OPTIMALISASI *ENEMY BEHAVIOUR* MENGGUNAKAN *LAYERED ANIMATION TRIGGER* PADA APLIKASI PERMAINAN EDUKATIF "NOKIDNAP"

SKRIPSI

Diajukan sebagai salah satu persyaratan dalam menyelesaikan studi untuk memperoleh gelar Sarjana Ilmu Komputer (S. Kom) dalam bidang Ilmu Komputer

PETRUS MARCELINO H. TAMPUBOLON
201401127

PROGRAM STUDI S-1 ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS SUMATERA UTARA
MEDAN
2024

OPTIMALISASI *ENEMY BEHAVIOUR* MENGGUNAKAN *LAYERED ANIMATION TRIGGER* PADA APLIKASI PERMAINAN EDUKATIF "NOKIDNAP"

SKRIPSI

Diajukan sebagai salah satu persyaratan dalam menyelesaikan studi untuk memperoleh gelar Sarjana Ilmu Komputer (S. Kom) dalam bidang Ilmu Komputer

PETRUS MARCELINO H. TAMPUBOLON
201401127

PROGRAM STUDI S-1 ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI
UNIVERSITAS SUMATERA UTARA
MEDAN
2024

LEMBAR PERSETUJUAN

Judul	:	OPTIMALISASI *ENEMY BEHAVIOUR* MENGGUNAKAN *LAYERED ANIMATION TRIGGER* PADA APLIKASI PERMAINAN "NO KIDNAP"
Kategori	:	SKRIPSI
Nama	:	PETRUS MARCELINO H. TAMPUBOLON
Nomor Induk Mahasiswa	:	201401127
Program Studi	:	SARJANA (S-1) ILMU KOMPUTER
Fakultas	:	ILMU KOMPUTER DAN TEKNOLOGI INFORMASI UNIVERSITAS SUMATERA UTARA
Tanggal Sidang	:	Medan, 27 Mei 2024
Komisi Pembimbing	:	

Dosen Pembimbing II

Dr. Maya Silvi Lydia, B.Sc., M.Sc.
NIP. 197401272002122001

Dosen Pembimbing I

Dr. Jos Timanta Tarigan, S.Kom., M.Sc.
NIP. 198501262015041001

Diketahui/Disetujui oleh
Program Studi S-1 Ilmu Komputer
Ketua,

Dr. Amalia S.T., M.T.
NIP. 197812212014042001

LEMBAR PERNYATAAN

Yang bertanda tangan dibawah ini:

Nama : Petrus Marcelino H. Tampubolon

NIM : 201401127

Jurusan : Ilmu Komputer

Fakultas : Ilmu Komputer dan Teknologi Informasi

Menyatakan dengan ini sesungguhnya bahwa skripsi berjudul "Optimalisasi *Enemy Behaviour* Menggunakan *Layered Animation Trigger* pada Aplikasi Permainan Edukatif "NoKidnap" adalah betul-betul karya saya sendiri, bukan plagiat dan tidak dibuatkan oleh orang lain. Skripsi ini saya hasilkan melalui proses penelitian, bimbingan dan diskusi. Semua kutipan yang diperoleh telah disertai identitas sumbernya dengan cara yang sebagaimana lazimnya dalam penulisan karya ilmiah.

Apabila di kemudian hari terbukti bahwa pernyataan ini tidak benar, maka saya bersedia menerima sanksi akademik berupa pencabutan skripsi dan gelar yang diperoleh dari skripsi tersebut.

Medan, 27 Mei 2024

Yang memberi pernyataan

Petrus Marcelino H. Tampubolon

NIM: 201401127

ABSTRAK

Penelitian ini bertujuan untuk meneliti aplikasi permainan edukatif “NoKidnap”, yakni sebuah aplikasi permainan berbasis *mobile* android yang bertujuan dalam mengajarkan anak-anak dalam mencegah pra-indikasi penculikan anak. Dalam pengembangan aplikasi permainan edukatif "NoKidnap," dibutuhkan sebuah penelitian yang bertujuan untuk meningkatkan perilaku musuh dengan mengimplementasikan konsep *layered animation trigger* pada tingkat animasi. Dengan memadukan teknologi animasi tingkat lanjut dan algoritma pencarian jalur A* berdasarkan penggunakan modul NavMesh Agent, penelitian ini berfokus untuk menciptakan pengalaman bermain pengguna (*user experience*) yang lebih adaptif dan responsif. Prototipe *game* yang dihasilkan pun didesain untuk anak-anak usia 7-11 tahun dan memanfaatkan algoritma A* untuk meningkatkan navigasi musuh. Dengan menggunakan metode *experimental* berbasis GDLC (*Game Development Lifecycle*), diharapkan sistem *layered animation trigger* mampu diterapkan untuk menciptakan animasi musuh yang dinamis dan responsif terhadap interaksi dengan pemain. Sistem ini bekerja dengan melakukan proses modulasi secara berlapis di setiap bagian animasi. Setiap modulasi akan diurutkan berdasarkan parameter terpisah seperti jarak jauh (*long-distance*), menengah (*medium-distance*), dan dekat (*short-distance*). Selanjutnya, *usability testing* dilakukan dengan partisipan anak-anak usia 7-11 tahun untuk mengevaluasi efektivitas dan daya tarik prototipe *game* yang dihasilkan. Hasil penelitian ini menunjukkan bahwa pengalaman bermain dan respons terhadap perilaku penculik dalam *game* "NoKidnap" menunjukkan bahwa, dibandingkan dengan AI konvensional seperti FSM, respons *play-tester* terhadap *game* secara keseluruhan dan adanya AI *Layered Animation Trigger* adalah relatif baik atau positif. Meskipun tidak mencapai kategori "Sangat Tinggi", respons *play-tester* tetap positif.

Kata kunci: “NoKidnap”, *layered animation trigger*, NavMesh Agent, *game, user experience,* anak-anak.

OPTIMIZATION OF ENEMY BEHAVIOR USING LAYERED ANIMATION TRIGGER IN THE EDUCATIVE GAME APPLICATION "NOKIDNAP"

ABSTRACT

This research aims to examine the educational game application "NoKidnap", namely, an Android mobile-based game application which aims to teach children to prevent pre-indications of child abduction. In developing the educational game application "NoKidnap," research is needed that aims to improve enemy behavior by implementing the concept of layered animation triggers at the animation level. By combining advanced animation technology and the A* path-finding algorithm based on the use of the NavMesh Agent module, this research focuses on creating a more adaptive and responsive user gaming experience. The resulting game prototype was designed for children aged 7–11 years and utilized the A* algorithm to improve enemy navigation. By using an experimental method based on GDLC (Game Development Lifecycle), it is hoped that the layered animation trigger system can be applied to create enemy animations that are dynamic and responsive to interactions with players. This system works by carrying out a layered modulation process in each part of the animation. Each modulation will be sorted based on separate parameters such as long-distance, medium-distance, and short-distance. Finally, usability testing was carried out with child participants aged 7–11 years to evaluate the effectiveness and attractiveness of the resulting game prototype. The results of this study indicate that the gaming experience and responses to the kidnapper's behavior in the game "NoKidnap" show that, compared to conventional AI such as FSM, play-testers' overall responses to the game and the presence of the AI Layered Animation Trigger were relatively favorable or positive. Although it did not reach the "Very High" category, the play-testers' responses remained positive.

Keywords : “NoKidnap”, *layered animation trigger*, NavMesh Agent, *game, user experience, child*.

KATA PENGANTAR

Puji syukur peneliti ucapkan kepada Allah SWT atas rahmat dan karunia-Nya peneliti dapat menyelesaikan skripsi yang berjudul "**Optimalisasi *Enemy Behaviour* Menggunakan *Layered Animation Trigger* Pada Aplikasi Permainan "NoKidnap"**". Pada kesempatan ini peneliti mengucapkan rasa terima kasih kepada semua pihak yang ikut membantu dalam proses pembuatan skripsi ini, terutama kepada:

Ucapan terima kasih penulis sampaikan kepada:

1. Tuhan Yang Maha Esa yang selalu melimpahkan kasih sukacita, berkat, dan rahmat-NYA kepada penulis, sehingga penulis dapat menyelesaikan skripsi ini.
2. Ayahanda Johannes Tua Pardamean Tampubolon, Ibunda Zentina Sitorus dan seluruh keluarga yang selalu mendoakan, memberi perhatian serta kasih sayang kepada penulis.
3. Ibu Dr. Amalia S.T., M.T. selaku Ketua Program Studi S1 Ilmu Komputer Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.
4. Sahabat dan kerabat dekat saya Benhard Tampubolon, Edwin Brayen Lumbantobing, Willy Rajagukguk, Idris Siahaan, Ramlan Siahaan, dan Rian Siahaan yang telah memberikan inspirasi dan semangat dikala penulisan skripsi.
5. Kerabat jauh Rustam Effendi Tampubolon, serta wali saya Romulus Tampubolon.
6. Bapak Dr. Jos Timanta Tarigan, S.Kom, M.Sc. selaku dosen pembimbing I dan Ibu Dr. Maya Silvi Lydia, B.Sc., M.Sc. selaku dosen pembimbing II yang telah memberikan bimbingan, kritik, dan saran kepada penulis dalam menyelesaikan skripsi ini.
7. Ibu Sri Melvani Hardi S.Kom., M.Kom selaku Sekretaris Program Studi S1 Ilmu Komputer Universitas Sumatera Utara.
8. Seluruh tenaga pengajar, *staff*, pegawai, serta seluruh Civitas Akademika di Fakultas Ilmu Komputer dan Teknologi Informasi USU.
9. Stambuk 2019 dan 2020, terutama Abangda Muhammad Ridho Harahap,

Wahyu Hidayat Siregar, dan Venerio Uvandy, yang telah memberikan semangat, nasihat dan pengetahuan, serta sebagai teman diskusi.

10. Seluruh teman-teman yang tidak dapat saya sebutkan namanya satu persatu yang telah memberikan bantuan moral, nasihat, semangat, serta berbagi canda tawa kepada penulis.
11. Semua pihak yang terlibat langsung atau tidak langsung yang tidak dapat dituliskan satu per satu.

Semoga Allah SWT selalu memberikan kasih sayangnya kepada semua pihak yang telah berperan penting bagi penulis dalam menyelesaikan skripsi ini. semoga skripsi ini bermanfaat untuk pribadi maupun pembaca untuk mencerdaskan kehidupan bangsa.

Medan, 27 Mei 2024

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

LAMPIRAN

BAB I
PENDAHULUAN

Bab ini akan menjelaskan mengenai latar belakang penelitian judul skripsi "Optimalisasi *Enemy Behaviour* Menggunakan *Layered Animation Trigger* Pada Aplikasi Permainan Edukatif "NoKidnap"", rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metode penelitian, dan sistematika penulisan skripsi.

1.1 Latar Belakang

Selama beberapa kurun waktu terakhir, perkembangan teknologi digital telah berkembang sangat pesat. Berbagai kemajuan teknologi di berbagai aspek kehidupan, telah mempermudah peranan manusia dalam menciptakan produk berupa barang dan jasa. Tidak sedikit dari produk tersebut juga memiliki peranan penting bagi bertumbuhnya perekonomian dan arus digitalisasi masyarakat. Hal ini dikarenakan, bahwa peranan teknologi dapat membantu manusia dalam menghasilkan produk berupa barang dan jasa tapi juga mengurangi tingkat efektivitas kesalahan manusia atau *human error*. Menurut Knight (2015), AI (*Artificial Intelligence*) berusaha melakukan tugas tertentu ratusan atau bahkan ribuan kali. Sebagian besar upaya menghasilkan kegagalan, namun dengan setiap keberhasilan, mesin perlahan-lahan belajar mendukung perilaku yang menyertai setiap upaya yang berhasil. *Artificial intelligence* juga dirasa telah banyak membantu kegiatan manusia dalam berbagai bidang seperti bisnis, perawatan kesehatan, transportasi, dan sebagainya (Enholm, Papagiannidis, Mikalef, & Krogstie, 2021); (Manoharan, 2019); (Ramesh, Kambhampati, Monson, & Drew, 2004). Berbagai peranan AI telah merambah ke berbagai bidang seperti dalam bidang bisnis, kesehatan, dan juga pendidikan. Hal ini dinilai oleh berbagai ahli dapat membuka peranan penting bagi perkembangan digital selanjutnya yang dapat berdampak baik bagi pengembangan teknologi kecerdasan buatan. Menurut (Anagnostopoulos et al., 2018), Sistem tutor AI digunakan untuk memberikan bimbingan dan umpan balik

kepada siswa secara individual, sesuai dengan kebutuhan dan tingkat pemahaman mereka (Anagnostopoulos, Vakali and Hadjiefthymiades, 2018). Menurut Janosov dan Szabo, kecerdasan buatan juga dapat digunakan untuk menganalisis data keuangan, mengidentifikasi pola, dan memprediksi pergerakan pasar (Janosov and Szabo, 2019). Selain itu dalam bidang kesehatan, kecerdasan buatan juga telah dikembangkan untuk mendeteksi kanker kulit dengan akurasi yang sebanding dengan dokter kulit berpengalaman (Esteva et al., 2017).

Penerapan teknologi *Artificial Intelligence* (AI) dalam *video game* bukanlah suatu hal yang baru. Sejak dekade 1950-an, kecerdasan buatan telah menjadi komponen inti dalam pengembangan *game* berbasis komputer. Umumnya, teknologi ini dimanfaatkan untuk menciptakan karakter non-pemain atau NPC (*Non-Playable Character*), seperti musuh atau lawan yang dihadapi oleh pemain. Beberapa perintis komputasi (seperti Turing, von Neumann, dan Shannon) juga merupakan perintis dalam bidang kecerdasan buatan (AI) pada tahap awal. Turing, khususnya, telah menjadi salah satu figur utama dalam bidang ini berkat sebuah makalah filosofis yang dipublikasikan pada tahun 1950. Kemudian, secara bertahap sepanjang tahun 1980-an hingga awal tahun 1990-an, inti dari kecerdasan buatan statistik modern mulai mencakup *Bayes Network*, *Support Vector Machine* (SVM), dan *Gaussian Process*. Secara bertahap, para peneliti AI utama menyadari bahwa kunci dari pendekatan baru ini tidak hanya terletak pada keterhubungan dengan dunia alam, tetapi juga pada kemampuan untuk mengatasi ketidakpastian dan pentingnya dalam menyelesaikan masalah dunia nyata. Perubahan besar dalam AI lebih dari sekadar tren. Hal ini telah menjadikan AI sebagai teknologi kunci yang relevan untuk memecahkan masalah dunia nyata. Penggunaan AI dalam permainan komputer *(computer games)* mengambil perubahan besar dalam proses memecahkan berbagai masalah. Permainan komputer telah terkait dengan kecerdasan buatan (AI) sejak program pertama dirancang untuk bermain catur (Shannon 1950). Tantangan untuk mengalahkan pemain ahli manusia dalam permainan strategi berbasis aturan seperti Catur, dan PokerGo telah sangat memajukan domain penelitian AI, mempengaruhi terobosan di bidang kecerdasan komputasional, algoritma, pembelajaran mesin, dan teori permainan kombinatorial Fujita dan Wu (2012). Sebaliknya, metode AI baru tersebut telah digunakan dalam permainan komputer, misalnya untuk meningkatkan realisme

grafis, menghasilkan tingkat permainan, pemandangan, dan alur cerita, menetapkan profil pemain, menyeimbangkan kompleksitas, atau menambahkan perilaku cerdas pada karakter non-pemain (NPC; Yannakakis dan Togelius 2015, 2018). Dalam konteks pendidikan, peranan AI dalam *game* merupakan salah satu metode efektif dalam meningkatkan pengalaman belajar sambil bermain atau yang lebih dikenal dengan *game-based learning.* Menurut (Raharjo dkk., 2022), menyediakan edukasi melalui permainan akan memberikan pengalaman menarik bagi individu yang mendapatkannya. Hal ini dikarenakan, tujuan dari bermain *game* itu sendiri adalah untuk membuat pemain melepaskan kelelahan mereka, bersenang-senang, dan menghilangkan kebosanan-kebosanan mereka. Oleh karena itu, untuk meningkatkan pengalaman permainan yang menarik, sistem kecerdasan buatan dapat ditambahkan pada *game*.

Dalam mengembangkan *game* agar dapat dimainkan semenarik mungkin, dibutuhkan suatu konsep yang melatarbelakangi ide pembuatan *game*. Salah satu aspek penting dalam pengembangan aplikasi permainan edukatif adalah pengaturan pergerakan karakter (*enemy behaviour*) dalam lingkungan virtual. Untuk menciptakan pengalaman bermain yang realistis dan menantang, diperlukan navigasi yang efisien dan cerdas. *NavMesh Agent*, yang merupakan salah satu komponen penting dalam pergerakan karakter pada platform permainan, digunakan untuk menghasilkan jalur pergerakan yang optimal berdasarkan lingkungan virtual yang diberikan. Perpaduan antara pengaturan sistem pergerakan karakter dan algoritma pemicu animasi berlapis (*layered animation trigger*) memungkinkan ide pembuatan *game* pada aplikasi semakin menarik.

Namun, dalam konteks aplikasi permainan edukatif yang menekankan pencegahan penculikan anak, diperlukan optimalisasi khusus pada *NavMesh Agent* dan *layered animation trigger*. Hal ini dikarenakan bahwa NavMesh Agent dan *animation trigger* saling berkaitan dan memunculkan pengalaman bermain yang kompleks. Penggunaan algoritma A* (*A-star*) telah terbukti menjadi metode yang efektif dalam menentukan jalur terpendek antara dua titik dalam lingkungan virtual Hart et al. (1968). Dengan menerapkan algoritma A* pada *NavMesh Agent*, dapat dicapai navigasi yang lebih efisien dan cerdas, sehingga meningkatkan realisme dan tantangan dalam aplikasi permainan edukatif "No Kidnap". Sedangkan, *layered animation trigger* merujuk pada suatu mekanisme yang memicu perubahan keadaan

animasi karakter dalam permainan berdasarkan parameter berlapis. Parameter berlapis ini memungkinkan pengembang *game* (*game developer*) dalam menentukan setiap perubahan animasi hanya berdasarkan pemicu animasi berlapis (*layered animation trigger*). Dalam hal aplikasi "No Kidnap," *layered animation trigger* dapat digunakan untuk menyimulasikan situasi-situasi tertentu, seperti respons karakter terhadap ancaman penculikan atau interaksi dengan lingkungan sekitar. Penerapan *layered animation trigger* dapat membantu membantu pemain tentang konteks keamanan dan melibatkan mereka secara lebih mendalam dalam pengalaman permainan, selain itu secara khusus juga konsep ini dapat membantu para pengembang *game* dalam menyimulasikan animasi musuh yang lebih efektif.

Dalam mendesain *layered animation trigger*, penting untuk mempertimbangkan aspek-aspek psikologis dan pendidikan anak-anak. Animasi yang baik dapat memberikan informasi visual yang mudah dipahami, memberikan petunjuk, atau merangsang respons emosional yang sesuai dengan konteks keamanan. Selain itu, integrasi *layered animation trigger* dengan deteksi pemain yang dioptimalkan dapat menciptakan pengalaman bermain yang lebih realistis dan menantang.

Pengembangan *animation trigger* yang terfokus pada situasi penculikan anak dapat meningkatkan efektivitas aplikasi "No Kidnap" dalam menyampaikan pesan keamanan kepada pemain. Animasi yang tepat pada momen-momen kritis dapat memperkuat pemahaman anak-anak tentang bahaya penculikan dan tindakan yang perlu diambil. Dengan menggabungkan aspek ini dalam penelitian, kita dapat menciptakan suatu solusi yang holistik untuk meningkatkan kualitas keseluruhan pengalaman bermain dalam konteks pencegahan penculikan anak.

Dalam konteks ini, penelitian ini bertujuan untuk mengoptimalkan perilaku musuh *(enemy behaviour)* pada aplikasi permainan edukatif "No Kidnap" dengan menerapkan algoritma A* pada modul *Navmesh Agent*, serta mempertahankan pengaruhnya terhadap *layered animation trigger*. Melalui penelitian ini juga, diharapkan bahwa pengoptimalan tidak hanya terfokus pada pergerakan karakter dengan algoritma A*, tetapi juga mencakup implementasi *layered trigger animation* yang memberikan kontribusi tidak hanya pada aplikasi permainan edukatif "No Kidnap" tetapi juga bagi perkembangan industri *game* di seluruh dunia. Dengan demikian, anak-anak dapat lebih aktif terlibat dan belajar secara efektif tentang situasi keamanan, sambil menikmati pengalaman bermain yang menyenangkan dan

mendidik.

