# PENERAPAN ALGORITMA *COLLISION DETECTION* DAN *BAYESIAN* UNTUK STRATEGI MENYERANG JARAK DEKAT PADA NPC (*NON PLAYER CHARACTER)* MENGGUNAKAN *UNITY* 3D

Siti Asmiatun
Fakultas Teknologi Informasi dan Komunikasi Universitas
Semarang siti.asmiatun@usm.ac.id

***Abstract*—Game merupakan bidang ilmu yang semakin berkembang pesat dan sangat menarik untuk dipelajari karena memiliki potensi baik dari segi ilmu pengetahuan maupun dari segi komersil. Untuk menghasilkan game yang atraktif dan realistik, ada beberapa hal yang perlu diperhatikan. Salah satunya adalah memperhatikan strategi penyerangan khususnya untuk *NPC (Non Player Character)*. Mengacu pada penelitian sebelumnya yang membahas tentang strategi menyerang jarak dekat untuk NPC *(Non Player Character)* [1]. Penelitian ini menggabungkan dua metode *Collision Detection* dengan bayesian untuk strategi menyerang jarak dekat. Dalam penelitian ini adalah membagi beberapa perilaku penyerangan NPC ketika berada pada posisi paling dekat dengan musuh. Penelitian ini menerapkan algoritma bayesian untuk klasifikasi perilaku penyerangan NPC dan algoritma collision detection untuk pengambilan keputusan perilaku ketika NPC menabrak player. Dengan penggabungan algoritma tersebut diharapkan dapat memperbaiki dari kelemahan penelitian sebelumnya. Klasifikasi penyerangan NPC dibiagi menjadi dua perilaku penyerangan yaitu perilaku smash dan perilaku kick. Sedangkan untuk variabel yang digunakan dalam klasifikasi bayesian adalah health point, attack point player dan jarak yang diperoleh dari kondisi NPC. Hasil penerapan metode collision detection dan bayesian ini membuktikan bahwa dengan metode collision detection NPC dapat mengambil keputusan perilaku menyerang meskipun berada pada jarak terdekat dengan player. Dari hasil pengujian metode dengan perhitungan confusion matrix mendapatkan akurasi sebanyak 90%.***

***Keywords—game; klasifikasi; naive; bayes; strategi; menyerang; collision; detection***

## I. PENDAHULUAN

Game adalah kebutuhan dasar setiap manusia untuk menikmati hidup dan sebagai media pembelajaran [7]. *Non Player Characters* (NPC) merupakan bagian penting
di dalam sebuah game. Pemilihan aksi yang independen mampu membuat permainan menjadi lebih menarik. Namun, perilaku independen tanpa disertai dengan kecerdasan justru dapat mengurangi daya tarik permainan [2]. Untuk menghasilkan NPC yang lebih atraktif dan realistik dibutuhkan penerapan AI (*Artificial Intellegence*) kedalam NPC. *Artifical Intelligence* (AI) pada *game* FPS umumnya terdiri dari perencanaan *path*, mengambil *item*, menggunkan *item*, dan berperang [3]. Khusus untuk berperang NPC diharapkan mempunyai strategi-strategi khusus seperti halnya manusia [4][5].

Model strategi menyerang bisa bermacam-macam, misalnya strategi menyerang dengan memancing serangan musuh, kemudian pada kondisi tertentu perilaku berubah menjadi menghindar. Model strategi menyerang dalam sebuah game harus direncananakan untuk membuat game yang tidak membosankan bagi pemain [1].

