PENERAPAN *FLOCKING BEHAVIOR* UNTUK PERGERAKAN BERKELOMPOK *NON PLAYER CHARACTER* PADA *2D ENDLESS RUNNER GAME*

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Yosua
NIM: 155150207111078

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2019

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

PENGESAHAN

PENERAPAN *FLOCKING BEHAVIOR* UNTUK PERGERAKAN BERKELOMPOK *NON PLAYER CHARACTER* PADA *2D ENDLESS RUNNER GAME*

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Yosua
NIM : 155150207111078

Skripsi ini telah diuji dan dinyatakan lulus pada
15 Februari 2019
Telahh diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing 2
Eriq Muhammad Adams Jonemaro, S.T, M.Kom	Muhammad Aminul Akbar , S.Kom., M.T
NIP: 198504102012121001	NIK: 201607891010131001

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP: 197105182003121001

repository.ub.ac.id

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar referensi.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 14 Januari 2019

Yosua

NIM: 155150207111078

UNIVERSITAS BRAWIJAYA

PRAKATA

Puji dan syukur penulis penjatkan kepada Tuhan Yang Maha Esa atas berkat, tuntunan dan karunia-Nya sehingga penulis dapat menyelesaikan skripsi yang berjudul "Penerapan *Flocking Behavior* Untuk Pergerakan Berkelompok *Non Player Character* Pada *2D Endless Runner Game*" sebagai salah satu persyaratan untuk menyelesaikan studi di Jurusan Teknik Informatika, Fakultas Ilmu Komputer, Universitas Brawijaya.

Penulis menyadari bahwa skripsi dapat terselesaikan berkat bantuan, petunjuk, bimbingan dan dukungan dari berbagai pihak yang telah banyak membantu proses penyelesaian tugas akhir ini. Oleh karena itu penulis ingin menyampaikan terima kasih yang sebesar-besarnya kepada :

1. Orang tua penulis, Barita Saor Tua Silalahi dan Lanny Ros Mawar Hutagalung yang tak henti hentinya memberikan dukungan moril dan materil.
2. Adik penulis, Andreas Silalahi, Fongki, Eddie dan Shila yang telah memberikan motivasi dan semangat demi terselesaikannya skripsi ini.
3. Eriq Muhammad Adams Jonemaro, S.T, M.Kom selaku pembimbing I dan Muhammad Aminul Akbar , S.Kom., M.T selaku pembimbing II yang telah membimbing dan mengarahkan penulis sehingga dapat menyelesaikan skripsi ini.
4. Seluruh dosen Fakultas Ilmu Komputer Universitas Brawijaya atas kesediaan membagi ilmunya kepada penulis sehingga penulis dapat menyelesaikan skripsi ini.
5. Sahabat Kelompok Tumbuh Bersama yaitu Sarah Yuli Simarmata dan Ilham Harazki yang selalu memberi dukungan , semangat dan pembelajaran pertumbuhan iman selama di FILKOM
6. Teman teman Slathem Studio yang selalu memotivasi dan memberi semangat kepada penulis di dalam penyelesaian skripsi ini
7. Ibu dari teman penulis Mami Jhon selaku penyedia tempat untuk mengerjakan skripsi.
8. Keluarga Komisi 4 2018 yang selalu memberi semangat hingga sampai menyelesaikan skripsi ini.
9. Teman teman Lindisanti Squad yang selalu menghibur, memberi motvasi dalam penyelesaian skripsi selama ini
10. Teman teman PMK Daniel yang selalu menghibur, memberikan motivasi serta rasa kekeluargaan selama ada di kota Malang.
11. Teman teman seperjuangan Fakultas Ilmu Komputer angkatan 2015 yang telah memberikan bantuan selama masa studi hingga penyelesaian skripsi ini
12. Pesenkopi yang telah memberi semangat hingga skripsi ini bisa selesai

Penulis sadar bahwa dalam penyusunan skripsi in masih banyak kekurangan, sehingga saran dan kritik yang membangun dapat disampaikan secara langsung untuk pengembangan dan penelitian selanjutnya.

Malang, 14 Januari 2019

Penulis
yosilahi10@gmail.com

repository.ub.ac.id

ABSTRAK

Yosua, Penerapan *Flocking Behavior* Untuk Pergerakan Berkelompok *Non Player Character* Pada *2D Endless Runner Game*

Pembimbing: Eriq Muhammad Adams Jonemaro, S.T, M.Kom dan Muhammad Aminul Akbar , S.Kom., M.T

2D *Endless Runner Game* merupakan game dimana seorang player akan bergerak maju terus menerus, dimana pada game ini tidak memiliki titik akhir dan juga memiliki rintangan yang membuat player menjadi tertantang untuk bermain. Rintangan yang ada pada beberapa game Endless ada beberapa macam, untuk penelitian ini game yang dibuat akan memiliki rintangan, mulai dari obstacle dan pergerakan kelompok. Pergerakan Kelompok dalam game endless ini memiliki rintangan dimana ada lawan yang berjumlah banyak akan bergerak secara bersamaan untuk menyerang player. Untuk kelompok yang dapat menyerang player ini akan diterapkan pada NPC yang bergerak secara bersamaan. Untuk pergerakan berkelompok sendiri salah satu nya yaitu Flocking. Flocking merupakan teknik paling popular dari kecerdasan buatan yang dapat menggerakan suatu kelompok.Dalam pergerakan flocking kadang memiliki saat dimana pergerakan tersebut akan tertahan maka dari itu akan di gunakan pathfinding yaitu A* agar pergerakan kelompok tidak tertahan. Dari permesalahan tersebut peneliti akan membuat game dengan pergerakan kelompok lawan yang memiliki Flocking Behavior. Hasil dari penelitian ini menunjukkan bahwa flocking dapat diterapkan di pergerakan berkelompok dan FPS yang dihasilkan berpengaruh terhadap jumlah NPC dalam kelompok, hasil ini dibuktikan dengan pengujian 3 NPC menghasilkan FPS berkisar 43,6 dan dengan 8 NPC menghasilkan FPS berkisar 33,7. NPC juga berhasil sampai ke tujuan tanpa ada yang terhambat.

Kata kunci: *2D Platform, Endless Runner Game, Flocking Behavior, NPC*

UNIVERSITAS BRAWIJAYA

repository.ub.ac.id

ABSTRACT

Yosua, ***Flocking Behavior Use for Non Player Character's Group Movement on 2D Endless Runner Game***

Supervisors: Eriq Muhammad Adams Jonemaro, S.T, M.Kom and Muhammad Aminul Akbar , S.Kom., M.T

2D Endless Runner Game is a game where the player will keep on moving forward thus no end point and will present obstacles for the player and keep the player's adrenaline driven to keep on playing. The obstacles on some of Endless game differs, for this research, the game which will be made will have differently obstacles such as obstacle and group movement. The game will have obstacle where there will be many opponents moving as a group to fight the player's character. The group of assailants which will fight the player, will be applied on simultaneously moving NPC. Flocking is the most famous methods from artificial intelligence which moves a group.Flocking's movement sometimes has this restrained moment, therefore, applied one of the pathfinding method that is A to make the group's movement not be restrained. Based on this problem, the researcher will develop a game which will have assailant's group have the Flocking Behavior. The results of this study indicate that flocking can be applied in group movements and the resulting FPS affects the number of NPCs in the group This result was proven by testing 3 NPCs to produce FPS ranging from 43.6 and with 8 NPCs producing FPS around 33.7. NPCs also make it to the destination without being blocked.*

Keywords: *2D Platform, Endless Runner Game, Flocking Behavior, NPC*

UNIVERSITAS BRAWIJAYA

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1 PENDAHULUAN

Bab satu menerangkan bagaimana awal dan penelitian ini akan berjalan. Bab ini akan menjadi acuan untuk bab berikuktnya. Bab ini akan berisi tentang apa alasan penulis melakukan penelitian, hal apa yang ingin di capai, dan sistematika penulisan

1.1 Latar belakang

2D Platformer Game merupakan game yang sudah popular dari jaman abad 20an, dimana pertama kali nya game permainan yang membawa tema ini adalah Super Mario, dimana player dapat membuat karakater berjalan melewati berbagai halangan untuk sampai ke titik akhir. Bisa dibilang Super Mario merupakan salah satu awal dari game *2D Endless Runner* tapi tidak bisa dibilang Endless Runner karena pada game ini memiliki titik final dalam permainannya atau sesuatu yang harus dicapai oleh pemain.

Game Endless Runner awalnya dipelopori oleh "Canabalt" dimana Canabalt ini membuat player untuk melewati halangan yang sudah di buat, seperti batu yang berjatuhan, peledak yang jatuh ke arah player. Uniknya game ini tidak ada sesuatu yang dicapai seperti batas akhir ataupun goal yang ingin dicapai dalam game tersebut. Jadi player harus terus bermain hingga bisa mencapai angka teratas.

