

Penerapan Algoritma Minimax dan Alpha Beta Pruning Dalam Permainan Tactical Role-Playing Game

Raihan Febryan^{*1}, Muhammad Bambang Firdaus², Aulia Khoirunnita³, Muhammad Fawaz Saputra⁴, Reza Wardhana⁵, Gubtha Mahendra Putra⁶

^{1,2,3,5,6}Informatika, Universitas Mulawarman, Samarinda

⁴Sistem Informasi, Universitas Mulawarman, Samarinda

e-mail: ^{*1}raihan.febryan@gmail.com, ²bambangf@unmul.ac.id,

³auliakhoirunnita@unmul.ac.id, ⁴muhammadfawaz@unmul.ac.id,

⁵rezawardhana@fkti.unmul.ac.id, ⁶gubthamp@unmul.ac.id

Abstrak

Tactical Role-Playing Game (TRPG) merupakan permainan strategi berbasis giliran yang menggabungkan elemen role-playing dan pengambilan keputusan taktis pada papan/grid, sehingga kualitas tantangan permainan sangat dipengaruhi oleh kecerdasan buatan (AI) dalam menentukan langkah optimal. Penelitian ini menerapkan algoritma Minimax sebagai pengambil keputusan AI dan mengoptimalkannya menggunakan Alpha-Beta Pruning untuk mempercepat proses pencarian tanpa mengubah keputusan pada kedalaman yang sama. Evaluasi dilakukan melalui pengujian performa keputusan AI pada permainan TRPG berbasis papan 8×8 yang memiliki atribut HP/MP dan multi-aksi (move, attack, magic attack, dan end turn). Hasil pengujian pada depth = 5 menunjukkan bahwa Minimax tanpa pruning menghasilkan waktu eksekusi lebih dari 6 detik per langkah dengan perluasan node rata-rata 182.400, sedangkan penerapan Alpha-Beta Pruning menurunkan waktu menjadi sekitar 2-3 detik dengan rata-rata expanded nodes 61.700. Pengujian variasi depth juga menunjukkan bahwa kedalaman pencarian berpengaruh langsung terhadap responsivitas permainan; depth rendah lebih responsif, sedangkan depth lebih tinggi meningkatkan beban komputasi dan berpotensi menyebabkan keterlambatan. Dengan demikian, optimasi Alpha-Beta Pruning terbukti meningkatkan efisiensi pengambilan keputusan AI pada TRPG dan membuat permainan lebih responsif pada perangkat dengan sumber daya terbatas.

Kata kunci—RPG, Turn Base, algoritma minimax, alpha-beta pruning, kecerdasan buatan

1. PENDAHULUAN

Tactical Role-Playing Game (TRPG) merupakan varian permainan strategi berbasis giliran yang menekankan pengambilan keputusan taktis pada papan/grid, sehingga kualitas pengalaman bermain sangat dipengaruhi oleh kemampuan musuh buatan (AI) dalam memilih langkah yang optimal [1]. Namun, karakteristik TRPG yang memadukan aturan pergerakan ala permainan papan dengan atribut pertarungan seperti HP/MP serta pilihan aksi move, attack, magic attack, dan end turn menyebabkan ruang pencarian langkah meningkat cepat dan dapat menimbulkan beban komputasi pada proses pengambilan keputusan AI. Kompleksitas komputasi pada permainan TRPG sejenis juga telah dibahas sebagai tantangan karena banyaknya kombinasi kemungkinan langkah pada tiap giliran [2], [3], [4].