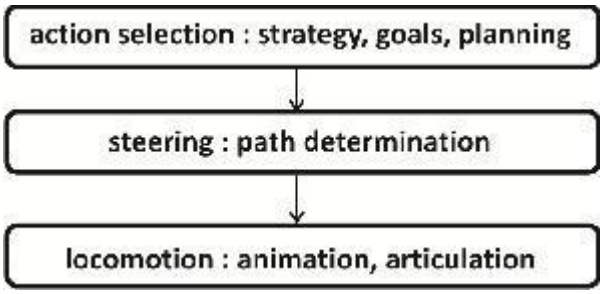
Meninjau penelitian dari siti asmiatun dkk tentang strategi menyerang jarak dekat menggunnakan klasifikasi bayesian [1], terdapat beberapa kekurangan salah satunya adalah NPC didalam *game* tidak dapat menyerang musuh ketika berada pada posisi paling dekat dengan musuh dikarenakan *collider* NPC menyentuh *collider player*, sehingga mengakibatkan NPC terbang dan serangan NPC tidak fokus. Dari kekurangan penelitian tersebut, penelitian ini difokuskan untuk menerapkan penggabungan algoritma *collision detection* dan algoritma *bayesian* untuk startegi menyerang jarak dekat. *Collision Detection* bertujuan untuk mendeteksi adanya tabrakan atau serangan dari musuh, sehingga ketika jarak antara NPC dan *player* telah diklasifikasikan oleh bayesian untuk menyerang jarak terdekat, maka NPC dapat berperilaku yang sesuai dengan kondisinya. Pengembangan penelitian ini diharapkan dapat menghasilkan sebuah game yang menarik dengan akurasi yang lebih baik lagi dari penelitian sebelumnya.

## II. LANDASAN TEORI

### *A. NPC (Non Player Character)*

NPC (*Non Player Character*) adalah sebuah objek atau karakter dalam sebuah game yang dijalankan oleh komputer dan dapat berinteraksi dengan pemain *game.* NPC biasa disebut sebagai agen otonom karena termasuk karakter atau tokoh yang memiliki kemampuan untuk improvisasi tindakan

dan perilaku mereka. Perilaku karakter yang otonom dapat dibagi menjadi tiga lapisan yang hirarkis seperti gambar dibawah ini :




Gambar 2.1 Hirarki Gerak Perilaku

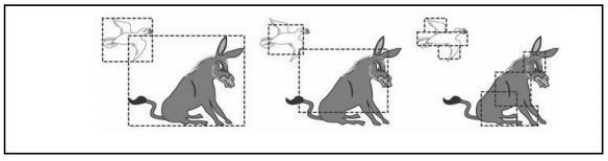
Gambar diatas menunjukkan pembagian perilaku karakter otonom menjadi tiga lapisan : seleksi tindakan, *steering*, dan penggerak. Pada penelitian ini difokuskan pada bagian *action selection* yang didalamnya berisi tentang strategi menyerang jarak dekat dengan *collision detection* dan klasifikasi *bayesian* [6].

### *B. Naive Bayesian Classification*

*Naïve Bayes* merupakan pengklasifikasian dengan metode probabilitas dan statistik yang dikemukakan oleh ilmuwan Inggris Thomas Bayes, yaitu memprediksi peluang di masa depan berdasarkan pengalaman di masa sebelumnya sehingga dikenal sebagai *Teorema Bayes*. Menurut Olson dan Delen (2008:102) menjelaskan *Naïve bayes* untuk setiap kelas keputusan, menghitung probabilitas dengan syarat bahwa kelas keputusan adalah benar, mengingat vektor informasi obyek. Algoritma ini mengasumsikan bahwa atribut obyek adalah independen. Probabilitas yang terlibat dalam memproduksi perkiraan akhir dihitung sebagai jumlah frekuensi dari "master" tabel keputusan [9].

### *C. Collision Detection*

Setiap permainan menerapkan collision detection (deteksi tabrakan), baik itu dalam hal tabrakan antara sprite dengan sprite maupun antara sprite dengan peluru dan lain-lain. Proses collision dapat dibagi menjadi dua kategori dasar, yaitu collision detection dan collision response, dengan jarak respon yang telah diaplikasikan secara spesifik. Terdapat banyak sekali jenis dari collision detection itu sendiri, pada gambar 2.11 merupakan tiga tipe dari collision detection [8].