Ada beberapa tahapan dalam proses implementasi algoritma A* pada aplikasi, tahap pertama adalah pengumpulan data lingkungan. Hal ini melibatkan identifikasi dan pengumpulan data lingkungan virtual yang akan digunakan dalam aplikasi permainan edukatif. Data lingkungan meliputi informasi tentang rintangan, jalan, dan area yang dapat dilalui oleh karakter dalam permainan. Setelah itu, tahap kedua melibatkan pembuatan *NavMesh*. Tahap ini meliputi konversi data lingkungan menjadi *NavMesh*, yaitu representasi grafis dari lingkungan yang terdiri dari poligon-poligon yang dapat dilalui oleh karakter. *NavMesh* akan menjadi dasar untuk pergerakan *NavMesh Agent* dalam lingkungan virtual. Algoritma A* akan diinisialisasi dengan menggunakan node awal dan node tujuan. Kemudian, algoritma A* akan dieksekusi untuk mencari jalur terpendek antara posisi awal dan tujuan dengan mempertimbangkan jarak dan estimasi biaya terbaik. Setelah mendapatkan jalur terpendek, jalur tersebut akan dipilih sebagai panduan pergerakan *NavMesh Agent*. Kemudian, modul tersebut akan diintegrasikan dengan *layered trigger animation. Layered trigger animation* akan dipasangkan ke dalam sebuah *enemy game object* (komponen musuh), sehingga pada sisi pengembangan akan didapati sebuah pemicu animasi berlapis yang dapat dipasangkan beberapa parameter sesuai kebutuhan pengembangan. Selanjutnya, teknik tambahan seperti *smooth pathfinding* dapat diterapkan untuk mengoptimalkan pergerakan *NavMesh Agent* agar lebih halus dan alami. Setelah tahap optimalisasi, hasilnya akan diimplementasikan ke dalam aplikasi permainan edukatif "No Kidnap" dan dilakukan pengujian untuk memastikan *NavMesh Agent* dapat bergerak secara efisien dan cerdas dalam menghindari rintangan serta mencapai tujuan dengan jalur terpendek menggunakan pemicu animasi berlapis (*layered trigger animation*). Melalui tahapan-tahapan ini, diharapkan dapat meningkatkan kualitas pergerakan *NavMesh Agent* dalam aplikasi permainan edukatif "No Kidnap" dengan pencegahan penculikan anak sebagai fokus utama.

Hingga saat ini, belum ada penelitian lanjutan yang mempelajari penggunaan *layered trigger animation* dalam konteks aplikasi permainan edukatif, terutama yang terkait dengan masalah penculikan anak. Perkembangan teknologi dapat mempengaruhi kemampuan dan keterampilan anak dalam memahami keadaan lingkungan sosial masyarakat terutama dalam konteks edukasi.

1.2 Rumusan Masalah

Minimnya efektivitas penggunaan pemicu animasi berlapis (*layered trigger animation*) dalam konteks aplikasi permainan edukatif pada *game* NoKidnap, terutama yang terkait dengan masalah penculikan anak-anak (7-11 tahun), mitigasi bahaya penculikan anak, dan serangkaian tahapan dalam menghindarinya yang membutuhkan logika animasi yang sesuai.

1.3 Batasan Masalah

Berdasarkan masalah dalam penelitian ini adalah sebagai berikut:

1) Menggunakan algoritma A*
2) Referensi data yang digunakan diambil dari jurnal yang berkaitan dengan penculikan, kecerdasan buatan (*artificial intelligence*), serta algoritma A*
3) Penyusunan program hanya ditujukan untuk anak-anak usia 7-11 tahun
4) Platform yang dipakai adalah berbasis mobile Android
5) Struktur pemicu animasi berlapis (*layered animation trigger*) memuat 3 kategori jarak (*outter layer*, *middle layer*, *base layer*).
6) Setiap lapisan pada masing-masing jarak dipetakan secara terrpisah menggunakan *script*.
7) Program dirancang dengan menggunakan bahasa C# dengan implementasi modul NavMesh Agent.

1.4 Tujuan Penelitian

Tujuan dilakukannya penelitian ini adalah:

1) Mengukur efektivitas penerapan konsep *layered animation trigger* menggunakan kombinasi modul *Navmesh Agent* dan Algoritma A*(*A-star*).
2) Memahami metode peningkatan optimasi perilaku musuh (*enemy behaviour*) pada aplikasi permainan edukatif “NoKidnap”.

1.5 Manfaat Penelitian

Penelitian optimasi perilaku musuh (*enemy behaviour*) menggunakan konsep *layered animation trigger* berbasis *mobile* Android ini diharapkan dapat memberikan manfaat yang baik antara lain:

1.5.1 Memberikan informasi tentang penerapan algoritma A* dan informasi

tentang tingkah laku penculik dalam aplikasi permainan menggunakan konsep kecerdasan buatan (*artificial intelligence*).

1.5.2 Mengembangkan panduan praktis bagi pengembang *game* atau aplikasi edukasi yang ingin mengimplementasikan pendekatan *layered animation trigger* dalam gamifikasi untuk tujuan edukasi anak.

1.5.3 Memberikan kontribusi bagi pengembangan ilmu pengetahuan dan teknologi di bidang pengembangan aplikasi permainan.

1.6 Metodologi Penelitian

Metodologi penelitian yang dilakukan dalam penelitian ini adalah:

1) Studi Pustaka

   Dalam penelitian ini, penulis menggunakan metode studi pustaka atau studi literatur untuk meninjau, dan mengumpulkan berbagai referensi dari buku-buku, jurnal, laporan-laporan dan tinjauan pustaka lainnya yang memiliki hubungan dengan penelitian yang akan dilakukan.

2) Analisis dan Perancangan Sistem

   Analisis sistem dilakukan menggunakan Flowchart dan Design Thinking, serta perancangan sistem dilakukan menggunakan metode *Game Development Life Cycle* (GDLC).

3) Implementasi

   Implementasi dari sistem yang akan dilakukan dibangun sesuai dengan perancangan yang dibuat dengan bahasa pemrograman C# yang berbasis mobile menggunakan arsitektur Unity Engine.

4) Pengujian

   Sistem yang telah dibuat akan diuji coba untuk melihat dan memastikan bahwa sistem tersebut berjalan dengan semestinya.

5) Dokumentasi

   Setelah implementasi, maka penulis akan membuat dokumentasi atau laporan dan kesimpulan akhir dari hasil akhir analisa dan pengujian dalam bentuk skripsi.

1.7 Sistematika Penulisan

Sistematika penulisan skripsi ini terdiri dari beberapa bagian utama yang

dijelaskan sebagai berikut.

BAB 1 PENDAHULUAN

Bab ini akan menjelaskan mengenai latar belakang penelitian judul skripsi "Optimalisasi *Enemy Behaviour* Menggunakan *Layered Animation Trigger* Pada Aplikasi Permainan Edukatif "NoKidnap"", rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metode penelitian, dan sistematika penulisan skripsi.

BAB 2 LANDASAN TEORI

Bab ini membahas tentang landasan teori pengembangan aplikasi, kecerdasan buatan(*artificial intelligence*), *Non-Playable* & *Playable Character*, Algoritma A*(A-star), Modul NavMesh Agent, Permainan Edukatif, Pembacaan Perilaku Musuh (*Enemy Behaviour*), *Animation Rigging*, dan *Animation Trigger*.

BAB 3 ANALISIS DAN PERANCANGAN

Bab ini membahas tentang analisis mengenai alur kerja dari *layered animation trigger* pada aplikasi permainan "NoKidnap" menggunakan flowchart dan design thinking serta perancangan sistem yang dibuat menggunakan metode *Research Experimental* berbasis *Game Development Lifecycle* (GDLC).

BAB 4 IMPLEMENTASI DAN PENGUJIAN

Bab ini membahas tentang pembuatan sistem sesuai dengan analisis dan perancangan. Kemudian melakukan pengujian sistem apakah sistem sesuai dengan yang dirancang sebelumnya.

BAB 5 KESIMPULAN DAN SARAN

Bab terakhir akan memuat kesimpulan isi dari keseluruhan uraian dari bab-bab sebelumnya dan saran-saran dari hasil yang diperoleh yang diharapkan dapat bermanfaat dalam pengembangan selanjutnya.

BAB II
TINJAUAN PUSTAKA

Bab ini akan menjelaskan mengenai tinjauan pustaka yang digunakan dalam penelitian. Tinjauan pustaka digunakan dalam menjelaskan aplikasi permainan "NoKidnap", selain itu tinjauan pustaka juga digunakan dalam memuat istilah-istilah penting yang terlibat dalam penelitian, seperti *Artificial Intelligence*, *Non-Playable* & *Playable Character*, Algoritma A*, Modul NavMesh Agent, Permainan Edukatif, Pembacaan Perilaku Musuh (*Enemy Behaviour*), *Animation Rigging*, dan *Animation Trigger*.

2.1 Aplikasi Permainan "NoKidnap"

NoKidnap adalah aplikasi permainan edukatif berbasis mobile pertama yang dikembangkan untuk menyimulasikan penculikan terhadap anak. Aplikasi ini juga bertujuan untuk mengedukasi tentang bagaimana sikap dan perilaku anak menyikapi pra-indikasi penculikan. *Game* ini dikembangkan dengan menggunakan teknologi *game engine* Unity, dimana aplikasi *game* yang dimainkan pengguna akan ditangkap pada layar gawai anda, sehingga pengguna dapat dengan mudah mengakses konten edukasi terkait pencegahan penculikan anak dengan visual yang menarik. Visual yang menarik akan memudahkan anak dalam memahami peristiwa yang dapat terjadi di lingkungan sekitarnya sehingga anak dapat mengambil keputusan dan melakukan tindakan pencegahan jika terjadi pra-indikasi penculikan.

2.1.1 Fitur Aplikasi

Aplikasi permainan "NoKidnap" sendiri memiliki beberapa ragam fitur yang mungkin ditemui dalam aplikasi, fitur-fitur ini bertujuan untuk mempermudah alur kerja permainan, sekaligus menyediakan pengalaman bermain yang lebih menarik. Adapun beberapa fitur yang mungkin didapat antara lain, sebagai berikut.

- Panduan orang tua, fitur ini dirancang khusus untuk orang tua dalam mengarahkan anak untuk bermain aplikasi.
- Catatan, fitur catatan berisikan ulasan dari keputusan yang diambil oleh pemain selain itu setiap misi ataupun cerita yang didapatkan akan dimuat

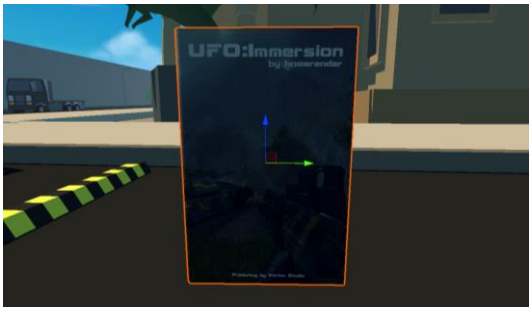
- di sini.
- *Dynamic Music*, fitur ini berfungsi untuk menyesuaikan musik yang didengar dengan keadaan yang sedang dialami pemain dalam permainan.
- *Artificial Intelligence*, fitur *artificial intelligence* atau kecerdasan buatan membekali karakter penculik kemampuan dalam meningkatkan kesulitan permainan.
- *Dialogue System*, fitur ini memungkinkan pemain dalam berinteraksi dengan karakter lain yang ada dalam permainan.
- *Path Finding*, fitur ini berfungsi dalam menyediakan algoritma khusus dalam pencarian jalur terdekat menuju objek tertentu dalam *game*.

2.1.2 Alur Kerja Aplikasi

Di bagian awal aplikasi ketika dijalankan terdapat *Scene* Menu yang berfungsi menampilkan halaman utama sebelum pemain memulai permainan. *Scene* ini juga menyediakan beberapa fitur utama untuk membantu pemain sebelum memulai permainan, seperti panduan orang tua, dan pengaturan suara. Selanjutnya, pada bagian halaman menu pemain dapat menekan tombol bermain pada bagian tengah halaman, sehingga memunculkan halaman level pada *scene* berikutnya. Pada *scene* level, pemain dapat memilih level permainan yang dapat bekerja menggunakan respons sentuhan pada layar *smartphone,* selanjutnya pemain dapat memainkan permainan seperti panduan yang telah diberikan pada bagian awal.

2.1.3 Objektif Permainan

Setelah memilih level, pemain disuguhkan dengan pengalaman cerita permainan. Setelah potongan adegan (*cutscene*) cerita ditampilkan pemain akan langsung memasuki lingkungan dunia permainan. Objektif dari permainan ini adalah mengumpulkan *points* sebanyak mungkin. Tampilan *points* ini dapat dilihat pada bagian tengah atas layar. Adapun *ponts* dapat diakumulasi berdasarkan jumlah buku yang berhasil diambil, serta objektif misi dalam cerita yang berhasil diselesaikan.



Gambar 2. 1 Buku pada Games "NoKidnap"

Gambar 2. 2 Points dan Waktu pada Games "NoKidnap"

Gambar 2. 3 Objektif Misi pada Games "NoKidnap"

2.1.4 Cerita Permainan

Pada awal permaian, pemain akan disuguhkan dengan karakter Leafy, diceritakan pada saat sepulang sekolah, Leafy diberikan beberapa misi yang menuntunnya untuk kembali ke rumah dengan aman. Misi awal permainan memandu Leafy untuk segera berbicara kepada guru dan menyelesaikan rintangan yang diberikan. Pemain dapat menghindari penculik, akan tetapi respons di awal permainan menunjukkan bahwa pemain (dalam hal ini anak-anak), tidak mengerti apakah seseorang tersebut penculik atau bukan, sehingga membutuhkan pengalaman bermain yang serius dalam menyelesaikan tantangan menggunakan fitur yang sudah dibenamkan, seperti pengalaman waktu bermain, catatan, level, dan koleksi buku yang telah dikumpulkan.

2.2 *Artificial Intelligence* (AI)

Artificial Intelligence (AI) berkaitan dengan membuat komputer mampu melakukan tugas-tugas berpikir yang dapat dilakukan oleh manusia dan hewan. Saat ini, kita sudah dapat memprogram komputer untuk memiliki kemampuan di atas manusia dalam menyelesaikan berbagai masalah, seperti perhitungan matematika, pengurutan, pencarian, dan sebagainya. Bahkan, kita bisa membuat komputer bermain beberapa permainan papan lebih baik daripada manusia (seperti Lichess, misalnya). Banyak dari masalah-masalah ini awalnya dianggap sebagai masalah kecerdasan buatan (AI), tetapi seiring dengan pemecahan yang semakin komprehensif, mereka keluar dari domain para pengembang AI.

Namun, ada banyak hal yang tidak bisa dilakukan oleh komputer yang kita anggap sepele, seperti mengenali wajah-wajah yang akrab, berbicara dalam bahasa kita sendiri, memutuskan apa yang harus dilakukan selanjutnya, dan bersifat kreatif. Ini adalah domain AI: mencoba memahami jenis algoritma apa yang diperlukan untuk menampilkan properti-properti ini.

Dalam ruang lingkup akademis, beberapa peneliti AI didorong oleh filosofi memahami sifat pemikiran dan kecerdasan serta membangun perangkat lunak untuk memodelkan bagaimana cara berpikir mungkin bekerja. Beberapa peneliti juga didorong oleh psikologi dalam memahami mekanika otak manusia dan proses mental, serta didorong oleh rekayasa membangun algoritma untuk melakukan tugas-tugas yang mirip dengan manusia. Distingsi tiga tingkat ini berada di pusat AI akademis, dan pandangan pikiran yang berbeda bertanggung jawab atas sub-bidang yang berbeda dari subjek tersebut.

Sebagai pengembang *game*, umumnya mereka tertarik pada sisi rekayasa, yakni membangun algoritma yang membuat karakter *game* tampak manusia atau mirip hewan. Pengembang selalu mengambil inspirasi dari penelitian akademis, di mana penelitian tersebut membantu mereka menyelesaikan pekerjaan mereka. Mungkin bermanfaat untuk memberikan gambaran singkat tentang pekerjaan AI yang dilakukan di akademis untuk mendapatkan pemahaman tentang apa yang ada dalam subjek tersebut dan apa yang mungkin layak untuk dicontoh. Kita tidak memiliki ruang (minat dan kesabaran) untuk memberikan pandangan lengkap tentang AI (*Artificial Intelligence*) akademis, tetapi akan membantu melihat jenis teknik apa yang akhirnya masuk ke dalam permainan.

2.3 *Non-Playable & Playable Character*

Non-Playable Characters (NPCs) dan *Playable Characters* (PCs) adalah dua elemen utama dalam dunia permainan video yang memainkan peran kunci dalam memberikan pengalaman bermain kepada pemain. NPCs adalah karakter yang tidak dikendalikan langsung oleh pemain, tetapi oleh program perangkat lunak atau kecerdasan buatan. Mereka dapat bertindak sebagai teman, musuh, atau karakter pendukung dalam cerita permainan. NPCs sering kali dirancang untuk memberikan kehidupan pada lingkungan permainan, memberikan tantangan, atau menyediakan informasi yang diperlukan. Di sisi lain, PCs adalah karakter yang sepenuhnya dikendalikan oleh pemain. Pemain dapat mengendalikan tindakan, keputusan, dan perjalanan karakter ini dalam permainan. Karakter ini sering menjadi perpanjangan dari pemain dalam menciptakan naratif permainan dan menjelajahi dunia virtual. Dalam banyak permainan, interaksi antara NPCs dan PCs menciptakan dinamika yang kompleks, memberikan pemain berbagai tantangan dan pengalaman yang mendalam. Oleh karena itu, pemahaman yang baik tentang peran dan interaksi antara NPCs dan PCs adalah kunci dalam perancangan permainan yang menarik dan memuaskan.

Gambar 2. 4 Contoh Karakter NPC Guru dalam Games "NoKidnap"

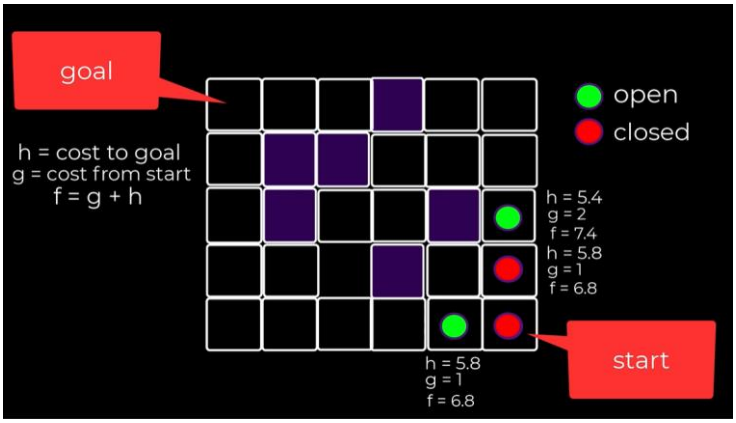
Gambar 2. 5 Contoh Karakter PC Leafy dalam Games "NoKidnap"

2.4 Algoritma A*

Penerapan algoritma A* menggunakan modul *NavMesh Agent* pada aplikasi permainan edukatif "No Kidnap" bertujuan untuk membaca perilaku musuh (*enemy behaviour*) dengan lebih efektif. Algoritma A* menjadi landasan penting dalam penelitian ini. Dalam artikel (Hart et al., 1968), diperkenalkan dasar-dasar heuristik dalam penentuan jalur *minimum cost*. Konsep ini menjadi dasar teoritis dalam pengembangan algoritma A*.

Menurut Russell et al. (2020), dalam bukunya yang berjudul "Artificial Intelligence: A Modern Approach", algoritma A* memberikan pemahaman yang komprehensif tentang algoritma pencarian. Wasserstein et al. (2005) juga menjelaskan mengenai teori dan praktik algoritma A*, termasuk penerapan heuristik dan strategi pencarian yang memberikan estimasi biaya atau jarak dalam mencari jalur optimal.

Selain itu, artikel Weber et al. (2010) mengenai aplikasi algoritma A* dalam *game* AI memberikan contoh konkret tentang penggunaan algoritma A* dalam navigasi karakter dalam permainan. Implementasi algoritma A* dalam navigasi karakter merupakan aspek penting dalam memahami perilaku musuh dalam *game* edukatif "No Kidnap". Artikel tersebut akan memberikan contoh nyata tentang penggunaan algoritma A* dalam konteks *game development*.



