berperan untuk memeriksa apakah telah melakukan tujuh putaran atau belum. Jika tujuh putaran telah selesai *leader* akan bergerak keluar dari area Ka'bah dan anggotanya akan mengikutinya.

3.2.2 Desain Rancangan *Interface* Simulasi



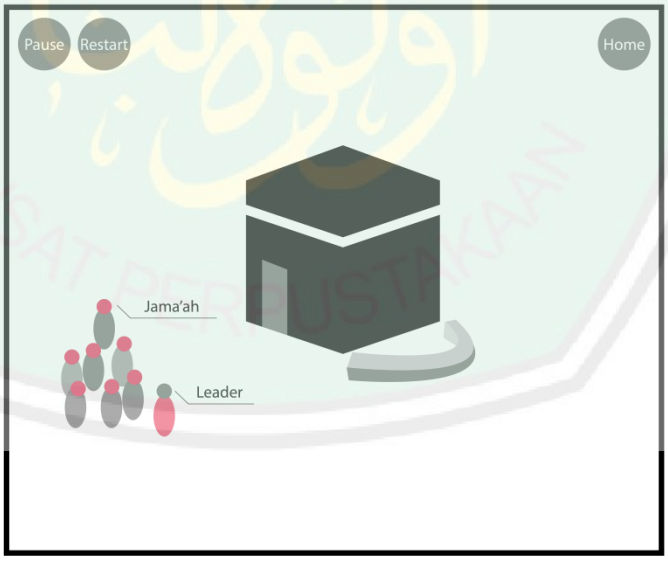
Gambar 3.6 Desain *Splashscreen*

Gambar 3.6 merupakan desain awal simulasi. Dimana nantinya akan muncul tiap kata satu persatu dan kata dimulai dari yang paling atas. Tiap kata yang keluar dari awal sampai akhir akan berselang 2 detik. Sehingga *splashscreen* diatas akan muncul selama 4 detik. Selain itu juga terdapat backsound ketika *splashscreen* dijalankan.



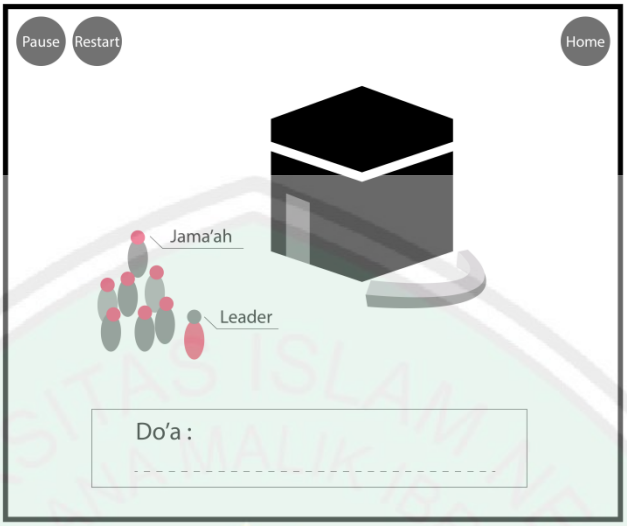
Gambar 3.7 Desain Menu

Gambar 3.7 desain menu dalam simulasi yang akan muncul setelah desain *splashscreen*. Di dalam menu terdapat 3 *button* (tombol). Pertama tombol mulai untuk memulai simulasi, kedua tombol informasi yang berisi tentang informasi simulasi, dan yang ketiga tombol keluar untuk keluar dari aplikasi.



Gambar 3.8 Desain Simulasi

Gambar 3.8 merupakan desain simulasi yang sedang berjalan ketika tombol mulai dipilih dan jemaah mengitari Ka'bah untuk melakukan ibadah thawaf.



Gambar 3.9 Desain petunjuk Do'a yang nantinya muncul beberapa *text box* yang berisi bacaan do'a yang harus dibaca jemaah saat sedang thawaf.

3.2.3 NPC

3.10 Jemaah (NPC) Laki-Laki

Gambar 3.10 ialah karakter jemaah laki-laki yanga da di simulasi thawaf.

3.3 Penerapan *Particle Swarm Optimization* (PSO)

Pada algoritma PSO, pencarian solusi dilakukan oleh suatu populasi yang terdiri dari beberapa partikel. Populasi akan dibangkitkan secara random dengan batasan dari nilai terkecil sampai nilai terbesar. Tiap partikel akan merepresentasikan posisi dari permasalahan yang sedang dihadapi. Partikel melakukan pencarian solusi secara optimal dengan melintasi ruang pencarian (*seacrh space)*. Hal ini dilakukan dengan cara tiap partikel melakukan penyesuaian terhadap posisi terbaikdari partikel tersebut (*local best*) dan penyesuaian terhadap posisi partikel terbaik dari seluruh kawanan (*global best*) selama melintasi ruang pencarian.

Setiap partikel akan diperlakukan seperti sebuah titik pada suatu dimensi ruang tertentu. Terdapat dua faktor yang akan memberikan karakter terhadap status partikel pada ruang pencarian yaitu posisi partikel dan kecepatan partikel [Kennedy and Eberhart, 1995].

3.4 Perhitungan Algoritma *Particle Swarm Optimization (PSO)*

Dimisalkan terdapat fungsi sebagai berikut :

$$\text{minimasi} \quad f(x) \tag{5}$$

dimana

$$x^{(B)} \leq x \leq x^{(A)}$$

Dimana x(B) adalah batas bawah dan x(A) adalah batas atas dari X.

Prosedur PSO dijelaskan dengan langkah-langkah sebagai berikut Rao [2009];

1. Di asumsikan bahwa ukuran kawanan (jumlah partikel) adalah *N*. Untuk mengurangi jumlah evaluasi fungsi yang diperlukan dalam menemukan solusi, ukuran *N* jangan terlalu besar dan jangan terlalu kecil, agar ada banyak kemungkinan posisi untuk menuju mencari solusi terbaik. Jika terlalu kecil maka sedikit kemungkinan untuk menemukan posisi partikel yang baik. Namun jika terlalu besar juga akan membuat perhitungan jadi panjajng. Biasanya ukuran kawanan menggunakan ukuran mulai 20 hingga 30 partikel.
2. Bangkitkan populasi awal x dengan rentang x(B)d an x(A) secara acak sehingga didapat x1,x2, ... hingga x*N*. Partikel *j* dan kecepatan pada iterasi *i* dinotasikan sebagai *xj(i)* dan *vj(i)* , sehingga partikel awal dinotasikan *x1(0), x2(0), ..., xN(0).*

   Vektor

   v1(0),v2(0),....,vN(0)

   disebut partikel atau vektor koordinat dari partikel. Selanjutnya dilakukan evaluasi nilai fungsi tujuan untuk tiap partikel dan nyatakan dengan

   f(x1(0), f(x2(0)),..., f(xN(0))

