BAB V
KESIMPULAN DAN SARAN

5.1. Penarikan Kesimpulan

Berdasarkan tahapan yang dilakukan mengenai implementasi algoritma *Perlin noise* untuk *procedural object generation* dengan *layer-based* dan *overlapping detection* dalam *video game* menggunakan *Unity Engine*, dapat disimpulkan bahwa:

1. Implementasi menggunakan *perlin noise, layer-based,* dan *euclidean distance* pada skenario *Single-Layer* menunjukkan penggunaan daya RAM yang hampir identik dengan *randomizer.* Pada *multi-layer*, terjadi sedikit peningkatan penggunaan RAM, yang disebabkan oleh pemrosesan tambahan yang diperlukan. Dalam hal penggunaan CPU, implementasi ini memang memakan lebih banyak daya dibandingkan *randomizer*, terutama pada generasi objek mulai dari 4096 ke atas. Hal ini disebabkan oleh kalkulasi tambahan dari *euclidean distance* dan *layer-based*, yang meskipun memakan lebih banyak sumber daya, mampu memberikan hasil yang lebih akurat.

2. Hasil akurasi dari penelitian menunjukkan keunggulan yang signifikan dibandingkan *randomizer.* Metode ini berhasil sepenuhnya mengatasi masalah *overlapping*, bahkan saat jumlah objek yang dihasilkan meningkat. Sebaliknya, pada metode randomizer, semakin banyak objek yang dihasilkan, semakin besar kemungkinan terjadinya overlapping, yang dapat mengurangi kualitas dan keakuratan generasi.

3. Perbedaan waktu generasi menjadi signifikan pada jumlah objek di atas 4096. Untuk jumlah objek di bawah 4096, perbedaan waktu antara ketiga metode pengujian ini tidak begitu mencolok. Namun, pada jumlah objek yang lebih tinggi, terutama dalam sistem multi-layer, waktu generasi

cenderung lebih lama. Ini merupakan timbal balik dari akurasi yang lebih tinggi dan pengurangan overlapping.

5.2. Saran Penelitian

Berdasarkan penelitian yang telah dilakukan, terdapat beberapa saran untuk implementasi lebih lanjut, yaitu:

1. Pada konteks *Procedural Object Generation*, diperlukan penelitian lebih lanjut dengan menggunakan metode alternatif dalam *overlapping detection*. Misalnya, metode seperti *Manhattan Distance, Chebyshev Distance, atau Minkowski Distance* serta dapat diuji untuk membandingkan efektivitasnya dalam mendeteksi *overlapping* pada objek dan membandingkannya dengan penelitian saat ini yang menggunakan *Euclidean-Distance*.

2. Mengatasi Limitasi *Layer-Based*, mengembangkan solusi untuk mengatasi keterbatasan jumlah layer pada *Layer-Based* atau *LayerMask()* yang saat ini terbatas pada 32 layer.