*Gambar 2. 6 Visualisasi Algoritma A**

(Sumber: learn.unity.com)

2.5 Modul NavMesh Agent

Unity Technologies (2021) menyediakan dokumentasi resmi tentang penggunaan modul NavMesh Agent dalam Unity. Manual ini memberikan panduan rinci tentang pengaturan dan penggunaan NavMesh Agent untuk navigasi karakter dalam permainan. Dengan mengacu pada dokumentasi ini, penerapan algoritma A* menggunakan NavMesh Agent dapat dilakukan dengan lebih baik dan efisien.

Selanjutnya menurut Julien (2014), dalam buku "Unity AI Programming Essentials", pengembangan kecerdasan buatan (AI) memberikan pemahaman yang komprehensif. Julien (2014) juga membahas berbagai konsep dan teknik AI yang relevan, termasuk algoritma pencarian dan algoritma A*. Prinsip-prinsip dasar dari algoritma A* dengan penggunaan modul NavMesh Agent dalam permainan edukatif "No Kidnap" dapat digabungkan untuk menghasilkan pengalam bermain yang lebih dinamis.

Secara umum ada empat prinsip-prinsip dasar algoritma A*. Pertama, representasi Grafik, algoritma A* beroperasi pada representasi grafik dari lingkungan atau ruang pencarian. Grafik ini terdiri dari simpul (*node*) yang mewakili lokasi dalam ruang dan tepi (*edge*) yang menghubungkan simpul-simpul tersebut. Misalnya, dalam konteks permainan, simpul dapat mewakili posisi karakter atau objek dalam lingkungan, sementara tepi dapat mewakili koneksi atau pergerakan antara simpul-simpul tersebut. Prinsip yang kedua adalah biaya dan heuristik. Setiap simpul dalam grafik memiliki biaya atau nilai yang menunjukkan seberapa mahal atau sulit untuk mencapai simpul tersebut. Algoritma A* menggunakan fungsi heuristik untuk memberikan estimasi biaya yang tersisa dari simpul saat ini ke tujuan akhir. Fungsi heuristik ini harus adil

(tidak *overestimate* atau *underestimate*) untuk memastikan algoritma A* dapat mencapai solusi optimal. Prinsip yang ketiga adalah pencarian terarah. Algoritma A* menggunakan pendekatan pencarian terarah dengan mempertimbangkan biaya aktual untuk mencapai simpul saat ini dan perkiraan biaya yang tersisa (heuristik) untuk mencapai tujuan. Dengan memilih simpul dengan biaya total terkecil (g(x) + h(x)) sebagai simpul berikutnya yang akan dieksplorasi, algoritma A* mencoba untuk mengarahkan pencarian menuju jalur optimal dengan efisiensi yang tinggi. Tahapan terakhir adalah penelusuran dan penyimpanan. Algoritma A* melibatkan penelusuran melalui simpul-simpul grafik dalam rangka mencari jalur terbaik. Selama proses penelusuran, informasi seperti biaya aktual dan heuristik diperbarui dan disimpan bersama dengan setiap simpul yang dikunjungi. Informasi ini digunakan untuk menghitung biaya total dan memutuskan urutan penelusuran berikutnya.

Selain itu, menurut Matić et al. (2019), perbandingan antara kerangka pengembangan permainan Unity3D dan Unreal Engine tidak berbeda jauh dalam segi penerapan menggunakan modul NavMesh Agent. Hanya saja kedua pengembangan permainan tersebut memiliki kelebihan dan kekurangan masing-masing. Seperti teknik perawatan pada pengembangan Unreal yang lebih sulit dan terkesan terlalu otomatis.

Gambar 2. 7 Modul Navmesh Agent
sumber: youtube.com/@Brackeys

2.6 Permainan Edukatif

Permainan edukatif merupakan pendekatan yang menarik dalam mengembangkan keterampilan dan pengetahuan pada konteks pendidikan. Menurut Miller et al. (2010), dalam artikelnya mengenai *serious games* menunjukkan bahwa permainan serius dapat menjadi alat efektif untuk mempromosikan pemahaman dan toleransi terhadap budaya yang berbeda. Dengan merujuk pada penelitian ini, permainan edukatif dapat

dirancang untuk memperkaya pengalaman belajar dengan memasukkan elemen yang berfokus pada kesadaran budaya.

Menurut Boyle et al. (2011) permainan edukatif yang dirancang dengan baik dapat meningkatkan motivasi belajar dan pencapaian akademik. Dengan melihat desain dan evaluasi permainan, perancang (*developer*) permainan edukatif dapat memperoleh pemahaman tentang faktor-faktor desain yang penting dan melibatkan evaluasi yang tepat dalam pengembangan permainan edukatif yang efektif.

Selain itu, menurut Prensky (2001) dalam artikelnya yang membahas tentang digital *game-based learning* (pembelajaran berbasis permainan digital). Menurutnya, permainan digital memiliki keunggulan sebagai alat pembelajaran yang menarik dan interaktif. Penggunaan permainan edukatif dengan pendekatan ini memungkinkan siswa untuk belajar melalui pengalaman praktis dan berpartisipasi aktif.

(A) **(B)** **(C)**

Gambar 2. 8 Contoh Aplikasi Permainan Edukatif (A) ABC Alphabet Tracing & Phonics (B) Brain Challenge (C) Puzzle Huruf

2.7 Pembacaan Perilaku Musuh *(Enemy Behaviour)*

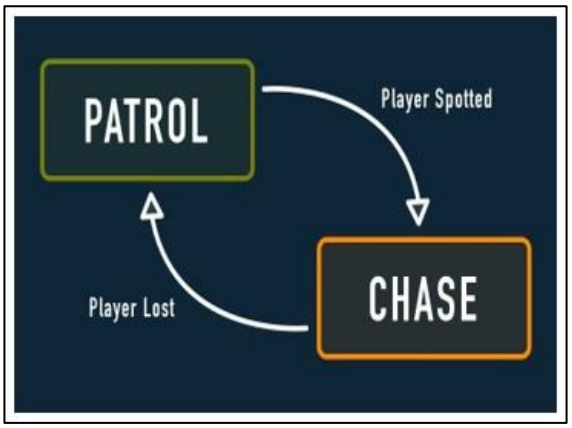
Pembacaan perilaku musuh (*enemy behaviour*) merupakan komponen kunci dalam pengembangan permainan yang menantang dan menarik. Menurut Buckland (2005), dalam bukunya yang berjudul "Programming Game AI by Example" membahas berbagai konsep dan teknik dalam pemrograman kecerdasan buatan (AI) dalam permainan. Contoh praktis dan studi kasus yang relevan dalam pengembangan perilaku musuh yang realistis dan menantang adalah menggunakan bantuan kecerdasan buatan. Dengan menggunakan bantuan kecerdasan buatan, maka efektivitas program yang dihasilkan juga akan berjalan signifikan.

Pengembangan kecerdasan buatan (AI) dalam permainan merupakan komponen kunci yang memberikan pengalaman bermain yang menarik dan menantang bagi para

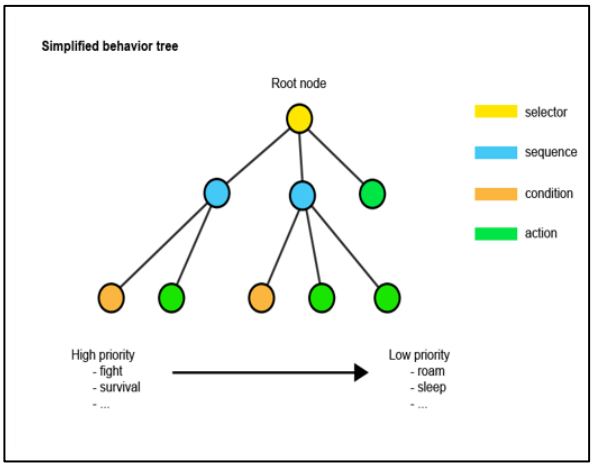
pemain. Seperti yang dikatakan oleh Millington (2009) dalam buku mereka yang berjudul "Artificial Intelligence for Games", AI dalam permainan memungkinkan karakter NPC (*Non-Playable Character*) untuk menampilkan perilaku yang adaptif, realistis, dan cerdas. Menurut penulis, kecerdasan buatan dalam permainan melibatkan penggunaan teknik dan algoritma untuk memberikan kemampuan kepada NPC (*Non-Playable Character*) untuk mengambil keputusan secara otomatis dan berinteraksi dengan lingkungan virtual yang kompleks.

```
public class SwitchExample : MonoBehaviour
{
    public State currentState = State.Idle;

    void Update()
    {
        switch (currentState)
        {
            case State.Idle:
                Debug.Log("Waiting...");
                break;

            case State.Attack:
                Debug.Log("Attacking!");
                break;

            case State.Retreat:
                Debug.Log("Run Away!");
                break;
        }
    }
}

[System.Serializable]
public enum State { Idle, Attack, Retreat}
```

(A)



(B)



(C)

Gambar 2. 9 Beberapa macam konsep Enemy AI Behaviour Control pada Aplikasi Permainan (Games) (A) Switch Statements (B) State Machine (C) Behaviour Tree

2.8 *Animation Rigging*

Animation rigging merupapakan teknik yang digunakan dalam pengembangan *game* dan ani masi untuk mengontrol dan memanipulasi animasi karakter secara dinamis. Dengan menggunakan *animation rigging*, animator dapat membuat karakter lebih hidup dan responsif terhadap interaksi pemain atau lingkungan sekitarnya. Landasan teori *animation rigging* mencakup beberapa konsep utama, seperti:

2.8.1 *Rigging*: *Rigging* adalah proses pengaturan struktur tulang dan kontrol yang digunakan untuk menggerakkan model karakter. Dalam konteks Animation Rigging, rigging melibatkan penambahan kontrol tambahan atau pemrograman khusus untuk meningkatkan fleksibilitas animasi.

2.8.2 *Control Rig*: *Control Rig* adalah set kontrol yang digunakan untuk mengatur karakter dalam animasi. Kontrol ini dapat berupa bone, ikon, atau objek lain yang dapat dipindahkan animator untuk mengubah pose atau ekspresi karakter.

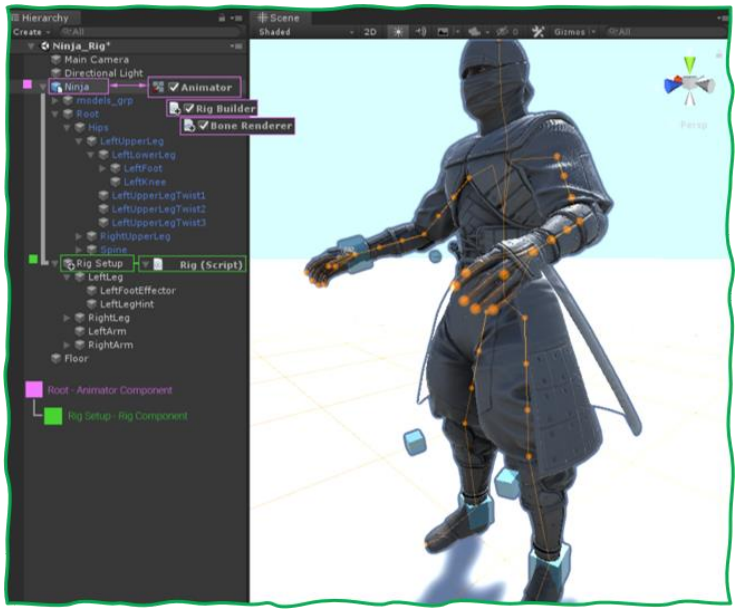
2.8.3 IK (*Inverse Kinematics*): IK adalah teknik yang digunakan untuk membuat gerakan realistis pada karakter dengan mengontrol posisi target, sementara sistem secara otomatis mengatur gerakan sendi dan tulang yang sesuai.

2.8.4 FK (*Forward Kinematics*): FK adalah teknik animasi yang mengendalikan posisi dan rotasi setiap tulang secara langsung, mulai dari

tulang teratas hingga tulang terbawah dalam hierarki tulang.

2.8.5 *Blend Trees*: *Blend Trees* adalah alat yang digunakan untuk menggabungkan beberapa animasi menjadi satu, sehingga karakter dapat beralih mulus antara pose atau gerakan yang berbeda tergantung pada kondisi tertentu, seperti kecepatan karakter atau input pengguna.

2.8.6 *Constraints*: *Constraints* adalah aturan atau pembatasan yang diterapkan pada animasi karakter untuk membatasi gerakan tertentu atau memastikan bahwa karakter bergerak sesuai dengan kebutuhan desainer.

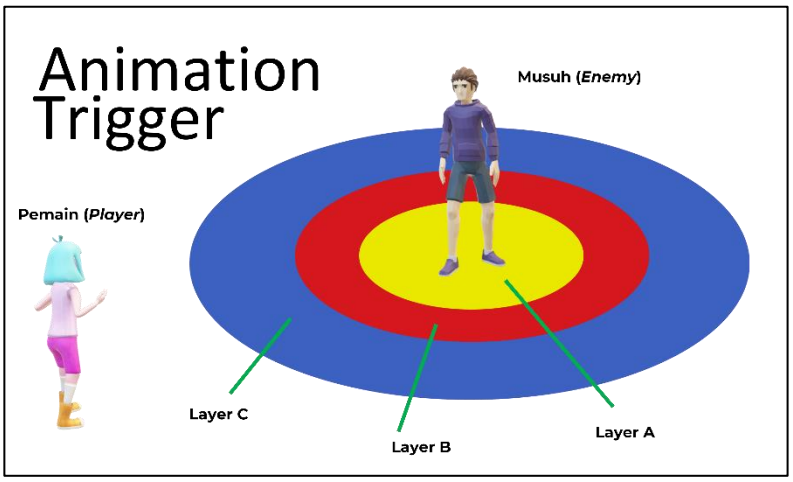


Gambar 2. 10 Animation Rigging

2.9 *Animation Trigger*

Animation trigger sebuah tanda yang akan direspon oleh *game object* (objek permainan) yakni NPCs (*Non-Playable Character*) dan PCs (*Playable Character*. *Animation trigger* merupakan aspek kritis dalam pengembangan permainan video yang memberikan kehidupan pada karakter dan lingkungan permainan. Sistem ini berfungsi sebagai mekanisme yang mengatur kapan dan bagaimana suatu animasi tertentu dipicu dalam respons terhadap peristiwa atau tindakan tertentu dalam permainan. Pada dasarnya, *animation trigger* memungkinkan pengembang permainan untuk mengontrol alur cerita dan interaksi karakter dengan lingkungan sekitarnya melalui animasi yang sesuai. Dalam konteks permainan edukatif, sistem ini dapat digunakan untuk menyampaikan informasi atau mengajarkan konsep-konsep tertentu

kepada pemain melalui animasi yang diaktifkan oleh perilaku tertentu. Pemahaman mendalam tentang cara merancang dan mengimplementasikan *animation trigger* dapat memberikan kontribusi signifikan terhadap kualitas dan efektivitas pengalaman bermain dalam aplikasi permainan, khususnya dalam konteks pencegahan penculikan anak yang menjadi fokus penelitian ini. Oleh karena itu, eksplorasi dan analisis literatur mengenai pengembangan serta penerapan sistem ini dalam konteks permainan edukatif menjadi penting untuk memahami landasan teoritis dan praktis yang dapat membimbing penelitian lebih lanjut.

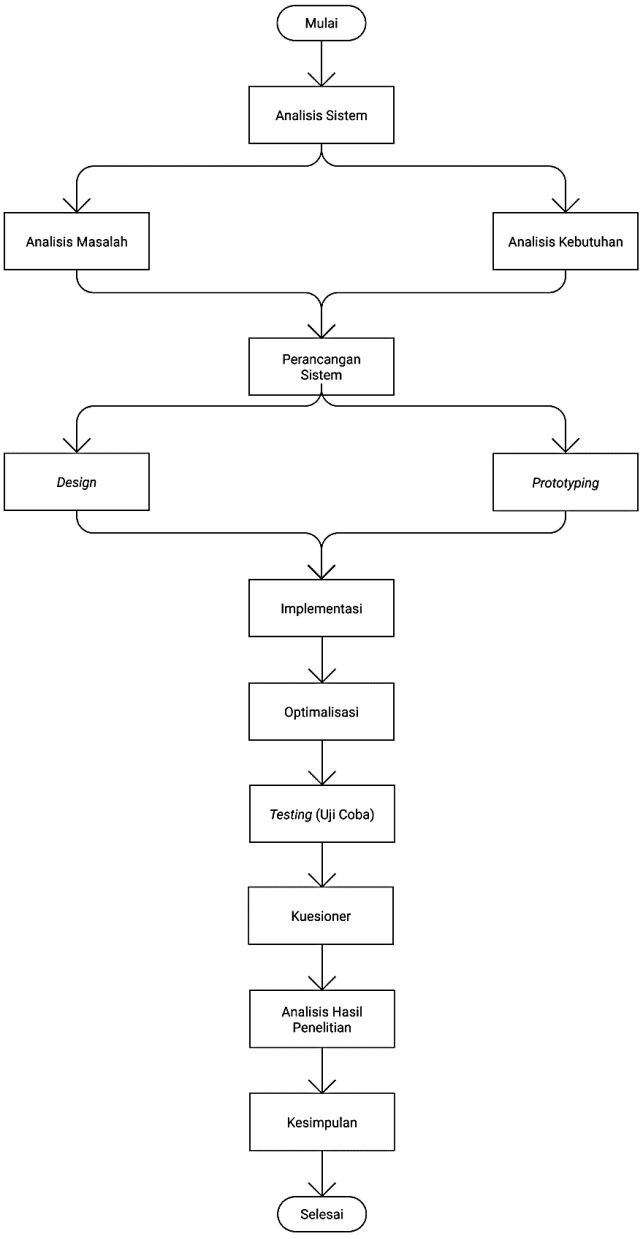


Gambar 2. 11 Contoh Animation Triger pada Aplikasi Permainan NoKidnap

BAB III
ANALISIS DAN PERANCANGAN

Bab ini akan memberikan serangkaian analisis berupa metodologi dan alur perancangan sistem *enemy behvaiour* pada aplikasi permainan "NoKidnap".

3.1 Metodologi



Gambar 3. 1 Alur Metodologi Penelitian

Adapun penelitian ini dikembangan menggunakan metode *research experimental* berdasarkan metode pengembangan aplikasi permainan *Game Development Life Cycle* (GDLC). Metodologi ini memungkinkan *research* dapat dijalankan secara organik dengan melewati tahapan berupa *design* & *prototyping*, implementasi, optimalisasi, serta melewati tahap pengujian (*testing).*

3.2 Analisis Sistem

Dalam membangun sebuah sistem, diperlukan suatu tahapan untuk membantu dalam proses perancangan sebuah sistem. Analisis sistem adalah proses yang dilakukan untuk memahami, mengidentifikasi, dan merencanakan perubahan atau perbaikan dalam suatu sistem. Analisis sistem bertujuan untuk mempermudah tahapan proses implementasi sistem, serta memberikan pemahaman dan penentuan atas apa yang dibutuhkan oleh sistem dan yang harus dikerjakan oleh sistem. Tahapan analisis sistem sangat penting, karena kesalahan dalam tahapan ini akan menyebabkan kesalahan pada tahapan yang berikutnya. Tahapan ini melibatkan pemahaman mendalam tentang bagaimana suatu sistem bekerja dan bagaimana komponen-komponennya saling berinteraksi. Bagian-bagian dari tahap ini mencakup 2 bagian yaitu: Analisis Masalah dan Analisis Kebutuhan.. Analisis masalah merupakan tahapan yang memberikan pemahaman bagaimana mengidentifikasi masalah dan mempelajari penyebab dari masalah yang akan diselesaikan dengan menggunakan sebuah sistem. Analisis kebutuhan merupakan tahapan yang memberikan penjelasan mengenai fungsi-fungsi sistem yang ditawarkan dan apa yang dikerjakan oleh sistem.

3.2.1 Analisis Masalah

Dalam pengembangan aplikasi permainan (*games*), dibutuhkan suatu implementasi sistem AI pada karakter musuh (*enemy*) dalam game. Hal ini dikarenakan bnhwa karakter musuh menjadi salah satu aspek penting yang tidak pernah terlewat. Namun, seringkali, karakter musuh (*enemy*) yang dihadirkan dalam aplikasi permainan (*game*) cenderung kurang responsif, terlalu kaku dan terstruktur, dan tidak mampu menyajikan pengalaman bermain (*user experience*) yang sesuai dengan logika pemain. Sehingga, peneliti merasa masih terdapat sejumlah masalah utama yang perlu diidentifikasi dan dipecahkan untuk meningkatkan pengalaman bermain (*user experience*) tersebut. Analisis masalah mendalam menjadi tahap kunci dalam

memahami tantangan-tantangan yang dihadapi dalam implementasi sistem AI pada karakter musuh (*enemy*). Adapun beberapa masalah yang ditemui oleh peneliti antara lain sebagai berikut.

