Penerapan *Probabilistic FSM* pada AI musuh dalam game ARPG untuk gerakan AI tidak monoton.

Nicolas Wiyendi, Djoni Haryadi Setiabudi, Hans Juwiantho
Program Studi Informatika, Fakultas Teknologi Industri, Universitas Kristen Petra
Jl. Siwalankerto 121-131 Surabaya 60236
Telp. (031) – 2983455, Fax. (031) - 8417658
nicolaswiyendi24@gmail.com, djonihs@petra.ac.id, hans.juwiantho@petra.ac.id

ABSTRAK

Game sudah sangat populer dan merupakan dari kehidupan manusia mulai dari anak-anak. Seiring berkembangnya zaman kebutuhan akan *AI* game yang tidak monoton menjadi semakin nyata, salah satu masalah yang membuat *AI* dalam *game* monoton adalah karena *AI* yang mengulang-ulang gerakan, oleh karena itu dibuatlah *AI* yang tidak mengulang-ulang gerakan dan dapat membuat pemain tidak bosan dengan gerakan yang diulang-ulang (*spam*) serta termotivasi untuk bermain. Penelitian sebelumnya sudah pernah dibuat namun dengan genre *game* yang berbeda dan teknik *random* yang berbeda.

Game akan dikembangkan dengan metode *Probabilistic finite state machine* dan digabungkan dengan *random shufflebag*, *Probability* akan digunakan dalam memunculkan animasi gerakan sehingga animasi yang keluar tidak hanya satu dan *random shufflebag* akan digunakan dalam pemilihan gerakan sehingga gerakan yang keluar terbagi rata dan tidak diulang-ulang.

Hasil dari pengujian menunjukkan bahwa AI tidak mengulang-ulang gerakan namun dapat mengulang pola gerakan, Masalah pengulangan pola dapat diselesaikan dengan adanya probability pada animasi yang keluar sehingga dengan pola yang sama gerakan yang dilihat pemain dapat berbeda.

Kata Kunci: *Probabilistic Finite State Machine, Random Shufflebag, Probability*, C#, *Unity 3D, ARPG Game.*

ABSTRACT

Game is really popular and becoming one of human aspects of life from child, Along with times the needs of game AI that's not monotone has become more and more real, The problem that made a game monotone is the AI repeating its movement, with that the AI that doesn't repeat its movement is made, this AI will make players not bored because of spam movement also motivated to play the game. Previous research has been made but with different genre and different random technic.

Game will be developed with Probabilistic finite state machine methods combine with random shuffle bag. Probability will be used for showing the animation so it will make the animation that's come out more than one, and random shuffle bag will be used for the decision making for the movement so its evenly divided and not repeated.

Result of the testing shows that AI is not repeating the movement but it can repeat the pattern of the movement, Problem with repeating pattern can be solve with probability on animation that's come out so with the same pattern movement player can see different movement.

Keywords: *Probabilistic Finite State Machine, Random Shufflebag*, *Probability*, C#, *Unity 3D, ARPG Game*.

1. PENDAHULUAN

Video game sudah sangat populer dan merupakan bagian dari kehidupan manusia mulai dari anak-anak, seiring berkembangnya zaman kebutuhan akan *AI game* yang tidak monoton menjadi masalah yang nyata [7]. Masalah pada *AI game* yang membuat *AI* monoton adalah gerakan yang mudah ditebak dan gerakan yang diulang-ulang, dapat membuat permainan terasa hampa, untuk mengatasi masalah ini akan dilakukan penerapan metode *Probabilistic Finite State Machine* dan teknik *random shufflebag* pada *AI* musuh dalam *game Action RPG*.

Game Action RPG pada dasarnya merupakan *game* yang memiliki kebebasan dalam berinteraksi dengan musuh yang artinya tidak ada halangan untuk menyerang, menghindar berbeda dengan *game turn based rpg* dimana *game* yang ber*genre turn based rpg* memerlukan pergantian giliran untuk melakukan serangan dan dibatasi oleh waktu. *AI* musuh dalam *game* juga harus membuat pengalaman bermain menjadi lebih menyenangkan dengan begitu *AI* dalam *game* tidak boleh terlalu menyusahkan pemain dan mendukung aspek dalam *game* seperti membuat musuh lebih aktif dan berkesan tidak monoton.

Metode yang digunakan adalah *Probabilitic Finite State Machine*, yang dasarnya adalah *Finite state machine* atau *FSM*, *FSM* sendiri sudah banyak dipakai dalam *game* karena sangat baik dalam memodelkan perilaku *AI* pada *game* [1]. Kelemahan dari metode ini adalah perilaku *AI* yang statis atau mudah ditebak. Namun dengan menambahkan *probability* pada *FSM* dapat membuat *AI* tidak mudah ditebak. *Probabilistic finite state machine* sebelumnya sudah dipakai oleh [3], permainan yang dibuat adalah *game* mencocokkan gambar dengan kata dalam bahasa dayak saat dicocokkan dan benar *AI* akan melakukan perubahan ekspresi senang dengan metode *probabilistic finite state machine* dan karena dengan *probability* ekspresi senang yang dipakai tidak hanya satu serta ekspresi senang yang keluar pun dapat berbeda yang membuat *AI* tampak lebih natural. *Shuffle Bag* merupakan teknik *random* yang digunakan, idenya adalah melakukan *shuffle* isi dari sebuah tas dan mengeluarkannya satu persatu setelah itu diulang lagi [9]. Random *shufflebag* akan dipakai oleh *AI* musuh dalam pengambilan keputusan dan *probabilistic* akan dipakai untuk mengeluarkan animasi dari gerakan sehingga animasi yang keluar tidak hanya satu.