Salah satu pendekatan yang umum digunakan untuk pengambilan keputusan pada permainan turn-based dua pihak adalah Minimax, yang mengevaluasi kemungkinan langkah secara rekursif untuk memaksimalkan peluang kemenangan pemain AI (MAX) sekaligus meminimalkan peluang kemenangan lawan (MIN) [1]. Untuk meningkatkan efisiensi pencarian

tanpa mengubah hasil keputusan pada kedalaman yang sama, Alpha-Beta Pruning digunakan untuk memangkas cabang yang tidak mungkin menghasilkan solusi lebih baik, sehingga jumlah node yang dievaluasi berkurang dan waktu keputusan dapat dipercepat [5], [6]. Temuan penelitian sebelumnya juga menunjukkan bahwa penerapan optimasi Alpha-Beta dapat meningkatkan efisiensi pengambilan keputusan AI dengan mengurangi jumlah node yang perlu dievaluasi pada pencarian game-tree [7], [8], [9], [10].

Berdasarkan latar belakang tersebut, penelitian ini menerapkan algoritma Minimax dan Alpha-Beta Pruning pada pengembangan permainan TRPG berbasis papan yang menyerupai catur namun setiap bidak memiliki HP/MP serta opsi aksi serangan fisik maupun sihir. Tujuan penelitian adalah meningkatkan kualitas tantangan permainan melalui keputusan AI yang lebih taktis sekaligus menjaga efisiensi komputasi, yang dievaluasi melalui pengujian performa pengambilan keputusan AI pada skenario permainan.

Berbeda dari permainan papan klasik yang ruang aksinya relatif sederhana, TRPG yang dikembangkan memiliki atribut HP/MP dan beberapa jenis aksi per giliran, sehingga branching factor meningkat dan membutuhkan desain evaluasi serta pembatasan depth yang tepat. Penelitian ini berkontribusi pada perancangan fungsi evaluasi berorientasi HP Raja dan threat, serta pembuktian empiris bahwa Alpha-Beta Pruning menurunkan waktu respons AI secara signifikan pada skenario TRPG papan 8×8.

2. METODE PENELITIAN

Penelitian ini berfokus pada penerapan algoritma Minimax dan Alpha-Beta Pruning sebagai pengambil keputusan pada musuh (AI) dalam permainan *Tactical Role-Playing Game* (TRPG) berbasis papan [1], [5]. Proses pengembangan game dilakukan sebagai tahap implementasi sistem, sedangkan kontribusi utama penelitian terletak pada perancangan model permainan ke dalam ruang pencarian Minimax, penyusunan fungsi evaluasi, serta pengujian performa keputusan AI pada berbagai skenario permainan [3], [6].

2.1. Desain Permainan TRPG

Game yang dikembangkan mengadopsi mekanik seperti catur pada papan/grid, namun setiap bidak memiliki atribut HP dan MP, serta pilihan aksi move, attack, magic attack, dan end turn. Kondisi menang/kalah ditentukan berdasarkan aturan permainan ketika seluruh bidak lawan habis atau Raja dikalahkan. Desain ini memastikan permainan bersifat turn-based dan kompetitif dua pihak sehingga sesuai dimodelkan sebagai permainan adversarial untuk Minimax.

2.2. Representasi State dan Ruang Aksi

Permainan dimodelkan pada papan berukuran 8×8 seperti catur, namun setiap bidak memiliki atribut **HP** dan **MP**. Setiap *state* merepresentasikan kondisi permainan pada suatu giliran yang terdiri dari konfigurasi posisi bidak pada koordinat (x, y), status sel, serta atribut tiap bidak (HP, MP, status hidup/mati, dan identitas pihak). Selain itu, state menyimpan penanda giliran aktif (turn owner) agar pembangkitan langkah hanya dilakukan oleh pihak yang sedang bermain. Pemodelan berbasis state ini sejalan dengan pendekatan umum *game-tree search* pada permainan dua pihak yang menjadi dasar Minimax dan optimasinya [11], [12], [13].

Ruang aksi (*action space*) pada setiap giliran mencakup: **move**, **attack**, **magic attack**, serta **end turn**. Serangan biasa menghasilkan kerusakan sebesar **20**, sedangkan *magic attack* mengonsumsi **35 MP** dari total **100 MP** dan menghasilkan kerusakan sebesar **30**. Aksi *magic attack* hanya valid ketika $MP \geq 35$. Setiap aksi menghasilkan *successor state* melalui fungsi

transisi yang memperbarui posisi, pengurangan HP target, pengurangan MP (untuk magic), dan perpindahan giliran.

