BAB I
PENDAHULUAN

1.1 Latar Belakang

Algoritma *Perlin-Noise* menjadi landasan utama dalam pengembangan *video game* dan teknik *rendering* secara *real–time*, dikarenakan algoritma tersebut dikenal akan kemampuannya untuk menghasilkan konten yang berpola dan acak, khususnya dalam generasi objek secara prosedural. Seperti yang telah ditetapkan oleh Lagae et al. [1], *Perlin-Noise* memainkan peran penting dalam generasi konten secara prosedural. Eastman et al. [2] lebih lanjut menunjukkan aplikasi praktisnya dalam generasi prosedural 2D berdasarkan *Perlin-Noise*. Signifikansi *Perlin-Noise* tidak hanya terletak pada keluwesannya tetapi juga pada potensinya untuk meningkatkan kualitas dan variasi lingkungan *video game* yang digenerasikan secara prosedural.

Dari beberapa studi yang ditemukan sebelumnya, McEwan et al. [3] mengeksplorasi optimasi generasi konten secara prosedural, menekankan pentingnya algoritma yang efisien. Maung et al. [4] memberikan wawasan tentang generasi tekstur dan penataan prosedural dengan menggunakan *Perlin-Noise* dan penelitian ini berkontribusi untuk menghindari repetisi (repetition) dalam tekstur. Togelius et al. [5][6] memperkenalkan generasi konten prosedural komposisional, menunjukkan signifikansi lebih luas *Perlin-Noise* dalam meningkatkan generasi konten secara prosedural. Korn et al. [7] menyoroti efektivitas algoritma *Perlin-Noise* dalam generasi konten prosedural untuk grafik *video game*, penelitian yang dilakukan menjelaskan tentang sebuah *video-game* dengan generasi secara prosedural dapat memberikan hasil bahwa mayoritas pemain saat melakukan survey pada penelitian menyatakan bahwa mereka lebih memilih game dengan PCG dengan suara 60% dibandingkan secara manual yang berisikan 40%.

Penelitian Shen [8] tentang generasi prosedural dalam *video game* menunjukkan aplikabilitas algoritma *Perlin-Noise*, mengidentifikasinya sebagai komponen utama dalam generasi. Frank, et al. [9], Ramadhan, et al. [10], serta Azzmi, et al. [11] menjelajahi implementasi generasi secara prosedural untuk berbagai hal seperti generasi sebuah kota, eksplorasi dunia *video game sandbox* yang terfokus pada generasi *terrain*. masing-masing, mengilustrasikan berbagai implementasi aplikasi *Perlin-Noise*. Selviana, et al. [12] melanjutkan penelitian sebelumnya dengan menekankan potensi percepatan proses pengembangan *video game* melalui generasi konten otomatis menggunakan *Perlin-Noise*.

Kesenjangan penelitian terdapat dalam lingkup generasi objek secara prosedural dalam pengembangan video game, khususnya dalam konteks penggunaan *Perlin-Noise*. Seperti yang dilakukan dalam penelitian-penelitian sebelumnya, telah dijelaskan bahwa objek yang dihasilkan secara prosedural masih tanpa menggunakan proses *pre-calculated* yang memakan performa tinggi, sepenuhnya acak tanpa aturan layer ataupun terjadi *overlapping*. Terdapat kekurangan penelitian yang dihadapi oleh pengembang video game yang bertujuan untuk mengaplikasikan *Perlin-Noise* tersebut untuk generasi objek prosedural berbasis Unity Engine.

Oleh karena itu, penelitian ini akan mencoba mengimplementasikan algoritma *Perlin-Noise* dalam *Unity Engine* dengan *layer-based* dan *Overlapping detection* menggunakan Algoritma *Euclidean Distance,* Pada buku yang ditulis oleh J.Tabak [13], dinyatakan algoritma *Euclidean Distance* dapat digunakan sebagai perhitungan jarak antar titik, dan algoritma ini dapat dipakai sebagai dasar untuk deteksi *overlapping*. Penelitian ini akan menutupi kesenjangan penelitian, mengeksplorasi cara dalam implementasi, menganalisis waktu penggunaan, menganalisis akurasi, dan menganalisis performa. Pengembangan ini bertujuan

memberikan pengembang *video game* sebuah algoritma yang sesuai dan akurat sesuai dalam implementasi video game untuk menghasilkan berbagai objek secara acak dalam *video game.*

1.2 Rumusan Masalah

Berikut ini merupakan rumusan masalah dalam penelitian :

1. Bagaimana performa dan daya yang digunakan pada perangkat keras saat melakukan generasi objek?
2. Seberapa tinggi tingkat akurasi layer dan objek yang terjadi *overlapping* pada tahap implementasi?
3. Bagaimana waktu yang diperlukan saat melakukan generasi objek dalam satuan waktu kerja?