Endless Runner dapat diartikan sebagai permainan yang akan membuat player berlari sejauh mungkin dan tidak memiliki titik akhir dan memiliki rintangan. Halangan dalam permainan ini akan beraneka ragam.

Permainan yang akan dibangun ini akan memiliki beraneka ragam halangan, bisa seperti kotak kecil yang menghalangi player, benda tajam yang akan menghalangi hingga membuat player kalah, dan juga musuh yang akan menghalangi player dalam berlari. Untuk musuh yang menghalangi pemain ini akan dibuat beberapa individu yang membentuk kerumunan dan terbang mengikuti pemain dan berusaha untuk mengahalangi pemain dalam bermain . Untuk kerumunan musuh yang akan menghalangi pemain ini akan di implementasikan ke *Non Player Character*, dimana *Non Player Character* ini akan bergerak menuju pemain yang secara bersamaan satu dengan yang lain. Dari permainan Endless yang ada selama ini pemain hanya akan di hadapakan dengan rintangan seperti objek yang tidak bergerak, serangan yang hanya bergerak satu arah seperti pada game Jetpack Joyride, dimana serangan roket yang bergerak hanya satu arah sehingga pemain dengan mudah dapat membaca pergerakan tersebut sehingga pemain dapat menghindari roket tersebut. Pada permainan seperti Temple Run yang memiliki konsep sama tapi dalam bentuk 3D juga ada batu yang hanya bergerak satu arah untuk menyerang pemain. Pada penelitian yang pernah dilakukan oleh Zhang Yancan, game yang dibuat oleh beliau menggunakan satu *enemy* yang bergerak sendiri dan dalam *game* ini akan mengejar player tapi hanya sampai range yang dapat dicapai, jika sudah

melewati range maka *enemy* tidak akan mengejar *player*(Yancan & Technology, 2016). Dalam penelitian yang dilakukan dilakukan pada *Endless Runner Game* yang di ciptakan oleh Pedersen, di *game* ini diterapkan kelompok lawan yang akan menyerang pemain tapi kelompok lawan ini hanya diam dan menambak pemain. Lawan dalam game ini tidak mengejar player yang bisa bergerak ke atas dan ke bawah(Pedersen, 2014). Maka dari itu akan diterapkan pergerakan berkempok *Non Player Character* pada 2D Endless Runner Game ini.

Flocking merupakan salah satu cara untuk membuat beberapa individu yang menjadi kerumunan dapat bergerak secara bersamaan atau berkelompok(Prasetyo, Muh, Jonemaro, & Akbar, 2017). *Flocking Behavior* juga memberikan hal yang penting pada aspek game dimana flocking diyakini dapat memberikan pergerakan yang realistis dalam *game* dan memberikan hasil yang tepat untuk melakukan suatu pergerakan berkelompok(Larsson, Lundgren, Larsson, & Lundgren, 2017)

Pegerakan berkelompok dapat di implementasikan dengan menggunakan Flocking. Flocking juga diterapkan pada pergerakan *Non Player Character* berkelompok di permainan. Craig Reynolds adalah pencipta pegerakan tersebut pada sekitar tahun 80an. Setelah itu tidak sedikit dari orang orang yang menerapkan *Flocking Behavior* pada permainan yang lain. Berdasarkan hasil penelitian yang sudah dilakukan juga dengan *Flocking* ini bahwa *Flocking* ini dapat mengatur dalam pergerakan berkelompok untuk mencapai seuatu tujuan(Dewi, Hariadi, & Purnomo, 2011).Pada Flocking ini kadang memiliki waktu dimana individu yang bersamaan bergerak ini akan terhenti pergerakannya, alhasil akan di terapkan A Star dalam penelitian ini agar individu yag bergerak tidak terhenti pada suatu waktu.

Sesuai dengan penjelasan yang sudah penulis jelaskan maka dengan ada nya Flocking ini pergerakan kumpulan individu ini akan menjadi tidak tertahan dan menambah halangand alam bermain permainan ini adapun judul penelitian ini adalah “Penerapan *Flocking Behavior* Untuk Pergerakan Berkelompok *Non Player Character* Pada *2D Endless Runner* Game”.

1.2 Rumusan masalah

1. Bagaimana menerapkan Flocking Behaviour pada pergerakan berkelompok *Non Player Character* untuk melakukan suatu pergerakan pada Endless Runner Game?
2. Bagaimana kinerja *Flocking* pada pergerakan berkelompok *Non Player Character 2D Endless Runner* Game?

1.3 Tujuan

Terwujudnya implementasi *Flocking* pada pergerakan berkelompok *Non Player Character 2D Endless Runner* Game

1.4 Manfaat

a. Bagi Peneliti

1. Mengimplementasikan wawasan yang dimiliki ke dalam pengembangan game
2. Memahami implementasi Flocking Behaviour untuk pergerakan berkelompok *Non Player Character* pada *2D Endless Runner* game

b. Bagi Pembaca

1. Mengerti bagaimana pergerakan berkelompok Non Player Charater dengan *Flocking Behavior*

1.5 Batasan masalah

Skripsi ini menekankan pada penerapan *Flocking Behavior* pada pergerakan perkelompok pada game *2D Endless Runner* Game dengan ketentuan :

1. Game berjenis Endless Runner 2D Platformer
2. Implementasi menggunakan Unity3D
3. *Flocking Behavior* terapkan pada map cenderung sederhana dan tidak kompleks

1.6 Sistematika pembahasan

BAB 1 PENDAHULUAN

Bab satu menerangkan bagaimana awal dan penelitian ini akan berjalan. Bab ini akan menjadi acuan untuk bab berikuktnya. Bab ini akan berisi tentang apa alasan penulis melakukan penelitian, hal apa yang ingin di capai, dan sistematika penulisan Flocking Behaviour untuk pergerakan kelompok *2D Endless Runner* Game

BAB 2 LANDASAN KEPUSTAKAAN

Menejelasakan tentang literature yang menjadi acuan pada paneilitian ini dan literature yang sudah ada sebelumnya, dengan tujuan membantuk penulisan Flocking Behaviour untuk pergerakan kelompok *2D Endless Runner* Game

BAB 3 METODOLOGI

Menjelaskan scenario yang akan dilakukan dalam mengimplementasikan Flocking Behaviour untuk pergerakan kelompok *2D Endless Runner* Game

BAB 4 PERANCANGAN

Menjelasakan bagaimana rancangan penerapan Flocking Behaviour pergerakan berkelompok *Non Player Character* pada *2D Endless Runner* Game

BAB 5 IMPLEMENTASI

Menjelaskan bagaimana cara menerapakan Flocking Behaviour untuk pergerakan kelompok *2D Endless Runner* Game

BAB 6 PENGUJIAN DAN ANALISIS

Memaparkan hasil dari implementasi yang sudah dilakukan sebelumnya, dan akan melihat hasil dari implementasi tersebut dan menilai ny

BAB 7 PENUTUP

Memaparkan hasil rangkuman dari penelitian yang sudah dilkaukan dan juga saran dari Flocking Behaviour untuk pergerakan kelompok *2D Endless Runner* Game

BAB 2 LANDASAN KEPUSTAKAAN

Bab ini menjelaskan teori yang digunakan oleh peneliti, seperti pengertian mengenai judul yang diangkat. Dalam bab ini juga terdapat berbagai sumber referensi penulis dalam melakukan penelitian ini.

2.1 2D Platform

Game 2D Platform adalah game dimana pemain dapat melewati berbagai halangan dan melompat antar platform untuk sampai ke titik akhir permainan(Sagala, Jonemaro, & Wardhono, 2017). Contoh game platformer yang terkenal adalah Super Mario Bros, dimana pemain mengendalikan karaker Mario melewati platform yang ada untuk sampai pada titik akhir.

Gambar 2.1 Super Mario Bros

Sumber : mario.nintendo.com(Nintendo,2018)

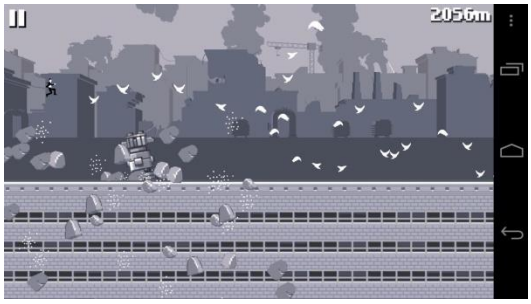
Di Game 2D Platformer juga biasanya player akan dihadapi dengan berbagai rintangan yang akan menghalangi player untuk mecapai titik akhir dari game, maka dari itu player harus bisa melewati berbagai rintangan agar dapat mencapai titik akhir.Jadi bisa dikatakan bahwa 2D Platform merupakan konsep *game* dimana membuat player untuk mengendalikan karakternya di berbagai platformer berbentuk 2D(Bhosale, Kulkarni, & Patankar, 2018)

Ada beberapa jenis permainan 2D, yang sering dipakai yaitu 2D Side-Scrolling, 2D Side Scrolling merupakan game yang dimana kameranya menampilkan sisi samping dari game yang dimainkan (Robert & Tihomir, 2018). Maka nanti game yang akan dibuat akan menerapkan 2D Side-Scrolling

2.2 Endless Runner Game

Permainan dengan jenis ini memiliki prinsiap dimna permain akan berusaha secara maksimal untuk mendapatkan skor tertinggi tanpa ada nya titik akhir. Endless Game banyak sekali implementasi nya dan yang di lakukan dalam penelitian ini adalah Endless Runner dimana konsep nya sama dengan Endless Game tapi player disini harus mengendalikan karakter berlari (Supervisors, Paraschakis, Mihailescu, & Eriksson, 2016).