3. Hitung kecepatan dari semua partikel yang bergerak menuju titik optimal dengan suatu kecepatan tertentu. Asumsikan kecepatan dari partikel = 0. Atur iterasi dengan *i=1*.
4. Di iterasi ke-*i*, tentukan 2 parameter penting untuk setiap partikel *j yaitu :*
   - Nilai terbaik sejauh *xj(i)* (koordinat partikel *j* pada iterasi *i)* dan nyatakan sebagai *Pbest j*, dengan nilai fungsi tujuan paling kecil, *f[xj (i)]*, yang ditemui partikel *j* di semua iterasi sebelumnya. Nilai terbaik untuk semua partikel *xj(i)* yang ditemukan sampai iterasi ke-1,*Gbest*, dengan nilai fungsi tujuan paling kecil diantara semua partikel di semua iterasi sebelumnya *f[xj (i)*].
   - Hitung kecepatan partikel *j* pada iterasi ke *i* dengan rumus :

     *Vj (i) = vj (*i-t*) +*c1r1*[Pbest.j-xj (*i-t*)]* *(6)*

     c2r2 [Gbest – *xj* (*i*-t)] = 1,2, ..., N

     dimana c1 dan c2 merupakan *learning rates* untuk kemampuan individu (cognitive) dan pengaruh sosial (kawanan), dan r1 dan r2 merupakan bilangan random yang terdistribusi uniformal dalam interval 0 dan 1. Sehingga parameter c1 dan c2 menunjukkan bobot dari memory (*position)* sebuah partikel terhadap memory (posisi) dari kawanan (swarm). Nilai dari c1 dan c2 biasanya =2 sehingga perkalian c1r1 dan c2r2 bisqa dipastikan bahwa partikel-partikel akan mendekati target sekisar setengah selisihnya.
   - Hitung posisi atau koordinat partikel *j* pada iterasi ke -1 dengan cara berikut :

     Xj (i) = xj (i-t) + vj (i), j = 1,2, ... N *(7)*

Kemudian evaluasikan nilai fungsi tujuan untuk setiap partikel dan nyatakan sebagai :

F[x1(i), f[x2(i)], ... , f[xN(i)]

5. Periksa apakah solusi yang sekarang sudah konvergen. Apabila posisi semua partikel menuju ke satu nilai yang sama, maka disebut konvergen. Apabila elum konbvergen maka ulangi langkah nomer 4 dengan memperbarui iterasi $i = i+1$, dengan menghitung nilai baru dari *Pbest* dan *Gbest.* Lakukan proses iterasi sampai semua partikel menuju ke satu tiitk solusi yang sama.

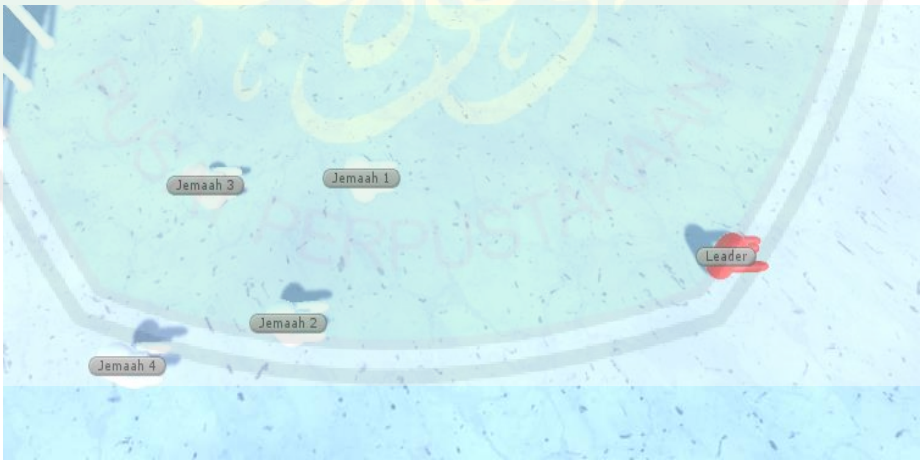
BAB IV
UJI COBA DAN PEMBAHASAN

4.1 Pengujian Algoritma *Boids*

Pengujian *Boids* pada simulasi ini, dilakukan dengan beberapa tahapan yaitu *Separation, Alignment, Cohesion,* dan *Obstacle Avoidance*.

4.1.1 Pengujian *Separation*

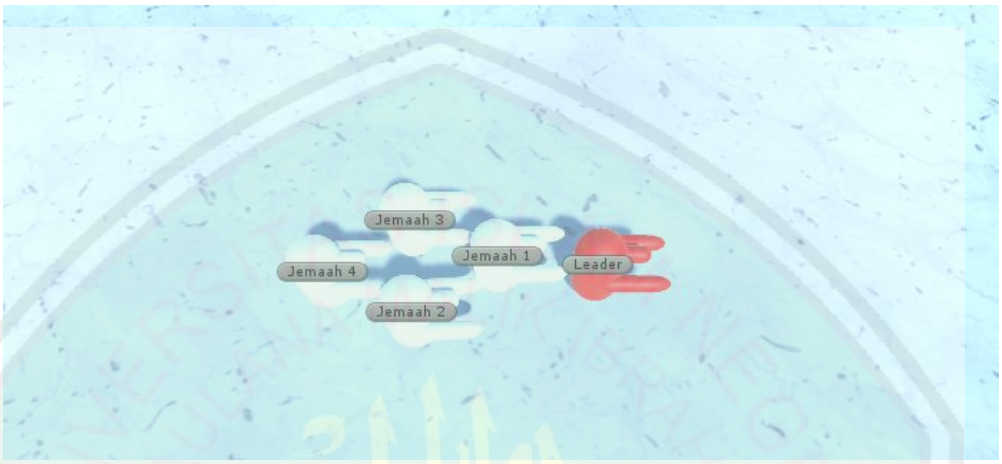
Pengujian *Separation* dilakukan untuk melihat bagaimana jalannya apliakasi tersebut tetap menjaga jarak agar tidak terjadi tabrakan antar jemaah yang dilakukan oleh *boids*. Pengujian ini dilakukan saat jemaah berada pada titik–titik koordinat yang sudah ditentukan (lihat Gambar 4.1). Tiap jemaah memiliki kecepatan maksimal bergerak 2.0f untuk mengikuti *leader* yang bergerak menuju target, dan setiap jemaah memiliki ambang batas radius 2.0f untuk menjaga agar tidak terjadi tabrakan. Lingkungan yang digunakan didalam pengujian ini tidak ada penghalang (*obstacle*) dan menggunakan 5 jemaah dan 1 sebagai *leader*.



Gambar 4. 1 Contoh tata letak awal jemaah

Dengan algoritma *boids* jemaah diharapkan dapat menjaga agar tidak terjadi tabrakan dengan jemaah lainnya. Pengujian dilakukan untuk mengecek apakah

algoritma telah berfungsi atau tidak sehingga jemaah dapat menjaga jarak agar tidak terjadi tabrakan. Pengujian dilakukan dengan jumlah iterasi sebanyak 25 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

Dari gambar diatas dapat dilihat bahwa jemaah dapat bergerombol dan saling menjaga jarak satu sama lain. Dari uji coba didapat jarak jemaah ketika mendekati jemaah lainnya (lihat Tabel 4.1). Jika jarak bernilai lebih dari 0, maka artinya tidak ada tabrakan antar jemaah satu dengan jemaah lainnya. Dari data tersebut didapatkan bahwa jarak terkecil dari seluruh jemaah untuk menjaga jarak antar jemaah lainnya adalah 1.98f pada iterasi 25. 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