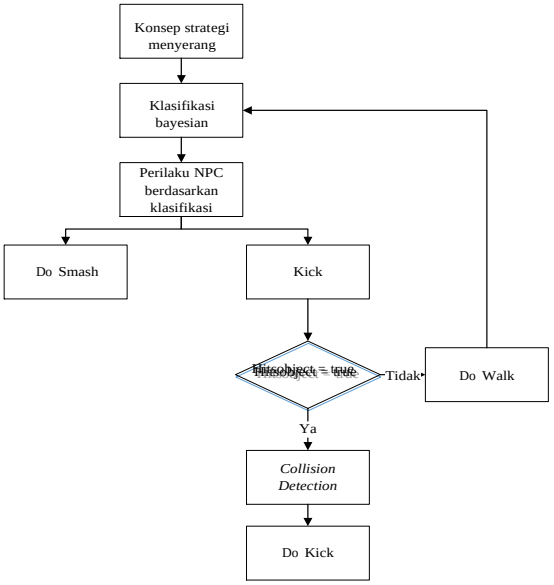


Gambar 2.2 Tipe *Collision*

Setiap objek diberi sebuah kotak sebagai acuan jika collision itu terjadi. Gambar pertama menunjukkan collision detection terjadi ketika objek burung bertabrakan dengan kotak yang memuat objek keledai, pada jenis collision detection ini rentan terhadap ketidaktelitian. Gambar kedua menunjukkan collision detection terjadi ketika kotak dari kedua objek tersebut bersinggungan. Jenis collision detection ini sudah lebih akurat dibanding jenis collision detection pada gambar pertama, namun untuk memilih faktor reduksi yang cocok itu sulit. Gambar ketiga menunjukkan collision detection terjadi ketika objek saling bersinggungan tanpa memperdulikan kotak secara keseluruhan.

## III. METODE PENELITIAN

Metode penelitian yang digunakan untuk penelitian ini adalah dengan menggunakan penggabungan metode *collision detection* dan *Bayesian*. Seperti yang diilustrasikan dalam *flowchart* dibawah ini menjelaskan tentang tahap – tahap yang akan dilakukan dalam penerapan metode *collision detection* dan *bayesian*.




Gambar 3.1 *Flowchart* Metode Penelitian

### *A.* Perancangan Strategi Menyerang

Penelitian ini akan menggunakan beberapa perilaku untuk strategi penyerangan yaitu perilaku Memukul dan Mengigit. Sedangkan parameter yang digunakan untuk menentukan perilaku adalah jarak, *Health Point* (HP) dan *Attack Power player* (APP) dari NPC, parameter dikelompokkan menjadi tiga kategori sebagai berikut:

Tabel 3.1 Kategori APP

| | |
|---|---|
| APP >= 0 dan APP <= 1000 | Lemah |
| APP >= 1001 dan APP <= 2000 | Kuat |

Tabel 3.2 Kategori HP

| | |
|---|---|
| HP>= 0 dan HP <= 55000 | kecil |
| HP >= 55001 dan HP <=114000 | besar |

Tabel 3.3 Kategori Jarak

| | |
|---|---|
| Jarak > 5.0 | Jauh |
| Jarak < 2.0 | Dekat |

Tabel 3.4 *Rule* penentuan perilaku

| |
|---|
| Jika APP kuat dan jarak jauh dan HP jauh maka smash |
| Jika APP kuat dan jarak jauh dan HP sedang maka smash |
| Jika APP kuat dan jarak dekat dan HP dekat maka kick |
| Jika APP kuat dan jarak dekat dan HP jauh maka kick |
| Jika APP lemah dan jarak jauh dan HP sedang maka smash |
| Jika APP lemah dan jarak jauh dan HP dekat maka smash |
| Jika APP lemah dan jarak dekat dan HP jauh maka kick |
| Jika APP lemah dan jarak dekat dan HP sedang maka kick |