3.2.1.1 Kecacatan Logika Penculik (*Enemy Behaviour*)

Dalam mensimulasikan semua aksi atau tindak nyata sebuah *enemy* terkadang membutuhkan respons yang lebih spesifik, seperti menentukan ke arah lawan, kapan penculik harus berlari mengejar, memberi salam, atau beragam interaksi yang unik sesuai dengan logika simulasi yang baik. Dalam games "NoKidnap" sudah terdapat indikasi penculikan, akan tetapi dalam proses penentuan logika simulasi masih terdapat ketidak sinkronan antara waktu antar setiap state animasi berlangsung.

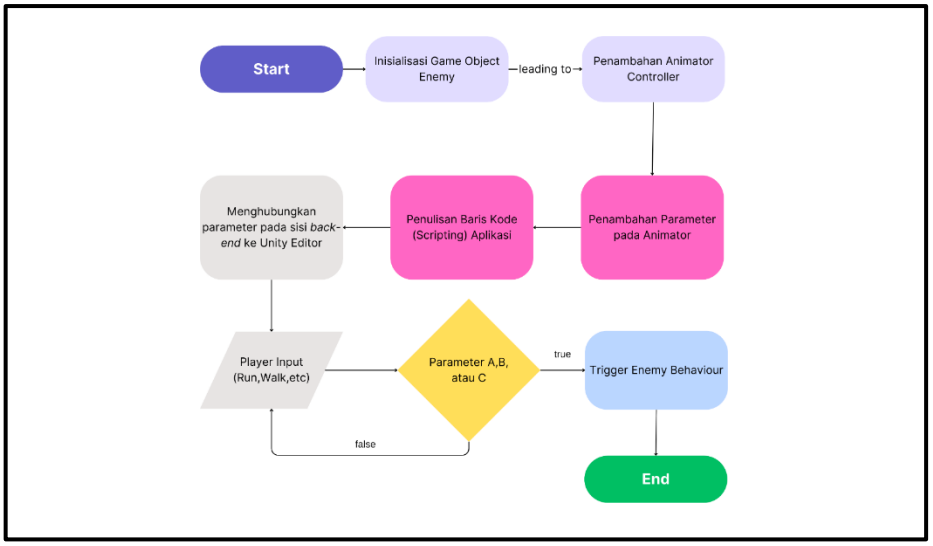
3.2.1.2 Kurangnya Efisiensi dalam Penampilan Animasi

Setiap animasi daripada sebuah game tentunya memiliki waktu efisiensi tersendiri dalam menuliskan baris perintah terhadap serangkaian informasi yang terdapat dalam game. Dalam menyajikan sebuah animasi, beragam informasi dibutuhkan keterkaitan yang mudah dalam penulisannya. Sehingga, pihak pengembang atau *developer* tidak merasa kesulitan saat menentukan struktur pengembangan animasi secara bertahap.

3.2.2 Analisis Kebutuhan

Analisis kebutuhan terbagi atas dua bagian, yaitu kebutuhan fungsional dan kebutuhan non-fungsional sistem. Pada tahap analisis ini dibutuhkan dalam mendukung kinerja sistem, apakah sistem dapat dibuat sesuai kebutuhan atau belum sesuai, karena kebutuhan sistem akan mendukung tercapainya tujuan. Dalam melaksanakan analisis kebutuhan fungsional diperlukan sebuah *flowchart*, sementara proses *design thinking* akan menjadi tahapan analisis kebutuhan non-fungsional. Artinya, setiap tahapan dalam pengembangan akan mencapai tahap minimum yang dibutuhkan oleh pengguna (*user*).

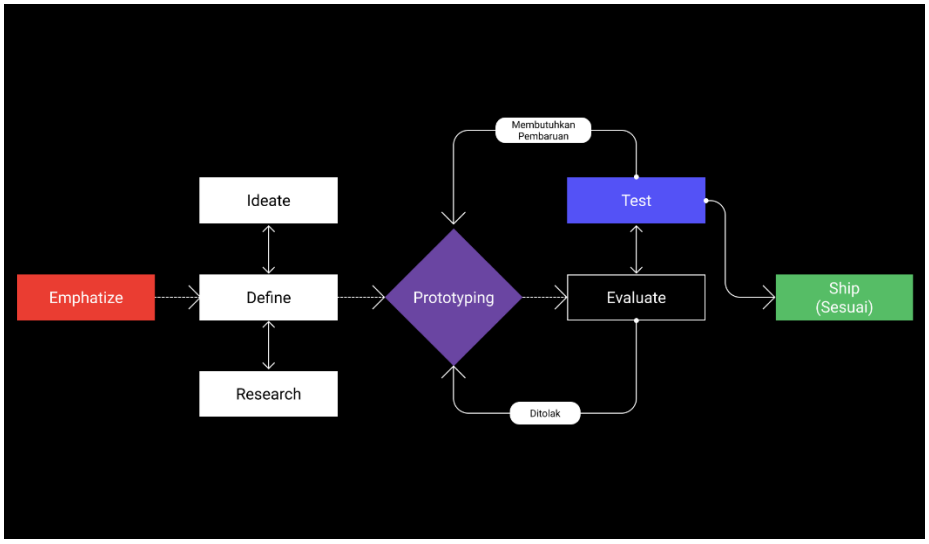
3.2.2.1 Analisis Fungsional



Gambar 3. 2 Analisis Kebutuhan Fungsional Menggunakan Flowchart

Analisis fungsional bertujuan dalam memberikan gambaran ringkas terkait langkah minimum yang harus ditempuh berdasarkan alur pengembangan awal aplikasi. Tahapan ini termasuk ke dalam kebutuhan dalam mengembangkan sebuah aplikasi berupa prototipe aplikasi.

3.2.2.2 Analisis Non-Fungsional



Gambar 3. 3 Analisis Kebutuhan Non-Fungsional Menggunakan Design Thinking Diagram

Analisis non-fungsional bertujuan dalam memberikan desain petunjuk dari mulai awal riset dimulai hingga nilai fungsional aplikasi tercapai yakni berupa prototipe. Setelah prototipe dihasilkan, maka prototipe tersebut aplikasi akan

melewati tahap uji evaluasi dan uji coba hingga menghasilkan produk yang sesuai dengan kebutuhan.

3.3 Perancangan Sistem

Pada penelitian ini digunakan GDLC (*Game Development Lifecycle*) dalam menentukan alur pengembangan sebuah *game* beserta arsitekturnya. Pemodelan kebutuhan sistem memastikan setiap komponen sesuai kebutuhan.

3.3.1 Perancangan Arsitektur Aplikasi

3.3.1.1 Konsep Aplikasi

Aplikasi Permainan "NoKidnap" merupakan sebuah *video games* berbasis mobile Android yang bertujuan dalam mensimulasikan panduan dalam mencegah tindakan penculikan terhadap anak-anak. *Game* ini memiliki tema pendidikan yang dikemas dalam tampilan interaksi yang menarik sehingga dapat menyesuaikan dalam pengalaman visual pengguna (*user*) khususnya anak-anak usia 7-11 tahun. *Game* ini menampikan perspektif pemain orang ketiga, sehingga memungkinkan anak-anak dalam mensimulasikan pengalaman bermaain yang lebih nyata. Dalam menyelesaikan permainan, pemain akan diarahkan untuk melakukan perintah terhadap Leafy, yakni karakter utama dalam game, sehingga dapat menyelesaikan panduan dan menjawab semua pertanyaan dalam buku, dimana buku tersebut berfungsi untuk memberikan pemahaman baru bagi pemain, sehingga dapat mengerti akan bahaya penculikan anak, serta dapat ditemukan secara acak pada lingkungan bermain.

3.3.1.2 Mekanik Aplikasi Permainan

Mekanik utama yang diterapkan dalam *game* ini adalah berfokus pada kontrol pemain dalam mengumpulkan item dan melakukan interaksi terhadap NPC (*Non-Playable Character*). Nantinya NPC dapat berupa musuh (penculik), orang tua, atapun karakter umum lainnya. Pada awal permainan pemain akan disuguhkan dengan pengalaman kontrol seperti lari, lompat, dan lihat sekitar, sehingga pemain dapat melakukan interaksi terhadap lingkungan sekitar. Selain daripada itu, pemain juga dapat melakukan interaksi, seperti mengumpulkan item dan juga berinteraksi untuk menjawab beberapa pertanyaan terhadap NPC. *Game* ini juga memiliki *Resource Management System* berupa *Healthbar*, *Point*, dan *Energy*, yang dapat berupa sesuai dengan beberapa aksi yang dilakukan dalam

bermain. Adapun cara kerja mekanik terhadap sistem ini, antara lain:

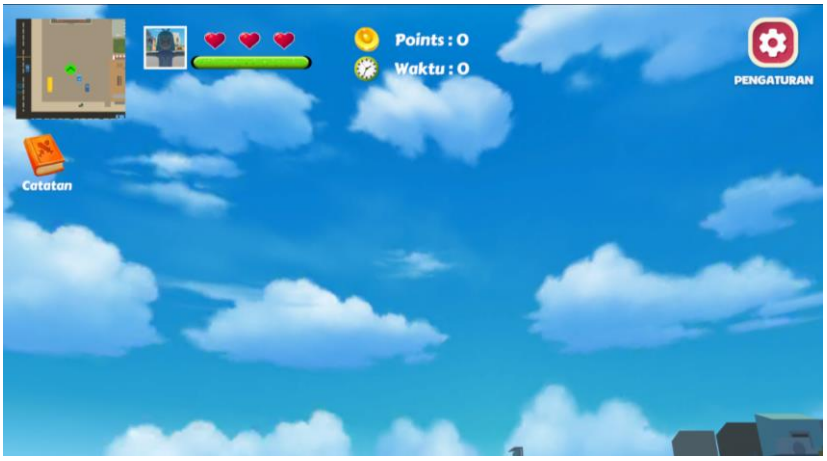
- *Look around* merupakan sebuah perintah yang dapat diberikan pemain dalam melihat keaadan lingkungan sekitar. Cara kerja mekanik ini adalah dengan menekan tombol *Look around* pada tampilan antarmuka pemain (UI), kemudian mengarahkan tombol ke arah yang dinginkan. Cara kerja look around ini adalah seperti indera pelihat di kehidupan nyata.
- *Talk* merupakan sebuah perintah yang dapat diberikan pemain dalam melakukan interaksi kepada pemain, seperti guru, orang tua, penculik, dan karakter umum lainnya.
- *Energy System* merupakan sumber daya yang diperlukan dalam melakukan kondisi tertentu. Dalam *games* "NoKidnap" terdaapat beberapa perintah yang mengharuskan pemain membuthkan energi, seperti berlari, melompat, dan sebagainya. Berbagai kondisi tersebut memungkinkan pemain merasakan kondisi tambahan sebagai batasan khusus dalam meningkatkan pengalaman bermain. Dalam konteks ini, kehabisan energi akan berdampak pada kondisi tambahan yakni "kelelahan", sehingga pemain tidak dapat melakukan input lari atau melompat selama rentang waktu tertentu.
- *Point Items* merupakan sumber daya permainan yang menampilkan tingkat keberhasilan pemain dalam menyelesaikan tugas yang diberikan dalam bentuk angka. Angka tersebut akan terus bertambah seiring dengan kemampuan user (pengguna) dalam menyelesaikan tugas yang diberikan.
- *Healthpoint System* merupakan sumber daya permainan yang menunjukkan keaadaan hidup pemain (nyawa) dalam *game,* sehingga *player* masih dapat bermain pada kesempatan yang sama. Kesempatan tersebut berfungsi sebagai sumber daya (*resource*) dalam mengelola proses pengambilan keputusan dalam *game,* seperti bagaimana mengurangi kemungkinan terkena serangan penculik, kapan mengambil risiko berdasarkan tingkat HP yang tersisa, dan berapa lama waktu yang dibutuhkan dalam mengumpulkan tugas yang diberikan.

3.3.1.3 Desain *User Interface* (UI)

Merancang tata letak antar muka pengguna (UI) merupakan langkah yang krusial karena tujuan utamanya adalah untuk menyampaikan informasi terkait

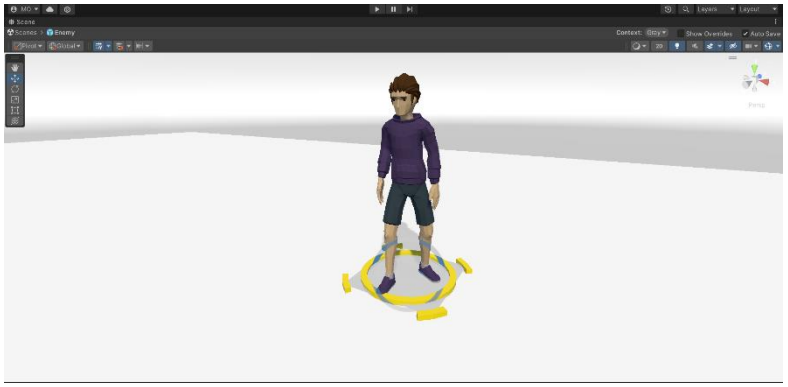
status avatar pemain dan musuh kepada pemain. Dalam penelitian ini, tata letak UI yang dirancang dibagi menjadi tiga bagian: UI Pemain, UI Musuh, dan UI Akhir Permainan.

- Player UI, *Player User Interface* (UI) adalah elemen-elemen UI yang berguna untuk menunjukkan informasi secara visual tentang *status player*. Perancangan *Player UI* yang akan dimplementasikan didasari pada mekanik-mekanik yang berhubungan dengan avatar *player* didalam *game*. Mekanik-mekanik yang memerlukan UI ini adalah *Health Point System*, *Stamina System*, dan Point *System*.

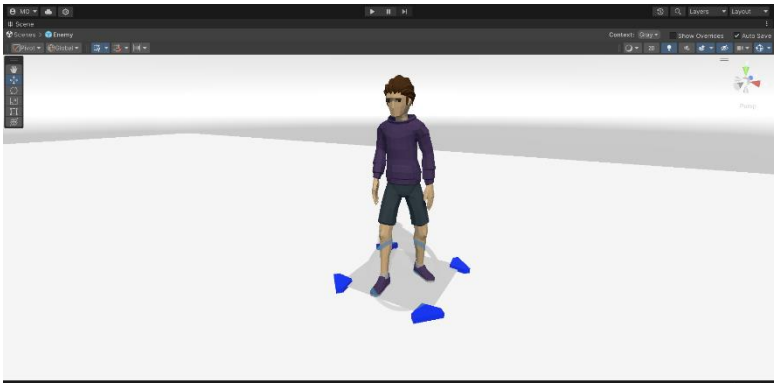


Gambar 3. 4 Player User Interface (UI)

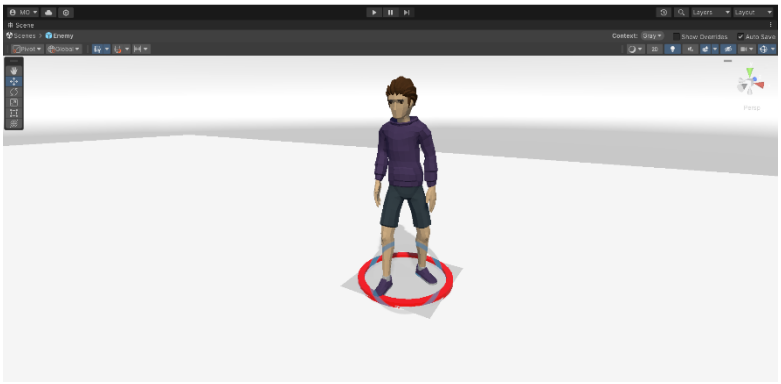
- Enemy UI, *Enemy User Interface* adalah elemen-elemen UI yang berguna untuk menunjukkan informasi secara visual tentang status *enemy*. Perancangan *enemy* UI yang dimplementasikan didasari pada status yang berhubungan dengan karakter enemy didalam game. status tersebut adalah: *Good Condition*, *Common Condition* dan *Bad Condition*. Setiap status pada *layer* nantinya akan direpresentasikan dalam perhitungan jarak pada radius musuh (penculik), yakni *Base Layer*, *Middle Layer*, dan *Outter Layer*.



Gambar 3. 5 Enemy UI Good Condition

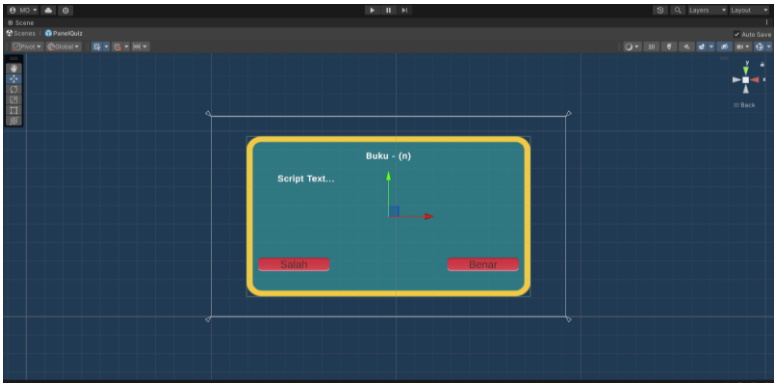


Gambar 3. 6 Enemy UI Common Condition



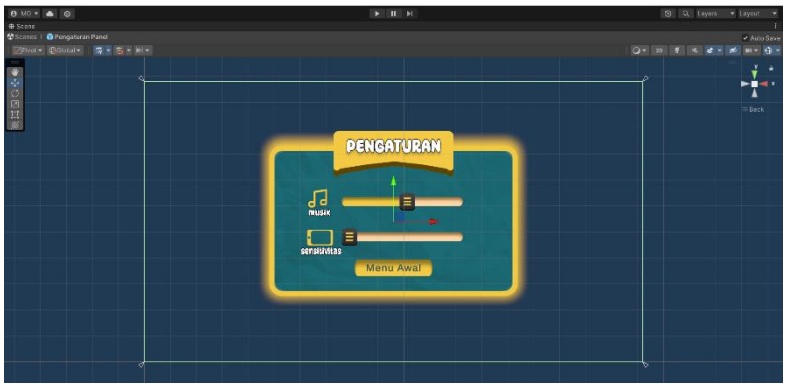
Gambar 3. 7 Enemy UI Bad Condition

- *Quiz Manager UI*, *Quiz Manager Interface* memungkinkan tampilan Kuis diaktifkan setiap kali *user* (pemain) berhasil mengumpulkan buku atau berinteraksi dengan NPC.



Gambar 3. 8 Quiz Panel User Interface (UI)

- *Sound & Settings UI, Sound and Settings User Interface* merupakan tampilan yang dapat diaktfikan apabila pemain menekan tombol pengaturan (*settings*) pada saat aplikasi sedang dimainkan. UI ini terletak pada ujung sebelah kanan atas.

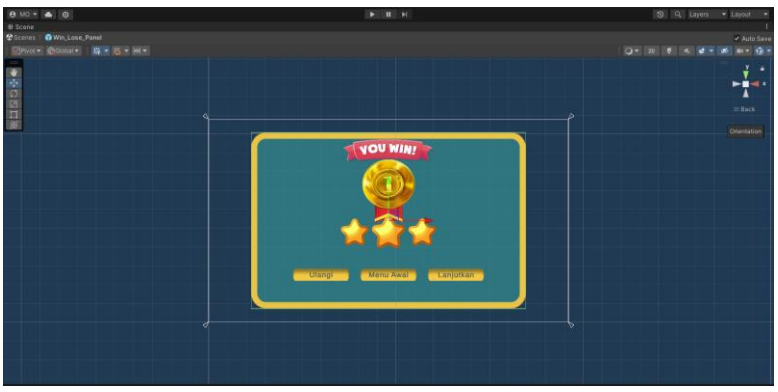


Gambar 3. 9 Sound & Settings User Interface (UI)

- *Game Menu UI, Game Menu User Interface* merupakan tampilan awal yang diaktifkan ketika pemain pertama sekali membuka aplikasi. Pada saat aplikasi dijalankan pemain akan disajikan dengan beberapa tampilan pra-bermain seperti tombol *play* untuk memulai, tombol panduan orangtua untuk melihat panduan bermain lengkap, tombol pengaturan untuk mengatur sensitivitas dan volume suara dan tombol *quit* untuk keluar dari aplikasi.