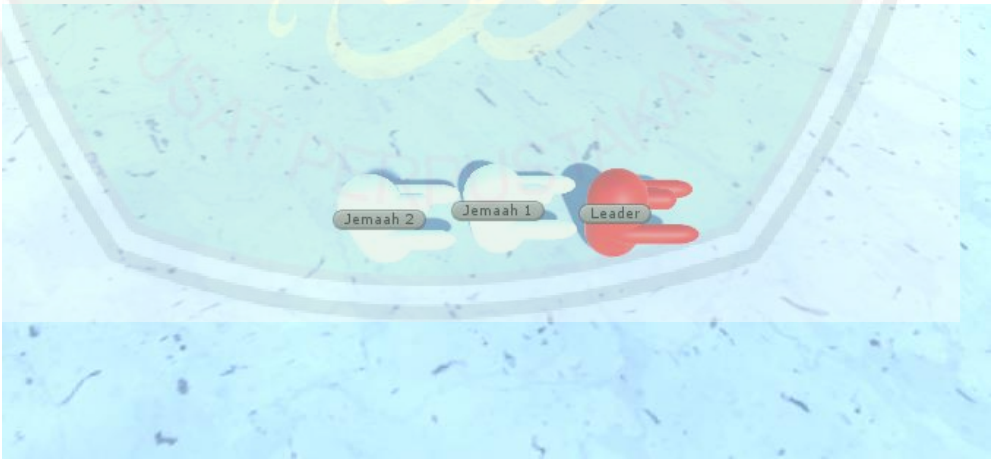
6	8.89	2.68	5.91	7.33
7	7.87	2.34	5.91	7.13
8	6.87	2.08	5.9	6.98
9	6.1	2.01	5.69	6.69
10	5.35	2	5.53	6.5
11	4.59	2.01	5.66	6.6
Iterasi	**Jarak Jemaah 1 Terhadap Jemaah Lainnya**			
	Leader	**Jemaah 2**	**Jemaah 3**	**Jemaah 4**
12	3.84	2	5.8	6.73
13	3.21	1.98	5.86	6.77
14	2.84	2.03	5.59	6.56
15	2.61	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

4.1.2 Pengujian *Alignment*

Pengujian *Alignment* ini dilakukan untuk melihat bagaimana jalannya apliakasi tersebut agar kecepatan jemaah saat bergerombol yang dilakukan oleh *boids* tetap terjaga. Pengujian ini dilakukan saat jemaah berada di titik–titik koordinat yang sudah ditentukan (lihat Gambar 4.3). Setiap jemaah memiliki 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 tidak ada penghalang (*obstacle*) dan menggunakan 3 jemaah dan 1 sebagai *leader*.

Gambar 4. 3 Contoh tata letak awal jemaah

Dengan algoritma *boids* jemaah diharapkan dapat menyesuaikan kecepatan dengan kelompok. Pengujian dilakukan untuk mengecek apakah algoritma telah berfungsi atau tidak sehingga jemaah dapat menyesuaikan kecepatan dengan kelompok. Pengujian dilakukan dengan jumlah iterasi sebanyak 25 kali. Hasilnya jemaah dapat bergerombol dan bisa menyesuaikan kecepatan dengan kelompoknya (lihat Gambar 4.4).



Gambar 4. 4 Jemaah menyesuaikan kecepatan dengan jemaah yang lainnya

Dari gambar diatas dapat dilihat bahwa jemaah dapat bergerombol dan berusaha menyesuaikan kecepatan dengan kelompoknya. Jemaah di detik awal

menggunakan kecepatan maksimal, karena jemaah mencoba untuk bergerombol mendekati *leader*, akan Tetapi pada detik ke-11 terjadi penurunan kecepatan. Dikarenakan jemaah berusaha menyesuaikan kecepatan dengan kelompok saat bergerombol.

4.1.3 Pengujian *Cohesion*

Pengujian *Cohesion* dilakukan untuk melihat bagaimana jalannya aplikasi menjaga jemaah tetap dekat dengan jemaah lainnya yang dilakukan oleh *boids*. Pengujian ini dilakukan saat jemaah berada di titik–titik koordinat yang sudah ditentukan (lihat Gambar 4.5). Setiap jemaah memiliki 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 tidak ada penghalang (*obstacle*) dan menggunakan 5 jemaah dan 1 sebagai *leader*.

Gambar 4. 5 Contoh tata letak awal jemaah

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

Gambar 4. 6 Jemaah menjaga jarak tetap dekat dengan jemaah lainnya

Dari gambar diatas dapat dilihat bahwa jemaah dapat bergerombol dan ketika jemaah mendekati jemaah lainnya, jemaah mencoba untuk tetap dekat dengan jamaah lainnya. Dari uji coba didapatkan 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 1.98f.