Konsep 2D Platform dan Endless Runner pertama kali diluncurkan dengan nama “Canabalt”, game ini keluar tahun 2009,dimana player harus membuat karakter melompat dari satu gedung ke gedung lain, bukan hanya itu player juga harus melewati berbagai rintangan yang ada.



Gambar 2.2 Canabalt

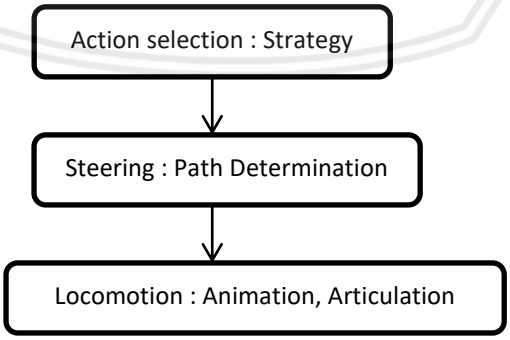
Sumber : fogstudios.com(Fog Studios,2018)

2.3 *Non Player Character* (*NPC*)

Non Player Character atau yang biasa disingkat dengan istilah *NPC* merupakan agen yang dapat diberikan sesuatu agar dapat berperilaku sesuai yang dinginkan. Dalam perkembangan *game* saat ini sudah banyak bermacam bentuk dari *NPC* mulai dari *monster*, manusia, binatang, dan berbagai bentuk lainnya. *NPC* dapat dikatakan juga sebagai player yang terlihat menyerupai player dan dapat berperilaku sesuai perintah yang diberikan. Perintah yang dimasudkan disini yaitu dapat berupa Kecerdasan Buatan(Warpefelt, 2016)

Non Player Character memiliki peran penting dalam berbagai *game*, bisa sebagai pelengkap dalam menambah keunikan dalam *game,* dan bisa juga sebagai salah satu rintangan ataupun lawan dalam sebuah game.

Perilaku *NPC* dibagi menjadi beberapa lapisan(Arif, Kurniawan, & Nugroho, 2011). Lapisan nya yaitu :



Dalam penelitian ini lebih difokuskan pada bagian Action Selection dimana akan dibuat strategi pergerakan kelompok *NPC* dengan menerapkan *Flocking Behavior*

2.4 Flocking Behaviour

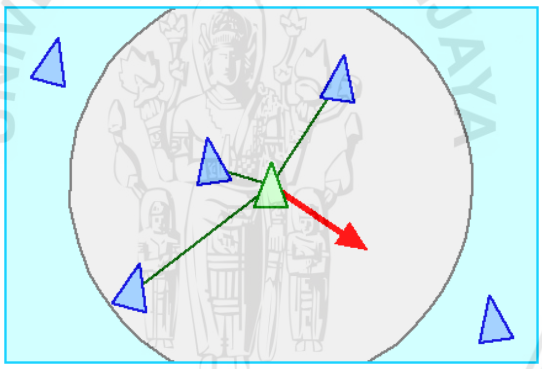
Flocking ini adalah teknik yang popular dari *Artificial Intelligence* dalam permainan untuk memandu individu yang banyak dalam melakukan gerakan yang digagas oleh Reynold(Reynolds, 1987) Pergerakan ini juga sudah banyak diimplementasikan dalam berbagai permainan komputer. DImana flocking ini juga sudah di terapkan pada game *Half Life* untuk pergerakan berkelompok lawan(Nugroho & Hariadi, 2014).

Flocking ini awalnya berasal dari pergerakan burung dan ikan secara berkelompok untuk menghindari predator, mencari makan, dan sebagainya. Lalu pada akhirnya konsep pergerakan ini di implementasikan ke dalam aspek computer yaitu dalam menggerakan agen(Girdhar, 2015)

Flocking Behavior memiliki beberapa prinsip, adalah :

1. Seperation (Pemisahan)

Memandu individu untuk tetap bergerak dengan syarat menjaga jauh antar individu

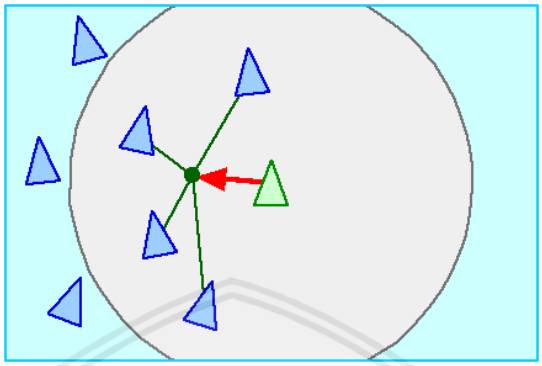


Gambar 2.3 Seperation

Sumber : red3d.com

2. Cohesion(Kohesi)

Memandu agen agar berada tidak jauh dari point tengah yang sudah ditentukan.

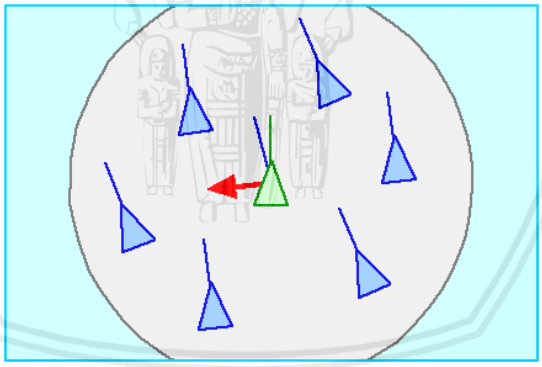


Gambar 2.4 Cohesion

Sumber : red3d.com

3. Alignment (Penyesuaian)

Menggerakan individu individu untuk tetap berada pada jalurnya sesuai dengan individu yang lain.



Gambar 2.5 *Alignment*

Sumber : red3d.com

Dalam penelitian ini akan lebih difokuskan flocking dengan *Alignment* karena flocking agan bergerak secara bersamaan untuk mencapai tujuan yang sama.

Alignment

Alignment

```
   // For every nearby boid in the system, calculate the average
velocity
     PVector align (ArrayList<Boid> boids) {
       float neighbordist = 50;
       PVector sum = new PVector(0, 0);
       int count = 0;
       for (Boid other : boids) {
         float d = PVector.dist(position, other.position);
         if ((d > 0) && (d < neighbordist)) {
           sum.add(other.velocity);
           count++;
         }
       }
       if (count > 0) {
         sum.div((float)count);
         // First two lines of code below could be condensed with
new PVector setMag() method
         // Not using this method until Processing.js catches up
         // sum.setMag(maxspeed);
         // Implement Reynolds: Steering = Desired - Velocity
         sum.normalize();
         sum.mult(maxspeed);
         PVector steer = PVector.sub(sum, velocity);
         steer.limit(maxforce);
         return steer;
       }
       else {
         return new PVector(0, 0);
       }
     }
```

Tabel 2.1 *Alignment Pseudocode*

2.5 Algortima A Star

A Star ini adalah salah satu algoritma penemuan jalur yang mengkalkulasikan titik agen dengan titik akhir yang akan dituju dan juga garis tarik lurus antar awal dengan akhir.

A Star merupakan metode pathfinding yang di garansikan dapat menumukan jalur dari awal menuju tujuan dengan cepat, dan A Star juga merupakan pathfinding yang sering digunakan dalam *game* computer(Cui & Shi, 2011)

Dimana algoritma A Star akan mencari titik tercepat dalam bergerak dalam mencari target yang sudah ditentukan.