2.3. Kondisi Terminal Permainan

Berbeda dari catur klasik yang mengandalkan konsep *check* dan *checkmate*, penelitian ini menerapkan kondisi terminal berbasis HP unit kunci. Permainan dinyatakan berakhir ketika Raja kehabisan HP atau seluruh bidak dari salah satu pihak habis, sehingga kemenangan ditentukan oleh status Raja meskipun bidak lain masih tersisa. Pendekatan terminal berbasis nilai utilitas/hasil akhir pada *game tree* (menang/kalah) konsisten dengan kerangka Minimax untuk permainan dua pemain.

2.4. Fungsi Evaluasi

Karena pencarian Minimax dibatasi oleh kedalaman, state non-terminal dinilai menggunakan fungsi evaluasi $V(s)$ untuk memperkirakan keuntungan AI terhadap pemain. Penggunaan fungsi evaluasi (heuristic evaluation) pada state non-terminal merupakan praktik umum dalam game-tree search untuk memberikan estimasi nilai posisi ketika pencarian dibatasi oleh depth limit [14]. Dalam literatur *minimax search*, kualitas evaluasi pada leaf node dan kedalaman pencarian berpengaruh langsung terhadap kualitas keputusan yang diambil. Mengingat permainan berakhir saat Raja kehabisan HP, evaluasi menempatkan kondisi Raja sebagai faktor dominan:

$$V(s) = wk \cdot \Delta HP_{King} + wh \cdot \Delta HP + wm \cdot \Delta MP + wt \cdot \Delta Threat_{King} \quad (1)$$

Komponen $\Delta Threat_{King}$ yang ditunjukkan pada **persamaan (1)** merepresentasikan ancaman langsung terhadap Raja, sehingga AI tidak hanya memaksimalkan HP/MP total tetapi juga melakukan langkah taktis yang relevan terhadap kondisi menang/kalah. Gagasan mengarahkan pencarian dan evaluasi agar lebih informatif terhadap perubahan nilai minimax juga banyak dibahas pada studi pengembangan *minimax search* dan variasinya.

2.5. Parameter Pencarian dan Alpha-Beta Pruning

Pencarian Minimax dilakukan hingga kedalaman tertentu (*depth limit*) atau berhenti ketika state terminal tercapai ($HP_{Raja} \leq 0$). Untuk meningkatkan efisiensi, Alpha-Beta Pruning digunakan untuk memangkas cabang yang tidak mungkin menghasilkan nilai lebih baik daripada batas alpha/beta yang sudah ditemukan, sehingga mengurangi jumlah node yang dievaluasi tanpa mengubah keputusan optimal pada kedalaman yang sama. Mekanisme ini merupakan optimasi standar pada *game-tree search* dan banyak digunakan untuk meningkatkan efisiensi komputasi. Pembatasan kedalaman diterapkan untuk menjaga waktu respons AI tetap layak pada permainan 8×8 yang memiliki banyak kombinasi aksi karena adanya opsi serangan fisik maupun sihir [4].

3. HASIL DAN PEMBAHASAN

3.1 Hasil Pengujian Minimax tanpa Alpha-Beta Pruning

Pengujian awal dilakukan dengan menerapkan algoritma Minimax tanpa optimasi Alpha-Beta Pruning pada permainan TRPG papan 8×8 dengan metrik waktu eksekusi per langkah AI pada $depth = 5$. Konsep Minimax sebagai pencarian keputusan pada permainan dua pihak berbasis giliran menjadi landasan pemilihan metode ini [1].

Hasil menunjukkan AI tetap mampu memilih langkah valid sesuai aturan permainan, namun waktu eksekusi per langkah AI tercatat lebih dari 6 detik pada perangkat uji Intel N4020, RAM 8GB, tanpa GPU, sehingga respons permainan melambat. Peningkatan waktu ini selaras

dengan karakteristik pencarian permainan berbasis pohon yang membesar cepat ketika kedalaman pencarian bertambah, sehingga biaya komputasi meningkat signifikan [3], [6].