1.3 Maksud dan Tujuan

Maksud utama dari penelitian ini adalah untuk mengatasi kesenjangan yang ada dalam penelitian kuantitatif sebelumnya untuk mengintegrasikan dan mengimplementasikan algoritma *Perlin-Noise* dalam Unity Engine dengan *layer-based* dan overlapping detection.

Berikut ini adalah tujuan dari penelitian yang dilakukan :

1. Mengevaluasi, menganalisis, serta membandingkan performa serta daya yang digunakan perangkat keras saat melakukan generasi.
2. Mengevaluasi, menganalisis, serta meningkatkan tingkat akurasi yang dihasilkan dari implementasi..
3. Mengevaluasi, menganalisis, serta membandingkan waktu kerja implementasi dalam generasi objek.

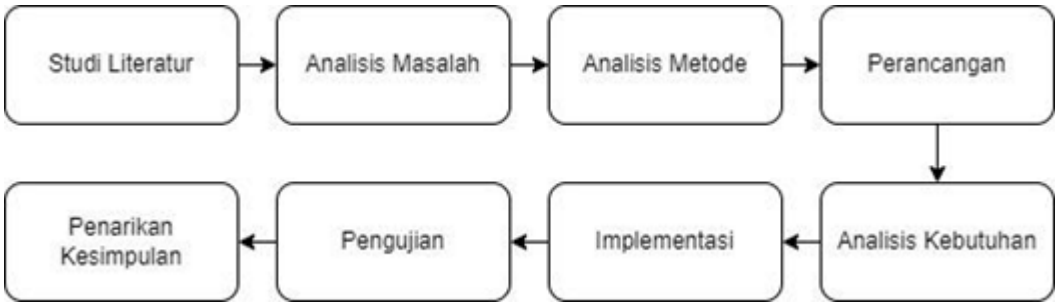
1.4 Batasan Masalah

Berikut adalah beberapa keterbatasan yang melekat dalam penelitian ini:

1. Batasan Lingkup dan Pendekatan yang Berpusat pada Unity versi 2022.3

2. Pengujian performa difokuskan dalam konteks *video game*.
3. Bahasa pemrograman yang digunakan yaitu C#.
4. Menggunakan *Euclidean-distance* pada *overlapping detection*.
5. Objek yang akan di generasi memiliki ukuran yang sama persis agar tidak terjadi perbedaan yang variatif saat pengujian

1.5 Metodologi Penelitian



Gambar 1.1
Metodologi Penelitian

Penelitian ini menggunakan metode kuantitatif untuk implementasi algoritma *Perlin-Noise* pada *Unity* untuk generasi objek secara prosedural dengan *Layer-Based* dan *overlapping detection* dengan algoritma *Euclidean Distance*.

1.5.1 Studi Literatur

Studi literatur merupakan teknik untuk mengumpulkan data dengan melakukan pencarian dan analisis pengetahuan dari beberapa jurnal, buku, paper, dokumentasi, dan aspek lainnya untuk mendapatkan teori mengenai implementasi *perlin-noise* pada *procedural object generation*, implementasi *Layer-Based* dan overlapping detection

1.5.2 Analisis Masalah

Analisis masalah dilakukan melalui pencarian isu-isu masalah yang sudah ada pada penelitian sebelumnya, serta merumuskan dan

merencanakan tindakan untuk tahap selanjutnya terkait implementasi *procedural object generation*. Berdasarkan masalah yang didapatkan pada latar belakang diantaranya masalah pada generasi objek tanpa adanya *pre-calculated,* masalah pada generasi yang tanpa layer, dan masih adanya *overlapping*.