Dalam penelitian ini *AI* musuh akan dibuat dengan metode *probabilistic finite state machine* dan ditambahkan dengan teknik *random shuffle bag* karena dengan teknik ini kemungkinan untuk musuh melakukan gerakan yang sama dapat diminimalisir serta diterapkan pada *genre game* yang berbeda yaitu *action RPG* dimana *AI* akan menjadi musuh *player*, diharapkan *AI* dapat

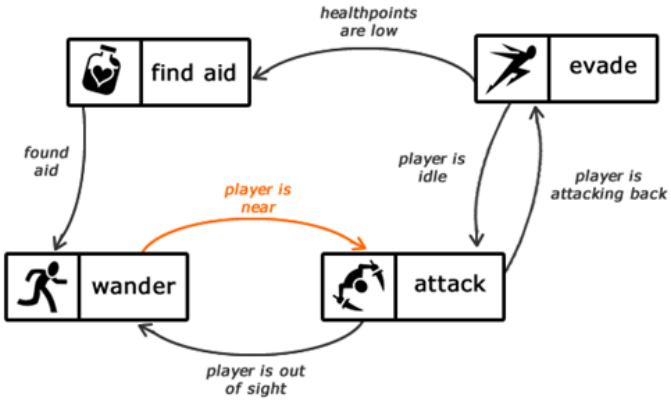
menjalankan perannya dengan baik sebagai musuh dan tidak mengulang-ulang gerakan dengan metode *probabilistic finite state machine*.

2. TINJAUAN PUSTAKA

2.1 *Finite State Machine*

Finite state machine atau *FSM* merupakan model umum yang sering digunakan untuk membuat AI karena mudah dipahami dan mudah diimplementasikan [8].

FSM biasanya digunakan untuk menggambarkan suatu *flow* dan sangat berguna untuk diterapkan dalam *AI* untuk sebuah *game*. Berikut adalah contoh *FSM* dalam sebuah *game*:



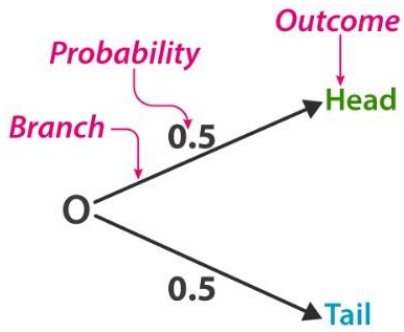
Gambar 1. Contoh *model finite state machine*

Sumber: Bevilacqua (2013)

Dapat dilihat pada Gambar 1 saat agen (*AI*) melihat *player* agen dapat melakukan serangan, namun saat tidak melihat player agen akan berkeliling, saat agen melakukan serangan dan *player* melakukan serangan balik agen dapat menghindar. Jika *hp* (nyawa) dari agen sedikit agen akan mencari obat dan setelah menemukan obat agen akan kembali berkeliling untuk mencari *player*.

2.2 *Probabilistic Finite State Machine*

Probabilistic Finite State Machine adalah metode *Finite State Machine* yang *non-deterministic*, pada dasarnya metode yang sering digunakan adalah metode *deterministic* [5] dimana metode ini hasil yang keluar adalah pasti, sedangkan dengan memberi probabilitas dalam pengambilan keputusan, hasil yang didapat belum tentu sesuai dengan perkiraan. Hasil yang dikeluarkan dapat berbeda dengan adanya probabilitas, tidak akan selalu sama.



Gambar 2. Contoh *model* sederhana dari probabilitas

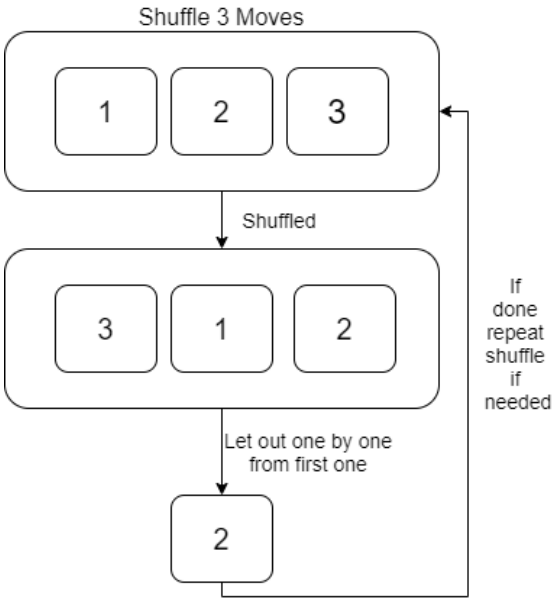
Sumber: *byjus.com* (2019)

Pada Gambar 2 merupakan contoh sederhana dari probabilitas dengan kasus lemparan koin, karena koin memiliki 2 kemungkinan yang keluar yaitu kepala dan ekor maka dapat diberi angka 50% pada setiap kemungkinannya dengan begitu dapat diketahui saat koin dilempar tidak selalu kepala yang keluar namun ekor juga dapat keluar.

