SPIRIT VOL 17 NO 2 (2025) 107 - 116
JURNAL SPIRIT
0173/C3/DT.05.00/2025

e-ISSN 2721-057X
p-ISSN 2085-3092

PEMODELAN LANSKAP REALISTIK DAN INTERAKTIF DENGAN METODE PERLIN NOISE PADA LINGKUNGAN VIRTUAL 3D

Sigit Riyadi[1], Salman Al Farizi[2]
[1,2]Sistem dan Teknologi Informasi, ITB Yadika Pasuruan, Pasuruan,Indonesia
sigitriyadi@stmik-yadika.ac.id[1],salmanalfariziii861@gmail.com[2]

Naskah diterima: 09 Nopember 2025 ; Direvisi : 16 Nopember 2025 ; Disetujui : 30 Nopember 2025

Abstrak

Pengembangan dan evaluasi kinerja sebuah aplikasi generator terrain 3D banyak menggunakan algoritma Perlin Noise pada platform Android. Aplikasi dirancang untuk menghasilkan lanskap virtual dengan berbagai tingkat kompleksitas, yang kemudian diuji pada empat perangkat Android dengan spesifikasi hardware yang berbeda. Pengujian dilakukan untuk mengevaluasi performa aplikasi dalam hal frame rate (FPS), waktu render, penggunaan memori, dan jumlah objek yang ditempatkan (spawned objects). Hasil pengujian menunjukkan bahwa spesifikasi perangkat, terutama kapasitas RAM, memiliki pengaruh signifikan terhadap kinerja aplikasi. Perangkat dengan RAM yang lebih besar menunjukkan performa yang lebih baik, dengan frame rate yang lebih tinggi dan waktu render yang lebih cepat, sementara perangkat dengan spesifikasi rendah mengalami penurunan performa. Meskipun penggunaan memori aplikasi terbukti efisien dan konsisten di semua perangkat yang diuji, hasil ini menyoroti perlunya optimasi lebih lanjut untuk memastikan aplikasi dapat berjalan dengan lancar pada perangkat dengan spesifikasi rendah. Penelitian ini menunjukan potensi optimasi dalam pengembangan aplikasi terrain generator di platform Android, serta langkah-langkah perbaikan untuk meningkatkan kinerja dan pengalaman pengguna di berbagai perangkat.

Kata kunci: *Generator Terrain 3D, Algoritma Perlin Noise, Platform Android,, Optimasi Perangkat, Spesifikasi Hardware, Evaluasi Performa.*

Abstract

Development and performance evaluation of a 3D terrain generator application that heavily utilizes the Perlin Noise algorithm on the Android platform. The application was designed to generate virtual landscapes of varying complexity and was then tested on four Android devices with different hardware specifications. Testing was conducted to evaluate application performance in terms of frame rate (FPS), rendering time, memory usage, and the number of spawned objects. The test results showed that device specifications, particularly RAM capacity, significantly impacted application performance. Devices with more RAM demonstrated better performance, with higher frame rates and faster rendering times, while devices with lower specifications experienced decreased performance. Although the application's memory usage proved efficient and consistent across all tested devices, these results highlight the need for further optimization to ensure the application runs smoothly on lower-spec devices. This study demonstrates potential optimizations in developing terrain generator applications on the Android platform, as well as suggested improvements to enhance performance and user experience across devices.

Keywords: *3D Terrain Generator, Perlin Noise Algorithm, Android Platform, Device Optimization, Hardware Specifications, Performance Evaluation.*

PENDAHULUAN

Perkembangan industri game dan simulasi virtual telah memacu kebutuhan akan teknologi yang dapat menciptakan lingkungan virtual yang realistis dan interaktif. Salah satu teknik yang banyak digunakan dalam menciptakan lanskap tersebut adalah Procedural Content Generation (PCG), khususnya dengan memanfaatkan algoritma Perlin Noise[15].

Algoritma Perlin Noise menjadi perangkat utama untuk menghasilkan noise pada permukaan peta, menciptakan efek alami yang menyeluruh dalam konteks game 3D[15]. Hasil penelitian mereka menunjukkan bahwa implementasi teknik generasi peta dan pemunculan objek secara acak dapat meningkatkan kualitas gameplay serta memberikan pengalaman bermain yang lebih mendalam dan memikat bagi pengguna. Implementasi PCG dengan memanfaatkan algoritma Perlin Noise untuk membangun kerajaan dalam konteks game[6]. Penelitian ini memberikan pemahaman mendalam tentang kemampuan algoritma Perlin Noise dalam menciptakan lingkungan virtual yang realistis dan menarik.