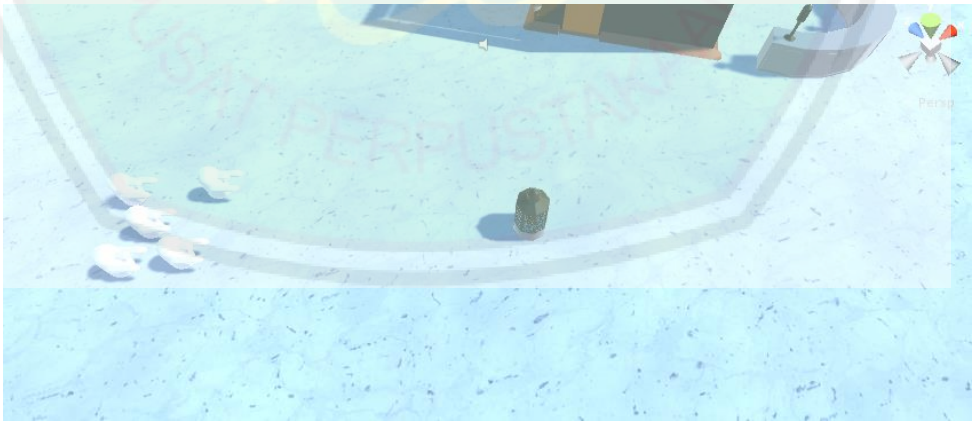
Tabel 4. 2 Pengujian *Cohesion*

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

4.1.4 *Obstacle Avoidance*

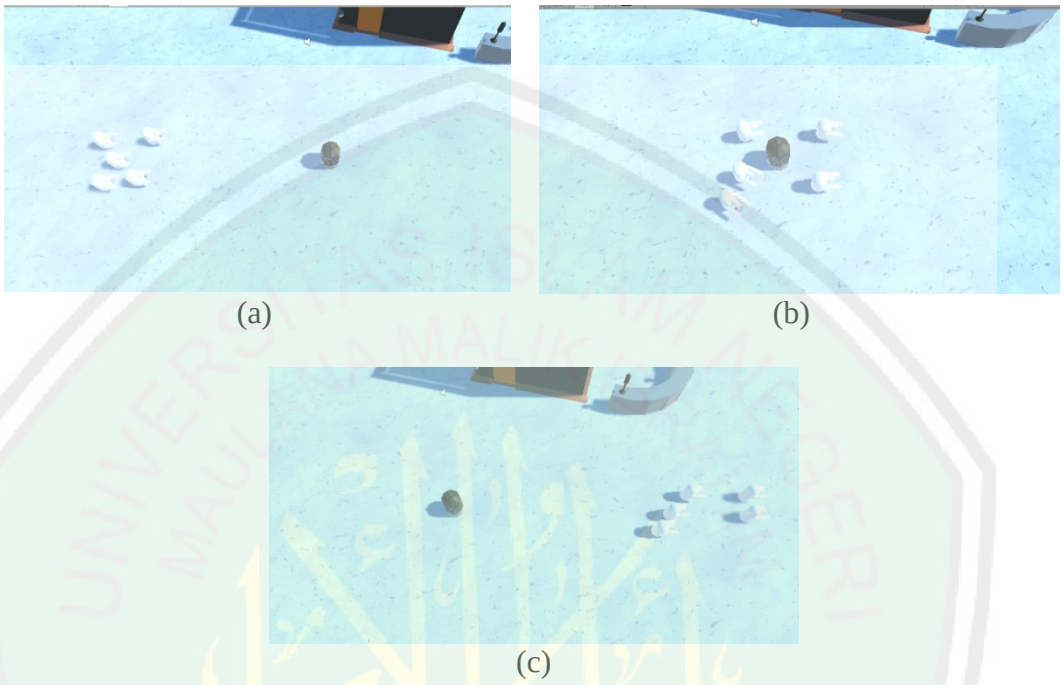
Pengujian *Obstacle Avoidance* ini dilakukan untuk melihat bagaimana proses menghindari *obstacle* yang dilakukan oleh *boids*. Pengujian ini dilakukan saat jemaah berada di titik–titik koordinat yang sudah ditentukan (lihat Gambar 4.7). Setiap jemaah memiliki kecepatan maksimal berjalan 2.0f dan setiap jemaah memiliki ambang batas radius 3.0f untuk mendeteksi *obstacle*. Lingkungan yang digunakan dalam pengujian ini ada halangan (*obstacle*) dan menggunakan 5 jemaah.



Gambar 4. 7 Contoh tata letak awal jemaah

Pengujian dilakukan untuk mengecek apakah algoritma *boids* telah berfungsi atau tidak sehingga 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 4.8).



(a) (b)

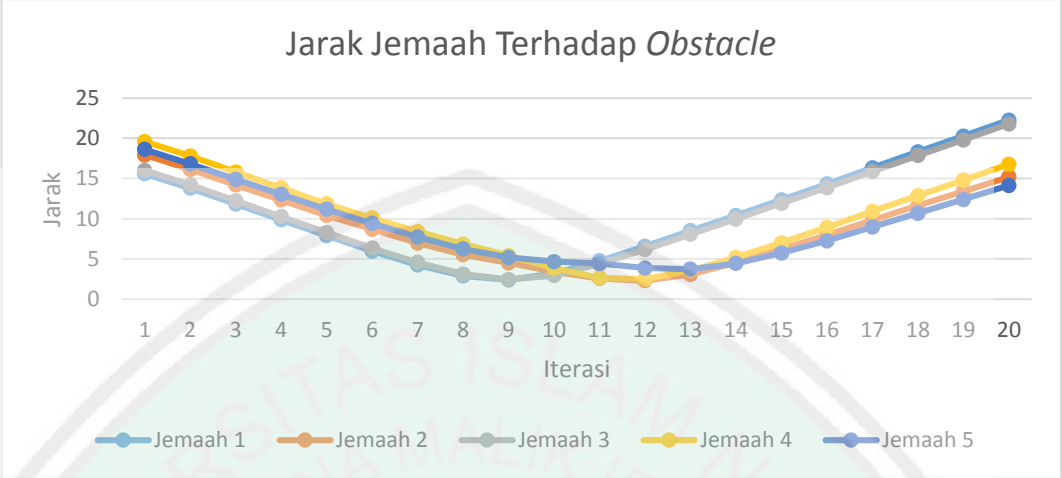
(c)

Gambar 4. 8 Uji coba pada iterasi ke-1(a), ke-9(b), dan ke-20(c).

Untuk mengetahui jemaah bertabrakan dengan *obstacle* atau tidak, kita menggunakan jarak *euclide* untuk setiap jemaah terhadap *obstacle*. Apabila jarak tersebut bernilai lebih dari 0, maka artinya jemaah tidak ada yang bertabrakan dengan *obstacle*.

Jika dilihat dari grafik jarak *obstacle* (lihat Gambar 4.9), berdasarkan tata letak posisi awal jemaah (lihat Gambar 4.8), maka saat jemaah menuju target, akan bertemu sebuah *obstacle*. Sehingga pada grafik jarak terlihat bahwa jarak jemaah yang awalnya jauh dari *obstacle* dan mendekati *obstacle* dengan masih menjaga jarak, kemudian setelah jemaah aman dari *obstacle*, maka jemaah akan

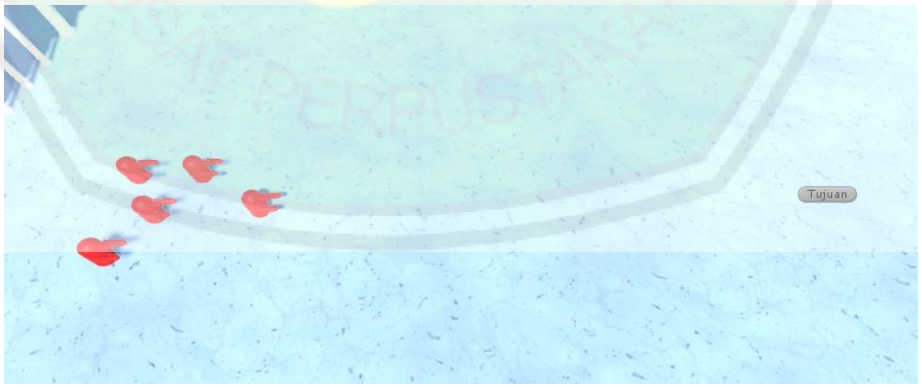
melanjutkan perjalanan menuju target. Setelah jemaah bebas dari *Obstacle,* jarak ke *obstacle* semakin lama semakin naik menjauh dari *obstacle*.



Gambar 4. 9 Grafik jarak jemaah terhadap *obstacle*

4.2 Pengujian Algoritma *Particle Swarm Optimization*

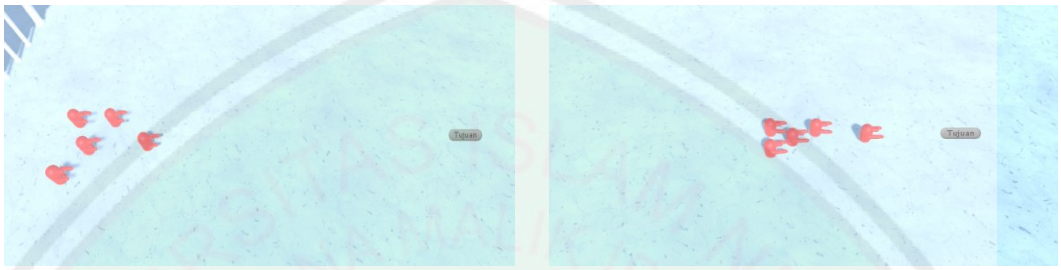
Pengujian ini dilakukan untuk melihat bagaimana jalannya aplikasi menuju target yang dilakukan oleh *Particle Swarm Optimization*. Pengujian ini dilakukan menggunakan 5 jemaah dan posisi awal jemaah berada pada titik–titik koordinat yang sudah ditentukan (lihat Gambar 4.10). Setiap jemaah memiliki kecepatan maksimal bergerak 1.0f.