2.3 *Shufflebag Random*

Shufflebag merupakan metode *random* yang sama dengan *shuffle random*, *shuffle random* adalah pengacakan urutan indeks dari sebuah *record* atau *array*. Pengacakan ini diibaratkan pengocokan pada dek kartu, dimana semua kartu dikocok sehingga susunannya teracak [4];[9]. Pada dasarnya *shufflebag* merupakan teknik untuk mengacak dan memilih dengan random index pada sebuah *array* atau *tas*.

Shufflebag akan dipakai sebagai pengambil keputusan untuk *AI* musuh, sehingga hasil dari gerakan yang keluar akan lebih merata. Idenya adalah memasukkan angka kedalam sebuah "tas" kemudian melakukan *shuffle* dan mengeluarkan angka itu satu per satu sampai habis. Setelah selesai lakukan hal yang sama berulang kali.



Gambar 3. Contoh dari *shufflebag*

Dapat dilihat pada Gambar 3 contoh dari shufflebag, misalnya musuh tipe *lord* memiliki 3 gerakan, *breath attack, melee attack, charge attack*, 3 gerakan ini akan diberi nomor 1, 2, 3 setelah itu dimasukkan kedalam sebuah "tas" dan di *shuffle* misalnya hasilnya adalah 3, 1, 2, maka musuh akan melakukan *charge attack* lalu *breath attack* dan *melee attack* kemudian ini akan di *shuffle* lagi misalnya menjadi 2, 3, 1 musuh akan melakukan *melee attack, charge attack* dan *breath attack* ini akan diulang sampai musuh mati. Sedangkan markov chain musuh setelah melakukan *breath attack* dapat melakukan *breath attack* lagi. Namun tidak menutup kemungkinan setelah *breath attack* dapat melakukan *melee attack* karena ada probabilitas. Dapat dilihat bahwa *shuffle bag* memungkinkan gerakan untuk tidak diulang-ulang.

2.4 *Action RPG*

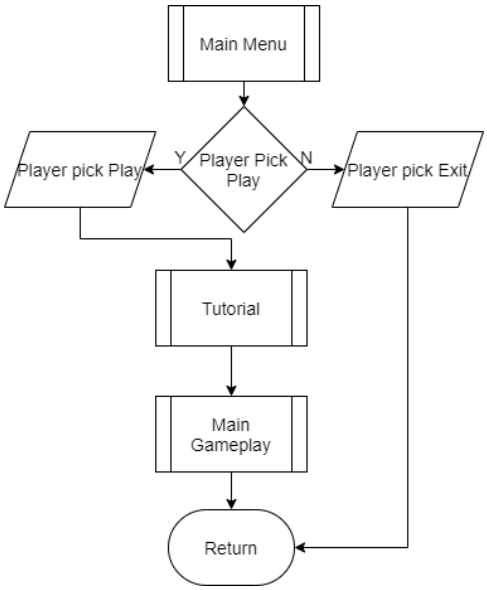
Action RPG game merupakan gabungan dari *Action game* dan *RPG game*, *Action game* adalah *game* yang membutuhkan reaksi

cepat dan koordinasi antara mata dan tangan, *RPG game* merupakan *game* yang para pemainnya akan memerankan peran [2].

Dalam *game Action RPG* biasanya memiliki lebih dari 1 karakter pemain dan musuh prajurit serta raja yang banyak [6]. Namun yang paling penting adalah kebebasan dari pergerakan dari pemain, hal itulah yang membedakan *Action RPG* dengan *genre RPG* lainnya.

3. DESAIN SISTEM

3.1 Alur Sistem *Game*



Gambar 4. Diagram alur sistem permainan.

1. *Main Menu*

Pada bagian *main menu*, pemain dapat memilih menu *Play* untuk memulai permainan.

2. *Tutorial Gameplay*

Pada bagian *tutorial*, pemain dapat mempelajari tombol-tombol yang digunakan untuk menggerakkan karakter pemain.

3. *Gameplay*

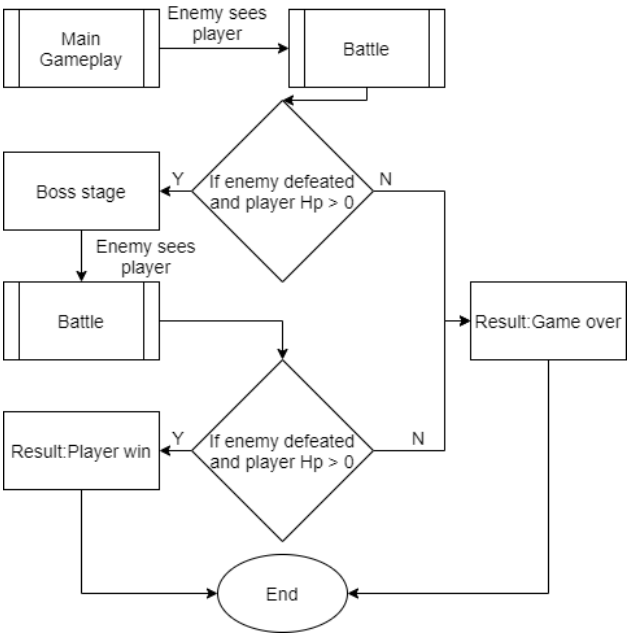
Pada bagian *gameplay*, pemain akan memainkan bagian utama dari *game* dan berusaha untuk mengalahkan musuh untuk menyelesaikan permainan.