1.5.3 Analisis Metode

Analisis metode merupakan proses untuk mencoba, mengobservasi, dan menganalisis metode-metode yang telah ditemukan dari studi literatur sebelumnya. Hal ini mencakup analisis matematis dan pemanfaatan pseudocode untuk memanfaatkan *perlin-noise* pada *Procedural Object Generation, Layer-Based,* dan mencari cara untuk deteksi *overlapping*

1.5.4 Perancangan

Berfokus pada perancangan yang harus dilakukan dalam fase implementasi dengan menggunakan gambaran umum sistem berdasarkan persyaratan temuan kuantitatif sebelumnya. Terfokus pada penentuan cara membuat algoritma dengan pseudocode untuk generasi objek dengan *perlin-noise* serta pemanfaatan *Layer-Based* dan *Overlapping Detection*

1.5.5 Analisis kebutuhan

Tahap ini dilakukan untuk mengetahui hal apa saja yang diperlukan dalam penelitian, seperti faktor secara perangkat keras maupun perangkat lunak yang berpengaruh pada tahap implementasi dan pengujian

1.5.6 Implementasi

Mengimplementasikan algoritma *Perlin-Noise* dengan *Layer-Based* dan Overlapping detection pada *Unity Engine* dengan menggunakan bahasa

pemrograman C#. Tahap ini memanfaatkan hasil studi literatur, analisis masalah, dan rancangan.

1.5.7 Pengujian

Merupakan tahap pembuktian dengan melakukan pengujian seperti performa, akurasi dan waktu dari algoritma untuk generasi objek, dengan memanfaatkan tools *System.Diagnostics* sebagai *debugger*. Serta melakukan perbandingan dengan penelitian sebelumnya untuk meyakinkan adanya peningkatan atau pengurangan performa, akurasi, dan waktu dari hasil generasi objek terhadap target yang ingin dicapai.

1.5.8 Penarikan kesimpulan

Bagian ini merupakan hasil akhir dari proses analisis dan implementasi yang telah dilakukan. Dalam bagian kesimpulan ini, merangkum temuan utama yang diperoleh dari penelitian, selain itu, memberikan saran untuk penelitian lanjutan yang dapat dilakukan untuk memperdalam pemahaman dan pengembangan lebih lanjut dalam bidang ini.

1.6 Sistematika Penulisan

Sistematika penulisan laporan akhir penelitian ini disusun untuk memberikan

gambaran umum tentang penelitian yang dijalankan. Sistematika penulisan tugas akhir ini adalah sebagai berikut :

BAB I. PENDAHULUAN

Bab ini memberikan gambaran menyeluruh tentang latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, serta sistematika penulisan skripsi.

BAB II. LANDASAN TEORI

Bab ini mengulas berbagai konsep dasar dan teori-teori yang relevan dengan penelitian yang dilakukan, seperti dasar-dasar *Procedural Content Generation (PCG)* yang terfokus pada *Procedural Object Generation*, teori *Perlin-Noise*, serta konsep layer-based dan *overlapping* detection dalam pembuatan objek prosedural dalam *video game* menggunakan Unity Engine.

BAB III. ANALISIS DAN PERANCANGAN

Bab ini menjelaskan secara detail mengenai metode analisis dan perancangan yang digunakan dalam penelitian, termasuk langkah-langkah analisis kuantitatif. Metode yang digunakan meliputi penggunaan rumus algoritma *Perlin-Noise*, rumus pencarian algoritma untuk mendeteksi layer secara spesifik, mencari teori algoritma *Euclidean Distance* untuk menghindari *overlapping*, pengukuran waktu, membandingkan akurasi untuk mengurangi overlapping, serta menganalisis performa dengan penelitian sebelumnya untuk analisis kuantitatif.

BAB IV. IMPLEMENTASI DAN PENGUJIAN

Bab ini memaparkan implementasi dan hasil-hasil pengujian dari penelitian yang telah dilakukan, serta pembahasan terhadap waktu, akurasi, dan performa implementasi tersebut. Pembahasan ini meliputi interpretasi terhadap hasil, hubungannya dengan teori, serta implikasi praktis dari temuan-temuan penelitian.

BAB V. KESIMPULAN DAN SARAN

Pada bab ini, disajikan kesimpulan dari hasil penelitian yang telah dilakukan, termasuk implikasi dari temuan-temuan tersebut. Selain itu, diberikan juga saran untuk penelitian lanjutan yang dapat dilakukan berdasarkan hasil penelitian ini, seperti pengembangan algoritma, penambahan fitur, atau penelitian terkait aspek lain dari *procedural object generation* dalam *video game.*