*B.* Klasifikasi *NPC (Non Player Character)*

Untuk menyusun strategi menyerang jarak dekat dengan klasifikasi bayesian membutuhkan dataset yang berupa status *Attack Power, Jarak, Health Point* dari NPC. Pada penilitian ini menggunakan dataset yang dihasilkan secara random. Berikut tabel dataset yang dibutuhkan untuk klasifikasi :

Tabel 3.5 Dataset atribut NPC 1

| ***Attack Power Player* (DPP)** | **Jarak NPC** | ***Health Point* (HP)** | **Perilaku** |
|---|---|---|---|
| kuat | jauh | besar | smash |
| kuat | dekat | besar | kick |
| kuat | jauh | kecil | smash |
| lemah | jauh | besar | smash |
| kuat | dekat | besar | kick |
| kuat | dekat | kecil | kick |
| lemah | jauh | besar | ? |

Dari dataset diatas, dapat dihitung dengan menggunakan teorema Bayes sebagai berikut :

$$P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)}.$$

Keterangan :
$P(C_i|X)$ = Probabilitas hipotesis $C_i$ jika diberikan fakta atau *record* X (*Posterior probability*)
$P(X|C_i)$ = mencari nilai parameter yang memberi kemungkinan yang paling besar (*likelihood*)
$P(C_i)$ = Prior probability dari X (*Prior probability*)
P(X) = Jumlah probability *tuple* yg muncul

Jadi perhitungan yang dihasilkan dari teorema bayes diatas adalah :
Asumsi :
Y = Perilaku
X1 = APP
X2 = Jarak
X3 = HP

Dengan metode bayes dapat dihitung :
P**(**X1 = Lemah, X2 = Jauh, X3 = Besar | smash**)** = {1/3.3/3.2/3}. 3/6 = 1
P**(**X1 = Lemah, X2 = Jauh, X3 = Kuat | kick**)** = {0/3.0/1.2/1}. 3/6 = 0

Sehingga menghasilkan table setiap perilaku sebagai berikut :

Tabel 3.6 Hasil klasifikasi bayesian

| **Perilaku** | **Hasil** |
|---|---|
| smash | 1 |
| kick | 0 |

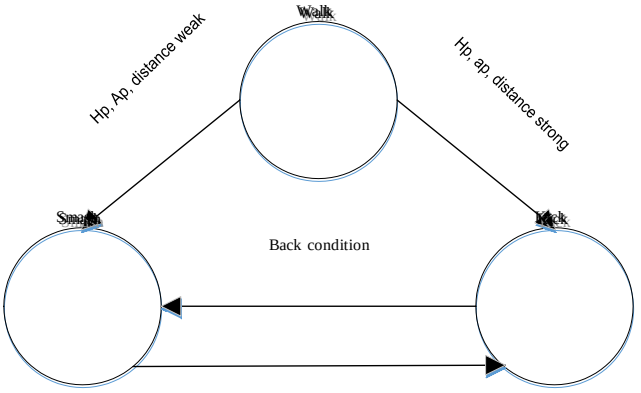
*C. Penentuan Perilaku NPC (Non Player Character)*

Dalam menentukan perilaku harus berdasarkan dari nilai probabilitas perhitungan bayesian. Jika nilai probabilitas yang paling tinggi dimiliki oleh kategori smash maka perilaku NPC akan masuk kategori smash. Tapi jika nilai probabilitas yang paling tinggi dimiliki oleh kategori kick maka perilaku tidak akan memilih perilaku kick, melainkan perilaku akan mengecek sesuai dengan metode *collision detection*, apakah *collider* yang dimiliki NPC menyentuh *collider player*, jika *collider* NPC terbukti menyentuh *collider player* NPC akan mengambil keputusan untuk memilih perilaku kick.