4. *Result*

Halaman *result* akan menampilkan hasil permainan dimana pemain dinyatakan menang atau kalah.

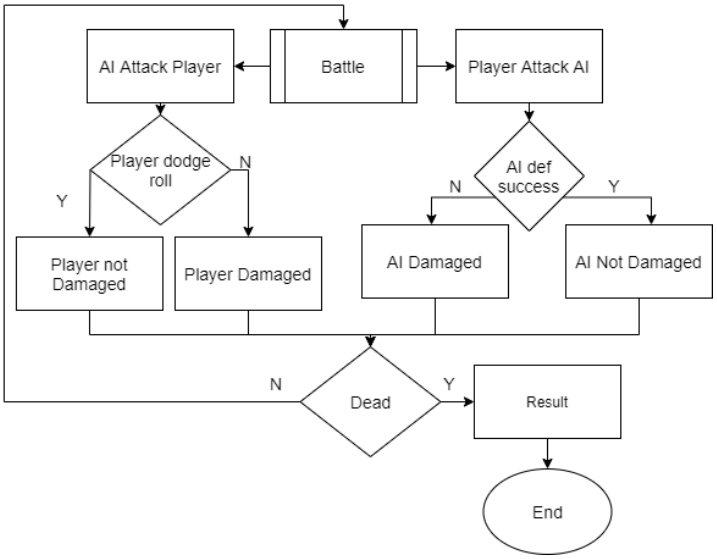
3.2 Alur *Gameplay*

Bagian ini akan menjelaskan tentang alur yang akan dialami pemain saat selesai dari *tutorial*. Dimana pada *tutorial* pemain akan dihadapkan dengan dialog untuk mengenalkan permainan dan setelah selesai pemain dapat masuk kedalam *main gameplay* yang akan dijelaskan melalui gambar berikut.



Gambar 5. Alur *gameplay*.

Gambar 5 saat memulai permainan pemain akan dihadapkan dengan musuh bertipe *minion* yang berjumlah 3 jika pemain berhasil mengalahkannya maka pemain akan disuruh menuju tempat *lord* berada yaitu *boss stage* setelah menang pemain akan dihadapkan ke *result screen* dimana pemain dapat kembali ke *main menu*, jika pemain kalah dalam *stage* pemain juga akan dihadapkan ke *result screen* dimana *player* dapat kembali ke *main menu*.



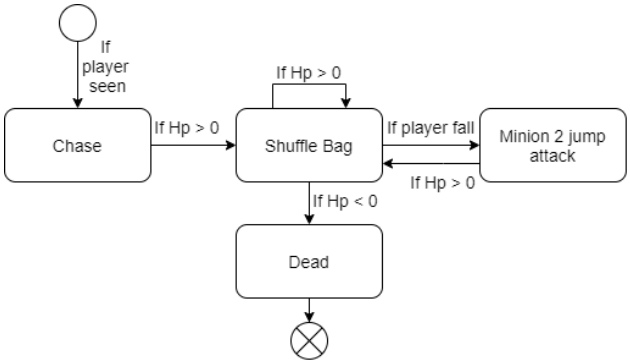
Gambar 6. Diagram alur *battle*.

Gambar 6 merupakan alur *battle* pada Gambar 5, saat *battle AI* akan menyerang pemain selama masih hidup dan pemain dapat melakukan 2 gerakan dimana gerakannya adalah menyerang *AI* dengan menembak panah dan melakukan *dodge roll* untuk menghindari serangan saat melakukan *dodge roll* pemain tidak akan terkena *damage* jika diserang *AI*.

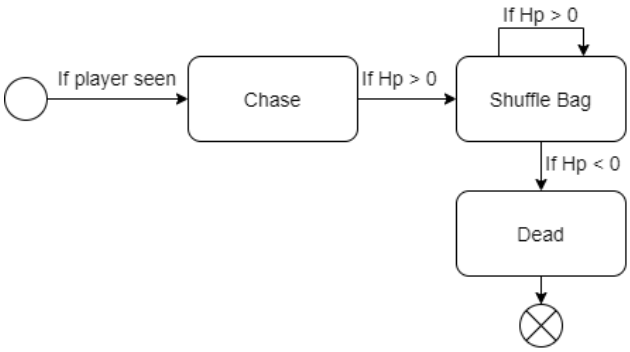
3.3 Desain *AI Minion*

Musuh tipe *minion* merupakan musuh yang akan dilawan pemain dalam *stage1* yaitu *stage* setelah *tutorial*. Jumlah *AI minion* dalam *game* ini akan ada 3 dimana 2 diantaranya menggunakan *shufflebag* dan 1 menggunakan *random* biasa. *AI minion* yang menggunakan *shufflebag* dapat bekerjasama dan dibagi menjadi 2 yaitu *minion1* dan *minion2*, *AI minion1* dan 2 akan bekerjasama saat *AI minion* 1 dan 2 melihat pemain, Jika tidak melihat maka *AI* tidak akan bekerjasama dan akan berjalan sendiri-sendiri. Gambar 7 merupakan gambar saat *AI minion* berkerjasama dan Gambar 8 saat *AI minion* tidak bekerjasama.