A Star

OPEN // set of nodes to be evaluated
CLOSED// set of nodes already evaluated
add the start node to OPEN
 	
loop
  cureent = nnode in OPEN with the lowest f_cost
  remove current from OPEN
  add current to CLOSED
 	
if current is the target node //path has been found
  return
 	
foreach neighbor of the current node
  if neighbor is not traversable or neighbout is in CLOSED
  skip to the next neighbor
 	
if new path to neighbor is sorter OR neighbor is n ot in OPEN
  set f_cost of neighbour
  set parent of neighbor tp current
  if neighbor is not in OPEN
  add neighbor to OPEN

Tabel 2.2 A* Pseudocode

BAB 3 METODOLOGI PENELITIAN

Bab 3 menjelaskan langkah langah yang akan dilakukan dalam penyususunan skripsi. Metode penelitian yang digunakan adalah :

3.1 Studi Literatur

Pada bagian ini peneliti akan mencari berbagai wawasan mengenai hal hal dasar dalam merancang pergerakan berkelompok. Pencarian akan bersumber dari penelitian yang sudah ada sebelumnya dimana akan dicari hal hal yang berkaitan dengan penelitian. Hal hal tersebut adalah : *Flocking Behavior*, *2D Endless Runner* Game, *NPC*

3.2 Perancangan Pergerakan Kelompok *NPC* dengan *Flocking Behavior*

Dalam merancang langkah kerja dari sistem yang akan dibangun maka digunakan perancangan sistem. Tahapan dari perancangan dari game ini adalah akan membuat tampilan game lalu akan memulai proses koding dengan menggunakan Unity3D dengan bahasa pemrograman C#

3.2.1 Perancangan *NPC*

Untuk perancangan *NPC*, *NPC* akan ditambahkan property *NPC* yang digunakan dalam game ini akan berbentuk burung yang berbentuk sederhana dimana NPC burung ini akan di melakukan pergerakan Flockng Behavior Alignment

3.2.2 Perancangan *Flocking Behavior Alignment*

Perancangan Flocking pada *NPC* ini akan diterapkan pergerakan secara *Alignment* dimana nanti akan dibuat leader, dimana leader tersebut akan memiliki perilaku Flocking Behavior Alignment, lalu NPC yang lain akan dijadikan tetangga dimana ada radius yang dimiliki leader jika NPC yang menjadi tetangga sudah berada di radius tersebut maka akan mengikuti gerakan leader.

3.3 Implementasi Pergerakan Kelompok *NPC* dengan *Flocking Behavior*

Pada tahap implementasi, akan dilakukan proses kodingan ke Unity3D, kodingan yang akan di implementasikan juga ada diambil screenshoot yang menunujukan penerapan *Flocking Behavior*. Implementasi pertama kali yang dilakukan adalah membuat karakter *NPC* yang tadi sudah dirancang property apa saja yang akan di implementasikan.

Setelah terbuat karakter *NPC*, maka akan diimplementasikan kode untuk pergerakan Flocking pada *NPC*.

3.4 Pengujian dan Analisis Pergerakan Kelompok *NPC* dengan *Flocking Behavior*

Untuk tahap pengujian, peneliiti menguji penerapan *Flocking Behavior* pada pergerakan kelompok *NPC*, apakah *NPC* yang tadi sudah diimplementasikan sudah bergeraka secara berkelompok dengan benar, lalu setelah itu akan dilakukan pengujian atas kinerja pergerakan flocking, pengujian ini akan dilakukan beberapa scenario :

3.4.1 Pengujian FPS

Pengujian ini akan dilakukakn dengan menempatkan beberapa *NPC* untuk pergerakan berkelompok untuk pengujian 3 kali pengujian, yang pertama akan di tempatkan 3 *NPC* lalu pengujian lalu untuk pengujian kedua akan di tempatkan 5 *NPC* dan pengujian ketiga 8 *NPC*, lalu akan disajikan dalam tabel 3.1

Pengujian ini dilakukan untuk mengetahui apakah dengan adanya pergerakan berkelompok Flocking Behavior oleh NPC, FPS yang dihasilkan.

Pengujian ke	Jumlah *NPC*	FPS
1	3	
2	5	
3	8	

Tabel 3.1 Perancangan Pengujian FPS

3.4.2 Pengujian Validitas

Pengujian ini dilakukan untuk mengetahui individu kelompok yang melakukan Flocking Behavior berhasil sampai ke tujuan yang sudah di tentukan

No	Jumlah NPC	Hasil yang Diharapkan	Hasil	Status
1	3	Non Player Character tiba ke player		
2	5	Non Player Character tiba ke player		
3	8	Non Player Character tiba ke player		

Tabel 3.2 Perancangan Uji Validitas

Tabel 3.2 merupakan table yang berisi apakah NPC dengan jumlah yang ditentukan dapat sampai ke tujuan yang sudah di tentukan.

BAB 4 PERANCANGAN

4.1 Perancangan *Non Player Character*

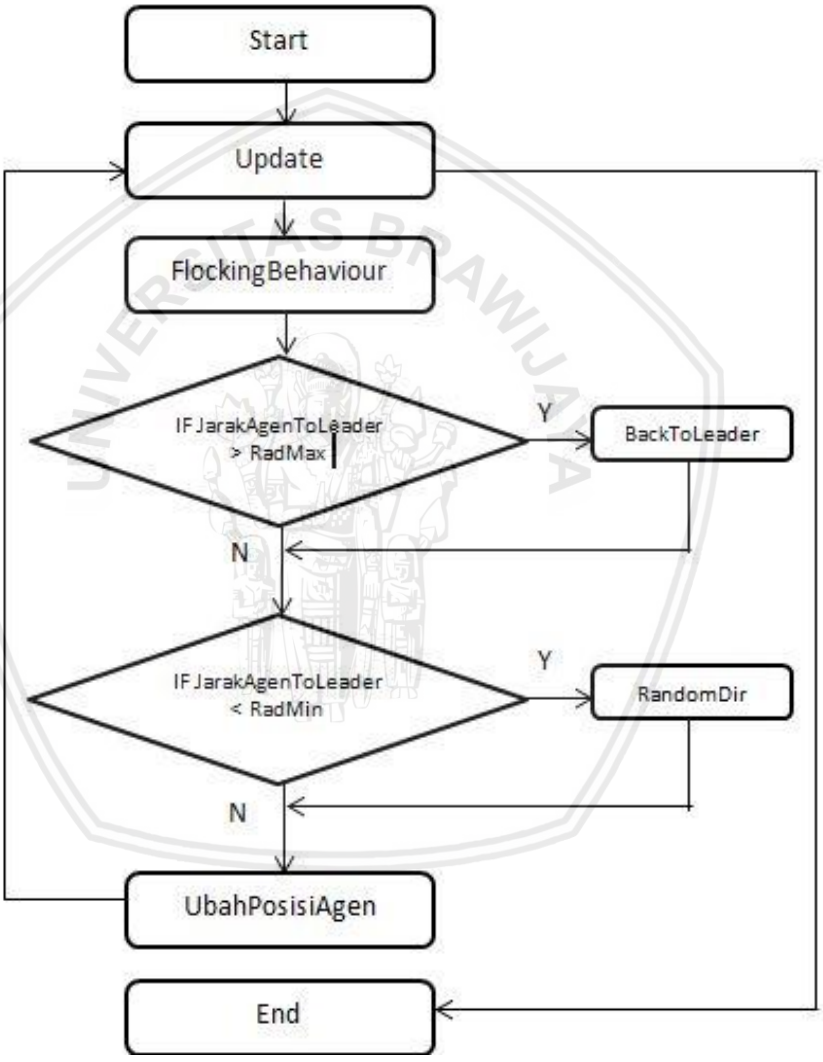
Pada bagian ini akan dirancangan *NPC* untuk melakukan pergerakan Flocking Behavior Alignment, *Non Player Charcater* akan dirancang berbentuk kawanan burung, dimana agen atau individu yang ada di dalam kelompok yang akann melakukan pergerakan berkelompok akan berbentuk burung berwarna hitam,dimana satu *Non Player Character* ini akan membuat kelompok dengan bergerak secara berkemlompok dengan individu yang lain, rancangan *Non Player Character* akan berbentuk seperti pada Gambar 4.1

Gambar 4.1 Rancangan Non Player Character

4.2 Perancangan *Flocking Behavior Penyesuaian*

Pada bagian ini akan dilakukan perancangan untuk pergerakan *NPC* dalam melakukan pergerakan berkelompok. Untuk Flocking yang diambil adalah Alignmet. Dimana Aligment ini merupakan pergerakan berkelompok dimana beberapa individu dalam kelompok akan bergerak bersama menuju arah yang sama.

Flocking Behavior Aligment memiliki alur kerja seperti terdapat pada flowchart dibawah ini



Gambar 4.2 Perancangan *Flocking Behavior* Aligment

Flowchart diatas memaparkan bagaimna pergerakan berkelompok akan dilakukan sesuai dengan konsep Flocking Penyesuaian(Alignment).

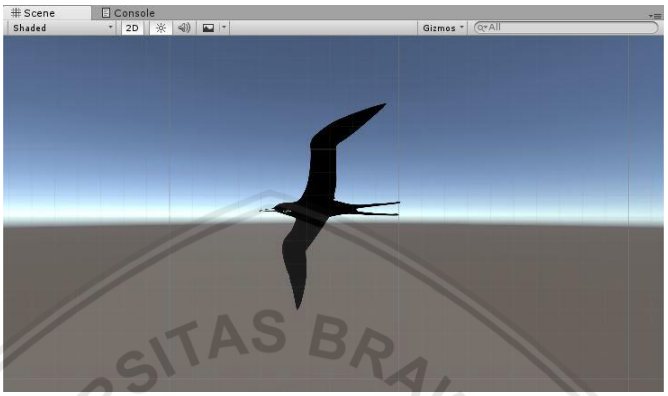
Dalam flowchart tersebut jika poisisi individu ke pimpinan agen lebih besar dari radius besar yang dimiliki pemimpin maka individu akan kembali posisi pimpinan.