Sejalan dengan penelitian-penelitian tersebut, penelitian ini bertujuan untuk mendalami dan mengembangkan penerapan teknik PCG dengan fokus utama pada pembangunan lanskap realistik dan interaktif dalam lingkungan virtual 3D berbasis Unity untuk platform Android. Penelitian ini akan melibatkan eksplorasi yang lebih komprehensif terhadap algoritma Perlin Noise guna menciptakan lingkungan game yang lebih dinamis dan mendalam. Dengan pendekatan ini, diharapkan dapat meningkatkan kualitas gameplay serta memberikan pengalaman bermain yang lebih memikat bagi pengguna[4].

Tujuan dari penelitian ini adalah untuk mengembangkan lanskap virtual yang realistis dan interaktif menggunakan algoritma Perlin Noise. Secara lebih spesifik, penelitian ini bertujuan untuk :

1. Mengimplementasikan algoritma Perlin Noise dalam menciptakan lanskap 3D yang beragam dan alami.
2. Mengoptimalkan aplikasi agar memiliki performa yang baik dan kompatibel dengan perangkat Android.
3. Menguji dan mengevaluasi lanskap yang dihasilkan dari segi visual dan performa untuk memastikan kualitas dan efisiensi.

METODE

Pada bagian ini, peneliti akan menjelaskan secara rinci desain percobaan, peralatan yang digunakan, metode pengumpulan data, serta tahapan-tahapan kegiatan yang dilakukan selama penelitian ini. Penelitian ini bertujuan untuk mengembangkan pemodelan landskap realistik dan interaktif dengan metode perlin noise pada lingkungan virtual 3d menggunakan unity berbasis android.

1. Perlin Noise

Perlin Noise memiliki variasi yang diperlukan dalam simulasi lanskap, sehingga sangat berguna dalam aplikasi seperti pengembangan game dan simulasi lingkungan menekankan pentingnya parameterisasi seperti skala, oktaf, dan lacunarity untuk mengontrol tingkat detail dan keragaman dari lanskap yang dihasilkan, memastikan bahwa setiap generasi lanskap tetap unik dan menarik[3].

Perlin Noise menghasilkan noise yang tidak berulang dengan menggunakan interpolasi smooth antara titik-titik acak dalam grid. Perlin Noise memiliki beberapa parameter yang dapat disesuaikan untuk mengontrol hasil akhirnya, termasuk skala, frekuensi, amplitude, dan persistence. Skala menentukan seberapa besar atau kecil detail dari noise yang dihasilkan, sementara frekuensi mengatur seberapa sering pola noise berulang dalam ruang tertentu. Amplitude menentukan seberapa besar variasi nilai noise, dan persistence mengontrol seberapa cepat amplitude berkurang dengan setiap octave tambahan. Dengan menyesuaikan parameter-parameter ini, pengembang dapat menciptakan berbagai jenis lanskap dan tekstur yang berbeda, mulai dari pegunungan yang tinggi hingga dataran yang datar.

Dalam pengembangan game, Perlin Noise sering digunakan untuk menciptakan medan 3D dengan mengubah nilai noise menjadi ketinggian terrain. Penggunaan Perlin Noise dalam Unity memungkinkan penciptaan lanskap yang dinamis dan interaktif, di mana setiap sesi permainan dapat memiliki peta yang berbeda[13]. Hal ini tidak hanya meningkatkan replayability dari game tetapi juga membuat pemain merasa bahwa dunia game tersebut hidup dan terus berubah. Dengan kemampuan untuk menghasilkan detail yang tinggi dan variasi yang besar, Perlin Noise menjadi alat yang sangat berguna dalam pengembangan game dan aplikasi realitas virtual.

2. **Procedural Content Generation (PCG**)

Procedural Content Generation (PCG) adalah teknik yang sangat esensial dalam pengembangan konten digital, terutama dalam industri game. Teknik ini memungkinkan pengembang untuk menciptakan konten secara otomatis melalui algoritma, sehingga mengurangi kebutuhan akan intervensi manual yang intensif[3]. Dalam konteks pengembangan game, PCG digunakan untuk menghasilkan berbagai elemen game seperti peta, karakter, objek, dan bahkan cerita secara acak dan dinamis, sehingga menciptakan pengalaman yang selalu berbeda bagi setiap pemain. Dengan PCG, pengembang dapat menghemat waktu dan sumber daya, terutama dalam proyek besar yang memerlukan pembuatan konten dalam jumlah besar.