## IV. HASIL DAN PEMBAHASAN

Sesuai dengan langkah-langkah pengembangan metode penelitian yang digunakan yaitu meliputi :

*A.* Perancangan *Finite State Machine* (FSM)

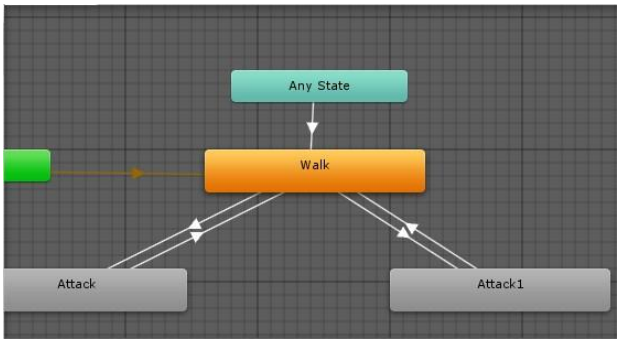



Gambar 4.1 Finite State Machine

*B.* Pengumpulan *Model Game (*Material Collecting*)*

Gambar 4.2 *Non Player Character*

Gambar 4.3 *Player*




Gambar 4.4 *Animation Control* NPC

*C.* Pembuatan Pengembangan *Game* (Assembly)

Pada tahap ini, ada beberapa data training yang akan di implementasikan dengan formula *bayesian*.

1) Dataset

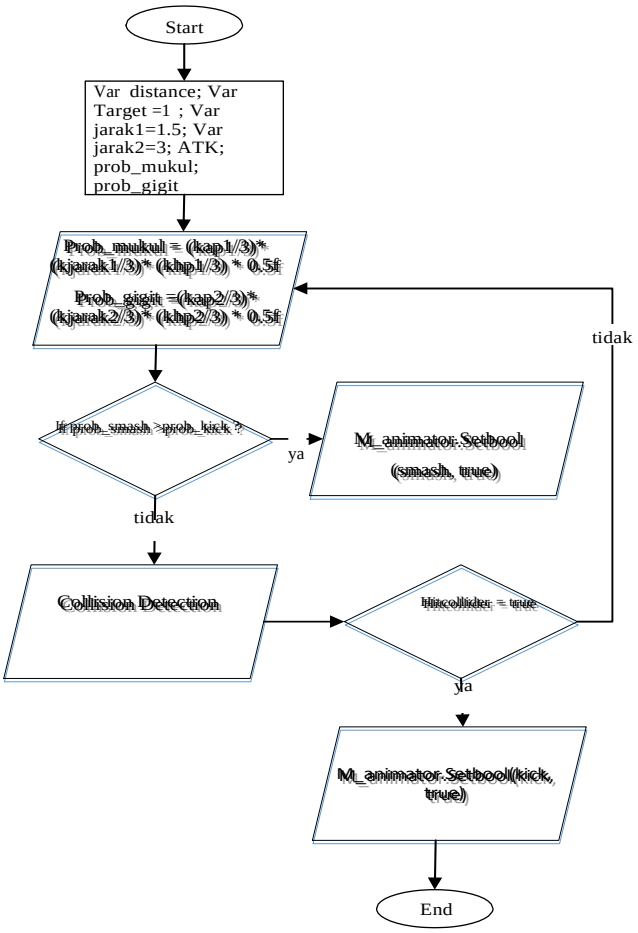
Tabel dibawah ini merupakan dataset yang akan diterapkan pada NPC. Dataset tersebut dapat dilihat pada tabel 4.3.

Tabel 4.1 Data training untuk klasifikasi bayes

| HP | Jarak | AP | Perilaku |
|---|---|---|---|
| kuat | jauh | besar | smash |
| kuat | dekat | Besar | kick |
| kuat | jauh | kecil | smash |
| lemah | jauh | besar | smash |
| kuat | dekat | besar | kick |
| kuat | dekat | kecil | kick |

2) Modul *AI Enemy*

Pada tahap ini akan menjelaskan tentang bahasa pemrograman yang digunakan untuk menerapkan algoritma *collision detection* dan *bayesian.* Berikut flowchart script penerapan algoritma *collision detection* dan *bayesian* :




Gambar 4.5 Flowchart penggabungan metode

*D.* Pengujian Skenario

Pada tahap pengujian skenario, dibutuhkan sebuah *rule* yang digunakan untuk mengetahui ketepatan pengambilan keputusan perilaku yang dilakukan oleh NPC. Pengujian skenario bertujuan untuk menentukan kebenaran pengambilan keputusan perilaku. *Rule* tersebut ditentukan oleh *developer game* yang bisa dilihat pada tabel 3.1 sampai dengan tabel 3.4.