Jika posisi individu lebih kecil dari radius kecil pimpinannya maka individu membuat movement random di dalam radisu kecil yang dimiliki pimpinan. Apabila hal tersebut tidak dipenuhi maka individu akan terus mencari pergerakan apa yang akan dilakukan

BAB 5 IMPLEMENTASI

5.1 Implementasi NPC

Bagian ini merupakan hasil dari implementasi NPC yang tadi sudah di rancang sebelumnya ke dalam Unity. Gambar 5.1 merupakan hasil implementasi NPC ke dalam Unity



Gambar 5.1 Implementasi NPC

5.2 Implementasi Flocking Behavior Alignment

Implementasi Flocking Behavior Alignment akan dilakukan sesuai dengan diagram alir yang telah dirancang pada perancangan

Implementasi Method Start terdapat Tabel 5.1

	Start
1	private void Start()
2	{
3	al.AgenLst.Add(this);
4	myrigidbody = GetComponent<Rigidbody2D>();
5	TargetPoint = new Vector2(Target.position.x,
6	Target.position.y);
7	startPos = new Vector2(transform.position.x,
8	transform.position.y);
9	leaderRadMin = led.radMin;
10	leaderRadMax = led.radMax;
11	}

Tabel 5.1 Implemntasi Method Start

Dalam method ini di inisialisasikan variable yang sudah di deklrasikan sebelumnya mulai dari menambahkan diri agen ke dalam list , rigidbody, posisi target, posisi agen dan radius dari agen leader.

	Flocking Behavior
1	`public Vector2 FlockingBehavior()`
2	`    {`
3	`        foreach (var item in al.AgenLst)`
4	`        {`
5	`            if (item != this)`
6	`            {`
7	`                if (WithInRange(item))`
8	`                {`
9	`                    if (!neighbourAgent.Contains(item))`
10	`                    {`
11	`                        neighbourAgent.Add(item);`
12	`                        nebCount++;`
13	`                        dir.x += item.dir.x;`
14	`                        dir.y += item.dir.y;`
15	`                    }`
16	`                }`
17	`                else`
18	`                {`
19	`                    neighbourAgent.Remove(item);`
20	`                }`
21	`            }`
22	`        }`
23	`        if (nebCount != 0)`
24	`        {`
25	`            dir.x /= nebCount;`
26	`            dir.y /= nebCount;`
27	`            dir.Normalize();`
28	`            return dir;`
29	`        }`
30	`        return dir;`
31	`    }`

Tabel 5.2 Implementasi *Flocking Behavior*

Baris 3-19 merupakan baris dimana agen akan memeriksa agen agen yang ada di Gameworld dan jika agen yang lain berada di radius yang ditentukan maka Agen tersebut akan menjadi menjadi neighbor nya dan jika ada neighbor maka arah dari agen tersebut akan disamakan

	Update
1	`    private void Update()`
2	`    {`
3	`        dir = FlockingBehavior();`
4	`        dir.Normalize();`
5	`        myPos = new Vector2(transform.position.x,`
6	`transform.position.y);`
7	`        leaderPos = new Vector2(led.transform.position.x,`
8	`led.transform.position.y);`
9	`        distanceToLeader = Mathf.Abs((myPos -`
10	`leaderPos).magnitude);`
11	`        if (distanceToLeader > leaderRadMax)`
12	`        {`
13	`            dir = BackToLeader();`
14	`            backToLeader = true;`
15	`        }`
16	`        if (distanceToLeader < leaderRadMin &&`
17	`backToLeader)`
18	`        {`
19	`            dir = Random.Range(0.0f, 1) > 0.2f ?`
20	`randomdir() : dir;`
21	`            backToLeader = false;`
22	`        }`
23	`    Vector3 arah = new Vector3(dir.x, dir.y, 0);`
24	`        transform.position += arah * speed *`
25	`Time.deltaTime;`
26	`    }`

Tabel 5.3 Implementasi Method Update

Penjelasan :

Baris 3-10 merupakan bagian dimana agen akan melakukan pergerakan flocking dengan terlebih dahulu menetukan posisi agen dengan leader yang dibuat

Baris 11-22 di baris ini akan dihitung jarak antara agen, jika jarak agen lebih dari radius maksimal yang dipunyai leader maka akan agen akan kembali ke leader lalu, jika agen masih ada di radius minus leader maka akan bergerak secara random tapi masih tetap dalam radius leader

Baris 22-26 berfungsi untuk pertama tama menggerakan arah agen, lalu agen akan digerakkan menuju arah yang ditunjukkan

	Method With Range
1	bool WithInRange(Agen a)
2	{
3	if (Mathf.Abs((transform.position –
4	a.transform.position).magnitude) < neighbourRad)
5	{
6	return true;
7	}
8	return false;
9	}

Tabel 5.4 Implementasi Method With Range

Penjelasan :

Method *WithRange* ini berfungsi untuk mengetahui dimana posisi agen berada, jika jarak agen yang satu dengan yang lain kurang dari radius tetangga yang dimiliki tiap agen maka agen tersebut berada di dalam jangkauan agen atau menjadi tetangga agen tersebut

	Method RandomDir
1	Vector2 randomdir()
2	{
3	Vector2 newdir = new Vector2(Random.Range(-1.0f,
4	1.0f), Random.Range(.0f, 1.0f));
5	return newdir;
6	}

Tabel 5.5 Implementasi Method RandomDir

Penjelasan :

Method *RandomDir* ini berfungsi untuk mengatur gerakan agen, dimana jika agen melakukan pergerakan random sesuai *range* yang sudah ditentukan

	Method BackToLeader
1	Vector2 BackToLeader()
2	{ Vector2 ledPos = new
3	Vector2(led.transform.position.x, led.transform.position.y);
4	Vector2 thisPos = new Vector2(transform.position.x,
5	transform.position.y);
6	Vector2 newdir = ledPos - thisPos;
7	return newdir;
8	

Tabel 5.6 Implementasi Method BackToLeader

Penjelasan :

Method *Back To Leader* ini berfungsi untuk mengubah posisi agen, dimana agen akan dikembalikan ke posisi dekat dengan leader yang sudah ditentukan

	A*
1	public Transform target;
2	public Grid grid;
3	Transform seeker;
4	public List<Node> SeekerPath;
5	public Color warnaPath;
6	public Walk walk;
7	bool done;
8	void Awake() {
9	seeker = this.gameObject.transform;
10	}
11	void Update() {
12	Vector2 seekerPos = new Vector2(seeker.position.x,
13	seeker.position.y);
14	Vector2 targetPos = new Vector2(target.position.x,
15	target.position.y);
16	FindPath(seekerPos, targetPos);
17	if (SeekerPath.Count!=0)
18	{
19	walk.GetStarted(SeekerPath);
20	}
21	}
22	public void FindPath(Vector2 startPos, Vector2 targetPos) {
23	Node startNode = grid.NodeFromWorldPoint(startPos);
24	Node targetNode = grid.NodeFromWorldPoint(targetPos);
25	List<Node> openSet = new List<Node>();

```
26          HashSet<Node> closedSet = new HashSet<Node>();
27          openSet.Add(startNode);
28          while (openSet.Count > 0) {
29          Node node = openSet[0];
30          for (int i = 1; i < openSet.Count; i ++) {
31                  if (openSet[i].fCost < node.fCost ||
32  openSet[i].fCost == node.fCost) {
33                  if (openSet[i].hCost < node.hCost)node =
34  openSet[i];
35          }
36  }
37          openSet.Remove(node);
38          closedSet.Add(node);
39          if (node == targetNode) {
40                  RetracePath(startNode,targetNode);
41                  return;
42                          }
43          foreach (Node neighbour in grid.GetNeighbours(node)) {
44          if (!neighbour.walkable ||
45  closedSet.Contains(neighbour)) {
46          continue;
47                                  }
48          int newCostToNeighbour = node.gCost +
49  GetDistance(node, neighbour);
50          if (newCostToNeighbour < neighbour.gCost ||
51  !openSet.Contains(neighbour)) {
52          neighbour.gCost = newCostToNeighbour;
53          neighbour.hCost = GetDistance(neighbour, targetNode);
54          neighbour.parent = node;
55                  if (!openSet.Contains(neighbour))
56                          openSet.Add(neighbour);
57                                  }
58                          }
59                  }
60          }
61          void RetracePath(Node startNode, Node endNode) {
62          List<Node> path = new List<Node>();
63          Node currentNode = endNode;
64                  while (currentNode != startNode) {
65                          path.Add(currentNode);
66                          currentNode = currentNode.parent;
```

```
67                  }
68                  path.Reverse();
69                  SeekerPath = path;
70          }
```