Salah satu tantangan utama adalah memastikan bahwa konten yang dihasilkan oleh PCG tetap koheren dan masuk akal. Tanpa kontrol yang tepat, algoritma PCG bisa menghasilkan konten yang tidak relevan atau tidak sesuai dengan narasi atau desain game. Oleh karena itu, pengembang perlu

menetapkan aturan dan parameter yang ketat untuk memastikan bahwa konten yang dihasilkan oleh PCG tetap sesuai dengan tujuan desain game.

Penyesuaian parameter seperti skala dan frekuensi Perlin Noise sangat penting untuk memastikan keseimbangan antara kualitas visual dan performa sistem. Dalam pengembangan game mobile, misalnya, pengembang harus mempertimbangkan keterbatasan hardware perangkat, seperti memori dan kemampuan grafis, untuk memastikan bahwa game dapat berjalan dengan lancar di berbagai jenis perangkat. Dengan mengoptimalkan algoritma PCG, pengembang dapat menciptakan dunia yang menarik dan dinamis, tanpa mengorbankan performa game.

3. Kebutuhan Sistem

Proses identifikasi kebutuhan sistem untuk menentukan elemen-elemen utama dalam jalannya proses pemodelan landskap realistis. Tahap ini mencakup:

- **Spesifikasi Fungsional**

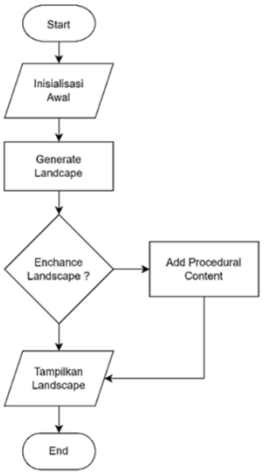
  Berikut adalah detail spesifikasi fungsional yang diharapkan diperlukan untuk mengembangkan model landskap ini yaitu :

  - Assets Konten : Mencakup pohon-pohonan, rumput, dan lain lain yang dihasilkan secara acak yang didapatkan dari Unity assets free.
  - Spesifikasi Perangkat

    Spesifikasi perangkat menetapkan persyaratan yang tidak langsung terkait dengan kinerja, efisiensi, dan pengalaman pengguna. Berikut adalah detail spesifikasi perangkat yang relevan untuk aplikasi ini :

    - RAM 6 GB: Memori ini penting untuk mengelola grafis yang kompleks dan operasi algoritma Perlin Noise yang intensif.
    - Prosesor 2.0 GHz: Kecepatan ini memastikan bahwa semua operasi, termasuk rendering grafis dan perhitungan algoritma, dilakukan dengan cepat dan efisien.
    - Ruang Penyimpanan Internal 10 GB: Ruang ini diperlukan untuk menyimpan semua aset, seperti model 3D, tekstur, dan data procedural lainnya.
    - Kompatibilitas dengan Perangkat Android : Versi Android: Aplikasi harus kompatibel dengan setidaknya versi Android 11.0 (Red Velvet Cake) dan yang lebih baru, memastikan cakupan perangkat yang luas.

4. Desain Sistem Flowchart



Gambar 1. Flowchart Proses Lanskap

Flowchart ini merupakan proses pembuatan lanskap virtual 3D yang realistis dan dinamis menggunakan kombinasi algoritma Perlin Noise dan teknik Procedural Content Generation (PCG) :

