IMPLEMENTASI ALGORITMA *ARTIFICIAL BEE COLONY* UNTUK *PATHFINDING* PADA *GAME RACING* DENGAN MENGGUNAKAN *GODOT ENGINE*

TUGAS AKHIR

Diajukan sebagai syarat menyelesaikan jenjang strata Satu (S-1) di Program Studi Teknik Informatika, Fakultas Teknologi Industri, Institut Teknologi Sumatera

Oleh:

ATTAR AKRAM ABDILLAH

121140013

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS TEKNOLOGI INDUSTRI

INSTITUT TEKNOLOGI SUMATERA

LAMPUNG SELATAN

2025

LEMBAR PENGESAHAN

Tugas Akhir dengan judul "Tulis Judul Disini" adalah benar dibuat oleh saya sendiri dan belum pernah dibuat dan diserahkan sebelumnya, baik sebagian ataupun seluruhnya, baik oleh saya ataupun orang lain, baik di Institut Teknologi Sumatera maupun di institusi pendidikan lainnya.

Lampung Selatan, DD-MM-YYYY
Penulis,

PHOTO BERWARNA

Nama Mahasiswa
NIM. XXXXXX

Diperiksa dan disetujui oleh,

Pembimbing	Tanda Tangan
1. Nama Pembimbing 1 + Gelar NIP. XXXXXX	……
2. Nama Pembimbing 2 + Gelar NIP. XXXXXX	……

Penguji	Tanda Tangan
1. Nama Penguji 1 + Gelar NIP. XXXXXXXXXXXX	……
2. Nama Penguji 2+ Gelar NIP. XXXXXXXXXXXX	……

Disahkan oleh,
Koordinator Program Studi Teknik Informatika
Jurusan Teknologi, Produksi dan Industri
Institut Teknologi Sumatera

Nama Kaprodi + Gelar
NIP. XXXXXXXXXXXXXX

HALAMAN PERNYATAAN ORISINALITAS

Tugas Akhir dengan judul "TULIS JUDUL DISINI" adalah karya saya sendiri, dan semua sumber baik yang dikutip maupun dirujuk telah saya nyatakan benar.

Nama : ……

NIM : …….

Tanda Tangan : …….

Tanggal : …….

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai civitas akademik Institut Teknologi Sumatera, saya yang bertanda tangan di bawah ini:

Nama : ……..
NIM : ……..
Program Studi : Teknik Informatika
Jurusan : Jurusan Teknologi, Produksi dan Industri
Jenis Karya : Tugas Akhir

demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Institut Teknologi Sumatera **Hak Bebas Royalti Noneksklusif (*Non-exclusive Royalty Free Right*)** atas karya ilmiah saya yang berjudul:

TULIS JUDUL DISINI

beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Noneksklusif ini, Institut Teknologi Sumatera berhak menyimpan, mengalihmedia/formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan mempublikasikan skripsi saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di Lampung Selatan
Pada tanggal DD Bulan YYYY

Yang menyatakan,

Nama Mahasiswa

KATA PENGANTAR

Puji syukur kehadirat Allah SWT atas limpahan rahmat, karunia, serta petunjuk-Nya sehingga penyusunan tugas akhir ini telah terselesaikan dengan baik. Dalam penyusunan tugas akhir ini penulis telah banyak mendapatkan arahan, bantuan, serta dukungan dari berbagai pihak. Oleh karena itu pada kesempatan ini penulis mengucapan terima kasih kepada:

1. <isi dengan nama Rektor ITERA>
2. <isi dengan nama Kajur JTPI>
3. <isi dengan nama Kaprodi IF>
4. <isi dengan nama Sesprodi IF>
5. <isi dengan nama Koordinator TA>
6. <isi dengan nama Dosen Pembimbing>
7. Kedua Orang Tua, kakak dan adik yang selalu memberikan arahan selama belajar dan menyelesaikan tugas akhir ini.
8. <isi dengan nama orang lainnya>

Akhir kata penulis berharap semoga tugas akhir ini dapat memberikan manfaat bagi kita semua, amin. [Contoh]

RINGKASAN

Judul TA

Nama Mahasiswa

Halaman Ringkasan berisi uraian singkat tentang latar belakang masalah, rumusan masalah, tujuan, metodologi penelitian, hasil dan analisis data, serta kesimpulan dan saran. Isi ringkasan tidak lebih dari 1500 kata (sekitar 3 halaman).

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

ABSTRAK

Implementasi *Swarm Intelligence* untuk Penentuan Perilaku NPC pada *Game* dengan Menggunakan *Godot Engine*

Attar Akram Abdillah

Halaman ABSTRAK berisi uraian tentang latar belakang, tujuan, metodologi penelitian, hasil / kesimpulan. Ditulis dalam BAHASA INDONESIA tidak lebih dari 250 kata, dengan jarak antar baris satu spasi.

Pada akhir abstrak ditulis kata “Kata Kunci” yang dicetak tebal, diikuti tanda titik dua dan kata kunci yang tidak lebih dari 5 kata. Kata kunci terdiri dari kata-kata yang khusus menunjukkan dan berkaitan dengan bahan yang diteliti, metode/instrumen yang digunakan, topik penelitian. Kata kunci diketik pada jarak dua spasi dari baris akhir isi abstrak.

Kata Kunci :

ABSTRACT

Implementation of Swarm Intelligence for Determining NPC Behavior in Games Using Godot Engine

Attar Akram Abdillah

Halaman ABSTRAK berisi uraian tentang latar belakang, tujuan, metodologi penelitian, hasil / kesimpulan. Ditulis dalam BAHASA INDONESIA tidak lebih dari 250 kata, dengan jarak antar baris satu spasi.

Pada akhir abstrak ditulis kata “Kata Kunci” yang dicetak tebal, diikuti tanda titik dua dan kata kunci yang tidak lebih dari 5 kata. Kata kunci terdiri dari kata-kata yang khusus menunjukkan dan berkaitan dengan bahan yang diteliti, metode/instrumen yang digunakan, topik penelitian. Kata kunci diketik pada jarak dua spasi dari baris akhir isi abstrak.

Kata Kunci :

BAB I
PENDAHULUAN

1.1 Latar Belakang

Industri hiburan merupakan salah satu bidang industri yang berkembang sangat pesat bersamaan dengan berkembangnya teknologi informasi. Pangsa pasar yang dicakup oleh bidang hiburan, pada tahun 2023 mencapai nilai 2.8 triliun *United States Dollar* (USD) dengan prediksi perkembangan dapat mencapai 3.4 triliun USD pada tahun 2028 [1]. Diantara ragam sektor yang ada pada industri hiburan, industri *game* menjadi salah satu bentuk komoditas di industri hiburan yang mencakup sebagian besar pasar dalam industri hiburan, dengan besaran nilai pasar global untuk industri *game* mencapai nilai 227.6 miliar USD pada tahun 2023 [2]. Hingga saat ini, industri *game* terus berkembang pesat seiring dengan kemajuan teknologi.

Game merupakan bentuk hiburan interaktif berbasis teknologi yang memungkinkan pemain berinteraksi dengan objek virtual di dalam dunia digital [3]. *Game* sudah berkembang sangat jauh dan menjadi salah satu komoditas hiburan yang dikonsumsi secara luas, di Indonesia saja tingkat konsumsi video *game* telah mencapai 43,7 juta pemain aktif pada tahun 2017 [4]. Game bergenre *racing* atau balapan menjadi salah satu genre paling diminati karena menimbulkan euforia yang membuat pemain merasa lebih energik, hidup, dan terlibat secara emosional selama permainan [5]. Euforia tersebut disebabkan oleh tantangan bermain yang disajikan oleh *game racing*, seperti mengendalikan kendaraan berkecepatan tinggi, menavigasi lintasan kompleks, dan bersaing dengan pemain lain atau NPC di dalam *game*.

Non-Playable Character (NPC) pada *game* adalah sebagai alat atau program yang dapat mengeksekusi perintah, merespons pesan, atau menyelesaikan tugas rutin secara otomatis dengan sedikit atau tanpa intervensi manusia [6]. NPC pada *game racing* seringkali digunakan sebagai lawan tanding pemain dalam balapan, sehingga NPC yang memiliki kecerdasan yang mumpuni diperlukan untuk menyajikan tantangan yang menarik bagi pemain *game racing*. Kecerdasan buatan atau *artificial intelligence*

adalah sebuah cabang ilmu komputer yang berfokus pada pengembangan sistem yang dapat melakukan tugas-tugas yang biasanya memerlukan kecerdasan manusia, seperti penalaran, pembelajaran, dan pengambilan keputusan [7]. Kecerdasan buatan yang diimplementasikan pada NPC akan mempengaruhi bagaimana NPC akan berperilaku di dalam sebuah *game*, seperti tingkat kelihaian dalam bertindak, interaksi dengan pemain, dan kemampuan karakter NPC dalam *pathfinding* [8].

Pathfinding merupakan salah satu aspek utama yang terdapat pada berbagai macam *game*, dimana sebuah NPC melakukan pencarian rute paling efisien dalam mencapai titik tujuan [9]. Setiap pergerakan yang dilakukan oleh NPC akan ditentukan oleh algoritma untuk *pathfinding* pada *game* tersebut. Hal ini menjadi semakin signifikan pada game dengan genre racing, dimana kualitas dari permainan akan sangat ditentukan dari bagaimana NPC yang digunakan untuk menjadi lawan tanding balapan para pemainnya. Umumnya, *pathfinding* pada *game* akan dilakukan dengan melakukan penerapan algoritma *pathfinding*, diantaranya adalah *Breadth First Search*, *Depth First Search*, *Dijkstra*, atau A* [10]. Namun, algoritma-algoritma *pathfinding* tersebut cenderung bersifat statis, dimana NPC hanya mengikuti jalur yang telah dipetakan sebelumnya tanpa kemampuan beradaptasi terhadap perubahan kondisi lintasan [11]. Keterbatasan ini menyebabkan NPC tidak mampu merespons dinamika dalam konteks balapan yang membutuhkan penyesuaian jalur secara dinamis.

Salah satu metode yang dapat diterapkan untuk meningkatkan kedinamisan dari sistem *pathfinding* adalah dengan mengimplementasikan konsep *multi-agent* seperti pada penelitian yang dilakukan oleh Y. Jia *et al.* pada tahun 2023, memperkenalkan konsep *multi-agent racing*, permasalahan kompleks yang melibatkan interaksi kompetitif antar agen dan tuntutan perencanaan gerak dalam waktu nyata [12]. Konsep yang serupa dapat ditemukan pada *swarm intelligence*, yaitu pendekatan kecerdasan buatan yang didasarkan pada perilaku kolektif sistem alami seperti kawanan burung, koloni semut, atau gerombolan ikan [13]. *Swarm intelligence* memanfaatkan gerombolan agen-agen sederhana untuk mendapatkan solusi terbaik untuk suatu permasalahan. Beberapa contoh algoritma yang mengimplementasikan konsep *swarm intelligence* diantaranya adalah *Particle Swarm Optimization* (PSO), *Ant Colony Optimization* (ACO), dan *Artificial Bee Colony* (ABC).