Gambar 3. 10 Game Menu User Interface (UI)

- *Game Over* UI adalah elemen-elemen UI yang berguna untuk memberitahu *player* mengenai situasi dimana *game* telah berakhir. Situasi ini terjadi jika *player Heatlhbar Point* pemain sudah habis (*Player* HP ≤ 0) atau *player* berhasil memenuhi tugas utama dari *game* ini, yaitu berhasil menuju level selanjutnya dan mengumpulkan buku. Pemain akan disajikan mengenai kondisi *Player* UI terakhir ketika permainan berakhir. Seperti kondisi *player badge* dan jumlah point yang berhasil didapatkan. Pada UI tersebut juga, pemain dapat menentukan beberapa perintah *input* (masukan) yang dapat dipilih setelahnya apabila informasiyang didapatkan sudah cukup, seperti tombol *restart*, menuju level selanjutnya, atau tombol *quit (keluar)*.



Gambar 3. 11 Game Over User Interface (UI)

3.3.1.4 Desain Audio

Adapun tema *game* ini adalah Kids & Fantasy, dan komposisi musik dan suara digunakan untuknya. Sebagian besar instrumen ceria

digunakan untuk memainkan musik ini, yang diiringi dengan lantunan musik instumental modern untuk menghasilkan pengalaman yang lebih menarik. Musik seperti ini digunakan untuk membuat dunia fantasi lebih menarik bagi anak-anak.

Dua jenis efek suara yang ada di *game* ini adalah *World Sounds* dan *Player Sounds*. Suara *World* adalah suara yang dihasilkan oleh pemain dari interaksi mereka dengan objek di dunia, seperti suara kendaraan yang berjalan di permukaan jalan. *Suara Player* adalah suara yang dihasilkan oleh pemain dari interaksi mereka dengan objek di dunia, dan digunakan untuk meningkatkan pengalaman pemain dengan *dunia game*. Suara NPC dan musik background mengisi suara ini.

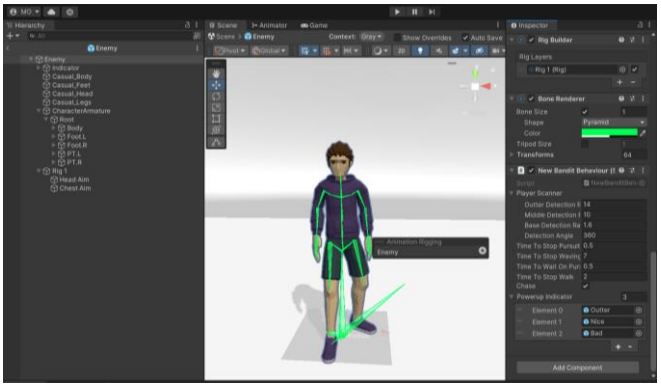
Semua musik dan efek suara yang digunakan dalam penelitian ini berasal dari Unity Assets Store dan Pixabay, yang keduanya dapat diakses secara gratis dan tanpa royalti.

3.3.1.5 Desain Animasi

Pada pengembangan *game* ini, animasi digunakan untuk meniru perilaku avatar pemain dan musuh. Ini dilakukan dengan tujuan memastikan bahwa *game* yang ditawarkan dapat berjalan dengan baik dan mengikuti logika pemain.

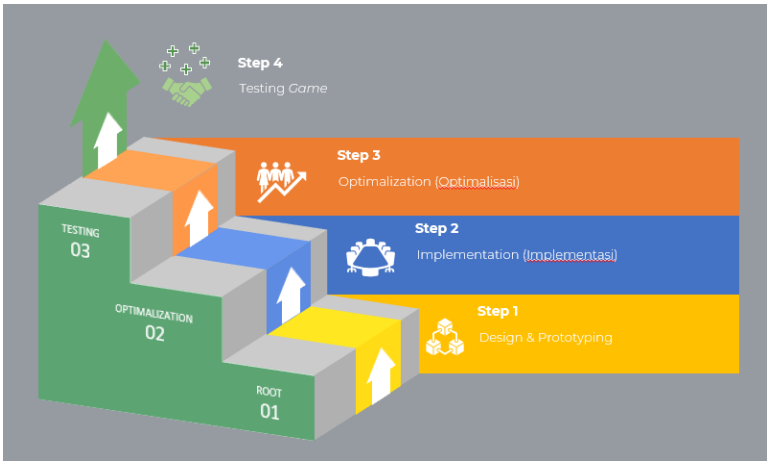
Dalam penelitian ini, semua animasi yang digunakan dapat diakses melalui *website* Mixamo, Poly.Pizza, dan Unity Assets Store. Perusahaan-perusahaan ini menyediakan layanan *rigging* dan animasi untuk model 3D secara gratis dan tanpa biaya royalti. Setelah animasi diunduh dengan sukses, langkah berikutnya adalah menerapkan proses *rigging* secara *procedural* pada objek musuh, atau musuh.

Dalam penelitian ini, perancangan *animation rigging* secara *procedural* dilakukan secara langsung dengan melakukan implementasi komponen *animation rigging* pada Unity Engine seperti berikut.



Gambar 3. 12 Enemy Procedural Rigging Animation pada Unity

3.3.2 Perancangan Teknis



Gambar 3. 13 Research Development Cycle based on GDLC (Game Development Life Cycle)

Arsitektur teknis sistem *Enemy AI Behaviour* pada *game* "NoKidnap" mencakup penggunaan Unity sebagai platform pengembangan utama, dengan NavMesh untuk navigasi AI, Animator untuk animasi karakter, dan sistem fisika Unity untuk interaksi dengan lingkungan. Sistem ini juga menggunakan C# sebagai bahasa pemrograman untuk mengembangkan logika *game* dan Visual Studio sebagai lingkungan pengembangan terintegrasi.

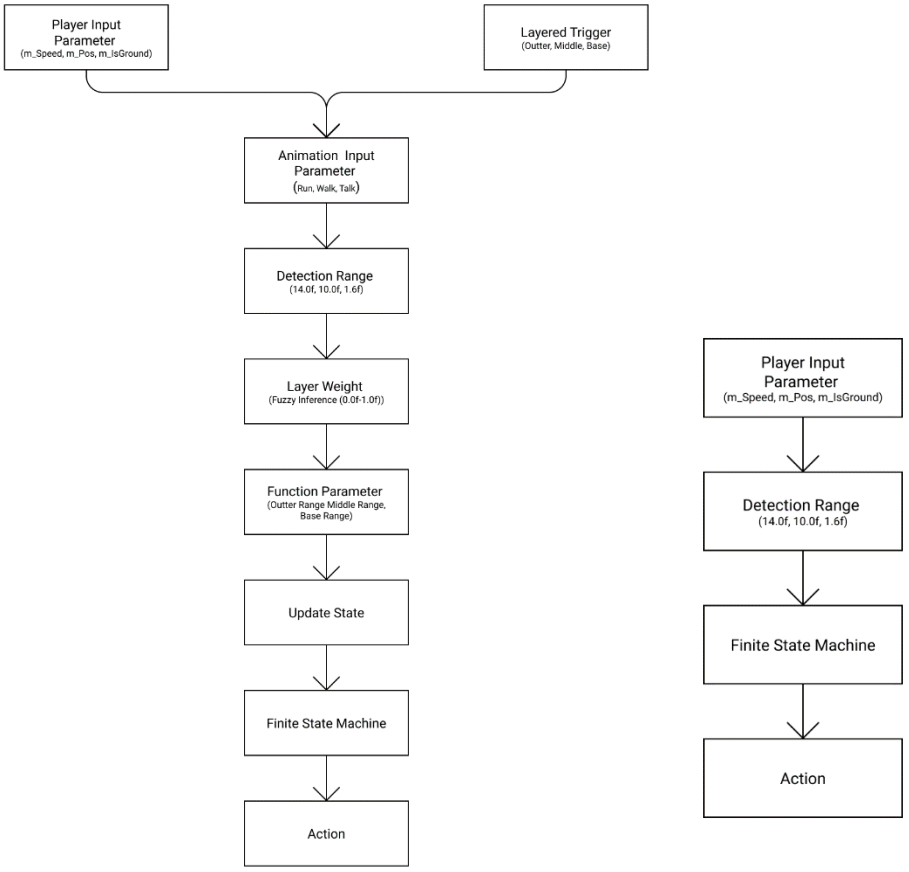
Perancangan sistem ini dimulai dari tahap *Design* dan *Prototyping*. Adapun pada tahap ini komponen-komponen sistem, seperti AI penculik, kontrol karakter pemain, dan antarmuka pengguna, dijelaskan dengan rinci untuk memastikan pemahaman yang jelas. Setelah itu tahap implementasi setiap komponen pada sisi *back-end* aplikasi akan dihubungkan dengan

sisi *front-end* (antarmuka pemain). Setelah proses implementasi berhasil, maka langkah selanjutnya adalah optimlasasi semua proses yang telah berhasil dilakukan, yang kemudian akan dilanjutkan pada tahap *testing* menggunakan *usability test*, sehinnga mendapatkan hasil yang sesuai objektif. Semua tahapan dalam alur perancangan teknis dilakukan menggunakan *tools* Unity Engine Platform.

Gambar 3. 14 Unity Engine

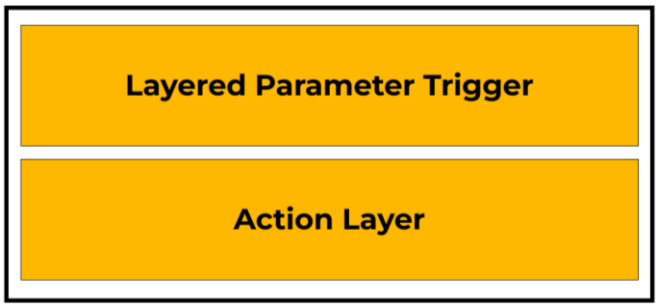
3.3.2.1 Perancangan *Layered Animation Trigger AI*

Secara garis besar, Sistem *enemy* AI yang dikembangkan bekerja dengan cara mengolah *triggers* atau pemicu-pemicu yang diterima oleh *enemy* dan menerjemahkannya menjadi sekumpulan aksi spesifik yang akan dilakukan *enemy*. Untuk tipe *enemy* normal, Hal ini dapat direalisasikan dengan menggunakan sistem FSM konvensional. Namun, untuk tipe *enemy* yang mampu mensimulasikan pergerakan, pendekatan *Layered Animation AI* yang digabungkan dengan FSM digunakan untuk menghasilkan perilaku yang diinginkan. Untuk penjelasan arsitektur *enemy* yang lebih mendetail akan dijelaskan sebagai berikut.



Gambar 3. 15(A) Diagram Sistem Enemy Layered Animation Trigger AI dan (B) Enemy Berbasis Finite-State Machine Konvensional

Dalam arsitektur sistem *Enemy Layered Animation Trigger* untuk *enemy* AI di *game* yang dikembangkan, terbagi menjadi 2 *layer* atau lapisan yang saling bekerja dan berinteraksi yaitu *Layered Parameter Trigger* (Pemicu Parameter Berlapis) dan *Action Layer* (Lapisan Aksi).



Gambar 3. 16 Diagram Layer atau lapisan dari Sistem Enemy Layered Animation Trigger AI

3.3.2.2 Pemanggilan Layer

3.3.2.2.1 Kalkulasi Jarak Berbentuk Lingkaran

Analisis jarak pada masing *layer* dapat dianalisis menggunakan sebuah perhitungan. Perhitungan ini digunakan untuk menentukan jarak antara posisi *player* dan musuh dalam lingkungan permainan. Hal ini penting untuk mengatur perilaku musuh berdasarkan jaraknya dari *player*.

Sehingga dapat digunakan rumus jarak Euclidean atau rumus Pythagoras untuk menghitung jarak antara dua titik dalam ruang tiga dimensi. Misalnya, jika koordinat player adalah (x_1, y_1, z_1) dan koordinat musuh adalah (x_2, y_2, z_2), maka jaraknya dapat dihitung menggunakan rumus:

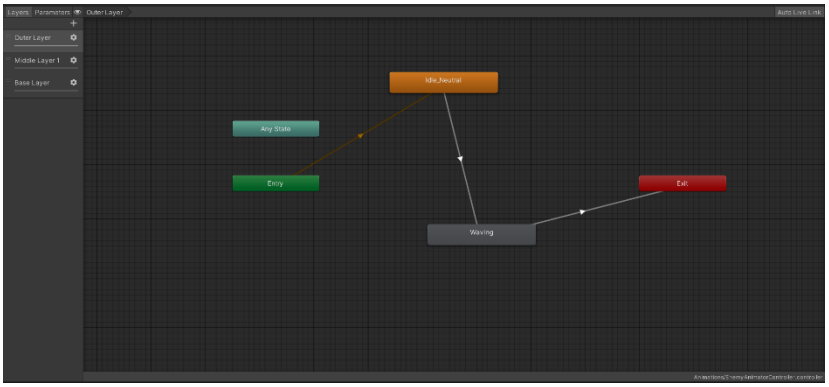
$$distance(d) = \sqrt{((x_2-x_1)^2+(y_2-y_1)2+(z_2-z_1)^2)} \quad (1)$$

Dalam hal ini perhitungan radius lingkaran dapat disimplifikasi menggunakan kode program (C#) seperti berikut:

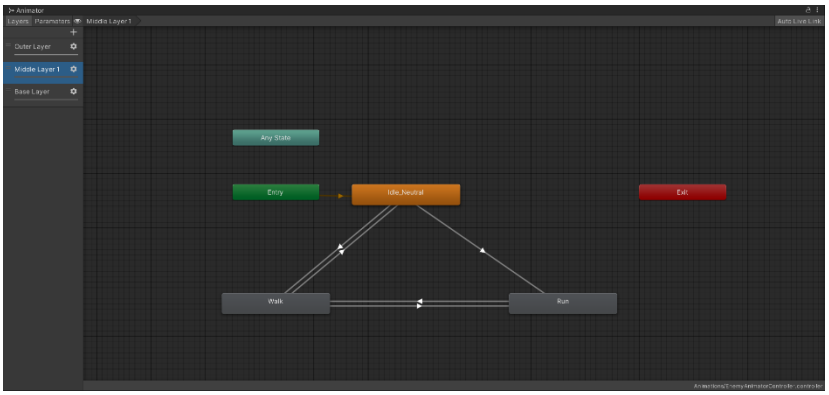
```
Vector3 toPlayer =
ThirdPersonController.Instance.transform.position -
detector.position;
toPlayer.y = 0;

if (toPlayer.magnitude <= outterDetectionRadius)
{
    // If enemy inside of the outterArea
    if (Vector3.Dot(toPlayer.normalized, detector.forward)>
        Mathf.Cos(detectionAngle * 0.5f * Mathf.Deg2Rad))
    {
        return ThirdPersonController.Instance;
    }
}
```

3.3.2.2.2 Penambahan *Animator Controller*

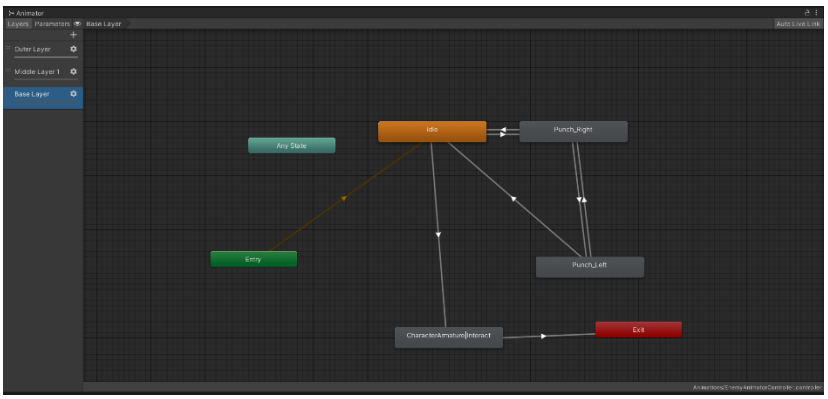
Setelah *game object* berhasil diinisalisasi maka *game object* akan ditambahkan komponen *Animator Controller*.



(A)



(B)



(C)

Gambar 3. 17 Animator Controller pada Object Enemy (A) Outter Layer (B) Middle Layer (C) Base Layer

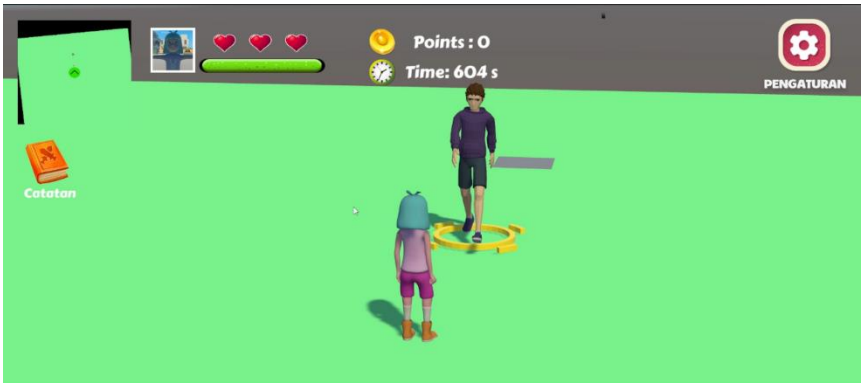
3.3.2.2.3 Penambahan Parameter pada Animator

Pada *Object Enemy* yang sudah dipasangkan komponen Animator Controller ditambahkan sebuah parameter yang memiliki tipe data seperti *boolean*, *float*, *integer*, dan *trigger*. Sehingga, dengan tipe data tersebut parameter dapat dipanggil ke dalam fungsi berupa *script*.

Gambar 3. 18 Penambahan Parameter pada Object Enemy Animator Controller

3.3.2.2.4 *Player Input Action*

Pada *Input Action Layer* karakter pada pemain mengambil peranan penting dalam terciptanya kondisi pemicu pada tingkat animasi penculik. Hal ini dikarenakan, setiap input yang diberikan oleh pemain, secara logika akan menghasilkan tindakan pemicu bagi NPC. Dalam penelitian ini karakter musuh (penculik) sekaligus NPC akan mendeteksi berbagai perintah inputan oleh pemain seperti berikut.



Gambar 3. 19 Player Character Input Action Walk

Gambar 3. 20 Player Character Input Action Stop

Gambar 3. 21 Player Character Input Action Run

3.3.2.2.5 Kalkulasi *Input Action Parameter*

Perhitungan ini digunakan untuk menentukan aksi yang didapatkan musuh terhadap *player input* (masukan). Hal ini penting untuk mengatur perilaku musuh berdasarkan *parameter input* oleh *player*.

Variasi hasil *parameter input* oleh *player* dapat dilihat pada Tabel 3.1.

Tabel 3. 1 Tabel Variasi Hasil Parameter Input oleh Player

Player Input	Action Parameter	Layer
m_Speed >= 5.335f	Run	Middle Layer
m_Speed >= 2.0f	Walk	Middle Layer
m_Speed >= 0.0f	Stop	Base Layer
{parameter input} $ {value}	Parameter Action Name	{Radius} Layer

Keterangan : {...} = *identifier*

$ = *input operator*

3.3.2.2.6 Kalkulasi Basis Logika Parameter

Perhitungan ini digunakan untuk mengatur waktu tunggu atau jeda antara aksi atau perilaku musuh. Hal ini penting untuk memberikan variasi dan alur permainan yang menarik.