Tabel 5.7 Implementasi Pathfinding

Penjelasan :

Baris 1-7 berisi akan variable variable apa saja yang akan digunakan dalam pathfinding yang akan digunakan

Baris 8-10 berisi method *Awake* yang berguna untuk memposisikan agen yang akan bergerak

Baris 11-21 berisi untuk menentukan posisi agen yang akan bergerak dan posisi tujuan yang akan dicapai, dan menggerakan agen ke posisi yang akan dituju

Baris 22-36 berfungsi untuk mengecek node yang ada disekitar agen dimana akan dihitung nilai fCost, jika nilai fCost dari open set lebih kecil atau sama dengan node fCost dan hasil hCost open set lebih kecil dari node hCost maka node selanjutnya akan menjadi openset

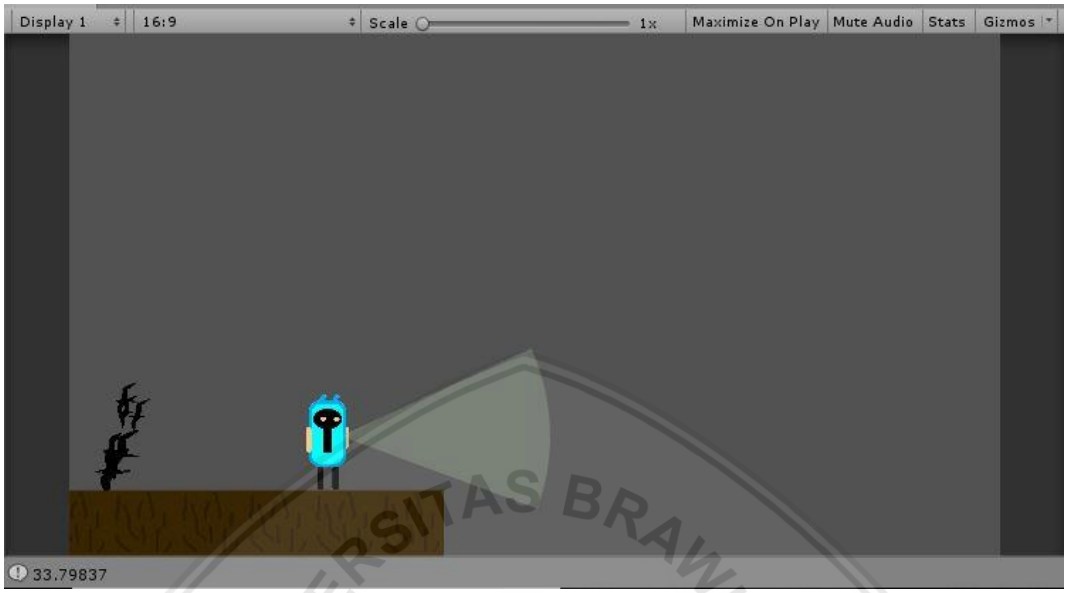
Baris 37-42 berfungsi untuk menghapus node yang sudah dilewati dan memasukkannya ke dalam closed set. Jika node yang dilewati merupakan node tujuan maka akan memanggil fungsi retrace path

Baris 43-47 berfungsi untuk mengecek node node yang ada, jika node nya dapat dilewati dan sudah masuk ke dalam closed set maka pencarian jalur dapat dilanjutkan

Barus 48-60 berfungsi untuk membuat variable newCostNeighbour, jika nilai variable tersebut kurang daari gCost dan merupakan tetangganya maka akan mengubah nilai gCost dan hCost dari neighbor

Baris 61-70 berfungsi untuk melihat apalakah jalur yang dilewati sudah dilewati atau belum

Gambar 5.2 merupakan hasil dari implementasi Flocking Behavior untuk pergerakan berkelompok dimana NPC sudah melakukan pergerakan berkelompok menuju target yang sudah ditentukan



Gambar 5.2 Implementasi Flocking Behavior Alignment

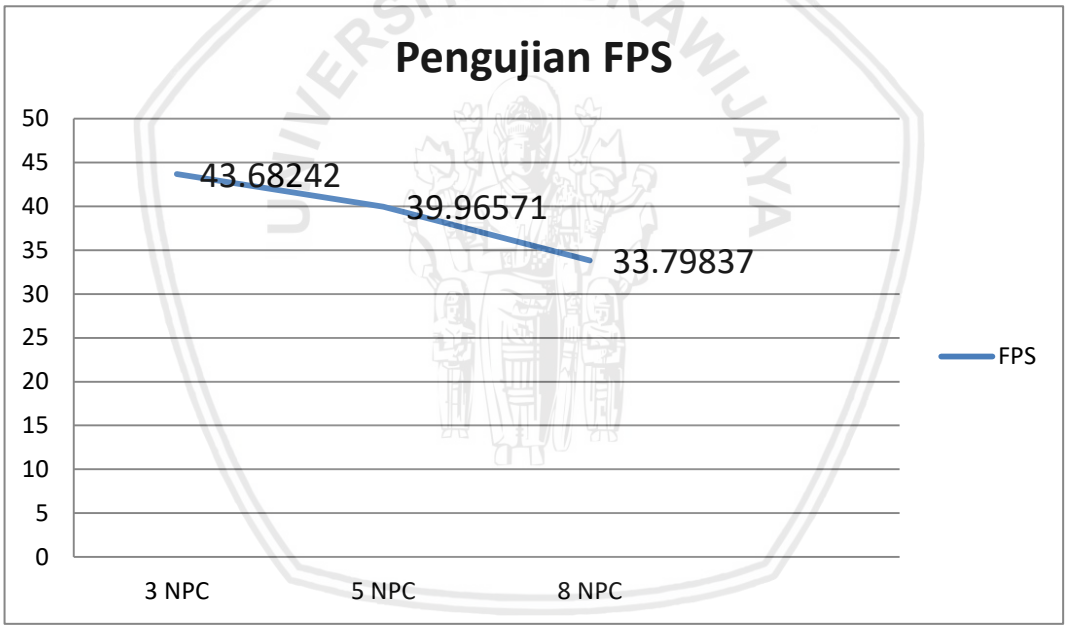
BAB 6 PENGUJIAN

6.1 Pengujian *Frame Per Second*

Pengujian akan dilakukan dengan beberapa individu dalam melakukan Flocking ini, pengujian akan dilakukan tiga kali, yaitu dengan 3 Non Player Character, 5 Non Player Character, dan 8 Non Player Character. Pengujian FPS ini berfungsi untuk mengetahui apakah *NPC* akan berpengaruh terhadapa FPS yang akan dihasilkan oleh *game.*

Spesifikasi Hardware :

- Intel Core I5-5200U 2.2 GHz
- RAM 8 GB
- NVIDIA GT840M

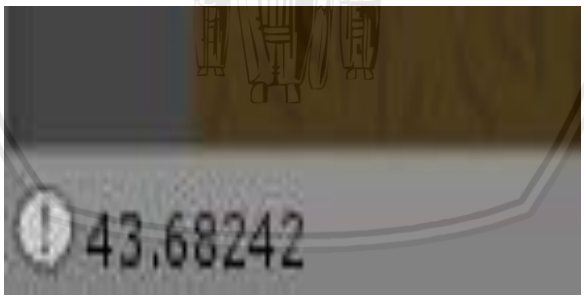


Gambar 6.1 Uji *Frame Per Second*

Uji ke	*Non Player Character*	*Frame Per Second*
Pertama	3	43.68242
Kedua	5	39.96571
Ketiga	8	33.79837