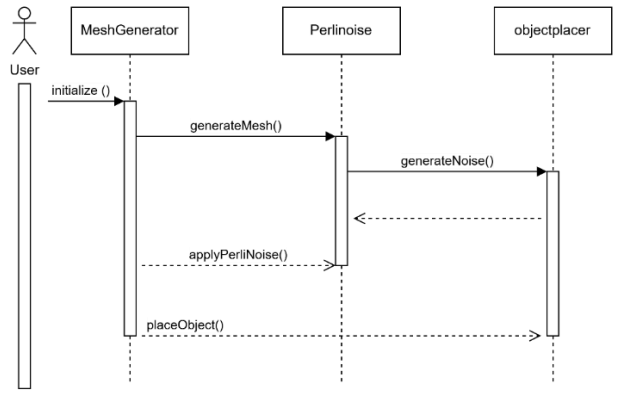
1. Start
   - Menandai awal eksekusi program. Program dimulai dan siap untuk melakukan inisialisasi komponen dan sumber daya yang diperlukan.
   - Tahap ini mempersiapkan program untuk menjalankan proses pembuatan lanskap dengan memuat modul, library, dan elemen lain yang dibutuhkan.
2. Initialize Program:
   - Program melakukan inisialisasi berbagai komponen penting untuk generasi lanskap.
   - MeshGenerator: Komponen ini diinisialisasi untuk mengelola dan menciptakan mesh dasar terrain.
   - Parameter Lanskap: Parameter-parameter yang akan memengaruhi bentuk terrain, seperti ukuran grid, resolusi, skala, oktaf Perlin Noise, frequency, amplitude, dan lacunarity, didefinisikan pada tahap ini
   - Lingkungan Kerja: Program menyiapkan lingkungan kerja 3D, termasuk pengaturan skala dunia virtual, sistem koordinat, dan elemen dasar lainnya.
3. Generate Landscape Using Perlin Noise:
   - Ini adalah inti dari proses generasi lanskap. Algoritma Perlin Noise digunakan untuk menciptakan variasi ketinggian pada terrain secara prosedural.
   - Pola Noise Alami: Algoritma Perlin Noise menghasilkan pola noise yang halus dan alami. Pola ini diterapkan pada mesh terrain untuk membentuk pegunungan, lembah, bukit, dan fitur geografis lainnya.
   - Pengaturan Parameter: Parameter Perlin Noise (octave, frequency, amplitude, lacunarity) disesuaikan untuk menghasilkan terrain dengan bentuk dan detail yang bervariasi.
4. Enhance Landscape with Procedural Content (Optional):
   - Langkah ini bersifat opsional tetapi sangat penting untuk meningkatkan realisme dan detail lanskap.
   - PCG (Procedural Content Generation): Setelah terrain dasar terbentuk, elemen-elemen lingkungan seperti pohon, bebatuan, dan tumbuhan.
   - Aturan Penempatan: PCG menggunakan aturan-aturan tertentu untuk mendistribusikan objek. Aturan ini dapat didasarkan pada ketinggian terrain, kemiringan, jenis permukaan, atau noise map untuk menciptakan pola distribusi yang tampak alami.
5. Finalize and Visualize Landscape::
   - Pada tahap ini, proses generasi lanskap diselesaikan, dan lanskap final divisualisasikan.

- Penyelesaian Mesh: Mesh terrain yang telah dibentuk dan dimodifikasi dengan Perlin Noise dan PCG diselesaikan.
- Pemetaan Warna: Warna dan tekstur diaplikasikan pada terrain, menciptakan variasi visual yang realistis.
- Penyesuaian Tambahan: Penyesuaian akhir, seperti pencahayaan dan efek visual lainnya, dapat ditambahkan untuk meningkatkan realisme dan estetika lanskap.

6. Finalize and Visualize Landscape::
   - Menandai akhir dari proses generasi lanskap.
   - Lanskap yang dihasilkan siap untuk digunakan dalam game, simulasi, atau visualisasi.

Langkah-langkah ini mendeskripsikan alur kerja dari proses generasi lanskap procedural yang dimulai dari inisialisasi hingga visualisasi hasil akhir.

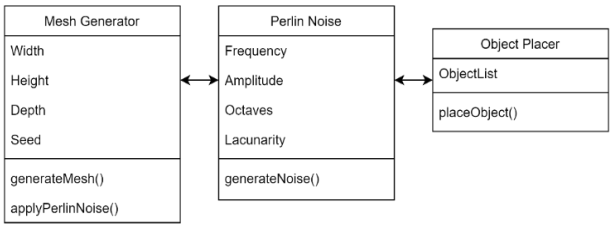
Sequence Diagram



Gambar 2. Sequence Diagram inisialisasi hingga visualisasi

1. Langkah ini bertujuan untuk mempersiapkan MeshGenerator bagi tahap-tahap berikutnya, memastikan bahwa semua parameter awal dan struktur dasar siap untuk digunakan.
2. Selanjutnya, MeshGenerator berinteraksi dengan PerlinNoise untuk mendapatkan nilai noise bagi setiap titik dalam mesh melalui metode generateNoise(x, y). Nilai-nilai ini memberikan efek alami pada mesh, yang penting untuk menciptakan lanskap yang realistis.
3. Setelah mesh dasar terbentuk, MeshGenerator kemudian meminta ObjectPlacer untuk menambahkan objek ke dalam lanskap melalui metode placeObject(). ObjectPlacer menempatkan objek seperti pohon atau batu secara procedural, berdasarkan nilai noise dan aturan tertentu yang memastikan penempatan objek yang alami dan tidak berulang.