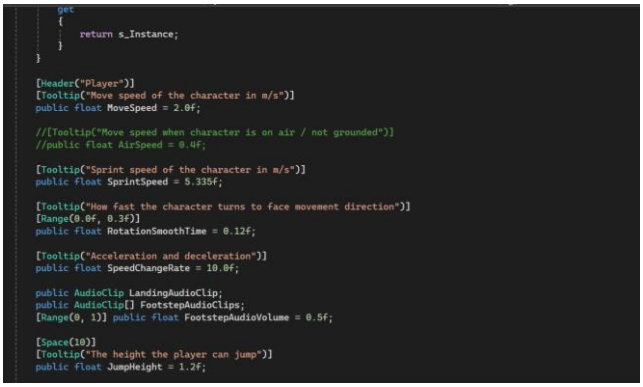
Digunakan rumus sederhana untuk menghitung durasi waktu atau jeda berdasarkan waktu yang diinginkan. Misalnya, jika musuh menunggu selama 3 detik sebelum melakukan aksi selanjutnya, maka nilai tersebut digunakan sebagai durasi waktu atau jeda. Kemudian dalam menentukan rotasi pemain dapat kembali pada posisi semula (*origin position*) apabila pemain (*player*) terdeteksi hilang selama beberapa detik pencarian. Beberapa fungsi dapat dipisah dan dapat kembali dalam fungsi "Invoke". Perhitungan Kalkulasi Basis Logika Parameter dapat dituliskan pada kode program (C#) sebagai berikut:

Perhitungan Basis Logika Parameter

```
private IEnumerator WaitOnPursuit()
{
    yield return new WaitForSeconds(timeToWaitOnPursuit);
m_EnemyController.SetFollowTarget(m_OriginPosition);
}
private IEnumerator StopOnWalk()
{
    m_EnemyController.StopFollowTarget(true);
    m_Animator.SetBool(m_HashOnPursuitPara, false);
    yield return new WaitForSeconds(timeToStopWalk);
    m_EnemyController.StopFollowTarget(false);
    m_Animator.SetBool(m_HashOnPursuitPara, true);
}
private void OnWaving()
{
    if (nearBase && isOutter)
    {
        m_Animator.SetTrigger(m_HashWaving);
    }
    if (Convert.ToBoolean(m_HashBaseTrigger))
    {
        m_Animator.SetTrigger(m_HashTalk);
    }
}
```

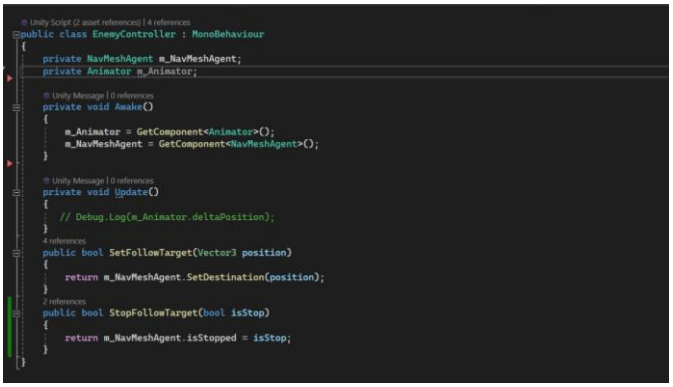
3.3.2.2.7 Penulisan Baris Kode (*Scripting*) Aplikasi

Diperlukan penulisan baris kode program atau *scripting* pada sisi *back-end* dalam menunjang kinerja fungsi kecerdasan buatan musuh.



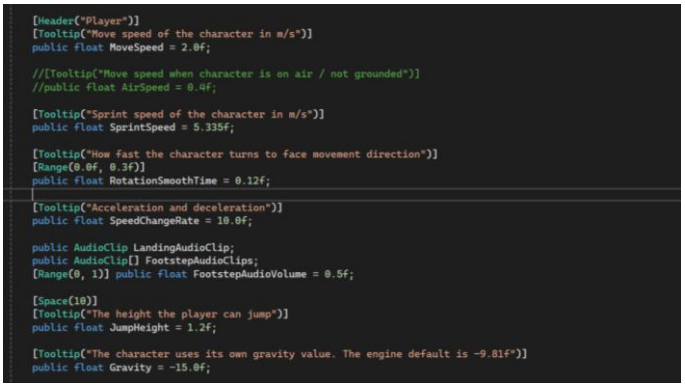
```
        get
        {
            return s_Instance;
        }
    }

    [Header("Player")]
    [Tooltip("Move speed of the character in m/s")]
    public float MoveSpeed = 2.0f;

    //[Tooltip("Move speed when character is on air / not grounded")]
    //public float AirSpeed = 0.4f;

    [Tooltip("Sprint speed of the character in m/s")]
    public float SprintSpeed = 5.335f;

    [Tooltip("How fast the character turns to face movement direction")]
    [Range(0.0f, 0.3f)]
    public float RotationSmoothTime = 0.12f;

    [Tooltip("Acceleration and deceleration")]
    public float SpeedChangeRate = 10.0f;

    public AudioClip LandingAudioClip;
    public AudioClip[] FootstepAudioClips;
    [Range(0, 1)] public float FootstepAudioVolume = 0.5f;

    [Space(10)]
    [Tooltip("The height the player can jump")]
    public float JumpHeight = 1.2f;
```

(A)



```
Unity Script (2 asset references) | 4 references
public class EnemyController : MonoBehaviour
{
    private NavMeshAgent m_NavMeshAgent;
    private Animator m_Animator;

    Unity Message | 0 references
    private void Awake()
    {
        m_Animator = GetComponent<Animator>();
        m_NavMeshAgent = GetComponent<NavMeshAgent>();
    }

    Unity Message | 0 references
    private void Update()
    {
      // Debug.Log(m_Animator.deltaPosition);
    }
    4 references
    public bool SetFollowTarget(Vector3 position)
    {
        return m_NavMeshAgent.SetDestination(position);
    }
    2 references
    public bool StopFollowTarget(bool isStop)
    {
        return m_NavMeshAgent.isStopped = isStop;
    }
}
```

(B)



```
    [Header("Player")]
    [Tooltip("Move speed of the character in m/s")]
    public float MoveSpeed = 2.0f;

    //[Tooltip("Move speed when character is on air / not grounded")]
    //public float AirSpeed = 0.4f;

    [Tooltip("Sprint speed of the character in m/s")]
    public float SprintSpeed = 5.335f;

    [Tooltip("How fast the character turns to face movement direction")]
    [Range(0.0f, 0.3f)]
    public float RotationSmoothTime = 0.12f;

    [Tooltip("Acceleration and deceleration")]
    public float SpeedChangeRate = 10.0f;

    public AudioClip LandingAudioClip;
    public AudioClip[] FootstepAudioClips;
    [Range(0, 1)] public float FootstepAudioVolume = 0.5f;

    [Space(10)]
    [Tooltip("The height the player can jump")]
    public float JumpHeight = 1.2f;

    [Tooltip("The character uses its own gravity value. The engine default is -9.81f")]
    public float Gravity = -15.0f;
```

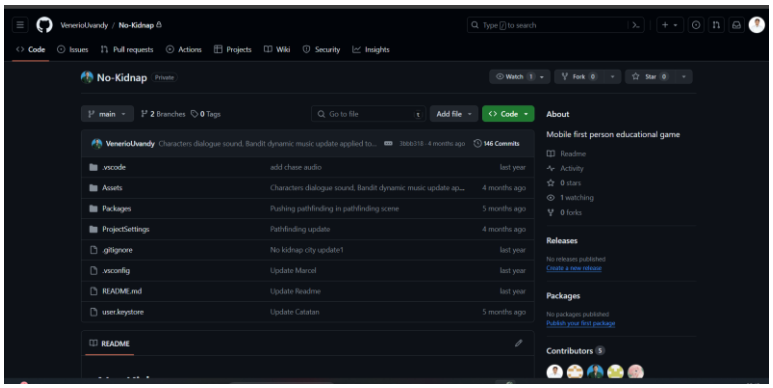
(C)

```
//private
private bool isOutter=false;
private bool nearBase;
private ThirdPersonContreller m_Target;
private EnemyContreller m_EnemyController;
private Animator m_Animator;
private float m_TimeSinceLostTarget = 0;
private Vector3 m_OriginPosition;
private Quaternion m_OriginRotation;

//Hash all the parameter
private readonly int m_HashOnPursuitPara = Animator.StringToHash("onPursuit");
private readonly int m_HashNearBasePara = Animator.StringToHash("NearBase");
private readonly int m_HashObserve = Animator.StringToHash("Observe");
private readonly int m_HashTalk = Animator.StringToHash("Talk");
private readonly int m_HashWaving = Animator.StringToHash("Waving");
private readonly int m_HashMiddleTrigger = Animator.StringToHash("Middle Trigger");
private readonly int m_HashBaseTrigger = Animator.StringToHash("Base Trigger");
private readonly int m_HashOutterTrigger = Animator.StringToHash("Outter Trigger");

Unity Message | 0 references
private void Awake()
{
    m_EnemyController = GetComponent<EnemyController>();
    m_OriginPosition = transform.position;
    m_OriginRotation = transform.rotation;
    m_Animator = GetComponent<Animator>();
}
```

(D)

Gambar 3. 22 Scripting Aplikasi (A) PlayerScanner (B) ThirdPersonController (C) EnemyController (D) NewBanditBehaviour

Dokumentasi penulisan baris kode program menggunakan bahasa pemrograman C# dan dapat dilihat dengan meminta akses kontribusi pada pemilik seperti pada Gambar 3.23.



Gambar 3. 23 Dokumentasi penulisan baris kode lengkap games "NoKidnap"
Sumber: https://github.com/VenerioUvandy/No-Kidnap

3.3.2.2.8 Menghubungkan parameter pada sisi *back-end* ke Unity Editor

Parameter pada sisi *back-end* dipanggil pada setiap fungsi transisi seperti pada Tabel 3.2.

Tabel 3. 2 Tabel Fungsi Transisi State pada Animator

Layer	Initial State	Final State	Transition Parameter
Outter	Idle_Neutral	Waving	Waving
Outter	Waving	Exit	None
Middle	Idle_Neutral	Walk	onPursuit

Middle	Walk	Idle_Neutral	NearBase
Middle	Walk	Run	MiddleTrigger
Middle	Run	Walk	MiddleTrigger
Middle	Idle_Neutral	Run	MiddleTrigger
Base	Idle	Interact	Talk
Base	Idle	Punch_Right	BaseTrigger
Base	Punch_Right	Idle	BaseTrigger
Base	Puch_Left	Idle	BaseTrigger
Base	Punch_Left	Punch_Right	BaseTrigger
Base	Punch_Right	Punch_Left	BaseTrigger
Base	Interact	Exit	None

3.3.2.2.9 Optimalisasi *Layered Animation Trigger*

Proses pengoptimalan pada masing-masing layer meliputi pembuatan *function* pada penulisan kode program. Hasil penulisan tersebut membentuk beberapa layer terpisah berdasarkan indikator jarak pada radius lingkaran. Sehingga, dapat dituliskan sebagai berikut:

Pemanggilan Fungsi OutterRange

```
private void OutterRange()
{
    isOutter = true;
    IndicatorSetup(true, false, false);
    Debug.Log("OutterDetectionTrigger");
    m_Animator.SetLayerWeight(0, 1f);
    m_Animator.SetLayerWeight(1, 0f);
    m_Animator.SetLayerWeight(2, 0f);
    m_Animator.SetBool(m_HashBaseTrigger, false);
    m_Animator.SetBool(m_HashMiddleTrigger, false);
    m_Animator.SetBool(m_HashOutterTrigger, true);
    m_Animator.SetBool(m_HashNearBasePara, false);
    m_Animator.SetBool(m_HashObserve, false);
    if (!nearBase)
    {
        m_Animator.SetLayerWeight(1, 1f);
        Debug.Log("Not nearBase but Outter");
        m_Animator.SetBool(m_HashOnPursuitPara, true);
    }
}
```

Pemanggilan Fungsi MiddleRange

```
private void MiddleRange()
{
    isOutter = false;
```

```
    IndicatorSetup(false, true, false);
    Debug.Log("MiddleDetectionTrigger");
    m_Animator.SetLayerWeight(0, 0f);
    m_Animator.SetLayerWeight(1, 1f);
    m_Animator.SetLayerWeight(2, 0f);
    Chase = true;
m_EnemyController.SetFollowTarget(m_Target.transform.positi
on);
    m_Animator.SetBool(m_HashBaseTrigger, false);
    if (m_Target.getSpeed() > 5)
    {
        IndicatorSetup(false, false, true);
        m_Animator.SetBool(m_HashMiddleTrigger, true);
    }
    else
    {
        m_Animator.SetBool(m_HashMiddleTrigger, false);
    }
    m_Animator.SetBool(m_HashOutterTrigger, false);
    m_Animator.SetBool(m_HashNearBasePara, false);
    m_Animator.SetBool(m_HashObserve, false);
    m_Animator.SetBool(m_HashOnPursuitPara, true);
}
```

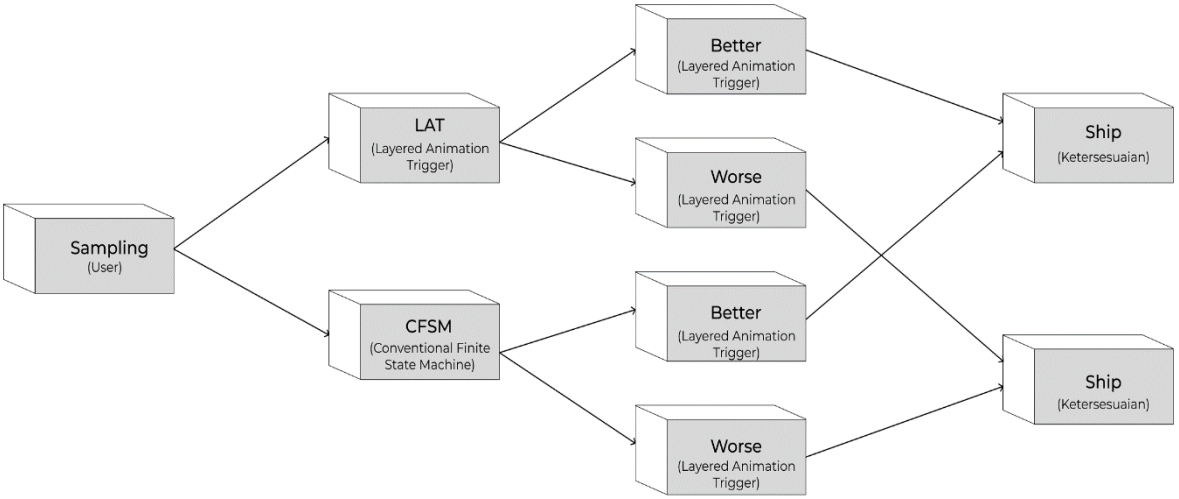
Pemanggilan Fungsi BaseRange

```
private void BaseRange()
{
    IndicatorSetup(false, true, false);
    m_Animator.SetLayerWeight(0, 0f);
    m_Animator.SetLayerWeight(1, 0.5f);
    m_Animator.SetLayerWeight(2, 1f);
    m_Animator.SetBool(m_HashBaseTrigger, false);
    m_Animator.SetBool(m_HashMiddleTrigger, false);
    m_Animator.SetBool(m_HashOutterTrigger, false);
    m_Animator.SetBool(m_HashNearBasePara, false);
    m_Animator.SetBool(m_HashObserve, false);
    if (m_Target.getSpeed() > 5)
    {
        m_Animator.SetBool(m_HashOnPursuitPara, false);
        m_Animator.SetBool(m_HashMiddleTrigger, true);
    }
    else if (m_Target.getSpeed() >= 2 &&
m_Target.getSpeed() < 5)
    {
        m_Animator.SetBool(m_HashOnPursuitPara, true);
        m_Animator.SetBool(m_HashMiddleTrigger, false);
    }
    else
    {
        StartCoroutine(StopOnWalk());
    }
}
```

3.4 Rencana Uji Coba (*Testing*) dan Evaluasi

Dalam pendekatan ini, akan disusun suatu eksperimen dengan menghadirkan sejumlah pemain sampel dalam skenario permainan di mana terdapat dua jenis

musuh yang berbeda, yang masing-masing diatur dengan dua sistem perilaku kecerdasan buatan yang berbeda pula. Satu jenis musuh akan menggunakan mekanisme *layered animation trigger*, sementara jenis musuh lainnya akan mengadopsi pendekatan perilaku konvensional yang sering digunakan, yaitu *Finite State Machine (FSM)*. Kunci dari pendekatan ini terletak pada pelaksanaan pengujian melalui Uji Ketergunaan (*Usability Testing*), di mana para pemain sampel akan diberitahu tentang perbedaan antara kondisi sebelum dan setelah tahap optimalisasi dalam sistem perilaku kedua implementasi dilakukan.



Gambar 3. 24 Diagram Randomized Control Trial (RCT) untuk metodologi eksperimental "Pengembangan Model Layered Animation Trigger dalam Sistem Perilaku Enemy AI Behaviour dalam Aplikasi Permainan "NoKidnap"

3.5 Teknik Analisis Data

Dalam penelitian ini, dilakukan analisis perbandingan antara dua jenis sistem perilaku AI yang berbeda, yaitu *sistem layered animation trigger* dan *Finite State Machine* (FSM). Sistem *layered animation trigger* menggunakan pendekatan yang memanfaatkan *animator layer* dan *trigger* untuk mengatur perilaku musuh, sementara FSM mengadopsi model berbasis keadaan yang terdiri dari serangkaian keadaan yang dapat berubah sesuai dengan kondisi lingkungan. Perbandingan dilakukan melalui pengujian kinerja kedua sistem pada skenario *game* yang sama, dengan melibatkan sampel pemain yang ditempatkan dalam kondisi eksperimental yang serupa.

Perbandingan dilakukan menggunakan metode kualitatif dengan tipe data skala ordinal. Skala ordinal dipilih dengan melakukan proses pengukuran menggunakan skala Likert untuk mengukur pendapat, sikap, atau persepsi seseorang terhadap suatu pernyataan atau topik. Skala Likert digunakan dalam menentukan serangkaian pilihan tanggapan.

BAB IV
IMPLEMENTASI DAN PENGUJIAN SISTEM

Bab ini menjelaskan hasil implementasi dan pengujian sistem pada Aplikasi Permainan ”NoKidnap”. Hasil implementasi dan pengujian sistem penelitian dituliskan berdasarkan objektif penelitian mulai dari tahap pengembangan, implementasi sistem *layered animation trigger*, dan pengujian sistem menggunakan *usability testing*.

4.1 Hasil Implementasi Aplikasi Permainan “No Kidnap”

4.1.1 Tampilan Lingkungan (*Environment)* Aplikasi

Tampilan lingkungan (*environment)* aplikasi mencerminkan karakter Leafy yang baru saja pulang sekolah, terlihat pada Gambar 4.1 Karakter Leafy menunjukan perspektif orang ketiga pada tampilan antarmuka aplikasi. Tampilan ini berfungsi dalam memvisualiskan objek/ruang yang lebih jelas serta tampilan ruang yang bersifat spasial.

Gambar 4. 1 Tampilan Dunia Game “No Kidnap” berdasarkan Perspektif Pemain

Gambar 4.2 menunjukkan momen ketika Leafy sedang dalam perjalanan pulang dan bertemu dengan ibu guru yang memberikan peringatan. Dalam gambar ini, ibu guru terlihat mengingatkan Leafy untuk berhati-hati selama perjalanan pulang. Terdapat dialog antara Leafy dan ibu guru yang tertulis dalam bentuk balon kata, menambah dimensi naratif pada gambar. Dialog tersebut berisi pesan-pesan keselamatan, seperti "Hati-hati di jalan, Leafy!" dan respons Leafy

yang mengindikasikan pemahaman dan kepatuhannya. Latar belakang gambar ini memperlihatkan area sekitar sekolah dengan beberapa siswa lain yang juga sedang pulang. Gambar ini tidak hanya menekankan pentingnya keselamatan bagi anak-anak, tetapi juga memperlihatkan interaksi positif antara guru dan murid, yang mendukung tema keseluruhan dari cerita dan memberikan konteks lebih lanjut tentang perjalanan Leafy.

Gambar 4. 2 Tampilan Pemain berinteraksi dengan NPC (Guru)

4.1.2 Hasil Implementasi UI (*User Interface*)

UI (*User Interface*) berfungsi dalam melakukan proses interaksi yang efektif antara pengguna dan aplikasi melalui pengalaman visual. Tampilan ini juga berfungsi dalam membantu pengguna melakukan navigasi antar halaman (*scene*) yang berbeda.

Gambar 4. 3 Tampilan UI Game Menu

Gambar 4.3 menunjukkan hasil impementasi tampilan antar muka pada UI (*User Interface*) *Game Menu*. UI ini berfungsi sebagai tampilan awal ketika menjalankan aplikasi. Pada tampilan kita disuguhkan dengan beberapa tombol yang dapat ditekan sebagai navigasi, seperti tombol panduan orang tua, tombol

pengaturan, tombol bermain, dan tombol keluar.

Gambar 4. 4 Tampilan UI Game Menu Settings

Gambar 4.4 menunjukkan hasil impementasi tampilan UI (User Interface) *Game Menu Settings*. tampilan ini berfungsi dalam mengatur volume musik ketika bermain.

Gambar 4. 5 Tampilan UI Parents Guide

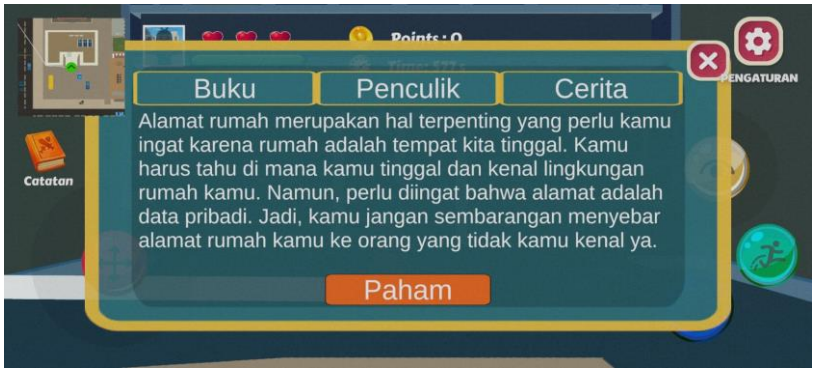
Gambar 4.5 menunjukkan hasil impementasi tampilan UI (*User Interface*) panduan orang tua *(parents guide)*. tampilan ini bertujuan dalam menyampaikan alur panduan bermain, tips, serta beberapa istilah penting yang harus dipahami oleh orang tua.