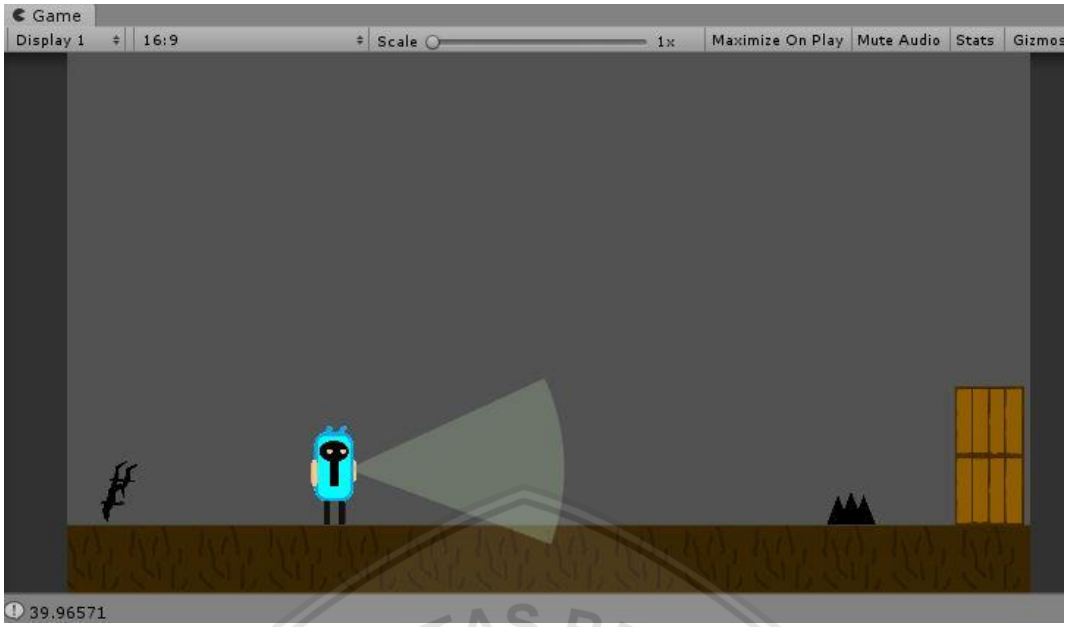
Tabel 6.1 Pengujian *Frame Per Second*

Gambar 6.2 Uji *Frame Per Second* dengan 3 *Non Player Character*

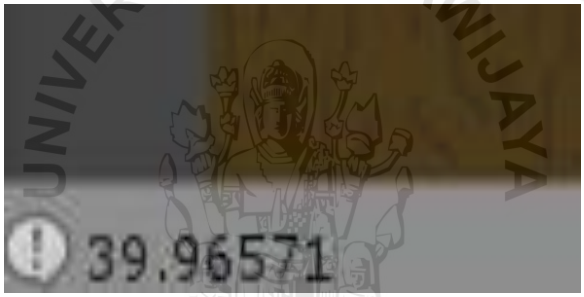


Gambar 6.3 Hasil *Frame Per Second* 3 *Non Player Character*

Pengujian Frame Per Second pergerakan berkelompok dengan 3 *NPC* menghasilkan FPS di angka 43.68424 seperti pada gambar 6.3

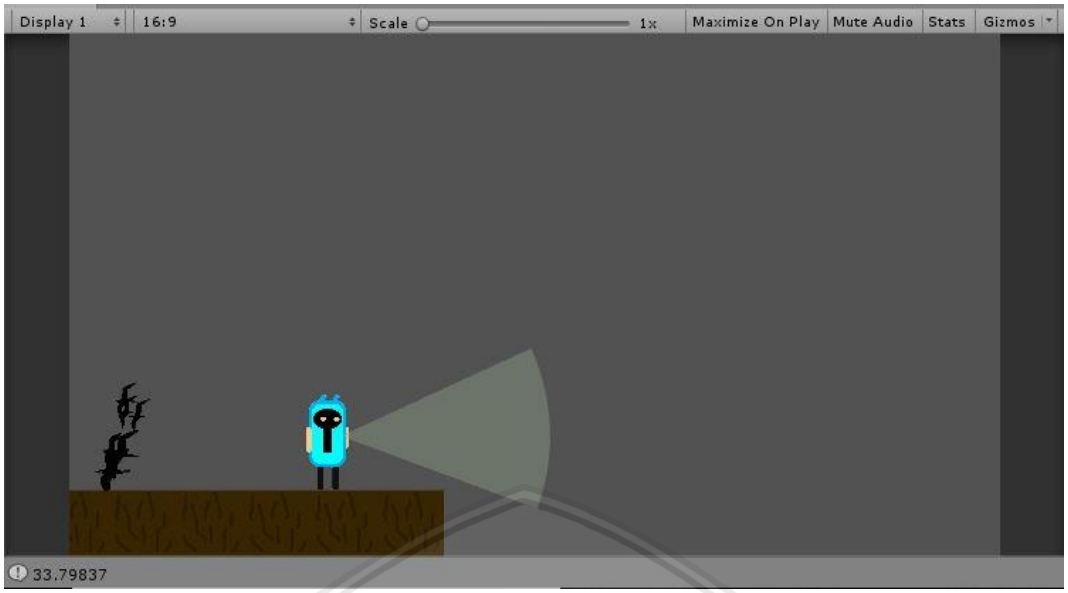


Gambar 6.4 Uji Frame Per Second dengan 5 Non Player Character

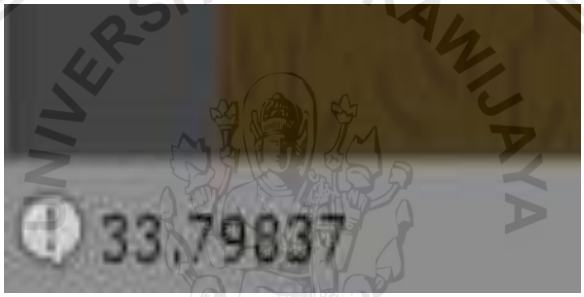


Gambar 6.5 Hasil *Frame Per Second* 5 *Non Player Character*

Pengujian Frame Per Second pergerakan berkelompok dengan 5 *Non Player Character* menghasilkan FPS di angka 39.96571 seperti pada gambar 6.5



Gambar 6.6 Pengujian Frame Per Second dengan 8 Non Player Character



Gambar 6.7 Hasil *Frame Per Second* 8 *Non Player Character*

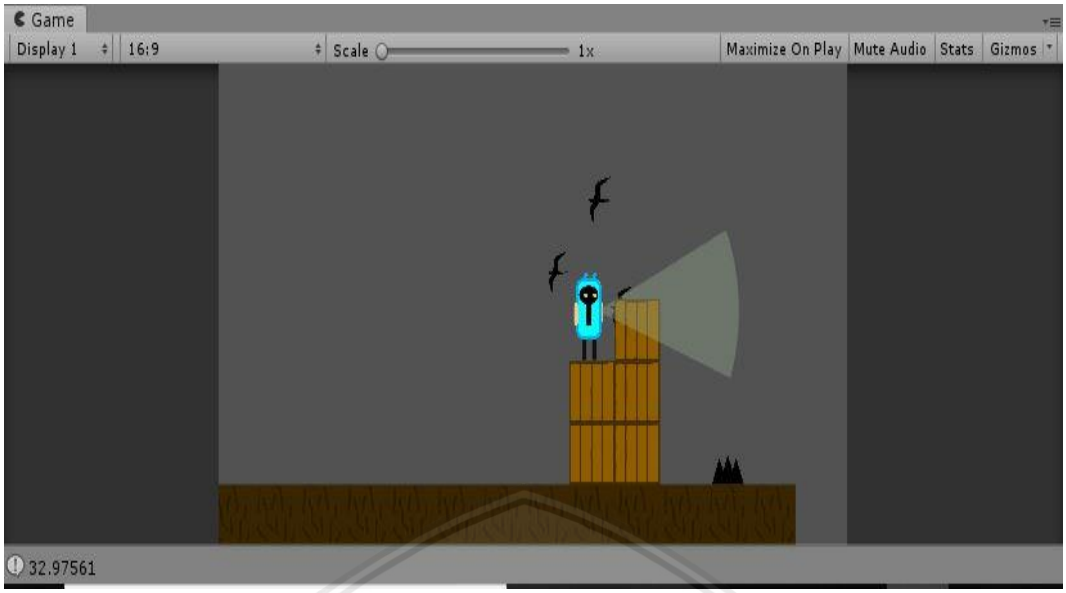
Pengujian Frame Per Second pergerakan berkelompok dengan 8 *Non Player Character* menghasilkan FPS di angka 33.79837 seperti pada gambar 6.7

6.2 Pengujian Validitas

Hal ini dilakukan untuk mendapatkan hasil yang Valid Non Player Character tiba ke target.

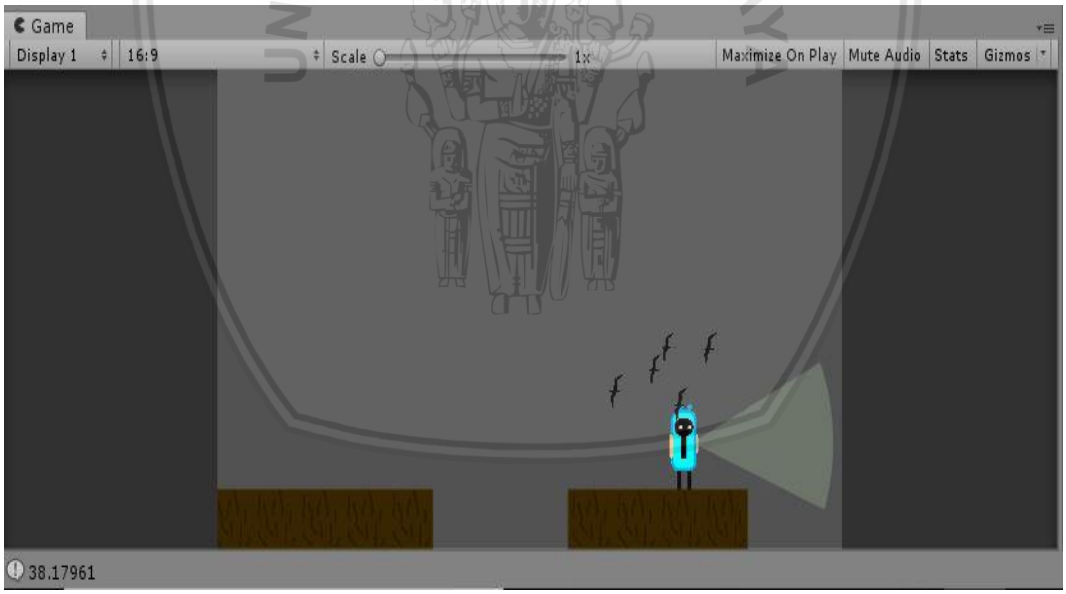
No	NPC	Target yang ingin dicapai	Keluaran	Kondisi
1	3	Non Player Character tiba ke player	Non Player Character tiba ke player	Valid
2	5	Non Player Character tiba ke player	Non Player Character tiba ke player	Valid
3	8	Non Player Character tiba ke player	Non Player Character tiba ke player	Valid