Secara konseptual, keterlambatan terjadi karena permainan TRPG memiliki ruang aksi yang lebih besar daripada permainan papan sederhana, sebab satu state dapat memunculkan banyak kombinasi aksi (move/attack/magic/end turn) sehingga branching factor menjadi tinggi. Dalam konteks permainan papan dengan ruang pencarian besar, Minimax tanpa optimasi berpotensi tidak layak untuk kebutuhan real-time pada perangkat komputasi terbatas [3], [4].

3.2. Hasil Penerapan Minimax dengan Alpha-Beta Pruning

Pada pengujian berikutnya, Minimax dioptimasi dengan Alpha-Beta Pruning dan diuji pada kondisi yang sama (jumlah bidak dan posisi identik) dengan depth = 5. Hasil pengujian menunjukkan waktu eksekusi per langkah AI turun menjadi sekitar 2-3 detik, lebih cepat dibanding Minimax tanpa pruning (>6 detik), sehingga AI menjadi lebih responsif.

Tabel 1. menyajikan perbandingan waktu eksekusi dan jumlah node yang dievaluasi

Metode	Depth	Skenario	Ulangan	Waktu rata-rata (detik)	Min-Max (detik)	Expanded Nodes (rata-rata)
Minimax	5	Pembukaan game dengan kondisi semua bidak masih lengkap	5	6.42	6.11-6.93	182,400
Minimax + Alpha-Beta	5	Pembukaan game dengan kondisi semua bidak masih lengkap	5	3.55	3.34-3.92	61,700

Percepatan ini mengindikasikan bahwa Alpha-Beta Pruning efektif dalam memangkas cabang pencarian yang tidak berkontribusi terhadap solusi terbaik, sehingga jumlah node/state yang perlu diproses berkurang dan waktu komputasi menurun. Temuan ini konsisten dengan prinsip Alpha-Beta yang menurunkan jumlah node yang dievaluasi tanpa mengubah keputusan pada kedalaman pencarian yang sama [1], [5].

3.3. Pengaruh Depth dan Branching Factor terhadap Responsivitas (Freeze)

Selain perbandingan waktu, pengujian menunjukkan bahwa pada depth = 3 permainan berjalan responsif, sedangkan pada depth = 5 terdapat kondisi keterlambatan yang signifikan dan dapat terjadi freeze. Hal ini memperkuat bahwa peningkatan depth memperbesar jumlah kemungkinan state yang harus dievaluasi, sehingga beban komputasi meningkat drastis seiring pertumbuhan pohon pencarian [3], [6].

Pada permainan ini, branching factor cenderung tinggi karena selain pergerakan, AI juga mempertimbangkan aksi serang fisik dan serang sihir (damage lebih tinggi namun dibatasi MP). Kombinasi pilihan aksi ini membuat pohon pencarian berkembang lebih cepat dibanding permainan dengan aksi yang lebih sederhana, sehingga pemilihan depth yang moderat menjadi kompromi antara kualitas keputusan dan waktu eksekusi yang dapat diterima dalam permainan [4].

Tabel 2. Pengaruh kedalaman pencarian terhadap waktu eksekusi AI pada TRPG 8×8

Metode	Depth	Skenario	Ulangan	Waktu rata-rata (detik)	Min-Max (detik)	Catatan
--------	-------	----------	---------	-------------------------	-----------------	---------

Minimax	3	beberapa bidak sudah mati, Raja mulai “terancam”	5	0.94	0.82-1.08	Responsif
Minimax	5	Pembukaan game dengan kondisi semua bidak masih lengkap	5	6.42	6.11-6.93	Lambat/berpotensi freeze
Minimax + Alpha-Beta	3	Pembukaan game dengan kondisi semua bidak masih lengkap	5	0.43	0.36-0.50	Responsif
Minimax + Alpha-Beta	5	beberapa bidak sudah mati, Raja mulai “terancam”	5	2.61	2.34-2.92	Responsif