Class Diagram



Gambar 3. Class Diagram

1. MeshGenerator bertanggung jawab untuk menghasilkan mesh dasar dari lanskap dengan atribut seperti width, height, depth, dan seed untuk menentukan dimensi dan variasi lanskap. Metode utama dalam kelas ini adalah generateMesh() yang digunakan untuk membuat mesh, dan applyPerlinNoise()

yang menerapkan algoritma Perlin Noise pada mesh untuk memberikan tekstur yang realistis dan alami.

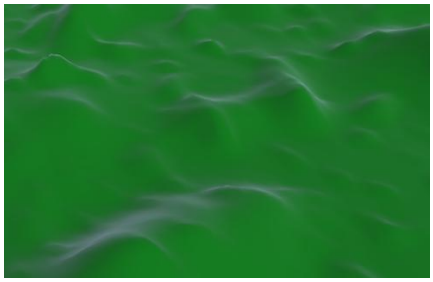
Kelas MeshGenerator ini merupakan inti dari sistem yang bertanggung jawab untuk mengelola dan mengkoordinasikan pembuatan terrain yang dasar, sebelum elemen-elemen lain ditambahkan ke dalamnya.

2. Metode generateNoise(x, y) menghasilkan nilai noise untuk koordinat tertentu, yang kemudian digunakan oleh MeshGenerator untuk menciptakan variasi dalam lanskap.
3. Kelas ObjectPlacer bertugas untuk menambahkan elemen-elemen seperti pepohonan, bebatuan, dan objek-objek lain ke dalam lanskap.

HASIL DAN PEMBAHASAN

Implementasi menunjukkan bahwa sistem yang dikembangkan, dengan memanfaatkan perangkat lunak yang telah ditentukan, dapat berfungsi dengan baik sesuai dengan kebutuhan dan spesifikasi. berikut adalah hasil penelitian ini :

A. Generasi Terrain dengan Perlin Noise

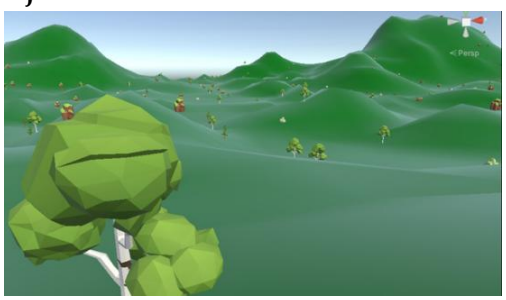


Gambar 4. Visualisasi Heightmap dan Terrain 3D yang Dihasilkan

Algoritma Perlin Noise diimplementasikan untuk menghasilkan heightmap 2D yang merepresentasikan ketinggian terrain. Parameter Perlin Noise, yaitu frequency, lacunarity, dan persistence, dikonfigurasi untuk mengontrol karakteristik terrain. Nilai frequency yang lebih tinggi menghasilkan terrain dengan fitur yang lebih rapat dan detail yang lebih halus. Lacunarity mengontrol perbedaan skala antara oktaf noise, menghasilkan variasi yang lebih kasar atau halus pada terrain. Persistence memengaruhi seberapa kuat detail pada skala yang lebih kecil dipertahankan, berkontribusi pada smoothness keseluruhan terrain.

Heightmap 2D yang dihasilkan kemudian digunakan untuk membentuk mesh terrain 3D. Setiap titik pada heightmap dipetakan ke vertex pada mesh, dengan ketinggian vertex sesuai dengan nilai pada heightmap. Penerapan tekstur dan material berdasarkan ketinggian (height-based texturing) memberikan tampilan visual yang lebih realistis, dengan tekstur yang berbeda diterapkan pada ketinggian yang berbeda, misalnya rumput di dataran rendah dan salju di puncak gunung.