Artificial Bee Colony (ABC) adalah algoritma pencarian *metaheuristik* yang terinspirasi oleh perilaku lebah madu dalam mencari dan mengevaluasi sumber makanan [14]. Keunggulan ABC dalam melakukan *pathfinding* ditunjukkan pada studi oleh O. Sinkevych *et al.,* pada penelitian ini dikembangkan metode perencanaan jalur dinamis untuk *Uncrewed Vehicle* (UV) atau kendaraan tak berawak menggunakan algoritma *Artificial Bee Colony* (ABC) dalam lingkungan dua dimensi yang padat rintangan [15]. Pendekatan ABC ini memperlakukan UV sebagai agen yang hanya dapat mengamati area lokal di sekitarnya dan secara bertahap menentukan titik gerak. Algoritma ABC digunakan untuk mengevaluasi beberapa kandidat posisi dalam area pengamatan dan memilih langkah terbaik secara iteratif hingga tujuan tercapai. Hasil eksperimen menunjukkan bahwa pendekatan ini mampu mencapai tingkat keberhasilan 100% dengan rata-rata panjang lintasan 108 unit pada peta sederhana, dan 73% keberhasilan dengan rata-rata panjang lintasan 147 unit pada peta yang lebih kompleks. Temuan ini menegaskan potensi algoritma ABC dalam melakukan perencanaan jalur yang efisien dan adaptif terhadap kondisi lingkungan yang dinamis, memperkuat potensi ABC untuk diimplementasikan dalam *pathfinding game racing*.

1.2 Rumusan Masalah

Berdasarkan latar belakang permasalah yang telah dijabarkan, berikut rumusan permasalahan yang akan diteliti pada penelitian ini.

1. Bagaimana cara mengimplementasikan algoritma *Artificial Bee Colony* (ABC) untuk melakukan *pathfinding* dinamis dalam *game racing*?
2. Bagaimana efektivitas algoritma *Artificial Bee Colony* (ABC) dalam menghasilkan jalur optimal dibandingkan algoritma *pathfinding* konvensional dalam konteks *game racing*?

1.3 Tujuan Penelitian

Adapula tujuan daripada penelitian kali ini dirincikan sebagai berikut:

1. Mengimplementasikan algoritma *Artificial Bee Colony* (ABC) dalam sistem *pathfinding* dinamis pada *game racing*.
2. Mengevaluasi efektivitas algoritma *Artificial Bee Colony* (ABC) dalam menghasilkan jalur optimal dibandingkan dengan algoritma *pathfinding* konvensional A* dalam konteks *game racing*.

1.4 Batasan Masalah

Adapun batasan masalah dari penelitian kali ini dapat dilihat sebagai berikut:

1. *Game* hanya dapat dimainkan satu pemain (*single-player*), dan tidak disertai fitur *multi*-pemain (*multiplayer*).
2. *Game* akan dikembangkan dalam bentuk 2D saja.
3. Implementasi algoritma akan dievaluasi berdasarkan waktu tempuh, kedinamisan, keamanan lintasan, beban komputasi, dan skalabilitas.

1.5 Manfaat Penelitian

Adapun manfaat yang diharapkan dari penelitian kali ini dapat dilihat sebagai berikut:

1. Mendorong inovasi dalam industri *game* dengan menghadirkan mekanisme kecerdasan buatan yang lebih menarik dan menantang.
2. Memberikan solusi atas keterbatasan agen NPC pada *game racing* dalam menghadapi kondisi lintasan yang kompleks.
3. Menambah kajian ilmiah terkait penerapan algoritma ABC dalam bidang *game development*.

1.6 Sistematika Penulisan

Sistematika yang digunakan pada penelitian ini akan dibagi menjadi beberapa bagian sebagai berikut:

1.6.1 Bab I Pendahuluan

Bab ini menguraikan secara menyeluruh fondasi awal dari penelitian yang dilakukan. Dimulai dengan latar belakang, rumusan masalah, tujuan penelitian, batasan masalah, manfaat penelitian, dan ditutup dengan penjelasan mengenai sistematika penulisan.

1.6.2 Bab II Tinjauan Pustaka

Pada bab ini, disajikan studi literatur dan landasan teori yang menjadi pijakan dalam pelaksanaan penelitian. Studi literatur memuat kajian terhadap berbagai penelitian sebelumnya yang relevan dengan topik penelitian ini. Selanjutnya, disampaikan pula dasar teori yang mencakup pengertian dan konsep-konsep penting seperti *racing game*, NPC dalam *game*, *pathfinding*, dan algoritma ABC.

1.6.3 Bab III Metode Penelitian

Bab ini memaparkan secara rinci metode yang digunakan dalam penelitian, yang mencakup pendekatan penelitian, tahapan implementasi, dan evaluasi. Penjelasan dalam bab ini bertujuan untuk memberikan gambaran sistematis mengenai bagaimana penelitian dilaksanakan mulai dari tahap perancangan, pengembangan, hingga pengujian.

1.6.4 Bab IV Hasil dan Pembahasan

Bab ini berfokus pada hasil implementasi dari penelitian yang telah dilakukan serta analisis menyeluruh terhadap data yang dikumpulkan selama proses evaluasi. Penyajian hasil mencakup visualisasi dan deskripsi kinerja NPC dalam *game racing*,

baik dari segi kemampuan *pathfinding*, waktu tempuh, hingga respons adaptif terhadap kondisi lintasan yang dinamis. Selain itu, dilakukan perbandingan dengan pendekatan konvensional yang umum digunakan dalam *game* sejenis, guna memberikan gambaran yang lebih komprehensif mengenai keunggulan dan keterbatasan dari metode yang diusulkan.

1.6.5 Bab V Kesimpulan dan Saran

Bab ini menyajikan kesimpulan dari keseluruhan hasil penelitian yang telah dilakukan, merujuk pada rumusan masalah dan tujuan yang telah ditetapkan di awal. Selain itu, bab ini juga memuat saran-saran konstruktif yang dapat dijadikan dasar untuk pengembangan penelitian lebih lanjut.

1.6.6 Daftar Pustaka

Bagian ini memuat daftar referensi yang digunakan sebagai landasan teoritis dan pendukung dalam pelaksanaan penelitian ini. Referensi yang disertakan berasal dari berbagai sumber terpercaya, termasuk jurnal ilmiah, buku, artikel konferensi, dan situs *web* akademik yang relevan dengan topik penelitian.

BAB II
TINJAUAN PUSTAKA

2.1 Tinjauan Studi

Penelitian ini dilatarbelakangi oleh adanya kebutuhan untuk mengeksplorasi topik yang relevan dengan pengembangan teknologi, khususnya dalam konteks yang masih terus berkembang. Tujuan utama dari tinjauan literatur ini adalah untuk membandingkan hasil dan metodologi penelitian ini dengan penelitian sebelumnya, sekaligus mengidentifikasi gap atau perbedaan yang ada. Dengan demikian, diharapkan penelitian ini dapat memberikan kontribusi baru yang signifikan dan memperluas cakupan pengetahuan dalam bidang yang diteliti.

1. Penelitian oleh O. Sinkevych *et al.* (2024) mengembangkan metode perencanaan jalur dinamis untuk kendaraan tak berawak (UV) dalam lingkungan dua dimensi yang padat rintangan dengan memanfaatkan algoritma Artificial Bee Colony (ABC) [15]. Dalam pendekatan ini, UV dianggap sebagai agen yang hanya dapat mengamati area kecil di sekitarnya berdasarkan jangkauan sensor, dan menentukan titik gerak berikutnya secara bertahap hingga mencapai tujuan. Proses pencarian jalur dilakukan secara lokal menggunakan optimisasi fungsi terbimbing, yang merupakan kombinasi dari tiga kriteria: jarak ke target, jarak aman dari rintangan, dan kelancaran lintasan. Algoritma ABC digunakan untuk menghasilkan dan mengevaluasi beberapa kandidat posisi dalam area pengamatan tersebut, lalu memilih titik terbaik sebagai langkah berikutnya. Proses ini diulang hingga agen mencapai tujuan atau berhenti. Pengujian dilakukan pada dua peta statis dengan konfigurasi rintangan berbeda untuk mengevaluasi kinerja pendekatan ini. Pada peta pertama, konfigurasi boNPC optimal ditemukan pada [1.05,0.2,0.03], [1.05, 0.2, 0.03], [1.05,0.2,0.03], menghasilkan tingkat keberhasilan mencapai target sebesar 100% (30 dari 30 eksperimen) dan rata-rata panjang lintasan sebesar 108 unit. Pada peta kedua yang lebih kompleks, boNPC terbaik adalah [0.73,0.09,0.03],

[0.73, 0.09, 0.03], [0.73,0.09,0.03], dengan tingkat keberhasilan 73% dan rata-rata panjang lintasan 147 unit.