Pada Gambar 7 *AI minion1* dapat menjatuhkan pemain demgan melakukan serangan tendangan serangan tendangan yang dapat menjatuhkan pemain hanya dapat dilakukan oleh *AI minion1* dan hanya bisa saat *AI* bertemu dengan *AI minion2*, setelah itu AI akan melakukan serangan kerjasama (*combo*) dengan *AI minion2* yang akan melakukan serangan lompatan.



Gambar 7. *State machine AI minion* dengan kerjasama



Gambar 8. *State machine AI minion* tanpa kerjasama.

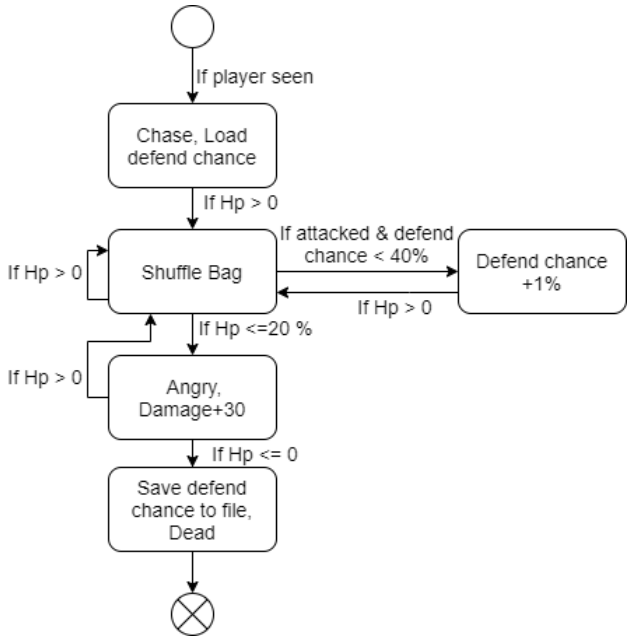
Pada Gambar 8, *AI* tidak dapat menjatuhkan pemain dan tidak akan melakukan serangan kerjasama. Saat *AI* tidak melihat pemain *AI* akan melakukan *wander* atau berkeliling untuk mencari pemain. Setelah *AI minion* melihat pemain *AI* akan melakukan *shufflebag* untuk menentukan gerakan apa yang akan keluar.

Gerakan yang dapat dilakukan oleh *AI minion* antara lain:

- *Punch attack*, merupakan serangan jarak dekat yang dapat dilakukan *minion*.
- *Kick attack*, merupakan serangan jarak dekat yang dilakukan *minion* untuk membuat pemain terjatuh, pemain hanya akan jatuh saat diserang oleh 2 *minion* karena gerakan ini ditujukan sebagai pemulai serangan kerjasama.
- *Jump attack*, merupakan serangan lompatan yang dilakukan oleh *minion*, serangan ini juga dilakukan oleh *minion* sebagai lanjutan dari serangan kerjasama, saat pemain terjatuh *minion* akan melakukan serangan ini dan serangan kerjasama akan selesai.
- *Defend*, merupakan gerakan yang dipakai *minion* untuk menghindari serangan pemain, gerakan ini akan diberi probabilitas sehingga tidak selalu berhasil.

3.4 Desain *AI Lord*

Musuh tipe terakhir yang akan dilawan pemain adalah musuh tipe *lord* sebagai *boss* (raja), *lord* dapat melakukan serangan jarak jauh yang terbagi menjadi 2 tipe dan serangan jarak dekat. *Lord* akan memiliki 2 state dimana *lord normal* dan *lord angry*, *state angry* akan keluar jika *HP* (darah) *lord* tersisa sekitar 20%. *Defend* pada *lord* akan meningkat seiring terkena serangan dan akan disimpan sehingga saat pemain melawan *lord*, *lord* dapat memiliki *defend* yang sudah lebih besar dari biasanya, tentunya *defend* akan diberi batasan sehingga tidak menyusahkan pemain.



Gambar 9. *State machine AI lord.*

Gambar 9 merupakan *state machine AI lord*, Saat pemain memasuk *stage* terakhir atau *stage2* dan memasuki area pengelihatan *AI Lord*, *lord* akan melakukan *load defend chance* dimana *lord* akan mengisi *defend chance* sesuai dengan yang pernah disimpan, jika tidak ada maka *defend chance* akan dimulai dari 0, *defend chance* akan ditambah sebanyak 1% selama *lord* menerima serangan dari pemain. Setelah itu *lord* akan melakukan *shufflebag* dan saat *lord* memiliki *HP* sekitar 20% *lord* akan memasuki *angry state* dan menambah ukuran, kecepatan dan *damage* dari serangan *lord*. Saat *lord* mati *lord* akan melakukan *save defend chance* dimana *defend chance* yang telah terisi disimpan kedalam sebuah *file*.