3.4. Pembahasan dan Perbandingan dengan Penelitian Terkait

Jika dibandingkan dengan penelitian Putri dkk . pada Tic-Tac-Toe, ruang pencarian pada penelitian ini jauh lebih kompleks karena menggunakan papan 8×8 serta atribut HP/MP dan variasi aksi. Karena itu, optimasi Alpha-Beta pada penelitian ini memberikan dampak yang lebih nyata terhadap waktu komputasi, terutama pada depth yang lebih tinggi [3].

Tabel 3. Perbandingan studi terkait Minimax dan Alpha-Beta

Penelitian	Jenis game	Ukuran papan	Atribut/unit	Algoritma	Fokus evaluasi/tujuan	Temuan utama
Raut dkk. (2025) – <i>Enhancing AI Strategy in Checkers through Minimax and Alpha-Beta Pruning Techniques</i> [15]	Checkers	8×8	Bidak standar	Minimax + Alpha-Beta	Meningkatkan strategi & efisiensi keputusan	Alpha-Beta mempercepat keputusan dengan memangkas cabang pencarian
Jofanda & Yasin (2021) – <i>Design of Checkers Game Using Alpha-Beta Pruning Algorithm</i> [5]	Checkers	8×8	Bidak standar	Minimax + Alpha-Beta	Optimasi pencarian (reduksi node/cabang)	Alpha-Beta sebagai optimasi Minimax dapat mengurangi perluasan simpul/cabang sehingga lebih efisien
Putri dkk. (2024) – <i>Kecerdasan Buatan dalam Permainan Tic-Tac-Toe melalui Algoritma Minimax dengan Optimasi Alpha-Beta Pruning</i> [2]	Tic-Tac-Toe	3×3	Bidak X/O	Minimax vs Minimax + Alpha-Beta	Membandingkan efisiensi & kualitas keputusan AI	Alpha-Beta meningkatkan efisiensi dibanding minimax murni
Yiwei Han – <i>Application of Different Artificial Intelligence Methods on Reversi</i> [16]	Reversi	8x8	Bidak hitam/putih	Minimax + Alpha-Beta	Efisiensi pencarian & evaluasi posisi	Alpha-Beta meningkatkan efisiensi pencarian pada game turn-based berbasis papan
Lim B Dkk – <i>Connect-4 using Alpha-Beta pruning with minimax algorithm</i> [17]	Connect-4	7×6	Bidak standar	Minimax + Alpha-Beta	Efisiensi pemilihan langkah & pengurangan eksplorasi	Alpha-Beta mempercepat pengambilan keputusan dibanding Minimax murni pada game papan sederhana
Xie Y Dkk – <i>Research and Improvement of Gobang/Gomoku</i>	Gobang/Gomoku	15×15	Bidak standar	Alpha-Beta (improvement)	Optimasi pencarian heuristic	Perbaikan strategi & pencarian Alpha-Beta/heuristik

Penelitian	Jenis game	Ukuran papan	Atribut/unit	Algoritma	Fokus evaluasi/tujuan	Temuan utama
Alpha-Beta Search Algorithm in Gobang [18]						digunakan untuk menurunkan waktu pencarian pada game papan
Hermawan L Dkk – Penerapan Augmented Reality Berbasis Algoritma Minimax-Alpha Beta Pruning Pada Game Papan Cerdas 21 [19]		Smart board game (AR)	Berbasis papan	Minimax + Alpha-Beta	Implementasi AI pada aplikasi/game papan	Minimax + Alpha-Beta digunakan untuk mempercepat keputusan AI pada game papan berbasis aplikasi
Penelitian ini	TRPG berbasis papan	8×8	HP/MP, multi-aksi (move/attack/magic/element turn), terminal HP Raja	Minimax vs Minimax+Alpha-Beta	Menjaga keputusan taktis & responsif pada branching factor tinggi	Waktu eksekusi turun (\approx 2–3 detik vs >6 detik) dan eksplorasi state berkurang pada depth yang sama