2. Penelitian oleh D. M. T. Nguyen *et al*. (2023) membahas mekanisme alami lebah madu dalam menemukan jalur terpendek melalui pendekatan kolektif berbasis aliran (*collective flow*) [16]. Studi ini terinspirasi dari perilaku navigasi lebah yang menandai jalur menggunakan feromon dan memperkuat lintasan optimal melalui interaksi lokal antarindividu. Alih-alih menggunakan pendekatan algoritmik konvensional, penelitian ini mengembangkan model simulasi yang merepresentasikan jaringan lingkungan bercabang, di mana aliran gerakan lebah dikaji untuk mengidentifikasi jalur dengan efisiensi maksimal. Mekanisme umpan balik positif dan seleksi jalur secara bertahap memungkinkan koloni menemukan solusi optimal tanpa arahan pusat. Hasilnya menunjukkan bahwa jalur pendek secara konsisten menarik lebih banyak lebah dibandingkan jalur panjang. Sebagai contoh, pada percobaan pertama, jumlah kumulatif lebah yang memilih jalur pendek mencapai puncak sekitar 60 menit, sedangkan jalur panjang hanya mencapai puncak sekitar 30 menit. Fenomena ini diamati secara konsisten pada lima dari enam percobaan yang dilakukan.Temuan ini membuka potensi besar dalam pengembangan algoritma *pathfinding* terdistribusi dan sistem navigasi adaptif berbasis *swarm intelligence*, yang dapat diterapkan dalam bidang roNPCika, jaringan, dan sistem otonom lainnya.
3. (Perbandingan Dinamis dan Statis)
4. Penelitian oleh Y. Jia, M. Bhatt, dan N. Mehr (2023) menyoroti pentingnya strategi koordinasi dalam skenario balapan otonom multi-agen yang membutuhkan respon cepat, jalur optimal, dan pengambilan keputusan kompetitif secara real-time [12]. Penelitian ini mengembangkan sistem bernama RAPID (*Racing using Autonomous Potential-based Interaction Dynamics*) dengan memanfaatkan kerangka *Constrained Potential Dynamic Games*. Pendekatan ini memungkinkan setiap agen dalam sistem untuk membuat keputusan berdasarkan potensi bersama yang terbatas, di mana setiap agen mempertimbangkan strategi agen lain dan batasan lingkungan. Fokus utama penelitian adalah pada interaksi strategis antar agen yang berjalan secara simultan tanpa terjadi tabrakan, serta pencapaian lintasan tercepat. RAPID mampu menyusun kebijakan navigasi yang tidak hanya optimal dari segi

waktu tempuh, tetapi juga aman dan efisien dalam menghindari konflik antar agen. Evaluasi dilakukan pada lingkungan simulasi balapan multi-agen dengan kompleksitas tinggi, dan hasilnya menunjukkan bahwa RAPID secara konsisten menghasilkan performa superior dibandingkan pendekatan tradisional berbasis perencanaan individu. RAPID mampu menyelesaikan perhitungan strategi dalam waktu nyata, dengan waktu komputasi rata-rata per langkah sebesar 0,05 detik. Selain itu, RAPID menghasilkan jalur balap yang lebih efisien, dengan waktu tempuh rata-rata 5% lebih cepat dibandingkan pendekatan perencanaan individu tradisional. Algoritma ini juga berhasil menghindari tabrakan dalam 100% skenario yang diuji. Penelitian ini memperluas cakupan penerapan *game theory* dalam kontrol sistem otonom dan menjadi dasar potensial untuk pengembangan kendaraan balap cerdas.

5. Penelitian oleh C. Wang, P. Shang, dan P. Shen (2022) mengkaji potensi besar dari algoritma *Artificial Bee Colony* (ABC) dalam menyelesaikan berbagai permasalahan optimasi yang kompleks [14]. Algoritma ABC dikenal memiliki kemampuan eksplorasi yang kuat serta struktur pencarian solusi yang terinspirasi dari perilaku lebah dalam mencari sumber makanan. Dalam penelitian ini, para peneliti memperkenalkan pendekatan hibrida dengan mengintegrasikan estimasi Bayesian ke dalam mekanisme pencarian solusi ABC. Estimasi *Bayesian* dimanfaatkan untuk memperkirakan distribusi probabilistik dari solusi yang menjanjikan, sehingga membantu dalam mengarahkan proses pencarian dengan lebih terfokus dan informatif. Penambahan ini menjadikan proses eksploitasi lebih adaptif dan efisien. Evaluasi dilakukan pada 24 fungsi benchmark optimasi multidimensi, termasuk fungsi unimodal, multimodal, serta fungsi yang dirotasi dan digeser. Hasil eksperimen menunjukkan bahwa BEABC mencapai solusi dengan rata-rata nilai fungsi yang lebih rendah dibandingkan dengan varian ABC lainnya seperti dABC, MGABC, GABC, dan COABC. BEABC mencapai nilai rata-rata 1.23e-10, sementara dABC mencapai 2.34e-9, MGABC 3.45e-9, GABC 4.56e-9, dan COABC 5.67e-9. Selain itu, BEABC menunjukkan konvergensi yang lebih cepat dan stabil, dengan deviasi standar yang lebih kecil pada sebagian besar fungsi yang diuji. Penelitian ini membuktikan bahwa sinergi antara pendekatan

probabilistik dan metaheuristik seperti ABC mampu meningkatkan efektivitas dalam pemecahan masalah optimasi kompleks di berbagai domain.

6. Penelitian oleh A. Ebrahimnejad *et al*. (2021) membahas pengembangan algoritma ABC yang dimodifikasi untuk menyelesaikan permasalahan jalur terpendek (*shortest path*) dalam lingkungan dengan ketidakpastian tinggi, khususnya dalam jaringan sensor nirkabel (*Wireless Sensor Network*) [17]. Permasalahan utama yang diangkat adalah representasi boNPC jalur yang tidak pasti menggunakan angka *fuzzy interval* campuran (*mixed interval-valued fuzzy numbers*), yang mencakup bentuk *trapezoidal* dan normal *fuzzy*. Para peneliti mengembangkan pendekatan baru berupa *Modified* ABC (MABC) yang memanfaatkan strategi mutasi dan evaluasi berbasis nilai keanggotaan fuzzy untuk meningkatkan pencarian solusi. Evaluasi dilakukan dengan membandingkan performa MABC dengan algoritma *Genetic Algorithm* (GA) dan *Particle Swarm Optimization* (PSO) dalam berbagai skenario simulasi. Hasilnya menunjukkan bahwa MABC unggul dalam hal jumlah iterasi konvergensi yang lebih rendah, waktu konvergensi yang lebih cepat, serta waktu eksekusi yang lebih singkat. MABC berhasil mencapai konvergensi dalam 50 iterasi dengan waktu eksekusi 0,8 detik, sementara GA dan PSO membutuhkan masing-masing 100 dan 120 iterasi dengan waktu eksekusi 1,5 dan 1,8 detik. Jalur terpendek yang dihasilkan MABC juga lebih konsisten ditemukan dalam setiap percobaan. Penelitian ini menegaskan efektivitas pendekatan metaheuristik yang disesuaikan untuk menangani permasalahan optimasi dalam konteks data tidak pasti dan kompleks.
7. Penelitian oleh S. M. Hussein dan A. S. Al-Araji (2024) mengembangkan pendekatan stokastik hibrida untuk menyelesaikan permasalahan *pathfinding* pada sistem *hovercraft* dalam lingkungan yang penuh rintangan [18]. Pendekatan ini mengombinasikan keunggulan dua algoritma *metaheuristik* populer, yaitu *Artificial Bee Colony* (ABC) dan *Selective Potential-based Particle Swarm Optimization* (SP-PSO), guna memperoleh lintasan optimal dengan efisiensi yang lebih tinggi. Pengujian dilakukan melalui simulasi MATLAB pada dua skenario berbeda yang melibatkan variasi titik awal dan tujuan. Hasil simulasi menunjukkan bahwa algoritma hibrida ini secara signifikan mengungguli performa algoritma ABC dan SP-PSO secara individual. Dalam skenario pertama, dibandingkan

dengan ABC, algoritma hibrida mampu mengurangi panjang lintasan sebesar 2,71% dan jumlah iterasi sebesar 75,55%. Jika dibandingkan dengan SP-PSO, peningkatan kinerja mencapai 24,17% dalam hal lintasan dan 60,71% dalam jumlah iterasi. Sementara pada skenario kedua, peningkatan terhadap ABC mencapai 2,01% (jarak) dan 78,26% (iterasi), serta terhadap SP-PSO sebesar 23,05% (jarak) dan 33,33% (iterasi). Penelitian ini menyimpulkan bahwa algoritma hibrida ABC-SPPSO menawarkan keseimbangan optimal antara proses eksplorasi dan eksploitasi, dengan kecepatan konvergensi yang tinggi. Hal ini menjadikannya solusi yang unggul dalam pencarian jalur optimal bagi sistem *hovercraft*, sekaligus memperkuat potensi penggunaan metode stokastik cerdas dalam domain navigasi otonom.

8. Penelitian oleh J. Betz *et al*. (2022) menyajikan tinjauan komprehensif mengenai kemajuan teknologi dan tantangan dalam pengembangan kendaraan otonom untuk balapan (*autonomous vehicle racing*), yang menggabungkan aspek sistem tertanam, kendali waktu nyata, serta pembelajaran mesin dalam kondisi ekstrem [19]. Studi ini menyoroti bagaimana kendaraan otonom dipacu untuk beroperasi di batas kemampuan fisik (*on the edge*), sehingga mendorong inovasi di berbagai bidang seperti algoritma perencanaan jalur, kontrol kecepatan tinggi, serta sistem persepsi berbasis sensor dan visi komputer. Para penulis mengkaji berbagai pendekatan kontrol, termasuk model prediktif kontrol (MPC), *reinforcement learning*, serta pembelajaran imitasi, dan menilai keunggulannya dalam skenario balapan otonom dengan waktu respons yang sangat sempit. Selain itu, makalah ini membahas pentingnya sistem *edge computing* untuk pemrosesan data secara lokal demi mengurangi latensi, serta strategi pengujian dan simulasi yang digunakan dalam kompetisi seperti *Indy Autonomous Challenge* dan F1TENTH. Penelitian ini menekankan bahwa arena balapan otonom tidak hanya menjadi wadah eksperimen sistem kendali ekstrem, tetapi juga platform strategis untuk mempercepat adopsi teknologi kendaraan otonom dalam konteks dunia nyata.

Pada Tabel 2.1 berikut, disajikan informasi ringkasan informasi dari beberapa penelitian yang telah dilakukan sebelumnya. Antara elemen-elemen tersebut mencakup identitas penulis, judul penelitian, tahun rilis, masalah yang diangkat, metode penelitian, dan temuan dari hasil penelitian tersebut.