Didalam metode bayesian yang berfungsi untuk mengklasifikasikan perilaku NPC, dibutuhkan perbandingan nilai probabilistik perilaku satu dengan nilai probabilistik lainnya. Jadi kondisi tersebut bisa dikatakan diterima jika :

Tabel 4.2 Kondisi yang diterima

| **Kondisi 1** | **Kondisi 2** | **Perilaku** |
|---|---|---|
| *Probabilitas smash > = probabilitas kick* | *Hitcollider = false* | *Smash* |
| *Probabilitas smash < = probabilitas kick* | *Hitcollider = true* | *Kick* |

Tabel 4.3 Hasil validasi

| **No.** | **Kondisi** | **Harapan** | **Realita** | **Kesimpulan** |
|---|---|---|---|---|
| 1 | Prob smash = 0.22<br>Prob kick = 0<br>collision = false | Smash | Smash | Valid |
| 2 | Prob smash = 0.11<br>Prob kick = 0<br>collision = false | Smash | Smash | Valid |
| 3 | Prob smash = 0<br>Prob kick = 0.33<br>collision = false | Idle | Kick | Tidak valid |
| 4 | Prob smash = 0.1<br>Prob kick = 0<br>collision = false | Smash | Smash | Valid |
| 5 | Prob smash = 0.23<br>Prob kick = 0<br>collision = false | Smash | Smash | Valid |
| 6 | Prob smash = 0<br>Prob kick = 0.33<br>collision = true | Kick | Kick | Valid |
| 7 | Prob smash = 0.12<br>Prob kick = 0<br>collision = false | Smash | Smash | Valid |
| 8 | Prob smash = 0<br>Prob kick = 0.111<br>collision = true | Kick | Kick | Valid |
| 9 | Prob smash = 0<br>Prob kick = 0.31<br>collision = false | Idle | Kick | Tidak valid |
| 10 | Prob smash = 0.22<br>Prob kick = 0<br>collision = false | Smash | Smash | Valid |
| 11 | Prob smash = 0<br>Prob kick = 0.33<br>collision = true | Kick | Kick | Valid |
| 12 | Prob smash = 0.33<br>Prob kick = 0<br>collision = false | Smash | Smash | Valid |
| 13 | Prob smash = 0.11<br>Prob kick = 0<br>collision = false | Smash | Smash | valid |
| 14 | Prob smash = 0<br>Prob kick = 0.33<br>collision = true | Kick | Kick | Valid |
| 15 | Prob smash = 0<br>Prob kick = 0.33<br>collision = true | Kick | Kick | Valid |
| 16 | Prob smash = 0.33<br>Prob kick = 0<br>collision = false | Smash | Smash | Valid |
| 17 | Prob smash = 0<br>Prob kick = 0.11<br>collision = true | Kick | Kick | valid |
| 18 | Prob smash = 0.22<br>Prob kick = 0<br>collision = false | Smash | Smash | Valid |
| 19 | Prob smash = 0<br>Prob kick = 0.44<br>collision = true | Kick | Kick | Valid |
| 20 | Prob smash = 0<br>Prob kick = 0.21<br>collision = true | Kick | Kick | Valid |

*E.* Pengujian Metode

Dari tabel diatas, dapat disimpulkan dengan tabel *confusion matrix* dengan *Class* a dan *Class* b pada tabel *confusion matrix* berturut-turut menunjukkan class smash dan kick yaitu sebagai berikut :