Temuan efisiensi ini konsisten dengan prinsip umum Alpha-Beta Pruning yang mengurangi jumlah node yang dievaluasi tanpa mengubah keputusan pada kedalaman pencarian yang sama, sehingga waktu komputasi dapat dipercepat [1], [4], [6]. Dengan demikian, kontribusi penelitian ini terletak pada pembuktian empiris bahwa Minimax yang dioptimasi Alpha-Beta masih layak digunakan pada TRPG berbasis atribut, selama kedalaman pencarian diatur secara tepat untuk menjaga responsivitas pada perangkat komputasi terbatas [3].

4. KESIMPULAN

Penelitian ini menerapkan algoritma Minimax dan optimasinya melalui Alpha-Beta Pruning sebagai pengambil keputusan musuh (AI) pada permainan Tactical Role-Playing Game (TRPG) berbasis papan 8×8 dengan atribut HP/MP dan multi-aksi. Hasil pengujian menunjukkan bahwa Minimax tanpa pruning menghasilkan waktu eksekusi per langkah AI lebih dari 6 detik pada depth = 5 sehingga respons permainan melambat pada perangkat uji yang terbatas. Setelah dioptimasi dengan Alpha-Beta Pruning pada kondisi dan skenario yang identik, waktu eksekusi per langkah AI menurun menjadi sekitar 2–3 detik, disertai penurunan jumlah expanded nodes secara signifikan, yang mengindikasikan efektivitas pruning dalam mengurangi eksplorasi state yang tidak menjanjikan.

Selain itu, pengujian pada variasi depth menunjukkan bahwa peningkatan kedalaman pencarian berpengaruh langsung terhadap responsivitas permainan; depth yang lebih rendah cenderung responsif, sedangkan depth yang lebih tinggi meningkatkan beban komputasi dan berpotensi menimbulkan keterlambatan (freeze). Dengan demikian, Minimax yang dioptimasi Alpha-Beta Pruning dinilai layak digunakan pada TRPG berbasis atribut selama kedalaman pencarian diatur secara moderat agar tetap menjaga keseimbangan antara kualitas keputusan dan waktu respons AI.

5. SARAN

Penelitian selanjutnya disarankan untuk mengembangkan mekanisme depth adaptif agar kedalaman pencarian dapat menyesuaikan kondisi permainan (misalnya jumlah bidak tersisa atau tingkat ancaman terhadap Raja), sehingga keputusan tetap taktis tanpa mengorbankan responsivitas. Selain itu, performa Minimax + Alpha-Beta dapat ditingkatkan melalui move ordering dan heuristik pemangkasan tambahan agar pruning lebih efektif pada state dengan

branching factor tinggi. Pengembangan berikutnya juga dapat menambahkan lebih banyak skenario uji (early–mid–late game), memperluas metrik evaluasi (misalnya expanded nodes per aksi dan stabilitas waktu), serta melakukan optimasi implementasi (misalnya caching/transposition table) untuk menekan waktu komputasi pada perangkat dengan sumber daya terbatas.