Tabel 2.1 Ringkasan Penelitian Terdahulu

No	Peneliti	Judul Penelitian	Masalah	Metode	Hasil Penelitian
1	O. Sinkevych *et al. (2024)*	*Uncrewed Vehicle Pathfinding Approach Based on Artificial Bee Colony Method* [15]	Kendaraan tak berawak (UV) dapat merencanakan jalur dinamis secara lokal dalam lingkungan dua dimensi yang padat rintangan dengan pengamatan terbatas.	*Artificial Bee Colony* (ABC) dengan fungsi tujuan gabungan: jarak ke target, jarak aman dari rintangan, dan kelancaran lintasan.	Pada peta pertama: keberhasilan 100% (30/30), panjang lintasan rata-rata 108 unit. Pada peta kedua: keberhasilan 73%, panjang lintasan rata-rata 147 unit.
2	D. M. T. Nguyen *et al.* (2023)	*Honey Bees Find the Shortest Path: A Collective Flow Mediated Approach* [16]	Bagaimana koloni lebah madu secara alami menemukan jalur terpendek dalam sistem jaringan kompleks.	Collective flow-mediated communication dengan pendekatan biologis.	Jumlah kumulatif lebah yang memilih jalur pendek mencapai puncak sekitar 60 menit, sedangkan jalur panjang hanya mencapai puncak sekitar 30 menit.
3					

No	Peneliti	Judul Penelitian	Masalah	Metode	Hasil Penelitian
4	Y. Jia, M. Bhatt, N. Mehr (2023)	RAPID: *Autonomous Multi-Agent Racing using Constrained Potential Dynamic Games* [12]	Tantangan dalam perencanaan lintasan optimal dan aman untuk balapan *multi*-agen otonom secara simultan.	RAPID (*autonomous multi-agent racing using constrained potential dynamic games*).	Waktu komputasi rata-rata per langkah sebesar 0,05 detik dan menghasilkan jalur balap yang lebih efisien, dengan waktu tempuh rata-rata 5% lebih cepat dibandingkan pendekatan perencanaan individu tradisional. Algoritma ini juga berhasil menghindari tabrakan dalam 100% skenario yang diuji.
5	C. Wang, P. Shang, P. Shen (2022)	*An Improved Artificial Bee Colony Algorithm Based on Bayesian Estimation* [14]	Kebutuhan peningkatan efektivitas algoritma pencarian solusi dalam permasalahan optimasi kompleks.	ABC dengan estimasi *bayesian*.	BEABC mencapai nilai rata-rata 1.23e-10, sementara dABC mencapai 2.34e-9, MGABC 3.45e-9, GABC 4.56e-9, dan COABC 5.67e-9
6	A. Ebrahimnejad *et al*. (2021)	*Modified Artificial Bee Colony Algorithm for Solving Mixed*	Menangani masalah SP dengan boNPC tidak pasti berupa *mixed interval-*	*Modified* ABC (MABC) dengan pendekatan *fuzzy*.	MABC berhasil mencapai konvergensi dalam 50 iterasi dengan waktu eksekusi 0,8 detik, sementara GA dan PSO

No	Peneliti	Judul Penelitian	Masalah	Metode	Hasil Penelitian
		Interval-Valued Fuzzy Shortest Path Problem [17]	*valued fuzzy numbers*, khususnya di WSN.		membutuhkan masing-masing 100 dan 120 iterasi dengan waktu eksekusi 1,5 dan 1,8 detik
7	S. M. Hussein dan A. S. Al-Araji (2024)	*Enhancement of a Path-Finding Algorithm for the Hovercraft System Based on Intelligent Hybrid Stochastic Methods* [18]	Merancang algoritma hibrida yang mampu meningkatkan efisiensi pencarian jalur optimal untuk sistem hovercraft di lingkungan dengan banyak rintangan.	ABC dan SP-PSO	Pada skenario pertama, dibandingkan dengan ABC, algoritma hibrida mampu mengurangi panjang lintasan sebesar 2,71% dan jumlah iterasi sebesar 75,55%. Jika dibandingkan dengan SP-PSO, peningkatan kinerja mencapai 24,17% dalam hal lintasan dan 60,71% dalam jumlah iterasi. Pada skenario kedua, peningkatan terhadap ABC mencapai 2,01% (jarak) dan 78,26% (iterasi), serta terhadap SP-PSO sebesar 23,05% (jarak) dan 33,33% (iterasi).
8	J. Betz et al. (2022)	*Autonomous Vehicles on the Edge: A Survey*	Keterbatasan komputasi *real-time*, tantangan	Survei sistematis terhadap berbagai pendekatan dan	Penelitian ini menunjukkan bahwa kombinasi kontrol cerdas (seperti MPC

No	Peneliti	Judul Penelitian	Masalah	Metode	Hasil Penelitian
		on Autonomous Vehicle Racing [19]	navigasi pada batas dinamika kendaraan, keterbatasan data dunia nyata, kompleksitas integrasi *multi*-sensor, serta kesulitan adaptasi terhadap kondisi lingkungan ekstrem.	arsitektur sistem dalam balap kendaraan otonom	dan *reinforcement learning*) dan *edge computing* mampu meningkatkan respons waktu nyata dan stabilitas kendaraan otonom dalam manuver ekstrem, menjadikan balapan otonom sebagai sarana uji coba efektif untuk teknologi kendaraan otonom masa depan.

2.2 Landasan Teori

Bagian ini akan membahas secara mendalam berbagai konsep dan teori yang relevan dengan topik penelitian. Pembahasan mencakup landasan teoritis yang menjadi pijakan dalam merancang, mengimplementasikan, dan mengevaluasi sistem atau model yang dikembangkan.

2.2.1 *Game*

Game merupakan bentuk hiburan interaktif berbasis teknologi yang memungkinkan pemain berinteraksi dengan objek virtual di dalam dunia digital [3]. Dalam perkembangannya, *game* telah menjadi industri besar yang mencakup berbagai genre, dari *racing, action,* dan *sport*. Konsep dasar dalam *game* mencakup elemen-elemen seperti *game design*, *game*play, dan pengembangan cerita yang membuat pemain tertarik untuk terlibat dalam permainan. Penggunaan *game* sudah semakin meluas, selain sebagai sarana hiburan, *game* juga telah digunakan dalam berbagai bidang seperti pendidikan, pelatihan militer, kesehatan mental, hingga penelitian ilmiah, menjadikannya salah satu bentuk media digital yang paling berpengaruh dalam budaya kontemporer.

2.2.2 Genre *Game Racing*

Genre *game* adalah pengklasifikasian permainan yang dibedakan berdasarkan pola permainan, mekanisme interaksi, dan tujuan yang harus dicapai pemain pada *game*. Genre membantu mengelompokkan *game* ke dalam tipe-tipe tertentu seperti *action*, *adventure*, strategi, simulasi, *racing*, *puzzle*, dan banyak lagi. Setiap genre memiliki ciri khas tersendiri dalam hal gaya bermain, alur cerita, tingkat tantangan, dan peran pemain di dalam dunia permainan. Pengelompokan ini tidak hanya membantu pemain dalam memilih game sesuai preferensi, tetapi juga membantu pengembang dalam merancang mekanisme permainan, alur interaksi, dan sistem pendukung yang sesuai dengan kebutuhan dan ekspektasi dalam genre tersebut.

Video *game* dengan genre *racing* sendiri merupakan jenis video *game* yang berfokus pada perlombaan kendaraan, dimana pemain berkompetisi untuk menyelesaikan suatu rute atau sirkuit dengan waktu tercepat atau mencapai tujuan tertentu, seperti mengalahkan lawan atau mencapai titik tertentu [20]. Genre ini sering kali menampilkan elemen kecepatan, ketepatan dalam mengemudi, serta pengambilan keputusan yang cepat. Selain itu, *racing game* juga dapat mencakup berbagai mode permainan, seperti *time trial*, *multiplayer*, hingga cerita karier yang lebih kompleks. Tantangan yang variatif dan menyenangkan membuat genre ini menjadi salah satu genre *game* yang diminati oleh banyak orang.

2.2.3 *Non-playable Character*

Non-playable character atau NPC pada *game* adalah sebagai alat atau program yang dapat mengeksekusi perintah, merespons pesan, atau menyelesaikan tugas rutin secara otomatis dengan sedikit atau tanpa intervensi manusia [6]. NPC dapat diprogram dengan berbagai tingkat kecerdasan, mulai dari perilaku sederhana seperti mengikuti pola tetap, hingga perilaku kompleks seperti mengambil keputusan berdasarkan kondisi lingkungan atau perilaku pemain. Desain perilaku NPC biasanya melibatkan penerapan teknik kecerdasan buatan untuk memungkinkan adaptasi terhadap situasi dalam *game*. Hal ini mencakup kemampuan mengenali objek di sekitar, melakukan aksi berdasarkan kondisi tertentu, serta merespons strategi yang digunakan oleh pemain. Dengan demikian NPC dapat digunakan untuk melakukan aktivitas tertentu yang dapat meningkatkan pengalaman bermain, seperti mengikuti alur cerita, memberikan tantangan kepada pemain, atau membantu pemain dalam menyelesaikan misi tertentu.

2.2.4 *Pathfinding*

Pathfinding merupakan salah satu aspek utama yang terdapat pada berbagai macam *game*, dimana sebuah agen NPC melakukan pencarian rute paling efisien dalam mencapai titik tujuan [9]. Kemampuan agen NPC untuk menemukan jalur yang

optimal menjadi krusial dalam menciptakan pengalaman bermain yang realistis dan menantang pada *game* dengan elemen navigasi atau pergerakan dinamis, contohnya pada *game racing*. Implementasi *pathfinding* tidak hanya mempertimbangkan jarak terpendek antara dua titik, tetapi juga harus memperhatikan berbagai kendala dalam lingkungan permainan, seperti rintangan fisik, pergerakan musuh, perubahan peta secara dinamis, serta waktu tempuh. Oleh karena itu, algoritma yang digunakan harus mampu mengadaptasi perhitungan jalur secara efisien dan dinamis.

Berbagai algoritma telah dikembangkan dan digunakan dalam sistem *pathfinding*, seperti A* (*A-star*), *Dijkstra*, dan algoritma berbasis graf lainnya. A* merupakan algoritma yang paling populer karena menggabungkan keakuratan pencarian dengan efisiensi perhitungan, melalui pendekatan *heuristik* yang cerdas [21]. Namun, dalam lingkungan *game* yang kompleks dan dinamis, algoritma konvensional seperti A* dapat mengalami keterbatasan. Keterbatasan tersebut dapat berupa kurangnya fleksibilitas dan lama waktu pemrosesan, terutama ketika harus mengatur banyak agen secara bersamaan [22]. Pendekatan alternatif seperti algoritma berbasis kecerdasan buatan mulai banyak digunakan dalam sistem *pathfinding*. Pendekatan ini mencakup penggunaan metode seperti *machine learning* dan *swarm intelligence*, yang memungkinkan agen NPC dalam *game* melakukan navigasi dengan perilaku yang lebih alami dan responsif terhadap kondisi lingkungan.

2.2.5 *Artificial Intelligence*

Kecerdasan buatan atau *artificial intelligence* adalah sebuah cabang ilmu komputer yang berfokus pada pengembangan sistem yang dapat melakukan tugas-tugas yang biasanya memerlukan kecerdasan manusia, seperti penalaran, pembelajaran, dan pengambilan keputusan [7]. Kecerdasan buatan dirancang untuk meniru cara berpikir dan bertindak manusia dalam menyelesaikan masalah, mengenali pola, dan menyesuaikan perilaku berdasarkan pengalaman atau data yang tersedia. Kecerdasan buatan telah diterapkan di berbagai bidang, tak terlepas diantaranya adalah industri game. Penerapan kecerdasan buatan dalam pengembangan *game* bertujuan untuk menciptakan pengalaman bermain yang lebih baik, dengan memungkinkan NPC

untuk berperilaku secara logis dan adaptif, kecerdasan buatan pada NPC dapat menciptakan tantangan yang sesuai dengan kemampuan pemain serta mendukung alur cerita permainan.