Gambar 4. 6 Tampilan UI Level Selection

Gambar 4.6 menunjukkan hasil impementasi tampilan UI (*User Interface*) *Level Selection*. Fitur ini memungkinkan pemain dapat beralih antar satu *level* (tingkatan) ke *level* (tingkatan) lainnya. Pada tampilan ini terdapat gambar yang dapat ditekan, sehingga memunculkan halaman (*scene*) bermain berikutnya. Pada sisi atas kanan juga terdapat tombol *exit* yang dapat membantu pemain kembali pada halaman (*scene)* sebelumnya.

Gambar 4. 7 Tampilan UI Sounds & Settings

Gambar 4.7 menunjukkan hasil impementasi tampilan UI (*User Interface*) *Sounds & Settings Level Selection*. Fitur ini memungkinkan pemain dapat besaran (*volume*) musik dan sensitivitas gerakan dalam permainan.



Gambar 4. 8 Tampilan UI Catatan

Gambar 4.8 menunjukkan hasil impementasi tampilan UI Catatan. Tampilan antar muka ini bertujuan dalam membantu pengguna dalam melihat perkembangan mereka dalam bermain, sekaligus menampilkan catatan singkat terkait materi pembelajaran yang sudah dipelajari selama bermain.

Gambar 4. 9 Tampilan UI Game Over

Gambar 4.9 menunjukkan tampilan UI *Game Over*. Tampilan antarmuka ini menunjukan hasil pencapaian akhir selama bermain, berbeda dengan tampilan antar muka *game over* pada umumnya, pada kasus aplikasi permainan anak-anak, hal-hal yang memicu emosi negatif seperti kekalahan, tekanan untuk menyelesaikan tugas, dan sanksi dihindarkan. Sebaliknya emosi negatif ditampilkan dalam memicu semangat atau motivasi anak dalam menyelesaikan edukasi di dalam aplikasi *game.*

4.2 Hasil Uji Program Dalam *Game*

4.2.1 Hasil Uji pada Jarak *Outter Range*

Hasil perhitungan menunjukkan bahwa pada jarak maksimum *Outter Range*,

distanceRange < 14.0f, maka *Layer Outter* akan diaktifkan sehingga dapat memicu skala animasi pada parameter tersebut. Hasil uji program dan skalasi perhitungan dapat dilihat pada Gambar 4.10 dan Tabel 4.1.

Gambar 4. 10 Hasil Uji Parameter Wave pada Jarak Outter

Gambar 4.10 menampilkan hasil uji parameter "Walk" pada jarak 14.0f atau 14 meter dalam lingkungan aplikasi. Jarak ini dapat dimanipulasi dalam lingkungan pengembangan sehingga dapat memunculkan situasi yang lebih dinamis pada aplikasi.

Tabel 4. 1 Tabel kalkulasi hasil pada Outter Range berdasarkan Parameter, Layer Weight, dan Input Action Parameter

Outter Parameter	***Layer Weight***	***Input Action***
Enemy Wave	1.0f	m_speed >= 0.0f
Enemy Observe	1.0f	m_speed = 0.0f

Tabel 4.1 menunjukkan hasil kalkulasi jarak yang dapat dimanipulasi menggunakan kombinasi Input Action m_speed, yakni kecepatan pemain saat diam dan bergerak. Selain itu skala *Layer Weight* menunjukkan pada skala 1.0f animasi menggunakan parameter bekerja secara penuh. Animasi pada parameter tersebut dapat dimanipulasi sehingga dapat menampilkan transisi gerakan dengan skala 0 – 1.

4.2.2 Hasil Uji pada Jarak *Middle Range*

Hasil perhitungan menunjukkan bahwa pada jarak maksimum *Middle Range*, distanceRange < 10.0f, maka *Layer Middle* akan diaktifkan sehingga dapat

memicu skala animasi pada Parameter tersebut. Hasil uji program dapat dilihat pada Gambar 4.11 dan Gambar 4.12, serta hasil skalasi perhitungan dapat dilihat pada Tabel 4.2.

Gambar 4. 11 Hasil Uji Parameter Walk pada Jarak Middle

Gambar 4.11 menampilkan hasil uji parameter “Walk” pada jarak 10.0f atau 10 meter dalam lingkungan aplikasi. Jarak ini dapat dimanipulasi dalam lingkungan pengembangan sehingga dapat memunculkan situasi yang lebih dinamis pada aplikasi.

Gambar 4. 12 Hasil Uji Parameter Run pada Jarak Middle

Pada jarak yang sama, parameter “Run” juga bekerja pada jarak 10.0f atau 10 meter dalam lingkungan aplikasi seperti pada Gambar 4.12. Jarak ini juga dapat dimanipulasi dalam lingkungan pengembangan sehingga dapat memunculkan situasi yang lebih dinamis pada aplikasi.

Tabel 4. 2 Tabel kalkulasi hasil pada Middle Range berdasarkan Parameter, Layer Weight, dan Input Action Parameter

Middle Parameter	***Layer Weight***	***Input Action***
Enemy Walk	0.0f - 1.0f	m_speed >= 2.0f
Enemy Hit	1.0f	m_HashMiddleTrigger = true
Enemy Run	0.0f - 1.0f	m_speed >= 5.0f

Tabel 4.2 menunjukkan hasil kalkulasi jarak yang dapat dimanipulasi menggunakan kombinasi *Input Action* m_speed, yakni kecepatan pemain saat berjalan, menyerang, ataupun berlari. Selain itu skala Layer Weight menunjukkan skala 0.0f – 1.0f, sehingga ketika *input action* tidak bekerja *layer weight* akan berkurang perlahan menuju ke skala 0. Animasi pada parameter tersebut dapat dimanipulasi sehingga dapat menampilkan transisi gerakan dengan skala 0 – 1.

4.2.3 Hasil Uji pada Jarak *Base Range*

Hasil perhitungan menunjukkan bahwa pada jarak maksimum *Base Range*, distanceRange < 1.6f, maka Layer *Base* akan diaktifkan sehingga dapat memicu skala animasi pada Parameter tersebut. Hasil uji program dan skalasi perhitungan dapat pada Gambar 4.13 dan Tabel 4.3.

Gambar 4. 13 Hasil Uji Parameter Talk pada Jarak Base

Pada Gambar 4.11, ditunjukkan bahwa hasil uji parameter “Walk” pada jarak 1.6f atau 1.6 meter dalam lingkungan aplikasi memicu animasi *walk* (berjalan), *talk* (berbicara), dan juga *run* (berlari). Jarak ini dapat dimanipulasi dalam lingkungan pengembangan sehingga dapat memunculkan situasi yang lebih

dinamis pada aplikasi.

Tabel 4. 3 Tabel kalkulasi hasil pada Base Range berdasarkan Parameter, Layer Weight, dan Input Action Parameter

Base Parameter	***Layer Weight***	***Input Action***
Enemy Walk	0.0f - 1.0f	m_speed >= 2.0f
Enemy Talk	1.0f	m_HashTalk = true
Enemy Run	0.0f - 1.0f	m_speed >= 5.0f

Tabel 4.3 menunjukkan hasil kalkulasi jarak yang dapat dimanipulasi menggunakan kombinasi *Input Action* m_speed dan m_HashTalk, yakni kecepatan pemain saat berjalan, dan juga aksi pemain ketika berbicara. Selain itu skala Layer Weight menunjukkan skala 0.0f – 1.0f, sehingga ketika *input action* tidak bekerja *layer weight* akan berkurang perlahan menuju ke skala 0. Animasi pada parameter tersebut dapat dimanipulasi sehingga dapat menampilkan transisi gerakan dengan skala 0 – 1.

4.3 Hasil Eksperimen

Peneliti telah melakukan eksperimen kepada total 31 orang sampel *play-tester*. Setiap *Play-tester* diberikan akses untuk memainkan dan membandingkan dua buah aplikasi *game* sebelum dan sesudah diimplementasikan fitur AI (*Artificial Intelligence*) *Layered Animation Trigger* menggunakan metode skala Likert. Melalui kuesioner yang terstruktur, setiap *play-tester* diminta untuk menilai berbagai aspek permainan, seperti *overall gameplay*, optimalisasi, dan efektivitas AI (*Artificial Intelligence*) dengan menggunakan skala Likert yang berfokus pada penilaian subjektif dengan skala 1-5 tergantung dari pertanyaan dalam kuisioner.

4.3.1 Pengambilan *Gameplay* Secara Menyeluruh

Responden diminta untuk menilai bagaimana pengalaman secara keseluruhan mengenai *game* yang telah dimainkan.

Tabel 4. 4 Jawaban Responden Mengenai Seberapa mendalam mereka merasa terdapat perubahan dalam game? (1 = Tidak mendalam, 5 = Sangat mendalam)

Tanggapan	**Bobot Skor**	**Frekuensi**	**Jumlah Skor**	**Persentase**
Sangat Mendalam	5	18	90	58,06%
Mendalam	4	3	12	9,67%
Netral	3	3	9	9,67%
Sedikit Mendalam	2	4	8	12,9%
Tidak Mendalam	1	3	3	9,67%
Total		31	113	100%

Berdasarkan tabel di atas, mayoritas responden (79,64%) memberikan penilaian "Sangat Mendalam" terhadap pengalaman bermain *game*. Jumlah responden yang merasa "Mendalam" juga menunjukkan penilaian "Mendalam" sebanyak (10,61%). Lalu, Hanya sedikit yang memberikan penilaian "Tidak Mendalam", yakni sebesar 2.65%, yang mungkin menunjukkan bahwa mayoritas pemain mampu merasakan perbedaan pengalaman bermain yang lebih baik dan memuaskan dibanding pengalaman bermain prototipe sebelumnya. Dari hal ini dapat disimpulkan bahwa mayoritas responden memberikan penilaian positif terhadap tingkat kedalaman perbedaan pengalaman pada *game*.

Tabel 4. 5 Jawaban Responden Mengenai Sejauh Apa Mereka Merasa Pengalaman Gameplay Ini Menarik? (1 = Tidak menarik, 5 = Sangat menarik)

Tanggapan	**Bobot Skor**	**Frekuensi**	**Jumlah Skor**	**Persentase**
Sangat Menarik	5	17	85	54,83%
Menarik	4	6	24	19,35%
Netral	3	4	12	12,9%
Sedikit Menarik	2	4	8	12,9%
Tidak Menarik	1	-	-	-
Total		31	129	100%

Sebagaimana ditunjukkan dalam tabel di atas, sebagian besar pemain (54,83%) menganggap *game* ini sebagai "Sangat Menarik", sementara sebagian besar orang juga menganggapnya sebagai "Menarik" (19,35%). Ini menunjukkan bahwa *game* ini sangat baik untuk dibawa ke khalayak umum, karena mayoritas responden menilainya menarik.

4.3.2 Pertemuan dengan Penculik

Responden diminta untuk menilai bagaimana pengalaman dalam pertemuan mereka dengan kedua tipe jenis penculik.

Tabel 4. 6 Jawaban Responden Mengenai Seberapa Berkesan Pertemuan Mereka Dengan Berbagai Musuh Di Dalam Game? (1 = Tidak berkesan, 5 = Sangat berkesan)

Tanggapan	**Bobot Skor**	**Frekuensi**	**Jumlah Skor**	**Persentasi**
Sangat Berkesan	5	11	55	35,48%
Berkesan	4	9	36	29,03%
Netral	3	8	24	25,8%
Sedikit Berkesan	2	3	6	9,67%
Tidak Berkesan	1	-	-	-
Total		31	111	100%

Berdasarkan tabel di atas, mayoritas responden (35,48%) menganggap pertemuan mereka dengan berbagai jenis musuh di game itu "Berkesan", dan sebagian besar responden (29,03%) juga menganggap pertemuan itu "Sangat Berkesan". Ini menunjukkan bahwa desain penculik dan interaksi dengan mereka sangat dihargai.

Tabel 4. 7 Jawaban Responden Mengenai Sejauh Apa Mereka Melihat Perbedaan Taktik Atau Perilaku Yang Jelas Antara Tipe Musuh Yang Berbeda? (1 = Tidak ada perbedaan, 5 = Perbedaan yang sangat jelas)

Tanggapan	**Bobot Skor**	**Frekuensi**	**Jumlah Skor**	**Persentase**
Perbedaan yang sangat jelas	5	18	90	58,06%

Perbedaan yang jelas	4	6	24	19,35%
Netral	3	6	18	19,35%
Sedikti Perbedaan	2	-	-	-
Tidak Ada Perbedaan	1	1	1	3,22%
Total		31	133	100%

Berdasarkan tabel di atas, Sejumlah besar responden (58,06%) melihat “perbedaan yang sangat jelas” dalam taktik atau perilaku antara tipe penculik yang berbeda dengan dua kelompok besar responden lainnya (19.35%) melihat “perbedaan yang jelas” dan “netral”. Dari hal ini dapat disimpulkan bahwa mayoritas responden memiliki tanggapan positif terhadap perbedaan dalam taktik atau perilaku antara tipe penculik yang berbeda di dalam *game* ini, menunjukkan desain perbedaan karakter musuh yang cukup efektif.

4.3.3 Pertanyaan Spesifik dengan *Layered Animation Trigger*

Responden diminta untuk menilai bagaimana tanggapan mereka mengenai aspek emosi yang mereka observasi berdasarkan *enemy* yang mereka jumpai.

Tabel 4. 8 Jawaban Responden Mengenai Seberapa Jelas Mereka Merasakan Adanya Emosi Atau Perilaku Dari Musuh Selama Bermain Game? (1 = Tidak jelas, 5 = Sangat jelas)

Tanggapan	**Bobot Skor**	**Frekuensi**	**Jumlah Skor**	**Persentase**
Sangat Jelas	5	16	80	51,61%
Jelas	4	8	32	25,8%
Netral	3	6	18	19,35%
Sedikit Jelas	2	1	2	3,22%
Tidak Jelas	1	-	-	-
Total		31	132	100%

Menurut tabel di atas, sebagian besar orang yang menjawab (51,61%) mengatakan bahwa mereka melihat "perbedaan *gesture* yang sangat jelas" selama bermain

game, dan sebagian besar orang lain (25,8%) juga mengatakan bahwa mereka melihat "perbedaan *gesture* yang jelas". Kesimpulannya menunjukkan bahwa sebagian besar orang yang menjawab mengalami perubahan *gesture* yang jelas selama bermain *game*. Ini menunjukkan betapa efektifnya sistem yang digunakan untuk mengubah perilaku penculik pada musuh.

Tabel 4. 9 Jawaban Responden Mengenai Sejauh Apa Mereka Merasakan Bahwa Perilaku Musuh Dipengaruhi Oleh Emosi Selama Permainan? (1 = Tidak dipengaruhi, 5 = Sangat dipengaruhi)

Tanggapan	**Bobot Skor**	**Frekuensi**	**Jumlah Skor**	**Persentase**
Sangat Dipengaruhi	5	19	95	61,29%
Lumayan Dipengaruhi	4	5	20	16,12%
Netral	3	5	15	16,12%
Sedikit Dipengaruhi	2	1	2	3,22%
Tidak Dipengaruhi	1	1	1	3,22%
Total		31	133	100%

Berdasarkan tabel di atas, Mayoritas besar responden (61,29%) merasakan bahwa “Perilaku musuh sangat dipengaruhi oleh pergerakan pemain”. Di samping itu dua kelompok besar responden (16,12%) juga merasakan bahwa "Perilaku penculik lumayan dipengaruhi oleh pergerakan pemain" dan, sebagian kelompok lagi merasakan bahwa perubahan perilaku bersifat “netral” selama permainan. Hasil tersebut menunjukkan bahwa sebagian besar responden merasakan adanya pengaruh pergerakan pemain pada perilaku musuh selama bermain *game* ini, menandakan efektivitas implementasi sistem *Layered Animation Trigger* berjalan efektif pada *game* ini.

4.3.4 Perbandingan Jenis *State* (Optimalitas)

Tabel 4. 10 Jawaban Responden Mengenai Seberapa Jelas Perbedaan Perilaku Antara Kedua Jenis Musuh Yang Mereka Temui? (1 = Tidak Jelas, 5 = Sangat Jelas)

Tanggapan	**Bobot Skor**	**Frekuensi**	**Jumlah Skor**	**Persentase**
Sangat Jelas	5	17	85	54,83%
Jelas	4	10	40	32,25%

Netral	3	4	12	12,9%
Sedikit Jelas	2	-	-	-
Tidak Jelas	1	-	-	-
Total		31	137	100%

Menurut tabel di atas, sebagian besar orang yang menjawab (54,83%) menganggap bahwa "Perbedaan perilaku antara kedua jenis penculik sangat jelas." Sebagian besar orang yang menjawab (32,25%) juga menganggap bahwa "Perbedaan perilaku antara kedua jenis penculik sangat jelas." Hal ini menunjukkan bahwa sebagian besar responden memiliki penilaian yang jelas atas penerapan perbedaan perilaku antara kedua jenis musuh dalam permainan, yang menunjukkan keberhasilan dalam membedakan perilaku penculik dalam *game*.

Tabel 4. 11 Jawaban Responden Mengenai Sejauh Mana Mereka Mampu Membedakan Perilaku Antara Kedua Jenis Musuh Tersebut? (1 = Tidak Dapat Dibedakan, 5 = Sangat Dapat Dibedakan)

Tanggapan	**Bobot Skor**	**Frekuensi**	**Jumlah Skor**	**Persentase**
Sangat Dapat Dibedakan	5	17	85	54,83%
Dapat Dibedakan	4	7	28	22,58%
Netral	3	6	18	19,35%
Sedikit Dapat Dibedakan	2	1	2	3,22%
Tidak Dapat Dibedakan	1	-	-	-
Total		31	133	100%

Tabel di atas menunjukkan bahwa sebagian besar responden (54.83%) merasa "sangat dapat dibedakan" antara perilaku kedua jenis penculik dalam permainan, dan sebagian besar responden bahkan merasa "dapat dibedakan". Dengan demikian, dapat disimpulkan bahwa mayoritas responden memiliki kemampuan yang cukup untuk membedakan.

4.3.5 Pengalaman Uji Buta

Tabel 4. 12 Jawaban Responden Mengenai Seberapa Besar Harapan Mereka Akan Perbedaan Perilaku Musuh Sebelum Uji Buta? (1 = Tidak Ada Harapan 5 = Harapan tinggi)

Tanggapan	Bobot Skor	Frekuensi	Jumlah Skor	Persentase
Harapan Tinggi	5	17	85	54,83%
Harapan	4	9	36	29,03%
Netral	3	5	15	16,12%
Sedikit Harapan	2	-	-	6.45%
Tidak Ada Harapan	1	-	-	-
Total		31	114	100%

Berdasarkan tabel di atas, sebagian besar responden (54,83%) menunjukkan "Harapan yang tinggi" terhadap perbedaan perilaku penculik sebelum uji buta, dan sebagian besar responden (29,03%) juga menunjukkan "Harapan" terhadap perbedaan tersebut. Ini menunjukkan kecenderungan umum bahwa harapan terhadap perbedaan perilaku musuh sebelum uji buta cenderung tinggi. Namun, sejumlah besar responden tetap memiliki harapan yang signifikan.

Tabel 4. 13 Jawaban Responden Mengenai Sejauh mana mereka terkejut dengan adanya perilaku yang tidak terduga selama uji buta? (1 = Tidak terkejut, 5 = Sangat terkejut)

Tanggapan	Bobot Skor	Frekuensi	Jumlah Skor	Persentase
Sangat Terkejut	5	1	5	64,51%
Terkejut	4	14	56	16,12%
Netral	3	14	12	12,9%
Sedikit Terkejut	2	2	4	3,22%
Tidak Terkejut	1	-	-	3,22%
Total		31	77	100%

Perilaku yang tidak terduga selama uji buta membuat mayoritas responden (64,51%) "Sangat Terkejut", sementara sebagian kecil responden lainnya

(16,12%) merasa "Terkejut". Hal ini menunjukkan bahwa mayoritas orang yang menjawab menunjukkan reaksi yang cenderung sangat terkejut terhadap perilaku yang tidak terduga selama uji buta, sementara jumlah orang yang benar-benar terkejut relatif kecil.