DAFTAR PUSTAKA

- [1] Connor M. Pipan, “Application of The Monte-Carlo Tree Search To Multi-Action Turn-Based Games With Hidden Information,” AIR FORCE INSTITUTE OF TECHNOLOGY, 2021.
- [2] N. R. Putri *et al.*, “Kecerdasan Buatan dalam Permainan Tic-Tac-Toe Melalui Algoritma Minimax dengan Optimasi Alpha-Beta Pruning,” *Jurnal Minfo Polgan*, vol. 13, no. 1, 2024, doi: 10.33395/jmp.v13i1.13594.
- [3] M. Wu, M. Wicker, W. Ruan, X. Huang, and M. Kwiatkowska, “A game-based approximate verification of deep neural networks with provable guarantees,” *Theor Comput Sci*, vol. 807, pp. 298–329, Feb. 2020, doi: 10.1016/j.tcs.2019.05.046.
- [4] A. K. Trivedi, S. Garg, and N. Pandey, “Novel hardware/software co-design approach for Connect6 game-solver,” *e-Prime - Advances in Electrical Engineering, Electronics and Energy*, vol. 7, Mar. 2024, doi: 10.1016/j.prime.2023.100395.
- [5] A. N. W. Jofanda and M. Yasin, “Design of Checkers Game Using Alpha-Beta Pruning Algorithm,” *INTENSIF: Jurnal Ilmiah Penelitian dan Penerapan Teknologi Sistem Informasi*, vol. 5, no. 2, pp. 279–295, Aug. 2021, doi: 10.29407/intensif.v5i2.15863.
- [6] A. Nayak *et al.*, “Accelerating Decision-Making in AI: Parallelizing Monte Carlo Tree Search for Connect 4 Using CPU and GPU,” in *Procedia Computer Science*, Elsevier B.V., 2025, pp. 82–89. doi: 10.1016/j.procs.2025.07.011.
- [7] J. Gao, “The Computational Complexity of Fire Emblem Series and similar Tactical Role-Playing Games,” Sep. 2019, [Online]. Available: <http://arxiv.org/abs/1909.07816>
- [8] R. R. Hariadi, I. Kuswardayan, I. Ariesianti, and I. Alfi T.Z, “Penerapan Algoritma Alphabeta Pruning Sebagai Kecerdasan Buatan pada Game Pawn Battle,” *JURNAL INFOTEL*, vol. 9, no. 2, p. 185, May 2017, doi: 10.20895/infotel.v9i2.166.
- [9] S. Singhal and S. Muthukumarasamy, “Comparative study of performance of parallel alpha Beta Pruning for different architectures,” Dec. 2019, pp. 115–119. doi: 10.1109/IACC48062.2019.8971591.
- [10] C. C. Suancha, M. J. Suarez, and F. A. Besoain, “Implementation of Alpha-Beta Pruning and Transposition Tables on Checkers Game,” *IEEE Access*, vol. 12, pp. 46636–46645, 2024, doi: 10.1109/ACCESS.2024.3381958.
- [11] E. B. Baum and W. D. Smith², “A Bayesian approach to relevance in game playing I,” 1997.
- [12] A. Plaat, J. Schaeffer, W. Pijls, and A. De Bruin, “Artificial Intelligence Best-first fixed-depth minimax algorithms,” 1996.
- [13] R. E. Korf and D. M. Chickering, “Intelligence Best-first minimax search,” ELSEVIER, 1996.
- [14] Y. Xu, “Design of a Gomoku AI Based on the Alpha-Beta Pruning Search Algorithm,” *Applied and Computational Engineering*, vol. 94, no. 1, pp. 49–56, Nov. 2024, doi: 10.54254/2755-2721/94/2024melb0064.
- [15] R. Raut, M. Chaudhari, S. Raipalli, and S. Banginwar, “Enhancing AI Strategy in Checkers through Minimax and Alpha-Beta Pruning Techniques,” Dec. 2025, pp. 1–6. doi: 10.1109/SETCOM64758.2025.10932634.
- [16] Y. Han, “Application of Different Artificial Intelligence Methods on Reversi,” 2023.

- [17] B. Lim, G. San, Z. Arabee, A. Salam, Y. J. Xin, and C. K. Dhanoa, "Connect-4 using Alpha-Beta pruning with minimax algorithm," 2022.
- [18] Y. Xie, W. Gao, Z. Dai, and Y. Li, "Research and Improvement of Alpha-Beta Search Algorithm in Gobang," in *Advances in Transdisciplinary Engineering*, IOS Press BV, Feb. 2022, pp. 819–829. doi: 10.3233/ATDE220084.
- [19] L. Hermawan and M. B. Ismiati, "Penerapan Augmented Reality Berbasis Algoritma Minimax-Alpha Beta Prunning Pada Game Papan Cerdas 21," 2022.