Kecerdasan buatan dapat diimplementasikan pada game dengan menggunakan berbagai metode, mulai dari aturan logika (*rule-based systems*), *finite state machine*, hingga algoritma berbasis pembelajaran dan optimasi seperti *machine learning* dan *swarm intelligence*. Metode *rule-based* merupakan salah satu pendekatan paling awal dan sederhana, di mana perilaku karakter dalam *game* dikendalikan oleh seperangkat aturan deterministik yang telah ditentukan sebelumnya. Misalnya, jika karakter musuh melihat pemain, maka ia akan menyerang; jika tidak, maka ia akan tetap diam. Meskipun mudah diimplementasikan, pendekatan ini memiliki keterbatasan dalam hal adaptivitas dan tidak mampu merespons kondisi yang tidak terduga secara fleksibel [23].

Finite State Machine (FSM) adalah metode yang lebih terstruktur, di mana perilaku karakter diatur dalam beberapa keadaan dengan transisi yang dikontrol oleh kondisi tertentu [24]. Sebagai contoh, karakter dalam *game* dapat berada dalam keadaan “berjaga”, “mengejar”, atau “menyerang”, dan akan berpindah di antara keadaan tersebut berdasarkan stimulus yang diterima dari lingkungan permainan. Dengan demikian, FSM menawarkan kontrol logis yang lebih jelas dibanding metode rule-based. Namun, FSM dapat menjadi kompleks dan sulit dipelihara jika jumlah *state* dan transisi terlalu banyak [25].

Untuk menangani permasalahan tersebut metode *machine learning* dapat digunakan untuk menciptakan NPC dalam *game* yang lebih adaptif. Meskipun dapat menciptakan NPC yang lebih adaptif, *machine learning* akan membutuhkan data pelatihan besar dan sumber daya komputasi signifikan yang akan sangat berdampak pada optimasi *game* secara keseluruhan [26]. Sebagai alternatif yang ringan dan efektif, *swarm intelligence* telah menjadi salah satu pendekatan yang menarik dalam pengembangan kecerdasan buatan, khususnya untuk kondisi dimana diperlukannya adaptasi terhadap lingkungan yang dinamis.

2.2.5.1 Swarm Intelligence

Swarm intelligence adalah konsep pendekatan kecerdasan buatan yang terinspirasi oleh perilaku kolektif sistem alami, seperti kawanan burung, koloni semut, atau gerombolan ikan [13]. Konsep ini berfokus pada bagaimana sekelompok agen sederhana dapat bekerja sama untuk mencapai tujuan bersama melalui interaksi lokal tanpa adanya pengendali pusat. Meskipun setiap agen memiliki perilaku individual yang sederhana, perilaku kolektif yang dihasilkan dapat membentuk solusi yang kompleks dan adaptif terhadap perubahan lingkungan. *Swarm intelligence* memiliki kemampuan untuk melakukan eksplorasi ruang solusi secara paralel dan beradaptasi dalam situasi yang berubah-ubah, menjadikannya cocok untuk menyelesaikan berbagai permasalahan optimasi, klasifikasi, hingga navigasi [27]. Beberapa algoritma yang termasuk ke dalam *swarm intelligence* adalah *Ant Colony Optimization* (ACO) yang meniru perilaku semut dalam mencari jalur terpendek menuju sumber makanan, *Particle Swarm Optimization* (PSO) yang meniru gerakan kawanan burung dalam mencari posisi terbaik, serta *Artificial Bee Colony* (ABC) yang terinspirasi dari strategi pencarian makanan oleh lebah madu.

2.2.6 Algoritma *Artificial Bee Colony* (ABC)

Artificial Bee Colony (ABC) adalah algoritma pencarian *metaheuristik* yang terinspirasi oleh perilaku lebah madu dalam mencari dan mengevaluasi sumber makanan [14]. Algoritma ini diperkenalkan oleh Dervis Karaboga pada tahun 2005 sebagai metode optimasi numerik berbasis *swarm intelligence* [28]. Dalam ABC, proses pencarian solusi diilustrasikan melalui tiga jenis lebah buatan, yaitu *employed bees* (lebah pekerja), *onlooker bees* (lebah pengamat), dan *scout bees* (lebah pencari). ABC bekerja dalam populasi solusi, yang masing-masing merepresentasikan posisi sumber makanan. Nilai dari setiap sumber makanan mencerminkan kualitas solusi (fitness) dari permasalahan yang dioptimasi. Proses iteratif dilakukan hingga ditemukan solusi terbaik atau kriteria penghentian terpenuhi.

Dalam eksekusinya, ABC akan mengikuti tahapan berikut ini:

```
Initialization Phase
REPEAT
  Employed Bees Phase
  Onlooker Bees Phase
  Scout Bees Phase
  Memorize the best solution achieved so far
UNTIL(Cycle=Maximum Cycle Number or a Maximum CPU time)
```

Gambar x.x pseudocode ABC

1. Inisialisasi Populasi

   Pada tahap awal algoritma *Artificial Bee Colony* (ABC), seluruh vektor populasi sumber makanan (dilambangkan sebagai $x_m\rightarrow$, dengan m = 1...SN, di mana SN adalah jumlah populasi) diinisialisasi oleh *scout bees* (lebah pengintai), dan parameter kontrol ditetapkan. Setiap sumber makanan mewakili sebuah solusi terhadap permasalahan optimasi, sehingga setiap vektor $x_m\rightarrow$ terdiri atas n variabel (x_{mi}, dengan i = 1...n) yang akan dioptimalkan guna meminimalkan fungsi objektif. Proses inisialisasi variabel-variabel ini dilakukan dengan membangkitkan nilai acak berdasarkan batas bawah (l_i) dan batas atas (u_i) dari masing-masing parameter, sesuai dengan persamaan:

$$xmi = li + rand(0,1) \times (ui - li) \quad (2.1)$$

   Dengan demikian, nilai awal dari setiap variabel ditentukan secara acak namun tetap berada dalam ruang pencarian yang dibatasi oleh nilai l_i dan u_i. Pendekatan ini memberikan keragaman awal solusi, yang krusial dalam proses eksplorasi solusi optimal.

2. Fase *Employed Bee*

Setelah proses inisialisasi, lebah pekerja (*employed bees*) akan mencari sumber makanan baru ($\upsilon_m\rightarrow$) yang mengandung lebih banyak nektar di sekitar lingkungan sumber makanan ($x_m\rightarrow$) yang mereka ingat. Proses pencarian ini dilakukan dengan mengevaluasi sumber makanan tetangga untuk menentukan apakah sumber tersebut lebih menguntungkan (*fit*). Salah satu metode yang digunakan untuk menentukan tetangga adalah dengan menghasilkan solusi baru menggunakan rumus berikut:

$$\upsilon_{mi} = x_{mi} + \phi_{mi} \times (x_{mi} - x_{ki}) \quad (2.2)$$

Di mana $x_k\rightarrow$ adalah sumber makanan lain yang dipilih secara acak dari populasi ($k \neq m$), i adalah indeks parameter yang juga dipilih secara acak, dan ϕ_{mi} adalah bilangan acak dalam rentang $[-a, a]$.

3. Evaluasi *Fitness*

Setelah solusi baru $\upsilon_m\rightarrow$ dihasilkan, nilai *fitness*-nya dihitung, kemudian dilakukan seleksi secara *greedy* antara solusi baru ($\upsilon_m\rightarrow$) dan solusi lama ($x_m\rightarrow$), dengan memilih solusi yang memiliki nilai *fitness* lebih baik untuk disimpan. Untuk permasalahan minimisasi, nilai *fitness* dari suatu solusi $x_m\rightarrow$ dihitung berdasarkan nilai fungsi objektif $f_m(x_m\rightarrow)$, dengan rumus sebagai berikut:

$$fitm(xm\rightarrow) = \begin{cases} \frac{1}{1+fm(xm\rightarrow)}, jika\ fm(xm\rightarrow) \geq 0 \\ 1 + abs\big(fm(xm\rightarrow)\big), jika\ fm(xm\rightarrow) < 0 \end{cases} \quad (2.3)$$

Rumus ini dimaksudkan agar solusi dengan nilai fungsi objektif lebih kecil mendapatkan nilai *fitness* yang lebih tinggi, sehingga berpeluang lebih besar untuk dipertahankan dalam populasi.

4. Fase *Onlooker Bee*

   Lebah yang tidak bekerja (*unemployed bees*) terdiri atas dua kelompok, yaitu *onlooker bees* dan *scout bees*. Dalam proses algoritma ABC, lebah pekerja akan membagikan informasi mengenai sumber makanan mereka kepada lebah pengamat (*onlooker*) yang menunggu di dalam sarang. Lebah pengamat kemudian memilih sumber makanan secara probabilistik berdasarkan informasi yang diberikan oleh lebah pekerja. Pemilihan ini bergantung pada nilai probabilitas yang dihitung dari nilai fitness masing-masing sumber makanan. Nilai probabilitas pemilihan sumber makanan makanan pm oleh lebah pengamat terhadap solusi $\vec{xm}$ dihitung menggunakan persamaan:

$$p_m = \frac{fit_m(\vec{x_m})}{\sum_{m=1}^{SN} fit_m(\vec{x_m})} \qquad (2.4)$$

   Setelah sumber makanan $\vec{xm}$ dipilih secara probabilistik, lebah pengamat akan menentukan solusi tetangga $\vec{vm}$ menggunakan rumus yang sama seperti pada tahap *employed bee*. Nilai *fitness* dari solusi baru tersebut dihitung dan kemudian dibandingkan dengan solusi sebelumnya menggunakan seleksi *greedy*. Dengan demikian, lebih banyak lebah pengamat akan direkrut untuk mengeksplorasi sumber makanan yang lebih kaya, yang menunjukkan adanya perilaku *positive feedback* dalam pencarian solusi optimal.