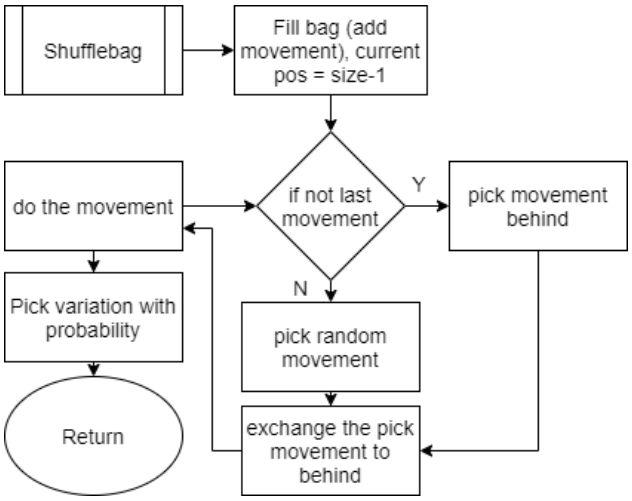
Gerakan yang dapat dilakukan oleh *AI lord* antara lain:

- *Punch attack*, merupakan serangan jarak dekat yang dapat dilakukan *lord*, pertama *lord* akan mendekati pemain dengan melakukan *chase*, setelah dalam jangkauan *lord* akan melakukan gerakan ini.

- *Range attack*, merupakan serangan jarak jauh yang dapat dilakukan *lord*, *lord* memiliki 2 tipe serangan jarak jauh yaitu tipe es yang dapat memperlambat gerakan pemain dan memberi *damage* serangan, serta tipe api yang hanya memberi *damage*.
- *Charge attack*, merupakan serangan tubrukan, *lord* akan berlari kearah pemain dan jika pemain diam *lord* akan menabrak pemain, dan pemain akan terkena damage.
- *Defend*, merupakan gerakan yang dipakai *lord* untuk menghindari serangan pemain, gerakan ini akan diberi probabilitas sehingga tidak selalu berhasil.

3.5 Desain *Shufflebag*

Shuffle bag disini merupakan metode yang akan dipakai *AI* untuk pengambilan keputusan dan akan digabung dengan *probabilistic* untuk variasi dari gerakannya. Contoh saat gerakan *melee attack* keluar akan diberi probabilitas serangan kiri atau serangan kanan sehingga gerakan *melee attack* yang keluar tidak hanya pukulan tangan kanan. Gambar 10 merupakan *flowchart* dari *shufflebag* pada Gambar 7, 8 dan 9.



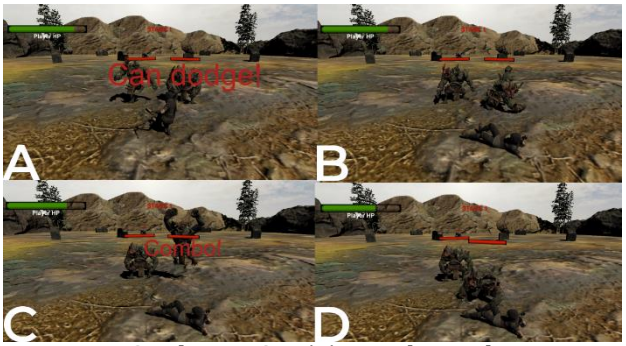
Gambar 10. *Flowchart shufflebag.*

Pada Gambar 10, Saat *AI* musuh melihat pemain baik *lord* maupun *minion AI* akan melakukan shufflebag perbedaannya adalah pada varian gerakan yang dapat dilihat pada Tabel 1 dan 2 sebelumnya, saat menjalankan *shufflebag AI* akan mengisi sebuah list dengan gerakan yang telah dibuat setelah itu *AI* akan memilih gerakan dengan *random* setelah itu gerakan yang dipilih akan ditaruh di paling belakang untuk pertama kali jalan lalu akan dipilih gerakan lainnya dan ditaruh lagi dibelakang sampai gerakan terakhir, proses ini akan dijalankan selama *AI* masih hidup. Setelah memilih gerakan *AI* akan memilih animasi gerakan yang akan keluar.

4. PENGUJIAN SISTEM

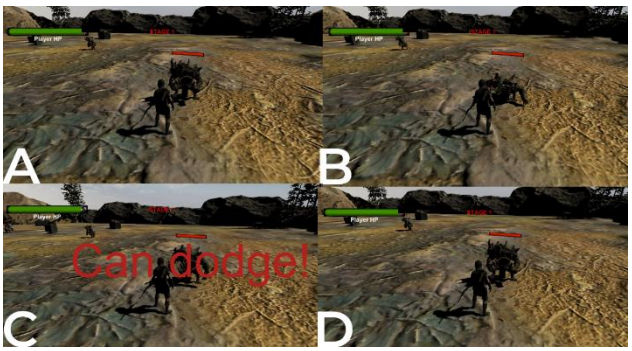
4.1 Pengujian fungsi *AI Minion*

Pengujian fungsi *minion* pertama dilakukan dengan mencoba melawan *AI* didalam *stage1*, Dilakukan pengecekkan terhadap semua fitur *AI*, terutama terhadap fitur serangan kerja sama dan pertahanan.



Gambar 11. *AI minion combo attack.*

Gambar 11 merupakan tampilan saat *AI minion* melakukan serangan kerja sama, Gambar A *minion1* akan menjatuhkan pemain dengan *kick attack*, Gambar B *minion2* akan mendekati pemain, Gambar C *minion2* akan melakukan *jump attack* dan mengeluarkan tulisan *combo* sebagai penanda, Gambar D *combo attack* selesai dilakukan dan pemain akan bangkit.