B. Penerapan Teknik PCG untuk Penempatan Objek



Gambar 5. Visualisasi Penempatan Objek pada Terrain

Teknik PCG diimplementasikan untuk mendistribusikan objek, seperti pohon dan bebatuan, secara prosedural pada terrain yang dihasilkan. Fungsi MapEmbellishments() dalam skrip MeshGenerator bertanggung jawab atas proses ini. Fungsi ini mengiterasi setiap vertex pada mesh terrain, mentransformasikan posisinya ke koordinat dunia, dan memeriksa kemiringan serta ketinggian terrain pada titik tersebut.Probabilitas penempatan juga digunakan untuk mengontrol kepadatan objek pada terrain. Dengan mengatur probabilitas, objek dapat tersebar secara acak namun tetap terkontrol, menghasilkan distribusi yang lebih alami. Seed acak (seed) digunakan untuk mengontrol keacakan dalam proses PCG, memungkinkan generasi lanskap yang berbeda-beda namun tetap konsisten dengan seed yang sama.

Evaluasi Kinerja pada Platform Android

Tabel 1. Evaluasi Kinerja

No.	Perangkat	Poco X6 Pro 5G	Vivo V15	Realme Narzo 50A	Samsung Galaxy A14 5G
1.	RAM	12 GB	6 GB	4 GB	8 GB
2.	Vertices	10201	10201	10201	10201
3.	Triangles	60000	60000	60000	60000
4.	Colors	10201	10201	10201	10201
5.	Spawned Objects	1525	1559	1634	1512
6.	Render Time (ms)	4581	13054	11050	8825
7.	Memory Usage (MB)	49.4	50.0	50.1	49.0
8.	FPS	30	26	18	30

Metrik performa yang diukur meliputi frame rate (FPS), waktu render (dalam milidetik), dan penggunaan memori (dalam megabyte). Pengujian ini dilakukan untuk mengetahui bagaimana aplikasi berperforma pada perangkat dengan kemampuan pemrosesan yang berbeda-beda dan mengidentifikasi potensi kendala performa.

C. Evaluasi Black Box

Tabel 2. Evaluasi Kinerja

Aspek yang Diuji	Hasil Pengujian	Deskripsi
Collision Detection	**Berhasil**	Objek bertabrakan dengan *terrain* dan tidak menembus permukaan.
Pergerakan di Atas *Terrain*	**Berhasil**	Objek dapat bergerak dengan lancar di atas *terrain*, mengikuti kontur ketinggian.
Penempatan Objek PCG	**Berhasil**	Objek yang ditempatkan secara prosedural (pohon, batu, dll.) ter-generate dengan benar dan berada pada posisi yang tidak tenggelam di dalam tanah.
Keberhasilan Generasi	**Berhasil**	Aplikasi dapat meng-generate *terrain* dengan berbagai parameter *Perlin Noise* dan PCG tanpa mengalami kesalahan atau crash.

Pengujian dilakukan untuk memvalidasi fungsionalitas aplikasi generator terrain. Aspek-aspek yang diuji meliputi: collision detection antara objek dan terrain, pergerakan objek di atas terrain, penempatan objek PCG, dan keberhasilan generasi terrain. Selain itu, untuk memvalidasi fungsionalitas aplikasi generator terrain parameter yang didapatkan yaitu: 1. Pengaruh RAM terhadap performa

mencapai frame rate stabil di 30 FPS, 2. Konsistensi Penggunaan Memori antara 49 MB hingga 50.1 MB, dan 3. Perangkat Low-End minimal frame rate terendah yaitu 18 FPS.

PENUTUP

Penelitian ini mengimplementasikan algoritma Perlin Noise dan teknik PCG untuk menghasilkan lanskap virtual 3D yang realistis dan interaktif pada platform Android. Perlin Noise memungkinkan penciptaan terrain dengan variasi ketinggian dan detail visual yang menarik, mensimulasikan bentang alam dunia nyata. PCG, melalui fungsi MapEmbellishments(), mendistribusikan objek seperti pohon dan bebatuan secara prosedural, meningkatkan keragaman dan realisme lanskap.

Evaluasi kinerja menunjukkan bahwa aplikasi berjalan dengan baik pada perangkat dengan spesifikasi memadai, untuk memvalidasi fungsionalitas aplikasi generator terrain. Aspek-aspek yang diuji meliputi: collision detection antara objek dan terrain, pergerakan objek di atas terrain, penempatan objek PCG, dan keberhasilan generasi terrain. Selain itu, untuk memvalidasi fungsionalitas aplikasi generator terrain parameter yang dapatkan yaitu: 1. Pengaruh RAM terhadap performa mencapai frame rate stabil di 30 FPS, 2. Konsistensi Penggunaan Memori antara 49 MB hingga 50.1 MB, dan 3. Perangkat Low-End minimal frame rate terendah yaitu18 FPS.