5. Fase *Scout Bee*

   Lebah yang tidak bekerja (*unemployed bees*) dan memilih sumber makanan secara acak disebut sebagai *scout bees*. Dalam algoritma ABC, lebah pekerja yang tidak mampu memperbaiki solusi yang mereka miliki setelah sejumlah percobaan tertentu, ditentukan oleh *parameter limit* atau *abandonment criteria*, akan berubah menjadi *scout*. Solusi yang tidak mengalami perbaikan tersebut kemudian ditinggalkan. Setelah itu, lebah yang telah berubah menjadi *scout*

akan mulai mencari solusi baru secara acak dalam ruang pencarian. Misalnya, jika solusi $\vec{x}m$ telah ditinggalkan, maka solusi baru yang ditemukan oleh *scout bee* didefinisikan menggunakan persamaan (2.1) yang sama pada tahap inisialisasi populasi dari algoritma ABC.

Dengan demikian, sumber makanan yang sejak awal memiliki kualitas buruk atau menjadi buruk akibat eksploitasi akan ditinggalkan, dan hal ini menciptakan perilaku *negative feedback* untuk menyeimbangkan mekanisme *positive feedback* yang sebelumnya terjadi pada fase *onlooker bees*.

6. Seleksi Global Solusi Terbaik

   Setelah seluruh fase dijalankan, solusi terbaik dari populasi diperbarui. Proses iterasi algoritma ABC akan terus berjalan hingga mencapai jumlah siklus maksimum atau tercapainya kriteria penghentian lainnya.

2.2.7 *Godot Engine*

Godot Engine adalah sebuah mesin pengembangan *game* sumber terbuka yang memungkinkan pengembang untuk membuat *game* 2D dan 3D dengan cepat dan efisien [29, 30]. *Engine* ini mendukung berbagai bahasa pemrograman, termasuk GDScript, yang mirip dengan Python, serta C# dan VisualScript. Godot terkenal dengan kemudahan penggunaan, fleksibilitas dalam desain *game*, serta kemampuannya untuk mengembangkan *game* lintas platform. Fitur-fitur seperti *scene system*, *scripting Application Programming Interface* (API), dan *toolset* yang kaya menjadikannya pilihan populer di kalangan pengembang *indie*.

2.2.8 *Game Development Life Cycle* (GDLC)

Game Development Life Cycle (GDLC) adalah metode pengembagan game yang membagi siklus pengembangan menjadi beberapa fase, sehingga memungkinkan perencanaan, pengelolaan sumber daya, dan kolaborasi yang lebih baik di antara tim

pengembangan [31]. Pengembangan game dengan metode GDLC akan dibagi ke dalam beberapa tahapan yang ditunjukkan pada gambar x.x berikut ini.

Gambar x.x. tahapan pengembangan pada GDLC

1. *Initiation*

   Tahap ini merupakan fase awal dalam pengembangan *game*, di mana ide dasar dan visi dari *game* mulai dirumuskan. Pada tahap ini, pengembang mendefinisikan *genre*, target pemain, *platform* yang digunakan, serta keunikan *gameplay* yang ingin ditonjolkan. Tujuan utama dari fase ini adalah menyusun dokumen awal seperti *concept document* atau *game pitch* yang menggambarkan inti dari proyek game tersebut.

2. *Pre-production*

   Setelah ide dasar disetujui, proses berlanjut ke tahap *pre-production*. Pada fase ini, tim pengembang mulai merinci spesifikasi teknis dan artistik dari *game*. Hal ini mencakup pembuatan *Game Design Document* (GDD), pengembangan konsep visual, seperti *storyboard*, karakter, dan latar, serta perencanaan sistem teknis seperti arsitektur pemrograman dan kebutuhan lainnya. Fase ini sangat penting karena menjadi fondasi dari pengembangan selanjutnya.

3. *Production*

   Production merupakan tahap inti dari pengembangan *game,* di mana seluruh elemen desain mulai diwujudkan ke dalam bentuk nyata. Aktivitas pada tahap ini meliputi pengkodean fitur *gameplay*, pembuatan aset visual dan audio, integrasi sistem kecerdasan buatan, serta pembangunan *level* atau dunia permainan. Ini juga termasuk penerapan mekanika permainan seperti kontrol pemain, interaksi objek, dan logika musuh atau NPC.

4. *Alpha Testing*

   Setelah fitur utama dikembangkan, pengembangan *game* akan masuk ke tahap *testing*. Tujuan dari fase ini adalah mengidentifikasi dan memperbaiki *bug*, mengevaluasi performa, serta memastikan seluruh elemen *game* berjalan sesuai rancangan. Jenis pengujian yang dilakukan bisa meliputi *unit testing*,

integration testing, *playtesting*, dan pengujian performa kecerdasan buatan. Tahap ini sangat penting untuk menjamin stabilitas dan pengalaman bermain yang baik.

5. *Beta Testing*

   Fase *beta* adalah tahap distribusi terbatas *game* kepada sekelompok pengguna tertentu untuk menguji *game* dalam konteks nyata. Umpan balik dari pemain *beta* akan digunakan untuk menyempurnakan aspek *gameplay*, keseimbangan, serta menemukan *bug* yang mungkin terlewat dalam pengujian internal. *Beta testing* membantu memastikan bahwa *game* telah memenuhi ekspektasi pengguna sebelum dirilis secara publik.

6. *Release*

   Tahap release adalah fase di mana game secara resmi diluncurkan ke publik. Pada fase ini, pengembang harus memastikan kesiapan sistem, dokumentasi pengguna, serta dukungan pasca-rilis seperti *patching* dan pembaruan konten. Evaluasi lanjutan terhadap perilaku pemain dan performa kecerdasan buatan juga dapat dilakukan sebagai dasar pengembangan versi berikutnya atau iterasi lebih lanjut.

2.2.9 *Game Design Document* (GDD)

Game Design Document (GDD) adalah sebuah dokumen komprehensif yang digunakan untuk merencanakan dan mengatur pengembangan sebuah *game* [32]. GDD berfungsi sebagai panduan bagi seluruh tim pengembang *game*, mulai dari desainer, pengembang perangkat lunak, hingga seniman grafis, untuk memastikan bahwa visi dan tujuan permainan tercapai dengan konsisten. Dokumen ini mencakup berbagai aspek desain *game*, seperti *game*play mechanics, narasi, karakter, *level design*, serta elemen estetika dan teknis lainnya. GDD juga dapat mencakup pengaturan *budget*, *timeline*, serta *prototyping* untuk menguji ide-ide *game*play. GDD penting dalam menjaga agar pengembangan *game* tetap terarah dan menghindari kesalahan yang dapat merugikan proses produksi. Dengan memiliki GDD yang jelas, pengembang dapat memastikan kualitas dan kesuksesan *game* yang sedang dikembangkan.

2.2.10 Pengujian

Pengujian atau *testing* adalah tahap dalam proses pengembangan game yang bertujuan untuk mengevaluasi kesesuaian antara hasil implementasi dan spesifikasi yang telah dirancang sebelumnya. Tahap pengujian perlu dilakukan untuk menemukan perbedaan antara kondisi aktual dan yang diharapkan serta menilai performa dari model *pathfinding* yang dikembangkan. Proses ini dapat dikatakan sebagai tahap paling penting dalam pengembangan dikarenakan jaminan kualitas dari model akan bergantung pada proses ini. Penelitian ini akan melakukan dua tahapan pengujian sesuai dengan kerangka kerja dari GDLC, yaitu *alpha testing* dan *beta testing*.

2.2.10.1 Alpha Testing

Pada tahap pengujian ini akan dilakukan beberapa pengujian untuk sistem *pathfinding* dengan menggunakan algoritma ABC yang telah diimplementasikan.

1. *Black-box testing*

   Black-box testing adalah metode pengujian perangkat lunak dimana pengujian akan berfokus pada fungsi sistem dari sudut pandang pengguna tanpa memerlukan *source code* [33]. *Black-box testing* akan digunakan untuk melakukan pengecekan pada tiap-tiap fitur dan bagian yang terdapat pada *game* yang dikembangkan bekerja sebagaimana mestinya. Beberapa teknik yang dapat digunakan dalam mengimplementasikan pengujian dengan menggunakan metode ini diantaranya adalah *all pair testing, boundary value analysis, cause-effect graph, equivalence partitioning, fuzzing, orthogonal array testing*, dan *state transition*. Setelah dilakukan rangkaian pengecekan tiap fungsi yang ada, kemudian hasil pengujian akan dimasukkan ke dalam rumusan berikut untuk mendapatkan persentase keberhasilan dari pengujian.

$$Tingkat\ Keberhasilan = \left(\frac{Test\ Case\ Berhasil}{Jumlah\ Test\ Case}\right) \times 100 \quad (2.5)$$

2. Waktu Tempuh

Pengujian ini akan terfokus pada pengukuran lama perjalanan pada jalur yang dihasilkan algoritma dengan menggunakan persamaan (2.6) berikut.

$$T_{total} = \sum_{i=1}^{n-1} \frac{dist(node_i, node_{i+1})}{v_i} \quad (2.6)$$

Persamaan ini akan menunjukkan waktu total (T_{total}) yang dibutuhkan NPC untuk menempuh jarak dari titik a ke titik b dengan kecepatan sebesar *v*.

3. Kedinamisan

Pada pengujian ini kemampuan algoritma beradaptasi terhadap perubahan lingkungan akan diukur dengan menggunakan persamaan (2.7) berikut ini.

$$D = \frac{R_{update}}{R_{total}} \quad (2.7)$$

Persamaan diatas akan menunjukkan rasio perubahan rute terhadap perubahan lingkungan. Nilai kedinamisan (D) akan didapatkan dari melakukan perbandingan jumlah rute yang diperbarui dengan benar saat terjadi perubahan (R_{update}) terhadap jumlah total perubahan lingkungan yang diujikan (R_{total}). Semakin dekat nilai D dengan 1 maka akan menunjukkan sistem semakin adaptif terhadap perubahan terhadap lingkungan.

4. Keamanan Lintasan

Pengujian ini akan mengukur berapa kali agen mengalami tabrakan dengan rintangan statis atau dinamis selama pengujian dengan menggunakan persamaan (2.8) berikut ini.

$$C_{total} = \sum_{t=1}^{T} colission(t) \quad (2.8)$$

Persamaan ini akan melakukan penjumlahan jumlah kolisi selama pengujian dilakukan. Nilai dari *colission*(*t*) akan bernilai 1 jika terjadi kolisi pada satuan waktu *t* dan akan bernilai 0 jika tidak terjadi kolisi. Selanjutnya akan didapatkan nilai normalisasi untuk mengetahui tingkat keamanan dari sistem dengan menggunakan persamaan (2.9)

$$C_{norm} = \frac{C_{total}}{Step_{total}} \quad (2.9)$$

Persamaan ini akan memberikan nilai probabilitas dari kecelakaan dari tiap node yang dilewati di dalam lintasan. Nilai ini didapatkan dengan membagi

nilai kolisi total selama pengujian (C_{total}) dengan total jumlah node yang dilewati pada pengujian ($Step_{total}$).