Gambar 12. *AI minion defemd success.*

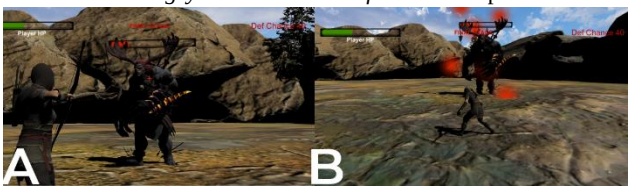
Gambar 12 merupakan tampilan saat *AI minion* melakukan *defend*, Gambar A merupakan awal saat melakukan *defend*, Gambar B *minion* akan melakukan animasi teriakan disini *chance defend* akan di *random* dan jika berhasil akan menjadi tampilan pada Gambar C dan mengeluarkan tulisan *can dodge* dan Gambar D adalah saat *defend* selesai.

Gambar 13. *AI minion attacked when defend active.*

Gambar 13 merupakan tampilan saat *AI minion* diserang dengan status *defend* aktif *damage* serangan tidak akan masuk dan akan keluar tulisan *miss*.

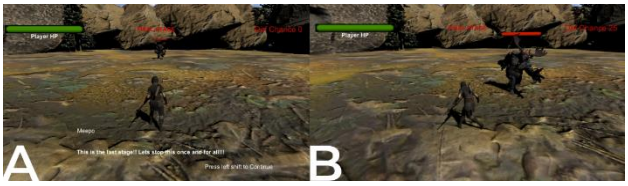
4.2 Pengujian fungsi *AI Lord*

Pengujian fungsi *lord* pertama dilakukan dengan melawan *AI* didalam *stage2*, dilakukan pengecekkan terhadap semua fitur *AI* terutama fitur *angry state* dan *save defend chance* pada *lord*.



Gambar 14. *AI lord angry state.*

Gambar 14 merupakan tampilan saat fitur *angry* aktif, Gambar A adalah bentuk *normal lord* dan Gambar B adalah bentuk *angry lord*, saat berada di *state* ini *lord* akan bertambah kuat dan cepat.



Gambar 15. *AI lord before and after defeated.*

Gambar 15 A merupakan tampilan awal saat *lord* belum pernah kalah *defend chance* akan terisi 0, kemudian saat pemain mengalahkan *lord* akan dilakukan save terhadap *defend chance* kedalam sebuah *file* dan Gambar B adalah saat *lord* sudah pernah dikalahkan dan *defend chance* akan di *load* dan diisi sesuai dengan isi *file* sebelumnya, pada gambar terisi 25.

4.3 Pengujian *Gameplay*

Pengujian dilakukan pada gerakan yang dilakukan oleh *AI*, kode gerakan dapat dilihat pada Tabel 1 untuk *AI minion* dan 2 untuk *AI lord*, Gerakan yang dilakukan dapat dilihat pada Tabel 3 pengujian gerakan dilakukan dengan memainkan game sebanyak 5x dan mencatat gerakan yang keluar pada saat melawan musuh.

Tabel 1. Kode gerakan *AI minion*.

Kode Gerakan	Gerakan
A	Melee / Punch Attack
B	Kick Attack
C	Jump Attack
D	Defend

Tabel 2. Kode gerakan *AI lord*.

Kode Gerakan	Gerakan
A	Melee / Punch Attack
B	Range Attack
C	Charge Attack
D	Defend

Tabel 3. Gerakan dalam 5x percobaan

AI	Gerakan				
AI Minion	A,D,C,B	B,A,C,D	C,B,D,A	A,D,C,B	A,B,D,C
AI Lord	B,A,C,D	A,D,B,C	B,D,A,C	A,D,C,B	C,A,D,B

Pada Tabel 3 dapat dilihat bahwa ada beberapa pola yang dapat dilakukan *AI*, pola tersebut dapat diulang oleh *AI* namun dapat dilihat bahwa pengulangan yang terjadi dalam 5x percobaan hanya ada pada *AI minion* sebanyak 1x, pengulangan pola dapat terjadi setelah ditest seperti pada gerakan *AI minion* pertama dan ke 4 sama. Untuk *AI lord* tidak ditemukan pengulangan pola namun pasti dapat terjadi karena metode yang dipakai sama, namun animasi gerakan yang keluar dapat menutupi karena saat pemain melihat gerakan *minion* pola pertama pada Tabel 3, pada pukulan atau gerakan *minion* kode A yang keluar adalah pukulan kanan dan saat pola ke empat pada Tabel 3 kode A yang keluar adalah pukulan kiri. Sehingga dapat disimpulkan bahwa gerakan yang dilakukan tidak berulang namun polanya dapat berulang.

4.4 Pengujian *Gameplay* oleh Responden

Pengujian dilakukan dengan menyebarkan *game* dengan target 15 responden dan memiliki kriteria sebagai berikut:

1. Pemain merupakan laki – laki.
2. Pemain berusia remaja-dewasa (15-35) tahun
3. Pemain merupakan orang yang suka bermain *game*.

Setelah bermain responden diminta untuk menjawab pertanyaan-pertanyaan yang bersangkutan dengan pengalaman saat bermain. Pertanyaan dan jawaban dapat dilihat pada tabel berikut ini:

Tabel 4. Tingkat kesulitan *AI Minion* menurut responden.