Tabel 6.2 Pengujian Validitas



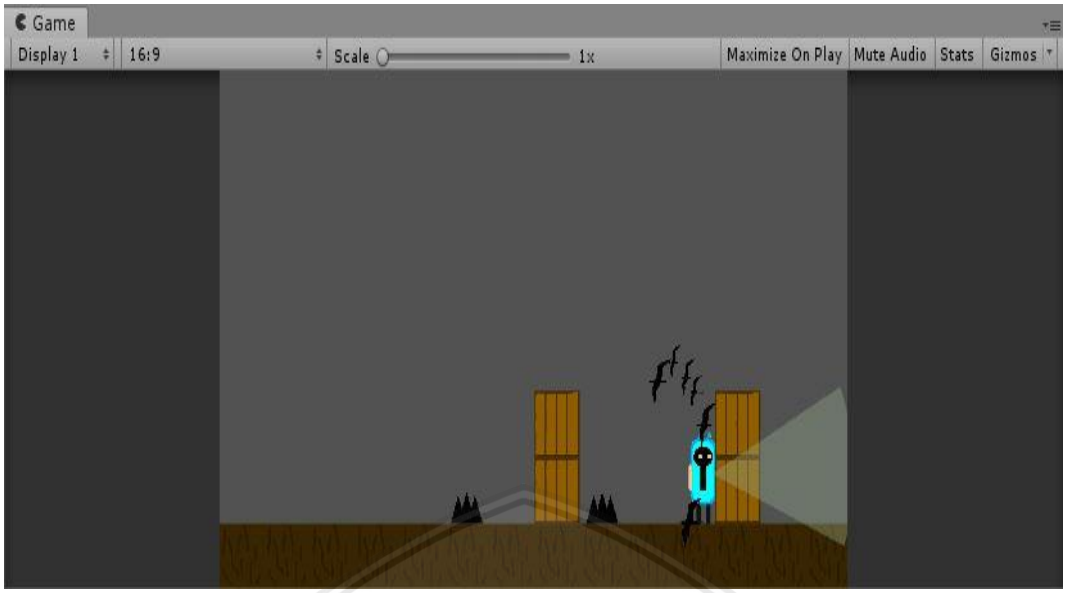
Gambar 6.8 Uji Validitas 3 *Non Player Character*

Hasil diatas menunjukkan bahwa NPC yang berjumlah 3 berhasil menuju tujuan yang sudah ditentukan yaitu player



Gambar 6.9 Uji Validitas 5 Non Player Character

Hasil diatas menunjukkan bahwa NPC yang berjumlah 5 berhasil menuju tujuan yang sudah ditentukan yaitu lingkaran berwarna hitam



Gambar 6.10 Uji Validitas 8 Non Player Character

Hasil diatas menunjukkan bahwa NPC yang berjumlah 8 berhasil menuju tujuan yang sudah ditentukan yaitu lingkaran berwarna hitam

BAB 7 PENUTUP

7.1 Kesimpulan

Berdasarkan hasil dari penelitian diatas, *Flocking* berhasil di implementasikan ke dalam *Non Player Character.* Adapun hasil yang di dapatkan yaitu :

1. *Flocking Alignment* berhasil diimplementasikan pada *Non Player Character* dalam bergerak menuju player yang sudah ditentukan. Hasil ini dibuktikan dari pengujian validitas yang dilakukan, dimana *Non Player Character* yang melakukan pergerakan berkelompok dapat sampai ke target yang sudah di tentukan.
2. Hasil dari uji menghasilkan bahwa *Non Player Character* bergerak dengan baik karena bisa sampai ke target yang sudah ditentukan akan tetapi banyaknya individu *Non Player Charater* akan menghasilkan *Frame Per Second* yang semakin turun. Ini dibuktikan dari pengujian *Frame Per Second* dalam *game ini* dimana dengan pengujian 3 *Non Player Character* menghasilkan *Frame Per Second* di angka 43.68, lalu dengan dengan pengujian 5 *Non Player Character* menghasilkan *Frame Per Second* di angka 39.96 dan dengan pengujian 8 *Non Player Character* menghasilkan *Frame Per Second* di angka 33.79.

7.2 Saran

Dengan hasil yang telah didapatkan dari penelitian ini maka peneulis memiliki saran yaitu :

Alignment(Penyesuaian) dapat diimplementasikan dalam permainan tersebut. Tapi tidak dipungkiri masih banyak metode lain dalam *Flocking* Seperti Kohesi dan Pemisahan, dalam permainan ini memang *Alignment* berhasil diterapkan tapi tidak menutup kemungkinan dengan metode yang lain. Selanjutnya dapat digunakan metode *Flocking* yang lain untuk pergerakan *Non Player Character.*

DAFTAR REFERENSI

Arif, M. U., Kurniawan, F., & Nugroho, F. (2011). A2-18 - YUNIFA_MISTACHUL_ARIF - DESAIN PERUBAHAN PERILAKU.pdf.

Bhosale, T., Kulkarni, S., & Patankar, S. N. (2018). 2D Platformer Game in Unity Engine, 3021–3024.

Cui, X., & Shi, H. (2011). A*-based pathfinding in modern computer games. *International Journal of Computer Science and …*, *11*(1), 125–130. Retrieved from https://www.researchgate.net/publication/267809499%0Ahttp://paper.ijcsns.org/07_book/201101/20110119.pdf

Dewi, M., Hariadi, M., & Purnomo, M. H. (2011). Simulating the movement of the crowd in an environment using flocking. *Proceedings - International Conference on Instrumentation, Communication, Information Technology and Biomedical Engineering 2011, ICICI-BME 2011*, (November), 186–191. https://doi.org/10.1109/ICICI-BME.2011.6108638

Girdhar, A. (2015). Swarm Intelligence and Flocking Behavior. *International Journal of Computer Applications*, (Icaet), 975–8887.

Larsson, M., Lundgren, S., Larsson, M., & Lundgren, S. (2017). Perception of Realistic Flocking Behavior in the Boid Algorithm.

Nugroho, P. R., & Hariadi, M. (2014). Pengaturan Perilaku Pasukan Non Player Character menggunakan metode Flocking Behavior berbasis Agent pada permainan Real Time Strategy, 1–5.

Pedersen, K. (2014). Procedural Level Generation in Games, 797–801. Retrieved from http://www.raywenderlich.com/wp-content/uploads/2014/05/rw_procedural_level_generation_final.pdf

Prasetyo, F. R., Muh, E., Jonemaro, A., & Akbar, M. A. (2017). Penerapan Algoritma Hybrid Pathfinding A * dan Boids untuk Game Pesawat Tempur, *1*(12), 1616–1621.

Reynolds, C. W. (1987). Flocks , Herds , and Schools : A Distributed Behavioral Model, *21*(4), 25–34.

Robert, S., & Tihomir, O. (2018). User Experience Evaluation of 2D Side-Scrolling Game Developed Using Overlap2D Game Editor and LibGDX Game Engine, 1580–1585.

Sagala, M. L., Jonemaro, E. M. A., & Wardhono, W. S. (2017). Pengembangan Game Platformer 2D Menggunakan Teknik Projection Mapping, *1*(11), 1160–1168.

Supervisors, D. C., Paraschakis, D., Mihailescu, R.-C., & Eriksson, J. (2016). Game Design Patterns in Endless Mobile Minigames, 73. Retrieved from https://muep.mau.se/bitstream/handle/2043/21375/DavidCao_GameDesig

nPatternsInEmm_2016.pdf?sequence=2

Warpefelt, H. (2016). *The Non-Player Character – Exploring the believability of NPC presen- tation and behavior*. Retrieved from https://www.diva-portal.org/smash/get/diva2:912617/FULLTEXT01.pdf%0Ahttps://people.dsv.su.se/~hw/

Yancan, Z., & Technology, I. (2016). Create an Endless Running Game in Unity, (May).