5. Skalabilitas

   Pengujian ini akan mengukur seberapa baik algoritma tetap efisien saat ukuran masalah bertambah. Persamaan (2.10) berikut ini akan menunjukkan rasio pertumbuhan waktu komputasi, menunjukkan tingkat skalabilitas dari sistem.

$$Scalability = \frac{T(n_2) - T(n_1)}{n_2 - n_1} \quad (2.10)$$

   Waktu komputasi untuk ukuran masalah n ditunjukkan dengan $T(n)$. Semakin kecil nilai variabel *Scalability* akan menunjukan semakin baiknya skalabilitas dari sistem.

6. Beban Komputasi

   Pengujian ini akan berfokus pada pengukuran penggunaan sumber daya saat eksekusi. Pada persamaan (2.11) akan diukur beban komputasi pada CPU selama pengujian berlangsung, sedangkan persamaan (2.12) akan mengukur beban komputasi pada memori komputer.

$$CPU_{avg} = \frac{\sum_{t=1}^{T} CPU(t)}{T} \quad (2.11)$$

$$Mem_{avg} = \frac{\sum_{t=1}^{T} Mem(t)}{T} \quad (2.12)$$

2.2.10.2 Beta Testing

Salah satu pendekatan umum dalam *beta testing* adalah *open playtest*, yaitu proses pengujian terbuka di mana pengguna eksternal diberikan akses terhadap produk untuk dimainkan secara langsung. Melalui metode ini, pengembang dapat memperoleh umpan balik nyata dari pengalaman pengguna dalam kondisi penggunaan yang mendekati situasi sebenarnya.

Untuk mendukung pengambilan data yang lebih terstruktur mengenai persepsi pengguna terhadap produk, digunakan instrumen evaluasi seperti *User Experience Questionnaire* (UEQ). UEQ adalah kuesioner terstandarisasi yang dirancang untuk

mengukur pengalaman pengguna berdasarkan enam skala utama: *Attractiveness*, *Perspicuity*, *Efficiency*, *Dependability*, *Stimulation*, dan *Novelty* [34, 35]. Berikut rincian dari keenam skala tersebut.

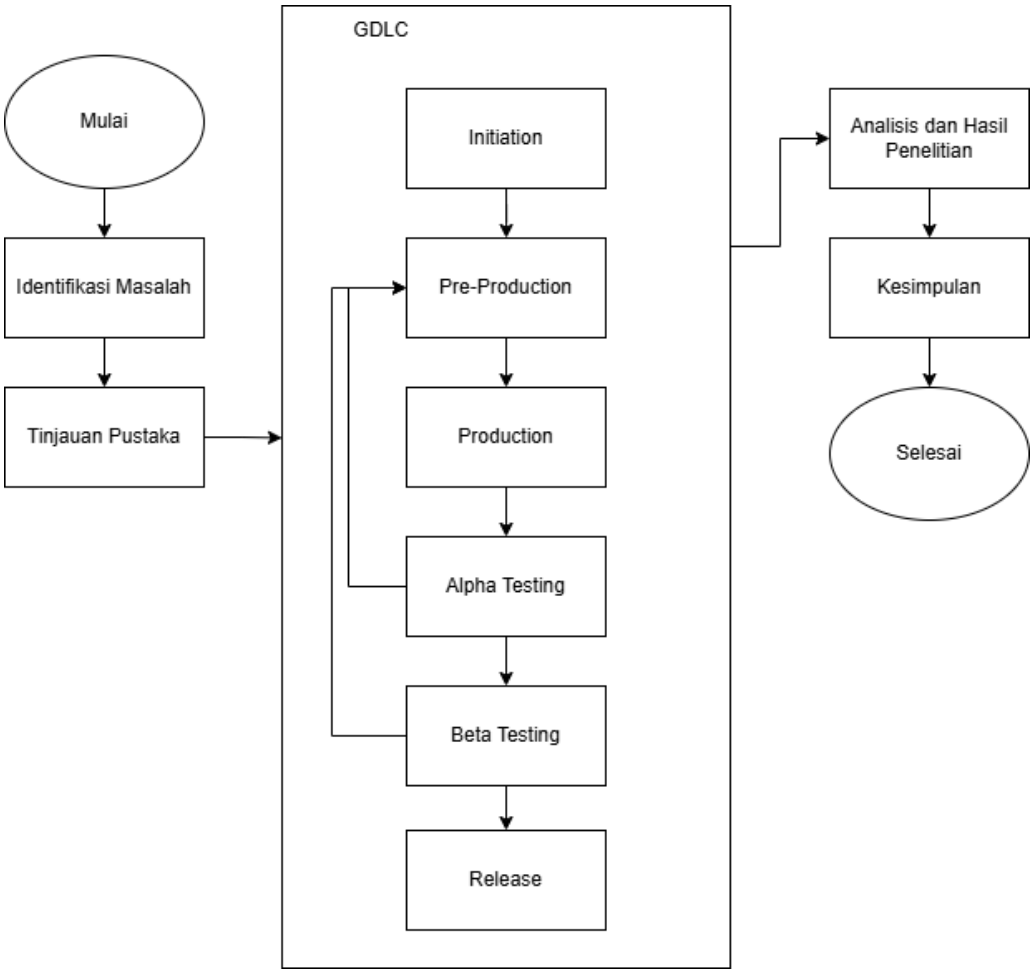
1. *Attractiveness*, mengukur kesan menyeluruh dari produk, apakah pengguna merasa menyenangkan atau tidak.
2. *Perspicuity*, mengukur kemudahan memahami dan mempelajari penggunaan produk.
3. *Efficiency*, mengukur kecepatan dan kemudahan pengguna mencapai tujuan.
4. *Dependability*, mengukur sejauh mana pengguna merasa dalam kendali dan dapat memprediksi perilaku sistem.
5. *Stimulation*, mengukur sejauh mana produk menarik, memotivasi, dan menginspirasi pengguna.
6. *Novelty*, mengukur kebaruan dan kreativitas desain produk.

Dengan menggunakan UEQ, pengembang dapat menilai kualitas pengalaman pengguna dari aspek kegunaan maupun emosional secara kuantitatif, sehingga hasil beta testing dapat digunakan sebagai dasar untuk melakukan perbaikan sebelum peluncuran akhir produk.

BAB III
METODOLOGI PENELITIAN

3.1 Alur Penelitian

Bab ini akan menjelaskan proses tahapan yang ditempuh dalam penelitian ini. Berikut diagram alir yang menunjukkan tahapan dalam penelitian ini.



Gambar 3.1 Alur Penelitian

3.2 Penjabaran Langkah Penelitian

3.2.1 Identifikasi Masalah

Identifikasi masalah merupakan langkah pertama dalam merumuskan permasalahan yang akan diinvestigasi sesuai dengan konteks yang diangkat dalam penelitian ini.

3.2.2 Tinjauan Pustaka

Tinjauan pustaka dilakukan untuk mendapatkan landasan yang konkrit dan kokoh sebagai landasan dari penelitian yang dilakukan kali ini. Pada tahap ini dilakukan studi terhadap penelitian terdahulu yang menjadi sumber utama landasan teori yang digunakan pada penelitian ini. Penelitian terdahulu ini berupa artikel, jurnal, buku, dan media semacamnya yang memiliki keterkaitan dengan konteks penelitian yang dilakukan kali ini. Hasil dari pada tahapan ini kemudian dituliskan pada BAB II yang berisikan hasil tinjauan studi dan landasan-landasan teori yang akan menunjang penelitian ini.

3.2.3 Game Development Life Cycle

Penelitian ini akan menggunakan kerangka kerja *Game Development Life Cycle* (GDLC) dalam pengembangan sistem *game racing* yang mengimplementasikan algoritma ABC.

3.2.3.1 Initiation

Fase ini adalah tahap awal dalam proses pengembangan game, di mana gagasan utama dan visi permainan mulai disusun. Pada tahap ini, pengembang menetapkan genre, sasaran pemain, *platform* yang akan digunakan, serta ciri khas *gameplay* yang ingin diunggulkan. Fokus utama pada fase ini adalah membuat dokumen awal, seperti *concept document* atau *game pitch*, yang merangkum esensi dari proyek game tersebut.

3.2.3.2 Pre-production

Setelah ide dasar disetujui, proses berlanjut ke tahap *pre-production*. Pada fase ini, tim pengembang mulai merinci spesifikasi teknis dan artistik dari *game*. Hal ini mencakup pembuatan *Game Design Document* (GDD), pengembangan konsep visual, seperti *storyboard*, karakter, dan latar, serta perencanaan sistem teknis seperti arsitektur pemrograman dan kebutuhan lainnya. Fase ini sangat penting karena menjadi fondasi dari pengembangan selanjutnya.

3.2.3.3 Production

Production merupakan tahap inti dari pengembangan *game*, di mana seluruh elemen desain mulai diwujudkan ke dalam bentuk nyata. Aktivitas pada tahap ini meliputi pengkodean fitur *gameplay*, pembuatan aset visual dan audio, integrasi sistem kecerdasan buatan, serta pembangunan *level* atau dunia permainan. Ini juga termasuk penerapan mekanika permainan seperti kontrol pemain, interaksi objek, dan logika musuh atau NPC.

3.2.3.4 Alpha Testing

Setelah fitur utama dikembangkan, pengembangan *game* akan masuk ke tahap *testing*. Tujuan dari fase ini adalah mengidentifikasi dan memperbaiki *bug*, mengevaluasi performa, serta memastikan seluruh elemen *game* berjalan sesuai rancangan. Jenis pengujian yang dilakukan bisa meliputi *unit testing*, *integration testing*, *playtesting*, dan pengujian performa kecerdasan buatan. Tahap ini sangat penting untuk menjamin stabilitas dan pengalaman bermain yang baik.

3.2.3.5 Beta Testing

Fase *beta* adalah tahap distribusi terbatas *game* kepada sekelompok pengguna tertentu untuk menguji *game* dalam konteks nyata. Umpan balik dari pemain *beta* akan digunakan untuk menyempurnakan aspek *gameplay*, keseimbangan, serta menemukan *bug* yang mungkin terlewat dalam pengujian internal. *Beta testing* membantu

memastikan bahwa *game* telah memenuhi ekspektasi pengguna sebelum dirilis secara publik.

3.2.3.6 Release

Tahap release adalah fase di mana game secara resmi diluncurkan ke publik. Pada fase ini, pengembang harus memastikan kesiapan sistem, dokumentasi pengguna, serta dukungan pasca-rilis seperti *patching* dan pembaruan konten. Evaluasi lanjutan terhadap perilaku pemain dan performa kecerdasan buatan juga dapat dilakukan sebagai dasar pengembangan versi berikutnya atau iterasi lebih lanjut.