Gambar 4. 10 Contoh tata letak awal jemaah dan target

Dengan algoritma *Particle Swarm Optimization* jemaah diharapkan dapat bergerak bergerombol menuju target yang telah ditentukan tanpa bertabrakan

dengan jamaah lainnya. Pengujian ini dilakukan untuk mengecek apakah algoritma telah berfungsi atau tidak sehingga jemaah dapat bergerak seperti burung menuju sebuah target. Pengujian dilakukan dengan jumlah iterasi sebanyak 30 kali dan hasilnya jemaah dapat bergerombol di setiap iterasi menuju target (lihat Gambar 4.11).

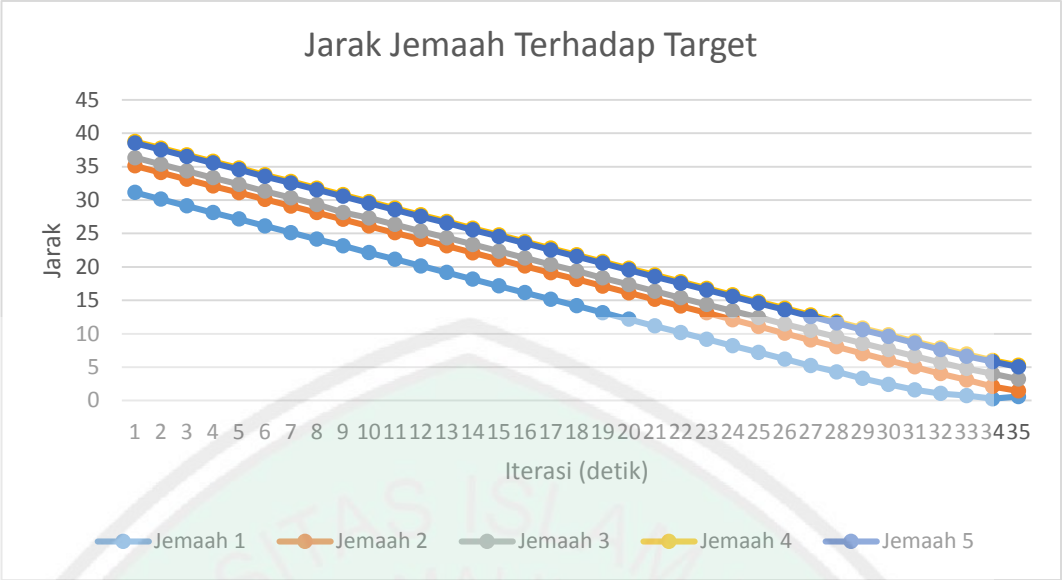


(a) (b)

(c)

Gambar 4. 11 Uji coba pada iterasi ke-1(a), ke-23(b), dan ke-30(c).

Dari gambar diatas didapat bahwa jemaah dapat bergerombol secara rapi layaknya burung dan jemaah bisa mencapai target yang dia tuju. Dari uji coba pun didapatkan jarak terhadap target yang terus menurun setiap dilakukan iterasi seperti yang terpampang pada grafik (lihat Gambar 4.12). Hal ini menerangkan bahwa di setiap iterasinya, jemaah telah berpindah menuju posisi yang lebih optimal dari posisi yang sebelumnya.



Gambar 4. 12 Grafik jarak jemaah terhadap target

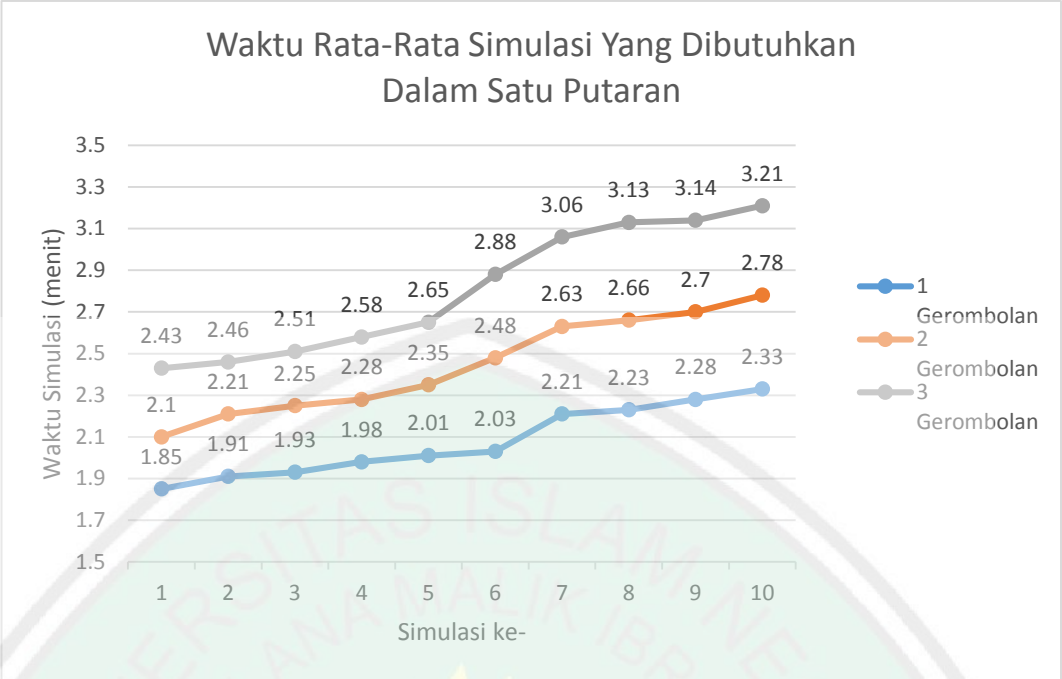
4.3 Pengujian Algoritma *Particle Swarm Optimization dan* Algoritma *Boids* pada Simulasi

Pengujian dilakukan untuk melihat apakah metode *Particle Swarm Optimization dan Boids* bisa bekerja secara baik atau tidak pada simulasi. Pengujian dilakukan saat jemaah berada pada titik–titik koordinat yang sudah ditentukan. Pengujian dilakukan 5 kali simulasi dengan 2 kali uji coba di setiap simulasinya. Pada setiap uji coba dibedakan pada jumlah gerombolan dan jumlah jemaahnya. Dari uji coba didapatkan juga rata-rata waktu yang dibutuhkan jemaah untuk melakukan 1 putaran thawaf dan presentasi jumlah jemaah yang berhasil melakukan putaran thawaf (lihat Tabel 4.3). Dari data yang didapatkan menunjukkan bahwa di setiap simulasi dan uji coba presentasi jemaah yang berhasil mengelilingi Ka'bah hasilnya lebih bagus menggunakan algoritma *Particle Swarm Optimization* daripada algoritma *Boids.*

Tabel 4. 3 Pengujian Algoritma *Particle Swarm Optimization* dan *Boids* 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	18	98	2.46
3	1	1	15	15	100	1.93
	2	2	30	30	100	2.25
	3	3	45	43	98	1.51
4	1	1	20	20	100	1.98
	2	2	40	39	99	1.28
	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	2.25

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 4.13). Pada grafik terlihat meningkatnya waktu yang diperlukan jemaah dalam melakukan 1 putaran thawaf dengan meningkatnya jumlah jemaah dan juga menurunnya waktu saat ada beberapa jemaah yang tidak berhasil mengelilingi Ka'bah.