Pertanyaan	Sangat sulit	Cukup sulit	Biasa Saja	Cukup mudah	Sangat mudah
Tingkat kesulitan *AI minion*	1	6	7	1	0

Tabel 5. Tingkat kesulitan *AI Lord* menurut responden.

Pertanyaan	Sangat sulit	Cukup sulit	Biasa Saja	Cukup mudah	Sangat mudah
Tingkat kesulitan *AI lord*	12	3	0	0	0

Tabel 6. Jumlah responden yang mengulang bermain untuk memenangkan *game*.

Pertanyaan	Ya	Tidak
Apakah anda sempat mengulang untuk mengalahkan musuh di dalam *game*?	15	0

Tabel 7. Jumlah pengulangan yang dilakukan responden untuk menang

Pertanyaan	1x	2x	3x	4x	5x	>5x
Berapa kali anda mengulang sampai bisa menang?	0	3	7	2	2	1

Tabel 8. Perbandingan *AI* yang membuat responden kesulitan

Pertanyaan	AI Lord	AI Minion
AI yang manakah yang membuat anda lebih kesulitan saat bermain?	15	0

Berdasarkan jawaban dari responden pada Tabel diatas dapat disimpulkan *game* cukup sulit untuk dimenangkan oleh para responden, jawaban pada Tabel 4 dan 5, semua termotivasi untuk memenangkan, jawaban Tabel 6 dan 7. *AI Lord* membuat 15 orang responden kesusahan, jawaban Tabel 8.

5. KESIMPULAN

Berdasarkan hasil pengujian dan respon dari para responden dapat disimpulkan bahwa:

1. Teknik *random shufflebag* dalam pembuatan keputusan musuh didalam *game* dapat membuat musuh tidak mengulang-ulang gerakan dan membuat *AI* tidak monoton.
2. *AI* didalam *game* cukup sulit untuk dikalahkan dan membuat pemain termotivasi untuk memenangkan permainan. Karena *game* tidak dapat diselesaikan dengan sekali coba berdasarkan data dari 15 orang tidak ada yang menang dengan 1 kali percobaan.
3. *AI Lord* merupakan *AI* paling sulit dikalahkan karena saat *state angry* para responden kaget dan tidak siap terhadap serangannya karena 15 orang responden memilih *AI Lord* lebih susah daripada *AI Minion*. *AI* yang memiliki *state* yang berubah sesuai dengan permainan terbukti lebih menarik dan menantang.

6. DAFTAR PUSTAKA

[1] Ekawati Yulsilviana, & Hanifah Ekawati. 2020. Penerapan metode finite state machine (FSM) pada game agent legenda anak borneo. *Sebatik*, *23*(1), 116–123. https://jurnal.wicida.ac.id/index.php/sebatik/article/view/453

[2] Kristo Radion Purba, Rini Nur Hasanah, & M. Aziz Muslim. 2013. Implementasi Logika Fuzzy Untuk Mengatur Perilaku Musuh dalam Game Bertipe Action-RPG. Jurnal EECCIS, 7(1), 15–20. https://jurnaleeccis.ub.ac.id/index.php/eeccis/article/view/196

[3] Lailiyah, S., Yunita, Y., Mallala, S., & Andrea, R. 2019. Probabilitas dalam Finite State Machine Agen Cerdas Edugame "Ajut-Ajut Kids." *Jurnal Nasional Teknik Elektro Dan Teknologi Informasi (JNTETI)*, *8*(2), 151. https://doi.org/10.22146/jnteti.v8i2.504

[4] Marsaglia, G., & Zaman, A. 1991. A New Class of Random Number Generators. The Annals of Applied Probability, 1(3), 462–480. https://doi.org/10.1214/aoap/1177005878

[5] *Probabilistic State Machines | AI Game Development: Synthetic Creatures with Learning and Reactive Behaviors*. 2017. Flylib.Com. https://flylib.com/books/en/2.71.1.298/1/

[6] Purba, K. R. 2016. Optimization of AI Tactic in Action-RPG Game. *Proceedings of Second International Conference on Electrical Systems, Technology and Information 2015 (ICESTI 2015)*, 131–137. https://doi.org/10.1007/978-981-287-988-2_14

[7] Simonov, A., Zagarskikh, A., & Fedorov, V. 2019. Applying Behavior characteristics to decision-making process to create believable game AI. *Procedia Computer Science*, *156*, 404–413. https://doi.org/10.1016/j.procs.2019.08.222

[8] Tito Bimantoro, & Hanny Haryanto. (2016). Pemodelan Perilaku Musuh Menggunakan Finite State Machine (FSM) Pada Game Pengenalan Unsur Kimia. *Journal of Applied Intelligent System*, *1*(3), 210–219. https://doi.org/10.33633/jais.v1i3.1254

[9] Yusnita, A., Sefty Wijayanti, & Putri Alysia Felita. 2017. Implementasi Algoritma Shuffle Random Pada Edugame Magic Time Berbasis Universal Windows Platform (UWP). Prosiding SNITT POLTEKBA, 2(1), 203–208. https://jurnal.poltekba.ac.id/index.php/prosiding/article/view/399