3.2.4 Analisis dan Hasil Penelitian

Hasil dari data dari pengujian yang telah dilakukan pada tahap *alpha testing* dan *beta testing* pada kerangka kerja GDLC kemudian akan dianalisis untuk mengkaji performa dari algoritma ABC dalam melakukan *pathfinding* pada *game* dengan genre *racing*. Hasil analisis kemudian akan dituliskan pada Bab 4 pada laporan tugas akhir ini.

3.2.5 Kesimpulan

Kesimpulan kemudian akan ditarik dari hasil temuan dan analisis pada penelitian ini. Kesimpulan ini akan merefleksikan pencapaian tujuan penelitian serta efektivitas pendekatan yang digunakan. Dengan demikian, kesimpulan diharapkan dapat memberikan gambaran yang menyeluruh terhadap kontribusi dan implikasi dari penelitian ini.

3.3 Alat dan Bahan Tugas Akhir

3.3.1 Alat

3.3.2 Bahan

3.4 Deskripsi Video Game

3.4.1 Cara Bermain

3.5 Rancangan Sistem Pathfinding

3.6 Rancangan Pengujian

3.6.1 Skenario Pengujian

3.6.2 Map Pengujian

DAFTAR PUSTAKA

[1] PwC, "PwC Global Entertainment and Media Outlook 2024-28," PwC, 16 Juli 2024. [Online]. Available: https://www.pwc.com/gx/en/news-room/press-releases/2024/pwc-global-entertainment-and-media-outlook-2024-28.html. [Accessed 24 Februari 2025].

[2] A. Kessel, "The Global Video Game Market Could Soon Approach $500 Billion, New Report Says," Investopedia, 13 September 2024. [Online]. Available: https://www.investopedia.com/global-video-game-market-could-double-in-10-years-report-says-8712136. [Accessed 6 Maret 2025].

[3] S. A. Fauzan, S. R. Pradana, M. Hikal, M. B. Ashfiya, Y. I. Kurniawan and B. Wijayanto, "Implementasi Game Development Life Cycle Model Pengembangan Arnold Hendrick's Dalam Pembuatan Game Puzzle-RPG Enigma's Dungeon," *Jurnal Ilmu Komputer dan Informatika*, vol. 2, no. 2, pp. 113-126, 2022.

[4] A. Mulachela, K. Rizki and Y. Wahyudin, "Analisis Perkembangan Industri Game di Indonesia Melalui Pendekatan Rantai Nilai Global (Global Value Chain)," *Indonesian Journal of Global Discourse* , vol. 2, no. 2, pp. 32 - 51, 2020.

[5] R. Furukado and G. Hagiwara, "Examining the effects of digital gameplay of the racing genre on mood and heart rate," *Journal of Digital Life*, vol. 1, no. 5, 2021.

[6] A. Serdouk and A. C. Bessam, " Bots in Newsrooms: What Future for Human Journalists?," *Media Watch*, vol. 14, no. 1, pp. 100 - 115, 2023.

[7] C. Zhang and Y. Lu, "Study on artificial intelligence: The state of the art and future prospects," *Journal of Industrial Information Integration*, vol. 23, p. 100224, 2021.

[8] A. Y. Ayas, H. Aydin, A. Çetinkaya and Z. Güney, "Artificial Intelligence (AI)-Based Self-Deciding Character Development Application in Two-Dimensional Video Gamesİki Boyutlu Video Oyunlarında Yapay Zekâ (AI) Tabanlı Kendi Kendine Karar Veren Karakter Geliştirme Uygulaması," *Bilgi ve İletişim Teknolojileri Dergisi,* vol. 5, 2023.

[9] "A Review of Pathfinding in Game Development," *CEPAT Journal of Computer Engineering: Progress, Application and Technology,* vol. 1, no. 1, pp. 46-55 , 2022.

[10] C. Bayar, "Game Mechanics #2: Path Finding," Medium, 7 November 2022. [Online]. Available: https://canbayar91.medium.com/game-mechanics-2-path-finding-ab4f55c1d580. [Accessed 5 Agustus 2025].

[11] J.-M. Kim and T. Woo, "The Study on Path Optimization and User Guidance Generation in Racing Games Using Reinforcement Learning," *Journal of Digital Contents Society,* vol. 26, no. 1, pp. 203-210, 2025.

[12] Y. Jia, M. Bhatt and N. Mehr, "RAPID: Autonomous Multi-Agent Racing using Constrained Potential Dynamic Games," in *2023 European Control Conference (ECC),* Bucharest, 2023.

[13] J. Tang, G. Liu and Q. Pan, "A Review on Representative Swarm Intelligence Algorithms for Solving Optimization Problems: Applications and Trends," *IEEE/CAA JOURNAL OF AUTOMATICA SINICA,* vol. 8, no. 10, pp. 1627-1643, 2021.

[14] C. Wang, P. Shang and P. Shen, "An improved artificial bee colony algorithm based on Bayesian estimation," *Complex &Intelligent Systems* , vol. 8, p. 4971–4991, 2022.

[15] O. Sinkevych, Y. Boyko, B. Sokolovskyy, M. Pavlyk, O. Yarosh and O. Futey, "UNCREWED VEHICLE PATHFINDING APPROACH BASED ON ARTIFICIAL BEE COLONY METHOD," *ADVANCES IN CYBER-PHYSICAL SYSTEMS* , vol. 9, no. 1, 2024.

[16] D. M. T. Nguyen, G. G. Fard, A. Atkins, P. Bontempo, M. L. Iuzzolino and O. Peleg, "Honey bees find the shortest path: a collective flow-mediated approach," *Artificial Life and Robotics,* vol. 28, pp. 1-7, 2023.

[17] A. Ebrahimnejad, M. Enayattabr, H. Motameni and H. Garg, "Modified artificial bee colony algorithm for solving mixed interval-valued fuzzy shortest path problem," *Complex & Intelligent Systems,* vol. 7, pp. 1527-1545, 2021.

[18] S. M. Hussein and A. S. Al-Araji, "Enhancement of a Path-Finding Algorithm for the Hovercraft System Based on Intelligent Hybrid Stochastic Methods," *INASS,* vol. 17, no. 2, pp. 346-364, 2024.

[19] J. Betz, H. Zheng, A. Liniger, U. Rosolia, P. Karle, M. Behl, V. Krovi and R. Mangharam, "Autonomous Vehicles on the Edge: A Survey on Autonomous Vehicle Racing," *Intelligent Transportation Systems,* vol. 3, pp. 458-488, 2022.

[20] "Investigation of player emotion and instinct on racing video game using 6-11 framework (Case study: split second (2010) single player modes)," *Procedia Computer Science,* vol. 245, no. 2, pp. 87-99, 2024.

[21] P. Mehta, H. Shah, S. Shukla and S. Verma, "A Review on Algorithms for Pathfinding in Computer Games," in *International Conference on Innovations in Information Embedded and Communication Systems*, 2015.

[22] S. Wang, H. Xu, Y. Zhang, J. Lin, C. Lu, X. Wang and W. Li, "Where Paths Collide: A Comprehensive Survey of Classic and Learning-Based Multi-Agent Pathfinding," *arXiv preprint arXiv:2505.19219*, 2025.

[23] S. J. Russell and P. Norvig, Artificial Intelligence A Modern Approach Third Edition, Upper Sadle RIver, New Jersey, Amerika: PRENTICE HALL, 2010.

[24] M. G. Khairi, "PENERAPAN FINITE STATE MACHINE DALAM GAME PADA MODEL KARAKTER PEMAIN MENGGUNAKAN GODOT ENGINE," *Scientica*, vol. 2, no. 7, pp. 44-54, 2024.

[25] J. Geist, T. Meade, S. Zhang and Y. Jin, "Improving FSM State Enumeration Performance for Hardware Security with RECUT and REFSM-SAT," *arXiv preprint arXiv:2311.10273*, 2023.

[26] N. C. Thompson, K. Greenewald, K. Lee and G. F. Manso, "THE COMPUTATIONAL LIMITS OF DEEP LEARNING," *arXiv preprint arXiv:2007.05558*, vol. 10, p. 2, 2022.

[27] C. WANG, S. ZHANG, T. MA, Y. XIAO, M. Z. CHEN and L. WANG, " Swarm intelligence: A survey of model classification and applications," *Chinese Journal of Aeronautics*, vol. 38, no. 3, p. 102982, 2025.

[28] D. Karaboga, "Artificial bee colony algorithm," *Scholarpedia*, vol. 5, no. 3, p. 6915, 2010.

[29] E. Beeching, J. Debangoye, O. Simonin and C. Wolf, "Godot Reinforcement Learning Agents," *arXiv preprint arXiv:2112.03636*, 2021.

[30] J. Linietsky, A. Manzur and R. Verschelde, "Github - Godot Engine," Godot, 17 December 2014. [Online]. Available: https://github.com/godotengine/godot. [Accessed 1 June 2025].

[31] T. Wibowo, D. A. Adnas, Mulyanto and A. A. Marvel, "CRAFTING AN INTERACTIVE VIDEO GAME COURT SYSTEM FOR MORAL DEVELOPMENT AND LEGAL INSIGHT," *ZONAsi: Jurnal Sistem Informasi,* vol. 7, no. 1, pp. 85-94, 2024.

[32] K. A. Ramelan, I. D. Rahadianto and M. Adharamadinka, "PERANCANGAN GAME DESIGN DOCUMENT UNTUK MENYAMPAIKAN NILAI SOSIAL DARI TRADISI NGARUMAT PUSAKA," *e-Proceeding of Art & Design,* vol. 12, no. 1, pp. 1891-1909, 2025.

[33] H. H. Alhabsji, A. Z. D. Ananda and M. A. Yaqin, "IMPLEMENTATION OF BLACKBOX TESTING IN THE PES GAME APPLICATION USING EQUIVALENT PARTITION TECHNIQUE," *Journal of Computing and Data Science,* vol. 2, no. 1, pp. 7-18, 2024.

[34] M. A. Kushendriawan, H. B. Santoso, P. O. H. Putra and M. Schrepp, "Evaluating User Experience of a Mobile Health Application Halodoc using User Experience Questionnaire and Usability Testing," *Jurnal Sistem Informasi,* vol. 17, no. 1, pp. 58-71, 2021.

[35] B. Laugwitz, T. Held and M. Schrepp, "Construction and evaluation of a user experience questionnaire," in *HCI and Usability for Education and Work,* Berlin, Springer Berlin Heidelberg, 2008, pp. 63-76.