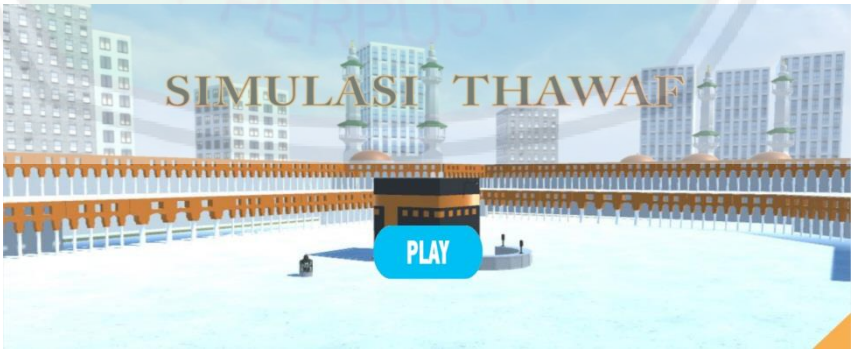


Gambar 4. 13 Waktu rata-rata simulasi dalam satu putaran

4.4 Implementasi Simulasi

Implementasi merupakan proses penerapan mekanisme suatu sistem. Mekanisme sistem ini dibangun berdasarkan desain dan rancangan yang sudah dibuat sebelumnya.

4.4.1 Tampilan Menu Awal



Gambar 4. 14 Tampilan Menu Awal

Tampilan menu awal merupakan tampilan yang akan pertama kali muncul saat aplikasi simulasi dimainkan. Penampilan menu dirancang dengan dua pilihan yaitu *play* : untuk menjalankan simulasi, dan *exit* : untuk keluar dari simulasi.

4.4.2 Tampilan Splashscreen

Gambar 4. 15 Tampilan *Splashscreen*

Tampilan *splashscreen* adalah tampilan yang muncul setelah tombol *play* pada menu awal diklik dan dia bertujuan memberi jeda waktu sebelum simulasi berjalan. *Splashscreen* muncul sekitar 3 detik.

4.4.3 Tampilan Simulasi pada Bagian Awal

Gambar 4. 16 Tampilan Simulasi pada Bagian Awal

Merupakan tampilan awal ketika simulasi ini dimainkan. Jemaah tersebar di posisi yang sudah ditentukan di sekitar area Masjidil Haram dan jemaah akan menuju Ka'bah untuk melekukan putaran pertama thawaf.

4.4.4 Tampilan Simulasi Saat Sejajar dengan Hajar Aswad

Gambar 4. 17 Tampilan Simulasi Saat Sejajar dengan Hajar Aswad

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

4.4.5 Tampilan Simulasi Saat Melakukan Putaran Thawaf

Gambar 4. 18 Tampilan Simulasi Saat Melakukan Putaran Thawaf

Tampilan simulasi ketika melakukan thawaf merupakan tampilan ketika jemaah sudah melewati Hajar Aswad. Di tampilan ini nantinya akan muncul konten yang berisi doa yang harus dibaca jemaah ketika melakukan thawaf, dimana bacaan dibaca mulai dari Hajar Aswad sampai Rukun Yamani. Di putaran pertama sampai putaran ketujuh pun juga ada bacaan doa yang berbeda dan konten tersebut dapat dihilangkan dengan meng-klik tombol "Doa".

4.4.6 Tampilan Simulasi Saat Sejajar dengan Rukun Yamani

Gambar 4. 19 Tampilan Simulasi Saat Sejajar dengan Rukun Yamani

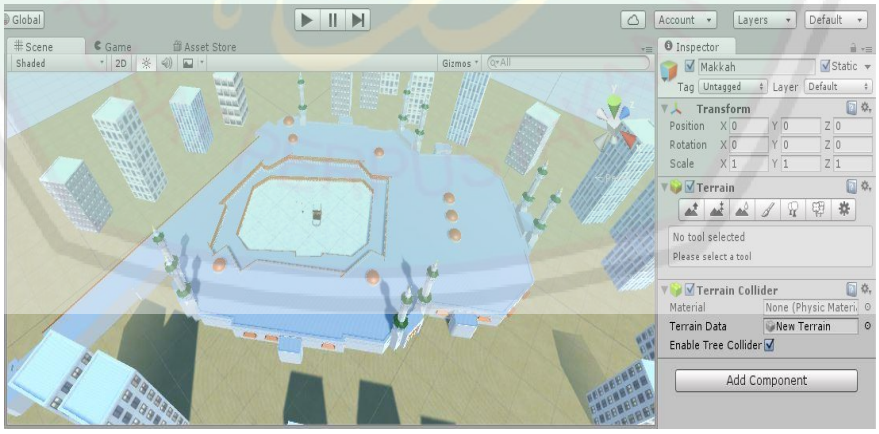
Tampilan simulasi ketika sejajar dengan Rukun Yamani merupakan tampilan ketika jemaah berada sejajar dengan Rukun Yamani. Pada simulasi ini nantinya akan muncul konten informasi mengenai apa yang harus dilakukan dan dibaca jemaah ketika berada sejajar dengan Rukun Yamani.

4.4.7 Tampilan Simulasi Saat Berjalan dari Rukun Yamani ke Hajar Aswad

Gambar 4. 20 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".

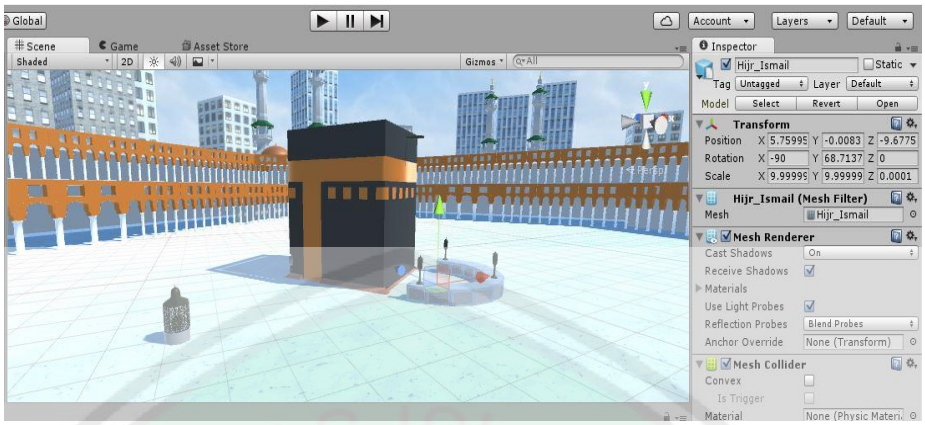
4.4.8 Pembangunan Enviroment



Gambar 4. 21 Tampilan Pembangunan Enviroment

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

4.4.9 Hambatan Statis padas Sekitar Area Ka'bah



Gambar 4. 22 Hambatan Statis padas Sekitar Area Ka'bah (Maqam Ibrahim dan Hijr Ismail)