Berdasarkan data dari jawaban kuesioner yang telah dikumpulkan, maka didapatkan sebuah tabel rekapitulasi skor total untuk menghitung setiap sub indikator pertanyaan, sebagai berikut:

Tabel 4. 14 Tabel Total Skor Untuk Setiap Sub Indikator Pertanyaan Kuisioner

No	**Pertanyaan**	**Skor**
1	Pengalaman *Gameplay* Secara Keseluruhan	
a.	Seberapa mendalam anda merasa terdapat perubahan dalam *game*? (1 = Tidak mendalam, 2 = Cukup, 3 = Sedang, 4 = Cukup Mendalam 5 = Sangat mendalam)	113
b.	Sejauh apa anda merasa bahwa perubahan AI dalam *game* menambah pengalaman gameplay ini menarik? (1 = Tidak mendalam, 2 = Cukup, 3 = Sedang, 4 = Cukup Mendalam 5 = Sangat mendalam)	129
2	Pertemuan dengan Penculik	
a.	Seberapa berkesan pertemuan Anda dengan penculik di dalam *game*? (1 = Tidak mendalam, 2 = Cukup, 3 = Sedang, 4 = Cukup Mendalam 5 = Sangat mendalam)	111
b.	Bisakah Anda melihat perbedaan taktik atau perilaku yang jelas antara tipe musuh baru dan lama? (1 = Tidak mendalam, 2 = Cukup, 3 = Sedang, 4 = Cukup Mendalam 5 = Sangat mendalam)	133
3	Pertanyaan Spesifik tentang *Layered Animation Trigger*	
a.	Seberapa jelas Anda merasakan adanya *gesture* atau perilaku dari musuh selama bermain *game*? (1 = Tidak mendalam, 2 = Cukup, 3 = Sedang, 4 = Cukup Mendalam 5 = Sangat mendalam)	132
b.	Sejauh apa Anda merasakan bahwa perilaku penculik dipengaruhi oleh pergerakan pemain? (1 = Tidak mendalam, 2 = Cukup, 3 = Sedang, 4 = Cukup Mendalam 5 = Sangat mendalam)	133
4	Perbandingan Jenis *State*	
a.	Seberapa jelas perbedaan perilaku antara kedua jenis penculik yang Anda temui? (1 = Tidak mendalam, 2 = Cukup, 3 = Sedang, 4 = Cukup Mendalam 5 = Sangat mendalam)	137

b.	Sejauh mana Anda mampu membedakan perilaku antara jenis penculik tersebut? (1 = Tidak mendalam, 2 = Cukup, 3 = Sedang, 4 = Cukup Mendalam 5 = Sangat mendalam)	133
5	Pengalaman Uji Buta	
a.	Seberapa besar harapan Anda akan perbedaan perilaku penculik sebelum uji buta? (1 = Tidak mendalam, 2 = Cukup, 3 = Sedang, 4 = Cukup Mendalam 5 = Sangat mendalam)	114
b.	Sejauh mana Anda terkejut dengan adanya fitur yang tidak terduga selama uji buta? (1 = Tidak mendalam, 2 = Cukup, 3 = Sedang, 4 = Cukup Mendalam 5 = Sangat mendalam)	77
Total		1212

Setelah skor total dari setiap sub indikator kuisioner dikumpulkan, analisis indeks minimum dilakukan untuk menghitung variabel pengalaman bermain dan reaksi terhadap perilaku penculik dalam *game*. Sangat Rendah (SR), Rendah (R), Sedang (S), Tinggi (T), dan Sangat Tinggi (ST) adalah lima rentang nilai yang digunakan untuk menghitung skor total. Dengan menggunakan teknik ini, analisis dapat dihitung sebagai berikut:

$$Rentang\ Nilai = nilai\ skor\ \times jumlah\ item\ pertanyaan\ \times jumlah\ responden \quad (2)$$

Sehingga, diperoleh:

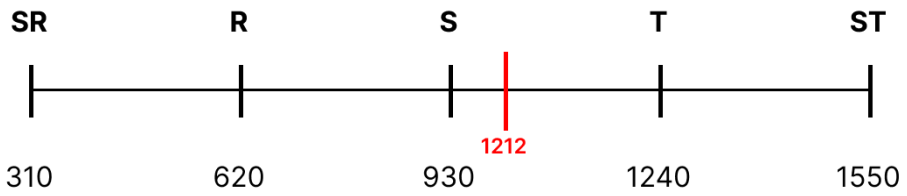
Sangat Rendah (SR) = 1 x 10 x 31 = 310

Rendah (R) = 2 x 10 x 31 = 620

Sedang (S) = 3 x 10 x 31 = 930

Tinggi (T) = 4 x 10 x 31 = 1240

Sangat Tinggi (ST) = 5 x 10 x 31 = 1550



Gambar 4. 14 Garis Kontinum Analisis Minimum Score Index (MSI)

Hasil penelitian ini menunjukkan bahwa pengalaman bermain dan respons terhadap perilaku penculik dalam *game* "NoKidnap" dengan nilai 1212 berada di antara sedang dan tinggi. Hasil ini menunjukkan bahwa, dibandingkan dengan AI konvensional seperti FSM, respons *play-tester* terhadap *game* secara keseluruhan dan adanya AI *Layered Animation Trigger* adalah relatif baik atau positif. Meskipun tidak mencapai kategori "Sangat Tinggi", respons *play-tester* tetap positif.

BAB V
KESIMPULAN DAN SARAN

Bab ini berisi kesimpulan dan saran yang didapatkan berdasarkan hasil penelitian. Adapun kesimpulan yang didapatkan merupakan hasil implementasi perilaku *Enemy AI (Artificial Intelligence) Behaviour* menggunakan *Layered Animation Trigger*, beserta hasil analsis respons (*usability test*). Adapun saran yang didapatkan merupakan pertimbangan dan harapan yang diharapkan mampu memberikan perubahan yang baik dan bersifat positif.

5.1 Kesimpulan

Berdasarkan respons sampel dan hasil analisis yang telah dilakukan. Terdapat beberapa kesimpulan yang bisa ditarik, antara lain:

1) **Kesimpulan Implementasi perilaku *enemy* AI dengan *Layered Animation Trigger***

   Implementasi sistem perilaku *enemy* yang didasari pada konsep *layered animation trigger* dengan menggunakan algoritma A* dan basis *Euclidean Distance* (metrik Pythagoras) telah berhasil diimplementasikan. Untuk setiap atribut *input action* akan dipetakan ke dalam beberapa lapisan pemicu (*triggered layer*) yang diharapkan mampu dilakukan oleh *Enemy AI Behaviour (penculik)* sesuai dengan parameter yang berhubungan dengan *action* dan *state* dari penculik.

   Metode yang ditawarkan oleh *Layered Animation Trigger* pada instalasi lingkungan virtual juga telah terbukti efektif mampu menghadirkan dan mengimplementasikan perilaku berdasarkan jarak dan tipe *input* (masukan) dari pemain, sehingga dapat dikombinasikan untuk melakukan lingkungan simulasi yang lebih nyata dalam konteks edukasi. Pendekatan ini memungkinkan penyesuaian perilaku *enemy* (penculik) dengan variabel parameter pada jarak tertentu, seperti *Outter Range*, *Middle Range*, dan *Base Range*. Sehingga, pendekatan ini dapat menciptakan respons yang lebih dinamis terhadap aksi *enemy* (penculik).

2) Kesimpulan Analisis Respons *Usability Test* (*Sampling*)

Berdasarkan hasil analisis dapat disimpulkan bahwa pengalaman bermain dan reaksi *play-tester* terhadap perilaku *enemy* (penculik) yang didasari pada *layered animation trigger* dalam *game* "*NoKidnap*" dinilai positif. Hasil ini dapat diartikan bahwa respons *play-tester* terhadap *game* secara keseluruhan dan adanya AI berbasis sistem tersebut lebih positif dibandingkan dengan AI konvensional seperti FSM. Meskipun tidak mencapai kategori "Sangat Tinggi," hasil ini menandakan bahwa implementasi AI berbasis *layered animation trigger* memberikan dimensi tambahan pada pengalaman bermain, menciptakan reaksi yang lebih mendalam dan responsif dari *play-tester* terhadap perilaku *enemy*. Terutama, terlihat bahwa interaksi dengan musuh (penculik) cenderung lebih memikat bagi para pemain.

Hasil positif ini juga memberikan pijakan yang kuat untuk memperluas penggunaan AI berbasis sistem dalam *aplikasi game* dan berpotensi memberikan dorongan untuk pengembangan lebih lanjut seperti peningkatan respons dan animasi musuh (*enemy)* serta mengembangkan fitur adaptif karakter (*enemy*) dalam menanggapi situasi permainan yang lebih kompleks.

Dengan demikian, hasil analisis menunjukkan bahwa integrasi AI berbasis *layered animation trigger* telah memberikan kontribusi yang signifikan terhadap kualitas pengalaman bermain dan reaksi *play-tester*, memberikan dasar yang solid untuk pengembangan lebih lanjut dan penerapan yang lebih luas dalam industri *aplikasi game*.

5.2 Saran

Adapun saran yang peneliti bisa berikan berdasarkan hasil penelitian ini:

1) Peneliti perlu meningkatkan lagi kualitas aplikasi permainan *(game)* dari berbagai aspek seperti dialog, *pathfinding*, audio, vfx (*virtual effect*), cerita (*story*), dan komponen lainnya di dalam permainan agar mampu memberikan pengalaman bermain yang lebih maksimal.
2) Peneliti perlu menambah kesulitan dari masing-masing musuh secara prosedural agar mampu menghadirkan simulasi pengalaman bermain yang lebih menantang.

3) Peneliti juga disarankan agar mampu melakukan analisis mendetail terkait kemampuan (*ability*) apa saja yang diperlakukan bagi karakter non-pemain (*non-playable character*) ataupun pemain (*playable character*), terutama dalam konteks edukasi. Analisis teknis secara mendalam dapat menghasilkan pengalaman bermain yang lebih komprehensif. Selain itu penambahan pencapaian pada peningkatan status pemain dapat menimbulkan pengalaman bermain di dalam aplikasi yang lebih baik.

DAFTAR PUSTAKA

Anagnostopoulos, I., Vakali, A. and Hadjiefthymiades, S. (2018) 'Artificial intelligence in education: A survey.', IEEE Transactions on Learning Technologies, 11(4), pp. 457–470.

Boyle, E. A., Connolly, T. M., Hainey, T., & Baxter, G. (2011). Designing and evaluating educational *game*s: Case studies in Scotland and Ireland. *Computers & Education*, *57*(2), 1509-1523.

Buckland, M. (2005). *Programming Game AI by Example*. Jones & Bartlett Learning.

Enholm, I. M., Papagiannidis, E., Mikalef, P., & Krogstie, J. (2021). Artificial Intelligence and Business Value: a Literature Review. *Information System Frontier*, 1709–1734.

Esteva, A. et al. (2017) 'Dermatologist-level classification of skin cancer with deep neural networks.', Nature, 542(7639), pp. 115–118.

Fujita, H., & Wu, I.-C. (2012). A special issue on artificial intelligence in computer games: AICG. *Knowledge Based Systems*, *34*,1–2. https://doi.org/10.1016/j.knosys.2012.05.014.

Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetic*s, 4(2), 100-107.

Janosov, M. and Szabo, L. (2019) 'Artificial intelligence in finance: An overview', Acta Polytechnica Hungarica, 16(3), pp. 41–59.

Julien, M. (2014). *Unity AI Programming Essentials*. Packt Publishing.

Knight, W. (2015) Baidu's Deep-Learning System Rivals People at Speech Recognition, MIT Technology Review. Available at: https://www.technologyreview.com/2015/12/16/9846/baidus-deeplearning-system-rivals-people-at-speech-recognition/ (Accessed: 1 June 2023).

Matić, D., & Grčić, M. (2019). *Comparison of Unity3D and Unreal Engine game development frameworks*. In 2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO) (pp. 256-261). IEEE.

Miller, D. P., & Robertson, J. L. (2010). Using serious *game*s to develop cultural awareness. *Computers in Human Behavior, 26*(6), 1378-1387.

Millington, I., & Funge, J. (2009). *Artificial Intelligence for Games* (2nd ed.). CRC Press.

Prensky, M. (2001). Digital *game*-based learning. *Computers in Entertainment (CIE), 1*(1), 21-21.

Raharjo, H. K., F, A. N., Rosyidin, N. F., Purboyo, T. W., & Nugrahaeni, R. A. (2022). Application Of Artificial Intelligence In Indonesian Debate Education Game. [CEPAT] *Journal of Computer Engineering: Progress, Application and Technology, 1*(03), 37. https://doi.org/10.25124/cepat.v1i03.5393

Russell, S., & Norvig, P. (2020). *Artificial Intelligence: A Modern Approach* (4th ed.). Pearson.

Shannon, C. E. (1950). Programming a Computer for Playing Chess. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, *41*(314), 256–275.

Unity Technologies. (2021). *Unity - Manual: NavMesh Agent*.

Weber, J., & Mateas, M. (2010). *Applying the A* algorithm to game AI. In AI Game Programming Wisdom 4* (pp. 269-274). Charles River Media.

Yannakakis, G. N., & Togelius, J. (2015). A panorama of artificial and computational intelligence in games. *IEEE Transactions on Computational Intelligence and AI in Games*, *7*(4), 317–335.

Yannakakis, G. N. & Togelius, J. (2018). Artificial intelligence and games. Berlin: Springer. Retrieved from http://gameaibook.org/. Accessed 12 July 2019.

LAMPIRAN

DAFTAR RIWAYAT HIDUP

CURRICULUM VITAE

Lampiran A-1 DAFTAR RIWAYAT HIDUP

I. DATA PRIBADI / *Personal Identification*

Nama Lengkap	:	Petrus Marcelino H. Tampubolon
Tempat/ Tgl. Lahir	:	Palembang/ 4 Agustus 2001
Jenis Kelamin	:	Laki-Laki
Agama	:	Katolik
Kebangsaan	:	Indonesia
Alamat	:	SOSOR MANGAMBIT SIBULELE
Telepon	:	082213970602
Email	:	petrus.marcellino15@gmail.com

II. KESEHATAN / *Health*

Tidak memiliki cacat mental maupun fisik dan tidak memiliki penyakit bawaan.

III. KEMAMPUAN / *Capabilities*

Bahasa	: Bahasa Indonesia, Bahasa Inggris
Bahasa Pemrograman	: C, C++, Pascal, Java, C#.
Database	: MySQL, Laragon.
Lainnya	: Unity, Microsoft Office, Adobe Photoshop, Paint, Microsoft Excel, Adobe Flash, Adobe Animate, Figma.

IV. PENDIDIKAN FORMAL / *Formal Education*

- [2020 – 2024] S1 Ilmu Komputer, Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara
- [2016 – 2019] SMAN2 Balige
- [2013 – 2016] SMP Budhi Dharma
- [2012 – 2013] SD Negeri 1 Sumberejo

V. PENDIDIKAN NON-FORMAL / *Informal Education*

- Skilvul Learning Track Game Development, DKI Jakarta

VI. PENELITIAN / *Research*

- *Generative AI* dalam Pengembangan *Game* Kartu

VII. PRESTASI / *Achievements*

1) Mahasiswa Berprestasi III Fasilkom-TI Universitas Sumatera Utara
2) Finalist Gemastik XVI Divisi Pengembangan Aplikasi Permainan Universitas Brawjiaya 2023
3) Program Kreativitas Mahasiswa (PKM-KC) Pendanaan 2023

VIII. PUBLIKASI / *Publications*

1) Culture Quest: A Post Mortem Analysis

IX. PENGALAMAN ORGANISASI / *Organizational Experiences*

1) Kepala Divisi Laboratorium Ilmu Komputer Laboratory Center (IKLC) [2021-2022]

X. PENGALAMAN KEPATINIAAN / *Committee Experiences*

1) Anggota Acara PMB Ilmu Komputer USU [2023]
2) Anggota Keamanan ILGANI [2022]
3) MC (Master of Ceremony) Bukber IKLC [2022]

XI. PENGALAMAN BEKERJA / *Working Experience*

- Asisten Laboratorium di IKLC (Ilmu Komputer Laboratory Center) [2022-2024]
- Asisten Dosen di LCVM USU (Laboratorium Computer Vision Universitas Sumatera Utara) Kota Medan [2022-2023]
- *Game Development* Mentor di IMILKOM Bootcamp 2024 [2024]

Lampiran B-1 KUESIONER EVALUASI USABILITY TESTING TERKAIT PERBANDINGAN KECERDASAN BUATAN PERILAKU MUSUH (ENEMY BEHAVIOUR AI) DALAM GAME "NOKIDNAP"

No	Pertanyaan	1	2	3	4	5
1	**Pengambilan Gameplay Secara Menyeluruh**					
a.	Seberapa mendalam anda merasa terdapat perubahan dalam *game*? (1 = Tidak mendalam, 5 = Sangat mendalam)					
b.	Sejauh apa anda merasa bahwa perubahan AI dalam *game* menambah pengalaman *gameplay* ini menarik? (1 = Tidak menarik, 5 = Sangat menarik)					
2	**Pertemuan dengan Penculik**					
a.	Seberapa berkesan pertemuan Anda dengan penculik di dalam *game*? (1 = Tidak berkesan, 5 = Sangat berkesan)					
b.	Bisakah Anda melihat perbedaan taktik atau perilaku yang jelas antara tipe musuh baru dan lama? (1 = Tidak ada perbedaan, 5 = Perbedaan yang sangat jelas)					
3	**Pertanyaan Spesifik dengan *Layered Animation Trigger***					
a.	Seberapa jelas Anda merasakan adanya *gesture* atau perilaku dari musuh selama bermain *game*? (1 = Tidak jelas, 5 = Sangat jelas)					
b.	Sejauh apa Anda merasakan bahwa perilaku penculik dipengaruhi oleh pergerakan pemain? (1 = Tidak dipengaruhi, 5 = Sangat dipengaruhi)					
4	**Perbandingan Jenis *State* (Optimalitas)**					
a.	Seberapa jelas perbedaan perilaku antara kedua jenis penculik yang Anda temui? (1 = Tidak jelas, 5 = Sangat jelas)					
b.	Sejauh mana Anda mampu membedakan perilaku antara jenis penculik tersebut? (1 = Tidak dapat dibedakan, 5 = Sangat dapat dibedakan)					
5	**Pengalaman Uji Buta**					
a.	Seberapa besar harapan Anda akan perbedaan perilaku penculik sebelum uji buta? (1 = Harapan rendah, 5 = Harapan tinggi)					
b.	Sejauh mana Anda terkejut dengan adanya fitur yang tidak terduga selama uji buta? (1 = Tidak terkejut, 5 = Sangat terkejut)					

Lampiran B-2 TABULASI JAWABAN RESPONDEN

Responden	P1a	P1b	P2a	P2b	P3a	P3b	P4a	P4b	P5a	P5b
1	5	4	4	4	4	4	4	4	4	4
2	5	5	4	5	5	5	5	4	5	5
3	4	4	3	4	5	3	4	5	4	3
4	2	2	2	1	5	3	3	2	5	1
5	5	5	5	5	4	5	5	4	4	5
6	2	4	4	4	4	4	4	4	4	4
7	1	3	2	3	3	2	3	3	3	2
8	4	5	4	3	3	5	3	3	3	4
9	5	4	4	3	2	1	5	3	5	5
10	3	5	5	5	4	5	5	5	5	5
11	5	5	3	5	3	3	4	5	3	5
12	5	3	5	3	5	4	4	3	5	3
13	5	5	3	5	5	3	4	4	5	5
14	5	5	4	5	5	5	5	5	5	5
15	4	5	3	5	5	5	5	5	5	5
16	1	2	4	5	4	5	5	5	5	5
17	5	5	5	5	5	5	5	5	5	5

18	5	4	5	4	4	5	4	5	4	4
19	5	5	5	5	5	5	5	5	5	5
20	5	5	5	5	5	5	5	5	5	5
21	2	3	2	5	5	5	5	5	5	5
22	3	3	3	4	3	3	4	3	3	4
23	5	5	5	5	5	5	5	5	4	5
24	5	5	5	5	5	5	5	5	5	5
25	1	2	3	3	3	4	4	4	4	5
26	5	5	4	5	4	5	4	4	4	5
27	5	5	5	5	5	5	5	5	5	5
28	5	5	5	5	5	5	5	5	5	5
29	5	5	4	5	5	5	5	5	5	5
30	2	2	3	3	3	4	3	3	3	3
31	3	4	3	4	4	5	5	5	4	3