Tabel 4.4 Contoh Confusion matrix

| | | *Predicted Class* | |
|---|---|---|---|
| | | *Class a = smash Class b= kick* | |
| *Actual Class* | *Class a* | 10 | 0 |
| | *Class b* | 8 | 2 |

Berdasarkan hasil *confusion matrix* terlihat bahwa 10 record pada class *a* diprediksi tepat sebagai *class a* dan sebanyak 0 *record* diprediksikan tidak tepat sebagai *class b*. Selanjutnya, seluruh record pada *class b* terdapat 2 record diprediksi tidak tepat pada *class a* dan 8 record diprediksi tepat sebagai *class b*. Dari hasil tersebut dapat dihitung akurasinya sebagai berikut :

$$\text{Akurasi} \quad \frac{10 + 8}{10 + 0 + 8 + 2} \quad X\ 100 = 90\ \%$$

Jadi dapat disimpulkan bahwa perhitungan persentase tingkat akurasi pada confusion *matrix* mencapai nilai persentase sebesar 90%.

*F.* Distribusi

Pada tahap distribusi merupakan tahapan yang digunakan untuk membangun *game* kedalam sebuah aplikasi *game* yang sesuai dengan perngkat yang dibutuhkan. Berikut adalah gamabaran hasil dari game :


Gambar 4.6 *Display Kick condition*


Gambar 4.7 *Display Smash condition*

## V. KESIMPULAN DAN SARAN

Dari hasil penelitian ini dapat disimpulkan bahwa penggabungan dua metode *collision detection* dan *bayesian* dapat memperbaiki dari penelitian sebelumnya yaitu NPC didalam *game* tidak dapat menyerang musuh ketika berada pada posisi paling dekat dengan musuh dikarenakan *collider* NPC menyentuh *collider player*. Hasil penerapan metode *collision detection* ini membuktikan bahwa dengan metode *collision detection* NPC dapat mengambil keputusan perilaku menyerang meskipun berada pada jarak terdekat dengan *player*. Dari hasil pengujian metode dengan perhitungan *confusion matrix* mendapatkan akurasi sebanyak 90% ketepatan dalam pengambilan keputusan perilaku menyerang. Untuk pengembangan penelitian selanjutnya dapat ditambahkan perilaku menyerang pada NPC.

## VI. DAFTAR PUSTAKA

[1] Asmiatun dan Hendrawan. "Implementasi Klasifikasi *Bayesian* untuk strategi menyerang jarak dekat pada NPC (*Non Player Character*) Menggunakan *Unity 3D*".

[2] Hidayah, Muliadi dan Ridwan. " Algoritma Boids dan Logika Fuzzy Pada Pergerakan dan Perilaku *Non Player Character* Permainan *Borneo Mission*". Kumpulan Jurnal Ilmu Kopmuter. Volume 04, No.01. ISSN : 2406-7857. 2016.

[3] JinHyuk Hong dan Sung-Bae Cho. "Evolving Reactive NPCs for the Real-Time Simulation Game". CIG, 2005.

[4] Yunifa Miftachul Arif, Fachrul Kurniawan dan Fresy Nugroho. "Desain Perubahan Perilaku pada NPC Game Menggunakan Logika Fuzzy". National Seminar on Electrical, Informatics, and Its Education, 2011.

[5] McPartland M,and Gallagher M. "Creating a Multi-Purpose First Person Shooter Bot With Reinforcement Learning ". IEEE Symposium on Computational Intelligence and Games, pp. 143-150,2008.

[6] Craig W. Reynolds. *Steering Behaviors For Autonomous Characters*. Sony Computer Entertainment America.

[7] Adams, E. *Fundamental Of Game Design, 2nd Edition*. Pearson Education, Inc, Berkeley. 2010

[8] Davison, Andrew. *Killer Game Programming in Java*. O'Reilly Media : Sebastopol. 2005

[9] D.L. Olson and D. Delen." Advanced data mining techniques". Springer Verlag. ISBN3540769161. 2008.