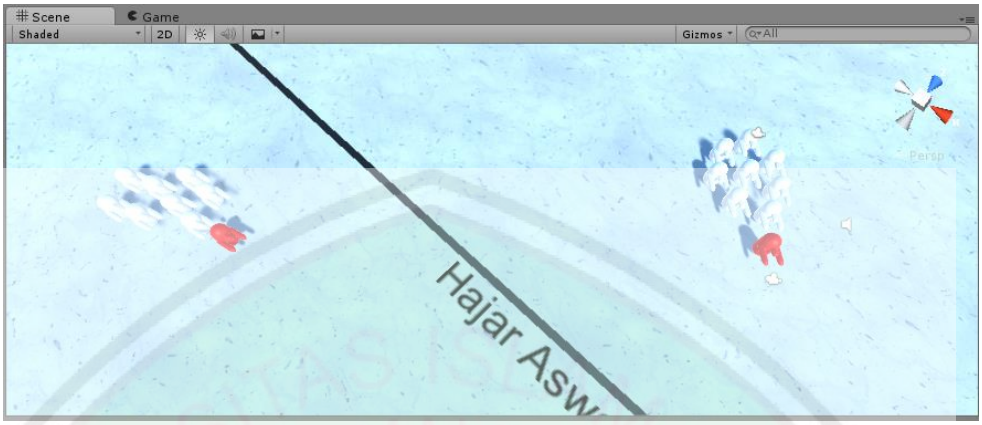
Hambatan statis pada sekitar area Ka'bah merupakan hambatan yang harus dihindari jemaah saat melewatinya. Hambatan–hambatan tersebut, yaitu Hijr Ismail dan Maqam Ibrahim.

4.4.10 Lintasan Jemaah saat Mengelilingi Ka'bah

Gambar 4. 23 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 *Particle Swarm Optimization*.

4.4.11 Jemaah



Gambar 4. 24 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.

4.5 Hasil Pembahasan

Dalam penelitian ini, ada tiga parameter yang dibandingkan penulis dengan peneliti sebelumnya. Tiga parameter yang dibandingkan yaitu pertama,*separation*, dimana jemaah tetap menjaga jarak agar tidak terjadi tabrakan antar jemaah yang lainnya. Kedua, *alignment*, dimana agar kecepatan jemaah tetap terjaga dengan jemaah yang lainnnya. Ketiga, *cohesion*, dimana jemaah tetap dengan jemaah yang lainnya.

Berikut tabel pengujian *separation* dari algoritma *particle swarm optimization* dan algoritma di penelitian sebelumnya.

Tabel 4.4 : Pengujian *Separation* Algoritma *Particle Swarm Optimization*

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

Tabel 4.5 : Pengujian *Separation* Penelitian Sebelumnya

Iterasi	Jarak Jemaah 1 Terhadap Jemaah Lainnya			
	Leader	Jemaah 2	Jemaah 3	Jemaah 4
1	15.51	7.13	8	10.18
2	14.25	5.45	6.75	7.67
3	12.09	5.03	6.54	9.56
4	11.39	3.96	6.78	8.92
5	10.10	4.20	5.98	8.10

Berikut tabel pengujian *alignment* dari algoritma *particle swarm optimization* dan algoritma di penelitian sebelumnya.

Tabel 4.6 : Pengujian *Alignment* Algoritma *Particle Swarm Optimization*

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

Tabel 4.7 : Pengujian *Alignment* Penelitian Sebelumnya

Iterasi	Jarak Jemaah 1 Terhadap Jemaah Lainnya			
	Leader	Jemaah 2	Jemaah 3	Jemaah 4
1	15.51	7.13	8	10.18
2	14.25	5.45	6.75	7.67
3	12.09	5.03	6.54	9.56
4	11.39	3.96	6.78	8.92
5	10.10	4.20	5.98	8.10

Berikut tabel pengujian *cohesion* dari algoritma *particle swarm optimization* dan algoritma di penelitian sebelumnya.

Tabel 4.8 : Pengujian *Cohesion* Algoritma *Particle Swarm Optimization*

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

Tabel 4.9 : Pengujian *Cohesion* Penelitian Sebelumnya

Iterasi	Jarak Jemaah 1 Terhadap Jemaah Lainnya			
	Leader	Jemaah 2	Jemaah 3	Jemaah 4
1	15.51	7.13	8	10.18
2	14.25	5.45	6.75	7.67
3	12.09	5.03	6.54	9.56
4	11.39	3.96	6.78	8.92
5	10.10	4.20	5.98	8.10

Berdasarkan hasil tabel pengujian dari tiga parameter diatas, dapat diketahui bahwa jumlah angka yang didapat dari *separation,alignment,cohesion* pada algoritma *Particle Swarm Optimization* (PSO) lebih kecil yang dimana lebih fleksibel dibandingkan angka pada penelitian sebelumnya. Selain itu juga metode

PSO ini dapat menghindari terjadinya tabrakan antar jemaah dan rombongan, jarak tetap optimal dimana tidak berada terlalu jauh ataupun terlalu dekat dengan jemaah atau rombongan lainnya. Sedangkan di metode sebelumnya masih ada terjadi tabrakan dan ada beberapa jemaah yang keluar dari gerombolan rombonga tersebut. Sehingga bisa diketahui, bahwa perhitungan algoritma *Particle Swarm Optimization* lebih tepat digunakan untuk masalah kerumunan dibandingan dengan metode yang sudah dilakukan peneliti sebelumnya. Oleh sebab itu, penulis mengembangkan penelitian sebelumnya dan tentunya juga dengan tambahan ilmu tata cara thawaf menurut islam, karena dalam pandangan islam, menyampaikan ilmu adalah sesuatu yang harus dilakukan oleh seorang muslim.

Dalam sabda 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)

Dari penjelasan diatas, menjadi langkah awal penelitian ini mempunyai konten bagaimana tata cara thawaf yang benar. Ketika *user* melihat simulasi ini, harapannya, secara otomatis akan menambah ilmu kepada *user* tentang bagaimana melakukan thawaf dengan baik dan benar karena banyaknya orang yang baru belajar / mengetahui tata cara thawaf hanya dari ketika mereka melakukan manasik haji tanpa tahu secara visual.

Penjelasan diatas menjadi alasan penulis melakukan penelitian dalam pembuatan media simulasi thawaf untuk membantu memberikan pengetahuan dasar tentang hal–hal yang berhubungan dengan thawaf secara animasi / visual.