DAFTAR PUSTAKA

[1] De Pontes, R. G., Gomes, H. M., & Seabra, I. S. R. (2022). Particle swarm optimization for Procedural Content Generation in an endless platform game. Entertainment Computing, 43, 100496.

[2] De Pontes, R. G., Oliveira, T., & Silva, M. (2022). Optimizing Procedural Content Generation for Android Games: Balancing Performance and Visual Quality. Journal of Mobile Game Development, 10(1), 45-61.

[3] Etherington, T. R. (2022). Perlin noise as a hierarchical neutral landscape model. Web Ecology, 22(1), 1-6.

[4] Fernández Galeote, D., & Hamari, J. (2021). Game-based climate change engagement: analyzing the potential of entertainment and serious games. Proceedings of the ACM on Human-Computer Interaction, 5(CHI PLAY), 1-21.

[5] Fernández, L., & Smiley, R. (2023). Advanced Terrain Generation Techniques for Game Development. Journal of Visual Computing.

[6] Gonzalez, T., & Williams, S. (2024). The Importance of Procedural Terrain Generation in Modern Game Development. Journal of Game Technology.

[7] Haryanto, M. (2024). Integrasi Sistem Kontrol Versi dengan Visual Studio Code. Jurnal Teknologi Informasi, 18(1), 22-30.

[8] Hart, J., & Kuck, C. (2020). Procedural

Generation: An Overview of Techniques and Applications. Computer Graphics Journal.

[9] Irawan, A., Nugroho, F., & Prasetya, R. (2022). Pengembangan Game dengan Unity: Tantangan dan Peluang. Jurnal Teknologi Informasi, 14(2), 89-105.

[10] Jones, M., & Taylor, S. (2020). Physics Simulations in Unity: Creating Realistic Interactions in Games. Journal of Visual Computing.

[11] Klaus, K., & Moore, J. (2022). Noise-Based Terrain Generation: Techniques and Challenges. International Journal of Computer Graphics.

[12] Klein, R. (2022). The Asset Store: Leveraging Ready-Made Assets for Game Development in Unity. Journal of Game Technology.

[13] Le, T. (2023). Procedural Terrain Generation Using Perlin Noise.

[14] Liu, J., Snodgrass, S., Khalifa, A., Risi, S., Yannakakis, G. N., & Togelius, J. (2021). Deep learning for Procedural Content Generation. Neural Computing and Applications, 33(1), 19-37.

[15] Liu, J., Wang, H., & Zhang, T. (2021). Procedural Content Generation in Game Development: Enhancing Replayability and Personalization. Journal of Game Design and Development, 12(3), 59-74.

[16] Lugata, D. B. F., Selviana, R., & Alimin (2023). Implementasi Generate Map dan Pemunculan Objek secara Acak pada Game 3D menggunakan Bahasa C# dan Metode Perlin Noise di Unity: Studi Kasus pada Game Development. SPIRIT, 15(1).

[17] Macklin, A. (2021). Creating Realistic 3D Terrain in Game Development. Journal of Game Design.

[18] Martinez, P., & Wang, T. (2024). The Importance of Community in Unity Game Development. Journal of Game Innovation.

[19] Suparman, S. (2021). Optimasi Performa Game pada Platform Android dengan Menggunakan Android NDK. Jurnal Teknologi Informasi dan Multimedia, 13(2), 123-134.

[20] Riyadi, S. (2023). APLIKASI PC SMART ORDER by CUSTOM PADA TOKO KOMPUTER BERBASIS ANDROID MENGGUNAKAN METODE LINEAR SEQUENTIAL SEARCH. SPIRIT, 14(2).

[21] Watkins, R. (2016). Procedural Content Generation for Unity Game Development. Packt Publishing Ltd.

[22] Watkins, R. (2016). Procedural Content Generation in Unity: Techniques and Applications. Computer Graphics Journal.

[23] Zhao, H., & Liu, J. (2021). Dynamic Terrain Generation: A New Approach to Interactive Environments. Journal of Interactive Media