4.6 Script Implementasi

4.6.1 Pathfinding

```
    public Transform[] path;
    public float speed = 2.0f;
    public float reachDist = 1.0f;
    public int currentPoint = 0;
    public float rotationSpeed = 3.0f;

    public Transform awal;

    private float xawal;
    private float zawal;
    private float yawal;

    private float xbaru;
    private float zbaru;

    private float xtarget;
    private float ztarget;

    private float D1;
    private float D2;

    private int count;

    float random_nilai_x;
    float random_nilai_z;

    Vector3 posisibaru;

    float dist;

    void Start()
    {
        xawal = awal.position.x; //Debug.Log("Xawal = "+xawa
l);
        zawal = awal.position.z; //Debug.Log("Zawal = "+zawa
l);
        yawal = awal.position.y;
    }
```

4.6.2 Kecepatan

```
public Rigidbody rb;
float speed;
float waktu = 0;
public int jemaah;

void Start()
{
    rb = GetComponent<Rigidbody>();

    StartCoroutine(prosesTask(1));
    StartCoroutine(prosesTask(2));

IEnumerator prosesTask(float wait) {
    yield return new WaitForSeconds(wait);
    speed = rb.velocity.magnitude;
    if (jemaah == 1) {
        Debug.Log("J 1 = " + speed.ToString("F"));
    } else if (jemaah == 2) {
        Debug.Log("J 2 = " + speed.ToString("F"));
    }
```

4.6.3 Leader Follow

```
var leftR = transform.position;
var rightR = transform.position;

leftR.x -= 2;
rightR.x += 2;

if(Physics.Raycast(leftR, transform.forward, hit, 10)){
    if(hit.transform != transform){
        Debug.DrawLine(leftR, hit.point, Color.red);
        dir += hit.normal * 5;
    }
```

4.6.4 Pathfollower

```
public Transform[] path;
public float speed = 5.0f;
public float reachDist = 1.0f;
public int currentPoint = 0;
public float rotationSpeed = 5.0f;
```

```
    void Start() {

    }

    void Update() {
        float dist = Vector3.Distance(path[currentPoint]
.position, transform.position);
        transform.position = Vector3.MoveTowards(transfo
rm.position, path[currentPoint].position, Time.deltaTime
 * speed);

    }
}
```

4.6.5 Jarak

```
    public Transform[] path;
    public int currentPoint = 0;
    public int jemaah;

    // Use this for initialization
    void Start () {
        StartCoroutine(prosesTask(0));
        StartCoroutine(prosesTask(1));
        StartCoroutine(prosesTask(2));
        yield return new WaitForSeconds(wait);
        float dist = Vector3.Distance(path[currentPoint]
.position, transform.position);
        if (jemaah == 1)
        {
            Debug.Log("Jemaah 1 = " + dist.ToString("F")
);
        }
        else if (jemaah == 2)
        {
            Debug.Log("Jemaah 2 = " + dist.ToString("F")
);
```

4.6.6 Jarak 1

```
    public Transform[] path;
    public int currentPoint = 0;
```

```
    public int currentPointa = 1;
    public int currentPointb = 2;
        StartCoroutine(prosesTask(0));
        StartCoroutine(prosesTask(1));
        StartCoroutine(prosesTask(2));
                Debug.Log("Jemaah_1 L = " + dist.ToStrin
g("F"));
        Debug.Log("Jemaah_1 2 = " + dista.ToString("F"))
;
        Debug.Log("Jemaah_1 3 = " + distb.ToString("F"))
}
```

4.6.7 Steering Behavior Crowd Alignment

```
    public class SteeringBehavior_CrowdAlignment : Steer
ingBehavior
    {
        #region Attributes

        // Adapt speed to crowd speed
        [Tooltip("If checked, this character adapt its o
wn speed to crow average velocity")]
        [SerializeField]
        private bool m_AdaptSpeedToCrowdSpeed = false;

            }
}
```

4.6.8 Steering Behavior Crowd Cohesion

```
    public class SteeringBehavior_CrowdCohesion : Steeri
ngBehavior
    {
        #region Attributes

        // Adapt speed to crowd speed
        [Tooltip("If checked, this character adapt its o
wn speed to crow average velocity")]
        [SerializeField]
        private bool m_AdaptSpeedToCrowdSpeed = false;

    }
}
```

4.6.9 Steering Behavior Crowd Separation

```
    public class SteeringBehavior_CrowdSeparation : Stee
ringBehavior
    {
        #region Attributes

        // Minimum neighbor hood unit count
        [Tooltip("Minimum number of units in neighbor ho
od to consider them as a crowd")]
        [SerializeField]
        private uint m_MinNeighborHoodUnitCount = 4;

        // Neighbor hood radius
        [Tooltip("Radius of zone detecting any crowd uni
t")]
        [SerializeField]
        private float m_NeighborHoodRadius = 6;

            }
}
```

4.6.10 Steering Behavior Leader Following

```
    public class SteeringBehavior_LeaderFollowing : Stee
ringBehavior
    {
        #region Attributes

        // Separation force scale
        [Tooltip("Scale separation force")]
        [SerializeField]
        private float m_SeparationForceScale = 1;

        // Leader ahead radius
        [Tooltip("Distance ahead leader where ahead poin
t will be placed")]
        [SerializeField]
        private float m_LeaderAheadPointRadius = 3;

    }
}
```

4.6.11 Steering Behavior Obstacle Avoidance

```
    public class SteeringBehavior_ObstacleAvoidance : St
eeringBehavior
    {
        #region Attributes

        // Layer mask for collision detection
        [Tooltip("Layers that will be used or ignored in
 obstacle avoidance")]
        [SerializeField]
        private LayerMask m_LayerMask;

        // Bounding sphere radius
        [Tooltip("A sphere of this radius will be used t
o anticipate a collision")]
        [SerializeField]
        private float m_BoundingSphereRadius = 1;
```

BAB V

PENUTUP

5.1 Kesimpulan

Dari penelitian yang sudah dilakukan terhadap simulasi kerumunan untuk pergerakan jemaah yang sedang melakukan thawaf, dapat disimpulkan bahwa dengan menggunakan metode *Particle Swarm Optimization* dan metode *Boids* untuk NPC jemaah telah telah dihasilkan perilaku yang halus yaitu perilaku yang fleksibel sesuai dengan keadaan lingkungan dari titik *waypoint* yang sudah ditentukan.

Berdasarkan pengujian yang sudah di uji coba, menunjukan bahwa jemaah tetap berada dekat dengan jemaah lain, tidak bertabrakan dengan rombongan lain dan hambatan statis di area Ka'bah. Jemaah juga terus berada pada titik *waypoint* yang sudah ditentukan selama menjalankan thawaf. Sedangkan hasil untuk rata-rata waktu yang diperlukan jemaah saat melakukan satu putaran simulasi akan meningkat dan juga bisa menurun dengan meningkatnya jumlah jemaah.

5.2 Saran

Simulasi ini tentunya masih banyak kekurangan dalam penelitian serta pembuatannya. Oleh karena itu, penulis menyarankan yang nantinya perlu dilakukan pengembangan, diantaranya:

1. Simulasi ini agar bisa dikembangkan pada *platform* android agar lebih mudah dinikmati oleh calon jemaah haji ketika belajar manasik haji dimana saja.

2. Penambahan karakter rombongan jemaah dan hambatan yang lebih banyak dan dari arah mana saja serta *visualisasi* didalam simulasi agar terlihat lebih nyata.

DAFTAR PUSTAKA

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.

J. Kennedy, R. Eberhart. 1942-1948. *Particle Swarm Optimization",* IEEE conference on *Neural Network*, pp. Perth, Australia, piscataway, NJ, IV, 1995.

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.

M. Ghanem. 2014. *Using Hybrid Artificial Bee Colony Algorithm and Particle Swarm Optimization For Training Feed-Forward Neural Networks*. 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.

Nurmaini. 2014. *Implementasi Prilaku Berkelompok pada Swarm Robots Menggunakan Teknik Logika Fuzzy-Particle Swarm Optimization*. Jurusan Teknik Elektro, Institut Teknologi Sepuluh November: Surabaya.

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

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

Wardani. 2015. *Penerapan Kecerdasan Kelompok untuk Penyelesaian Teka-Teki Sudoku dengan Metode Particle Swarm Optimization.* Teknik Elektro. ITS Surabaya